Université de Liège

Faculté des Sciences Appliquées

Institut Montefiore

Département d'Electricité, Electronique et Informatique

Decomposition, Localization and Time-Averaging Approaches in Large-Scale Power System Dynamic Simulation

Liège, Belgium, July 2012

A dissertation submitted by Davide Fabozzi
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy (Ph.D.) in Engineering Sciences

A mia mamma

The history of sparsity is an interesting example
of a development whose time was long overdue.
Why did it take so long to discover and catch on?
The answer might shed light on similar
occurrences of technological blindness.
Since the possibilities of sparsity exploitation
should have been self-evident to a village idiot,
it is perplexing how they escaped notice
by applied mathematicians. As it turned out,
sparsity was pioneered by a scattering of
problem solvers, not mathematicians.

This suggests that similar seemingly obvious
ideas may still be escaping detection,
not because of their obscurity
but because of their elusive simplicity [1]

[1]William F. Tinney. This excerpt is taken from the 1979 issue of *This Week's Citation Classics* [Tin79]: in this commentary, Dr. Tinney recalls the first efforts in exploiting the sparse structure of matrices, which can be traced back to 1957. Work on sparsity went unnoticed for about a decade and finally gained wider recognition only as late as 1968.

Examining Committee

Sì ch'io fui sesto tra cotanto senno [1]

Prof. Christophe Geuzaine, Université de Liège, Belgium

Prof. Jean-Louis Lilien (*President of Jury*), Université de Liège, Belgium

Prof. Federico Milano, Universidad de Castilla - La Mancha, Spain

Prof. Bikash C. Pal, Imperial College London, United Kingdom

Prof. Patricia Rousseaux, Université de Liège, Belgium

Prof. Thierry Van Cutsem (*PhD advisor*), Université de Liège, Belgium

Prof. Louis Wehenkel, Université de Liège, Belgium

Prof. Wolfram H. Wellßow, Technische Universität Kaiserslautern, Germany

[1]*And I was sixth amid so learn'd a band*. Dante, né Durante di Alighiero degli Alighieri, in *Comedìa*, later *Commedia*. English translation by H. F. Cary in *The Divine Comedy of Dante*. In this excerpt from *Canto IV* of *Inferno*, Dante is passing through *Limbo*, the first circle of hell where virtuous unbaptized adults are hosted (in Latin *limbus* means ”edge” or ”border”). Virgil, Dante's guide throughout this part of his journey, introduces him to fellow poets Homer, Horace, Ovid and Lucan: they invite Dante to join their discussion, during which Dante feels his wisdom ranks sixth (i.e. last) among those great poets.

Acknowledgements

Gratia pro rebus merito debetur inemtis [1]

I would like to thank my advisor, Prof. Thierry Van Cutsem, for giving me the opportunity to pursue my PhD under his guidance. The research group he leads at University of Liège is a unique working environment, where I felt free to develop my own creativity but at the same time I knew I could rely on his pragmatism and scientific rigor: his advice was fundamental to direct my research towards promising areas, and to avoid dead-ends.

I wish to express my gratitude to each member of the examining committee, for devoting their time to read this report. A special thanks to Prof. Patricia Rousseaux also for the many enriching discussions during my PhD.

The research documented in this report has been performed, for three years, in the context of the PEGASE project funded by European Commission's Seventh Framework Programme (grant agreement No. 211407), whose support I gratefully acknowledge.

Within this project I collaborated with the R&D department of the French transmission system operator RTE. I would like to thank Angela S. Chieh, Patrick Panciatici and Vincent Sermanson for welcoming me in my visits to Paris: I found extremely useful to discuss with partners who excel in both scientific research and transmission system operator practice.

[1]*Thanks worthily are due for things unbought*. Publius Ovidius Naso (Ovid) in *Amores*, English translation by Christopher Marlowe.

The Power System Consulting group of Tractebel Engineering - GDF Suez was the co-ordinator of the PEGASE project. I benefitted from the broad variety of skills of Tractebel experts. Many thanks to Dr. Bertrand Haut, Karim Karoui, Dr. Christian Merckx, Christel Moisse, Ludovic Platbrood and Fortunato Villella.

After my involvement in the PEGASE project I worked with the Canadian transmission system operator Hydro-Québec, TransÉnergie division. Thanks to Simon Lebeau and Francis Talbot I had access to real data for my tests and this collaboration allowed me to deal with real engineering problems.

Now that I am close to the end of my Belgian experience, I want to mention late Prof. Paolo Marannino, thanks to whom this experience began: he encouraged me to cross the Alps, in 2007, for the Erasmus exchange program that marked the start of my research at University of Liège. He is deeply missed.

I owe a special thanks to Dr. Mevludin Glavic for the many insightful discussions during our many coffee breaks outside the department entrance.

The experience in Liège was marked also by the enriching encounters of my senior PhD colleagues, Dr. Adamantios (Diamantis) Marinakis and Dr. Ninel (Bogdan) Otomega. They contributed to the friendly atmosphere of our research group, which I tried to pass over to my junior PhD colleagues, Petros Aristidou and Frédéric Plumier: I wish to thank them both for our close collaboration and friendship in the last years of my PhD thesis.

Three scholars helped me finding some of the original texts for the citations I used in the beginning of each chapter: many thanks to Prof. Laurence Bouquiaux at the Department of Philosophy of University of Liège, Prof. Gregory Brown at the Department of Philosophy of University of Houston, Texas, and Dr. Siegmund Probst at the Leibniz archive of Gottfried Wilhelm Leibniz Bibliothek, Niedersächsische Landesbibliothek, Hannover, Germany.

I cannot list them all here, but I heartily thank all my friends, the old friends from Italy as well as the people I befriended during these last 5 years in Belgium.

The last thanks go to my family, especially to my mother Giovanna, to whom this work is dedicated, and to Elena and Max thanks to whom I will soon become an uncle.

Grazie.

Davide

Abstract

Il sugo di tutta la storia [1]

Present-day interconnected electric power systems are the largest machines in the world. Guaranteeing a stable supply of electric power is vital for modern societies: power systems must be able to withstand plausible disturbances. A certain number of simulations of the post-disturbance behavior are routinely executed by some transmission system operators to assess that the system is operated in a secure way. Usually this assessment has to be performed within a predefined time frame, using the available computing resources. Improving the simulation speed allows the operators to perform a wider assessment, thus making better use of the available computational power. A large part of this research took place in the context of the PEGASE project, supported by European Commission (Seventh Framework Programme) and has resulted in some novel algorithms for faster dynamic simulations, one of the PEGASE project main goals. First, this thesis revisits the Newton method used to solve the differential-algebraic model. Then, three original algorithmic improvements are presented, namely (i) decomposition, (ii) localization and (iii) time-averaging of the system response. Finally, the combination of these approaches is shown to provide a fast and reliable tool for dynamic security assessment. All the presented techniques have been thoroughly tested on an academic system, a large real-life system and a realistic system of unprecedented size, representative of the whole continental European synchronous grid.

[1] *The gist* (literally ”the sauce”) *of the argument*. Alessandro Manzoni in *I Promessi Sposi* (*The Betrothed*).

List of main symbols

Non si può intendere [questo libro] *se prima non s'impara*
a intender la lingua, e conoscer i caratteri, ne' quali è scritto. [1]

Matrices are indicated with capital letters, vector and matrices are in bold font, vectors are column vectors unless otherwise stated.

N	number of network buses
$\mathbf{V}$	$2N$-dimensional vector of rectangular components of bus voltages
$\mathbf{z}$	vector of discrete variables
$\mathbf{x}$	vector of continuous variables
$\mathbf{C}$	matrix including 0's and 1's linking injector currents to network equations ($2N \times dim(\mathbf{x})$)
$\mathbf{B}$	matrix of sensitivities of injectors equations with respect to voltages ($dim(\mathbf{x}) \times 2N$)
$\mathbf{A}$	block diagonal matrix, each block being the Jacobian of injector equations with respect to injector variables ($dim(\mathbf{x}) \times dim(\mathbf{x})$)
$\mathbf{D}$	network matrix ($2N \times 2N$) stemming from decomposition of bus admittance matrix
$\mathbf{J}$	Jacobian of both network and injectors ($(2N + dim(\mathbf{x})) \times (2N + dim(\mathbf{x}))$)

[1] *We cannot understand* [this book] *if we do not first learn the language and grasp the symbols, in which it is written.* Galileo Galilei in *Il Saggiatore*. English translation by Thomas Salusbury in *The Assayer*, as quoted in *The Metaphysical Foundations of Modern Science* by E. A. Burtt.

n	number of injectors
n_i	number of continuous variables of injector i
$\mathbf{z}_i$	vector of discrete variables of injector i
$\mathbf{x}_i$	vector of continuous variables of injector i (n_i)
$\mathbf{C}_i$	matrix including $0'$s and $1'$s linking injector i currents to network equations ($2N \times n_i$)
$\mathbf{B}$	matrix of sensitivities of injectors equations with respect to voltages ($dim(\mathbf{x}) \times 2N$)
$\mathbf{B}_i$	matrix of sensitivities of injectors equations to voltages in injector i ($n_i \times 2N$)
$\mathbf{A}_i$	Jacobian matrix of injector equations with respect to injector variables in injector i ($n_i \times n_i$)
$\tilde{\mathbf{C}}_i\mathbf{B}_i$	Schur correction $\mathbf{C}_i\mathbf{A}_i^{-1}\mathbf{B}_i$ of injector i ($2N \times 2N$)
$\tilde{\mathbf{D}}$	Schur complement $\mathbf{D} + \sum_{i=1}^{n} \mathbf{C}_i\mathbf{A}_i^{-1}\mathbf{B}_i$ of $\mathbf{A}$ ($2N \times 2N$)

Solution schemes are indicated with capital letters.

I	Integrated scheme (see Chapter 3)
A	Decomposed scheme using acceleration techniques (see Chapter 4)
L	Localized scheme: same as A, using localization (see Chapter 5)
T	Time-averaged scheme: same as I, using time-averaging (see Chapter 6)
C	Combined scheme (see Chapter 7)

Table of Contents

Chapter 1

Introduction

Li arcieri prudenti pongono la mira
assai più alta che il loco destinato [1]

1.1 Motivation

Dynamic simulations under the phasor approximation are routinely used throughout the world for the purpose of checking the response of electric power systems to large disturbances. Over the last decades, dynamic models have been standardized and simulation tools have been developed to meet the specifics of power system models [Kun94].

However, in spite of the increase in computational power, dynamic simulations of large-scale systems remain time consuming. Indeed, they require solving a large set of nonlinear stiff hybrid Differential-Algebraic Equations (DAEs). A large interconnected system may involve hundreds of thousands of such equations spanning very different time scales and undergoing many discrete transitions imposed by limiters, switching devices, etc.

There are applications where this computational effort is an obstacle. Let us quote non exhaustively: (i) detailed operator training simulator; (ii) analysis of large set of scenari, for instance in Monte-Carlo simulations; (iii) Dynamic Security Assessment (DSA), particularly in real-time applications.

[1] *The clever archers take aim much higher than the mark.* Niccolò di Bernardo dei Machiavelli, in *Il Principe*, English translation by W. K. Marriott in *The Prince*.

The last item is probably the most constraining [gC07]. This is even more true when, instead of the 10 to 20 seconds simulated for short-term stability, the evolution is computed until a new steady state is reached, which requires simulating long-term dynamics over several minutes after the initiating event. This computational burden justifies the fact that many control centers routinely perform static security assessment, typically AC (if not DC for some problems) power flows and resort infrequently to DSA. Static computations seek the post-contingency equilibrium point, without considering the system evolution that may - or may not - lead to that point.

One can expect, however, that the need for DSA will go increasing. Operation of non-expandable grids closer to their stability limits and unplanned generation patterns owing to the penetration of renewable energy sources require dynamic studies. Furthermore, under the pressure of electricity markets and with the support of active demand response, it is likely that system security will be more and more guaranteed by emergency controls responding to the disturbance. In this context, security analysis requires checking the sequence of events that take place after the initiating disturbance, a task for which static calculation of the operating point in a guessed final configuration is inappropriate. Last, in stressed conditions, power flow computations may diverge, thus leaving operators with no information in circumstances where the support of computer simulation is mostly needed.

1.2 A classification of existing approaches

Several approaches have been tried over the past decades to speed-up time simulation. They can be roughly classified into *relaxation* and *direct* methods [PCLG^{+}11].

Approaches by relaxation rely on a decomposition of the set of equations in sub-domains in which the equations are solved independently. Information, such as states or time evolution at the boundaries, is then exchanged between sub-domains before a new iteration is carried out [MR06]. The advantage of these methods lies in the ability to solve the sub-domains in parallel using multiple processors/cores [CZBT91] or even graphics processing units [JMD10], while the main drawback is the convergence degradation due to the above mentioned iterations. Examples of relaxation methods that have been applied to power systems are: waveform relaxation on a time interval [ISCP87] or at one time step [JMD09], multi-rate simulation [CC94], and Picard iterations [LSBTC90].

Among the direct methods, the Newton method is prominent. It is used to solve the nonlinear equations stemming from the algebraic equations of the model and from the algebraization of the differential equations at a given time step. A major advantage of this method lies in its convergence properties. In its standard form, it requires solving a large system of linear equations based on a sparse Jacobian matrix [KMC08]. For power system dynamic models, a salient feature of this matrix is its Bordered Block Diagonal (BBD) structure. It allows solving the linear equations in a decomposed way, the numerous small diagonal blocks and their Schur complement [Zha05] being processed separately. This also opens the way for some parallel processing, while retaining the good convergence property of the direct Newton method. The BBD structure is exploited in VLSI circuit simulation for instance [Ogr94], and its application to power system dynamics can be traced back to [Boe77]. Since then, the technique has been used in some production grade simulation packages (e.g. [Kun94] p. 863) although, to the author's knowledge, there are few references on the subject.

1.3 Outline of this work

In this work the BBD structure is exploited in a novel way that is still suitable to some parallel processing [HBJVN77], although it is not the topic of this thesis. The proposed algorithm resorts to the above mentioned decomposition: (i) to solve only the parts of the system that need to be solved, and (ii) to avoid global updates of the Jacobian when local updates would suffice. Both situations occur very often in large-scale power system simulation.

The above cited methods speed up the Newton method while retaining full accuracy. An additional possibility consists in speeding-up numerical simulations by relaxing accuracy to an extent compatible with DSA needs. Two tracks were followed in this work, leading to relaxation of accuracy in space and in time, respectively.

Previous attempts to speed up dynamic simulation often relied on model reduction or model simplification. Models are simplified in order to become smaller and/or less stiff. An example that combines reduction in size and stiffness is the quasi steady-state approximation of long-term dynamics [VCV08]. It consists in replacing the short-term dynamics by their equilibrium equations while the simulation focuses on the long-term dynamics. However, automatic methods to simplify real-life, hybrid power system models are yet to be developed. Hence, the current practice consists in either creating and maintaining a reduced model for a specific system, or adopting right away a generic simplified model. The concept of accuracy relaxation considered in thesis is definitely different from model simplification, in so far as the same model as in detailed simulations is adopted. Thus, no effort is spent in model reduction; instead the gain in efficiency comes from the relaxable accuracy solver that is used.

Spatial relaxation of accuracy exploits the fact that, in a large-scale system, most of disturbances give rise to localized effects, i.e. some system components do not participate much in the dynamic response. This holds true at least for some periods of time and is more noticeable in long-term simulations. *Localization* concepts have been used in time-domain simulation in the variable partitioning multi-rate method [CC08]. Adaptations of this concept have been applied also to static security analysis [Bra93], mainly in extensions of the sparse vector method [TBC85] known as zero mismatch approach [BT89], approximate sparse vector method [BET91] and adaptive localization [EPT92].

Similar techniques have been used successfully in the simulation of VLSI circuits [HSV81]. In VLSI simulation, the system components which are nearly unaffected by a disturbance are called *latent*, and the objective is to skip useless numerical operations relative to a hopefully large proportion of latent blocks. To this purpose, the network is torn into one main circuit and many other blocks in order to obtain a block-bordered diagonal structure favorable to latency exploitation. The already converged sub-networks are not solved.

With temporal relaxation of accuracy we refer to the fact that some fast components of the response may not be of interest, especially in long-term simulations. In nonlinear dynamical systems theory, the procedure of replacing a vector field by its average (over time or an angular variable) in order to obtain asymptotic approximations to the original system is called *averaging* [SVM07]. A less rigorous but straightforward way to obtain averaged responses is by using a step size which is large with respect to the neglected dynamics. Once more, no simplification is performed on the models, i.e. they are the same as in detailed simulation. Significant reductions of the computational effort can be expected from such large steps.

It is known that an L-stable method must be used to integrate with steps significantly larger than the smallest system time constants [AP98, CK06]. For larger and larger step sizes, the trajectory converges towards a solution that satisfies the equilibrium equations of the neglected dynamics. Those methods have been used to simulate quasi-sinusoidal power system models [FVC09, WC11] as well as other engineering problems (e.g. [BW98]).

Among them, the Backward Euler method was proposed to suppress numerical oscillations in electromagnetic transient simulations [ML89]. On the other hand, the price to pay with L-stable methods is that they overdamp oscillations whose period is comparable to the step size [AP98].

For a large class of continuous-time dynamics, in so far as accuracy can be somewhat relaxed, the step size of an L-stable method is limited only by the convergence of the Newton iterations used to solve the algebraized equations [FVC09]. However, power system models are also hybrid and the proper handling of discrete events requires some caution when the time step is increased. To the author's knowledge, this aspect has been relatively less investigated in the literature; our preliminary investigations were reported in [FCPVC11].

When a power system is subject to a large disturbance (typically a fault), its dynamic response can be decomposed into a short-term and a long-term period, the former requiring a more accurate solution than the latter [KOO$^+$93].

In order to speed up dynamic simulation, with our main focus being on the long-term, we advocate the use of a combination of detailed dynamic simulation in the short-term with relaxation of accuracy in the long-term by means of localization and time-averaging. The advantage of the proposed method lies in the seamless integration between detailed short-term simulation and relaxed-accuracy long-term simulation.

1.4 Structure of the report

This report is organized as follows. In Chapter 2, power system modelling is revisited, along with the description of the three test systems used throughout this thesis. Chapter 3 is devoted to solution methods. Chapter 4 focuses on speeding up the Newton scheme by infrequently updating Jacobian sub-matrices and skipping iterations on converged components, without degradation of accuracy. Chapters 5 and 6 deal with the relaxation of accuracy in space and in time, respectively, in order to obtain faster simulations. In Chapter 7, novel ways of combining accurate simulation with localization and time-averaging are discussed. Also, this final chapter provides the author's recommendations on how to achieve faster dynamic simulations by accepting a fair compromise between efficiency and accuracy. Simulation results obtained with the application of each proposed technique are presented in the corresponding chapter.

Appendices close this thesis. They deal with the choice of angle and speed references in A.1, the adopted synchronous machine and load models (A.2 and A.3, respectively), the computation of the coefficients of the numerical integration method (A.4) and technical information on how the simulation results were obtained (A.5).

1.5 Publications

This thesis expands material which has been published, or is under revision, in journals and conferences:

[FCHVC12] D. Fabozzi, A. S. Chieh, B. Haut, and T. Van Cutsem. Fast dynamic simulations of large power systems. Part I: Accelerated and localized Newton scheme and Part II: Combined detailed-simplified simulation. *Submitted and now under revision for publication in IEEE Trans. Power Syst.*, 2012.

[FVC11a] D. Fabozzi and T. Van Cutsem. Assessing the proximity of time evolutions through dynamic time warping. *IET Generation, Transmission & Distribution*, 5(12):1268–1276,Dec. 2011.

[FCPVC11] D. Fabozzi, A. S. Chieh, P. Panciatici, and T. Van Cutsem. On simplified handling of state events in time-domain simulation. In *Proc. 17th Power System Computation Conference (PSCC)*, pages 1–9, Aug. 2011.

[FVC11b] D. Fabozzi and T. Van Cutsem. On angle references in long-term time-domain simulations. *IEEE Trans. Power Syst.*, 26(1):483–484, Feb. 2011.

[FVC10] D. Fabozzi and T. Van Cutsem. Localization and latency concepts applied to time simulation of large power systems. In *Proc. 2010 Bulk Power System Dynamics and Control - VIII IREP Symposium*, pages 1–14, Aug. 2010.

[FVC09] D. Fabozzi and T. Van Cutsem. Simplified time-domain simulation of detailed long-term dynamic models. In *Proc. 2009 IEEE PES General Meeting*, pages 1–8, July 2009.

CHAPTER 2

Power system modelling

Il semble que la perfection soit atteinte
non quand il n'y a plus rien à ajouter,
mais quand il n'y a plus rien à retrancher [1]

In this chapter, the models used for short and long term stability calculations are described. Some standard models of both the network and the dynamical parts of the system, together with the choice of reference axes are analysed and revisited. It will be discussed how the chosen non-causal equation-based hybrid modelling can encompass the standard model used in stability studies, but provides more flexibility with respect to the latter, especially regarding the modelling of the algebraic and discrete parts.

Examples are given on two academic cases, that despite their simplicity are representative of the modelling issues that can be met in more complex, realistic power system models, such as the synchronous machine and restorative load models reported in Appendix A.2 and A.3, respectively. The description of the test systems used throughout this thesis closes this chapter: the three test systems chosen offer a wide variety since they include a small academic system (Nordic32), one large real system (Hydro-Québec), and one very large realistic system (PEGASE).

[1] *A designer knows he has achieved perfection not when there is nothing left to add, but when there is nothing left to take away.* Antoine de Saint-Exupéry, in *Terre des Hommes*, English translation by L. Galantière in *Wind, Sand and Stars*.

2.1 Differential-algebraic models

The Differential-Algebraic (DA) models of a power system used in stability studies can be written as:

$$\begin{aligned}
\mathbf{0} &= \mathbf{D}\,\mathbf{V} - \mathbf{I} && (2.1)\\
\mathbf{0} &= \boldsymbol{\psi}_n(\mathbf{y}, \mathbf{w}, \mathbf{z}, \mathbf{V}, \mathbf{I}, \omega_{ref}) && (2.2)\\
\mathbf{0} &= \boldsymbol{\psi}_c(\mathbf{y}, \mathbf{w}, \mathbf{z}, \mathbf{V}, \mathbf{I}, \omega_{ref}) && (2.3)\\
\dot{\mathbf{w}} &= \boldsymbol{\psi}_d(\mathbf{y}, \mathbf{w}, \mathbf{z}, \mathbf{V}, \mathbf{I}, \omega_{ref}) && (2.4)\\
\mathbf{z}(t_j^+) &= \boldsymbol{\psi}_z(\mathbf{y}, \mathbf{w}, \mathbf{z}(t_j^-), \mathbf{V}, \mathbf{I}, \omega_{ref}) && (2.5)
\end{aligned}$$

The equations above make up a system of stiff and hybrid differential-algebraic equations.

The definition of stiffness is controversial, as it relates both to the numerical difficulties encountered in the solution of a stiff systems with some specific solvers and to the coexistence of fast and slow components in system dynamics [AP98, CK06]. Further discussion on the former definition can be found in Chapter 3, while Chapters 6 and 7 are devoted to the latter.

Under the quasi-sinusoidal approximation, all current and voltage phasors are projected onto orthogonal reference axes, denoted by x and y respectively, rotating at the angular speed ω_{ref} (the choice of the latter is discussed later in this section). Thus the vector of complex currents is decomposed into its $\mathbf{i}_x$ and $\mathbf{i}_y$ components, and the vector of complex bus voltages similarly into $\mathbf{v}_x$ and $\mathbf{v}_y$. The network equations take on the form:

$$\mathbf{i}_x + j\mathbf{i}_y = (\mathbf{G} + j\mathbf{B})(\mathbf{v}_x + j\mathbf{v}_y) \tag{2.6}$$

where $\mathbf{G}$ and $\mathbf{B}$ are respectively the conductance and susceptance matrices, i.e. the real and imaginary parts of the bus admittance matrix. Decomposing and re-assembling (2.6) yields:

$$\begin{bmatrix} \mathbf{G} & -\mathbf{B} \\ \mathbf{B} & \mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{v}_x \\ \mathbf{v}_y \end{bmatrix} - \begin{bmatrix} \mathbf{i}_x \\ \mathbf{i}_y \end{bmatrix} = \mathbf{0} \tag{2.7}$$

By reordering the voltage components with the V_{xi} variable relative to bus i occupying the position $2i-1$ and V_{yi} the position $2i$, and similarly for the current components, (2.7) can be rewritten as in (2.1), where $\mathbf{D}$ is a sparse, structurally symmetric matrix and $\mathbf{V}$ (resp. $\mathbf{I}$) collects all voltage (resp. current) components.

Equation (2.2) connects the network to the various components: for instance, in case of a synchronous machine, those are the stator equations. Other algebraic equations, coming for instance from controllers or physical laws, are included in (2.3). Vector $\mathbf{y}$ is made up of the corresponding algebraic variables.

Vector $\mathbf{w}$ groups the differential variables of all devices whose dynamics are described by (2.4), a set of Ordinary Differential Equations (ODEs).

Note that the DA model of a power system is often written in more compact form (e.g. [Kun94]). Assuming that (2.2) and (2.3) can be transformed to express $\mathbf{y}$ and $\mathbf{I}$ as functions of the other variables, those expressions are substituted in (2.1) and (2.4) to obtain a DA model that has only $\mathbf{V}$ as algebraic states:

$$\mathbf{0} = \mathbf{m}_a(\mathbf{w}, \mathbf{V}) \tag{2.8}$$

$$\dot{\mathbf{w}} = \mathbf{m}_d(\mathbf{w}, \mathbf{z}, \mathbf{V}) \tag{2.9}$$

$$\mathbf{z}(t_j^+) = \mathbf{m}_z(\mathbf{w}, \mathbf{z}(t_j^-), \mathbf{V}) \tag{2.10}$$

This transformation of (2.2) and (2.3), however, may be impractical, if not impossible for some models, e.g. the synchronous machine model which includes saturation effects. The above mentioned elimination of $\mathbf{y}$ and $\mathbf{I}$ is a typical step of input-output oriented modelling, also referred to as *causal* in the dynamic simulation literature [CK06] as opposed to *non-causal* models. The advantages of non-causal modelling are further discussed in Section 2.2.

Finally, equation (2.5) takes into account the discrete dynamics of the system such as responses of discrete controls or changes of flows in continuous-time equations, an important feature of power system dynamic simulation [HP00]. The respective discrete variables are stored in $\mathbf{z}$.

Continuous-time states $\mathbf{w}$ evolve over intervals of time. The algebraic variables in $\mathbf{V}$, $\mathbf{I}$ and $\mathbf{y}$ can also vary over instants, when some $\mathbf{z}$ components change. The discrete states $\mathbf{z}$ evolve over instants. With the notation of (2.5), $\mathbf{z}$ changes from $\mathbf{z}(t_k^-)$ to $\mathbf{z}(t_k^+)$ at time instants t_k dictated by *events*, as it will be explained in detail in Section 2.3.

The choice of the speed ω_{ref} at which the xy reference axes rotate is arbitrary and can be made for convenience. Standard practices are either to choose the nominal angular frequency $\omega_{ref} = 2\pi f_o$, where f_o is the nominal system frequency, either to adopt the Center-Of-Inertia (COI) reference frame initially proposed in [TS72]. A recent contribution [Mil12] reports on extraneous instabilities which can possibly arise in systems with non-synchronous distributed energy resources due to the reference frame used. For different reasons, that will be analyzed in detail in Appendix A.1, we preferred resorting to the "explicit" COI reference approach that we proposed in [FVC11b]. This approach consists of using, in Eqs. (2.2-2.5), at each instant the value of ω_{ref} at the previous time step, which can be computed explicitly; this allows eliminating ω_{ref} from the set of variables.

2.2 Non-causal equation-based models in compact form

In the previous section we introduced the difference between causal and non-causal models equations. An equation is causal [CK06] when the equal sign is interpreted as an assignment more than an equality: this is typical of block-diagram modelling. While on one hand block-diagram modelling represents a major advance by allowing a graphical representation of a model, on the other hand, it forces to assign the output as a function of the input [Lar04]. Moreover, the interconnection of different subsystems may require to change the model description whenever this brings a change in causality.

By way of illustration, let us take the example of a resistor described by Ohm's law: $V = R\ I$.

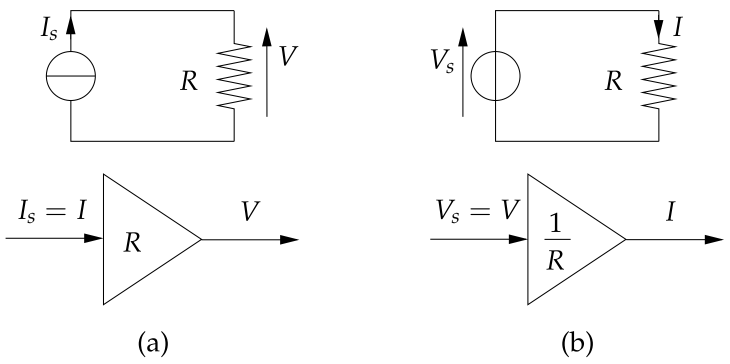


Figure 2.1: Change in causality in a block-diagram modelled resistor

If the resistor subsystem is connected to another subsystem, for instance a current source, then the natural choice is to have I as the input and V as the output, as Fig. 2.1.a shows. In Fig. 2.1.b, in case the resistor is connected to a voltage source, it is the opposite. Thus the system description is valid only for a given causality.

Moreover, block-diagram description brings the additional problem of finding the causality of a given DA model, which in some cases, known as "algebraic loops", is impossible [CK06]. In fact, by resorting to the incidence matrix of the blocks, it can be shown that if the corresponding incidence matrix cannot be put in a lower triangular form a causality cannot be established. A relevant example is shown in Appendix A.2 with the description of the adopted synchronous machine model including saturation, which indeed includes an algebraic loop.

An equation is non-causal when it expresses a relation between peers more than a hierarchy of signals: this is typical of physical modelling. The physical modelling approach expresses constraints on variables which have to be obeyed regardless of any causality: this approach naturally leads to equation-based models which can then easily be interconnected in order to form more complex systems [PC11].

The advantages of equation-based over block diagram modelling will be further discussed in Section 2.3 in the context of hybrid systems and in Appendix A.2 with the aforementioned algebraic loop problem.

Without loss of generality, the model of Eqs. (2.1-2.5) can be rewritten in a non-causal equation-based compact form as:

$$\mathbf{0} = \mathbf{D}\,\mathbf{V} - \mathbf{C}\,\mathbf{x} = \mathbf{g}(\mathbf{x},\mathbf{V}) \tag{2.11}$$

$$\mathbf{\Gamma}(\mathbf{z})\,\dot{\mathbf{x}} = \boldsymbol{\phi}(\mathbf{x},\mathbf{z},\mathbf{V}) \tag{2.12}$$

$$\mathbf{z}(t_j^+) = \mathbf{h}(\mathbf{x},\mathbf{z}(t_j^-),\mathbf{V}) \tag{2.13}$$

in which the expanded state vector $\mathbf{x}$ contains the current components $\mathbf{i}_x$ and $\mathbf{i}_y$, the algebraic variables $\mathbf{y}$ and the differential states $\mathbf{w}$. Equation (2.11) is essentially the same as (2.1) and will be used either in matrix form or in $\mathbf{g}$ function form, for writing convenience. $\mathbf{C}$ is a matrix with zero's and one's whose purpose is to extract the current components from $\mathbf{x}$. Equation (2.12) encompasses both the algebraic equations (2.2, 2.3) and the differential equations (2.4). $\mathbf{\Gamma}$ is a diagonal matrix with $(\mathbf{\Gamma})_{\ell\ell} = 0$ if the ℓ-th equation (2.12) is algebraic, and $(\mathbf{\Gamma})_{\ell\ell} = 1$ if the ℓ-th equation (2.12) is differential. Equation (2.13) is unchanged with respect to (2.5). The dependence of $\mathbf{\Gamma}(\mathbf{z})$ with $\mathbf{z}$ is clarified in the next section.

The continuous-time dynamics equations (2.11-2.12) deal with the network and various phenomena and controls, ranging from the short-term dynamics of power plants, static var compensators, induction motors, etc., to the long-term dynamics of secondary frequency control, load self restoration, etc. These equations can be either algebraic or differential. Vector $\mathbf{x}$ includes the corresponding (differential and algebraic) state variables, such as rectangular components of bus voltages, rotor angles, flux linkages, motor slips, controller state variables, etc. The components of $\mathbf{x}$ take values in $\mathbb{R}$ or in an interval of $\mathbb{R}$.

2.3 Non-causal equation-based hybrid models

Power systems are also hybrid systems, i.e. they involve variables which change at specific instants of time [HP00].

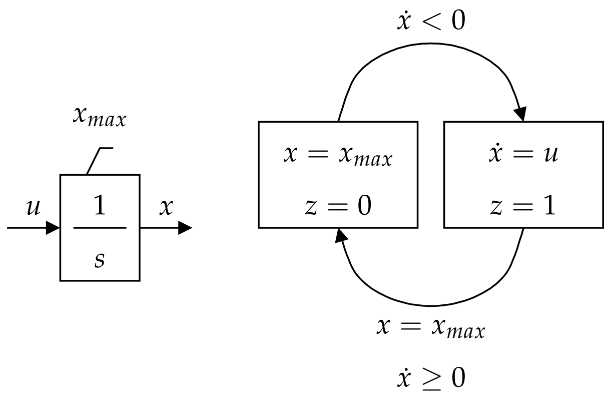
The discrete-time equations (2.13) capture:

1. the response of discrete controls and protections acting with various delays in automatic shunt compensation, secondary frequency and voltage control, load tap changers, phase shifting transformers, overexcitation limiters, etc. The corresponding components of $\mathbf{z}$ are shunt susceptances, generator setpoints, transformer ratios, etc.

2. the change in continuous-time equations caused by limits, switches, minimum gates, hysteresis, etc. The corresponding variables $\mathbf{z}$ act as "switches". Note that the diagonal entries of $\mathbf{\Gamma}$ may vary with $\mathbf{z}$, as suggested by (2.12), since a change in z may cause an equation to change from differential to algebraic and vice versa.

The components of $\mathbf{z}$ take values in a countable subset of $\mathbb{R}$.

A classical example of the second category above is the integrator with non-windup limit, shown in Fig. 2.2.a.



(a) (b)

Figure 2.2: Integrator with non-windup limit

Let us define z as a discrete variable with value 1 if $x < x_{max}$ and value 0 if $x = x_{max}$. This system is described by the following continuous-time equation of type (2.12):

$$z\,\dot{x} = z\,u + (1 - z)\,(x - x_{max})$$

Thus, $\Gamma(z) = z$. The changes in z can be described by the *state transition diagram* of Fig. 2.2.b [Lyg04, CK06].

Continuous-time states $\mathbf{x}$ evolve over intervals of time. The algebraic states in $\mathbf{x}$ and $\mathbf{V}$ can also vary over instants, when some $\mathbf{z}$ components change. The discrete states $\mathbf{z}$ evolve over discrete instants. With the notation of Eq. (2.13), $\mathbf{z}$ changes from $\mathbf{z}(t_k^-)$ to $\mathbf{z}(t_k^+)$ at time instants t_k dictated by *events*. In the example of Fig. 2.2, the events are the transitions from $z = 0$ to $z = 1$ and vice versa.

The dynamic behavior over a continuous interval according to Eqs. (2.11-2.12) is referred to as *flow*. The variation of some components of $\mathbf{z}$ according to Eq. (2.13) causes the passage from one flow to another at one point in time. This passage is referred to as a *jump*.

Events are classified according to whether it is possible or not to forecast when they take place [CK06]. In the former case, they are called *time events*, and in the latter case, *state events*. State events take place when a *jump condition* is satisfied, as a result of system evolution. It is usually possible to specify the event condition as a zero-crossing function, i.e. the event takes place when the value of that function passes through zero. A general solver is thus in charge of monitoring the zero-crossing functions in order to identify the *event times*, as will be detailed in Section 3.5.

There are interactions between hybrid systems and block-diagram modelling that reinforce the advantages of equation-based modelling: in fact, the occurrence of an event might require to change the model description whenever this brings a change in causality. As an example, let us add a diode in the loop of the simple circuit considered in Section 2.2. The resulting circuit is depicted in Fig. 2.3.

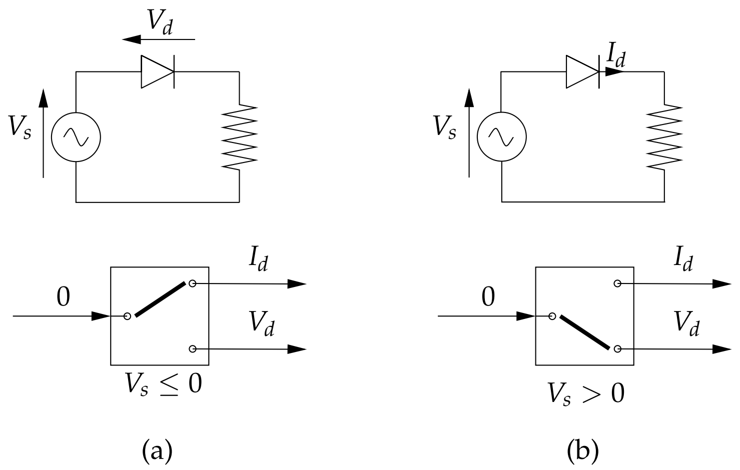


Figure 2.3: Change in causality in a block-diagram modelled ideal diode

When the diode is blocking, i.e. V_d and V_s are non positive, the diode equation is $I_d = 0$, as Fig. 2.3.a shows. Otherwise, as shown in Fig. 2.3.b, when the diode is conducting, its equation is $V_d = 0$. This implies a change in causality for the diode but also for the resistor: in fact, in the former case, the resistor current is given ($I = I_d = 0$) and the voltage is computed as in Fig. 2.1.a, while in the latter case the resistor voltage is given ($V = V_s - V_d = V_s$) and the current is computed from the block diagram shown in Fig. 2.1.b.

The hybrid system example above, together with its continuous part considered in Section 2.2, shows that it is very likely that changes in causality happen when dealing with connections of even very simple systems, both continuous and hybrid. This consideration, together with the algebraic loop problem mentioned in Section 2.2 which is likely to be present in realistic power system models (as shown in Appendix A.2), motivates the use of non-causal equation-based hybrid modelling. Another example of this model is given in Appendix A.3 with the description of the restorative load model adopted in our simulations.

2.4 Description of Nordic32 system

The proposed system is a variant of the so-called Nordic32 test system, proposed by K. Walve[1] and detailed in [Stu95]. As indicated in this reference, the system is fictitious but similar to the Swedish and Nordic system (at the time of setting up this test system). The system includes 52 (generator and transmission) buses and 80 branches. When including the distribution buses and transformers, there are a total of 74 buses and 102 branches, respectively. Appendices A.2 and A.3 describe the synchronous machine and the load model, respectively. The synchronous machines are equipped with generic excitation systems, voltage regulator, power system stabilizers, speed governors and turbine models.

The one-line diagram is shown in Fig. 2.4. This system consists of four areas:

- "North" with hydro generation and some load
- "Central" with much load and thermal power generation
- "Equiv" connected to the "North", it includes a very simple equivalent of an external system
- "South" with thermal generation, rather loosely connected to the rest of the system.

[1]at that time with Svenska Kraftnät, the Swedish transmission system operator.

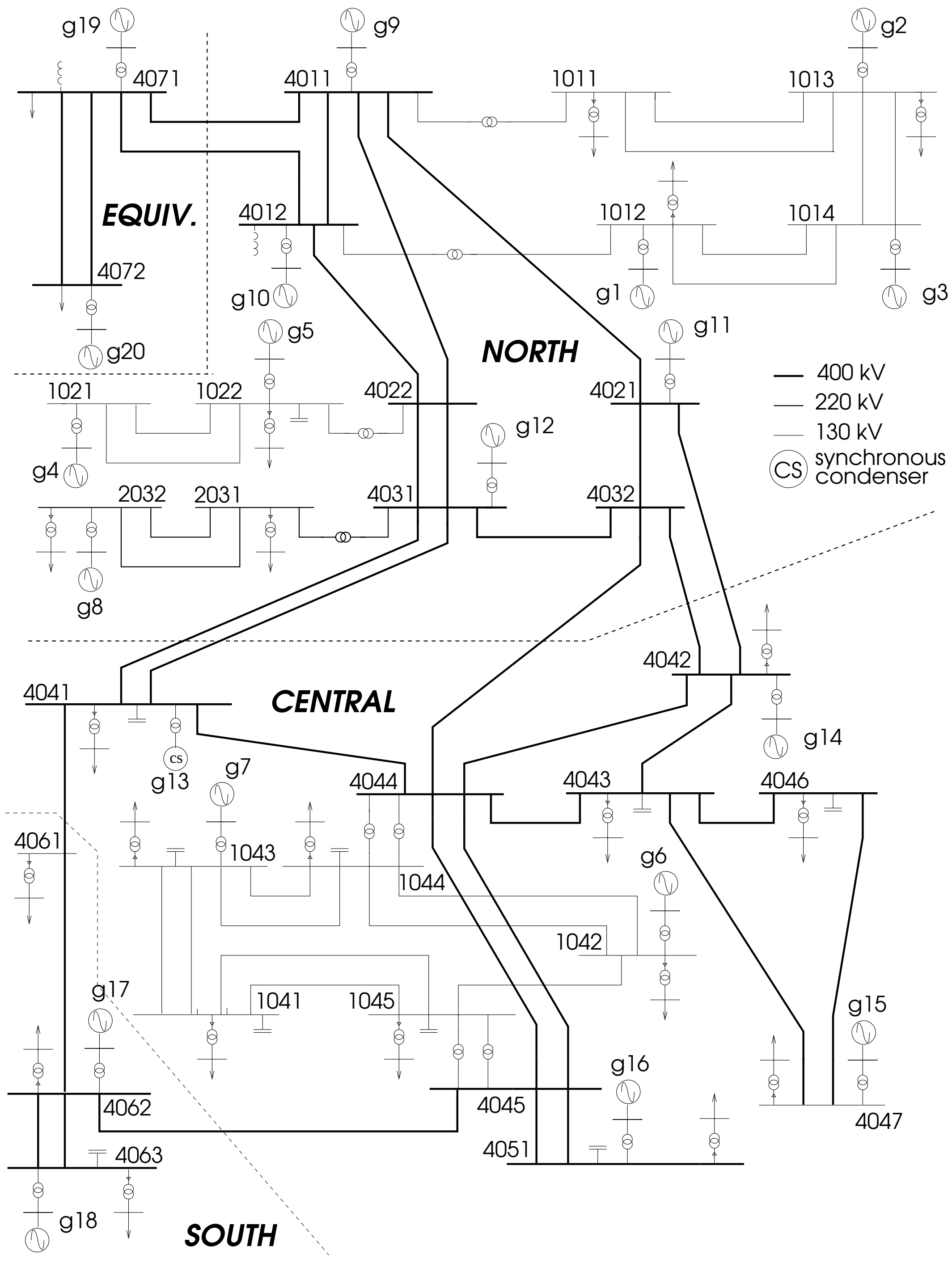


Figure 2.4: The Nordic32 system

The system has rather long transmission lines of 400-kV nominal voltage. Figure 2.4 shows the structure of the 400-kV backbone, along with the representation of some regional system operating at 220 and 130 kV.

The nominal frequency is 50 Hz. Frequency is controlled through the speed governors of the hydro generators in the North and Equiv areas only. g20 is an equivalent generator, with a large participation in primary frequency control. The thermal units of the Central and South areas do not participate in this control.

The system is heavily loaded with large transfers essentially from North to Central areas. Secure system operation is limited by transient (angle) and long-term voltage instability. The contingencies likely to yield voltage instability are:

- the tripping of a line in the North-Central corridor, forcing the North-Central power to flow over the remaining lines;
- the outage of a generator located in the Central region, compensated (through speed governors) by the Northern hydro generators, thereby causing an additional power transfer over the North-Central corridor.

The maximum power that can be delivered to the Central loads is strongly influenced by the reactive power capabilities of the Central and some of the Northern generators. Their reactive power limits are enforced by OverExcitation Limiters (OELs). On the other hand, Load Tap Changers (LTCs) aim at restoring distribution voltages and hence load powers. If, after a disturbance (such as a generator or a line outage), the maximum power that can be delivered by the combined generation and transmission system is smaller than what the LTCs attempt to restore, voltage instability results. The latter is of the long-term type. It is driven by OELs and LTCs and takes place in one to two minutes after the initiating event. A similar mechanism takes place in case of a demand increase.

The Nordic32 system is used to provide examples of the dynamic phenomena of interest, but its computational effort measures will not be given because they are not significant due to the small size of the system.

2.5 Description of Hydro-Québec system

Hydro-Québec is the transmission system operator of the Québec province, Canada. The Hydro-Québec system model used has been provided to us by the TransÉnergie division in order to investigate the performance of the proposed solver on a real-life system, which due to its characteristics requires extensive use of dynamic simulation.

A system map is shown in Fig. 2.5.

The model includes 2204 buses and 2919 branches, whose nominal voltage ranges from 6 to 735 kV. 135 synchronous machines are represented in detail together with their excitation systems, voltage regulators, power system stabilizers, speed governors and turbines. 976 restorative loads of the type described in Appendix A.3 are also present. The resulting system has 11774 differential-algebraic equations and 2695 discrete variables. The peak load is 35 000 MW.

Reference [TBS99] provides a brief overview of the system, which is hereafter transcribed:

> Geographic constraints have played a decisive role in the development of the Hydro-Québec system. The fact that most of the generating facilities are large hydroelectric power stations located more than 1,000 km from the main centres of consumption led Hydro-Québec to design a very extensive, extra high voltage 735-kV transmission system. The Hydro-Québec system has a number of characteristics that make stability and voltage control important. Among the most important of these characteristics are the following:
>
> - the isolation of the Hydro-Québec system (no synchronous link with neighboring systems);
> - the great distances between generation and load and the concentration of generation at large hydroelectric sites (85% of total generation takes place in 3 large, faraway hydroelectric complexes);
> - the use of a 735-kV transmission system that is very extensive (more than 11,000 km of 735-kV lines) but has a relatively limited number of lines located in two main axes.
>
> Because of its very extensive structure and the relatively small number of lines in its main transmission system (in comparison with the power carried), the system is sensitive to incidents that could cause chain reactions and the loss of several transmission lines. In fact, Hydro-Québec experienced three general power failures in the 1980s caused by extreme contingencies.

To prevent such possible failures, many corrective controls have been implemented. Among these, a prominent role is played by 22 automatic shunt reactor switching devices - named MAIS [BTS96]. These devices are in operation in 735-kV substations and control a large part of the total 25 500 Mvar shunt compensation. Along with the afore-mentioned 22 MAIS, 1077 LTCs control the voltages at intermediate and distribution levels. The system is also equipped with 14 static var compensators, some of them with stabilizing signal.

The Hydro-Québec system has been used in this work to assess the capability of reproducing important dynamic phenomena as well as to analyze the computational effort measured in seconds of cpu time. The system mostly benefits from decomposition and time-averaging, while the gain brought by localization is only marginal, due to the medium

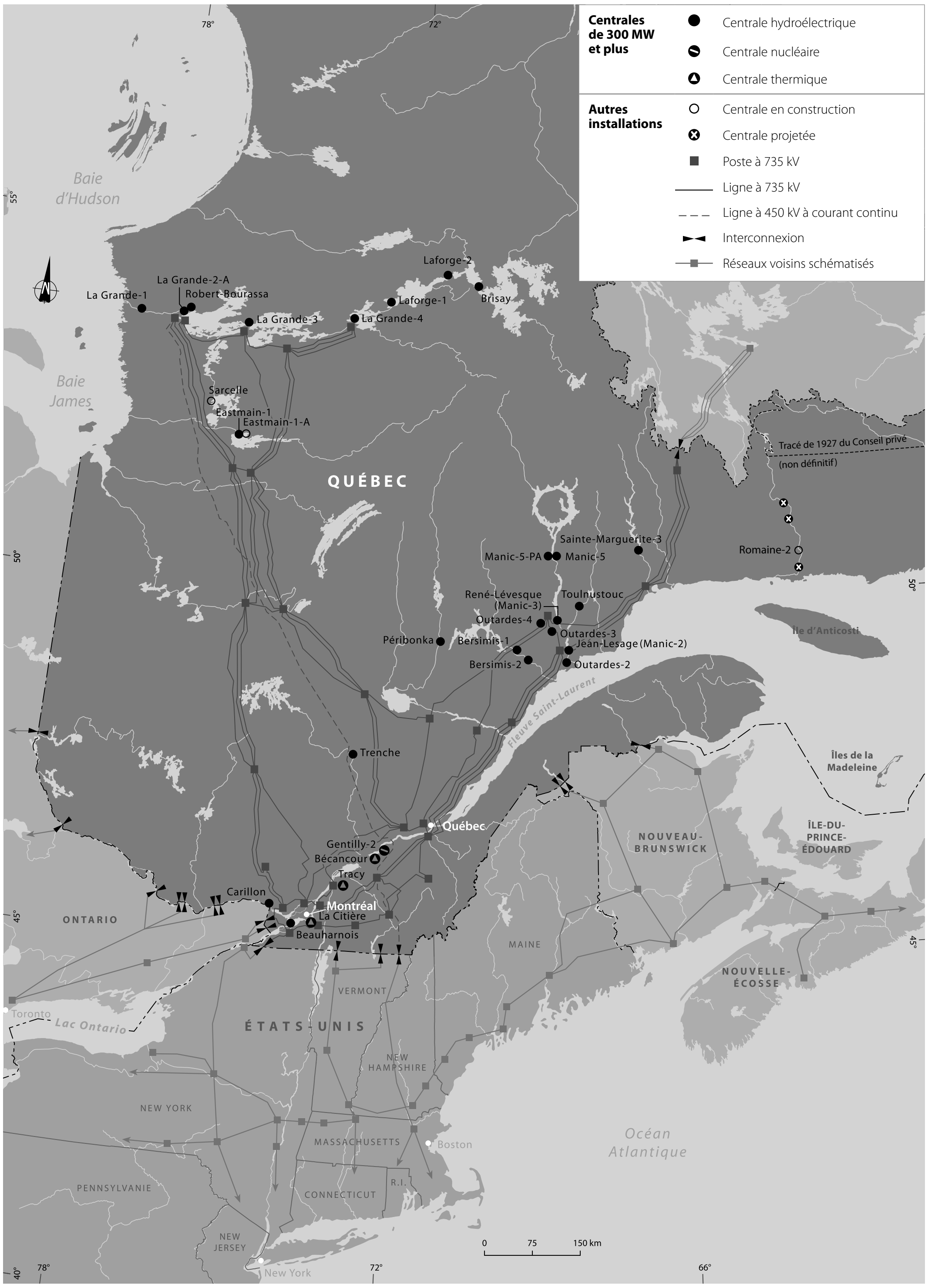


Figure 2.5: Map of the Hydro-Québec system (as of 2010)

size of the system, its isolated nature and the widespread effects of the considered disturbances.

2.6 Description of PEGASE system

PEGASE (Pan European Grid Advanced Simulation and State Estimation) is a four-year (2008-2012) R&D collaborative project funded by the Seventh Framework Programme (FP7) of the European Union. It is carried out by a consortium coordinated by Tractebel Engineering and composed of 21 Partners which include transmission system operators, expert companies and leading research centres in power system analysis and applied mathematics[1].

Its overall objectives are to define the most appropriate state estimation, optimization and simulation frameworks for the European transmission network. It also considers their performance and dataflow requirements to achieve an integrated security analysis and control of the European transmission network.

The heart of PEGASE project involves the development of advanced algorithms and prototype software and the demonstration of the feasibility of real-time state estimation, multi-purpose constrained optimization and time-domain simulation of very large models[2].

Within this framework, a test system comprising the continental European synchronous area (see Fig. 2.6) was set up by Tractebel Engineering, starting from network data made available by ENTSO-E, the European Network of Transmission System Operators for Electricity. Dynamic components and their detailed controllers were added by Tractebel Engineering in a realistic way.

The main features of the model are:

- 15226 buses and 21765 branches;
- 3483 synchronous machines with generic models of their excitation systems, AVRs, OELs, turbines and speed governors;
- 7211 user-defined injectors. Some of them are equivalents of distribution systems with step-down transformers, medium-voltage feeders and induction motor loads;

[1]Please refer to [peg12] for more details on the PEGASE project and the project partners. The University of Liège is the fourth partner in terms of budget.
[2]The author of this thesis was involved for more than 3 years in this last task.

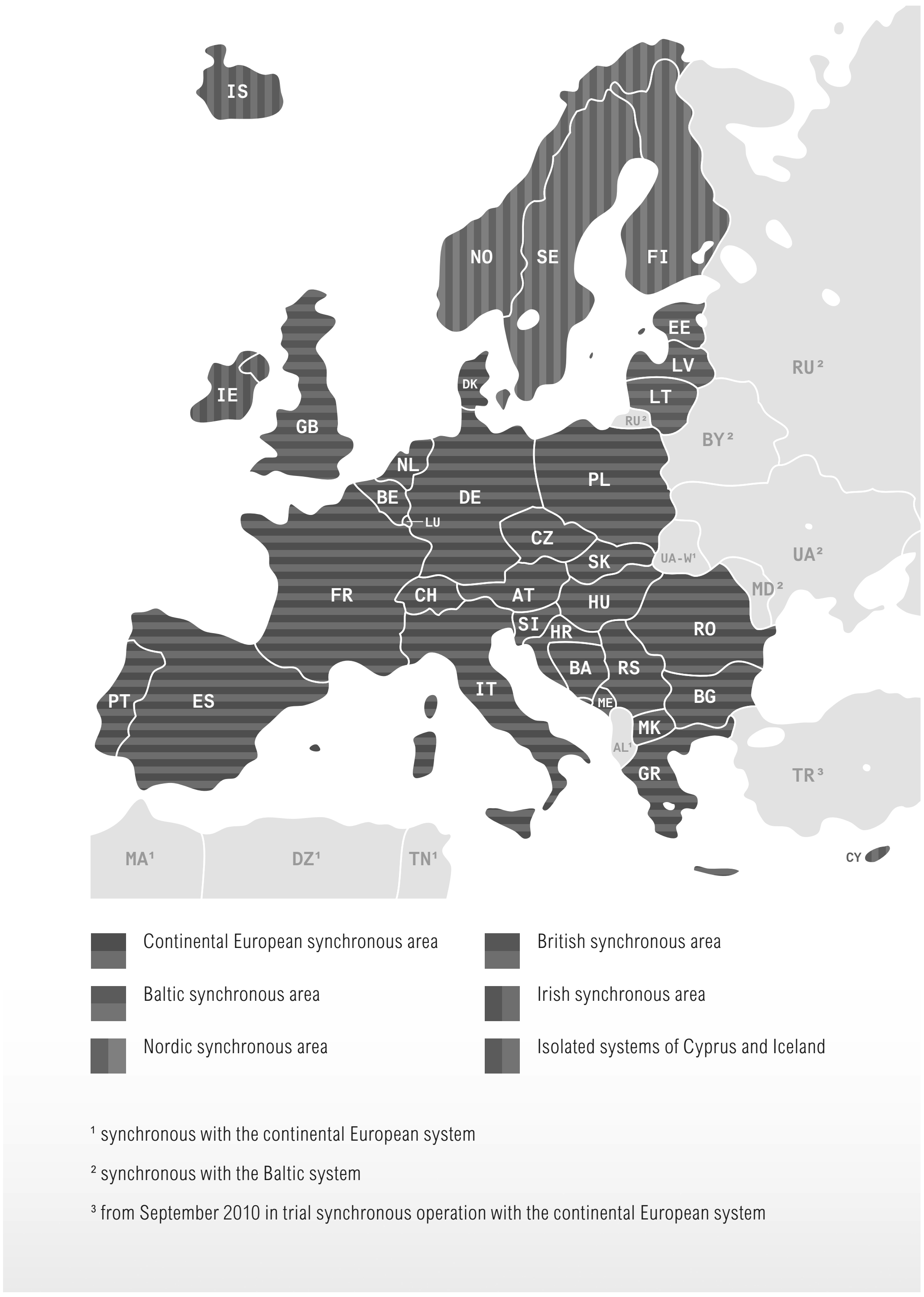

IS
NO
SE
FI
EE
LV
LT
RU²
IE
GB
DK
RU²
BY²
NL
BE
DE
PL
LU
CZ
SK
UA-W¹
UA²
FR
CH
AT
HU
MD²
RO
SI
HR
BA
RS
PT
ES
IT
ME
BG
MK
AL¹
GR
TR³
MA¹
DZ¹
TN¹
CY
Continental European synchronous area
British synchronous area
Baltic synchronous area
Irish synchronous area
Nordic synchronous area
Isolated systems of Cyprus and Iceland
¹ synchronous with the continental European system
² synchronous with the Baltic system
³ from September 2010 in trial synchronous operation with the continental European system

Figure 2.6: European map with indication of the interconnected ENTSO-E system members (2011); continental European synchronous area is represented in the PEGASE system

- 2945 Load Tap Changers (LTCs) represented as discrete devices.

This leads to a model with 146239 states. Among these, 72293 are differential and 73946 algebraic (out of which 30452 are V_x, V_y voltage components). There are also 62899 discrete variables, which yields an average of 22 state variables per power plant and 5 per dynamic load.

The PEGASE system is used here to assess the computational effort of the proposed algorithms on very large cases. The achievable gain is indeed very significant when all the techniques are combined, leading to a simulation which is not much slower than real-time and in some cases even faster.

Chapter 3

Simulating power system models

Indignum enim est excellentium virorum
horas servili calculandi labore perire [1]

In this chapter the numerical solution of the hybrid differential-algebraic system of equations set up in Chapter 2 is discussed. Popular integration formulae such as the backward differentiation formulae and even more the trapezoidal method and the role of their stability properties (A-stability, L-stability and hyper-stability) for the algebraization of the differential part of the model are discussed.

The solution of the continuous part of the model is detailed with a bias towards the use of the Newton method, that is our method of choice. The possible convergence criteria are outlined with emphasis on the practical implications of their adoption for the different components of power systems. The discretization of the continuous time and the choice of the integration time step are analysed with particular attention to its interaction with the discrete part of the problem. A discussion on the accuracy of simulated output and ways to measure it closes this chapter.

[1]*It is unworthy of excellent men to lose hours like slaves in the labor of calculation.* Gottfried Wilhelm Leibniz, in the unpublished manuscript *Machina arithmetica in qua non additio tantum et subtractio sed et mutiplicatio nullo, diviso vero paene nullo animi labore peragantur*, made available only centuries later by Dr. W. Jordan and C. Steppes in *Die Leibniz'sche Rechenmaschine*. English translation by Dr. M. Kormes in *A source book in mathematics*.

3.1 Integration formulae

The differential equations (2.4) need to be algebraized in order to be solved. The continuous time is discretized in a succession of instants (t_0, t_1, ...). There is a vast choice of integration formulae for different applications [AP98] with various features and computational efficiencies.

Integration formulae can make use of one past discretization instant or more than one: in the latter case these formulae are called multi-step. Both one- and multi-step methods will be used in this thesis.

Integration formulae that use low-accuracy intermediate discretization points which are then discarded from the solution are called multi-stage methods. The increased accuracy obtained with multi-stage methods with respect to one-stage ones comes at the cost of computing the intermediate stages, which is comparable to the computation of an additional point in one-stage methods (if not higher, as in the case of fully implicit Runge-Kutta methods). Since the main issue of power system dynamic simulation is to decrease the computational effort, and not to increase the numerical accuracy of the simulation, multi-stage methods have not been considered in this work.

Another important classification is between explicit and implicit methods. The simplest explicit integration formula is the Forward Euler Method (FEM):

$$\mathbf{w}(t_j) = \mathbf{w}(t_{j-1}) + h\dot{\mathbf{w}}(t_{j-1}) \tag{3.1}$$

where the vector of interest is $\mathbf{w}$, its value is known in t_{j-1} and sought in t_j, and the step size, i.e. the time length between the two instants, is h. In explicit methods, the sought value can be computed in closed form.

The simplest implicit integration formula is the Backward Euler Method (BEM):

$$\mathbf{w}(t_j) = \mathbf{w}(t_{j-1}) + h\dot{\mathbf{w}}(t_j) \tag{3.2}$$

Another popular implicit integration formula in power system dynamics is the Trapezoidal Method (TM):

$$\mathbf{w}(t_j) = \mathbf{w}(t_{j-1}) + \frac{h}{2}\left(\dot{\mathbf{w}}(t_{j-1}) + \dot{\mathbf{w}}(t_j)\right) \tag{3.3}$$

Despite their simplicity, the use of explicit methods is limited by their poor numerical stability, which degrade fast with the increase of the step size: this drawback is a major obstacle for their use in stiff systems. Better convergence for a greater range of step size values can be achieved with implicit methods, which, conversely, are methods where the sought value cannot be computed in closed form, but is the solution of a numerical computation. Since the possibility of increasing the step size to some extent is a crucial feature to achieve faster simulations on power systems, this thesis concentrates on implicit methods.

Backward Differentiation Formulae (BDF) are a family of implicit methods. The BDF of order p takes on the form [Gea71]:

$$\mathbf{w}(t_j) = \sum_{\ell=1}^{p} \gamma_\ell \, \mathbf{w}(t_{j-\ell}) + h\,\beta\,\dot{\mathbf{w}}(t_j) \tag{3.4}$$

where the γ_ℓ and β coefficients are computed according to a simple procedure [BGH72]: a polynomial $\boldsymbol{\phi}(t)$ which interpolates the sought point t_j and p previous points is found. The derivative of the polynomial, evaluated at the sought point t_j, is then imposed equal to the derivative function $\dot{\mathbf{w}}(t_j)$ at the sought point:

$$\dot{\boldsymbol{\phi}}(t_j) = \dot{\mathbf{w}}(t_j) \tag{3.5}$$

A procedure for the computation of these coefficients using Newton polynomials is recalled in Appendix A.4. The popular Nordsieck formulation [Nor61], which allows an easier treatment of step size changes for higher-order BDF, is not considered since in this work the order p will be limited to 2, for reasons discussed later in this section. Examples of coefficients for the first two BDF are given in Table 3.1. BEM is the BDF of order 1. The second-order BDF is denoted BDF2.

Table 3.1: γ_ℓ and β coefficients of the first two BDF

BDF order	γ_1	γ_2	β
1	1		1
2	$\frac{4}{3}$	$-\frac{1}{3}$	$\frac{2}{3}$

The BDF2 coefficients in Table 3.1 have been computed under the hypothesis of constant step size, and in principle cannot be used when the step size is changed. Appendix A.4 shows the derivation of the coefficients in the general case of a variable step, which allows for an exact and flexible handling of step size changes.

Equation 3.4 is easily rewritten as:

$$\frac{1}{h\,\beta} \sum_{\ell=1}^{p} \gamma_\ell \, \mathbf{w}(t_{j-\ell}) + \boldsymbol{\psi}_d(\mathbf{y}, \mathbf{w}(t_j), \mathbf{z}, \mathbf{V}, \mathbf{I}) - \frac{1}{h\,\beta}\mathbf{w}(t_j) = \mathbf{0}$$

Merging this equation with (2.2) and (2.3) and using the same variables as in (2.12) yields:

$$\frac{1}{h\,\beta}\boldsymbol{\Gamma}(\mathbf{z}) \sum_{\ell=1}^{p} \gamma_\ell \, \mathbf{x}(t_{j-\ell}) + \boldsymbol{\phi}(\mathbf{x}(t_j), \mathbf{z}, \mathbf{V}) - \frac{1}{h\,\beta}\boldsymbol{\Gamma}(\mathbf{z})\mathbf{x}(t_j) = \mathbf{0}$$

which can be written in compact form as:

$$\mathbf{0} = \mathbf{f}(\mathbf{x}, \mathbf{z}, \mathbf{V}) \tag{3.6}$$

where the dependency on time (t_j) has been omitted for simplicity.

Equations (2.11) and (3.6) make up the nonlinear system which will be solved at each time step by the Newton method, as shown in Section 3.2.

Desired properties of integration formulae are A-stability and L-stability. These properties are defined from the behavior of the formulae when applied to the so-called test equation:

$$\dot{w} = \lambda w \tag{3.7}$$

It is desirable that, whenever the exact solution is stable, i.e. λ has negative real part, so is the simulated one. When this hold true whatever the step size h, the method is said to be A-stable [AP98, CK06]. This is the case for TM and BDF up to order 2. It is not the case for FEM.

The poor stability of explicit methods is strictly linked with stiffness: in fact, another definition for the latter is that a system is stiff when, the stability requirements of FEM integration impose a smaller step size than the accuracy requirements [AP98].

Consider now Eq. (3.7) modified as follows:

$$\dot{w} = \lambda(w - g(t)) \tag{3.8}$$

where $g(t)$ is a bounded function. Assuming that the system is stable, an integration method is L-stable if, for a given $t_j > 0$:

$$|w(t_j) - g(t_j)| \to 0 \quad \text{as} \quad h \to \infty \tag{3.9}$$

This result can be interpreted as the system settling to the underlying equilibrium when the step goes to infinity. L-stability is also referred to as stiff-decay property [AP98, CK06].

The advantage of L-stable integration methods lies in their ability to skip rapidly varying solution “details” while maintaining a decent description of the coarse behavior of the solution. Conversely, non L-stable integration methods need to be used with small enough time steps even if only the coarse behavior of the solution is sought, otherwise errors get propagated and numerical oscillations are experienced. Chapter 6 hosts further discussions on L-stability property and its centrality in the context of stiff systems.

In spite of its popularity in detailed simulations, due to its A-stability property, TM is not L-stable. Hence, in principle, it has to be used with "small" steps (compared to the time constants of the underlying dynamics). On the other hand, BDF are L-stable. BDF of order higher than 2 are not considered in this work since they are not A-stable.

It is also desirable that, whenever the exact solution of (3.7) is unstable, i.e. λ has positive real part, so is the simulated one. For TM this holds true whatever the value of h. For BDF methods, however, there are values of h for which it does not. This is referred to as *hyper-stability*, which takes place when the step size is too large or the system marginally unstable. This drawback is more pronounced for BEM than for BDF2 and it suggests to use caution when simulating systems with lightly undamped modes of oscillation, and to not increase the step size beyond reasonable values. This drawback will be further discussed in Chapter 6.

The integration formula of choice in this thesis is the BDF of order 2, for both the benchmark and the proposed methods. This formula will be initialized by the order one BDF, i.e. BEM.

3.2 Solution of nonlinear systems of equations

We showed in the previous section that solving a system of differential equations in the form

$$\dot{\mathbf{x}} = \boldsymbol{\phi}(\mathbf{x}) \tag{3.10}$$

requires to algebraize them and solve a system of nonlinear equations of the type:

$$\mathbf{0} = \mathbf{f}(\mathbf{x}) \tag{3.11}$$

In case BEM is used, for instance, for the algebraization process the following holds:

$$\mathbf{f}(\mathbf{x}_{t_j}) = \mathbf{x}_{t_j} - \mathbf{x}_{t_{j-1}} - h\boldsymbol{\phi}(\mathbf{x}_{t_j}) \tag{3.12}$$

Functional iteration is a basic method for solving (3.12). Given a first guess $\mathbf{x}^0_{t_j}$, it computes the successive iterates ($k = 1, 2, \ldots$) according to:

$$\mathbf{x}^k_{t_j} = \mathbf{x}_{t_{j-1}} + h\boldsymbol{\phi}(\mathbf{x}^{k-1}_{t_j}) \tag{3.13}$$

which is merely a rewrite of (3.12). Functional iteration is a very simple method but it has poor convergence properties: in fact in order to converge it requires $h||\dfrac{\partial \boldsymbol{\phi}}{\partial \mathbf{x}}|| < 1$ in some norm, bringing a restriction in the step size h [AP98]. This requirement strongly limits the effective usage of functional iteration for power system dynamic simulation.

Newton method exhibits much better convergence properties. Given a first guess $\mathbf{x}_{t_j}^0$, it is based on the first-order truncation of the Taylor series expansion

$$\mathbf{f}(\mathbf{x}_{t_j}^k) \simeq \mathbf{f}(\mathbf{x}_{t_j}^{k-1}) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \left(\mathbf{x}_{t_j}^{k-1}\right) \left(\mathbf{x}_{t_j}^k - \mathbf{x}_{t_j}^{k-1}\right) \tag{3.14}$$

in which assuming $\mathbf{f}(\mathbf{x}_{t_j}^k) \simeq \mathbf{0}$ yields:

$$\mathbf{J} \left(\mathbf{x}_{t_j}^{k-1}\right) \Delta\mathbf{x} = -\mathbf{f}(\mathbf{x}_{t_j}^{k-1}) \tag{3.15}$$

where $\mathbf{J} = \dfrac{\partial \mathbf{f}}{\partial \mathbf{x}}$ is the Jacobian matrix and $\Delta\mathbf{x} = \mathbf{x}_{t_j}^k - \mathbf{x}_{t_j}^{k-1}$ is the correction at iteration k.

Although theoretically $\Delta\mathbf{x}$ could be isolated by pre-multiplying on the left by $\mathbf{J}\left(\mathbf{x}_{t_j}^{k-1}\right)^{-1}$, the computation of a matrix inverse is a heavy operation. In practice, the Jacobian matrix is factorized as $\mathbf{J} = \mathbf{L}\,\mathbf{U}$, the product of a lower triangular $\mathbf{L}$ and an upper triangular $\mathbf{U}$ matrix. Efficient sparse solvers [TW67] are widely available to solve (3.15) for $\Delta\mathbf{x}$ through LU factorization of the Jacobian matrix.

According to (3.15), $\mathbf{J}$ should be evaluated with $\mathbf{x}_{t_j}^{k-1}$ and factorized at each new solution for $\mathbf{x}_{t_j}^k$. A more efficient strategy, known as "dishonest Newton method", consists in evaluating and factorizing the Jacobian as few times as possible and reusing the same $\mathbf{J}$ for successive iterations and over multiple time steps: this naturally leads to some degradation of convergence speed (i.e. an increase in the number of iterations) that depends on the update frequency, but the overall computational effort decreases substantially.

3.3 Practical application of Newton method

The equations to solve with the Newton method at each time step are (2.11) and (3.6), repeated here for convenience:

$$\mathbf{g}(\mathbf{x},\mathbf{V}) = \mathbf{D}\,\mathbf{V} - \mathbf{C}\,\mathbf{x} = \mathbf{0} \tag{3.16}$$

$$\mathbf{f}(\mathbf{x},\mathbf{z},\mathbf{V}) = \mathbf{0} \tag{3.17}$$

with respect to $\mathbf{V}$ and the state vector $\mathbf{x}$. Since [DS72], the simultaneous solutions of these equations with the dishonest Newton method can be considered the standard in dynamic simulation of power systems.

This requires solving a sequence of linear systems ($k = 1, 2, \ldots$):

$$\mathbf{J} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{V} \end{bmatrix} = - \begin{bmatrix} \mathbf{f}(\mathbf{x}^{k-1}, \mathbf{z}, \mathbf{V}^{k-1}) \\ \mathbf{g}(\mathbf{x}^{k-1}, \mathbf{V}^{k-1}) \end{bmatrix} \tag{3.18}$$

where $\mathbf{J}$ is the Jacobian matrix:

$$\mathbf{J} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ -\mathbf{C} & \mathbf{D} \end{bmatrix} \tag{3.19}$$

and incrementing the variables according to:

$$\begin{aligned} \mathbf{x}^k &= \mathbf{x}^{k-1} + \Delta\mathbf{x} \\ \mathbf{V}^k &= \mathbf{V}^{k-1} + \Delta\mathbf{V} \end{aligned}$$

In (3.19), $\mathbf{A}$ is the Jacobian of $\mathbf{f}$ with respect to $\mathbf{x}$, $\mathbf{B}$ the Jacobian of $\mathbf{f}$ with respect to $\mathbf{V}$.

Another option to solve the above system is to use a partitioned approach, i.e. to solve for $\mathbf{x}$ and $\mathbf{V}$ alternatively, with a degradation of accuracy due to the tearing of the system of equations. This approach was favored in early dynamic simulation programs since dealing with the whole system matrix was not computationally feasible in the early years of scientific computing, and because, for the solution of the network, efficient solvers for structurally symmetric problems were widely available.

In this thesis, the simultaneous approach solved with the dishonest Newton method will be used as a benchmark because of its better convergence properties. This method is also regarded as the reference method with respect to which the performance of new algorithms can be evaluated [IEE92].

The convergence criteria used to stop the Newton iterations significantly impacts the overall efficiency of the solution scheme: in Chapter 4 we will show to which extent.

The vector in the right-hand side of (3.18) equation is usually referred to as "mismatch vector" (of the $\mathbf{f}$ and $\mathbf{g}$ equations, respectively). To ensure that *all* equations are solved within the specified tolerance, the infinite-norm (i.e. the largest component magnitude) of each mismatch vector should be brought below some tolerance. Thus, the Newton iterations are stopped at iteration k if:

$$||\mathbf{g}(\mathbf{x}^k, \mathbf{V}^k)||_\infty < \epsilon_g \tag{3.20}$$

$$||\mathbf{f}(\mathbf{x}^k, \mathbf{z}, \mathbf{V}^k)||_\infty < \epsilon_f \tag{3.21}$$

For the network equations $\mathbf{g}$, whose components are currents in per unit on the system base, the choice of ϵ_g is rather easy. For the remaining $\mathbf{f}$ equations, on the other hand, it may be difficult to choose an appropriate ϵ_f value in so far as the simulation software hosts user defined models and the solver has no knowledge of those models. This makes it difficult to decide whether a component of the mismatch vector in (3.21) is "negligible" or not.

This issue can be solved by checking the magnitude of the correction vector $\Delta\mathbf{x}$ instead of the mismatch vector. Each state variable correction can be compared to the state variable itself. More precisely, the test (3.21) is replaced by:

$$\left|\frac{(\Delta\mathbf{x}^k)_\ell}{(\mathbf{x}^k)_\ell}\right| < \epsilon_\Delta \qquad \ell = 1,\ldots,dim(\mathbf{x}) \tag{3.22}$$

Clearly, the price to pay for this better suited test is the computation of $\Delta\mathbf{x}$, which requires to solve (3.6) at least once before deciding that no further iteration is necessary. Safeguards are put in place for values of $\mathbf{x}$ approaching zero.

The choice of the infinite-norm in Eqs. (3.20-3.22) is of particular importance. In fact, the choice of any other norm would be system-dependent insofar as it would be prone to a sort of "mismatch dilution" in systems of different size; for instance, a large mismatch in one equation could be "shaded" by very small mismatches in the rest of the system: although unlikely in small systems, this possibility cannot be ruled out in larger ones. The infinite-norm, conversely, despite being slightly more computationally intensive, for equal tolerances, with respect to lower order norms, assures that any equation is satisfied according to a predefined tolerance.

3.4 Step size control

The integration step is in general not constant throughout the whole simulation. The main reason for the step size variation has to do with the different phases that a system evolution normally goes through. In fact, power system dynamic simulations are usually initiated by a disturbance that excites fast dynamics: this phase is usually referred to as short-term, and it requires integration with small step size, otherwise numerical divergence or inaccuracy might be experienced. Then the system goes through the long-term phase, where the step size may be increased, if the integration method allows it, since this period is dominated by slow dynamics. Eventually, some fast dynamics might be triggered also in this phase, and the step size should be adjusted in order to prevent numerical difficulties.

In Section 2.3 we introduced the difference between the flows of continuous variables, i.e. the smooth evolution over intervals of time and the jumps of discrete variables, i.e. the instantaneous change over instants. We saw in Section 3.1 that the evolution of flows over continuous time intervals has to be discretized in a succession of time steps if a numerical solution is sought. For these reasons, the time set of hybrid system simulation has been referred to as "hybrid time set" [Lyg04] insofar as it results from the succession of continuous time intervals and discrete time instants.

The discretization of the continuous interval in time steps presupposes that the duration of the time-step is chosen by the algorithm according to accuracy or efficiency considerations or often a compromise between the two. Since the step size h appears in the right-hand side of Eq. (3.4), and thus in the Jacobian matrix $\mathbf{J}$, it is customary that a significant variation of step, even within a dishonest solution framework, requires an immediate Jacobian update in order to avoid convergence difficulties with outdated values of h.

The savings in Jacobian updates motivated the use of fixed time-step algorithms, which can be adopted even in case of stiff systems provided that an L-stable formula (e.g. BDF) is used for integration. For L-unstable methods such as TM, variable time steps need to be used on stiff systems to avoid inefficiently small steps. We will see in Chapter 6 that fixed time step L-stable algorithms have further advantages if applied to stiff systems when the step size is limited by the presence of discrete events. The interaction of the discrete instants with the time step size will be discussed in the following Section 3.5.

Variable time step algorithms choose different time steps according to desired error or effort. A popular way to achieve the desired precision is through the estimation of the Local Truncation Error (LTE) [Gea71]. In fact since for a method of order p and time step h, the corresponding LTE is of the order of h^p, it is straightforward to bound the error by varying h. Since this estimation of error relies on a Taylor series expansion, it must be kept in mind that its validity is limited to step sizes smaller than the dynamics excited [CK06]. Alternative estimations of effort have relied on other measures, most notably the number of Newton iterations [Sto79].

Another possibility consists in varying the step size according to the estimated computational effort: this is based on the fact that the computational effort varies with the step size. Estimations of effort have relied on the magnitude of the mismatch vector in our earlier investigations [FVC09], while presently for effort estimation we favor the number of Newton iterations, originally proposed in [Sto79] for error estimation.

Variable step size implicit algorithms based on LTE are popular in power systems dynamic simulation, due to the presence of phenomena belonging to very different time scales:

the underlying assumption is that, once the fast dynamics have died out, the integration can be performed with larger steps, while preserving the accuracy through procedures that bound the LTE. Reference [ABJ94] introduces the mixed Adams-BDF formula, a method that would combine the symmetrically A-stable property of TM and the L-stability of BDF2: the overall accuracy is controlled by bounding the LTE, not only as defined traditionally on differential variables but also extended to the algebraic part of the problem. Reference [SGDPP95] proposes the use of the Theta method[1] for similar purposes. More recently, reference [FMT12] advocates the use of Hammer-Hollingsworth 4 (HH4), an implicit two-stage Runge-Kutta method. Although the above references claim to retain full accuracy, it must be kept in mind that the integration formulae and the measures of accuracy are based on interpolations and approximations, and should thus be limited to reasonable step sizes, if accuracy has to be preserved. In some of these references, conversely, steps of 10, 20 and even 180 s are taken during the simulation. In the author's opinion, a dynamic simulation that might retain full accuracy with such steps is the simulation of a scenario which is already very close to steady-state and that experiences very few discrete events!

In fact, although variable time step algorithms allow increasing the step size to very large values, provided that the error estimation lies within the desired bounds, the actual limitation to the step size, in a large system, comes from the time discretization due to discrete events. Figure 3.1 shows a typical discrete event distribution obtained on the PEGASE test system (Case P3, detailed in Section 4.8.3). This simulation has been performed with steps of 0.05 s for most of the simulation: smaller steps have been used only in the half second after the initiating disturbance. 1988 discrete events of the type (2.13) occur during this 4-minute simulation, i.e. one event per 0.12 s. The ordinate axis displays a spike with value 1 whenever an event takes place: the theoretical maximum step size is limited by the distance between two spikes and it is shown in Fig. 3.2. Furthermore this is a theoretical limit, not the actual step size: in fact, the latter may be smaller due to the higher LTE estimation which generally follows discrete jumps [CK06].

[1]The Theta method is defined as $\mathbf{w}(t_j) = \mathbf{w}(t_{j-1}) + h\left(\theta\dot{\mathbf{w}}(t_{j-1}) + (1-\theta)\dot{\mathbf{w}}(t_j)\right)$. For $\theta = 0$ it corresponds to BEM, for $\theta = 0.5$ to TM and for $\theta = 1$ to FEM, respectively.

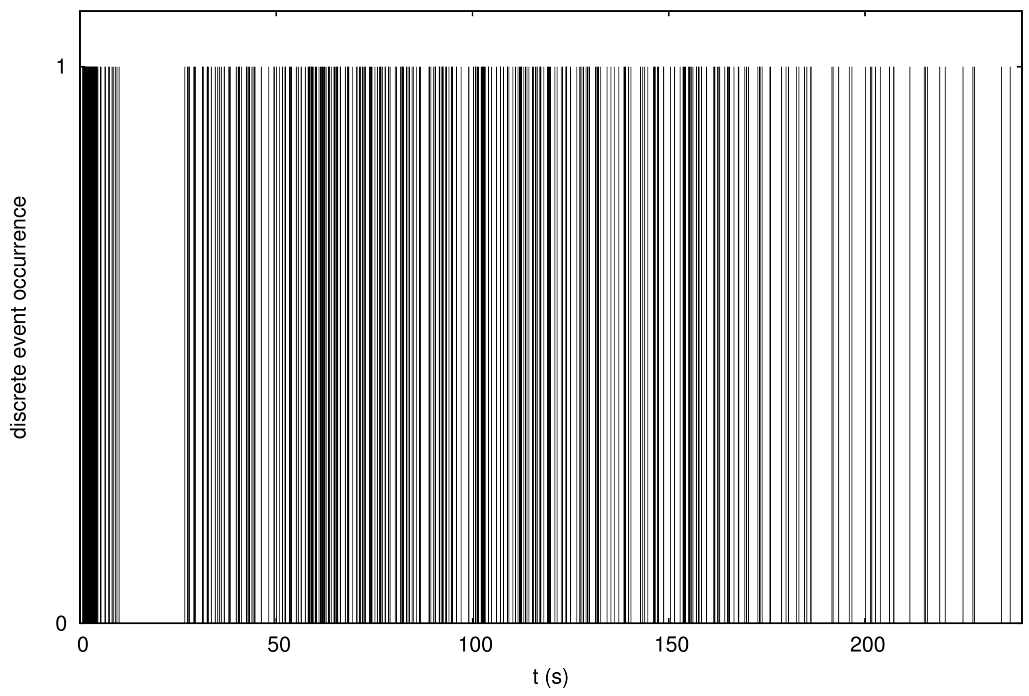


Figure 3.1: Discrete event occurrence in Case P3

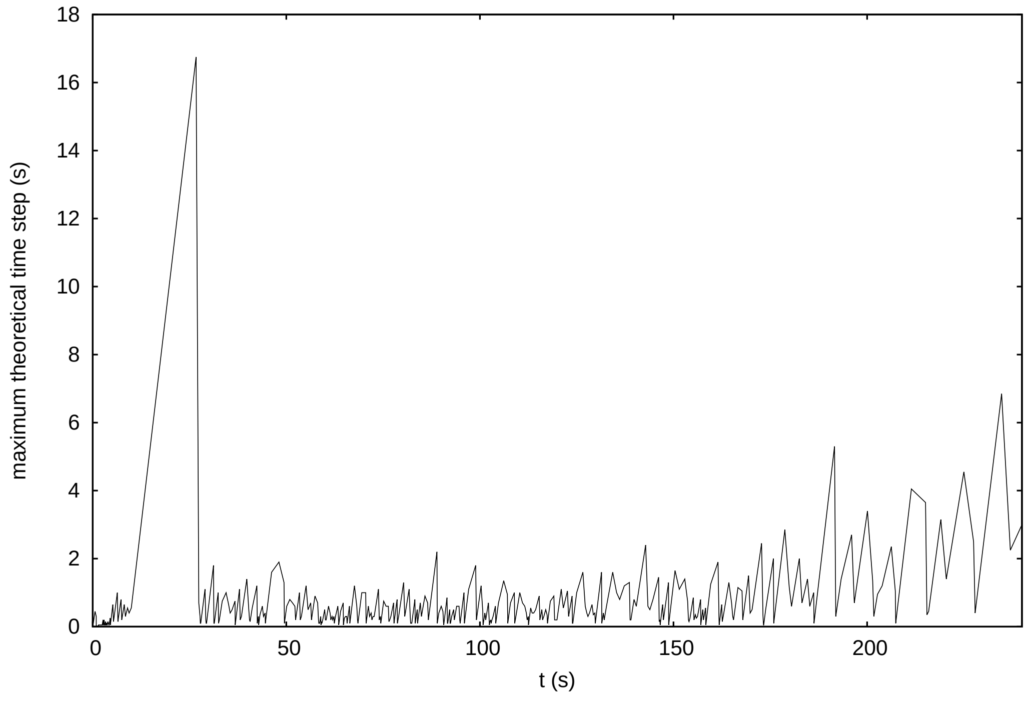


Figure 3.2: Maximum theoretical step size in Case P3

3.5 Events and discontinuities

Relatively few attention has been paid in the literature on power system dynamic simulation to the handling of discrete parts of the model. In the general case, whether the solver uses fixed or variable step size, those instants where variables change do not coincide with a simulation time step. Thus, the most important effect of events on dynamic simulations is that they force the solver to adjust the length of continuous-time evolution intervals in order to land over the event times [CK06].

This mechanism, in case of many events, leads to a reduction of the average step size (and hence an increase in the number of steps). Moreover, as seen in Section 3.4, when Newton method is used to solve (3.16) and (3.17), the Jacobian of those equations with respect to the state $(\mathbf{x}, \mathbf{V})$ has to be updated and re-factorized when the step size h changes significantly.

In detailed simulation the handling of events requires the solver to perform several tasks in addition to the aforementioned step size adjustment: these tasks affect equally state events and time events, regardless whether they stem from the simulation itself or from a user-imposed event. By definition, a jump induces a change in some components of $\mathbf{z}$, and hence a discontinuity of $\dot{\mathbf{w}}$. The voltages will undergo a discontinuity, i.e. $\mathbf{V}(t_j^-) \neq \mathbf{V}(t_j^+)$. Furthermore, the jump may cause some algebraic variables to undergo a discontinuity, i.e. there is at least one ℓ such that $x_\ell(t_j^-) \neq x_\ell(t_j^+)$ and $(\mathbf{\Gamma})_{\ell\ell} = 0$. In this case, the solver has to determine the new values $x_\ell(t_j^+)$ and $\mathbf{V}(t_j^+)$. In principle, the occurrence of a discontinuity requires as well to momentarily decrease the order of the integration formula to one.

Indeed, from a strict mathematical viewpoint, it is not correct to integrate over one of those discontinuities; in fact, all numerical integration methods are based on polynomial expansions which cannot reproduce discontinuities. However, BEM allows to integrate over a discontinuity because in the corresponding equation (3.2), if the discontinuity takes place at time t_{j-1}, the derivative $\dot{\mathbf{w}}(t_{j-1}^+)$ is not used to compute $\mathbf{w}(t_j)$. Therefore this scheme can be used for the time step that immediately follows a discontinuity. However, in some cases, it is recommended to take a small h (0.001 s, for instance) after a large discontinuity, e.g. a short circuit.

An alternative, seemingly more accurate solution, would consist in solving the algebraic part of the model at the discontinuity instant. Two reasons suggest not to do so, though:

- the system Jacobian matrix cannot be used for this computation, since the differential variables are discarded, so a reduced Jacobian for the algebraic variables only has to be evaluated and factorized;

- the accuracy gain is somewhat illusory since the network is modelled by phasor equations, which are themselves an approximation (neglecting electromagnetic transients). This kind of illusional accuracy gain is discussed further in Chapter 7.

The computations typically performed by a detailed solver are as follows. Let us assume that, at time t, it is intended to take a step of length h. Let us suppose that a zero-crossing function analysis informs the solver that a state event is going to take place at $t + h' < t + h$, where h' has to be determined with proper accuracy, or equivalently that a time event is scheduled to take place at $t + h'$. The solver will thus perform the integration on (3.16) and (3.17) with a step h', impose the event according to (2.13), solve the associated discontinuity, change the flow and proceed with the simulation. In practice, the accurate identification of the state event time $t + h'$ may require additional steps if the zero-crossing function is strongly nonlinear.

The firing of many discrete events in short periods of time, which cannot be avoided in the simulation of large systems, is probably the main factor preventing the step size to increase, and the main obstacle in the way towards faster simulations. Attention is paid to this problem in Chapter 6.

3.6 Dynamic time warping measure of accuracy

In this work we have often assessed the accuracy of the simulated output provided by some algorithms by comparing it to a reference evolution. The problem is to assess how close a dynamic evolution characterized by a time-series $\mathbf{q} = [q_1, q_2, \dots, q_n]$ is to a reference characterized by a time-series $\mathbf{r} = [r_1, r_2, \dots, r_n]$. The former is referred to as the query and the latter as the reference time-series.

In the power system literature, the accuracy of query with respect to reference is often depicted by qualitative, and admittedly subjective, evaluation terms such as "we see a good overall match between $\mathbf{q}_1$ and $\mathbf{r}$", "$\mathbf{q}_2$ and $\mathbf{r}$ have nearly the same degree of accuracy" or "$\mathbf{q}_3$ is almost undistinguishable from $\mathbf{r}$", etc.

The Euclidean distance, involving pairs of points of the two data series aligned in time, is not suited to the practical time evolutions met in power systems, which often involve variable time delays and jumps at discrete times. An interesting alternative is Dynamic Time Warping (DTW) [FVC11a], an algorithm originally developed some decades ago for speech recognition [Vin68, SC78] and now used in various fields such as medicine [TGQS09], data mining [KP01], signature verification [EFV07], and others. In power systems it has been previously used to classify power quality events [YAGESS04] and also to detect instability modes [KL10].

DTW warps the time axis to guarantee the best match between both time-series. More precisely, it maps the points of both curves so that the sum of their squared ordinate differences is minimum, under some constraints. This is equivalent to determining the deformation of the time axes which brings the two time-series as close as possible in the Euclidean-norm sense. The minimization is performed with the help of an efficient dynamic programming algorithm.

Let us define the squared distance between one point of $\mathbf{q}$ and one point of $\mathbf{r}$ as :

$$\ell_{(i,j)} = (q_i - r_j)^2 \quad i,j = 1,\ldots,n \tag{3.23}$$

The Euclidean distance in between the two time-series is given by:

$$E(\mathbf{q},\mathbf{r}) = \sqrt{\sum_{i=1}^{n} \ell_{(i,i)}} \tag{3.24}$$

Relaxing the constraint that the associated points of $\mathbf{q}$ and $\mathbf{r}$ relate to the same time instants, we define a warping path as an N-dimensional array of pairs

$$\mathbf{w} = [w_1\, w_2 \ldots w_N] = [(i_1,j_1)\,(i_2,j_2)\ldots(i_N,j_N)] \tag{3.25}$$

where the first element of each pair is an index referring to a point of $\mathbf{q}$ and the second element to a point of $\mathbf{r}$. In addition, a warping path obeys the following three constraints:

1. *Boundary condition*. The path starts at the initial point of both time-series, i.e. $w_1 = (1,1)$, and ends at the final point of both time-series, i.e. $w_N = (n,n)$. The requirement on the final point can be somewhat relaxed to cope with evolutions not ending at the same time [TGQS09].

2. *Continuity*. Given $w_k = (i,j)$ and $w_{k+1} = (i',j')$, the following inequalities hold true: $i' \leq i+1$ and $j' \leq j+1$. Thus, the path does not "skip" any point of either time-series.

3. *Monotonicity*. Given $w_k = (i,j)$ and $w_{k+1} = (i',j')$, the following inequalities hold true: $i' \geq i$ and $j' \geq j$ and $(i'-i)+(j'-j) \geq 1$. Thus, the path neither "goes back in time" nor "stalls".

For a given warping path $\mathbf{w}$, one can define the distance D between $\mathbf{q}$ and $\mathbf{r}$ along that path as:

$$D(\mathbf{q},\mathbf{r},\mathbf{w}) = \sqrt{\sum_{k=1}^{N} \ell_{w_k}} \tag{3.26}$$

where ℓ_{w_k} is the quantity defined by (3.23) for the pair (i,j) present in w_k, the k-th component of $\mathbf{w}$.

The local distance matrix $\mathbf{L}$ is defined as follows:

$$(\mathbf{L})_{ij} = \ell_{(i,j)} \quad i,j = 1,\ldots,n \tag{3.27}$$

To determine "how close" the time-series $\mathbf{q}$ and $\mathbf{r}$ are from each other, it is appropriate to consider the warping path which minimizes $D_d(\mathbf{q},\mathbf{r},\mathbf{w})$, i.e. to determine the $\mathbf{w}^\star$ solution of:

$$\min_{\mathbf{w}} D(\mathbf{q},\mathbf{r},\mathbf{w}) = \min_{\mathbf{w}} \sqrt{\sum_{k=1}^{N} \ell_{w_k}} \tag{3.28}$$

and use the corresponding distance $D(\mathbf{q},\mathbf{r},\mathbf{w}^\star)$ as the sought measure. The latter will be referred to as the DTW distance.

Finding the optimal warping path is a combinatorial optimization problem. In spite of the large search space, this problem can be solved with remarkable efficiency through Dynamic Programming (DP) [Vin68, SC78].

Although it makes sense to minimize $D(\mathbf{q},\mathbf{r},\mathbf{w})$ to identify the optimal warping path between two time-series, the value of $D(\mathbf{q},\mathbf{r},\mathbf{w}^\star)$ by itself may not be a meaningful measure of their proximity. For instance, if a larger number of points is used for the comparison, the value of $D(\mathbf{q},\mathbf{r},\mathbf{w}^\star)$ will increase accordingly. To deal with this situation, it sounds more appropriate to consider the normalized DTW distance:

$$\delta = \sqrt{\frac{\sum_{k=1}^{N} \ell_{w_k}}{N}} = \frac{D(\mathbf{q},\mathbf{r},\mathbf{w}^\star)}{\sqrt{N}} \tag{3.29}$$

If the two curves are smooth enough compared to the sampling period, a higher sampling will lead to a higher value of N but one can expect δ to be few affected.

Further practical information can be obtained from the optimal warping path. With the pairs of indices in $\mathbf{w}^\star$ denoted as in (3.25), we define the vector of curve offsets as:

$$\mathbf{y} = [q_{i_1} - r_{j_1}, q_{i_2} - r_{j_2}, \ldots, q_{i_N} - r_{j_N}] \tag{3.30}$$

The differences between the two curves can be measured by:

- the average curve offset:

$$\mu = \frac{\sum_{k=1}^{N} [\mathbf{y}]_k}{N} = \frac{1}{N} \sum_{k=1}^{N} \left(q_{i_k} - r_{j_k} \right) \tag{3.31}$$

- the standard deviation of the curve offsets:

$$\sigma = \sqrt{\frac{\sum_{k=1}^{N} ([\mathbf{y}]_k - \mu)^2}{N}} \tag{3.32}$$

For instance, when comparing two time-series, a small value of μ indicates that, with a proper deformation of the time axes, the curves coincide on the average. When comparing several queries to a reference, a larger value of σ denotes more "volatility" with respect to that reference. In case of oscillatory responses, for instance, a small σ indicates that both curves have similar damping.

For a more accurate description of the used measures, and further details, including examples on Nordic32 and Hydro-Québec systems, the reader is invited to refer to [FVC11a].

Chapter 4

Decomposing Newton iterations

Dicebat Bernardus Carnotensis
nos esse quasi nanos, gigantium humeris insidentes,
ut possimus plura eis et remotiora videre [1]

This chapter starts with a review of decomposition methods which have been applied to power system dynamic simulation. The bordered block diagonal decomposition is derived from the power systems model and the corresponding Newton method equations established in Chapters 2 and 3. The resulting particular structure is then exploited in order to solve only the parts of the system that need to be solved, and avoid updates of the whole Jacobian matrix when updating sub-matrices suffices. The description of the scenari and the benchmarks used on the three test systems throughout this thesis closes this chapter along with relevant results showing that the accuracy of the simulated outputs is not degraded with respect to the standard Newton method, while the computational effort is reduced.

4.1 Decomposition methods in literature

It is widely recognized that present-day interconnected power systems are the largest man-made machines in the world (e.g. [Bos03]): these systems result, more than from a central-

[1]*If I have seen further it is by standing on ye sholders of giants* [sic]. Originally attributed to Bernard de Chartres (Bernardus Carnotensis) as reported by John of Salisbury (Johannes Parvus) in *Metalogicon*. Free translation in English by Sir Isaac Newton who notably used it in a letter to Robert Hooke, published in *The correspondence of Isaac Newton*.

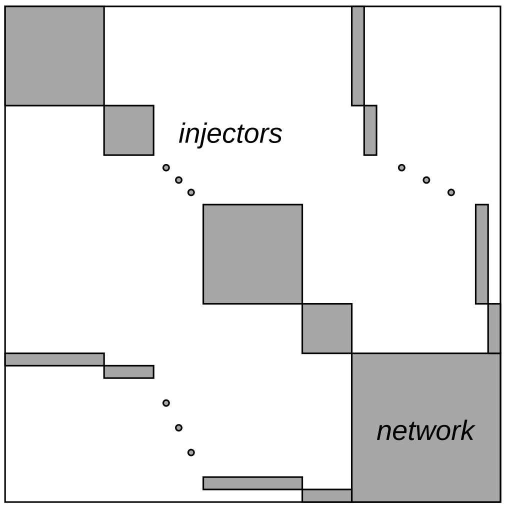


Figure 4.1: Bordered block diagonal structure

ized building effort, from the successive interconnection of many several smaller systems. From this point of view, it is not surprising that power system practitioners, whenever they are facing simulation challenges that stem from the problem size, resort to decomposition (and often simplification, through equivalencing and/or aggregation) of the system. Any power system model relies, at some extent, on such decomposition and simplification processes (e.g. the representation of loads as equivalent injections in power flow computations and dynamic simulations).

The exploitation of a peculiar structure is transversal to many power system simulation problems and it is crucial for large-scale systems [RM09].

The Bordered Block Diagonal (BBD) decomposition exploits the structure of a power system, seen as a set of power components (or *injectors*) that interact with each other through the network only [Boe77]. Figure 4.1 shows the sparsity pattern of typical power system matrices, composed of diagonal blocks, the injectors, bordered by the network.

A similar BBD structure is exploited also in VLSI circuit simulation [Ogr94]. The BBD formulation of the power system models and equations established in the previous chapters will be derived in Section 4.2. While historically the main function of the BBD decomposition was to provide an algorithm for parallel multi-processor architectures [HBJVN77, FP78], in this work it will be exploited in a novel way on single processor machines (in this chapter) for applications that preserve the accuracy of the simulated output, and (in the next chapter) for further speed-ups at the cost of reduced accuracy. The proposed approaches will still be suitable for parallel processing, but this is outside the scope of this thesis.

An extension of BBD decomposition is the so-called sub-tree reduction method [FX06], which allows a hierarchal and recursive application of the BBD algorithm: the network is partitioned in local subnetworks that interact with each other through the common "backbone" and then the local subnetworks may be further decomposed recursively into multiple layers until deemed profitable. The sub-tree approach is particularly interesting for those problems where several voltage levels are considered, for instance with the inclusion of distribution networks, whose importance for stability is growing due to the development of dispersed generation.

The BBD structure can be exploited in both *direct* and *relaxation* methods. The latter are more popular insofar as they allow a more straightforward parallelization: for instance, Gauss-Jacobi methods have been applied to a BBD decomposition of power systems in [LSBTT90].

Probably the best-known domain decomposition methods for power system dynamic simulation are those used in waveform relaxation. In waveform relaxation methods, the system of equations is decomposed in subdomains [MR06] which are algebraized and solved independently over a time window [ISCP87], possibly in parallel [CI90]. To this family belongs as well the "instantaneous relaxation" method [JMD09], where the time window coincides with the integration step, and the methods based on Picard iterations [LSBTC90], where the decomposition and algebraization steps are just swapped with respect to the traditional waveform relaxation scheme.

Multi-rate methods [CC94], although they can be classified in a generalized waveform relaxation family, are worth mentioning in a *sui generis* relaxation category. These methods could indeed be seen as waveform relaxation algorithms in which the different subdomains are algebraized and solved with different integration steps. Their use is motivated by the multiple time scales present in the power system model especially when long-term phenomena and devices are taken into account [CC96]. Variable partitioning multi-rate methods [CC08], which adapt the subdomains according to the "level of activity" of the variables involved, cannot be easily categorized as mere decomposition methods and they will be discussed in more detail in Chapter 5, from a space perspective, and in Chapter 6, from a time perspective.

While the decomposition techniques described above affect the systems of equations regardless of the solution scheme chosen, there exist approaches that exploit decomposition only in the computation of the Newton Jacobian. These approaches fall into the category of compensation methods [AST83], and were primarly investigated and used on the power flow problem. The basic idea behind these methods is to avoid full Jacobian refactorization when only a (small) part of this matrix needs to be updated. Among them one can distinguish between methods that use the Inverse Matrix Modification Lemma (IMML) and Partial Refactorization (PR) techniques.

When a matrix has been factorized, its LU factors are stored. In case the matrix is modified, the IMML is used to adapt the LU factors in order to take into account the modification that the original matrix experienced: this allows to reuse most of the elements of the **L** and **U** matrices while updating only some. Its efficiency decays with the increase of the rank of the modification matrix, which is usually small for power flow problems: indeed, in those problems, the Jacobian matrix may require an update because of a change in network topology that in general affects a small number of equations, resulting in a low-rank modification. On the other hand, this technique is not easily transposable to the dynamic problem, in so far as the dynamic part of the problem requires in general more frequent and widespread updates, not only due to changes of topology but mainly imposed by the nonlinearity of the involved equations themselves.

Algorithms based on PR [CB86], conversely, perform better in so far as they conserve as much as the original factorization as possible. The Jacobian refactorization strategy detailed in Section 4.4 is loosely inspired from these techniques.

4.2 Bordered Block Diagonal decomposition

Equations (2.11) and (3.6) have a structure that reflects the physical structure of the system itself, i.e. a set of n *injectors* interacting with each other through the network only, as sketched in Fig. 4.2. Note that the term injector is used here in a wide sense, since it may relate to a device that either produces or consumes power. The derivations of this section have been extended to two-port injectors, i.e. components connected to two buses (such as HVDC links for instance), as it will be detailed later in Section 4.5.

The state vector $\mathbf{x}$ and the discrete state vector $\mathbf{z}$ can be split into $\mathbf{x}_i$ and $\mathbf{z}_i$, respectively, relative to the i-th injector. Without loss of generality, we assume that the first two (out of n_i) components of $\mathbf{x}_i$ are I_{xi} and I_{yi}, the rectangular components of the complex current

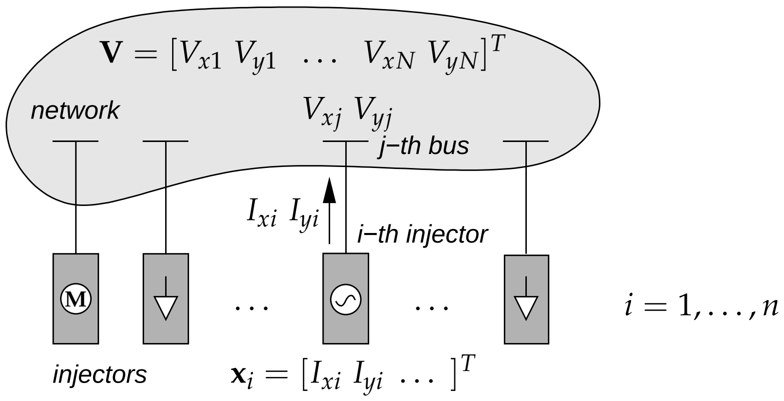


Figure 4.2: Power system decomposition into network and injectors

injected into the network:

$$\mathbf{x}_i = [I_{xi}\ I_{yi}\ \ldots]^T \qquad i = 1, \ldots, n$$

The $\mathbf{C}$ matrix is decomposed as well as follows:

$$\mathbf{C} = [\mathbf{C}_1\ \mathbf{C}_2\ \ldots\ \mathbf{C}_n]$$

$\mathbf{C}_i$ aims at extracting the I_{xi} and I_{yi} components of $\mathbf{x}_i$ and subtracting them from the appropriate current mismatch equation. Thus, assuming that the i-th injector is connected to the j-th bus ($j = 1, \ldots, N$):

$$\mathbf{C}_i = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & 0 & \ldots & 0 \\ 1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ \vdots & \vdots & 0 & \ldots & 0 \\ 0 & 0 & 0 & \ldots & 0 \end{bmatrix} = \left[\mathbf{e}_{2j-1}\ \mathbf{e}_{2j}\ \mathbf{0} \ldots \mathbf{0}\right] \qquad i = 1, \ldots, n \tag{4.1}$$

where $\mathbf{e}_{2j-1}$ denotes a unit column vector of dimension $2N$ with the unit component in position $2j-1$ and similarly for $\mathbf{e}_{2j}$.

It is thus possible to rewrite (2.11) and (3.6) as follows:

$$\mathbf{0} = \mathbf{D}\,\mathbf{V} - \sum_{i=1}^{n} \mathbf{C}_i \mathbf{x}_i = \mathbf{g}(\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{V}) \tag{4.2}$$

$$\mathbf{0} = \mathbf{f}_i(\mathbf{x}_i, \mathbf{z}_i, \mathbf{V}) \qquad i = 1, \ldots, n \tag{4.3}$$

At a given time step the Newton method is used to solve equations (4.2) and (4.3) with respect to $\mathbf{V}$ and $\mathbf{x}_i$. This requires solving a sequence of linear systems ($k = 1, 2, \ldots$):

$$\mathbf{J} \begin{bmatrix} \Delta\mathbf{x}_1 \\ \Delta\mathbf{x}_2 \\ \vdots \\ \Delta\mathbf{x}_n \\ \Delta\mathbf{V} \end{bmatrix} = - \begin{bmatrix} \mathbf{f}_1(\mathbf{x}_1^{k-1}, \mathbf{z}_1, \mathbf{V}^{k-1}) \\ \mathbf{f}_2(\mathbf{x}_2^{k-1}, \mathbf{z}_2, \mathbf{V}^{k-1}) \\ \vdots \\ \mathbf{f}_n(\mathbf{x}_n^{k-1}, \mathbf{z}_n, \mathbf{V}^{k-1}) \\ \mathbf{g}(\mathbf{x}_1^{k-1}, \ldots, \mathbf{x}_n^{k-1}, \mathbf{V}^{k-1}) \end{bmatrix} \tag{4.4}$$

where $\mathbf{J}$ is the Jacobian matrix:

$$\mathbf{J} = \begin{bmatrix} \mathbf{A}_1 & & & & \mathbf{B}_1 \\ & \mathbf{A}_2 & & & \mathbf{B}_2 \\ & & \ddots & & \vdots \\ & & & \mathbf{A}_n & \mathbf{B}_n \\ -\mathbf{C}_1 & -\mathbf{C}_2 & \ldots & -\mathbf{C}_n & \mathbf{D} \end{bmatrix} \tag{4.5}$$

and incrementing the variables according to:

$$\begin{aligned} \mathbf{x}_i^k &= \mathbf{x}_i^{k-1} + \Delta\mathbf{x}_i \qquad i = 1, \ldots, n \\ \mathbf{V}^k &= \mathbf{V}^{k-1} + \Delta\mathbf{V} \end{aligned}$$

In (4.5), $\mathbf{A}_i$ is the Jacobian of $\mathbf{f}_i$ with respect to $\mathbf{x}_i$, $\mathbf{B}_i$ the Jacobian of $\mathbf{f}_i$ with respect to $\mathbf{V}$ and the empty entries are zero sub-matrices. Because the i-th injector involves only the V_{xj} and V_{yj} components of $\mathbf{V}$, $\mathbf{B}_i$ is also a very sparse matrix with two nonzero columns only.

The convergence criteria (3.20), (3.21) and (3.22) are decomposed accordingly into:

$$||\mathbf{g}(\mathbf{x}_1^k, \ldots, \mathbf{x}_n^k, \mathbf{V}^k)||_\infty < \epsilon_g \tag{4.6}$$

$$||\mathbf{f}_i(\mathbf{x}_i^k, \mathbf{z}_i, \mathbf{V}^k)||_\infty < \epsilon_f \qquad i = 1, \ldots, n \tag{4.7}$$

$$\left| \frac{(\Delta\mathbf{x}_i^k)_j}{(\mathbf{x}_i^k)_j} \right| < \epsilon_\Delta \qquad j = 1, \ldots, n_i \,; \quad i = 1, \ldots, n \tag{4.8}$$

When applied to very large systems, the above standard Newton scheme suffers from three drawbacks:

- when a discrete variable z_i changes in any injector, for instance due to a switching or a limit on a state variable, $\mathbf{A}_i$ and $\mathbf{B}_i$ change and the whole Jacobian $\mathbf{J}$ has to be updated and factorized;

- the same holds true when a few injectors (or the network) undergo large changes and trigger an update of the whole Jacobian **J** as indicated above;
- the fact that many injectors having "little activity" yield very small components in the right hand side of (4.4) is not exploited.

To tackle these issues, it is convenient to solve (4.4) in a decomposed manner. The decomposition of concern in this work can be seen as a Gaussian elimination with pivoting on the block of equations and unknowns relative to injectors.

One easily obtains from (4.4-4.5):

$$\mathbf{A}_i \Delta \mathbf{x}_i = -\mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{z}_i, \mathbf{V}^{k-1}) - \mathbf{B}_i \Delta \mathbf{V} \tag{4.9}$$

Assuming that $\mathbf{A}_i$ is nonsingular, $\Delta \mathbf{x}_i$ can be obtained from this equation and replaced into the last row of (4.4), which yields:

$$\sum_{i=1}^{n} \mathbf{C}_i \mathbf{A}_i^{-1} \left(\mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{z}_i, \mathbf{V}^{k-1}) + \mathbf{B}_i \Delta \mathbf{V} \right) + \mathbf{D} \Delta \mathbf{V} = -\mathbf{g}(\mathbf{x}_1^{k-1}, \ldots, \mathbf{x}_n^{k-1}, \mathbf{V}^{k-1}) \tag{4.10}$$

Reorganizing the terms in (4.10) yields:

$$\tilde{\mathbf{D}}\, \Delta \mathbf{V} = -\mathbf{g}(\mathbf{x}_1^{k-1}, \ldots, \mathbf{V}^{k-1}) - \sum_{i=1}^{n} \mathbf{C}_i \mathbf{A}_i^{-1} \mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{z}_i, \mathbf{V}^{k-1}) \tag{4.11}$$

where:

$$\tilde{\mathbf{D}} = \mathbf{D} + \sum_{i=1}^{n} \mathbf{C}_i \mathbf{A}_i^{-1} \mathbf{B}_i \tag{4.12}$$

is the so-called Schur complement [Zha05] of $diag(\mathbf{A}_1 \ldots \mathbf{A}_n)$. At this point, it is convenient to define:

$$\tilde{\mathbf{C}}_i = \mathbf{C}_i \mathbf{A}_i^{-1} \tag{4.13}$$

and rewrite (4.11) and (4.12) respectively as:

$$\tilde{\mathbf{D}}\, \Delta \mathbf{V} = -\mathbf{g}(\mathbf{x}_1^{k-1}, \ldots, \mathbf{V}^{k-1}) - \sum_{i=1}^{n} \tilde{\mathbf{C}}_i \mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{z}_i, \mathbf{V}^{k-1}) \tag{4.14}$$

$$\text{with} \qquad \tilde{\mathbf{D}} = \mathbf{D} + \sum_{i=1}^{n} \tilde{\mathbf{C}}_i \mathbf{B}_i \tag{4.15}$$

A variant consists of replacing the computation of the right-hand side of (4.9) by a single $\mathbf{f}_i$ evaluation. Indeed:

$$\begin{aligned} \mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{V}^{k-1}) \quad &+ \quad \mathbf{B}_i \Delta \mathbf{V} \\ &= \quad \mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{V}^{k-1}) + \frac{\partial \mathbf{f}_i}{\partial \mathbf{V}} (\mathbf{V}^k - \mathbf{V}^{k-1}) \\ &\simeq \quad \mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{V}^k) \end{aligned} \tag{4.16}$$

where the last but one expression has been considered as a Taylor series expansion of the last one, limited to the first order. In principle, the latter approximation will result in $\Delta \mathbf{x}_i$ values a bit different from the ones obtained by solving the original system (4.4): however, since this variant affects only the computation of the Newton correction, while the convergence tests (4.6-4.8) keep on being used, the accuracy is the same as in the original Newton method. On the other hand, one may expect that evaluating $\mathbf{f}_i$ with the newly updated $\mathbf{V}^k$ as in (4.16) yields an even more satisfactory iterative scheme. Indeed, experience shows that while it requires an additional function evaluation per iteration, using (4.16) yields an overall faster simulation than using (4.9). The convergence speed is particularly enhanced at the first iteration after a short circuit: in this case, $\mathbf{V}^k$ is far from $\mathbf{V}^{k-1}$, and, as one may expect, the $\mathbf{f}_i$ function may not be approximated in a satisfactory way by a first-order truncation of its Taylor series expansion.

It must be emphasized that the correction $\tilde{\mathbf{C}}_i \mathbf{B}_i$ brought by the i-th injector only involves a (2×2) sub-matrix centered on the diagonal of $\mathbf{D}$. Hence $\tilde{\mathbf{D}}$ inherits the structural symmetry of $\mathbf{D}$.

Assuming that the $\mathbf{A}_i$ and $\tilde{\mathbf{D}}$ matrices are available in factorized form, and the corresponding $\tilde{\mathbf{C}}_i$ matrices have been computed, the procedure to solve the k-th Newton iteration in a decomposed manner consists of:

1. solving (4.14) with respect to $\Delta \mathbf{V}$ to obtain $\mathbf{V}^k$

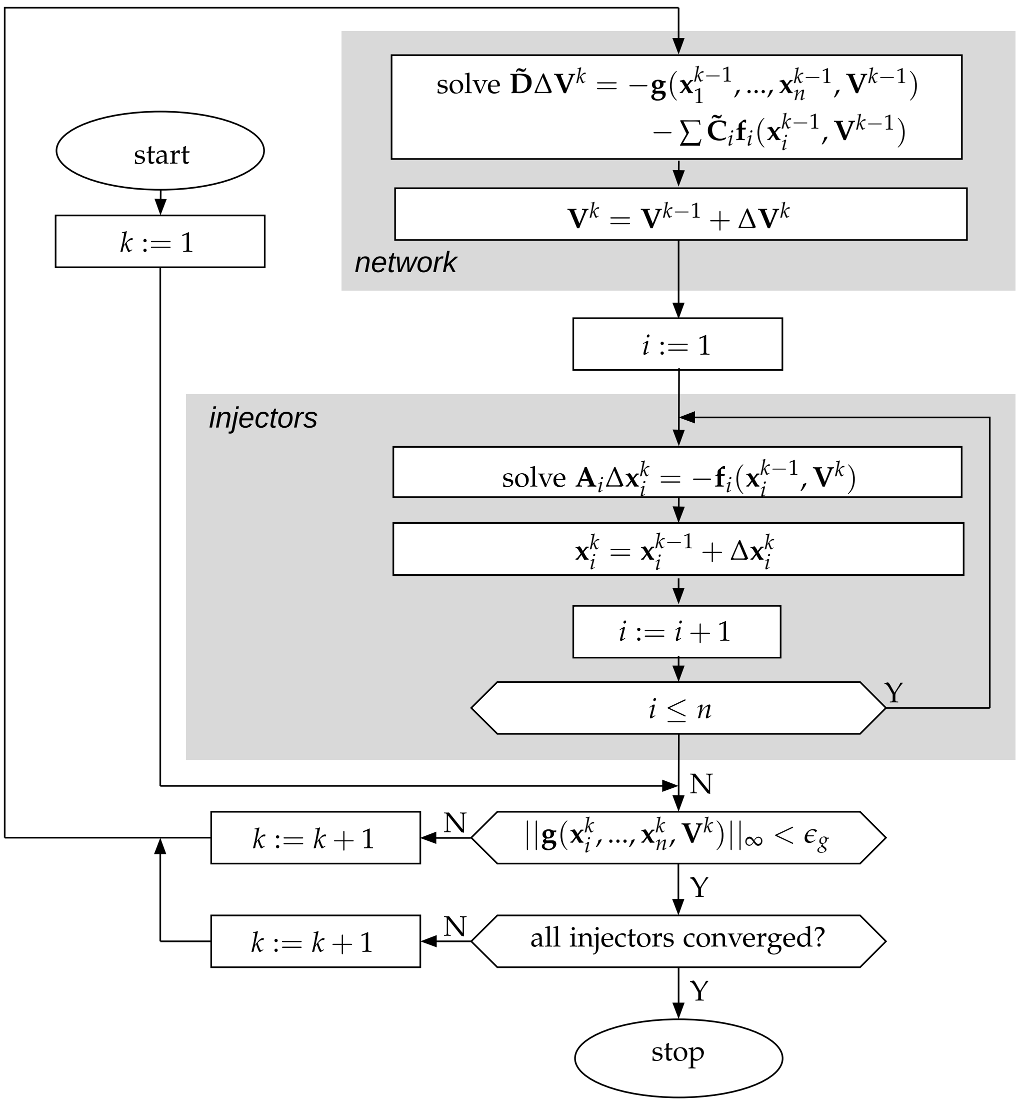
2. solving (4.9) with respect to $\Delta \mathbf{x}_i$ to obtain $\mathbf{x}_i^k$ for each injector.

Let us emphasize that these two steps are mathematically equivalent to solving the original system (4.4). The corresponding flow-chart is shown in Fig. 4.3.

The original system has been decomposed into $n+1$ systems with matrices $\tilde{\mathbf{D}}$ and $\mathbf{A}_i$ ($i = 1, \ldots, n$) respectively. Efficient sparse solvers are widely available for solving the large, sparse and structurally symmetric system (4.14) with matrix $\tilde{\mathbf{D}}$. For the $\mathbf{A}_i$ matrices, the decision whether to use a sparse solver or a dense one, depends on their size, which impacts the gain to be expected from sparsity.

We finally consider the computation of $\tilde{\mathbf{C}}_i$. Equation (4.13) can be rewritten as:

$$\mathbf{A}_i^T \, \tilde{\mathbf{C}}_i^T = \mathbf{C}_i^T$$



start
$k := 1$
network
solve $\tilde{\mathbf{D}}\Delta\mathbf{V}^k = -\mathbf{g}(\mathbf{x}_1^{k-1}, ..., \mathbf{x}_n^{k-1}, \mathbf{V}^{k-1}) - \sum \tilde{\mathbf{C}}_i \mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{V}^{k-1})$
$\mathbf{V}^k = \mathbf{V}^{k-1} + \Delta\mathbf{V}^k$
$i := 1$
injectors
solve $\mathbf{A}_i \Delta\mathbf{x}_i^k = -\mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{V}^k)$
$\mathbf{x}_i^k = \mathbf{x}_i^{k-1} + \Delta\mathbf{x}_i^k$
$i := i + 1$
$i \le n$
Y
N
$||\mathbf{g}(\mathbf{x}_i^k, ..., \mathbf{x}_n^k, \mathbf{V}^k)||_\infty < \epsilon_g$
N
$k := k + 1$
Y
all injectors converged?
N
$k := k + 1$
Y
stop

Figure 4.3: Decomposed Newton scheme

Since $\mathbf{C}_i^T$ has only two nonzero columns, so has $\tilde{\mathbf{C}}_i^T$. Those columns are the solutions $\tilde{\mathbf{x}}_1$,$\tilde{\mathbf{x}}_2$ of:

$$\mathbf{A}_i^T \tilde{\mathbf{x}}_1 = \mathbf{e}_1 \qquad \mathbf{A}_i^T \tilde{\mathbf{x}}_2 = \mathbf{e}_2$$

where $\mathbf{e}_1$ (respectively $\mathbf{e}_2$) is a unit column vector of dimension n_i with the unit component in first (respectively second) position.

4.3 Skipping solution of converged components

It is time now to show the real advantages brought by the decomposed Newton scheme. They lie in the possibility to skip a significant amount of computations by:

- not updating the state vector of an injector when it has already converged
- not updating the network voltages if the corresponding mismatches fall below the tolerance.

Assume that solving (4.2, 4.3) at a given time step requires E Newton iterations, i.e. E executions of the outer loop in Fig. 4.3. The idea is, for injectors with low dynamic response, to perform a lower number e of iterations ($e < E$).

Let us recall that each injector state vector is corrected at least once, in order the convergence test (4.8) to be checked. Suppose that this test is indeed passed after e iterations ($e \geq 1$). It can be concluded that the injector mismatch just before making the last correction, i.e. $||\mathbf{f}_i(\mathbf{x}_i^{e-1}, \mathbf{z}_i, \mathbf{V}^{e-1})||_\infty$ had already a negligible value.

From there on, we merely check that the injector remains converged. Furthermore, this check does no longer involve the correction vector $\Delta \mathbf{x}_i$, according to (4.8), but the mismatch vector, according to (4.7), where ϵ_f has been set to $||\mathbf{f}_i(\mathbf{x}_i^{e-1}, \mathbf{z}_i, \mathbf{V}^{e-1})||_\infty$.

The overall procedure can be summarized as follows: recognizing the difficulty to specify *a priori* a value for ϵ_f we resort to the test (4.8) on the correction vector. Once the latter is satisfied, we use the mismatch at the last but one iteration as the value for ϵ_f, and check convergence through the test (4.7) on the mismatch vector.

Note that it is possible (although not common, according to our experience) for an injector to have the convergence test (4.8) passed at some iteration and the inequality (4.7) violated at a subsequent iteration, in which case (4.9) is solved again.

A similar treatment is applied to the network: if the convergence test (4.6) is satisfied, Eq. (4.14) is not solved.

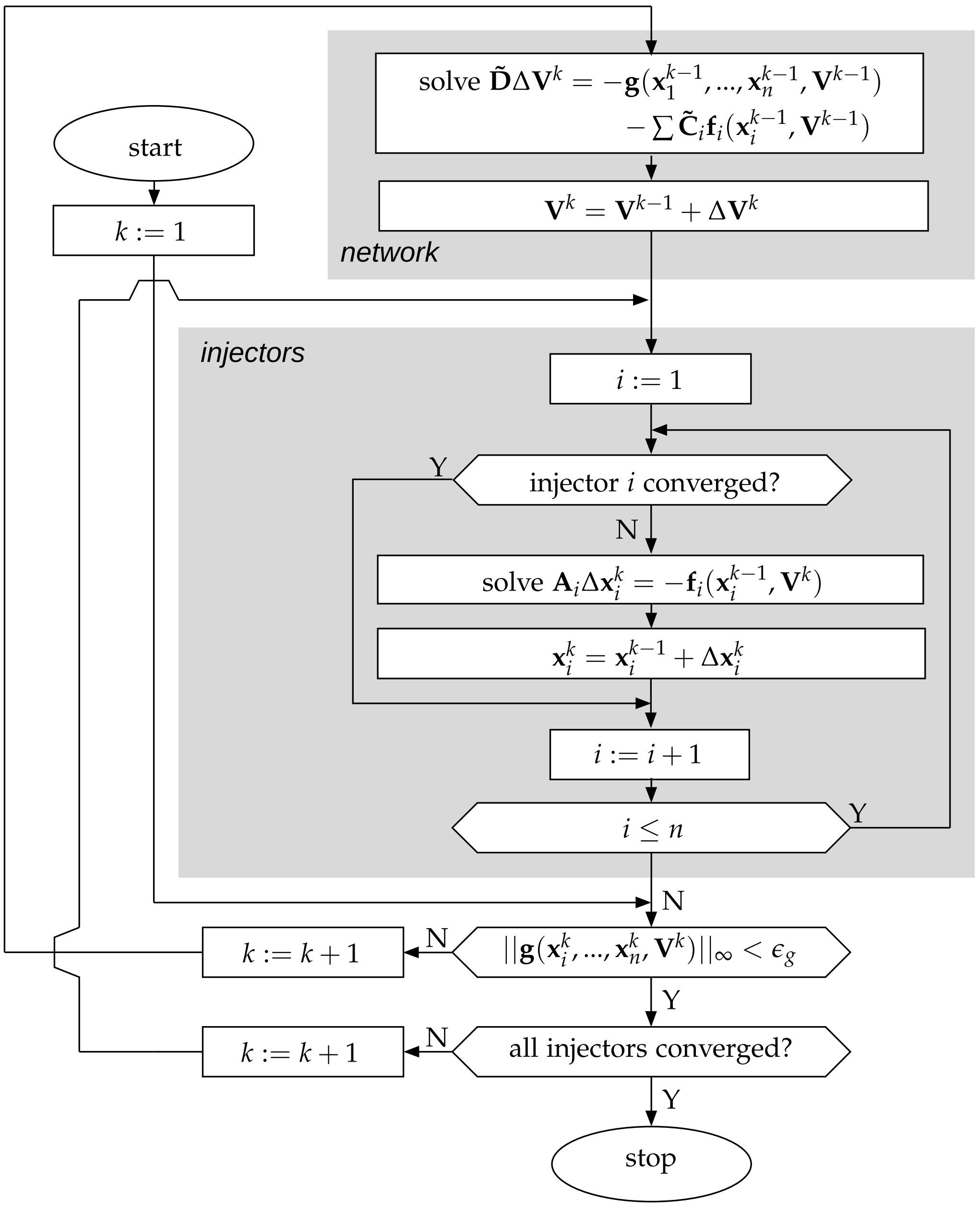

solve $\tilde{\mathbf{D}}\Delta\mathbf{V}^k = -\mathbf{g}(\mathbf{x}_1^{k-1},...,\mathbf{x}_n^{k-1},\mathbf{V}^{k-1}) - \sum\tilde{\mathbf{C}}_i\mathbf{f}_i(\mathbf{x}_i^{k-1},\mathbf{V}^{k-1})$
start
$\mathbf{V}^k = \mathbf{V}^{k-1} + \Delta\mathbf{V}^k$
$k := 1$
network
injectors
$i := 1$
Y
injector i converged?
N
solve $\mathbf{A}_i\Delta\mathbf{x}_i^k = -\mathbf{f}_i(\mathbf{x}_i^{k-1},\mathbf{V}^k)$
$\mathbf{x}_i^k = \mathbf{x}_i^{k-1} + \Delta\mathbf{x}_i^k$
$i := i + 1$
$i \leq n$
Y
N
$k := k + 1$
N
$||\mathbf{g}(\mathbf{x}_i^k,...,\mathbf{x}_n^k,\mathbf{V}^k)||_\infty < \epsilon_g$
Y
$k := k + 1$
N
all injectors converged?
Y
stop

Figure 4.4: Decomposed Newton scheme with skipping of converged components

Note that no attempt is made to exploit the presence of negligible components in the right-hand side vector of (4.14). Indeed, this would require restructuring matrix $\tilde{\mathbf{D}}$ during the simulation, while it is very convenient to perform optimal ordering of this matrix once for all before the simulation starts. However, if the solver handling the sparse system (4.14) can take advantage of those negligible components, additional computational efficiency is to be expected.

The corresponding flowchart is shown in Fig. 4.4 and it should be compared with the one shown in Fig. 4.3.

4.4 Skipping update of Jacobian matrices

Another important saving in the Newton method consists of keeping the Jacobian constant over several iterations, as seen in Section 3.2, which can be advantageously combined with the decomposition.

As already mentioned, the Jacobians $\tilde{\mathbf{D}}$ and $\mathbf{A}_i$ ($i = 1, \dots, n$) should be kept constant over as many Newton iterations and time steps as possible. On the other hand, the Jacobians should be updated when there is a degradation of convergence (detected from insufficiently decreasing mismatches) or following events such as the change in a $\mathbf{z}_i$ component of an injector or a modification of the network topology (e.g. fault inception and clearing).

In a strict implementation of the Newton method, after updating the Jacobian $\mathbf{A}_i$ of the i-th injector, the $\tilde{\mathbf{D}}$ matrix should be also updated and factorized owing to the presence of the $\mathbf{C}_i\mathbf{A}_i^{-1}\mathbf{B}_i$ term in (4.12). For the same reason, when $\tilde{\mathbf{D}}$ has to be updated, the $\mathbf{A}_i$ matrices should be computed and factorized in order to bring the latest available $\mathbf{C}_i\mathbf{A}_i^{-1}\mathbf{B}_i$ terms in (4.12).

In practice, however, extensive tests have shown that it is possible to update the $\tilde{\mathbf{D}}$ and $\mathbf{A}_i$ matrices independently of each other. More precisely:

1. when the i-th injector calls for an update of $\mathbf{A}_i$, this matrix is recomputed and factorized. The $\tilde{\mathbf{C}}_i$ matrix must be also recomputed since it is used in (4.14). However,

   - *the other $\mathbf{A}_\ell$ matrices ($\ell \neq i$) are not updated.* This is exact, since an injector matrix is completely decoupled from the other injector matrices;

   - *the $\tilde{\mathbf{D}}$ matrix is not updated.* This comes from the fact that changes in injector equations usually have a moderate impact on network equations. In fact, it can be shown that the correction $\tilde{\mathbf{C}}_i \mathbf{B}_i$ brought by the i-th injector is the sensitivity of the current injection to the bus voltage. In general, the latter only slightly changes with operating conditions;

2. when the network calls for a Jacobian update, the $\mathbf{B}_i$ matrices of all injectors are updated and $\tilde{\mathbf{D}}$ is computed from (4.15) using the available $\tilde{\mathbf{C}}_i$ matrices. *However, neither the $\mathbf{A}_i$ nor the $\tilde{\mathbf{C}}_i$ matrices are updated.* This is a compromise between accuracy and efficiency: indeed updating all $\mathbf{B}_i$ is inexpensive, while updating and refactorizing all $\mathbf{A}_i$ matrices would be cumbersome.

In the results shown in this chapter it will be shown how the decomposed Newton scheme with skipping solution of converged components (with corresponding flowchart shown in Fig. 4.4) and update of Jacobian matrices will yield the highest speed-up without sacrificing the accuracy of the simulation. Further speed-ups, as shown with more detail in Chapters 5 and 6, will come at the cost of reduced accuracy.

4.5 Extension to two- and multi-port components

For simplicity's sake, the decomposition of Newton iteration detailed in this chapter was assuming one-port injectors, i.e. power system components which are connected to one bus only. This section deals with the extension to two- and multi-port injectors.

Without loss of generality, we assume that one of the $\bar{n}$ two-port elements in Fig. 4.5 links the origin bus m with the extremity bus p, and that the first four (out of n_i) components of $\mathbf{x}_i$ are I^m_{xi}, I^m_{yi}, I^p_{xi} and I^p_{yi}, the rectangular components of the complex currents injected into the network at the origin and at the extremity of the two-port injector:

$$\mathbf{x}_i = [I^m_{xi}\; I^m_{yi}\; I^p_{xi}\; I^p_{yi}\; \ldots]^T$$

$\mathbf{C}_i$ aims at extracting the I^m_{xi}, I^m_{yi}, I^p_{xi} and I^p_{yi} components of $\mathbf{x}_i$ and is adapted to the above state vector:

$$\mathbf{C}_i = \left[\mathbf{e}_{2m-1}\; \mathbf{e}_{2m}\; \mathbf{e}_{2p-1}\; \mathbf{e}_{2p}\; \mathbf{0} \ldots \mathbf{0}\right] \tag{4.17}$$

where $\mathbf{e}_{2m-1}$ denotes a unit vector of dimension n_i with the unit component in position $2m-1$ and similarly for $\mathbf{e}_{2m}$,$\mathbf{e}_{2p-1}$ and $\mathbf{e}_{2p}$.

$\mathbf{B}_i$, the Jacobian of $\mathbf{f}_i$ with respect to $\mathbf{V}$ is changed as well, insofar as it is composed of four nonzero columns instead of two.

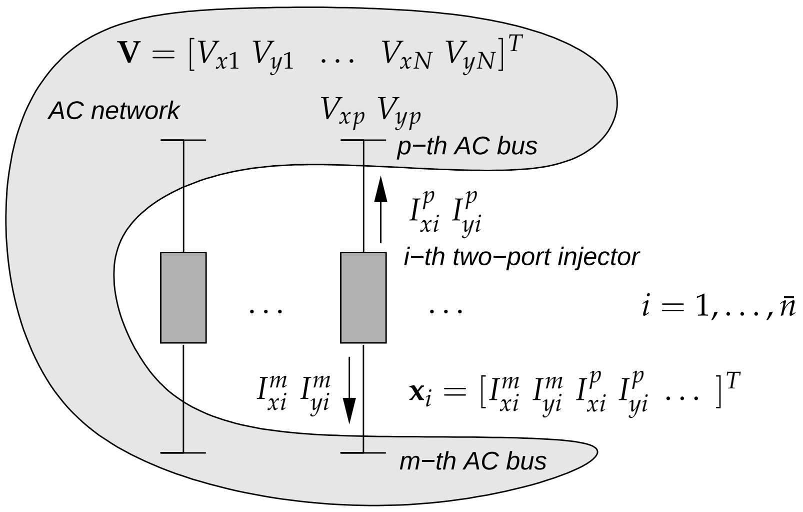


Figure 4.5: Extension to two-port injectors

The Schur complement procedure is then the same as in the one-port case. The correction $\tilde{\mathbf{C}}_i\mathbf{B}_i$ brought by the i-th injector into (4.15) only involves a (4×4) sub-matrix centered on the diagonal of $\mathbf{D}$. In the one-port case the correction was added to four elements of $\mathbf{D}$ which already existed: a shunt admittance at the bus of concern would bring one (2×2) correction at the same location. In the general case, however, the two-port injector brings two (2×2) corrections on the diagonal (2×2) elements of the origin and the extremity bus, along with two (2×2) *off-diagonal* corrections, which would correspond to the Jacobian elements brought by a network element connecting the same buses. Thus, $\tilde{\mathbf{D}}$ is structurally symmetric also in the two-port case.

Two-port injectors allow to include black-box models of network elements in the general BBD framework. It is the case, for instance, of HVDC links, that fit the general formulation of Fig. 4.5, including the HVDC line itself as well as the converters and their controllers.

While the extension to $\bar{n}$-port, for $\bar{n}$ arbitrary, is feasible, it is not useful. For instance, a multi-terminal HVDC system connected to $\bar{n}$ AC buses can be decomposed into a set of $\bar{n}$ two-port injectors, including the ACDC converters only, together with a purely DC network including $\bar{N}$ DC buses and a number of DC lines, as sketched in Fig. 4.6.

To fit the general framework of phasor formulation, in which each voltage takes values in $\mathbb{R}^2$, a fictitious second component of the voltage pair is introduced and set to zero for each DC voltage. The same applies for current components. Within this framework, AC voltage vector $\mathbf{V}$ and DC voltage vector $\mathbf{V}^{DC}$ can be merged into a single vector and solved together. It should be remarked that, as Fig. 4.6 shows, the resulting network is not connected.

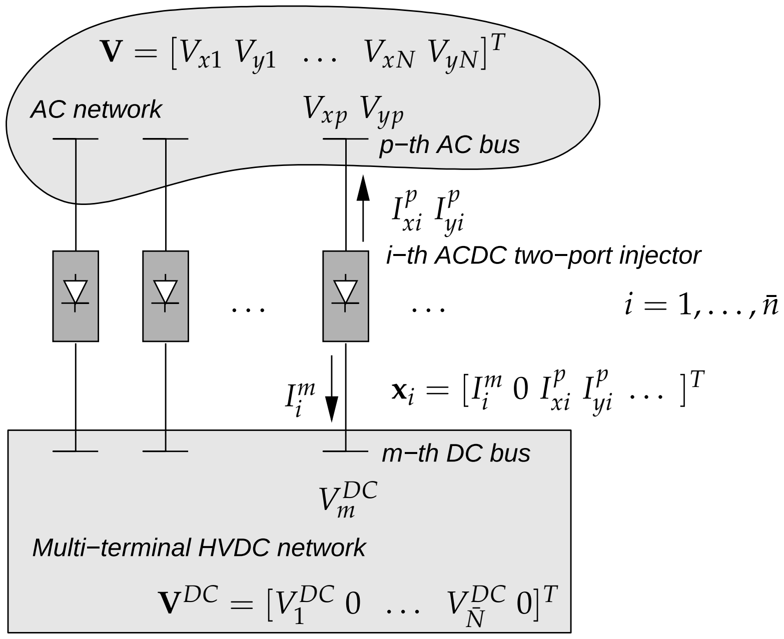

$\mathbf{V} = [V_{x1}\ V_{y1}\ \ldots\ V_{xN}\ V_{yN}]^T$
AC network
$V_{xp}\ V_{yp}$
p−th AC bus
$I^p_{xi}\ I^p_{yi}$
i−th ACDC two−port injector
$i = 1, \ldots, \bar{n}$
I^m_i
$\mathbf{x}_i = [I^m_i\ 0\ I^p_{xi}\ I^p_{yi}\ \ldots\ ]^T$
m−th DC bus
V^{DC}_m
Multi−terminal HVDC network
$\mathbf{V}^{DC} = [V^{DC}_1\ 0\ \ldots\ V^{DC}_{\bar{N}}\ 0]^T$

Figure 4.6: Extension to multi-terminal HVDC networks

4.6 Scenari and results on Nordic32 system

This section gives details about the scenari that are simulated on the Nordic32 test system, described in Section 2.4. The reference outputs have been obtained with an Integrated dishonest Newton scheme (denoted as "I" in the plots). This section also provides the results obtained by applying decomposition to the Nordic32 test system, on the whole set of the proposed scenari.

Outputs have been obtained also with an accelerated decomposed dishonest Newton scheme (denoted "A" in the plots) that makes use of the acceleration techniques proposed in Sections 4.3 and 4.4. More details about this and the other solvers are given in Appendix A.5. All evolutions obtained with scheme A are almost undistinguishable from those of the benchmark scheme I. DTW distances between the outputs of scheme A and I (the latter being the reference) are reported in Table 4.1. This section serves as a validation of the techniques explained in this chapter; relevant results concerning the speed-up obtained, while preserving accuracy, are detailed in the next sections.

4.6.1 Case N1

Case N1 is a line-tripping scenario. One circuit of line 1013-1014, located in the North area (see Fig. 2.4), is tripped at $t = 1$ s. This short-term scenario is simulated until $t = 30$ s, when most of the dynamics have died out.

The variables of interest for this case are:

- active power flow in the other circuit of line 1013-1014, taking over the tripped circuit (Fig. 4.7);
- field voltage of generator g15, located in the Central area (Fig. 4.8).

The trajectories of output A with respect to output I are indistinguishable as confirmed by the DTW distances shown in Table 4.1.

The differences between the outputs are within the chosen tolerance and can be appreciated only when very small variations are plotted, such as in Fig. 4.8, that shows the field voltage of generator g15, only marginally excited by the disturbance. An almost imperceptible difference is the slightly higher damping provided by scheme A, but let us highlight the very small range of the over-damped oscillation: the corresponding DTW distances, in Table 4.1, are in fact negligible.

4.6.2 Case N2

Case N2 is another line-tripping scenario: line 4043-4047, in the Central area, is tripped at $t = 1$ s. The effect of this disturbance is stronger than in Case N1, leading to some long-term dynamics, namely two LTC moves restoring the voltages at buses 4043 and 4046. The simulation is stopped at $t = 75$ s, when the system approaches steady state.

The variables of interest for this case are:

- voltage at the distribution bus fed by 4043, which evolves under the effect of the line outage and the LTC moves (Fig. 4.9);
- field voltage of and active power produced by generator g15, located in the Central area, very close to the disturbance (Fig. 4.10 and 4.11, respectively).

4.6.3 Case N3

Case N3 is an unstable scenario initiated by the tripping of line 4032-4044, which is a crucial element of the North-Central corridor, at $t = 1$ s. After some short-term oscillations, the system evolves under the combined effects of the field current limitation imposed by several

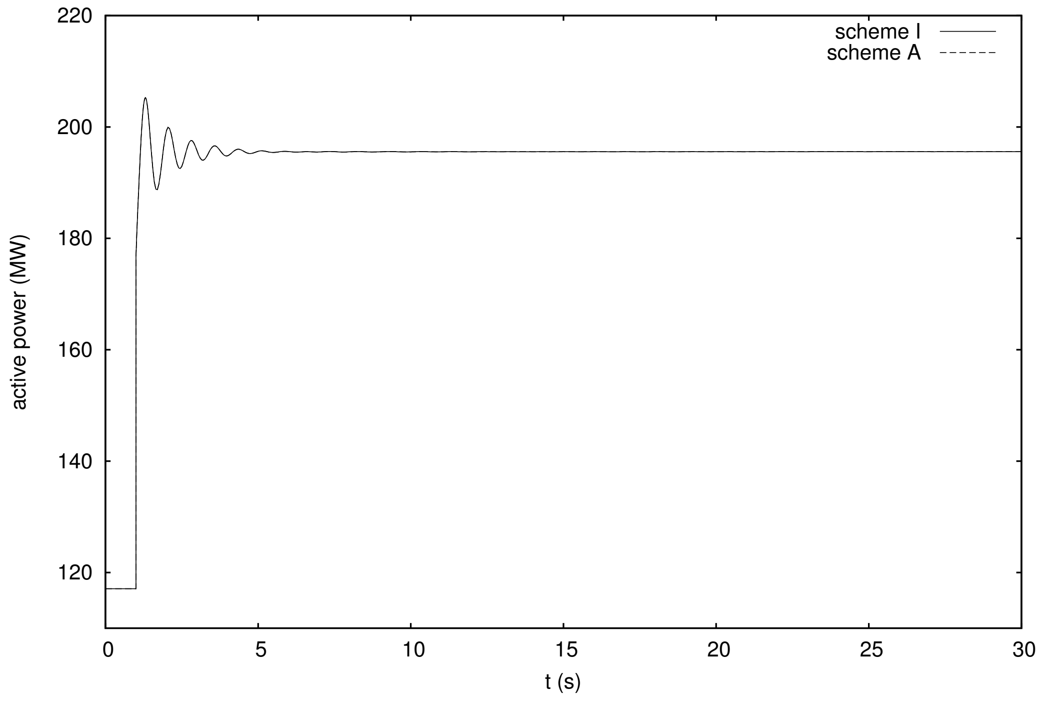


Figure 4.7: Case N1: active power flow in one circuit of line 1013-1014

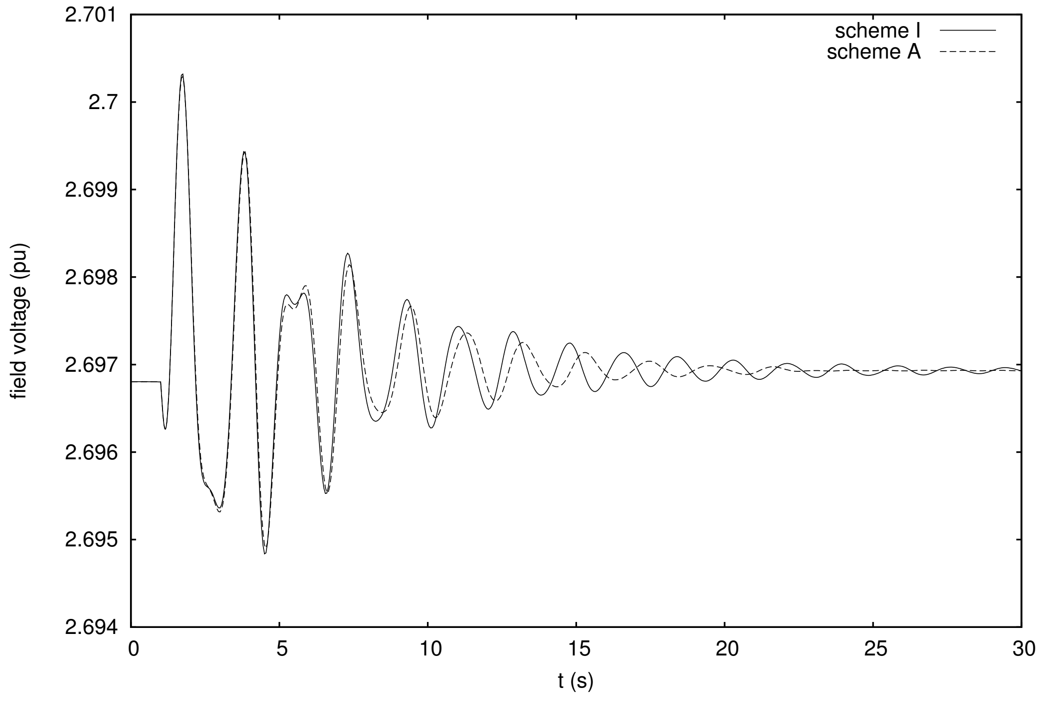


Figure 4.8: Case N1: field voltage of generator g15

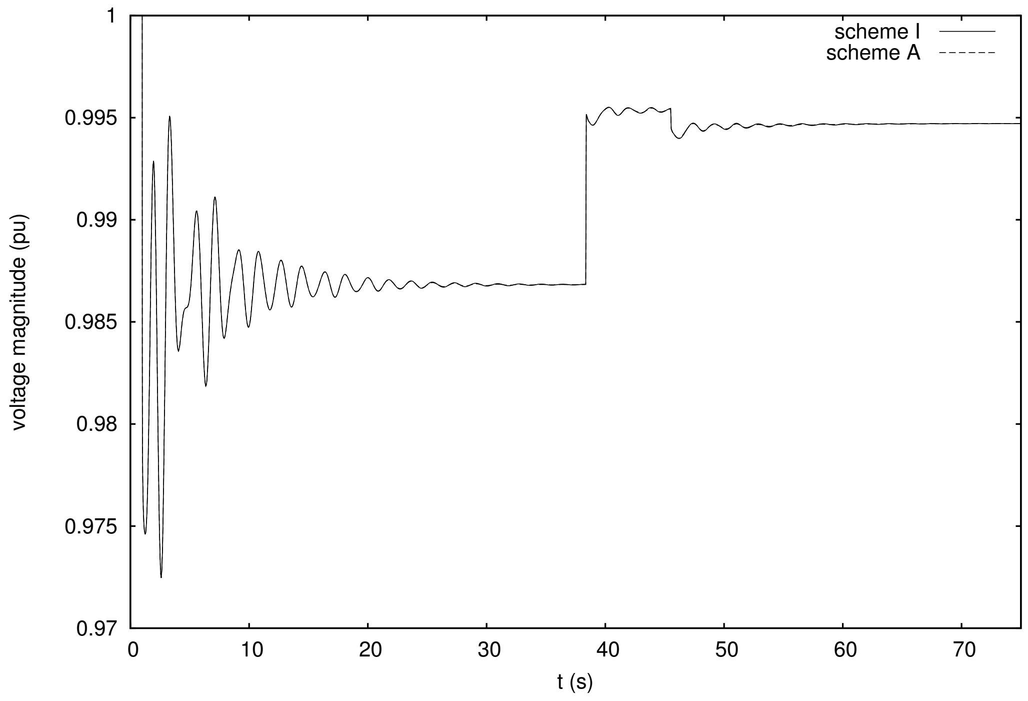


Figure 4.9: Case N2: voltage at distribution bus fed by bus 4043

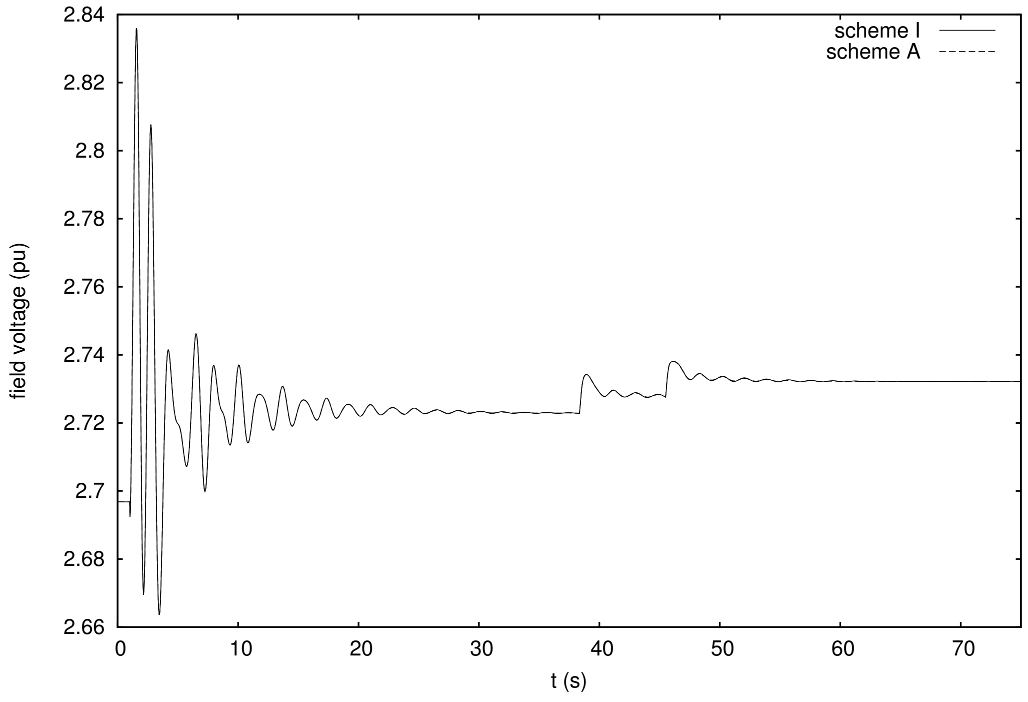


Figure 4.10: Case N2: field voltage of generator g15

OELs and the LTCs trying to restore the distribution voltage. The trajectory is computed until the collapse of the system, which takes place short before $t = 180$ s.

The variables of interest for this case are:

- voltage at transmission bus 1041, which shows the voltage evolution towards collapse in the Central area (Fig. 4.12);
- speed deviation of generator g6, closely located to the initiating disturbance. This machine has its field current limited at around $t = 113$ s and loses synchronism towards the end of the simulation (Fig. 4.13). In this figure, the speed deviation of generator g20 is also plotted in order to assess the loss of synchronism.

4.6.4 Case N4

Case N4 is an oscillatory unstable scenario. In normal conditions, three synchronous machines are equipped with power system stabilizers that provide damping torque even after a wide variety of disturbances. For this case only, the three power system stabilizers are disabled in order to obtain an oscillatory unstable evolution. The tripping of line 4061-4062, at $t = 1$ s, triggers an oscillation in the South area which is simulated until the collapse of the system, which takes place short before $t = 25$ s.

Case N4 allows to observe other small differences between the output A and the reference I, which may happen when the system is under stress. For instance, Fig. 4.14 shows the field voltage of generator g17, strongly impacted by the disturbance. A small discrepancy can be observed around $t = 21$ s, in the form of a slight delay.

Figure 4.15 provides the speed deviation of generator g17, directly affected by the initiating disturbance, which loses synchronism towards the end of the simulation.

4.6.5 Case N5

Case N5 adds to the disturbance considered in case N3 a short circuit at bus 4032 lasting from $t = 0.9$ s to $t = 1$ s. The long-term evolution is then similar to the one in case N3.

The variable of interest for this case is the voltage at transmission bus 1041, which shows the voltage evolution towards collapse in the Central area (Fig. 4.16).

Table 4.1 also gives the DTW distances between the outputs of scheme A and I (the latter being the reference). It further confirms that the accelerated decomposed dishonest

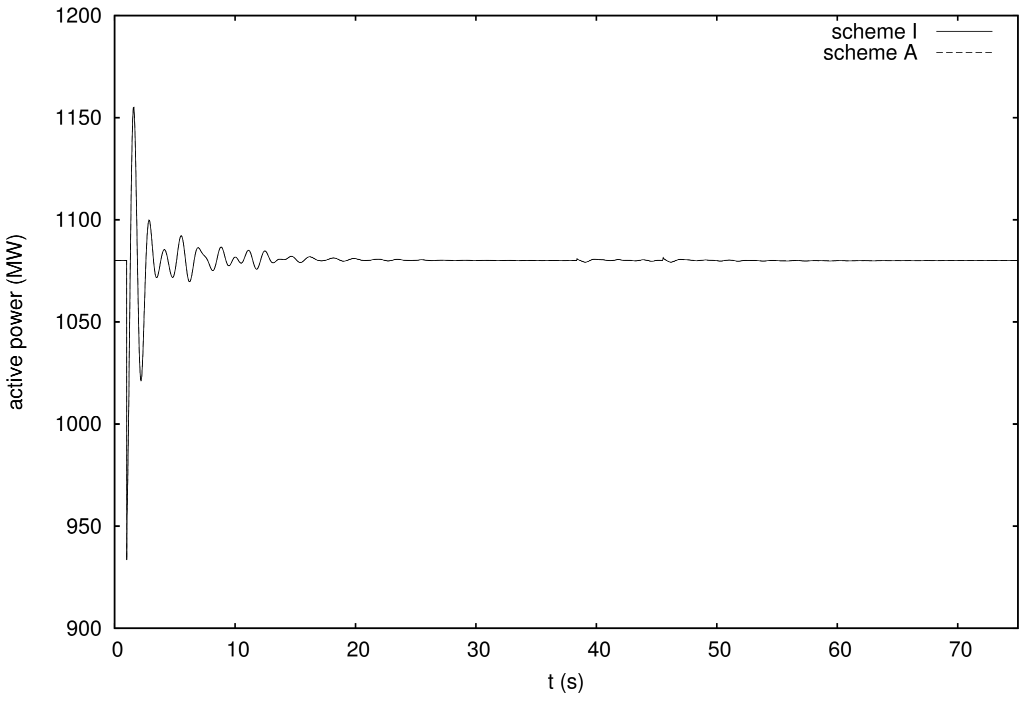


Figure 4.11: Case N2: active power produced by generator g15

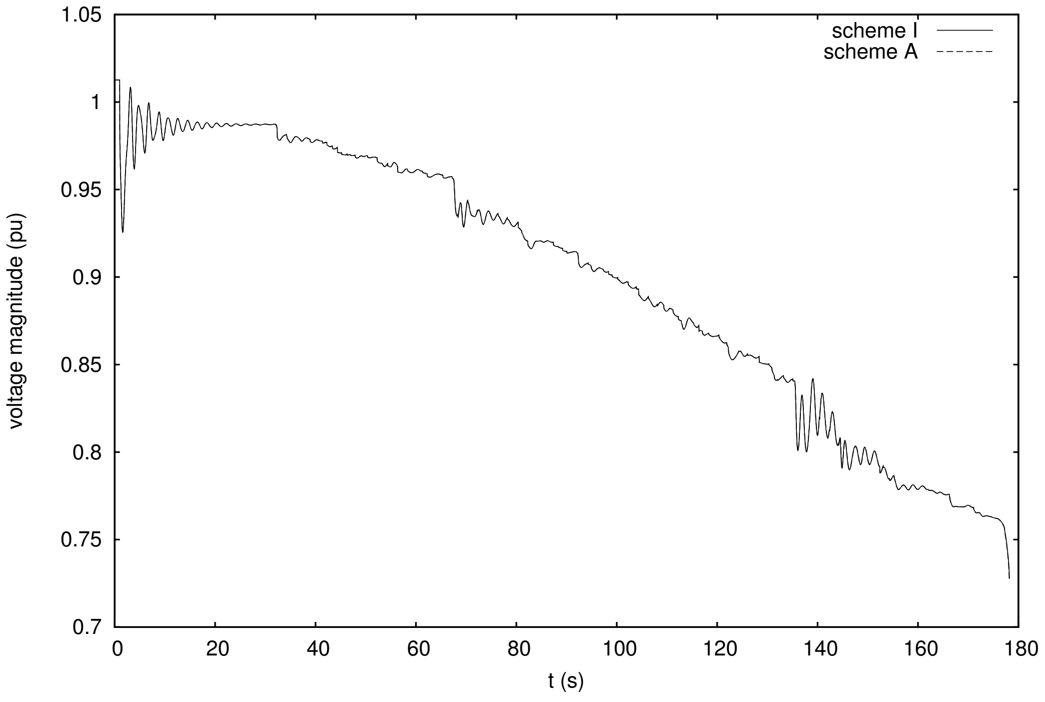


Figure 4.12: Case N3: voltage at transmission bus 1041

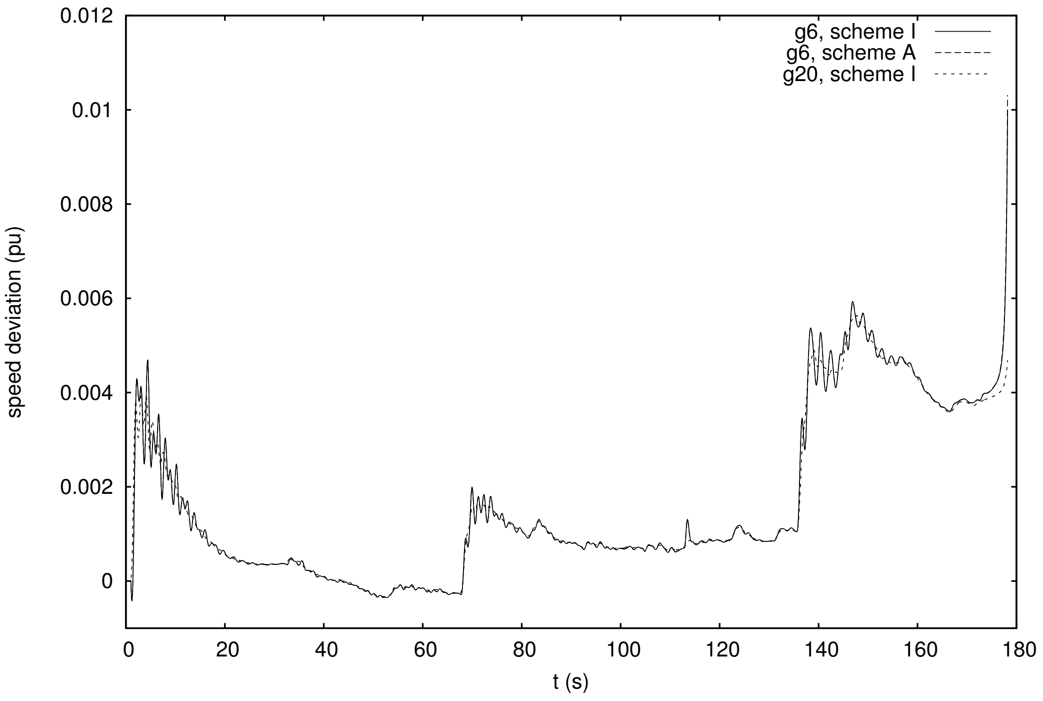


Figure 4.13: Case N3: synchronous machine speed of generator g6

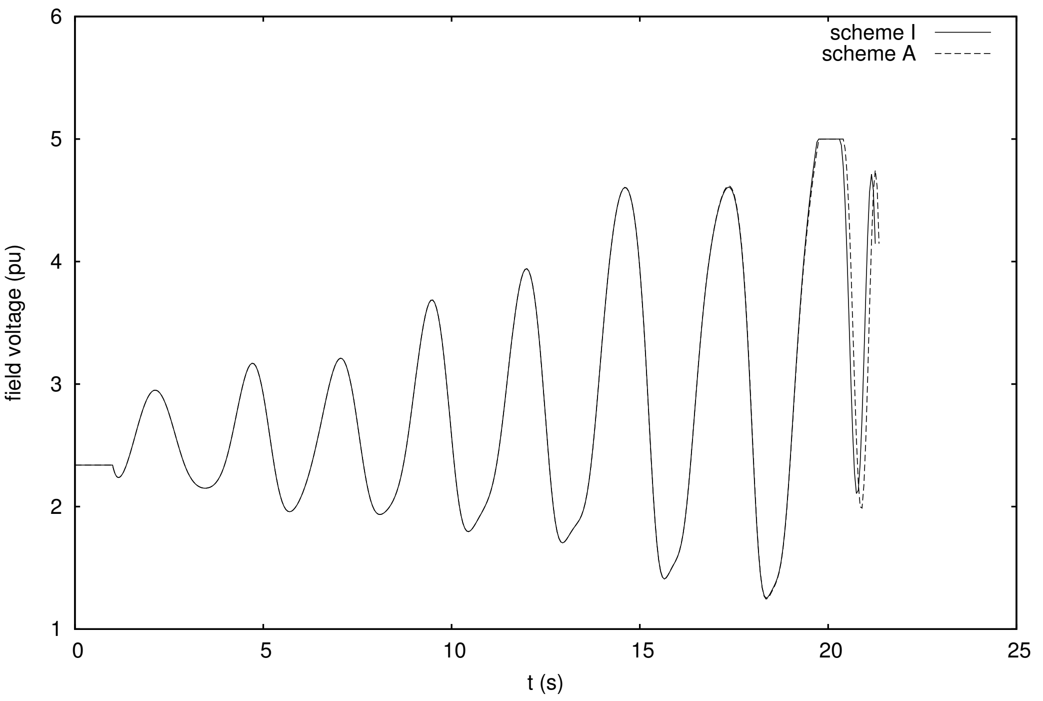


Figure 4.14: Case N4: field voltage of generator g17

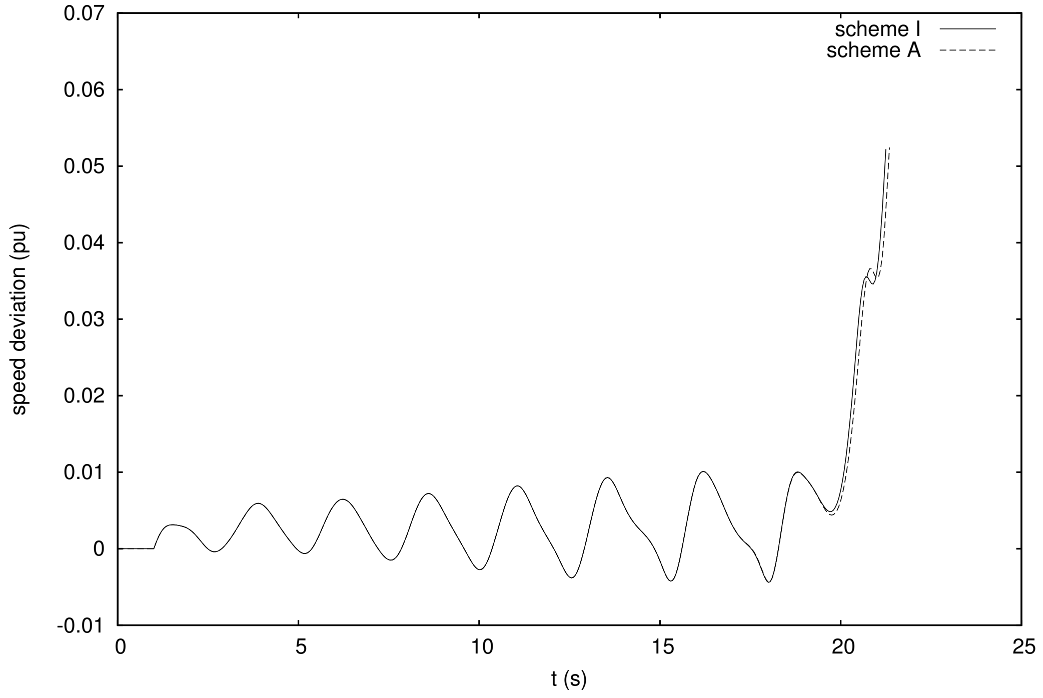


Figure 4.15: Case N4: synchronous machine speed of generator g17

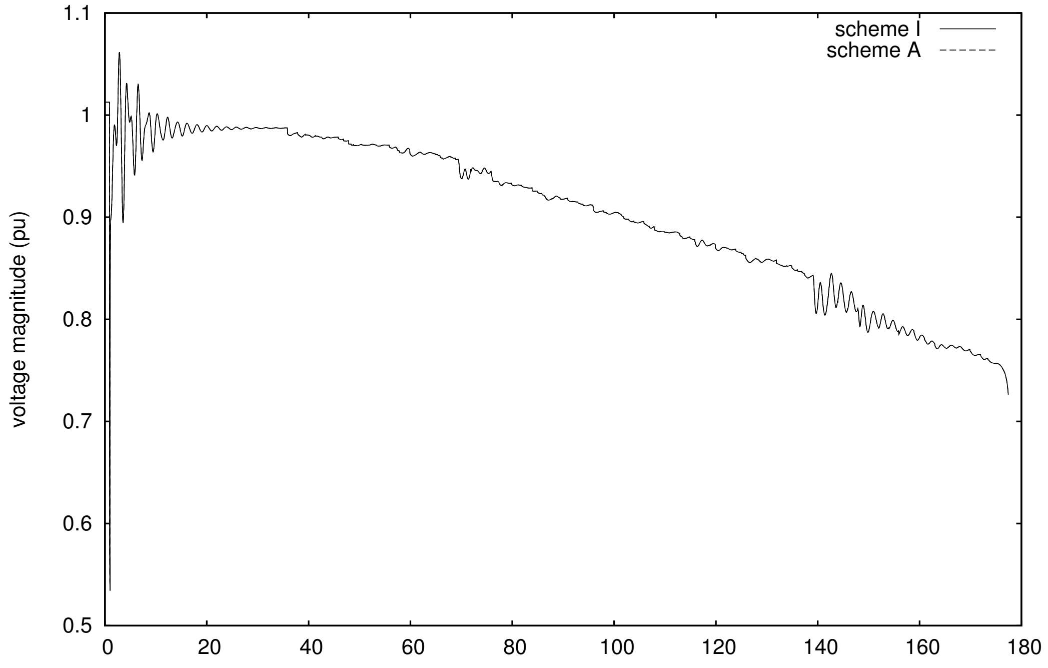


Figure 4.16: Case N5: voltage at transmission bus 1041

Newton scheme A (with skipping solution of converged components and update of Jacobian matrices) preserves the accuracy of the simulated outputs, when compared to the integrated scheme I.

Table 4.1: DTW distances

Fig.	δ	μ	σ	Unit
4.7	0.0041	0.0001	0.0041	MW
4.8	0.0001	0.0000	0.0001	pu
4.9	0.0000	-0.0000	0.0000	pu
4.10	0.0001	0.0000	0.0001	pu
4.11	0.0182	-0.0009	0.0181	MW
4.12	0.0001	0.0000	0.0001	pu
4.13	0.0000	0.0000	0.0000	pu
4.14	0.0211	-0.0033	0.0208	pu
4.15	0.0002	0.0000	0.0002	pu
4.16	0.0000	0.0000	0.0000	pu

4.7 Scenari and results on Hydro-Québec system

This section details the scenari simulated on the Hydro-Québec system, described in Section 2.5. The reference outputs have been obtained with an Integrated dishonest Newton scheme (denoted as "I" in the plots).

Outputs have been obtained also with an accelerated decomposed dishonest Newton scheme (denoted "A" in the plots) that makes use of the acceleration techniques proposed in Sections 4.3 and 4.4. More details about this and the other solvers are given in Appendix A.5.

4.7.1 Case Q1

Case Q1 is a short-term scenario which is initiated by a short circuit at bus 713, lasting for 13 cycles (1 cycle at 60 Hz is $\frac{1}{60}$ s), and cleared at $t = 1$ s by the opening of a series-compensated 735-kV transmission line. This type of fault endangers rotor angle stability of large northern hydro plants. The duration of the fault corresponds to a marginally stable case. The evolution of the system is simulated up to $t = 10$ s.

The variables of interest for this case are:

- voltage at transmission bus 713, which undergoes wide oscillations after the short circuit has been cleared (Fig. 4.17);

- speed deviations of synchronous machines 49 and 59, located in two different corridors and swinging against each other (Fig. 4.18).

4.7.2 Case Q2

Case Q2 corresponds to a severe event that would lead to tripping generators with a production of more than 1000 MW. The disturbance is applied at $t = 1\ s$, and the resulting transients are computed up to $t = 100$ s.

The variables of interest for this case are:

- speed deviation of the COI (see Appendix A.1, showing the average frequency evolution (Fig. 4.19);
- dynamic voltage correction produced by the power system stabilizer of a synchronous condenser (machine 294) equipped with a low-frequency band aimed at damping oscillations of the type shown in Fig. 4.19. This signal is applied to the AVR summing junction of that machine (Fig. 4.20).

4.7.3 Case Q3

An even more severe generator tripping scenario is provided in case Q3, when three large neighboring generators, producing altogether more than 2100 MW, are tripped at $t = 1\ s$.

Figure 4.21 provides the speed deviation of the COI, showing that the spinning reserve is exhausted and frequency cannot recover close to its nominal value. Underfrequency load shedding would take place in such a case, but has not been modelled in this work.

4.7.4 Case Q4

Case Q4 is a 300 s evolution of voltage under the effect of MAIS (see Section 2.5) and LTCs, after the tripping, at $t = 1\ s$, of an important transmission line feeding the Montréal area (where most of the load is located).

The variable of interest for this case is the voltage at transmission bus 702, provided in Fig. 4.22.

It should be highlighted that the computed outputs are indistinguishable up to $t = 32$ s.

This scenario has a particular interest from the numerical point of view because the computed evolution differs insofar as the switching of the first reactor, at bus 715, is de-

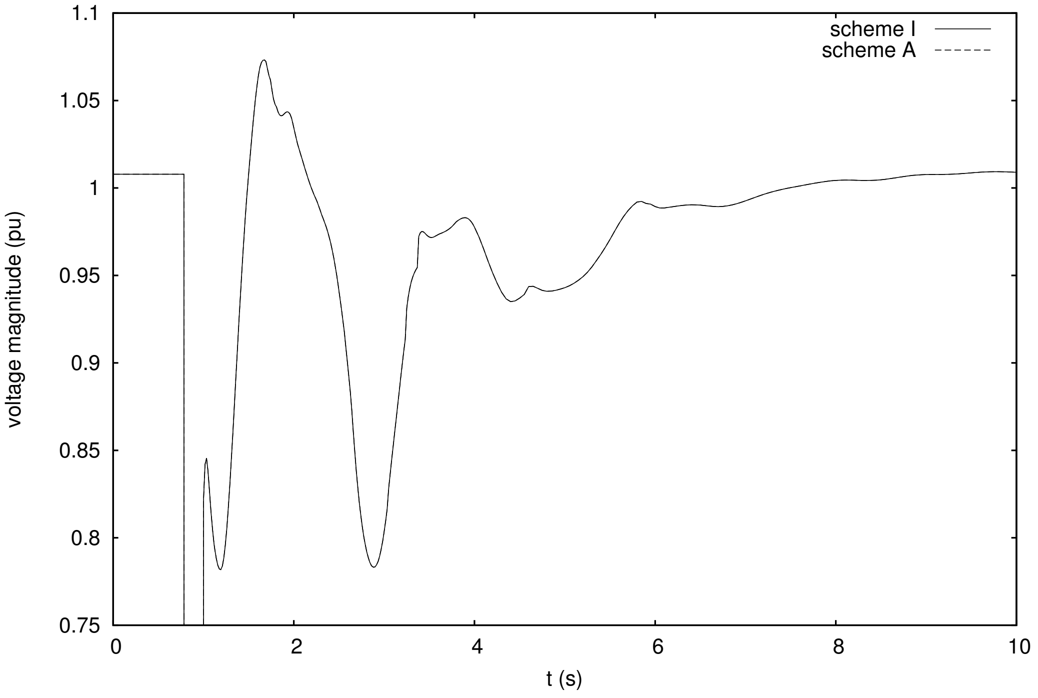


Figure 4.17: Case Q1: zoom of the voltage evolution at faulted bus 713

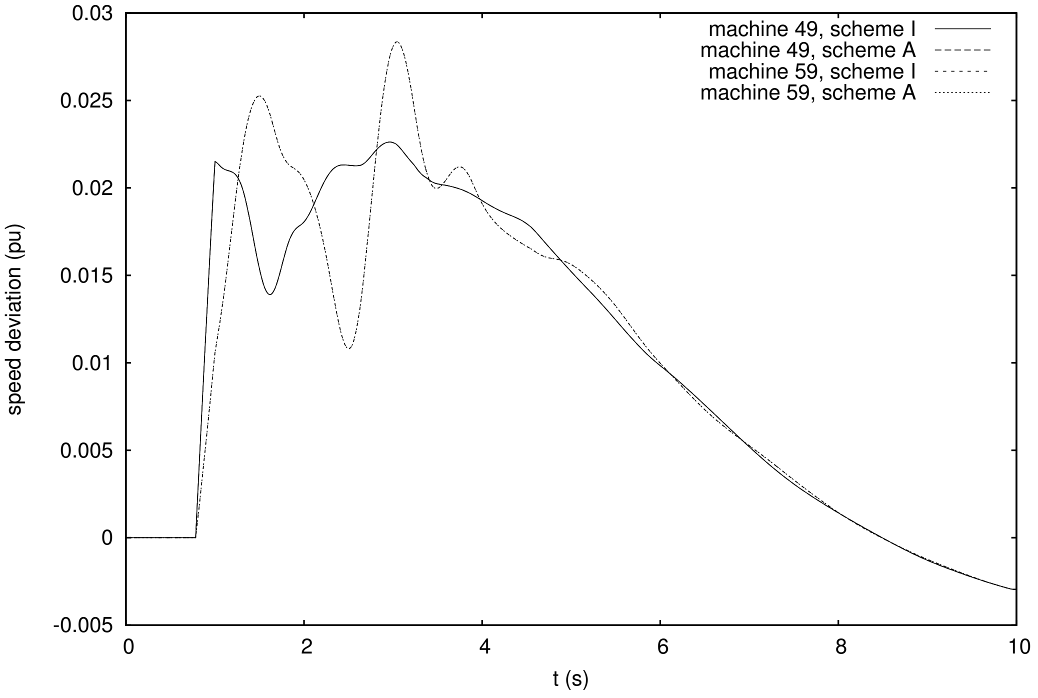


Figure 4.18: Case Q1: synchronous machine speed of generators 49 and 59

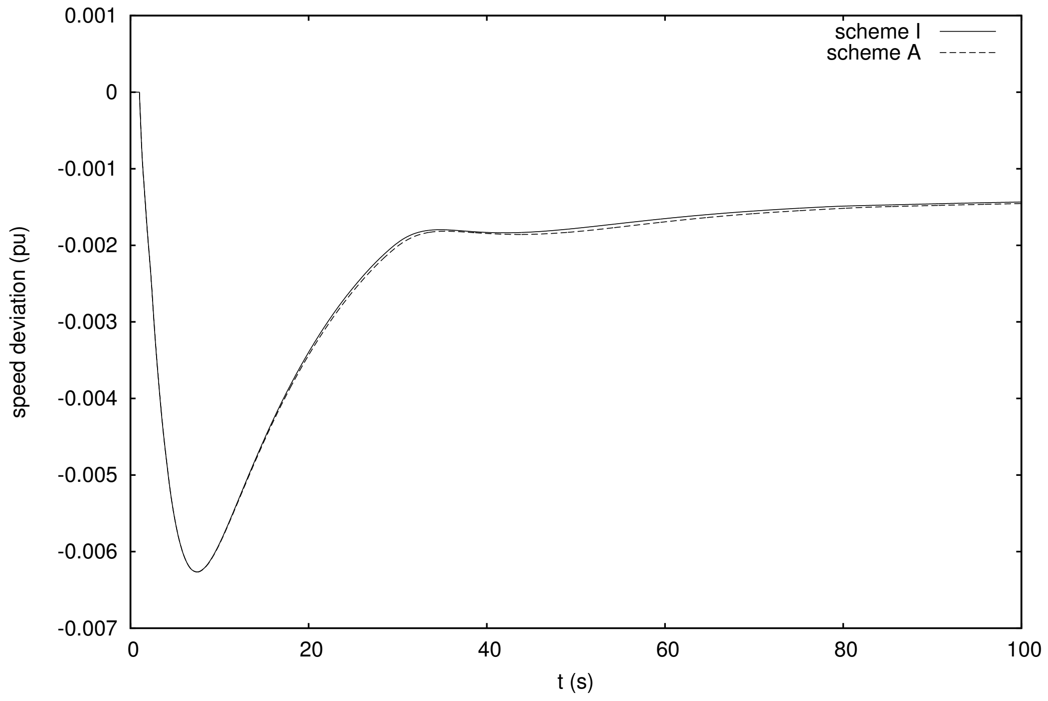


Figure 4.19: Case Q2: deviation of COI speed

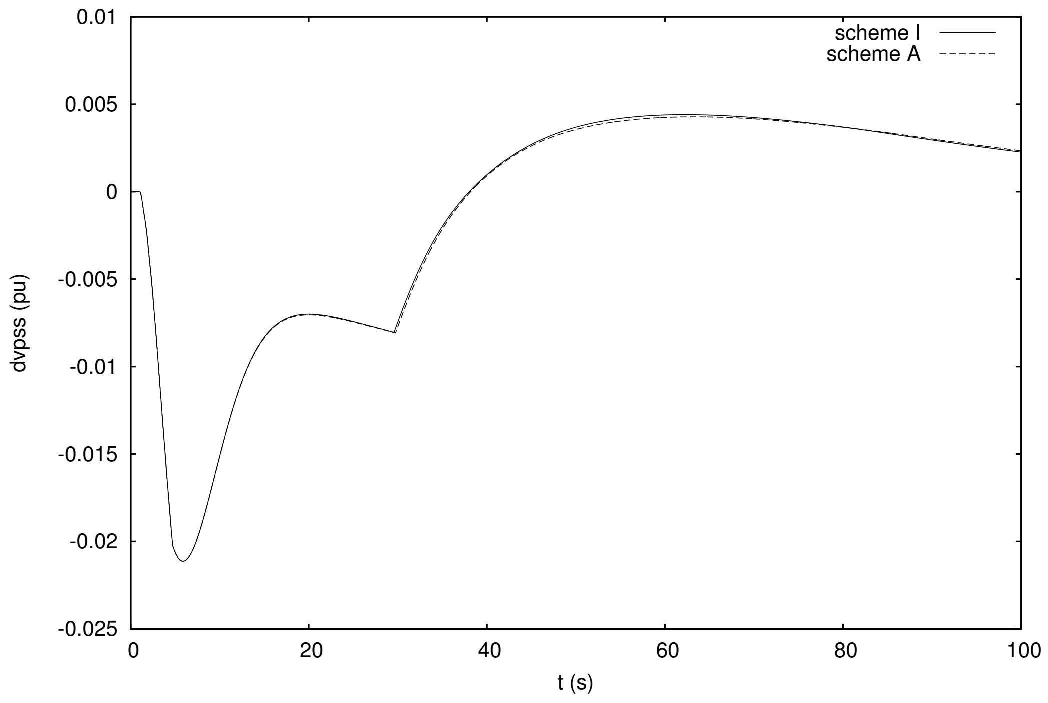


Figure 4.20: Case Q2: PSS signal of generator 294

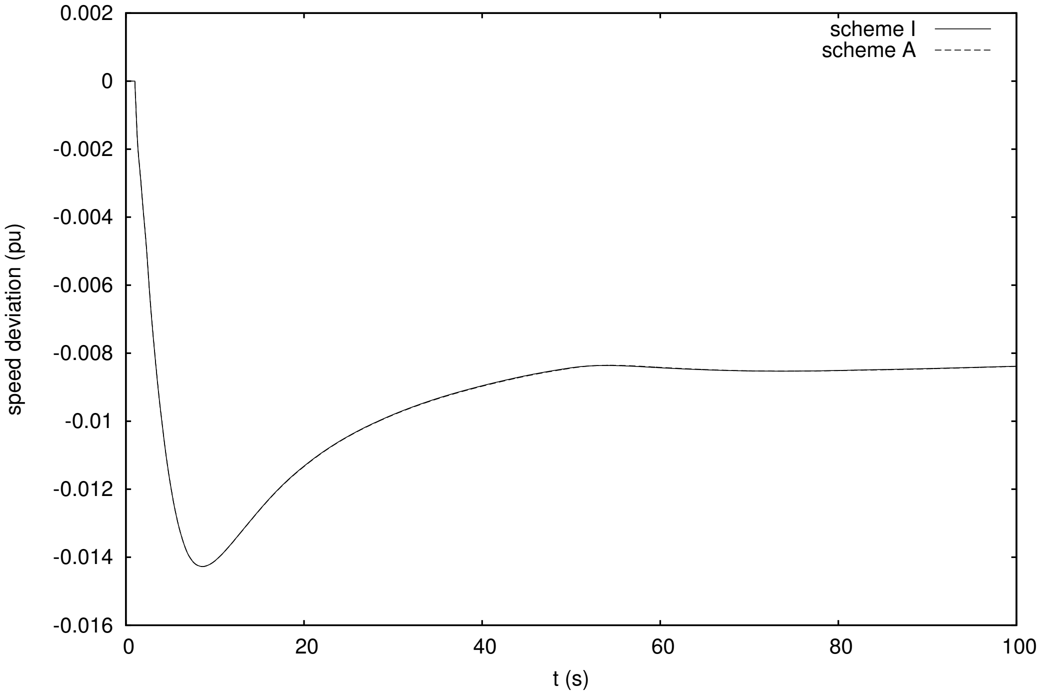


Figure 4.21: Case Q3: deviation of COI speed

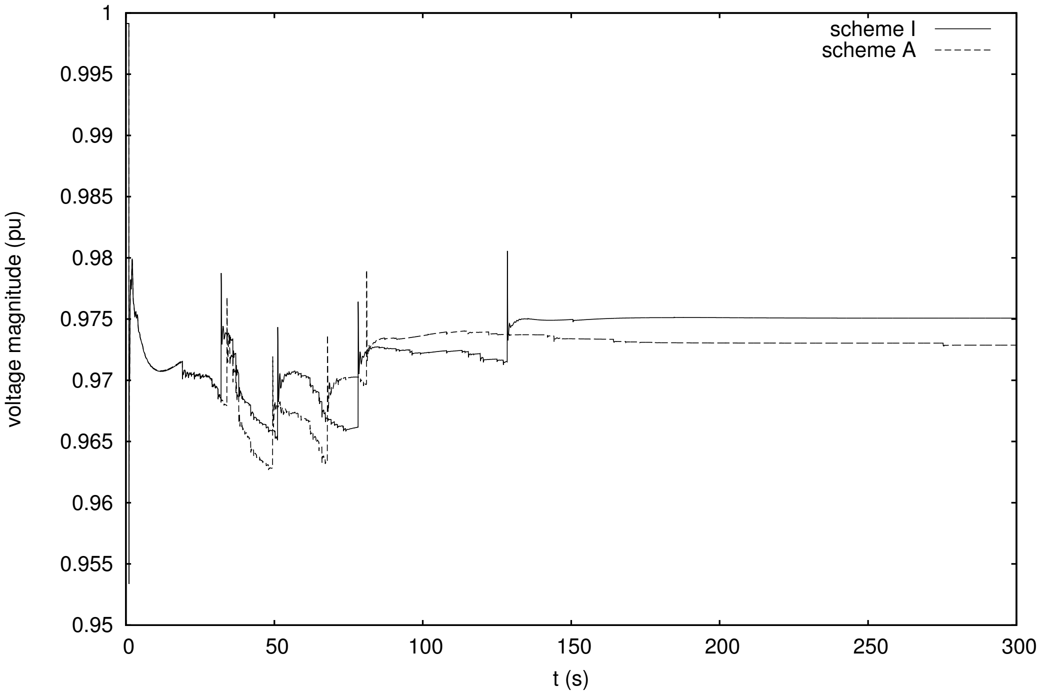


Figure 4.22: Case Q4: voltage at transmission bus 702

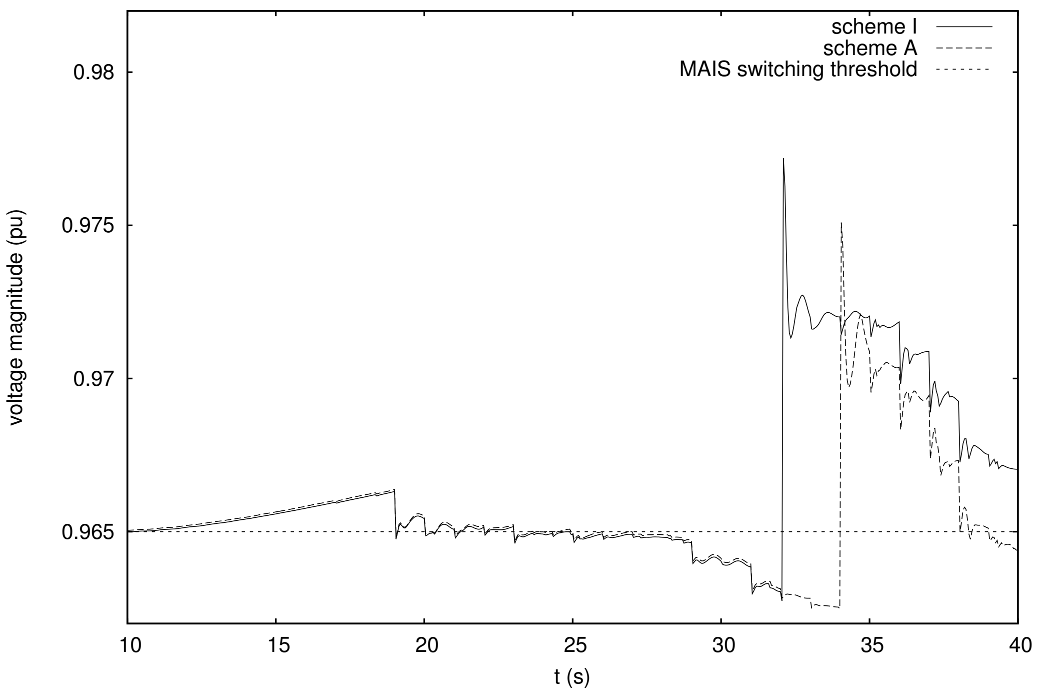


Figure 4.23: Case Q4: zoom of the voltage evolution at transmission bus 715

layed in the simulation obtained with scheme A. Figure 4.23 shows a zoom of the voltage evolution at bus 715 with the indication of the voltage threshold. The MAIS device switches off the reactor if the voltage stays below that threshold for 9 s consecutively. In the output computed with scheme I, the voltage of bus 715 crosses the threshold at $t = 23$ s and the MAIS device acts at $t = 32$ s. The voltage computed with scheme A, on the other hand, crosses the threshold at $t = 25$ s for the last time, leading to a trip at $t = 34$ s, although the overall evolution up to that point can be considered accurate. A sort of "butterfly effect" then produces a different switching time for the following three MAIS devices, leading to apparently different system trajectories.

This case highlights the mismatch that can occur between marginally different cases, when high impact discrete events, such as reactor tripping in this system, are taken into account. It would be honest to claim that anyway both simulations can be considered correct, as the acting MAIS are identified correctly, their action being merely shifted in time as shown in Fig. 4.22. This fact is well recognized by TransÉnergie engineers who consider that a discrepancy by one reactor tripping is acceptable in practice.

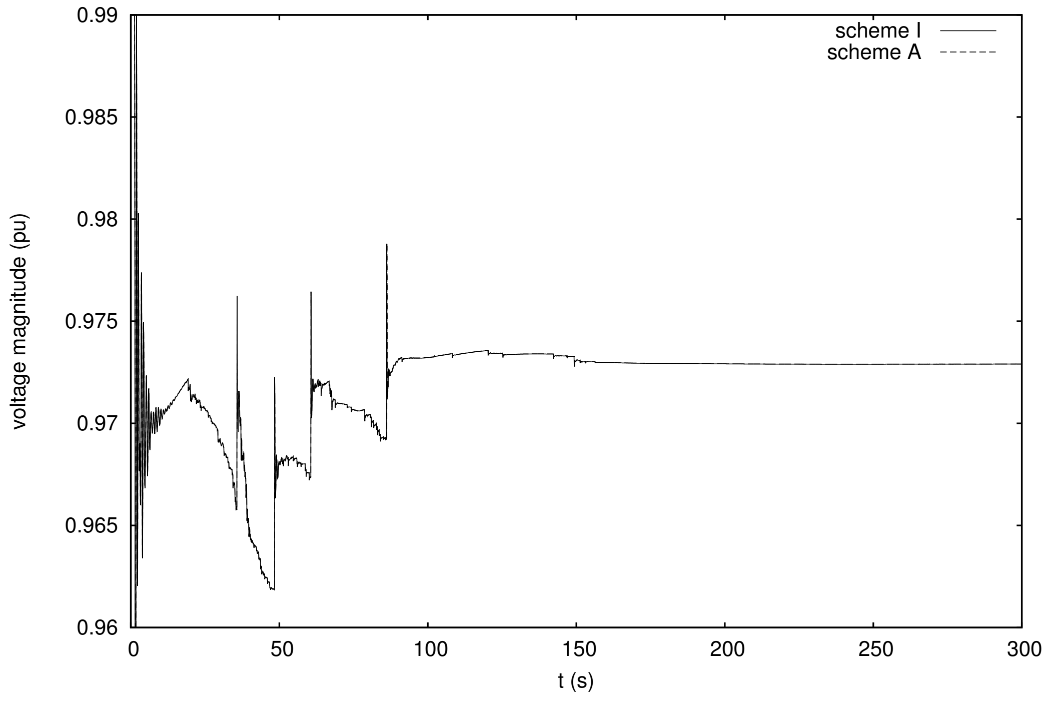


Figure 4.24: Case Q5: zoom of the voltage evolution at transmission bus 702

4.7.5 Case Q5

Case Q5 considers a short circuit near bus 702 lasting for 7 cycles, that is cleared by opening the same transmission line as in case Q4. The long-term evolution is then similar to the one in case Q4. Figure 4.24 shows the zoomed evolution of voltage at faulted bus 702. In this simulation there is no marginally switching MAIS device, thus the computed outputs are indistinguishable.

4.7.6 Computational effort

Table 4.2 summarizes the computing times obtained with the application of scheme A with respect to the benchmark provided by scheme I. It should be kept in mind that the speedup comes at no cost concerning the overall accuracy of the simulated outputs.

Table 4.2: CPU times (s)

Case	Scheme I	Scheme A
Q1	11.11	5.93
Q2	35.84	11.91
Q3	27.89	13.16
Q4	60.16	30.55
Q5	61.34	34.36

4.8 Scenari and results on PEGASE system

This section details the scenari simulated on the PEGASE test system, described in Section 2.6. The reference outputs have been obtained with an Integrated dishonest Newton scheme (denoted as "I" in the plots).

Outputs have been obtained also with an accelerated decomposed dishonest Newton scheme (denoted "A" in the plots) that makes use of the acceleration techniques proposed in Sections 4.3 and 4.4. More details about this and the other solvers are given in Appendix A.5.

4.8.1 Case P1

Case P1 is a short-term scenario which is initiated by the tripping of a double-circuit line. The evolution of the system is simulated up to $t = 20$ s.

Figure 4.25 shows the voltage at bus F0322411, one extremity of the tripped line. The difference between both schemes is imperceptible.

4.8.2 Case P2

In Case P2 the system is simulated over a period of 240 s, after the tripping, at $t = 1$ s, of two important double-circuit transmission lines. In the long-term the system evolves mainly under the effect of LTCs.

The variable of interest for this case is the voltage at transmission bus F0322411, the extremity shared by the tripped lines. The time evolution is provided in Fig. 4.26. It should be highlighted that the computed outputs are indistinguishable up to the end of the simulation.

4.8.3 Case P3

Case P3 considers a short circuit near bus F0322411 lasting for 5 cycles, that is cleared by opening the same transmission lines as in case P2. Figure 4.27 shows the zoomed short-term evolution of the voltage at the faulted bus. The long-term evolution is then similar to the one in case P2 and it is shown in Fig. 4.28.

Slightly less than 2000 discrete events of the type (2.13) occur during this 4-minute simulation, i.e. one event per 0.12 s on the average, as it was already commented in Section 3.4 and shown in Fig. 3.1.

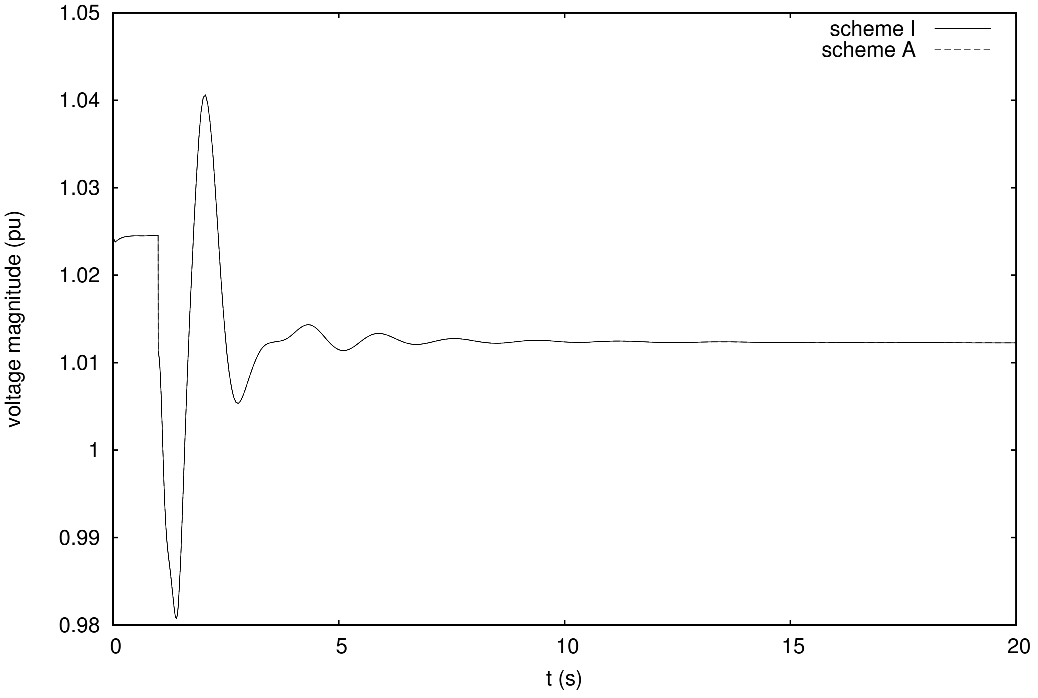


Figure 4.25: Case P1: voltage at transmission bus F0322411

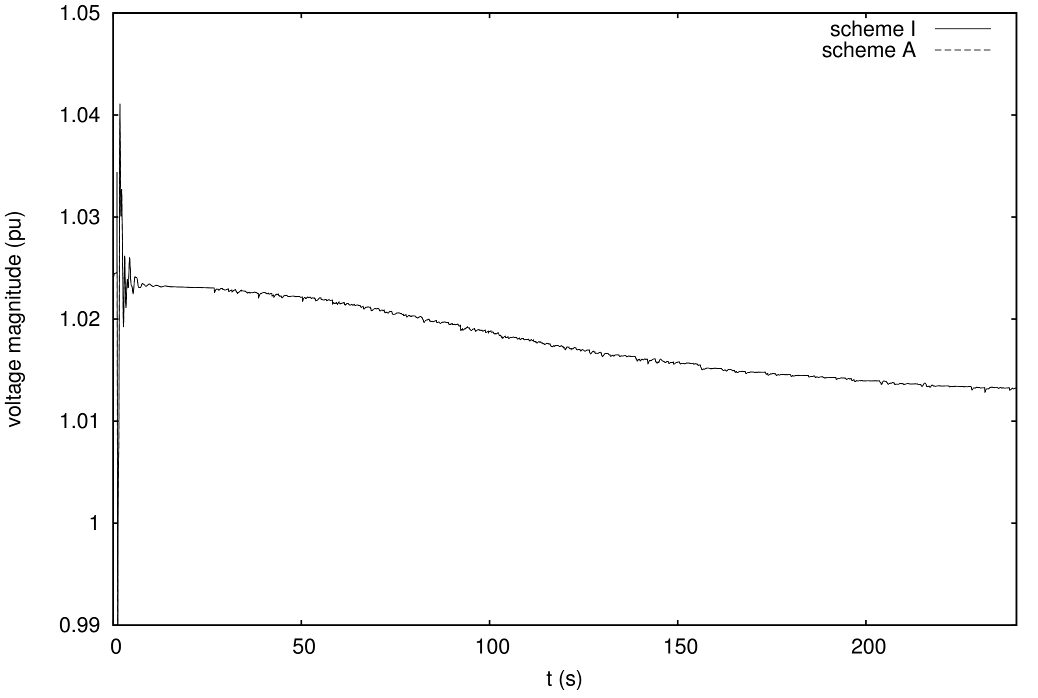


Figure 4.26: Case P2: voltage at transmission bus F0322411

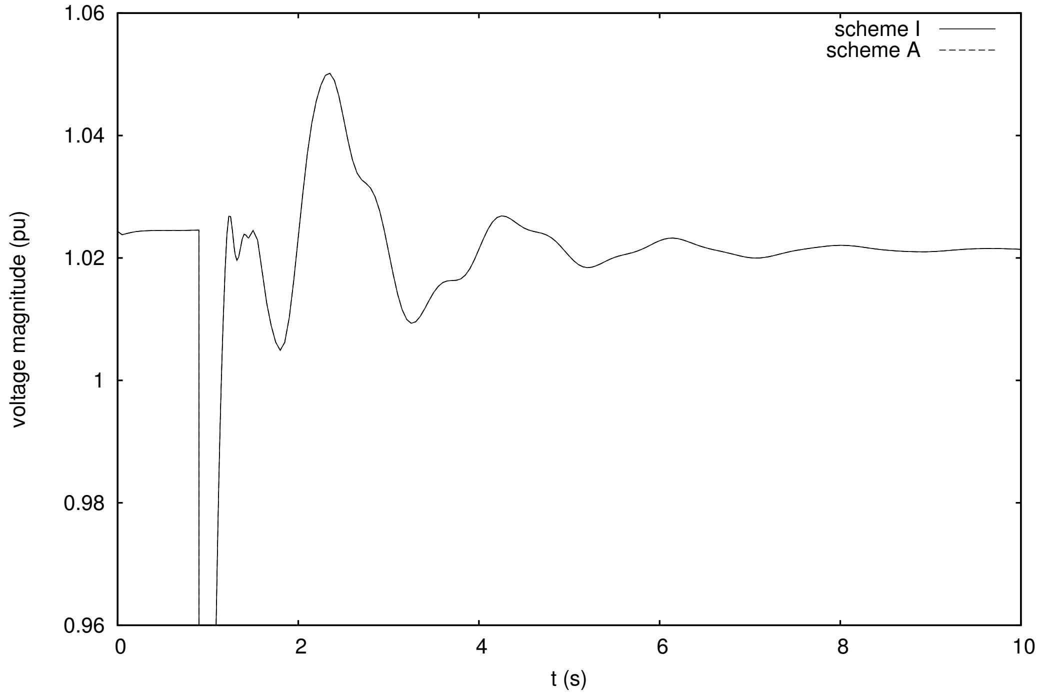


Figure 4.27: Case P3: zoom of the short-term voltage evolution at transmission bus F0322411

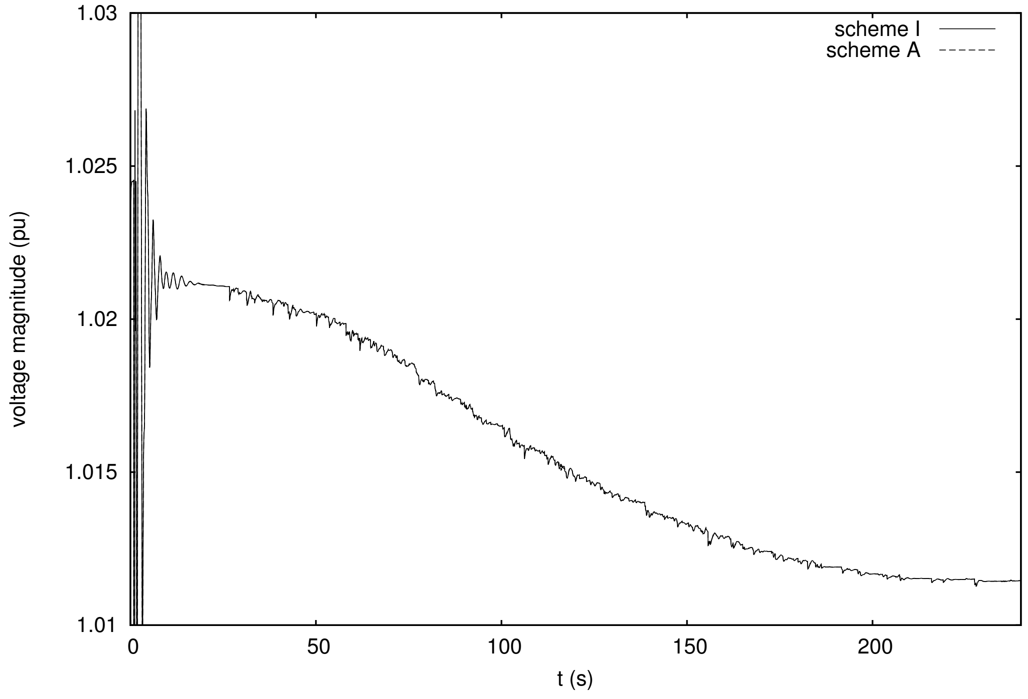


Figure 4.28: Case P3: zoom of the long-term voltage evolution at transmission bus F0322411

4.8.4 Computational effort

Tables 4.3 and 4.4 detail the computing times obtained and the number of operations performed in case P1 using the benchmark scheme I and scheme A, respectively. Similar results for Case P2 are given in Tables 4.5 and 4.6, for Case P3 in Tables 4.7 and 4.8. It should be kept in mind that the speedup comes at no cost concerning the accuracy of the simulated outputs. All these tables show that the highest gain is at the factorization level, since scheme A allows to perform partial Jacobian updates only where it is needed while scheme I has to perform full Jacobian updates. A significant gain is also provided by skipping injector solutions, while a slightly larger time is spent in injector function evaluations, since two evaluations per iteration and per non-converged injector are needed by scheme A (see Section 4.2, using the variant (4.16) instead of (4.9), which introduces the aforementioned additional function evaluation) while only one is performed by scheme I (see Section 3.3).

Table 4.3: Case P1: CPU times (s)

	Scheme I		Scheme A
total	90.29	total	32.81
factorizing $\mathbf{J}$	57.51	factorizing $\tilde{\mathbf{D}}$	0.69
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	11.19	solving for $\mathbf{\Delta V}$	0.82
evaluating $[\mathbf{g}\ \ \mathbf{f}]^\mathbf{T}$	13.11	evaluating $\mathbf{g}$	1.97
		factorizing $\mathbf{A}_i$ *	2.84
		solving for $\mathbf{\Delta x}_i$ *	6.51
		evaluating $\mathbf{f}_i$ *	14.75

* $i = 1, \dots, n$

Table 4.4: Case P1: Number of operations

	Scheme I		Scheme A
factorizing $\mathbf{J}$	446	factorizing $\tilde{\mathbf{D}}$	25
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	1393	solving for $\mathbf{\Delta V}$	1031
evaluating $[\mathbf{g}\ \ \mathbf{f}]^\mathbf{T}$	1891	evaluating $\mathbf{g}$	2326
		factorizing $\mathbf{A}_i$ *	26
		solving for $\mathbf{\Delta x}_i$ *	792
		evaluating $\mathbf{f}_i$ *	2321

* average per injector

Table 4.5: Case P2: CPU times (s)

	Scheme I		Scheme A
total	470.00	total	220.21
factorizing $\mathbf{J}$	207.58	factorizing $\tilde{\mathbf{D}}$	0.75
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	90.00	solving for $\mathbf{\Delta V}$	7.42
evaluating $[\mathbf{g}\ \ \mathbf{f}]^\mathbf{T}$	111.86	evaluating $\mathbf{g}$	14.04
		factorizing $\mathbf{A}_i$ ⋆	2.51
		solving for $\mathbf{\Delta x}_i$ ⋆	41.53
		evaluating $\mathbf{f}_i$ ⋆	117.25

⋆ $i = 1, \ldots, n$

Table 4.6: Case P2: Number of operations

	Scheme I		Scheme A
factorizing $\mathbf{J}$	1597	factorizing $\tilde{\mathbf{D}}$	27
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	11106	solving for $\mathbf{\Delta V}$	8514
evaluating $[\mathbf{g}\ \ \mathbf{f}]^\mathbf{T}$	15988	evaluating $\mathbf{g}$	17160
		factorizing $\mathbf{A}_i$ ⋆	22
		solving for $\mathbf{\Delta x}_i$ ⋆	5634
		evaluating $\mathbf{f}_i$ ⋆	19044

⋆ average per injector

Table 4.7: Case P3: CPU times (s)

	Scheme I		Scheme A
total	438.21	total	232.56
factorizing $\mathbf{J}$	179.40	factorizing $\tilde{\mathbf{D}}$	0.93
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	88.93	solving for $\mathbf{\Delta V}$	8.19
evaluating $[\mathbf{g}\ \ \mathbf{f}]^\mathbf{T}$	111.34	evaluating $\mathbf{g}$	13.91
		factorizing $\mathbf{A}_i$ ⋆	4.49
		solving for $\mathbf{\Delta x}_i$ ⋆	47.35
		evaluating $\mathbf{f}_i$ ⋆	119.44

⋆ $i = 1, \ldots, n$

Table 4.8: Case P3: Number of operations

	Scheme I		Scheme A
factorizing $\mathbf{J}$	1374	factorizing $\tilde{\mathbf{D}}$	32
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	10975	solving for $\mathbf{\Delta V}$	8556
evaluating $[\mathbf{g}\ \ \mathbf{f}]^\mathbf{T}$	15946	evaluating $\mathbf{g}$	17020
		factorizing $\mathbf{A}_i$ ⋆	40
		solving for $\mathbf{\Delta x}_i$ ⋆	5757
		evaluating $\mathbf{f}_i$ ⋆	19285

⋆ average per injector

Figure 4.29 shows the number of injector solutions per step in scheme I. The first 10 s are not shown because deemed less interesting. With scheme I, solutions are computed for all injectors at each Newton iteration. Thus, given that the number of injectors of this system is $n = 10694$, this plot only shows values in multiples of n: for most of the steps, 1,2,3 or 4 iterations are enough to solve the system of equations up to the desired tolerance. For legibility, values larger than 50000 are not displayed, although present at a few time steps with more difficult convergence (corresponding to 5,6 and 7 iterations).

Figure 4.30 shows the number of injector solutions per step in scheme A. The lower bound of the plotted values is again n, since at least one iteration per injector is performed at each step. As in scheme I, some of the injectors need up to 4, or in some cases even 6 iterations in order to be solved: as expected, however, not all the injectors need the same number of iterations, as many are already solved up to convergence with just one Newton iteration.

The small subset of injectors that require more iterations is also responsible for the much higher number of Jacobian factorization in scheme I with respect to scheme A: indeed, while in scheme A a slow convergence of one injector forces to update only its own Jacobian matrix, in scheme I a slow convergence of just one injector (or, for that matter, even of just one equation inside one injector) slows down the convergence of the whole system, many times forcing the solver to perform a full Jacobian update.

In integrated schemes, this problem can be easily avoided at some risk by temporarily disabling the infinite-norm or adopting a varying-order-norm, as discussed in Section 3.3: a few large mismatches would then be shaded by the majority of small mismatches, and considered converged before triggering Jacobian updates. In the author's opinion, however, the different rate of convergence of variables is inherent in large power systems made up of many components which experience different level of activity, and thus converge each at its own speed. This fact provides valid arguments in favor of decomposition for large-scale power systems, where the natural choice of the infinite-norm may be kept at (almost) no cost.

4.9 Chapter conclusion

The most important contributions of this chapter are the acceleration techniques made possible by decomposing the Newton method algorithm. The skipping solution of converged

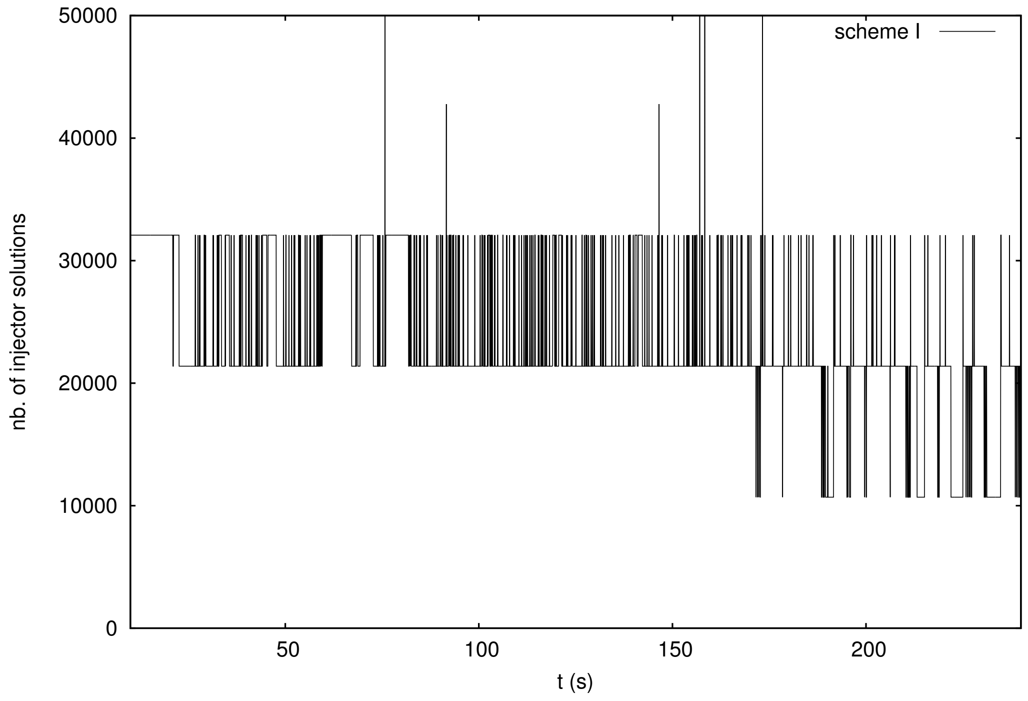


Figure 4.29: Case P3: number of injector solutions per step (scheme I)

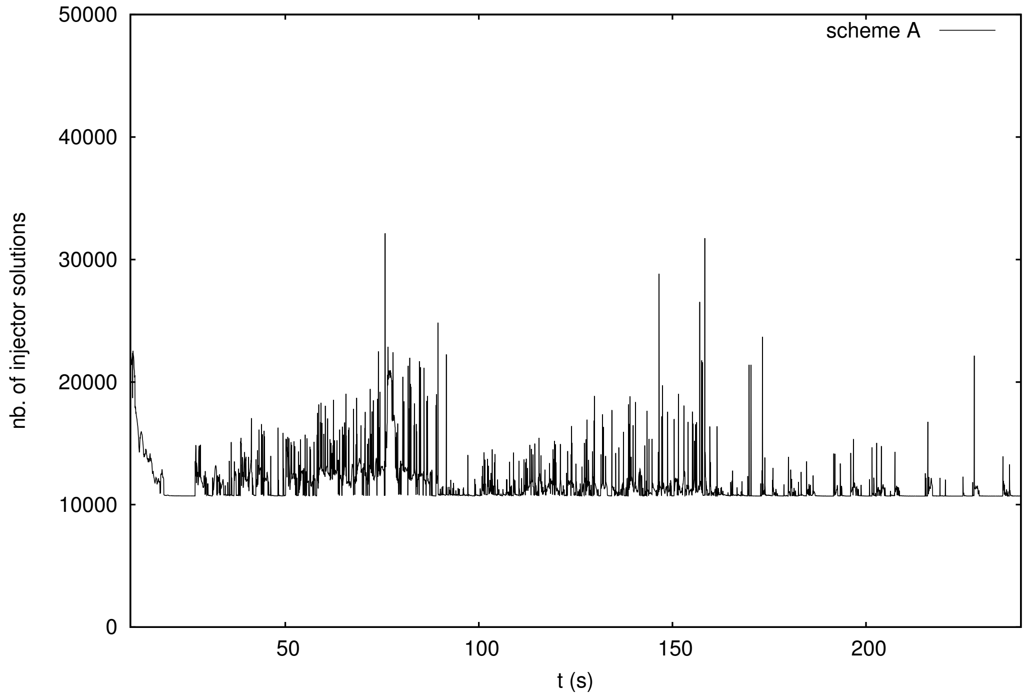


Figure 4.30: Case P3: number of injector solutions per step (scheme A)

components detailed in Section 4.3 and the partial Jacobian update strategy detailed in Section 4.4 bring a speed-up ranging between 2 and 3. This speed-up comes at no cost, since the accuracy of the computed outputs is preserved.

Chapter 5

Relaxing accuracy in space

Divide et impera [1]

A brief review of spatial accuracy relaxation methods opens this chapter, with a particular emphasis on localization methods for power flow and latency exploitation in electronic circuits. The idea of localization, borrowed from the power flow problem, but novel in its application to dynamic simulation is presented and implemented thanks to the BBD decomposition detailed in Chapter 4. The influence of the latency tolerance is discussed and results are presented on the three test systems showing that the computational effort can be reduced gradually, along with the accuracy of the simulated outputs, especially for system components which are marginally participating to the system dynamics.

5.1 Spatial accuracy relaxation in literature

The concept of *localization* results from the observation of the limited effect that a vast majority of disturbances has over a large system, and it has been used extensively both in power systems [Bra93] and in electronic circuits [HSV81].

This phenomenon has been exploited since the dawn of power system numerical simulation, although it was not mentioned under the name of localization. For instance, the network reduction leading to the so-called Ward equivalents [War49], which is simply the

[1] *Divide and rule.* Apocryphal, attributed to Gaius Julius Caesar and Napoléon Bonaparte among others.

Gaussian elimination of the external buses in the system admittance matrix equation, has long been considered the standard for static equivalencing [MDGS79]. The main hypothesis justifying the widespread use of equivalencing is that the role of the external network is marginal for the evaluation of the phenomena of interest; historically its adoption was motivated by computational efficiency reasons, especially in the context of online security assessment [WM83], while recently the use of external equivalents in order to deal with confidentiality issues in data exchanges between system operators has also been advocated [FGWVC09].

Equivalents have been also used in dynamic simulation. As regards electromechanical oscillations, the techniques used [ANL$^+$12] can be roughly classified into (i) *modal* methods [Cho82], based on the linearized state-space model, (ii) *coherency* methods [GP78], based on clustering the system components according to their response to a given disturbance and (iii) *measurement* methods [YES81], based on system identification techniques applied to recorded or simulated responses.

The main drawback of techniques based on equivalencing is that the model simplification is performed at the beginning of the simulation and it cannot, in principle, be changed to suit different scenari. Furthermore, a *universal* equivalent that would be valid for all power system dynamic phenomena, to the author's knowledge is still to be found.

Contingency analysis has been another area of application of localization concepts and a step forward towards an adaptive use of spatial relaxation. Several approaches exploited the limited widespread of contingencies in order to compute the corresponding post-disturbance state without taking into account the whole system: among those approaches let us mention concentric relaxation [ZWP80], bound estimates method [Gal84] and bounding method both for DC [Bra88] and AC [BL89] power flow computations.

The approaches listed above, while allowing to vary the degree of accuracy relaxation to match the different scenari, still suffer from the fact that different reduced systems have to be computed as the affected area changes, bringing additional overhead if a direct method, such as the Newton method, is used for the solution of the resulting system of equations. The algorithms based on sparse vector method [TBC85] tackle this problem by allowing a partial solution of the whole system, i.e. a variation of the relaxation degree without the need of restructuring the underlying systems of equations. Successive refinements, such as the zero mismatch approach [BT89], the approximate sparse vector method [BET91] and adaptive localization [EPT92] improved the original approach, while being still limited to the power flow problem, though.

To our knowledge, extension of sparse vector approaches to power system dynamics, although intuitively appealing, has not been investigated. The localization algorithm described in Section 5.2 bears the spirit of those approaches insofar as it allows a varying degree of accuracy relaxation, while keeping the problem structure intact.

Variable partitioning multi-rate methods [CC08], which adapt the subdomains according to the "level of activity" of the variables involved, are probably the most advanced exploitation of the localization property in power system dynamics. In this approach, variables are divided in different subsets which are solved with different time steps. The two main drawbacks are that (i) the method suffers from degradation of convergence due to decomposition and interface problems inherent to relaxation methods in general, especially when there is no overlap between subdomains [TW05], and (ii) the decomposition is based on the "dynamic" activity of variables, not on their location, resulting in subsets which, in general, may not be contiguous. For this last reason this method cannot be fully classified as a spatial relaxation of accuracy, as will be further discussed in Chapter 6. The decomposition into network and injectors considered in this work is free from the above drawbacks.

In electronic circuits simulations, localization is referred to as *latency*. It has been exploited for quite some time. The high popularity of this method in electronics can be explained by the nature of the involved circuits which are composed of simple repetitive subnetworks [RSVH79]. Especially in digital circuits, the latency state of a component can be determined rather easily compared to power systems; moreover, subnetworks are latent for most of the simulation. As it will be detailed in Section 5.3, the application of localization to power systems requires a more careful test to detect the latent state of a component, by the comparison of its level of activity to a latency tolerance.

5.2 Localization in the decomposed Newton scheme

The idea behind localization is to solve only a subset of injectors, identified as *active*, and skip solution of the other ones, identified as *latent*.

The active injectors are solved with the same accuracy as for a detailed simulation. Furthermore, with the techniques presented in Chapter 4, Eqs. (4.9) are solved more times for injectors with higher dynamic response. On the other hand, for the latent ones, Eqs.

(4.9) will not be solved at all; in fact, their functions $\mathbf{f}_i$ will not even be computed. Note that the whole network will still be solved according to (4.11).

We found it appropriate to replace a latent injector by a linear approximation between its current and voltage. This approximation is obtained from (4.9), assuming that the injector internal dynamics are negligible, i.e.

$$\mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{z}_i, \mathbf{V}^{k-1}) \simeq \mathbf{0}$$

Under this approximation, (4.9) becomes:

$$\mathbf{A}_i \Delta\mathbf{x}_i \simeq -\mathbf{B}_i \Delta\mathbf{V} \tag{5.1}$$

from which $\Delta\mathbf{x}_i$ is easily obtained:

$$\Delta\mathbf{x}_i \simeq -\mathbf{A}_i^{-1}\mathbf{B}_i \Delta\mathbf{V} \tag{5.2}$$

The variation of the current components is obtained by pre-multiplying (5.2) by $\mathbf{C}_i$:

$$\begin{bmatrix} \Delta I_{xi} \\ \Delta I_{yi} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \mathbf{C}_i \Delta\mathbf{x}_i \simeq -\mathbf{C}_i\mathbf{A}_i^{-1}\mathbf{B}_i\Delta\mathbf{V} = -\tilde{\mathbf{C}}_i\mathbf{B}_i\Delta\mathbf{V} \tag{5.3}$$

which is the sought linear relation between current and voltage. Extracting the nonzero elements of (5.3) yields:

$$\begin{bmatrix} \Delta I_{xi} \\ \Delta I_{yi} \end{bmatrix} = -\mathbf{S}_i \begin{bmatrix} \Delta V_{xi} \\ \Delta V_{yi} \end{bmatrix} \tag{5.4}$$

where the (2×2) matrix $\mathbf{S}_i$ contains the 4 nonzero elements of $\tilde{\mathbf{C}}_i\mathbf{B}_i$.

Since the above relations are linear, their variations can be rewritten around a linearization point at an arbitrary instant t_i^*, characterized by current values $\begin{bmatrix} I_{xi}(t_i^*) & I_{yi}(t_i^*) \end{bmatrix}^T$ and voltage values $\begin{bmatrix} V_{xi}(t_i^*) & V_{yi}(t_i^*) \end{bmatrix}^T$, without loss of generality:

$$\begin{bmatrix} I_{xi} \\ I_{yi} \end{bmatrix} = \begin{bmatrix} I_{xi}(t_i^*) \\ I_{yi}(t_i^*) \end{bmatrix} - \mathbf{S}_i \begin{bmatrix} V_{xi} - V_{xi}(t_i^*) \\ V_{yi} - V_{yi}(t_i^*) \end{bmatrix} \tag{5.5}$$

Going back to the full state vector $\mathbf{x}_i$ of the injector, the above formula can be rewritten in compact form as

$$\mathbf{x}_i = \mathbf{x}_i^* - \tilde{\mathbf{C}}_i\mathbf{B}_i\Delta\mathbf{V}^* \tag{5.6}$$

where $\mathbf{x}_i^*$ is the full state vector value at the arbitrary instant t_i^* and $\Delta\mathbf{V}^*$ is defined as $\mathbf{V} - \mathbf{V}^*$.

Note that the linear formula (5.5) is simple and merely involves matrix multiplications. It does not require a Newton solution, which is precisely the objective for latent injectors.

Furthermore, one important feature is that the correction $-\tilde{\mathbf{C}}_i\mathbf{B}_i$ brought by the i-th injector to the $\mathbf{D}$ matrix is the same whether the injector is active or latent. In other words, when an injector switches from active to latent or viceversa, it does not modify the $\tilde{\mathbf{D}}$ matrix. The latter need not be updated, and the convergence of the Newton method is not degraded by re-using the same matrix in a "dishonest" way.

If the i-th injector is latent, its current components I_{xi} and I_{yi} are updated according to (5.3) but the other components of its state vector $\mathbf{x}_i$ are no longer updated. There is an exception, however, for synchronous machines: it is assumed that a latent synchronous machine swings coherently with the center of inertia of the system. Hence, at each time step, its rotor speed ω is set to ω_{coi}, the angular speed of the center of inertia.

In the sequel, the resulting algorithm is referred to as *localized Newton scheme*. The corresponding flowchart is shown in Fig. 5.1, which should be compared to Figs. 4.3 and 4.4.

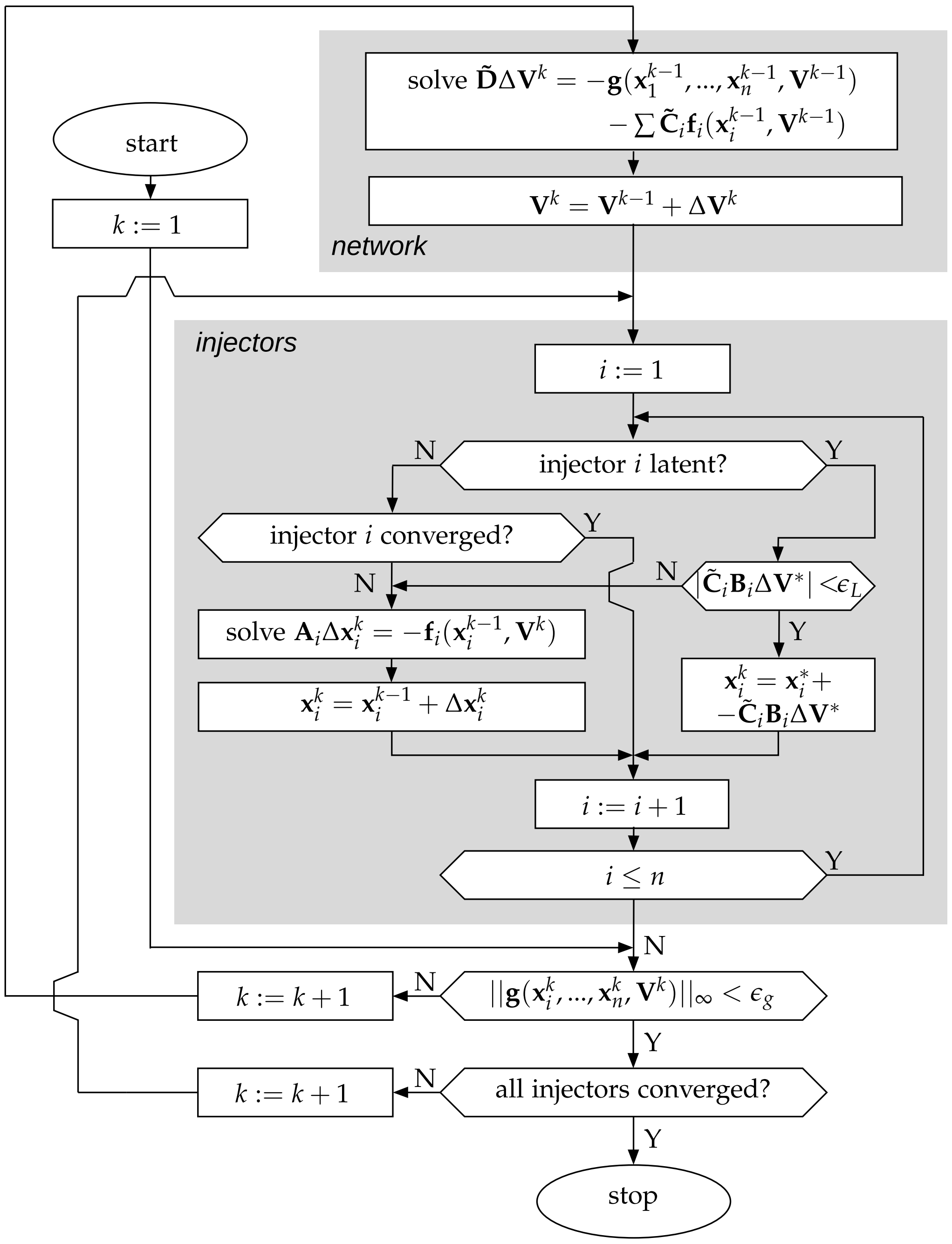
5.3 Latency detection algorithm

Which injectors are latent and which ones are active cannot be decided *a priori* because it depends on the simulated disturbance. This is a major advantage of the proposed method with respect to equivalencing techniques. Furthermore, it may happen that an injector changes from active to latent after some transients have died out, then switches back to active under the effect of the system evolution.

We define the linearization instant t_i^* as the last time the i-th injector changed from active to latent and was replaced by the linear model (5.3) or its equivalent relation (5.5). At the beginning of the simulation, t_i^* is initialized to zero for all injectors.

As a result of extensive tests, the following latency identification rules were found the most satisfactory:

1. at the beginning of the simulation or after a user-imposed event: all injectors are classified active (in the absence of information regarding the dynamics that will be excited);



network
solve $\tilde{\mathbf{D}}\Delta\mathbf{V}^k = -\mathbf{g}(\mathbf{x}_1^{k-1}, ..., \mathbf{x}_n^{k-1}, \mathbf{V}^{k-1}) - \sum \tilde{\mathbf{C}}_i \mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{V}^{k-1})$
$\mathbf{V}^k = \mathbf{V}^{k-1} + \Delta\mathbf{V}^k$
start
$k := 1$
injectors
$i := 1$
injector i latent?
N
Y
injector i converged?
Y
N
N
$|\tilde{\mathbf{C}}_i\mathbf{B}_i\Delta\mathbf{V}^*| < \epsilon_L$
Y
solve $\mathbf{A}_i\Delta\mathbf{x}_i^k = -\mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{V}^k)$
$\mathbf{x}_i^k = \mathbf{x}_i^{k-1} + \Delta\mathbf{x}_i^k$
$\mathbf{x}_i^k = \mathbf{x}_i^* + -\tilde{\mathbf{C}}_i\mathbf{B}_i\Delta\mathbf{V}^*$
$i := i + 1$
$i \leq n$
Y
N
$||\mathbf{g}(\mathbf{x}_i^k, ..., \mathbf{x}_n^k, \mathbf{V}^k)||_\infty < \epsilon_g$
N
$k := k + 1$
Y
all injectors converged?
N
$k := k + 1$
Y
stop

Figure 5.1: Localized Newton scheme

2. at the end of each time step, say at time t : the i-th injector switches from active to latent if its current components have not changed by more than a threshold ϵ_L since the time t_i^*, i.e. if

   $$|I_{xi}(t) - I_{xi}(t_i^*)| < \epsilon_L \text{ and } |I_{yi}(t) - I_{yi}(t_i^*)| < \epsilon_L \tag{5.7}$$

   Thus, the current is used to identify the low level of dynamic activity of the injector, and skip it in the subsequent computations;

3. inside one time step, i.e. during the Newton iterations, a latent injector switches from latent to active if at least one of its current components exhibits a variation above the threshold ϵ_L, i.e. if

   $$|I_{xi}^k(t) - I_{xi}(t_i^*)| > \epsilon_L \text{ or } |I_{yi}^k(t) - I_{yi}(t_i^*)| > \epsilon_L \tag{5.8}$$

   where k is the iteration counter. This gives the injectors a chance to react to upcoming dynamics.

Thus, once an injector is latent, its current is computed from (5.5). Hence, the test (5.8) is performed on the currents computed from that linear approximation. Former latency identification rules are documented in [FVC10].

Therefore, the tests (5.7) and (5.8) can be seen as an indirect way of monitoring the variations of the voltage at the terminal of the latent injector: in fact, Eq. (5.7) can be rewritten in compact form as:

$$|\tilde{\mathbf{C}}_i \mathbf{B}_i \Delta \mathbf{V}^*| < \epsilon_L$$

and Eq. (5.8) as

$$|\tilde{\mathbf{C}}_i \mathbf{B}_i \Delta \mathbf{V}^*| > \epsilon_L$$

where the norm is intended to be an infinite norm.

Figure 5.2 shows a typical partition of injectors into active and latent, during a simulation that makes use of localization. It should be compared to Fig. 4.2.

If the threshold ϵ_L is set to zero, no injector is latent and the solution accuracy is the same throughout the whole system and unchanged with respect to the decomposed Newton scheme. For increasing values of ϵ_L, injectors with low dynamic response, likely located at some distance from the disturbance area, are considered latent. Localization enables to solve at each time step different parts of the system with different levels of accuracy depending on the level of activity.

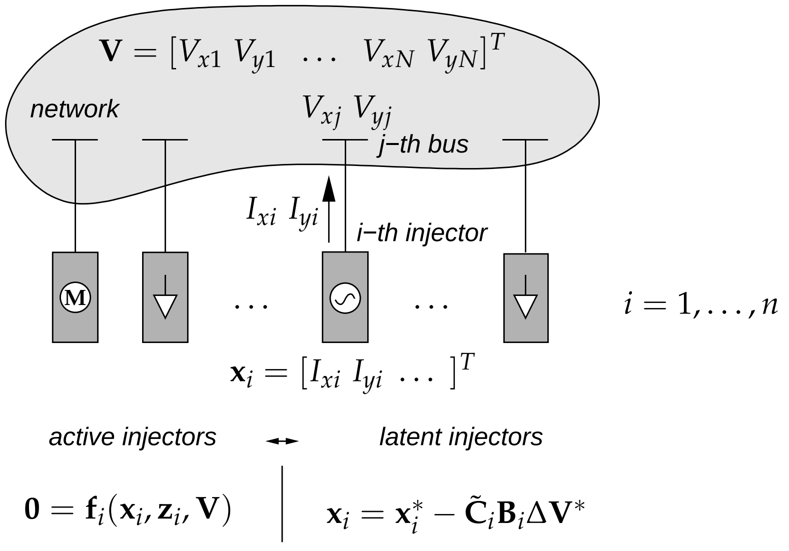

$\mathbf{V} = [V_{x1}\ V_{y1}\ \dots\ V_{xN}\ V_{yN}]^T$
network
$V_{xj}\ V_{yj}$
j–th bus
$I_{xi}\ I_{yi}$
i–th injector
M
$i = 1, \dots, n$
$\mathbf{x}_i = [I_{xi}\ I_{yi}\ \dots\ ]^T$
active injectors
latent injectors
$\mathbf{0} = \mathbf{f}_i(\mathbf{x}_i, \mathbf{z}_i, \mathbf{V})$
$\mathbf{x}_i = \mathbf{x}_i^* - \tilde{\mathbf{C}}_i \mathbf{B}_i \Delta \mathbf{V}^*$

Figure 5.2: Partition of injectors into active and latent

5.4 Localization results on Nordic32 system

This section illustrates how localization works in practice, in the scenari described in Section 4.6. The effect of localization on such a small system is limited: for this reason only Cases N1 and N2 will be shown in detail; in the other cases, which end with a collapse of the system, localization is exploited only marginally. The outputs have been obtained with a localized dishonest Newton scheme, denoted "L" in the plots, using different latency tolerances ϵ_L. Localization has not been used during the first half second after a user-imposed disturbance. More details about this and the other solvers are given in Appendix A.5. Only some representative curves are shown, since the trajectories obtained with scheme L are very close to the benchmark trajectories of scheme I in all cases, at least in the DTW sense (i.e. without penalizing too much small delays). The reported DTW distances are computed on the output given by scheme L with respect to the one given by scheme I, and are given in Table 5.1. Relevant results concerning the speed-up are detailed in the next sections, on Hydro-Québec and PEGASE systems, which, due to their larger size, profit more from localization.

5.4.1 Case N1

Figure 5.3 shows the active power flow in the other circuit of line 1013-1014. It can be seen that the simulated response is accurate whatever the value of the latency tolerance. With

$\epsilon_L = 0.001$, 23 injectors out of 42 are latent for most of the simulation. Increasing the latency tolerance brings this number up to 28, for $\epsilon_L = 0.002$, and 36 for $\epsilon_L = 0.01$.

Figure 5.4 depicts the field voltage of generator g15, only marginally excited by the disturbance. The instants where the generator has switched from active to latent are easily identified from the horizontal line. The higher the value of ϵ_L, the sooner g15 becomes latent and its field voltage evolution shows a straight line. At first glance, the evolution provided by setting $\epsilon_L = 0.01$ does not look accurate; however, bearing in mind the small range of values of the displayed output, it is clear that the approximation brought by even the highest latency tolerance is acceptable for many practical applications.

5.4.2 Case N2

Case N2 involves a long-term scenario. Figure 5.5 shows the voltage at the distribution bus fed by 4043. The two LTC moves of the transformers, at around $t = 38$ s and $t = 46$ s, are both shifted in time for $\epsilon_L = 0.01$, but this does not significantly impact the overall evolution of the system. For $\epsilon_L = 0.002$, only a single LTC move only is delayed, while $\epsilon_L = 0.001$ returns no difference with respect to the benchmark.

As in Case N1, a progressively larger number of injectors becomes latent with the increase of the latency tolerance.

The field voltage and active power of generator g15 are shown in Figs. 5.6 and 5.7, respectively. This machine is very close to the disturbance, and for $\epsilon_L = 0.01$ it only becomes latent towards the end of the simulation. Tighter latency tolerances do not allow it to switch to latent. For increasing latency tolerances, the initial oscillations tend to be over-damped, as the increasing values of σ in Table 5.1 also indicate. The simulated outputs, overall, are accurate, as shown by the δ values in Table 5.1: the DTW distance, in fact, allows catching the similar pattern of outputs, without over-penalizing the delays as the Euclidean distance would do in this case.

5.4.3 Other cases

Cases N3, N4 and N5, which all end up collapse, offer less possibilities to exploit localization. Nonetheless, their output are still accurate, as shown by the DTW distances given in Table 5.1.

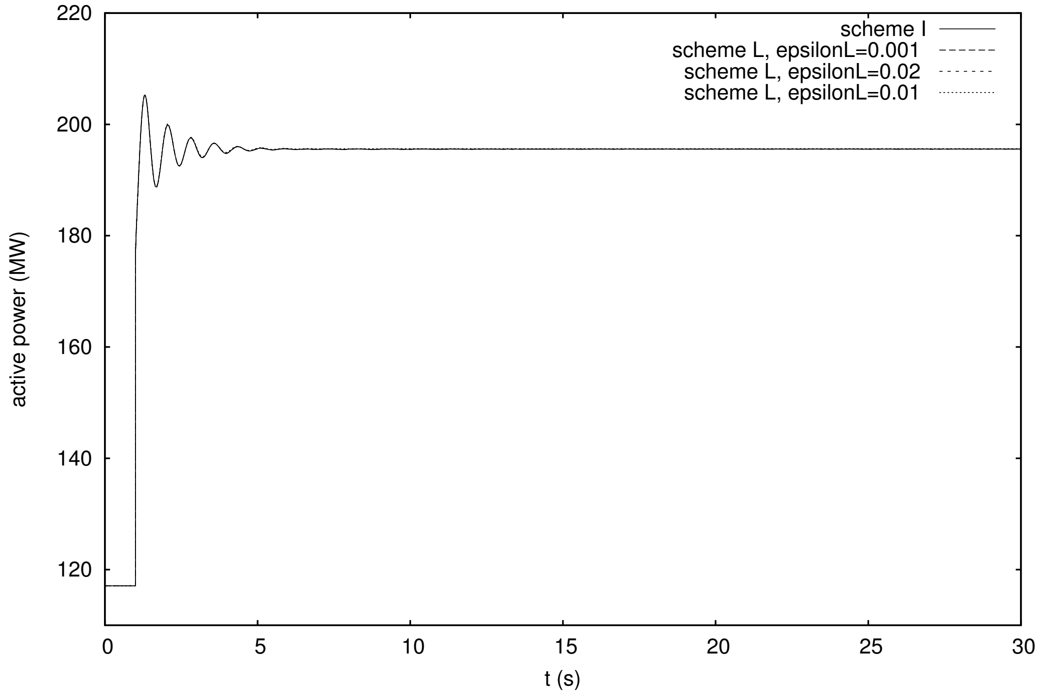


Figure 5.3: Case N1: active power flow in one circuit of line 1013-1014

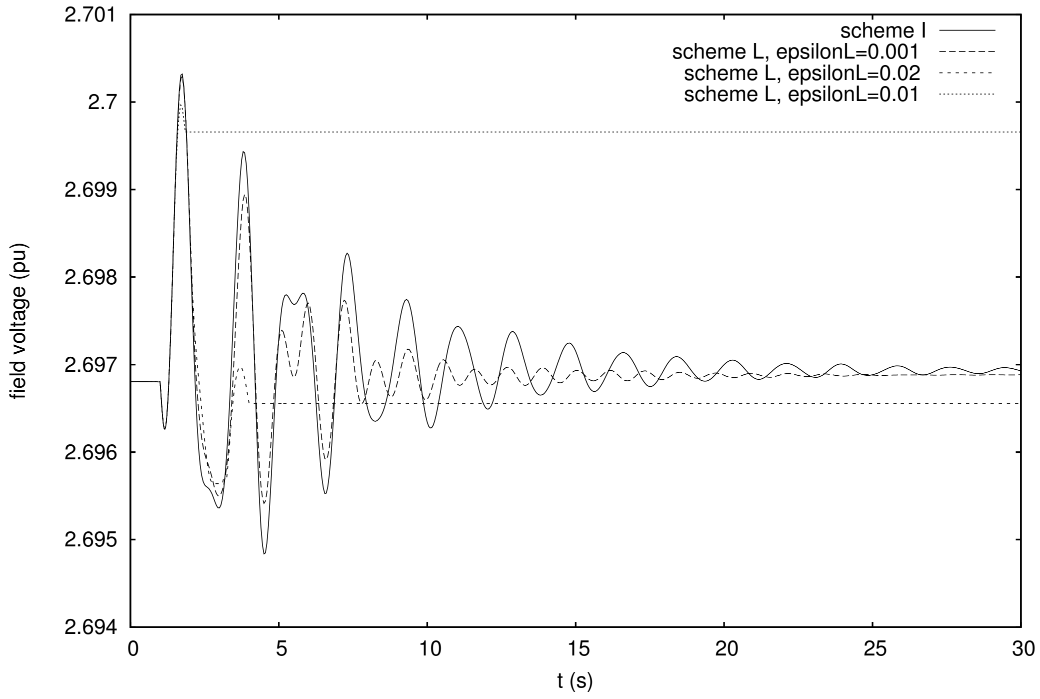


Figure 5.4: Case N1: field voltage of generator g15

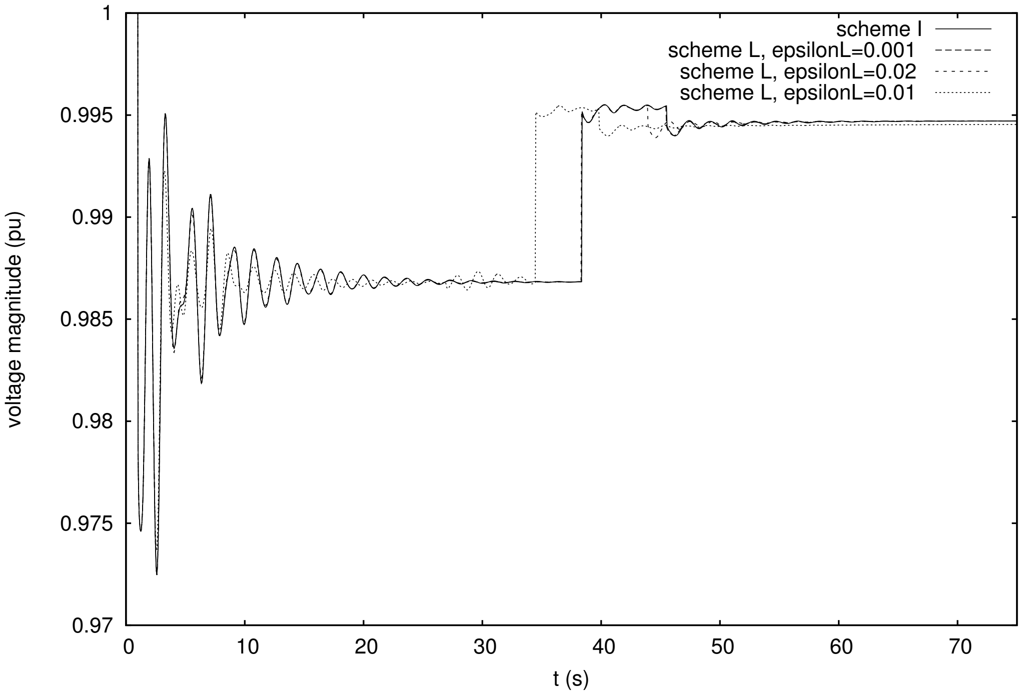


Figure 5.5: Case N2: voltage at distribution bus fed by bus 4043

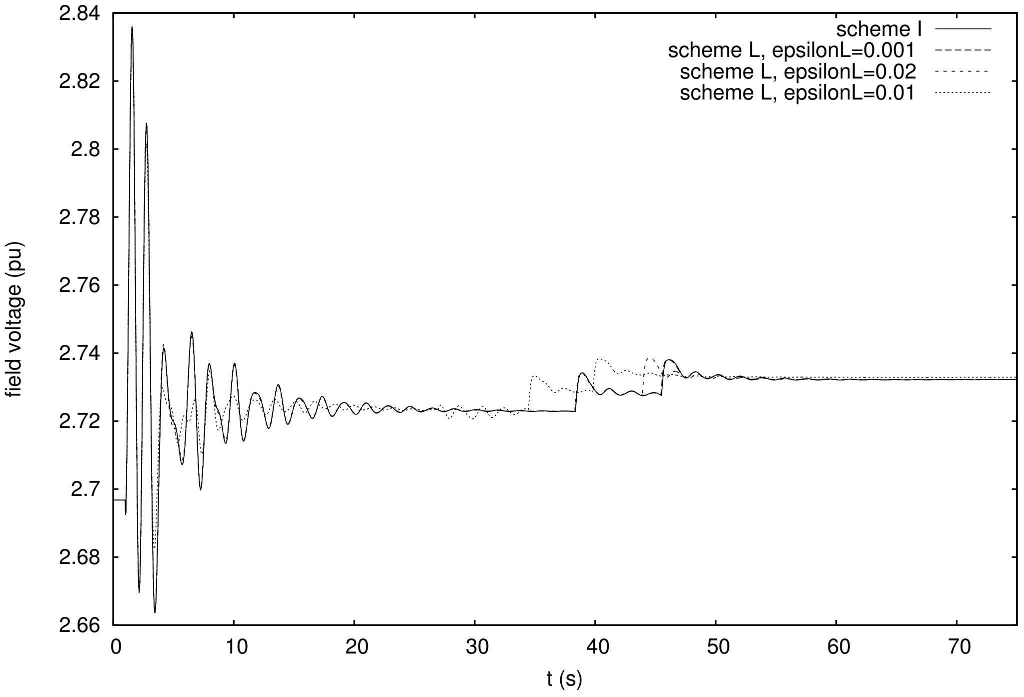


Figure 5.6: Case N2: field voltage of generator g15

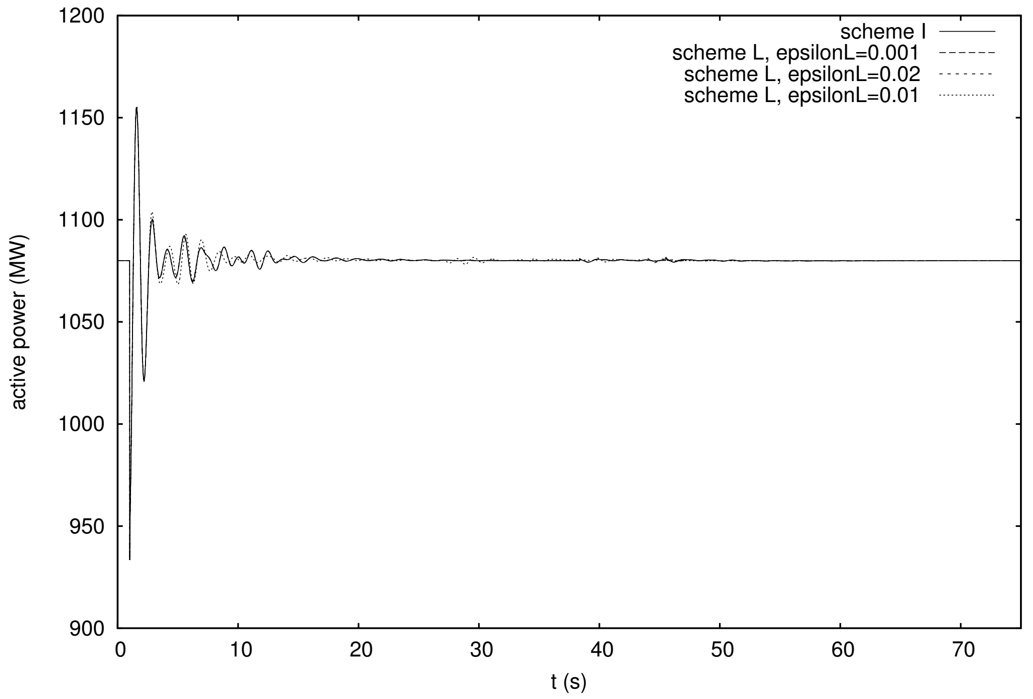


Figure 5.7: Case N2: active power produced by generator g15

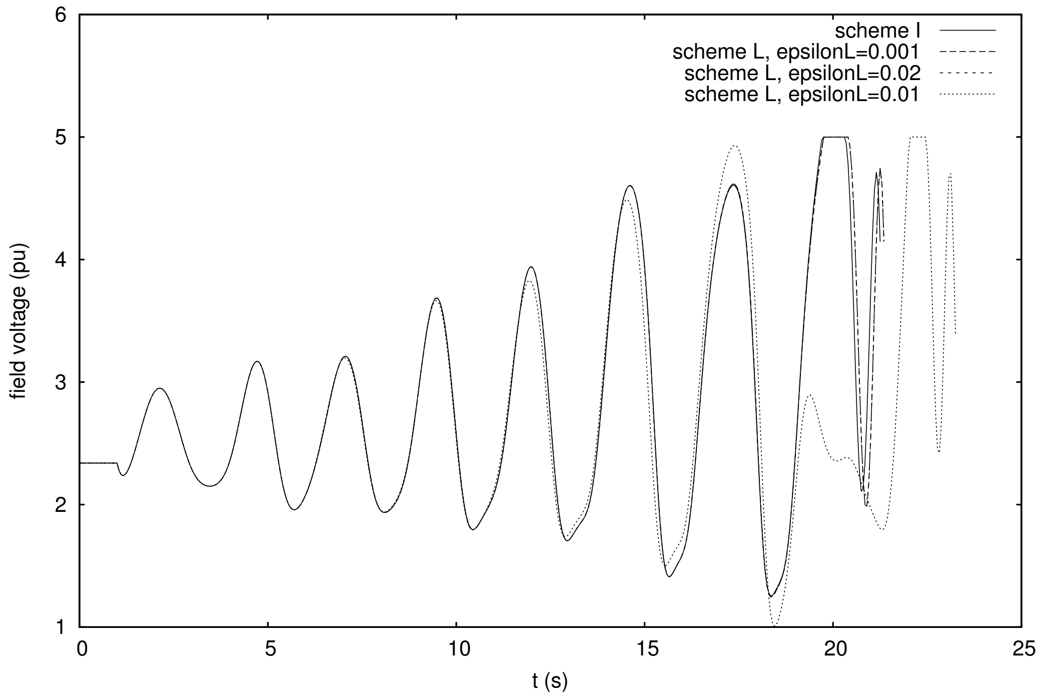


Figure 5.8: Case N4: field voltage of generator g17

The slightly lower accuracy when the system undergoes a large deviation also shows up in slightly larger DTW distances. For instance, in case N4, Fig. 5.8 shows the field voltage of generator g17 which loses synchronism towards the end of the simulation. In particular, the output obtained with $\epsilon_L = 0.01$, shows the loss of synchronism delayed by one period of rotor oscillation: it must be stressed though that the outcome of the simulation is the same as in the benchmark.

Table 5.1: DTW distances

Fig.	ϵ_L	δ	μ	σ	Unit
5.3	0.001	0.0058	-0.0007	0.0058	MW
	0.002	0.0134	-0.0101	0.0088	MW
	0.01	0.0823	0.0740	0.0360	MW
5.4	0.001	0.0002	-0.0001	0.0002	pu
	0.002	0.0005	-0.0002	0.0005	pu
	0.01	0.0025	0.0022	0.0011	pu
5.5	0.001	0.0000	-0.0000	0.0000	pu
	0.002	0.0001	-0.0000	0.0001	pu
	0.01	0.0004	-0.0000	0.0004	pu
5.6	0.001	0.0001	0.0000	0.0001	pu
	0.002	0.0003	0.0000	0.0003	pu
	0.01	0.0024	0.0000	0.0024	pu
5.7	0.001	0.0185	-0.0011	0.0184	MW
	0.002	0.1230	0.0021	0.1230	MW
	0.01	0.5135	0.0049	0.5136	MW
4.12	0.001	0.0001	0.0000	0.0001	pu
	0.002	0.0001	0.0000	0.0001	pu
	0.01	0.0001	0.0000	0.0001	pu
4.13	0.001	0.0000	0.0000	0.0000	pu
	0.002	0.0000	0.0000	0.0000	pu
	0.01	0.0000	0.0000	0.0000	pu
4.14	0.001	0.0211	-0.0033	0.0208	pu
	0.002	0.0212	-0.0033	0.0209	pu
	0.01	0.2642	0.0391	0.2616	pu
4.15	0.001	0.0002	0.0000	0.0002	pu
	0.002	0.0002	0.0000	0.0002	pu
	0.01	0.0008	0.0002	0.0008	pu
4.16	0.001	0.0000	0.0000	0.0000	pu
	0.002	0.0001	0.0000	0.0001	pu
	0.01	0.0002	-0.0000	0.0002	pu

5.5 Localization results on Hydro-Québec system

In this section, localization techniques are applied to the scenari described in Section 4.7. The gain in computing time introduced by localization is somewhat limited on this "medium size" model of a system which operates isolated and is subject to large disturbances. The outputs have been obtained with a localized dishonest Newton scheme, denoted "L" in the plots, using different latency tolerances ϵ_L. Localization has not been used during the first three seconds after a user-imposed disturbance. More details about this and the other solvers are given in Appendix A.5.

5.5.1 Case Q2

Figure 5.9 shows the speed deviation of the COI, after the loss of active power generation. The curves computed with scheme L are increasingly closer to the benchmark for smaller ϵ_L.

5.5.2 Case Q4

Case Q4 involves the long-term evolution of the system under the antagonistic effects of MAIS and LTCs. The voltage at transmission bus 702 is provided in Fig. 5.10.

The differences between the various simulated outputs are due to the marginally delayed switchings of the first reactor, located at bus 715, as already discussed in Section 4.7.4. These differences are definitely negligible for practical applications. The curve computed by using the smallest tolerance ϵ_L is indistinguishable from the benchmark though.

5.5.3 Computational effort

Table 5.2 gives the CPU times to evaluate the speed-up brought by scheme L with various latency tolerances ϵ_L in all scenari. It confirms that some gain can be obtained, especially in long-term cases. The largest advantages of scheme L, however, will be shown in the next section, on the PEGASE system, whose size allows a more profitable exploitation of localization. The CPU times of scheme A are reported for comparison, since scheme L, for $\epsilon_L \to 0$, is identical to scheme A.

While Cases Q2, Q3 and Q4 show an increasing speed-up for increasing ϵ_L, as expected, Cases Q1 and Q5 offer some perspective over the limits of application of localization.

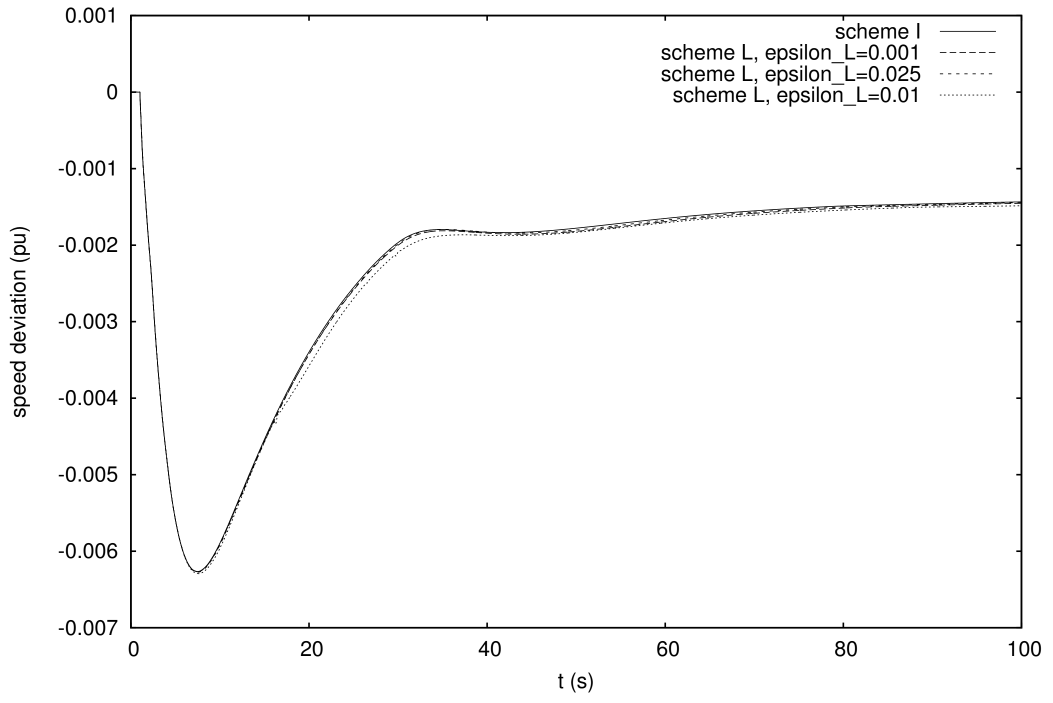


Figure 5.9: Case Q2: deviation of COI speed

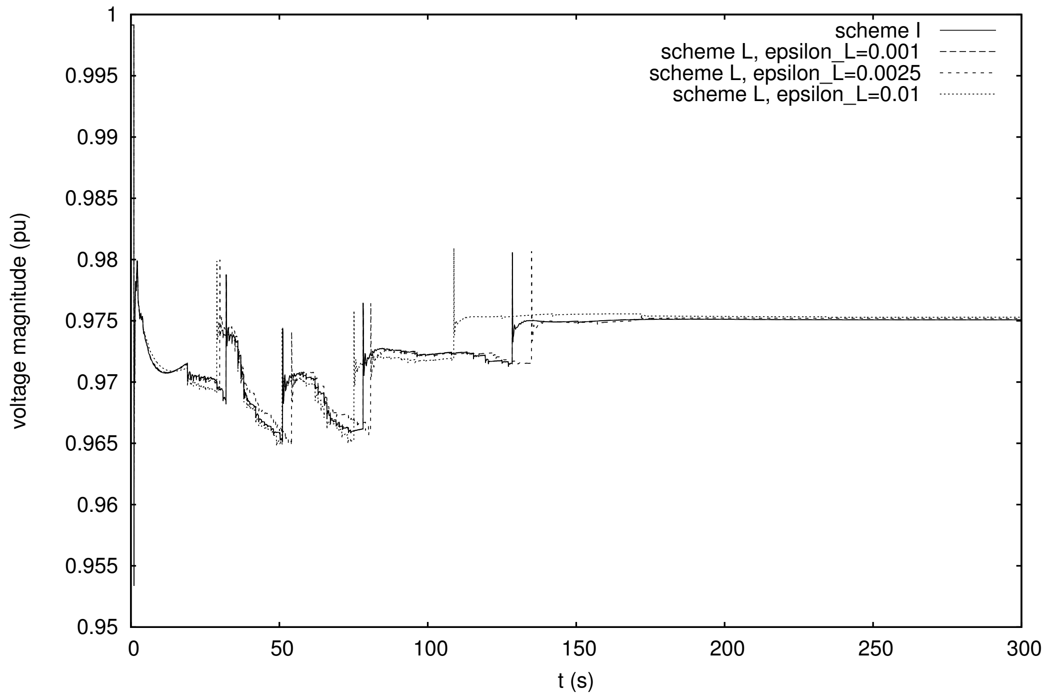


Figure 5.10: Case Q4: voltage at transmission bus 702

Table 5.2: CPU times (s)

Case	Scheme I	Scheme A	Scheme L $\epsilon_L = 0.001$	Scheme L $\epsilon_L = 0.0025$	Scheme L $\epsilon_L = 0.01$
Q1	11.11	5.93	6.04	5.74	5.72
Q2	35.84	11.91	11.06	9.46	8.30
Q3	27.89	13.16	12.19	11.45	9.93
Q4	60.16	30.55	28.51	26.96	23.43
Q5	61.34	34.36	32.80	31.26	32.19

5.5.4 Other cases

Case Q1 shows even a degradation of performance, with respect to scheme A, when the strictest latency tolerance is used: in fact this is a short-term scenario simulating a fault and the subsequent transients during which, most probably, only a really small part of the system, if any, can be considered latent. For increasing tolerances, though, some speed-up is found.

Case Q5, on the other hand, shows that increasing ϵ_L brings a more efficient simulation up to some threshold, where the slow-down due to approximations brought by latency eventually offset the speed-up. For this case for instance, a value of ϵ_L larger than 0.0025 is not useful.

5.6 Localization results on PEGASE system

In this section, localization techniques are applied to the scenari described in Section 4.8. Expectedly, on the PEGASE system, the gain in computing time brought by localization is the highest among the three systems considered in this work. The outputs have been obtained with a localized dishonest Newton scheme, denoted "L" in the plots, using different latency tolerances ϵ_L. Localization has not been used during the first half second seconds after a user-imposed disturbance. More details about this and the other solvers are given in Appendix A.5.

5.6.1 Cases P1, P2 and P3

The voltage evolutions at bus F0322411 are shown in Figs. 5.11, 5.12, 5.13 and 5.14 for the three scenari. As expected, the approximation increases with the latency tolerance, but even the highest latency tolerance brings acceptable results for many practical applications.

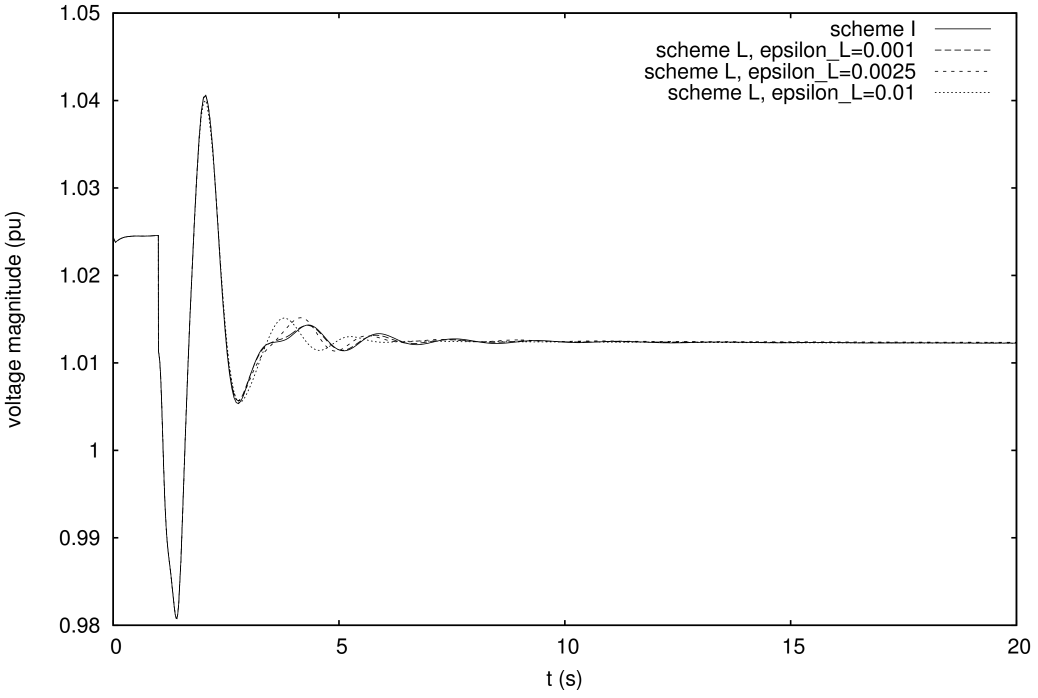


Figure 5.11: Case P1: voltage at transmission bus F0322411

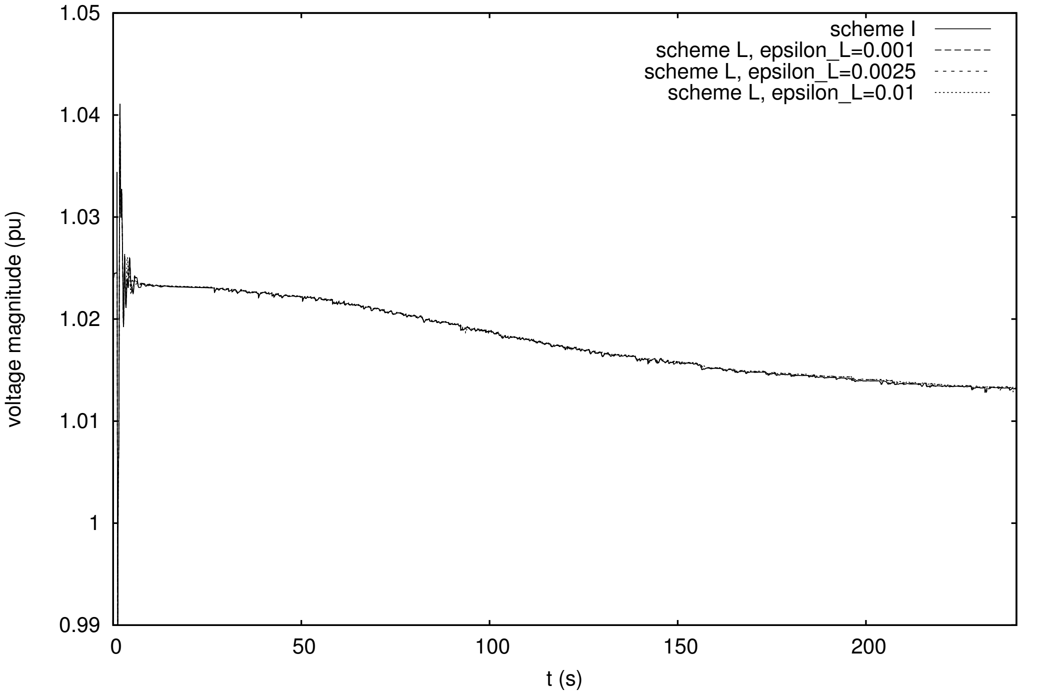


Figure 5.12: Case P2: voltage at transmission bus F0322411

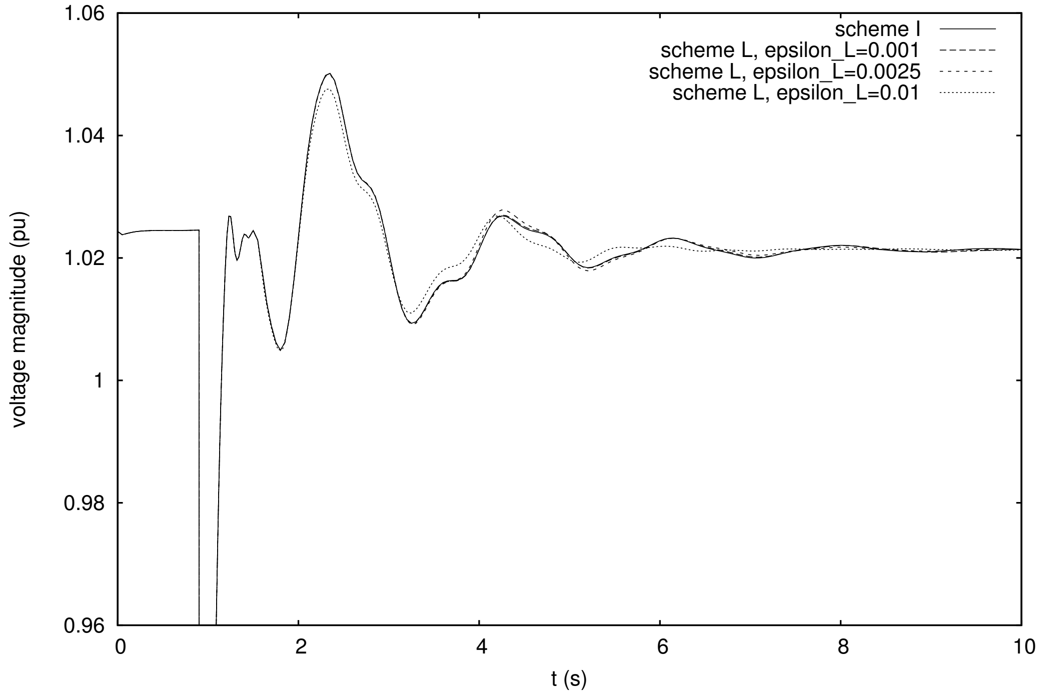


Figure 5.13: Case P3: zoom of the short-term voltage evolution at transmission bus F0322411

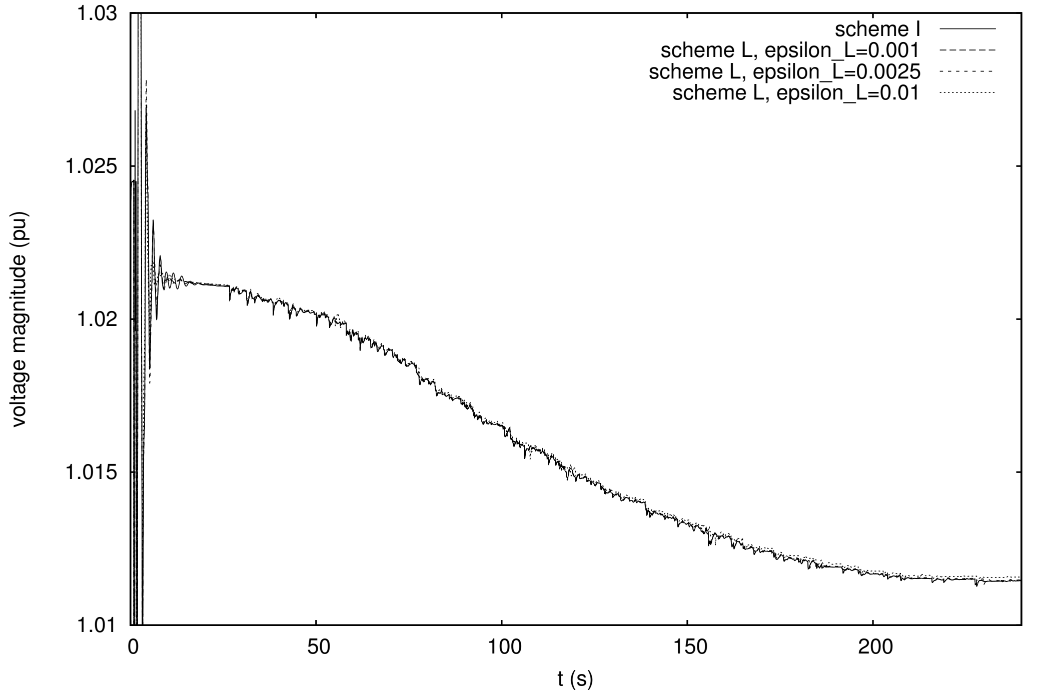


Figure 5.14: Case P3: zoom of the long-term voltage evolution at transmission bus F0322411

Table 5.3: Case P1: CPU times (s)

	Scheme I		Scheme A	Scheme L $\epsilon_L = 0.001$	Scheme L $\epsilon_L = 0.0025$	Scheme L $\epsilon_L = 0.01$
total	90.29	total	32.81	27.49	24.45	15.10
factorizing $\mathbf{J}$	57.51	factorizing $\tilde{\mathbf{D}}$	0.69	0.97	0.76	0.71
solving for $[\boldsymbol{\Delta}\mathbf{V}\ \ \boldsymbol{\Delta}\mathbf{x}]^T$	11.19	solving for $\boldsymbol{\Delta}\mathbf{V}$	0.82	1.06	1.06	0.69
evaluating $[\mathbf{g}\ \ \mathbf{f}]^{\mathbf{T}}$	13.11	evaluating $\mathbf{g}$	1.97	1.75	2.07	1.81
		factorizing $\mathbf{A}_i$ *	2.84	1.34	1.23	1.04
		solving for $\boldsymbol{\Delta}\mathbf{x}_i$ *	6.51	5.26	4.66	2.43
		evaluating $\mathbf{f}_i$ *	14.75	12.67	10.28	5.12

* $i = 1, \ldots, n$

Table 5.4: Case P1: Number of operations

	Scheme I		Scheme A	Scheme L $\epsilon_L = 0.001$	Scheme L $\epsilon_L = 0.0025$	Scheme L $\epsilon_L = 0.01$
factorizing $\mathbf{J}$	446	factorizing $\tilde{\mathbf{D}}$	25	32	26	25
solving for $[\boldsymbol{\Delta}\mathbf{V}\ \ \boldsymbol{\Delta}\mathbf{x}]^T$	1393	solving for $\boldsymbol{\Delta}\mathbf{V}$	1031	1158	1009	844
evaluating $[\mathbf{g}\ \ \mathbf{f}]^{\mathbf{T}}$	1891	evaluating $\mathbf{g}$	2326	2320	2429	2012
		factorizing $\mathbf{A}_i$ *	26	16	15	13
		solving for $\boldsymbol{\Delta}\mathbf{x}_i$ *	792	594	478	255
		evaluating $\mathbf{f}_i$ *	2321	1805	1443	719

* average per injector

Table 5.5: Case P2: CPU times (s)

	Scheme I		Scheme A	Scheme L $\epsilon_L = 0.001$	Scheme L $\epsilon_L = 0.0025$	Scheme L $\epsilon_L = 0.01$
total	470.00	total	220.21	181.35	132.41	75.34
factorizing $\mathbf{J}$	207.58	factorizing $\tilde{\mathbf{D}}$	0.75	1.05	0.96	0.79
solving for $[\mathbf{\Delta V} \ \ \mathbf{\Delta x}]^T$	90.00	solving for $\mathbf{\Delta V}$	7.42	7.39	8.25	8.11
evaluating $[\mathbf{g} \ \ \mathbf{f}]^{\mathbf{T}}$	111.86	evaluating $\mathbf{g}$	14.04	14.92	14.66	13.46
		factorizing $\mathbf{A}_i$ *	2.51	1.13	1.25	0.98
		solving for $\mathbf{\Delta x}_i$ *	41.53	33.07	20.73	7.57
		evaluating $\mathbf{f}_i$ *	117.25	88.07	54.76	19.61

* $i = 1, \ldots, n$

Table 5.6: Case P2: Number of operations

	Scheme I		Scheme A	Scheme L $\epsilon_L = 0.001$	Scheme L $\epsilon_L = 0.0025$	Scheme L $\epsilon_L = 0.01$
factorizing $\mathbf{J}$	1597	factorizing $\tilde{\mathbf{D}}$	27	36	32	28
solving for $[\mathbf{\Delta V} \ \ \mathbf{\Delta x}]^T$	11106	solving for $\mathbf{\Delta V}$	8514	8620	8476	8167
evaluating $[\mathbf{g} \ \ \mathbf{f}]^{\mathbf{T}}$	15988	evaluating $\mathbf{g}$	17160	18104	18310	17725
		factorizing $\mathbf{A}_i$ *	22	14	16	13
		solving for $\mathbf{\Delta x}_i$ *	5634	3521	2089	718
		evaluating $\mathbf{f}_i$ *	19044	12212	7192	2305

* average per injector

Table 5.7: Case P3: CPU times (s)

	Scheme I		Scheme A	Scheme L $\epsilon_L = 0.001$	Scheme L $\epsilon_L = 0.0025$	Scheme L $\epsilon_L = 0.01$
total	438.21	total	232.56	209.18	174.35	102.48
factorizing $\mathbf{J}$	179.40	factorizing $\tilde{\mathbf{D}}$	0.93	0.99	1.03	1.14
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	88.93	solving for $\mathbf{\Delta V}$	8.19	7.87	7.83	7.65
evaluating $[\mathbf{g}\ \ \mathbf{f}]^{\mathbf{T}}$	111.34	evaluating $\mathbf{g}$	13.91	13.94	14.32	14.53
		factorizing $\mathbf{A}_i$ *	4.49	3.26	2.56	3.39
		solving for $\mathbf{\Delta x}_i$ *	47.35	38.74	29.69	13.69
		evaluating $\mathbf{f}_i$ *	119.44	106.06	83.23	34.00

* $i = 1, \ldots, n$

Table 5.8: Case P3: Number of operations

	Scheme I		Scheme A	Scheme L $\epsilon_L = 0.001$	Scheme L $\epsilon_L = 0.0025$	Scheme L $\epsilon_L = 0.01$
factorizing $\mathbf{J}$	1374	factorizing $\tilde{\mathbf{D}}$	32	34	34	35
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	10975	solving for $\mathbf{\Delta V}$	8556	8547	8362	8368
evaluating $[\mathbf{g}\ \ \mathbf{f}]^{\mathbf{T}}$	15946	evaluating $\mathbf{g}$	17020	17052	17337	18303
		factorizing $\mathbf{A}_i$ *	40	35	27	42
		solving for $\mathbf{\Delta x}_i$ *	5757	4744	3429	1290
		evaluating $\mathbf{f}_i$ *	19285	15991	11572	4278

* average per injector

5.6.2 Computational effort

Tables 5.3 and 5.4 detail the computing times obtained and the number of operations performed in case P1 using the benchmark scheme I and scheme L, respectively. Similar results for Case P2 are given in Tables 5.5 and 5.6, for Case P3 in Tables 5.7 and 5.8. As expected, the computational effort is reduced with the increase of the latency tolerance. All these tables show that the highest gain relates to the evaluations and the solutions of the injectors: indeed, with respect to scheme A, several injectors are latent for some part of the simulation. The gain is higher in long-term cases, as also expected. The decrease in injector factorization is milder, as it is the most "active" injectors which need to be factorized the most, in both schemes, and are thus not affected by latency. On the network, on the other hand, the gain is imperceptible, if not negative in some cases: indeed, the network cannot be latent, so no gain is to be expected, while a slight performance degradation is plausible due to the approximations brought by the latent injectors.

Figure 5.15 reproduces the plot previously presented in Fig.4.30 for ease of comparison with the following three figures. This plot shows the number of injector solutions per time step in scheme A. The lower bound of the plotted values is $n = 10694$, since at least one iteration per each of the 10694 injectors is performed at each step.

Figures 5.16, 5.17 and 5.18 show the corresponding number of injector solutions per step in scheme L, for the three different latency tolerances. For increasing ϵ_L there is a dramatic decrease of the number of solutions. The lower bound corresponds to the number of injectors which cannot become latent, because of a discrete jump in their equations or because of too large a current variation with respect to their linearization current, which is often the current at the beginning of the simulation for injectors which experienced strong dynamic activity. This lower bound is around 9000 for $\epsilon_L = 0.001$, 7000 for $\epsilon_L = 0.0025$ and 2000 for $\epsilon_L = 0.01$.

The yield in computing time brought by localization may be easily increased by relaxing the too strict criteria that lead to the above mentioned lower bound. For instance, an injector which experienced discrete jumps could be considered a candidate for latency after some time has passed since the last event, and equivalently, an injector which underwent large changes could be linearized around a subsequent point, provided that some conditions are fulfilled, such as the rate of change of its current measured over a long period of time. Such fine tuning, however, has not been considered in this thesis.

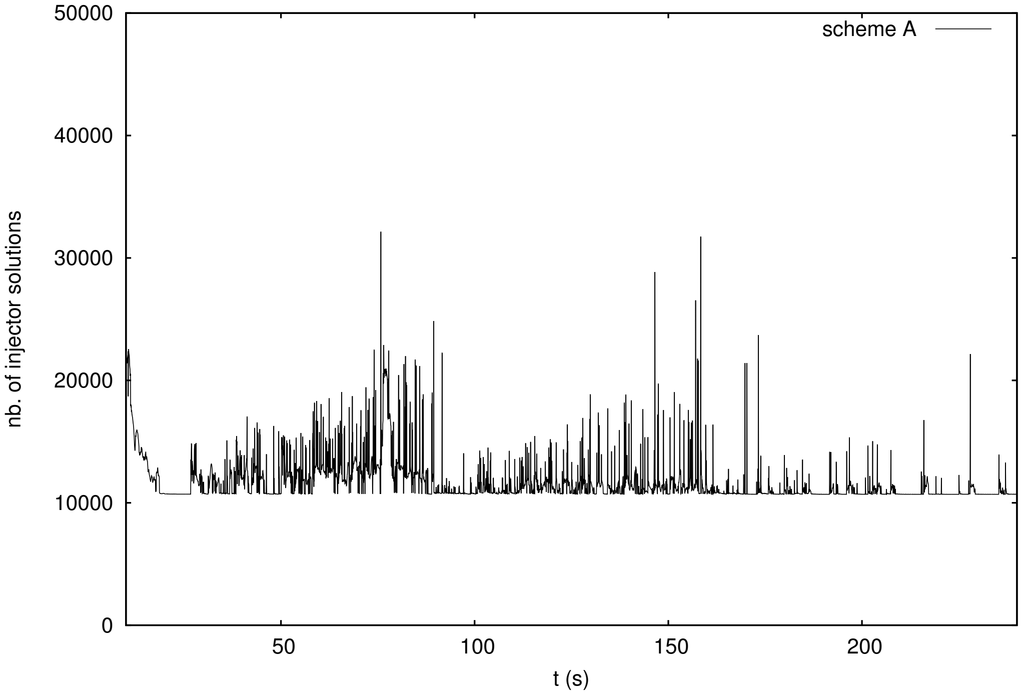


Figure 5.15: Case P3: number of injector solutions per step (scheme A)

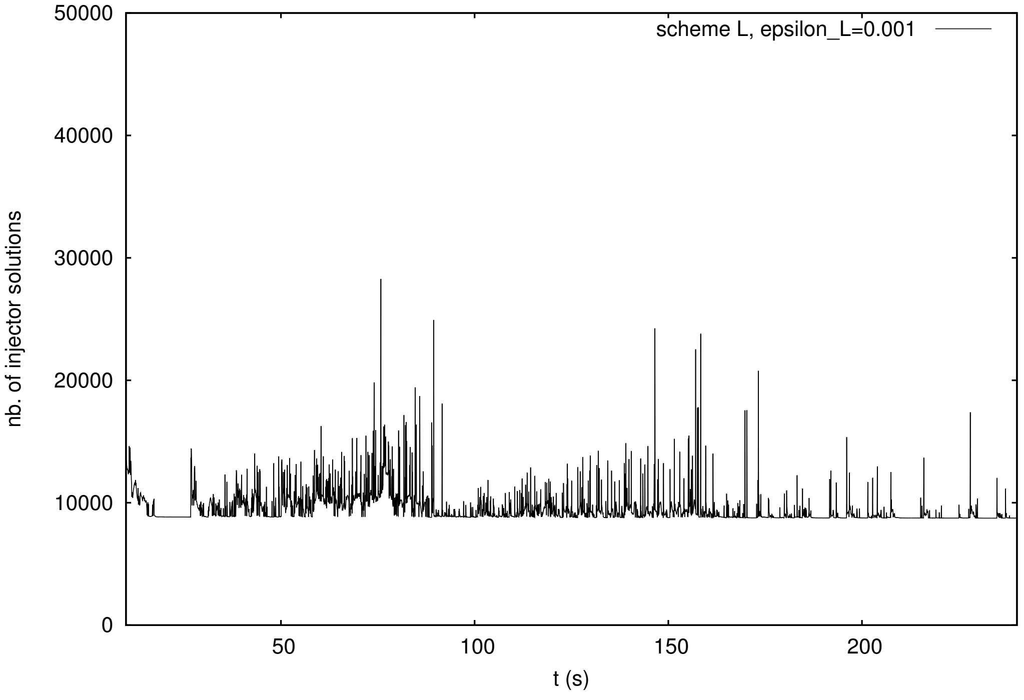


Figure 5.16: Case P3: number of injector solutions per step (scheme L, $\epsilon_L = 0.001$)

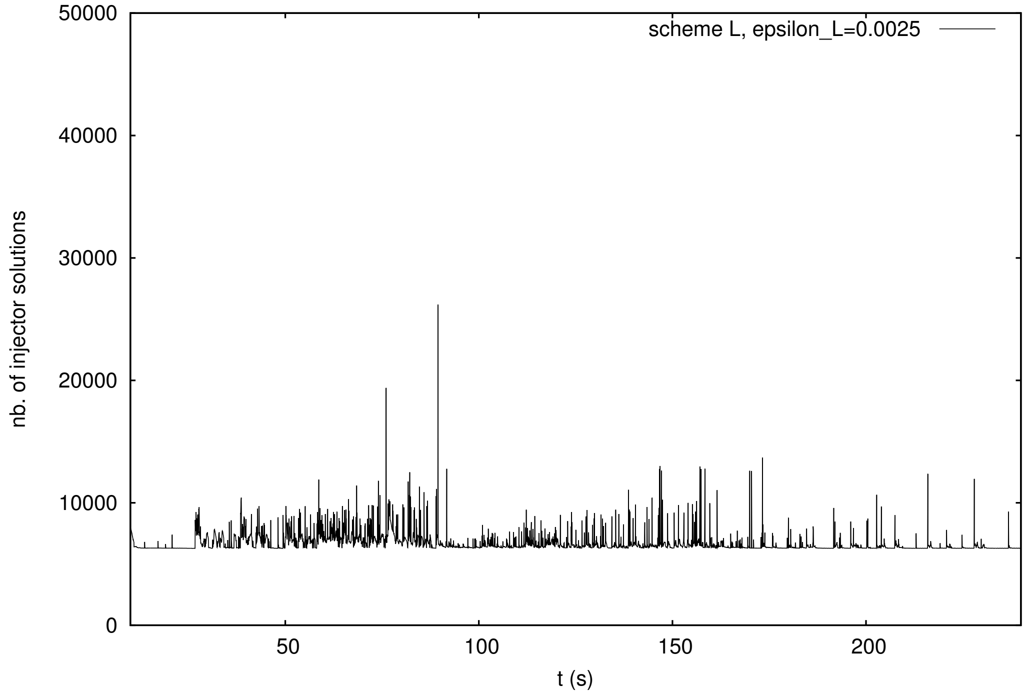


Figure 5.17: Case P3: number of injector solutions per step (scheme L, $\epsilon_L = 0.0025$)

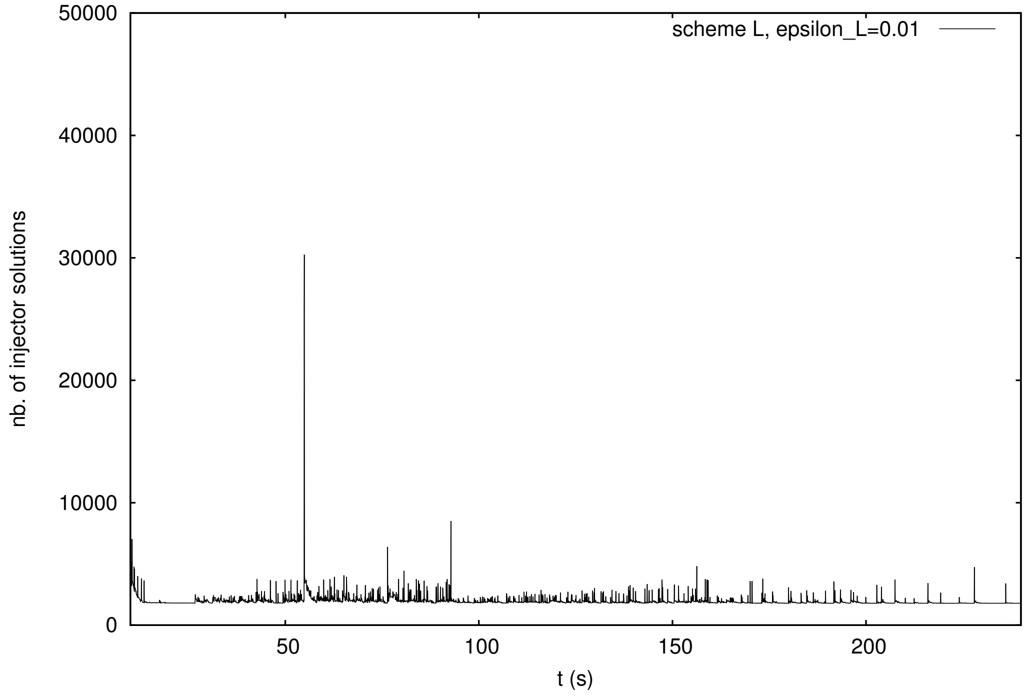


Figure 5.18: Case P3: number of injector solutions per step (scheme L, $\epsilon_L = 0.01$)

5.7 Chapter conclusion

The most important contribution of this chapter is the localization applied to the decomposition of the dishonest Newton method algorithm: it brings a speed-up ranging between 2 and 3, with respect to scheme A, and between 4 and 6, with respect to scheme I, on the PEGASE test system. The speed-up on the Hydro-Québec system is not so important, on the other hand, due to the relatively smaller size of the system.

Chapter 6

Relaxing accuracy in time

O wer weiß, was in der Zeiten Hintergrunde schlummert? [1]

This chapter starts with a review of methods which have been used to solve the power system dynamic problem with a varying level of accuracy: the principles of Quasi-Steady-State method are also briefly described. The concept of time-averaging of the continuous dynamics is introduced and the role of the discrete part of the model is analyzed in more detail. Modelling and solving guidelines have been established and safeguards against cycling between discrete states are considered. Relevant results on the three test systems close this chapter showing that the computational effort of the proposed approach is greatly reduced with respect to a detailed simulation, while the fast transients of the simulated outputs can be gradually filtered out with the increase of the step size.

6.1 Temporal accuracy relaxation in literature

With temporal relaxation of accuracy we refer to the fact that some fast components of the response may not be of interest, especially in long-term simulations, and could thus be partially or totally omitted: this can be achieved through the use of a different solver or through a simplification of the model.

[1] *Oh who can see what sleeps in time, behind the face of things?* Johann Christoph Friedrich von Schiller, in *Don Karlos, Infant von Spanien*. English translation by H. Collier Sy-Quia and P. Oswald in *Don Carlos*.

The complete omission of fast dynamics through model simplification is the heart of the Quasi-Steady-State (QSS) approximation of long-term dynamics [VCJMP95]. In the QSS approximation, Equations (2.4) are split into slow and fast components, $\mathbf{w}_s$ and $\mathbf{w}_f$, respectively:

$$\dot{\mathbf{w}}_s = \boldsymbol{\psi}_{ds}(\mathbf{y}, \mathbf{w}_f, \mathbf{w}_s, \mathbf{z}, \mathbf{V}, \mathbf{I}, \omega_{ref}) \tag{6.1}$$

$$\dot{\mathbf{w}}_f = \boldsymbol{\psi}_{df}(\mathbf{y}, \mathbf{w}_f, \mathbf{w}_s, \mathbf{z}, \mathbf{V}, \mathbf{I}, \omega_{ref}) \tag{6.2}$$

The fast components are considered at equilibrium and Eq. (6.2) is replaced by

$$\mathbf{0} = \boldsymbol{\psi}_{df}(\mathbf{y}, \mathbf{w}_f, \mathbf{w}_s, \mathbf{z}, \mathbf{V}, \mathbf{I}, \omega_{ref}) \tag{6.3}$$

A more recent version of the QSS algorithm also considered frequency dynamics [GLVC05]. While the main advantage of the QSS approach is a speed-up of orders of magnitude with respect to detailed simulation of the full model [VCM97], some drawbacks limit the scope of this method:

- the separation of slow and fast components might not be possible for a complex black box model. Automatic model simplification methods that can deal with realistic, hybrid power system models are yet to be developed. More on this problem will come later in this section;
- the approach itself needs a different model than the one used for detailed simulation, which requires additional overhead for creation, validation and maintenance. The creation of a simplified model also requires a good knowledge of the process itself, so it is not suitable for black box models;
- a long-term degradation of system operating conditions may result in motor stalling or loss of synchronism [VCV08]. In this case, a fast instability of the full model (2.4) appears as an early singularity of the QSS model (6.1,6.3) that gives little information about the nature of the problem[1];
- following a large disturbance, the system may become unstable in the short-term period and, hence, even not enter the long-term period. Detecting this situation requires to couple detailed and QSS simulations, as it will be seen in greater detail in Chapter 7;

[1]In practical applications [VCM97], however, such fast instabilities were found to take place at low voltages, where the system evolution was anyway unacceptable in the context of DSA.

- a more accurate simulation may require to retain some of the fast dynamics.

Another algorithm that relies on the time-scale separation of power system phenomena to solve the different parts of the system with different levels of details (i.e. step sizes) is the variable partitioning multi-rate method [CC08], which adapts the subdomains according to the "level of activity" of the variables involved. It is our perception that the separation of variables in different subsets might be cumbersome, if not impossible at all, for realistic power system models. While in classical multi-rate methods the subdivision of differential variables is decided beforehand, reference [CC08] proposes heuristics for the separation of algebraic variables in between slow and fast subsets, a delicate operation insofar as an algebraic dynamics is *infinitely fast* by definition, and the algebraic variables can only be clustered based on their interaction with other variables.

In the author's opinion, though, the main problem in the use of QSS or multi-rate approaches lies in the subdivision of the differential variables themselves: in fact, while the open-loop "time constant" of a process can be easily determined, it is the closed-loop one which actually matters. And if the closed-loop includes the network, as it is often the case in power systems studies, this computation is not practical, as it involves the elimination of a large number of algebraic variables [1].

6.2 Time-averaging of continuous dynamics

In nonlinear dynamical systems theory, the procedure of replacing a vector field by its average (over time or an angular variable) in order to obtain asymptotic approximations to the original system is called *averaging* [SVM07]. In spite of the mathematical niceties of this theory, its application to power system dynamics might be hindered by several factors: (i) the averaging period is unknown in advance, (ii) the averaging period varies widely, since the oscillations experienced in power systems belong to very different time-scales and (iii) the physical meaning of averaging systems which exhibit hybrid continuous-discrete behavior is not straightforward.

[1] In 2011, after a seminar given in Liège by a Control specialist, the author once raised the question on how to discriminate between fast and slow variables in large differential-algebraic systems in a feasible way (usually the subsets are already given as input data in test problems). The answer was to eliminate the algebraic variables. However, a practical way to eliminate algebraic variables on a system slightly larger than academic toy problems, is, in the author's opinion, still to be found. The gist of this brief autobiographical argument is that the problem of variable partitioning in black box models seems still to remain largely unsolved, even outside the power system community.

Nonetheless, the concept behind this mathematical theory, i.e. the possibility of getting averaged trajectories, is highly appealing. The idea behind *time-averaging* is to perform time simulation with L-stable methods using large enough steps so that some fast dynamics are discarded while concentrating on the average evolution. In the general case, no theoretical proof can be given that the obtained solution coincides with an averaged form of the original trajectory. However, a good interpretation of the results obtained in practice is provided by the following limit:

$$\lim_{h\to\infty} \frac{\left(-\mathbf{w}(t_j) + \sum_{\ell=1}^{o} \gamma_\ell \, \mathbf{w}(t_{j-\ell}) + h\,\beta\,\dot{\mathbf{w}}(t_j)\right)}{h} = \beta\,\dot{\mathbf{w}}(t_j) \tag{6.4}$$

Equation (6.4) is the limit of the BDF (3.4), divided by h, for $h \to \infty$. After the algebraized equations have been solved, the numerator in the left-hand side of (6.4) is brought to zero. This means that at the limit the derivative $\dot{\mathbf{w}}(t_j)$ is zero, i.e. the simulated response of the system is at equilibrium. A wider interpretation of this result is that the BDF formula tends to the underlying system equilibrium as the step size is increased.

Conversely, non L-stable methods may produce spurious oscillations if they are used with a step size significantly larger than the smallest time constant. Figure 6.1 shows a simple example of application of respectively an L-stable (BEM) and a non L-stable (TM) method to the test equation (3.7) with $\lambda = 10$, $w(0) = 1$, using a time step size h of 1 s, i.e. ten times larger than the time constant 0.1 s. The output computed with BEM shows a consistent delay and reaches equilibrium quite soon. Conversely, TM returns an unacceptable purely numerical oscillatory evolution.

The approach followed in this work consists of simulating the original, detailed model with a solver that allows "filtering" out the fast dynamics by taking intentionally large steps. The most significant advantage of this approach is that no simplification is performed of the models, i.e. *they are the same as in detailed simulation*. Significant reductions of the computational effort can be expected from the use of large steps, approximately given by the ratio between the step size used in detailed simulation and the larger step used for time-averaging.

The property of L-stable methods that allows integrating with steps significantly larger than the smallest system time constants is known in numerical analysis [AP98, CK06]. The use of BDF for their L-stable properties in power system dynamics can be traced back to [ABJ94], while more recent uses are documented in [FVC09, WC11]. The method is also

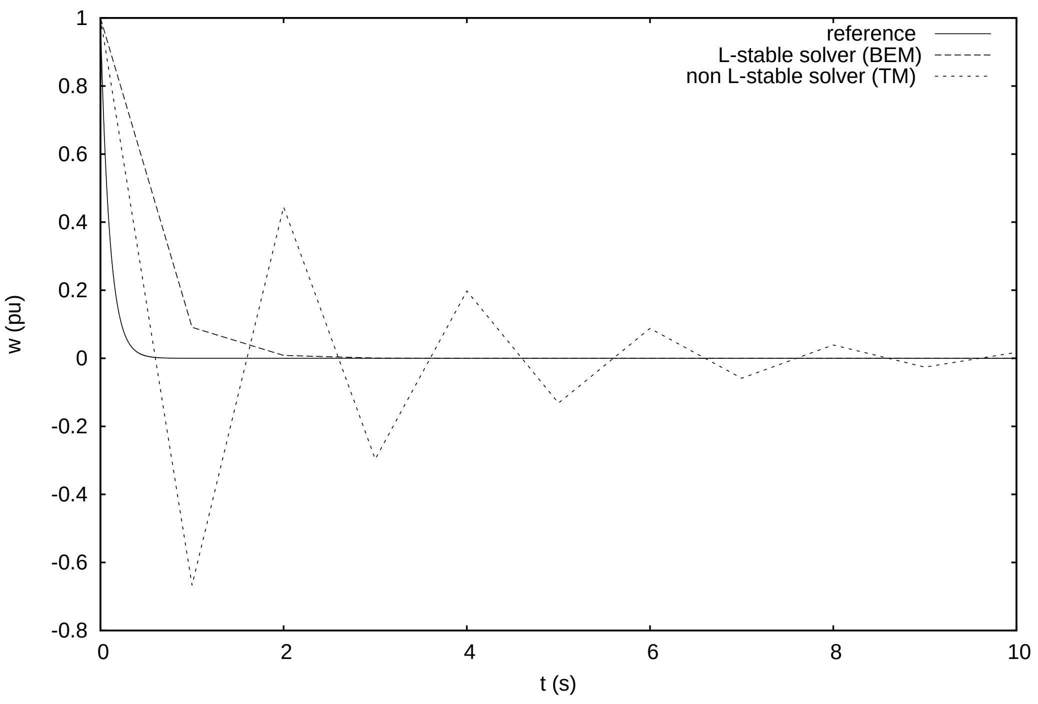


Figure 6.1: Solutions of $\dot{w} = -10\ w$ with different methods

used in a wide variety of other engineering problems, e.g. cloth dynamic simulation in computer graphics [BW98].

On the other hand, the price to pay with L-stable methods is that they overdamp oscillations whose period is comparable to the step size. This undesired hyper-stability property limits the step size used for detailed simulation. It suggests to use higher order BDF, since the hyper-stability decreases with the order of the formula used. Unfortunately, BDF of order larger than 2 do not exhibit the equally important property of A-stability.

Note the the hyper-stability property, which represent a limitation of the method for detailed simulation of power system dynamics, has been used in electromagnetic transient simulations as a feature to suppress the numerical oscillations arising from strong discontinuities [ML89].

6.3 Handling of discrete dynamics

For a large class of continuous-time dynamics, in so far as accuracy can be somewhat relaxed, the step size of an L-stable method is limited only by the convergence of the Newton iterations used to solve the algebraized equations [FVC09]. However, power system models are also hybrid [HP00] and the proper handling of discrete events requires caution when the time step is increased. To the author's knowledge, this aspect has been relatively less investigated in the literature; preliminary investigations were reported in [FCPVC11].

In the context of time-averaging, the following scheme is proposed for the *ex post* treatment of discrete events:

1. at time t, starting from the state vector $\mathbf{x}_t$, take a step of length h. Let $\mathbf{x}_{t+h}^{(1)}$ be the corresponding state vector;
2. at this time the jump conditions (2.13) are checked;
3. if it is detected that a state event has occurred in between t and $t+h$, $\mathbf{z}$ is changed accordingly, and the step from t to $t+h$ is repeated with the flow (2.12) corresponding to the new $\mathbf{z}$, yielding the new value $\mathbf{x}_{t+h}^{(2)}$ of the state vector;
4. steps 2 and 3 are repeated, updating the state vector, until no jump occurs any longer or a maximum number of jumps is reached. This yields the final state vector for the current step;
5. then, the simulation proceeds with a new step.

Newton iterations are performed to pass from $\mathbf{x}_t$ to $\mathbf{x}_{t+h}^{(j)}$ $(j = 1, 2, \ldots)$. It is essential to iterate until reaching a solution $\mathbf{x}_{t+h}^{(j)}$ accurate for the current flow, before checking the jump conditions (step 2). In other words, the flow must not be changed during the Newton iterations, otherwise, incorrect and/or unpredictable jumps can be experienced.

Figure 6.2 shows a classical example of hybrid dynamics, namely an integrator with a non-windup lower limit.

Let us define z as a discrete variable with value 1 if $x < 3$ and value 0 if $x = 1$. This system is described by the following continuous-time equation of type (2.12):

$$z\,\dot{x} = z\,(u - x) + (1 - z)\,(x - 1)$$

At $t = 0^-$ the system is at equilibrium and $x(0) = u(0^-) = 3$. At $t = 0^+$, $u(0^+) = 0$. The trajectories computed with different methods are plotted in Fig. 6.2 with

- solid line: closed-form exponential solution $x(t) = \max(3e^{-t}; 1)$;
- dotted line: BEM numerical solution computed with steps of 1 s and detailed handling of state events;

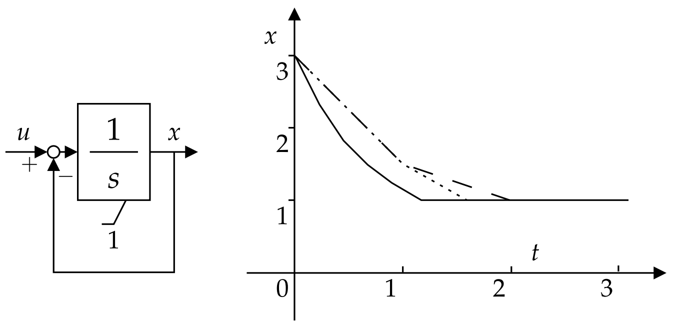


Figure 6.2: Integrator with non-windup limit, and example of response

- dashed line: BEM numerical solution computed with steps of 1 s and *ex post* handling of state events as we propose.

The corresponding discrete event is taking place at $t = ln(3) \simeq 1.1$ s, in the closed-form exponential solution. Let us stress that the closed-form solution is just provided as a reference, but cannot practically be computed in even slightly more complex systems.

The most important effect of events on dynamic simulations is that they force the solver to adjust the length of continuous-time evolution intervals in order to land over the event times, as explained in Section 3.5. This mechanism, in case of many events, leads to a reduction of the average step size (and hence an increase in the number of steps). Moreover, all (network and injector) Jacobians used in the Newton iterations have to be updated and re-factorized when h changes significantly (the latter is involved in the algebraized differential equations). Coming back to the example in Fig. 6.2, with the detailed handling, the state event should take place at 1.5 s: let us stress that this is dependent on the integration formula and on the discretization interval. The detailed handling of discrete events is accurate when the solution is performed with small integration steps. However, paradoxically, the *detailed* handling is not as *accurate* as the name would suggest if comparatively large steps are taken, as Fig. 6.2 shows: in fact, the delay in computing the switching time is due, in this example, to the large truncation error committed because of the large steps used.

With the *ex post* treatment of discrete events, conversely, the event is not precisely located in a specific instant, but it takes place in between the beginning and the end of a step, in the case of Fig. 6.2 in between 1 and 2 s. The system equations are satisfied at the start and final point of the step, and no attempt is made to identify the actual switching instant. It is believed that for the envisaged application this constitutes an acceptable approximation.

Another option for the treatment of discrete events is the interpolation technique, used for instance in electromagnetic transients simulation, especially where the computational effort constraints are tighter, namely in real time simulations [SLM^{+}00, Str04] or more recently in hardware-in-the-loop simulations [MFL^{+}05]. This technique consists in neither adapting the step (as in detailed simulation) nor in repeating the step (as in the proposed *ex post* treatment): it relies on interpolation and extrapolation to compute a valid approximation of the trajectory in presence of an event, without recomputing the system equations at the switching time. The application of this technique to the power system simulation under the phasor approximation though is not straightforward, for several reasons: (i) in black box models, the choice of the interpolation law is not obvious, (ii) in presence of many concurring or cascading events, the interpolations might not be valid.

In large systems, in the occurrence of many events, the *ex post* approach will show a speed-up compared to the detailed approach. Firstly, the step size will not be reduced, leading eventually to a smaller number of steps. Moreover, Jacobian updates to account for the varying step size are avoided. When a jump takes place in an injector, the corresponding Jacobian must be updated. In the case of the integrated scheme, this means an update of the Jacobian of the whole system, since traditional partial update algorithms seems not well suited for the dynamic problem. In the proposed approach, conversely, thanks to the decomposition detailed in Chapter 4, this update could be performed only partially, with a great saving in computing time especially in large systems: the advantages of the combination of these two techniques will be shown in more detail in Chapter 7.

The *ex post* treatment of discrete events may yield wrong results, or even divergence, unless precautions are taken when modelling and solving. Attention must be paid to formulating the flow so that it is differentiable with respect to $\mathbf{x}$. Additionally, both the jump conditions and the flows must be formulated in a way that accommodates large deviations from the actual solution. Last but not least, a proper state transition graph has to be defined to avoid cycling between flows. The interested reader can find more exhaustive explanations and numerical examples in [FCPVC11].

6.4 Step size and step size reduction

So far, in this chapter, the use of "large" steps has been stressed without making any reference to numerical values. This is because the presented method is valid for any step size, with the reservation, of course, that accuracy degrades with the increase of the step size. It is thus possible to state that, in case of step size which are comparable to the smallest time constant of the system, the proposed approach will perform as accurately as a detailed simulation using steps of the same order of magnitude.

For instance, both BDF of order 2 and TM are second-order integration methods and for the step size used in detailed simulation, their solution accuracy is comparable. On the other hand, for increasing h, BDF cannot reproduce fast oscillations by the combined effects of hyper-stability and limited resolution (supposing, for instance, that at least 4 points per period are needed to describe a sinusoid, oscillations faster than 0.5 Hz are not reproduced with steps larger than 0.5 s). Time-averaging enables the simulation to concentrate over aperiodic changes and slow oscillations, while neglecting the fast ones.

Another reason for not taking too large step sizes has to do with the discrete dynamics also present in the model. As discussed above, to preserve computational efficiency, the exact jump instants are not identified: instead, they are accounted for at the next discrete time. Hence, too large a step size would delay the transitions and artificially synchronize multiple events.

Finally, larger step size generally means more Newton iterations per step. Hence, it may be required to momentarily reduce the step size in case of divergence, e.g. when the system undergoes a significant aperiodic change [FVC09].

The proposed approach consists in: (i) integrating with a maximum step $\bar{h}$ chosen in accordance with the requirements above, (ii) reducing the step size automatically when the Newton iterations exhibit convergence difficulties, and (iii) recovering to $\bar{h}$ as soon as possible.

Note that this step size control differs from the traditional local-truncation-error based control [Gea71]. Indeed, the latter criterion makes little sense in so far as accuracy requirements are relaxed and the emphasis is put on the coarse response of the system.

The logic to control the step size, when convergence difficulties are met, consists of bounding the computational effort estimated as the number of Newton iterations. The step size is first reduced by an arbitrary factor: $h_{new} = h/k$. If good convergence is regained, the simulation proceeds with another step; otherwise successive reductions are applied until reaching the minimum step size (in case of failure at the minimum step size divergence is declared and the simulation stops). In our tests $k = 2$ was found to work satisfactorily.

Subsequent steps are taken following a logic of estimation of the step computational effort, supposed to grow with the step size. Namely, the effort is estimated by the number n of Newton iterations performed in order to solve the last time step. The step size is chosen in order to match a desired number of iterations N. A "dead-band" M on the effort is considered in order to avoid too frequent step size variations.

More precisely, the step size is varied as follows, depending on the observed value of n:

$$h_{new} = \begin{cases} h\dfrac{N}{n+M} & \text{if} \quad n < N - M \\ h & \text{if} \quad N - M \leq n \leq N + M \\ h\dfrac{N}{n-M} & \text{if} \quad N + M < n \end{cases} \tag{6.5}$$

the purpose being to come back as quickly as possible to the default step chosen. When the default step size value is reached, the fixed step size strategy is resumed. The choice of parameters k, N and M may be system dependent and usually results from of a trial-and-error procedure. In the present implementation $N = 5$ and $M = 2$.

In an earlier publication [FVC09] the mismatch vector of the first Newton iteration was suggested as a measure of computational effort but experience has shown that it is impractical for two reasons: (i) in case of large systems it would require a significant computational overhead and (ii) the mismatch vector value cannot be interpreted easily in black box models, especially for the algebraic part of the model.

The hybrid behavior of power systems can give rise to another numerical difficulty, especially when overstepping an event with a large step. The problem most likely to remain unsolved, even after adjusting the flow equations and/or the jump conditions, is the cycling between flows: this phenomenon happens when a discrete state cannot be determined after a step and this step is repeatedly computed while the system keeps passing through two or more discrete states without settling in any.

In order the simulation to proceed, a first option consists in leaving the time step with the last available solution $\mathbf{x}^{(j)}$, once the number of flow jumps reaches a maximum, and proceeding with the next step. We found in many cases that the problem disappears at the next time step. This temporary discrepancy may be acceptable for the envisaged applications. However, there is no guarantee that the problem will disappear.

In so far as the problem stems from severely overstepped events, another option is to reduce the step size h until the problem is no longer met. h is first reduced by a factor k: $h_{new} = h/k$ (our simulations have shown that $k = 2$ is a convenient choice). If no event cycling takes place any longer, the step is accepted; otherwise further reductions are applied

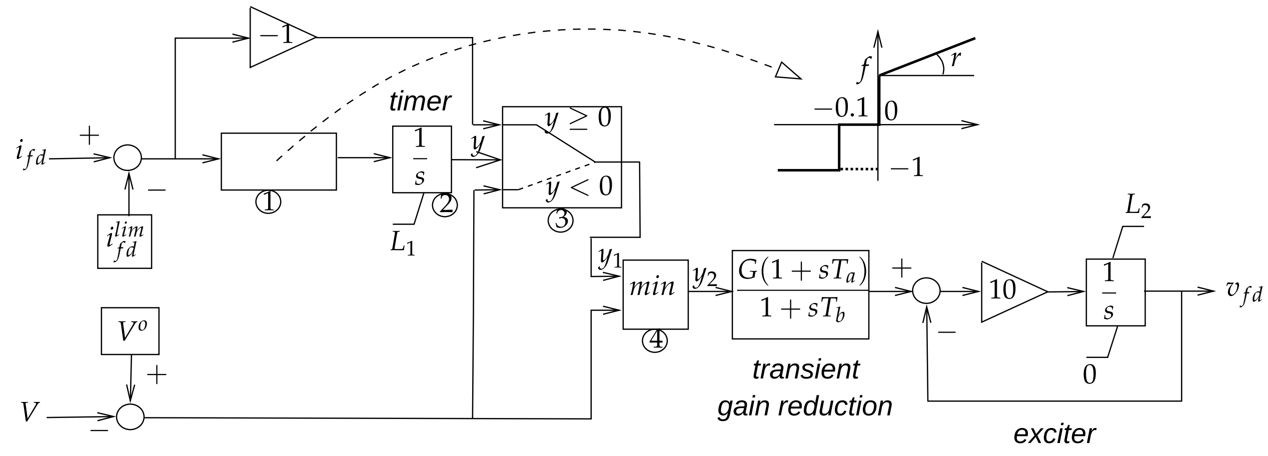


Figure 6.3: Block diagram of voltage regulator, field current limiter and exciter

until the problem is solved. Then, the original step size $\bar{h}$ is restored and the simulation proceeds. The application of this strategy to a power system example is detailed in the next section.

6.5 Example of cycling between flows

The example has been obtained with on the Nordic32 system Case N3 (see Section 4.6). At $t = 1$ s, the transmission line $4032 - 4044$ is tripped. This causes the neighbouring voltage-controlled generator g14 to operate above its permanent field current limit. The block-diagram of the voltage regulator, field current limiter and exciter is shown in Fig. 6.3.

In normal operating conditions the field current i_{fd} is much lower than its permanent limit i_{fd}^{lim} and the output of block 1 is -1. This keeps the integrator of block 2 at its lower limit L_1, which is negative. It results that the switch in block 3 remains in the lower position and passes the input $V^o - V$. The minimum gate receives two equal inputs, and we assume it selects y_1, the output of block 3.

When the machine operates with $i_{fd} > i_{fd}^{lim}$, the output y of the integrator block 2 grows, until it reaches a positive value, causing block 3 to switch. Thus, block 2 acts as a timer and provides inverse-time characteristic to the OEL. After block 3 has switched, the inputs of the minimum gate are $V^o - V > 0$ and $y_1 = i_{fd}^{lim} - i_{fd} < 0$, respectively. Hence, the gate keeps on selecting y_1.

The purpose of this minimum gate is to reset the machine under voltage control whenever operating conditions improve to the point that $V^o - V$ becomes negative. However, in

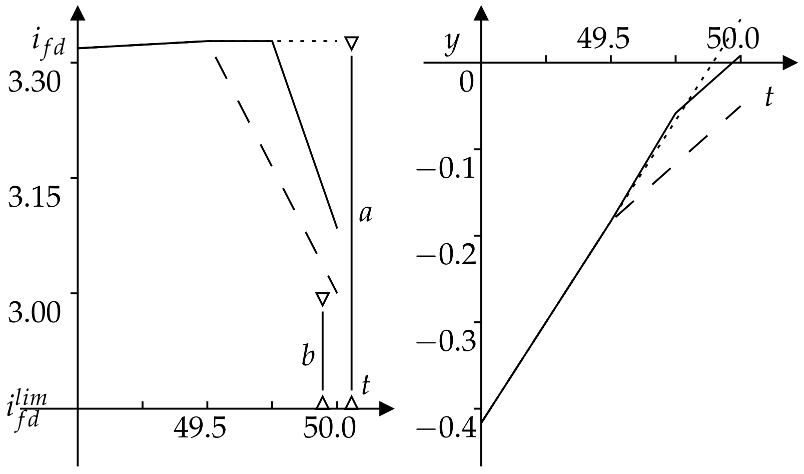


Figure 6.4: Detailed view of iterations when the OEL becomes active

the following example, no such reset takes place and the minimum gate selects its w input throughout the whole simulation.

After the line outage, i_{fd} starts increasing and becomes larger than i_{fd}^{lim} at $t = 1.5$ s. The step from $t = 1$ to $t = 1.5$ is thus repeated to account for the jump from flow $f = 0$ to flow $f = i_{fd} - i_{fd}^{lim}$ in block 1. The output of block 1 becomes positive, causing a jump in block 2, corresponding to y leaving its lower limit L_1.

Variable y keeps on increasing until it becomes positive, causing the switch in block 3 to change, i.e. the machine to switch from voltage to field current control in between $t = 49.5$ and $t = 50$. A zoom on the evolution of i_{fd} and y is given in Fig. 6.4.

Here is a detailed analysis of this step:

- at the beginning of the step, $y < 0$, block 3 is selecting its input $V^o - V$; the step converges to a positive y value at $t = 50$, triggering a jump in block 3. This is shown with dotted lines in Fig. 6.4;

- the step is restarted with block 3 passing $i_{fd}^{lim} - i_{fd}$. However, when passing through the exciter and the synchronous machine this signal has such a pronounced effect on the system that a value i_{fd} close to i_{fd}^{lim} is retrofitted to the OEL, which causes the updated value of y at $t = 50$ to get back to negative value. This is shown with dashed lines in Fig. 6.4. However, this triggers the reverse jump of block 3, and causes the jump handling procedure to endlessly oscillate between voltage and field control.

The variable y that controls block 3, when integrated with "large" derivative $i_{fd} - i_{fd}^{lim} = a$

(see Fig.6.4) triggers the jump to the flow with "small" derivative $i_{fd} - i_{fd}^{lim} = b$, where $b < a$ due to the above mentioned fast retroaction. Conversely, when integrated with the "small" derivative b, it triggers the jump to the flow with "large" derivative a.

As indicated in Section 6.4, two options are available to proceed with the simulation: (i) break the cycle and accept the last available solution, or (ii) reduce the step to have more chances to avoid cycling. The first option yields satisfactory convergence but delays the OEL action by one step. The result of second option is shown with solid line in Fig. 6.4. The step size has been halved after meeting the problem at $t = 49.5$. The simulation passes from $t = 49.50$ to $t = 49.75$ without jump. Then the machine correctly switches under field limit in between $t = 49.75$ and $t = 50.00$. At 49.75 s, the value y escapes the region of non-existence of solutions and can thus converge successfully.

6.6 Time-averaging results on Nordic32 system

This section shows applications of time-averaging on Cases N1, N2, N3 and N4, respectively (see Section 4.6). Case N5 is not considered here since it involves a short circuit that cannot be simulated with the large steps considered in this chapter. The outputs have been obtained with a time-averaged dishonest Newton scheme (denoted "T" in the plots). More details about this and the other solvers are given in Appendix A.5. Table 6.1 summarizes the DTW distances between the outputs of scheme T and I (the latter being the reference). It should be stressed right away that δ and σ are significantly higher than μ in most of the cases, confirming that time-averaging renders the average system evolution, thus keeping μ relatively small. Relevant results concerning the speed-up are detailed in the next sections, on the larger systems.

6.6.1 Case N1

Figure 6.5 shows a zoom of the active power flow in the other circuit of line 1013-1014. The long-term evolutions given by the T and the I schemes then become indistinguishable. As expected, increasing of the step size h smooths the oscillations, up to the limit case of $h = 0.5$ s that nearly reaches the underlying equilibrium in one step.

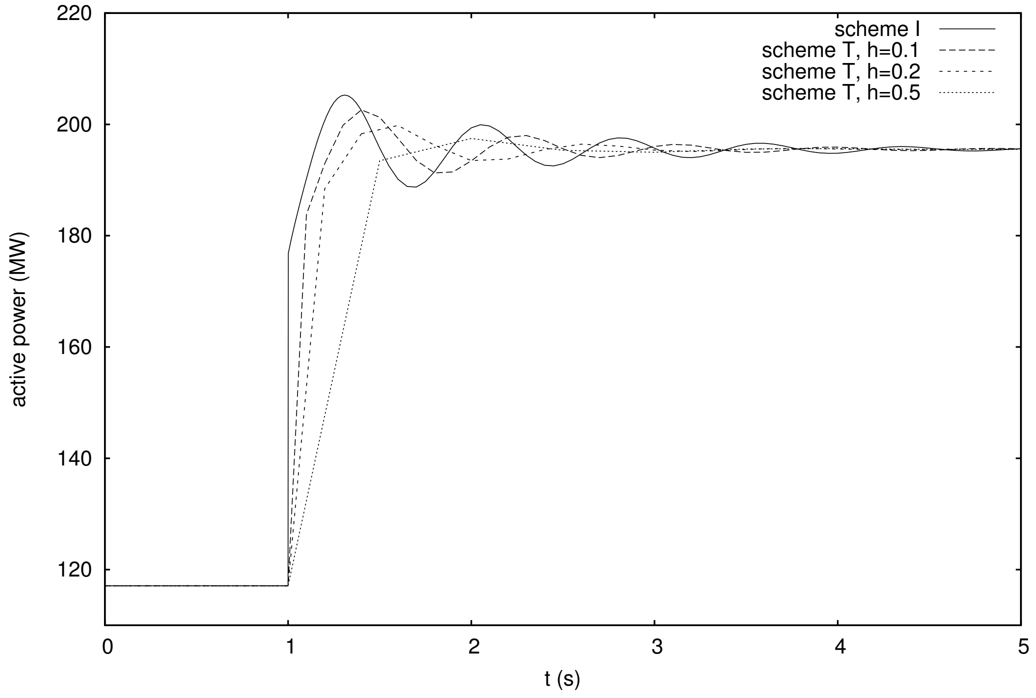


Figure 6.5: Case N1: active power flow in one circuit of line 1013-1014

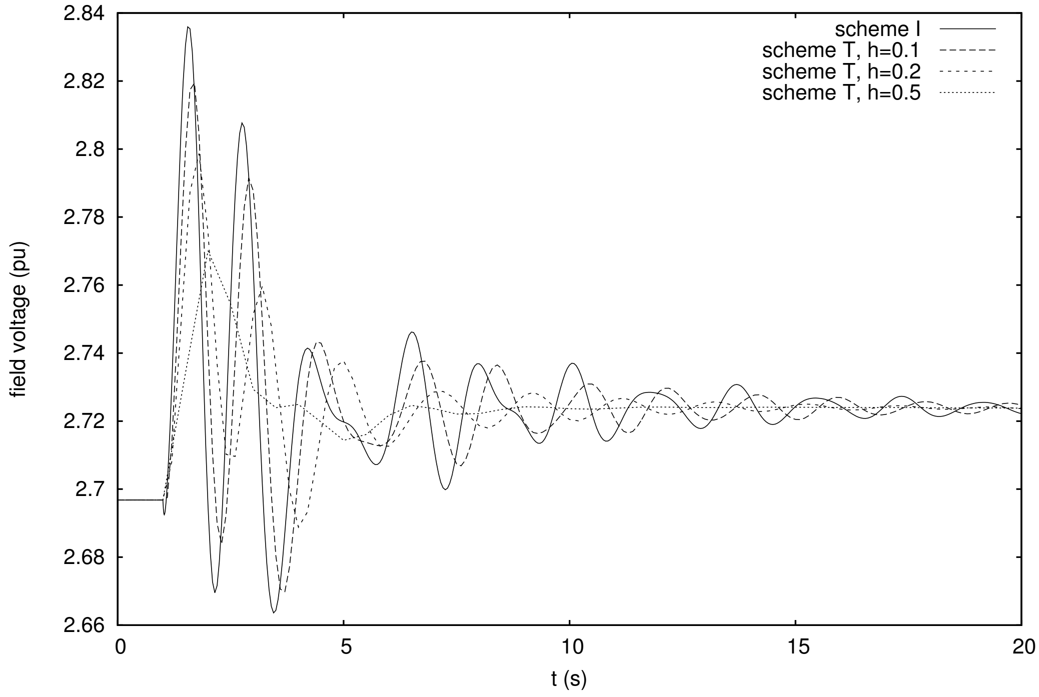


Figure 6.6: Case N2: field voltage of generator g15

6.6.2 Case N2

Case N2 is another scenario where the smoothing effect of time-averaging is highlighted. A zoom on the field voltage evolution of generator g15 is provided in Fig. 6.6. As expected, the time-averaging effect is increased with the step size. The long-term evolution is not very different from the one shown in Fig. 4.10.

6.6.3 Case N3

Case N3 allows to test time-averaging on a collapse scenario. The voltage at bus 1041, evolving towards collapse, is plotted in Fig. 6.7. While the overall aperiodic behavior is tracked, smaller oscillations are filtered out. The evolutions show some differences towards the end of the simulation, but it is important to observe that the transmission voltage is already quite low, and the fate of the system already inevitable!

Figure 6.8 depicts the speed deviation of generator g6. The loss of synchronism is detected some seconds in advance by the time-averaged simulations with respect to the benchmark; this does not mean, however, that time-averaging generally tends to anticipate events with respect to detailed simulation: delays have been also experienced.

The outputs are computed with the indicated steps up to the end of the simulation, since the Newton convergence is satisfactory and no step size reduction is required, which is quite noticeable!

6.6.4 Case N4

Case N4 underlines the weak point of stiff-decay methods, when the step size is not limited. Figures 6.9 and 6.10 show the field voltage and speed deviation of generator g17.

Let us recall that this simulation is purely an academic exercise, since the power system stabilizers of the system are disabled in order to obtain an oscillatory evolution. One may argue that this can happen when the system undergoes a progressive degradation such that the operative point becomes oscillatory unstable, for instance under the effect of OELs by-passing the stabilizers [VCV08], or due to cascade line trippings.

In the test of Figs. 6.9 and 6.10, however, the variety of the chosen step sizes allow us to appreciate the influence of hyper-stability. With steps of 0.1 s, the loss of synchronism is detected two rotor oscillations later. Integrating with steps of 0.2 s, returns a case of

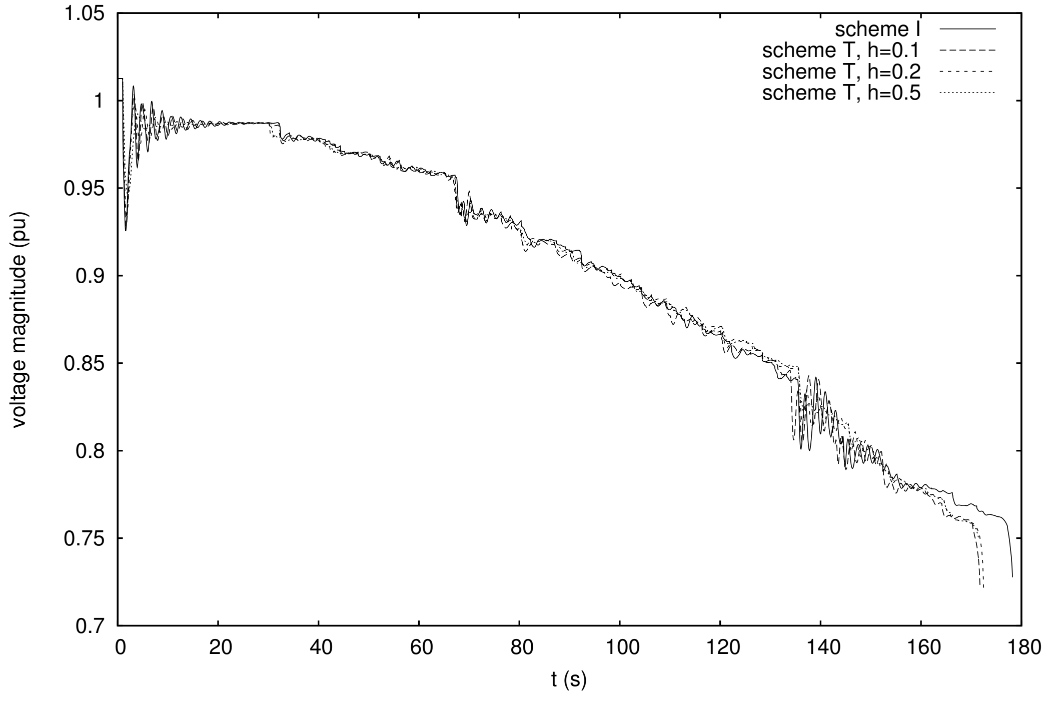


Figure 6.7: Case N3: voltage at transmission bus 1041

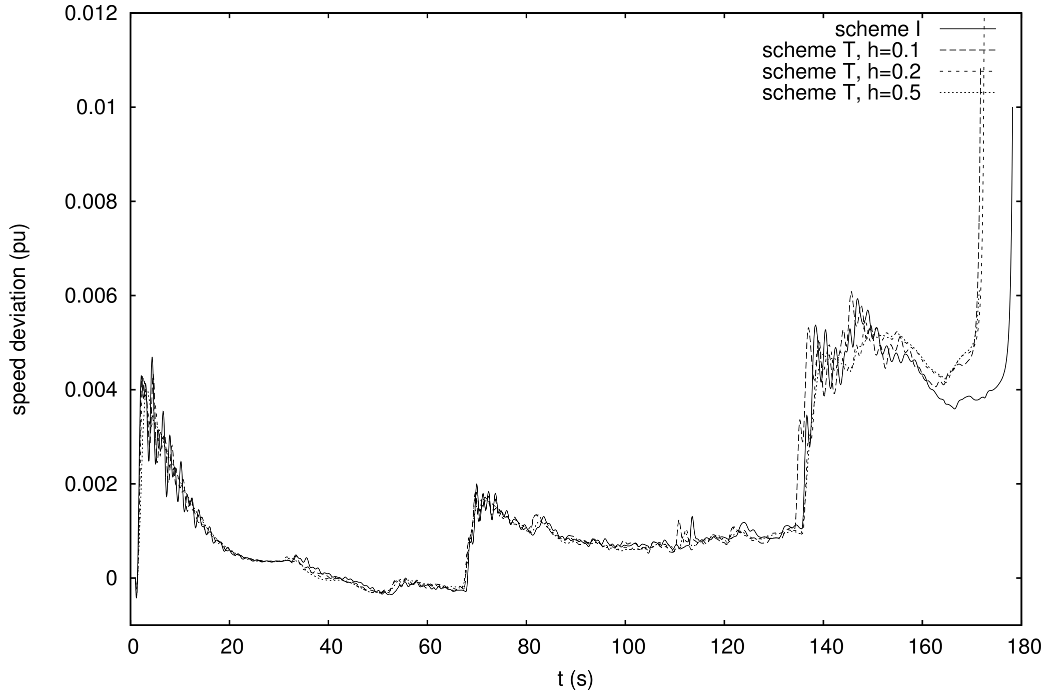


Figure 6.8: Case N3: synchronous machine speed of generator g6

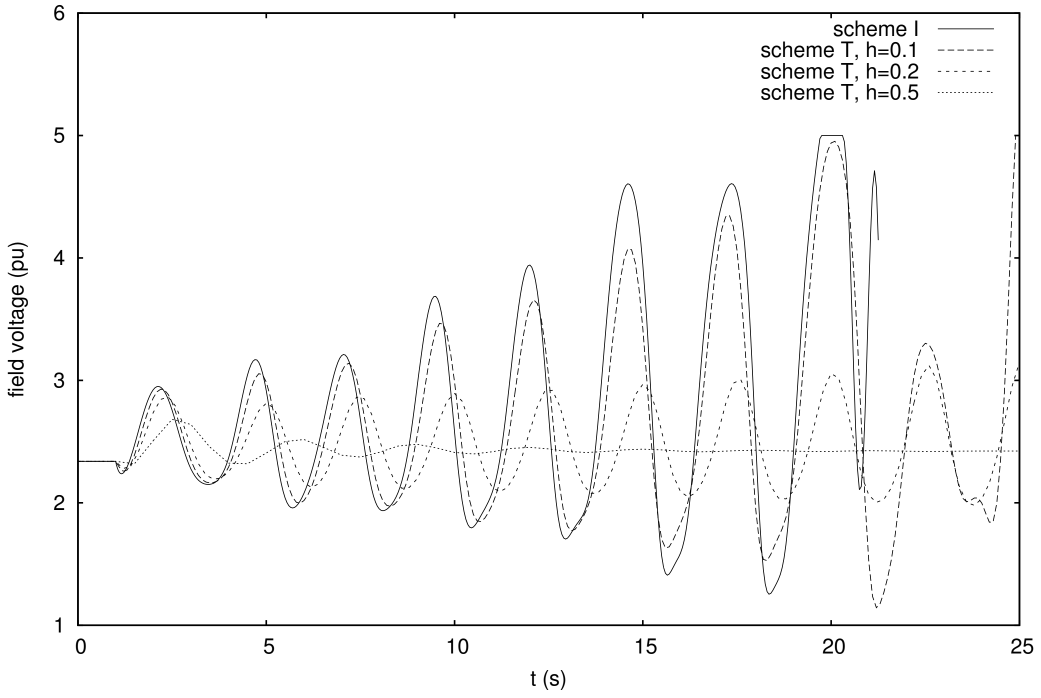


Figure 6.9: Case N4: field voltage of generator g17

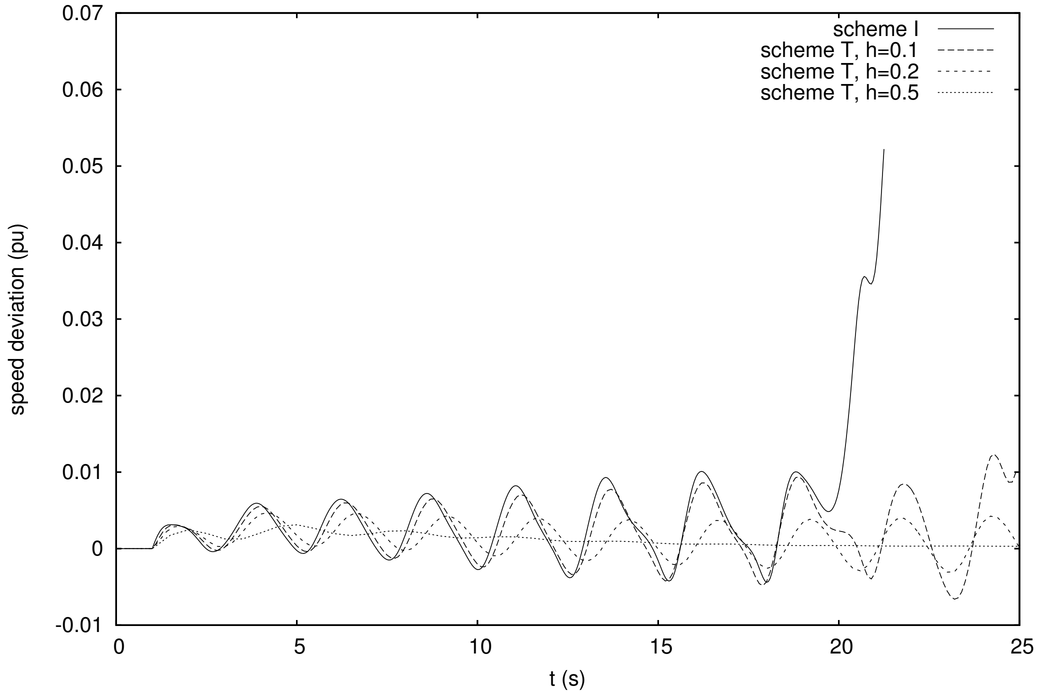


Figure 6.10: Case N4: synchronous machine speed of generator g17

marginal instability, which could still be flagged as dangerous [1]. A step of 0.5 s, conversely, completely overlooks the instability and returns a stable trajectory. In Chapter 7, we will show that simulating for a short period with "small steps" can mitigate the hyper-stability problem.

Table 6.1: DTW distances

Fig.	h	δ	μ	σ	Unit
6.5	0.1	1.5244	0.0578	1.5247	MW
	0.2	1.8562	-0.0856	1.8560	MW
	0.5	2.6842	-0.0467	2.6864	MW
4.8	0.1	0.0001	0.0000	0.0001	pu
	0.2	0.0003	-0.0000	0.0003	pu
	0.5	0.0006	-0.0000	0.0006	pu
4.9	0.1	0.0017	-0.0000	0.0017	pu
	0.2	0.0008	-0.0000	0.0008	pu
	0.5	0.0013	0.0000	0.0013	pu
6.6	0.1	0.0001	0.0000	0.0001	pu
	0.2	0.0049	-0.0001	0.0049	pu
	0.5	0.0087	-0.0003	0.0086	pu
4.11	0.1	1.6896	-0.0499	1.6893	MW
	0.2	2.9135	-0.0497	2.9139	MW
	0.5	5.5272	-0.0406	5.5287	MW
6.7	0.1	0.0015	0.0001	0.0015	pu
	0.2	0.0021	0.0002	0.0021	pu
	0.5	0.0032	0.0003	0.0032	pu
6.8	0.1	0.0001	0.0000	0.0001	pu
	0.2	0.0002	0.0000	0.0002	pu
	0.5	0.0002	0.0000	0.0002	pu
6.9	0.1	0.1297	0.0205	0.1282	pu
	0.2	0.4631	-0.0794	0.4567	pu
	0.5	0.6161	-0.1385	0.6010	pu
6.10	0.1	0.0007	0.0002	0.0007	pu
	0.2	0.0018	-0.0005	0.0017	pu
	0.5	0.0030	-0.0009	0.0029	pu

[1] Upon detection of such a situation, the simulation could be rerun with smaller steps or an eigenvalue analysis could be carried out to eliminate any ambiguity.

6.7 Time-averaging results on Hydro-Québec system

This section shows applications of time-averaging on Cases Q2, Q3 and Q4, respectively (see Section 4.7). Time-averaging cannot be applied to Cases Q1 and Q5 since they simulate a short circuit which is not compatible with the large steps used. The outputs have been obtained with a time-averaged dishonest Newton scheme (denoted "T" in the plots). More details about this and the other solvers are given in Appendix A.5. Since it is known that the system is prone to instability, time-averaging has been limited to steps of 0.2 s.

6.7.1 Case Q2

Figure 6.11 shows the speed deviation of the COI. A good match can be observed between the output of scheme T and the benchmark.

6.7.2 Case Q3

Case Q3 is interesting owing to the differences in MAIS activation. Figure 6.12 shows a zoom of the voltage evolution at bus 709 with the indication of the switching voltage threshold. The MAIS device switches off the reactor if the voltage stays below the threshold for 1 s consecutively. In the output computed with scheme I, the voltage of bus 709 stays below the threshold for slightly less than one second, thus no reactor is manoeuvered, while with scheme A it stays just slightly longer, so that it triggers this large impact control.

Figure 6.13 provides the speed deviations of the COI, which settle to different values due to one reactor unduly tripped. This case is slightly too severe to use time-averaging over the whole simulation: the method presented in Chapter 7 is to be preferred.

6.7.3 Case Q4

A similar behavior happens in Case Q4 where the switching delay of the first MAIS, in bus 715, triggers different sequences of reactor trippings, leading to some differences in the system trajectory.

The variable of interest for this case is the voltage at transmission bus 702, provided in Fig. 6.14. The overall match is acceptable for the reasons given in Section 4.7.4. Note that the computed trajectories are indistinguishable up to the first reactor tripping.

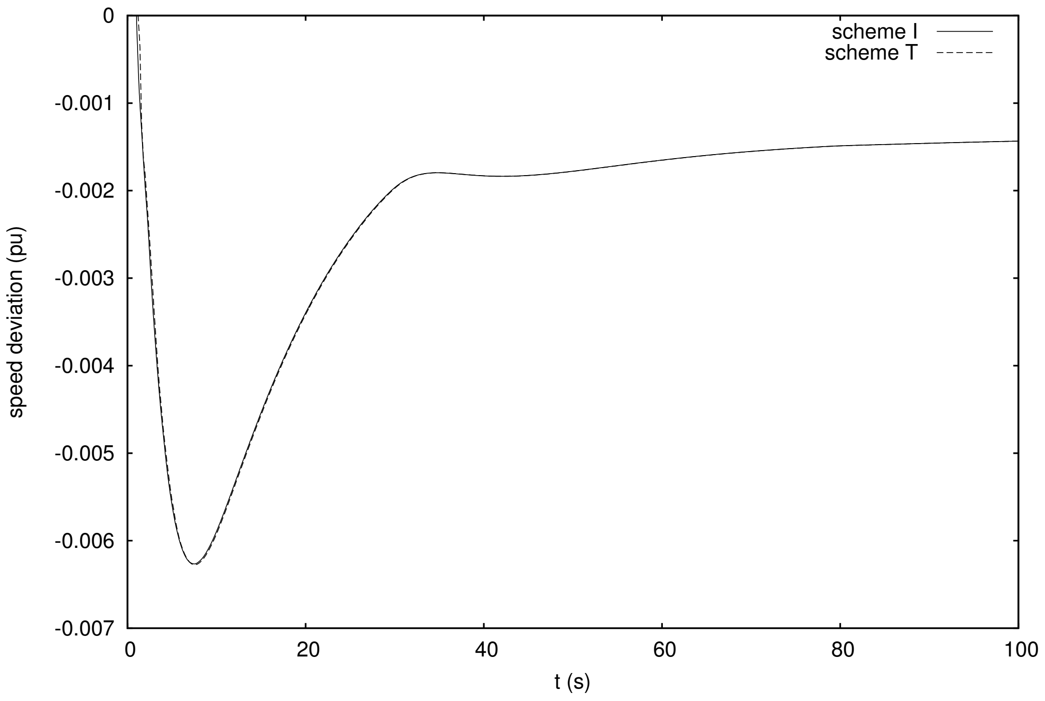


Figure 6.11: Case Q2: deviation of COI speed

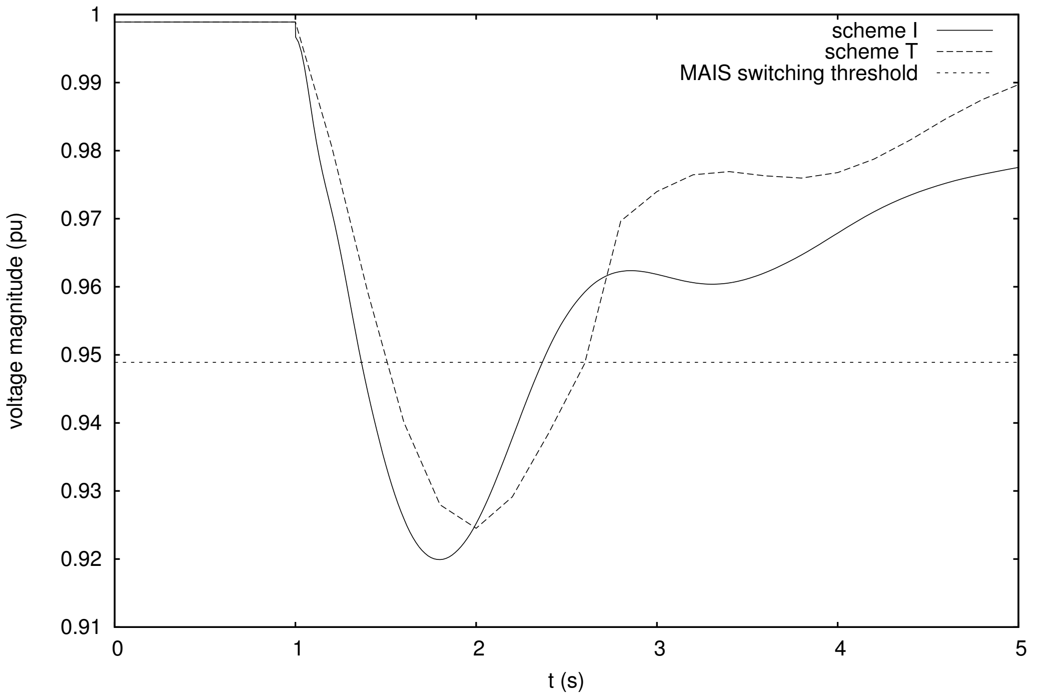


Figure 6.12: Case Q3: zoom of the voltage evolution at transmission bus 709

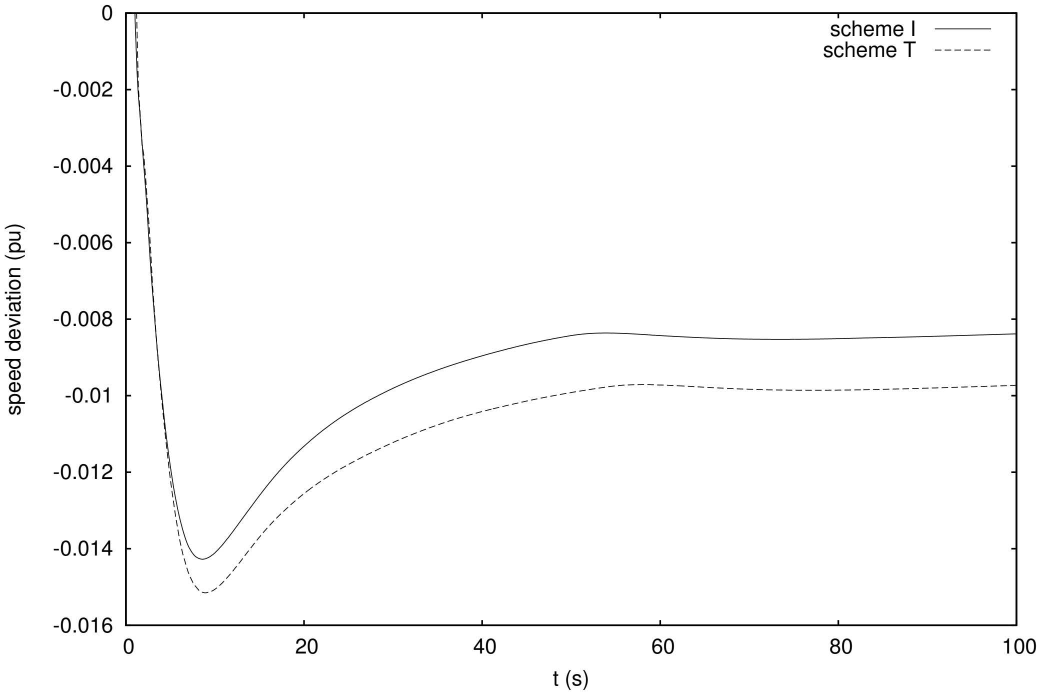


Figure 6.13: Case Q3: deviation of COI speed

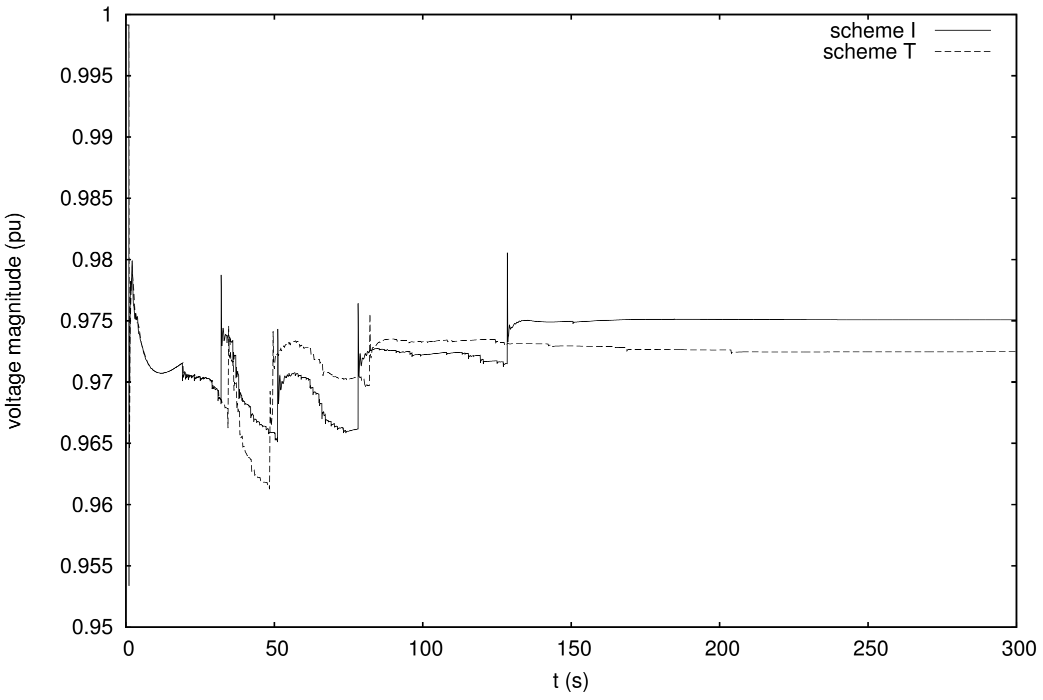


Figure 6.14: Case Q4: voltage at transmission bus 702

6.7.4 Computational effort

Table 6.2 summarizes the gain in computing time which is obtained with the application of time-averaging with respect to the benchmark. The gain upper bound can be estimated by the ratio of the average step sizes used in scheme T and in scheme I, which in this case is around 4. The actual gain is less because more iterations per step are performed, on average, due to the slightly more difficult convergence inherent to larger-step integration.

Table 6.2: CPU times (s)

Case	Scheme I	Scheme T
Q2	35.84	10.53
Q3	27.89	9.91
Q4	60.16	23.79

6.8 Time-averaging results on PEGASE system

This section shows applications of time-averaging on Cases P1 and P2 (see Section 4.7). Time-averaging cannot be applied to Case P3 since it simulates a short circuit which is not compatible with the large steps used. The outputs have been obtained with a time-averaged dishonest Newton scheme (denoted "T" in the plots). More details about this and the other solvers are given in Appendix A.5.

6.8.1 Cases P1 and P2

The voltage evolutions at bus F0322411 are shown in Figs. 6.15 and 6.16. As expected, the approximation increases with the step size used, but even the highest step brings acceptable results for many practical applications.

The comparison with an accurate solution obtained with steps of 0.05 s is fair given the high number of discrete events (slightly less than 2000) of the type (2.13) that occur during this 4-minute simulation, i.e. one event per 0.12 s on the average. As it was already commented in Section 3.4 and shown in Fig. 3.2, the maximum theoretical step size is not much higher than 0.05 s during most of the simulation, while the actual step size used by a variable step implicit algorithm could be much smaller. In this respect, the reported gains have to be interpreted as lower bounds of the achievable speed-ups.

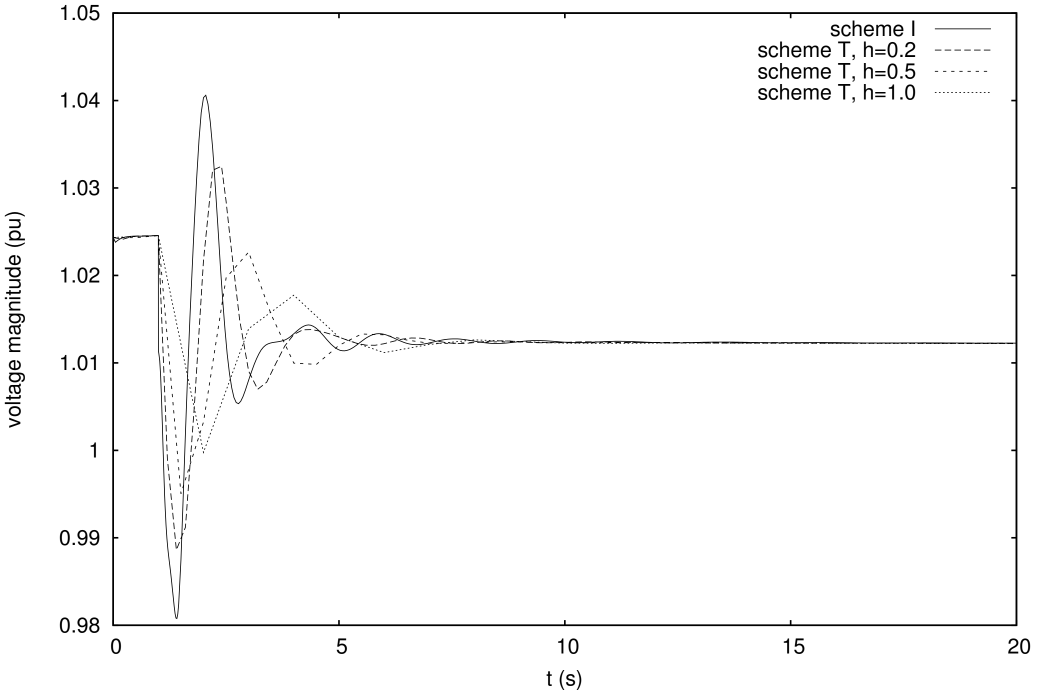


Figure 6.15: Case P1: voltage at transmission bus F0322411

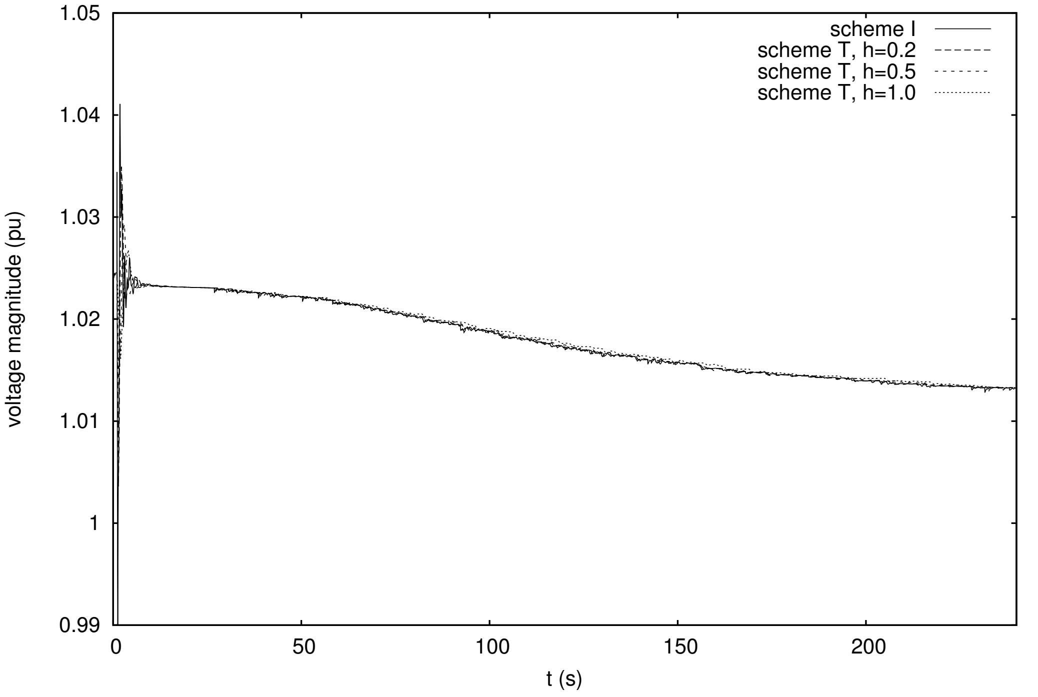


Figure 6.16: Case P2: voltage at transmission bus F0322411

6.8.2 Computational effort

Tables 6.3 and 6.4 detail the computing times obtained and the number of operations performed in case P1 using the benchmark scheme I and scheme T, respectively. Similar results for Case P2 are given in Tables 6.5 and 6.6. The overall computational effort is reduced with the increase of the step size only in the long-term Case P2; in Case P1, the effort is minimal for $h = 0.5$ s, indicating that the larger step size causes slower convergence, which in turn offsets the gain brought by the reduced number of steps. This is particularly evident from the larger effort required for Jacobian factorization in Case P1. A step larger than 0.5 s is thus not recommended for this system.

Table 6.3: Case P1: CPU times (s)

	Scheme I	Scheme T $h = 0.2$ s	Scheme T $h = 0.5$ s	Scheme T $h = 1.0$ s
total	90.29	32.17	21.13	25.67
factorizing $\mathbf{J}$	57.51	20.13	13.90	17.46
solving for $[\boldsymbol{\Delta}\mathbf{V}\ \ \boldsymbol{\Delta}\mathbf{x}]^T$	11.19	4.36	2.78	3.28
evaluating $[\mathbf{g}\ \ \mathbf{f}]^{\mathbf{T}}$	13.11	4.36	2.67	2.63

Table 6.4: Case P1: Number of operations

	Scheme I	Scheme T $h = 0.2$ s	Scheme T $h = 0.5$ s	Scheme T $h = 1.0$ s
factorizing $\mathbf{J}$	446	139	109	134
solving for $[\boldsymbol{\Delta}\mathbf{V}\ \ \boldsymbol{\Delta}\mathbf{x}]^T$	1393	485	355	404
evaluating $[\mathbf{g}\ \ \mathbf{f}]^{\mathbf{T}}$	1891	660	410	433

Table 6.5: Case P2: CPU times (s)

	Scheme I	Scheme T $h = 0.2$ s	Scheme T $h = 0.5$ s	Scheme T $h = 1.0$ s
total	470.00	198.67	105.89	79.54
factorizing $\mathbf{J}$	207.58	118.49	68.80	51.10
solving for $[\boldsymbol{\Delta}\mathbf{V}\ \ \boldsymbol{\Delta}\mathbf{x}]^T$	90.00	28.09	13.86	11.20
evaluating $[\mathbf{g}\ \ \mathbf{f}]^{\mathbf{T}}$	111.86	32.33	13.38	10.16

Table 6.6: Case P2: Number of operations

	Scheme I	Scheme T $h = 0.2$ s	Scheme T $h = 0.5$ s	Scheme T $h = 1.0$ s
factorizing $\mathbf{J}$	1597	921	511	399
solving for $[\mathbf{\Delta V} \;\; \mathbf{\Delta x}]^T$	11106	3451	1704	1394
evaluating $[\mathbf{g} \;\; \mathbf{f}]^\mathbf{T}$	15988	4673	2196	1644

6.9 Chapter conclusion

The most important contribution of this chapter is time-averaging applied to the dishonest Newton method algorithm. The proposed *ex post* treatment of discrete events is a key feature. The application of time-averaging brings a speed-up ranging between 4 and 6, with respect to scheme I, on the PEGASE test system. The speed-up on the other systems is consistent with the PEGASE results as well.

CHAPTER 7

Combining detailed simulation with accuracy relaxation techniques

Truth is much too complicated
to allow anything but approximations [1]

This chapter delivers our final proposal on how to achieve a good compromise between accuracy and efficiency of power system dynamic simulations. A general discussion on accuracy of both modelling and simulation opens this chapter. Several approaches to combine short-term detailed simulation with long-term simplified simulation are reviewed. An important aspect of this combination is the criterion to switch between short- and long-term simulation: a universal criterion is not sought here, as the author believes that a combination of numerical experience and system knowledge can better inspire the heuristics for such decision. The chosen heuristic criteria are described and relevant results for the three test systems close this chapter showing that acceptable outputs can be obtained by combining detailed simulation with the accuracy relaxation techniques presented in Chapters 5 and 6.

[1]John von Neumann, né Neumann János Lajos, excerpt taken from *The Mathematician*, appeared in *The Works of the Mind*.

7.1 How *accurate* is a *detailed* simulation?

Before going further in this chapter, this section reviews the concepts of detailed and simplified simulation that were used throughout this work: until Chapter 4, it was assumed that a "detailed simulation" is the solution of Equations (2.1-2.5) discretized with a small enough step size, with detailed handling of events and proper solution of discontinuities.

Now the question is: how *accurate* is a *detailed* simulation?

In fact, this question is crucial for the correct interpretation of simulation results. And the answer is: a detailed simulation is not as accurate as its name might suggest.

It is convenient to split the question in two parts: model accuracy and solver accuracy. Concerning the model, several reasons suggest that the absolute accuracy of a power system simulation is kind of a myth:

- equations (2.1-2.5) represent the system under a few approximations (single phase representation of an assumed balanced three-phase system, quasi-sinusoidal approximation, load aggregation or equivalencing, assumed load pattern, assumed intermittent generation output, equivalencing of external networks, etc.) which might indeed deliver a response which has some significant differences with respect to the real system trajectory;

- the models and the parameters behind equations (2.1-2.5) are somewhat uncertain or are anyway the result of model identification procedures which might not be extremely accurate for any operation mode, especially when the system conditions are degraded.

In view of the elements above, it is clear that the detailed solution might indeed be numerically accurate, but with respect to a model which is itself not fully accurate. The overall simulation, even if it is performed with as much numerical accuracy as possible, might give the illusion of accuracy, since it will not be fully accurate with respect to the real, somewhat unknown, underlying process.

Seen from this perspective, it would be possible to relax some of the solver accuracy constraints in favour of a faster simulation, especially when the obtained loss of accuracy is comparable with the accuracy limitations of the model. The techniques described in Chapter 5 and 6, if carefully applied, can indeed preserve the overall accuracy, using a small enough latency tolerance and a small enough time step size where needed, while providing a substantial speed-up.

7.2 Combination of short- and long-term simulations

When a power system is subject to a large disturbance (typically a fault), its dynamic response can be decomposed into a short-term and a long-term period (if the system does not lose stability in the short-term period, in which case it will not enter long-term, obviously). The first few seconds of simulation, which are dominated by electromechanical oscillations, belong to the short-term period, while thermal and voltage restoration processes are classified long-term and act in a time span ranging from around ten seconds to a few minutes. It should be kept in mind though, that this division is a rough simplification since there is no clear-cut boundary between short- and long-term. Also, in some cases, long-term events may trigger phenomena in the short term, whose time scale is in the order of seconds: this takes place for instance in Cases N3 and N5 (see Sections 4.6.3 and 4.6.5), where a field current limited generator eventually loses synchronism.

In the simplest case, the long-term evolution is determined more by the topology change resulting from clearing the fault than by the initial "impulse" given by the initiating short circuit.

This simple case is also dealt with by the post-contingency power flow computation (of the kind used for static security assessment), which aims at determining the final steady-state, assuming that the system survived the short-term period and is stable in the long-term. Post-contingency power flows suffer from two limitations: (i) the computation may diverge for purely numerical reasons, and (ii) if the system has no long-term equilibrium (this is the case in particular for long-term voltage instability), there is no solution, the computation diverges and leaves the user without information.

There is, however, an additional limitation of power flow computation: the presence of post-disturbance automatic controls (such as the MAIS devices in the Hydro-Québec system). Under the pressure of the electricity market, there is a trend to operate the system with less (expensive) preventive security margins and to rely to a greater extent on post-disturbance, corrective controls to stabilize the system and bring it within desired operating limits. The identification of those control actions requires simulating the trajectory followed by the system. This is why we advocate the use of dynamic simulation instead of power flow computations for security assessment purposes.

In "mild" cases however, where the system is assumed to be short-term stable and its long-term evolution is basically determined by the topological change resulting from the fault clearing, it is possible to use some of the techniques mentioned in Chapter 6 over the *whole* simulation period. Of course, in this case, the initiating fault is not simulated, only the topological change that results from its clearing is considered.

However, this simplified handling may be inappropriate in three cases:

- the system is prone to short-term (angle or voltage) instability, which requires to check its response to the initiating fault;
- the fault triggers some fast controls that help stabilizing the system. As already mentioned, the identification of those controls requires simulating the system evolution;
- that fault triggers some protections that disconnect some equipment (for instance tripping of induction motors stalled under the effect of a fault lasting too long, disconnection of power electronics due to abnormal voltages, etc.).

Note that in the last two cases, the system enters the long-term period non only with the topological change resulting from fault clearing but also with the changes from the fast controls and/or the protections. To deal with those cases, it is appropriate to combine detailed and simplified simulations, as follows:

- check the short-term period with detailed simulation;
- switch to simplified simulation for the long-term period.

In [KOO$^+$93], a scheme is proposed that allows to switch between three simulation modes:

- Transient stability simulation, using a fixed-time-step explicit integration formula;
- Extended-time simulation, using a variable-time-step implicit integration formula;
- QSS simulation.

The switching conditions are based on error estimation and frequency deviation. The QSS mode requires the development and maintenance of a separate model, and brings additional interfacing problems when switching, insofar as the system states do not coincide in all simulation modes, i.e. not all variables are present in all modes of operation.

The problem of interfacing a detailed simulation with a QSS simulation was further investigated in [LRLVC01], where the QSS mode is enabled only when the frequency oscillations have died out. Another approach [VCGL06] consists in running for a few seconds a detailed simulation that identifies the discrete events with great impact (such as for instance shunt compensation switching or load shedding), and then, from $t = 0$ on, performing a QSS simulation in which the so identified events are imposed as exogenous disturbances. A recent contribution [RBFEVC12] proposes a switching criterion based on singular perturbation theory.

The most important disadvantage of the above methods is the presence of additional models and the associated interfacing problem. Also, in some cases of long-term instability, under the effect of the degraded operating conditions, the short-term dynamics can become unstable: clearly, if the dynamics have been eliminated, such instabilities are overlooked or their symptoms are not easily interpreted. This is illustrated for instance by Cases N3 and N5.

A different approach with a single model was suggested by [ZA05], which relies on a continuation parameter to employ larger step sizes for slow dynamics and smaller ones for fast dynamics.

7.3 Combining detailed and relaxed accuracy simulations

The combination of short- and long-term simulations is not a novel idea in power system simulation. This section discusses ways of combining detailed simulation with accuracy relaxation techniques, thus using a single model only: this is the main advantage of the proposed approach with respect to some of the previous works reviewed in Section 7.2. The illustrative results of this chapter show that the obtained evolutions are reasonably similar to the benchmark.

Relaxing the accuracy in time is obtained by increasing the step size. As detailed in Chapter 3, the default step size which has been chosen for detailed simulation of a long term scenario is 0.05 s, while the default step size chosen for short-term scenari, including short circuits, is 0.01 s. The above times do not take into account temporary step size reductions, such as the solution of discontinuities (performed with steps of 0.001 s, as described in

Section 3.5), and the solution of non-converged or flow cycling case, as detailed in Section 6.4. In Chapter 6, conversely, time-averaging was obtained by imposing steps of 0.2, 0.5 or 1 s for the solution of long-term cases. The combined simulation will be performed on both short- and long-term cases with a *gradual* increase of the step size from the value used in detailed simulation to the value used in relaxed-accuracy simulation.

In the author's opinion, the variation of the step size is highly problem- and system-dependent: an experienced engineer could probably give indications on the step size increase in a better way than a universal criterion. A safeguard is that, in case of difficult convergence, the step size reduction and restoration is automatically taken care of by the procedure described in Section 6.4.

The accuracy in space will be relaxed by varying the latency tolerance. The conservative approach that has been chosen consists in disabling localization, i.e. setting $\epsilon_L = 0$, during the short-term evolution and increasing the latency tolerance ϵ_L in the long-term simulation. The switching from short- to long-term simulation will be dictated by the same rules as in the case of time-averaging. In case of difficult convergence of the Newton method, which may indicate insurgence of short-term dynamics in degraded system conditions, accuracy relaxation techniques are temporarily disabled: the step size is reduced, thus weakening the time-averaging effects, and ϵ_L is set to zero, preventing any component to be latent.

The above criteria represent an early effort in the combination of detailed and relaxed-accuracy simulation. Faster performances could be obtained with the use of other criteria, maybe inspired from the ones mentioned in Section 7.2, or with fine-tuning of the ones presented above. The elaboration of better or universal criteria was outside the scope of this work, whose purpose is to show that valid outputs can be obtained by gradual relaxation of accuracy requirements, as illustrated in the next sections.

The advantage of the proposed method lies in the seamless integration between detailed short-term simulation and relaxed-accuracy long-term simulation. In fact, since switching between simulation modes does not require any restructuring of the model, no interfacing problem between the two simulations is experienced. The variations of step size and latency tolerance concern the solver only, and allow performing a flexible, fast and reliable simulation in most practical applications.

However, the proposed method is not free from the drawback of possibly overlooking some instabilities, for instance taking place under the effect of long-term degraded operating conditions. The risk is anyway smaller than with model reduction methods. It is mainly linked to growing electromechanical oscillations, due to the hyper-stability of the numerical integration formula used. Aperiodic instabilities, such as those encountered in Cases N3 and N5 (see Sections 4.6.3 and 4.6.5) are correctly tracked.

7.4 Combined simulation results on Nordic32 system

This section shows the results obtained with the combined detailed-simplified simulation of the Nordic32 test system, in Cases N1, N2, N3 and N5 described in Section 2.4. The outputs have been obtained with a dishonest Newton scheme, combining detailed simulation with localization and time-averaging techniques (denoted "C" in the plots and referred to as "combined" in the tables). More details about this and the other solvers are given in Appendix A.5. In this scheme, after a user-imposed disturbance at $t = t_d$, the step will be increased gradually up to its maximum value according to Table 7.1, following the good practice of avoiding the step size to increase too steeply. In fact, thanks to the use of the modified BDF coefficients, that account for step size variations (see Appendix A.4), it is possible to obtain an "exact" interpolated integration for any variation of step size: however, in very extreme cases, it has been observed that reasonably small variations of the proposed kind are to be preferred.

Table 7.1: Time step size and integration formulae used

time interval	detailed simulation	combined simulation
$[t_d\ ;\ t_d + 0.001]$ s	0.001 s (BEM, $\epsilon_L = 0$)	0.001 s (BEM, $\epsilon_L = 0$)
$[t_d + 0.001\ ;\ t_d + 0.010]$ s	0.009 s (BEM, $\epsilon_L = 0$)	0.009 s (BEM, $\epsilon_L = 0$)
$[t_d + 0.010\ ;\ t_d + 0.170]$ s	0.010 s (BDF2, $\epsilon_L = 0$)	0.010 s (BDF2, $\epsilon_L = 0$)
$[t_d + 0.170\ ;\ t_d + 0.450]$ s	0.020 s (BDF2, $\epsilon_L = 0$)	0.020 s (BDF2, $\epsilon_L = 0$)
$[t_d + 0.450\ ;\ t_d + 1.200]$ s	0.050 s (BDF2, $\epsilon_L = 0$)	0.050 s (BDF2, $\epsilon_L = 0$)
$[t_d + 1.200\ ;\ t_d + 2.400]$ s	same as above	0.100 s (BDF2, $\epsilon_L = 0$)
$[t_d + 2.400\ ;\ t_d + 5.000]$ s	same as above	0.200 s (BDF2, $\epsilon_L = 0$)
$[t_d + 5.000\ ;\ +\infty]$ s	same as above	0.500 s (BDF2, $\epsilon_L = 0.01$)

DTW distances between the outputs of scheme C and I (the latter being the reference) are given in Table 7.2: μ is almost negligible, confirming the averaging effect, while δ and σ are significantly smaller than in the case of time-averaging only, revealing a better overall accuracy. The final speed-up brought by combined simulation is discussed in the next sections, on the larger Hydro-Québec and PEGASE systems.

Table 7.2: DTW distances

Fig.	δ	μ	σ	Unit
7.1	0.0711	0.0003	0.0712	MW
7.2	0.0003	-0.0000	0.0003	pu
7.3	0.0005	-0.0000	0.0005	pu
7.4	0.0024	0.0000	0.0024	pu
7.5	0.9988	-0.0193	0.9989	MW
7.6	0.0022	-0.0000	0.0022	pu
7.7	0.0001	0.0000	0.0001	pu
7.8	0.0034	-0.0003	0.0034	pu

7.4.1 Case N1

Fig. 7.1 shows the active power flow in the other circuit of line 1013-1014. The averaging effect is almost imperceptible, affecting only the last oscillations.

The effect of localization is clearly seen in Fig. 7.2, depicting the field voltage of generator g15. Its negligible oscillation is not computed after a few seconds, when the machine becomes latent.

7.4.2 Case N2

The voltage at distribution bus fed by 4043, involved in both short- and long- term dynamics is given in Fig. 7.3. While the first oscillations are slightly over-damped, the main difference between the output C and the benchmark is the misplacement of both the LTC moves by a few seconds. This is deemed fully acceptable for practical applications.

Figure 7.4 shows the field voltage of generator g15. The trajectory computed by combined simulation is sufficiently close to the benchmark, bearing in mind that the largest difference is given by the afore-mentioned LTC move displacements.

Figure 7.5 provides the active power produced by generator g15. This quantity, which is not directly related to the bus voltage, is not influenced by the time-shift of the LTC moves. The output could indeed be considered being as accurate as the benchmark.

7.4.3 Case N3

Case N3 offers an example of the advantage of combined simulation, which is very significant in cases of aperiodic instability long after the initiating disturbance. In fact, as Fig. 7.6 shows, both short- and long-term evolutions of the voltage at bus 1041, up to the final

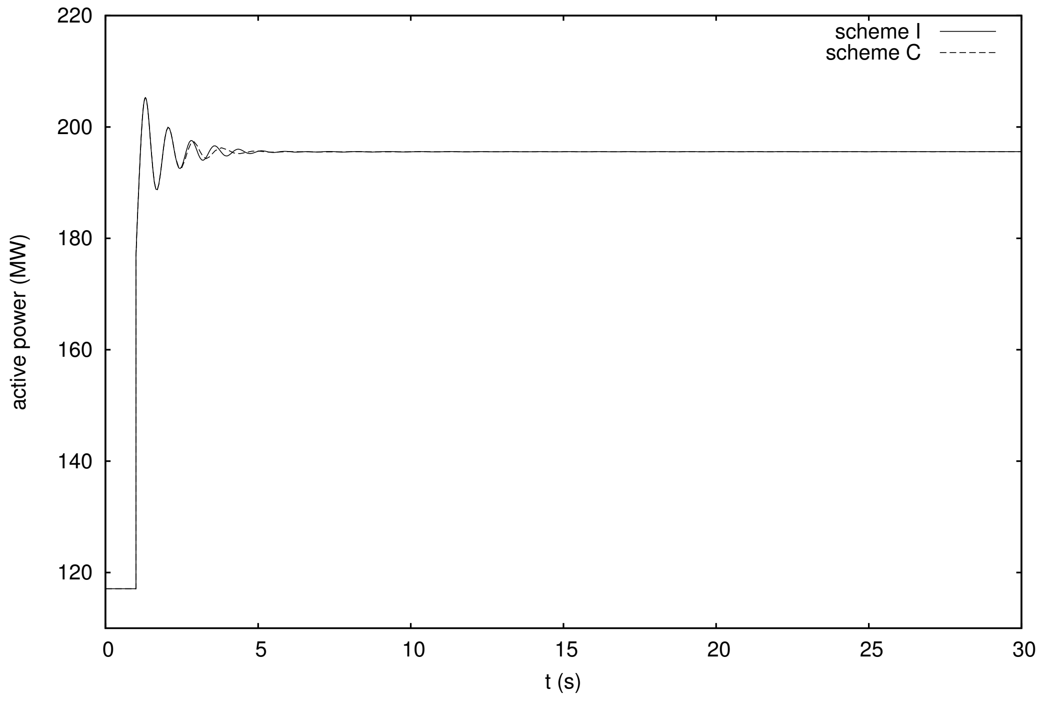


Figure 7.1: Case N1: active power flow in one circuit of line 1013-1014

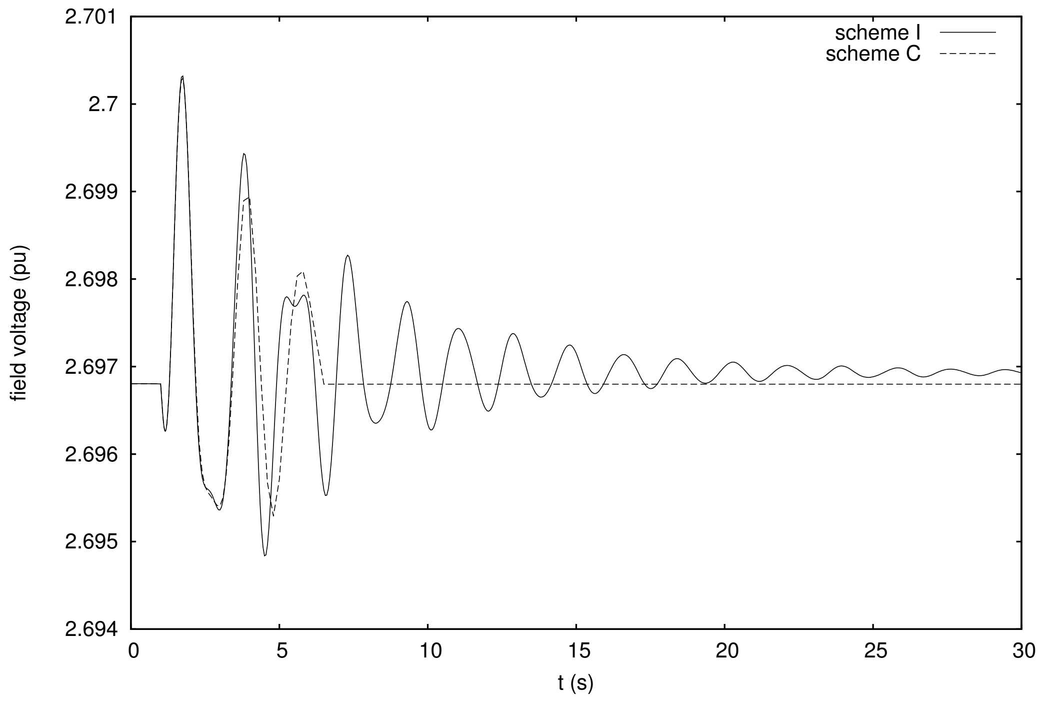


Figure 7.2: Case N1: field voltage of generator g15

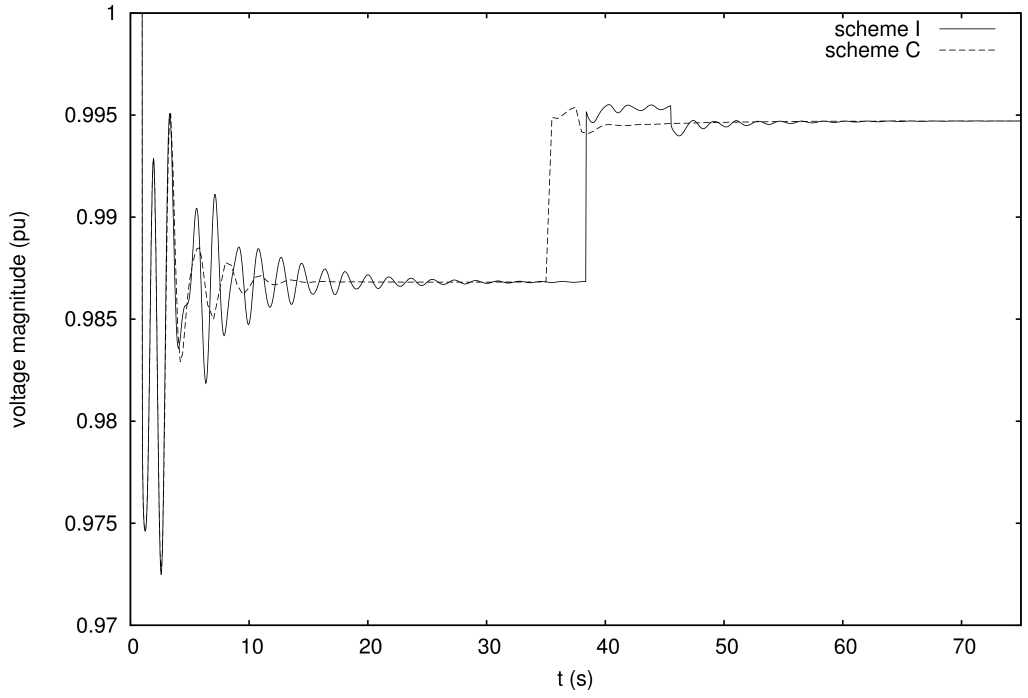


Figure 7.3: Case N2: voltage at distribution bus fed by bus 4043

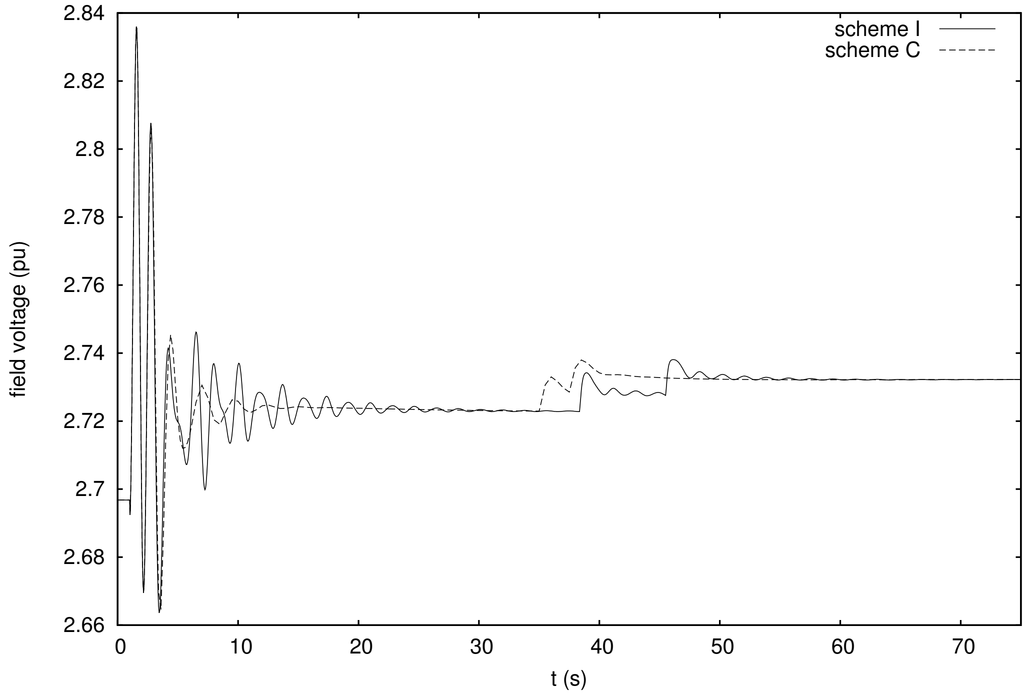


Figure 7.4: Case N2: field voltage of generator g15

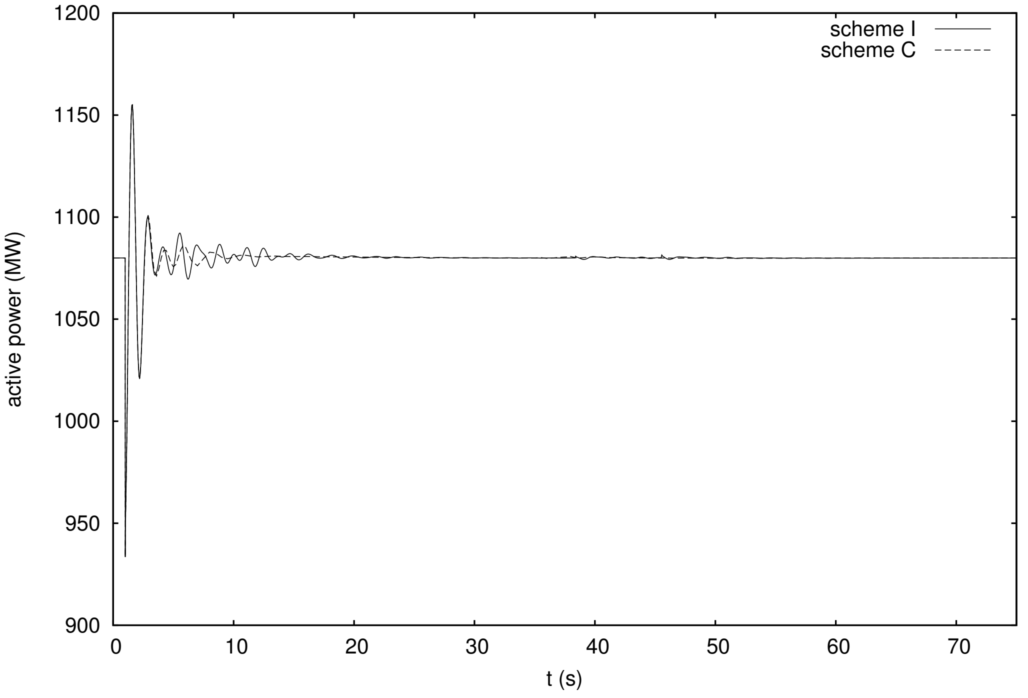


Figure 7.5: Case N2: active power produced by generator g15

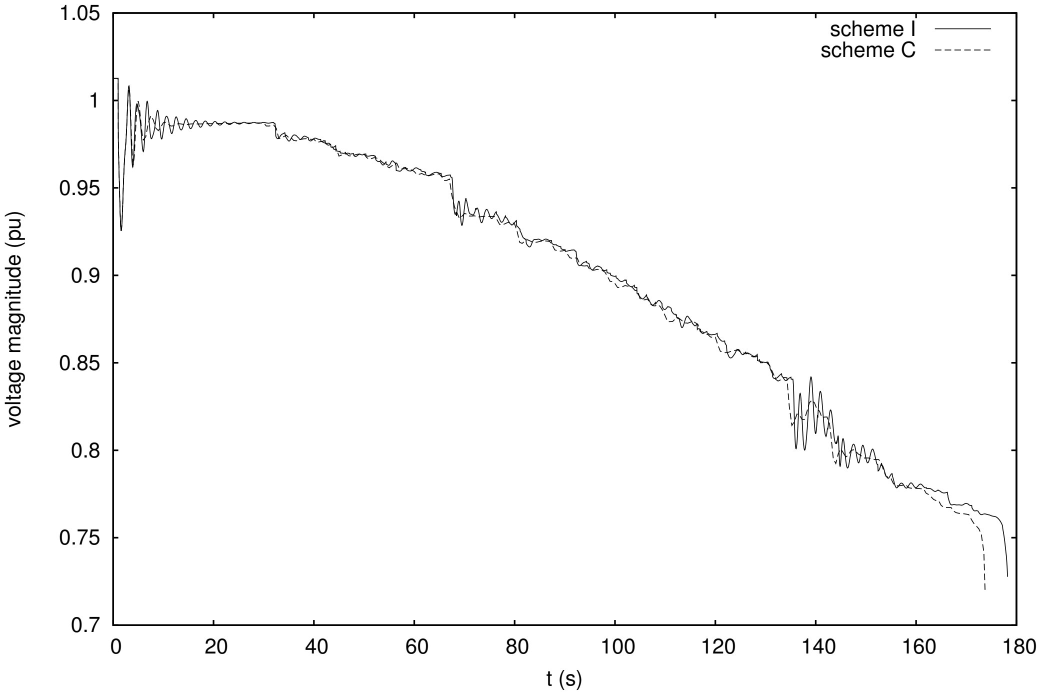


Figure 7.6: Case N3: voltage at transmission bus 1041

voltage collapse, are correctly computed, since the final discrepancy happens only when the system is already in very stressed conditions.

Figure 7.7 provides the evolution of the speed deviation of generator g6, whose loss of synchronism towards the end of the simulation is computed in slight advance by the combined simulation with respect to the benchmark. As for Fig. 7.6, this is deemed of no practical importance.

7.4.4 Case N5

Figure 7.8 provides the voltage at bus 1041, which evolves towards collapse. This case involves a short circuit, that could not be simulated with time-averaging only because significantly smaller steps are needed for the short-term period.

Our approach of combined simulation, however, can deal with the short-term evolution in the same way that a detailed simulation would do, and then increase gradually the step size. Besides a slightly accentuated damping of some oscillations, the long-term output of scheme C matches the average behavior of the benchmark.

We can thus conclude that combined simulation is here able to deliver a reliable response.

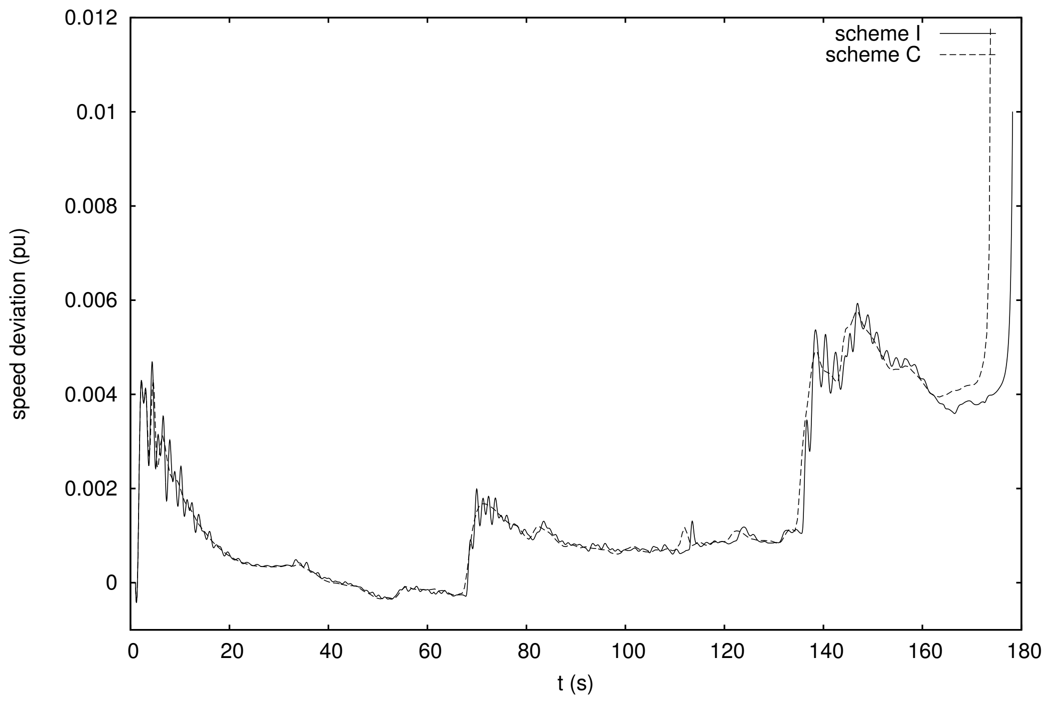


Figure 7.7: Case N3: synchronous machine speed of generator g6

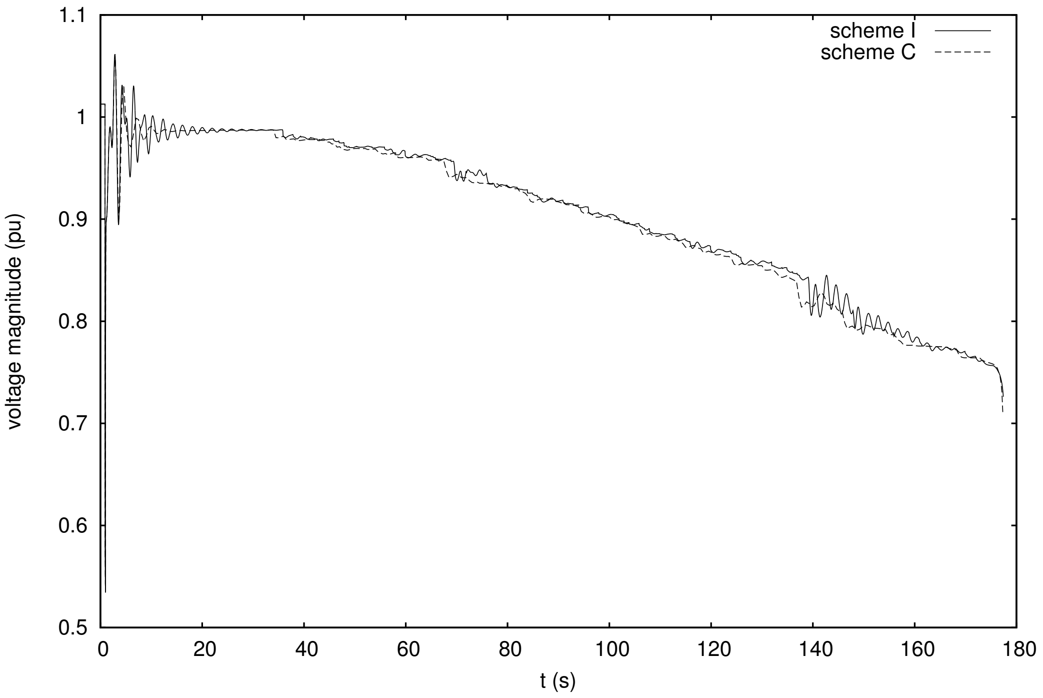


Figure 7.8: Case N5: voltage at transmission bus 1041

7.5 Combined simulation results on Hydro-Québec system

This section shows the results obtained with the combined simulation of the Hydro-Québec system, on all the scenari described in Section 2.5. The outputs have been obtained with a dishonest Newton scheme, combining detailed simulation with localization and time-averaging techniques (denoted "C" in the plots). More details about this and the other solvers are given in Appendix A.5. In this scheme, after a user-imposed disturbance at $t = t_d$, the step will be increased gradually up to its maximum value according to Table 7.3.

Table 7.3: Time step size and integration formulae used

time interval	detailed simulation	combined simulation
$[t_d \; ; \; t_d + 0.001]$ s	0.001 s (BEM, $\epsilon_L = 0$)	0.001 s (BEM, $\epsilon_L = 0$)
$[t_d + 0.001 \; ; \; t_d + \frac{1}{60}]$ s	0.016 s (BEM, $\epsilon_L = 0$)	0.016 s (BEM, $\epsilon_L = 0$)
$[t_d + \frac{1}{60} \; ; \; t_d + 3.000]$ s	$\frac{1}{60}$ s (BDF2, $\epsilon_L = 0$)	$\frac{1}{60}$ s (BDF2, $\epsilon_L = 0$)
$[t_d + 3.000 \; ; \; t_d + 9.000]$ s	$\frac{3}{60}$ s (BDF2, $\epsilon_L = 0$)	$\frac{3}{60}$ s (BDF2, $\epsilon_L = 0$)
$[t_d + 9.000 \; ; \; +\infty]$ s	same as above	$\frac{12}{60}$ s (BDF2, $\epsilon_L = 0.004$)

7.5.1 Case Q1

The performance of scheme C in Case Q1 is identical to the performance of scheme A (see Section 4.7.1), since the simulation spans the short-term period only.

7.5.2 Case Q2

Case Q2 allows scheme C to obtain speed-up seemingly without loss of accuracy. The outputs of both Figs. 7.9 and 7.10 are indistinguishable from the benchmark.

7.5.3 Case Q3

Figure 7.11 provides the speed deviation of the COI, showing that the spinning reserve is exhausted and frequency cannot be recovered close to its nominal value. Also in this case, the outputs of scheme C fully preserve the accuracy.

7.5.4 Case Q4

Case Q4 is particularly interesting owing to the marginal activation of one MAIS device, similar to the ones mentioned in previous chapters. Figure 7.12 shows a zoom of the voltage evolution at bus 755 with the indication of the switching threshold. The MAIS device switches off the reactor if the voltage stays below the threshold for 13 s consecutively. The output computed with scheme I stays below the threshold for slightly less than 9 s, thus triggering no MAIS tripping, while the output computed with scheme C, conversely, stays below the threshold for just a slightly longer time than 13 s, the tripping delay. This results in 4 MAIS activations in scheme I and 5 in scheme C. As already mentioned, a mistake in the identification of one MAIS device is not considered critical in the context of DSA.

The variable of interest for this case is the voltage at transmission bus 702, provided in Fig. 7.13.

7.5.5 Case Q5

Figure 7.14 shows the zoomed evolution of voltage at faulted bus 702. The action of the third MAIS device, at bus 730, is anticipated by scheme C by some ten seconds, which makes the second and third MAIS devices act nearly simultaneously. This produces a 5 s anticipation of the fourth (and last) MAIS move, but ultimately no difference in the number of MAIS

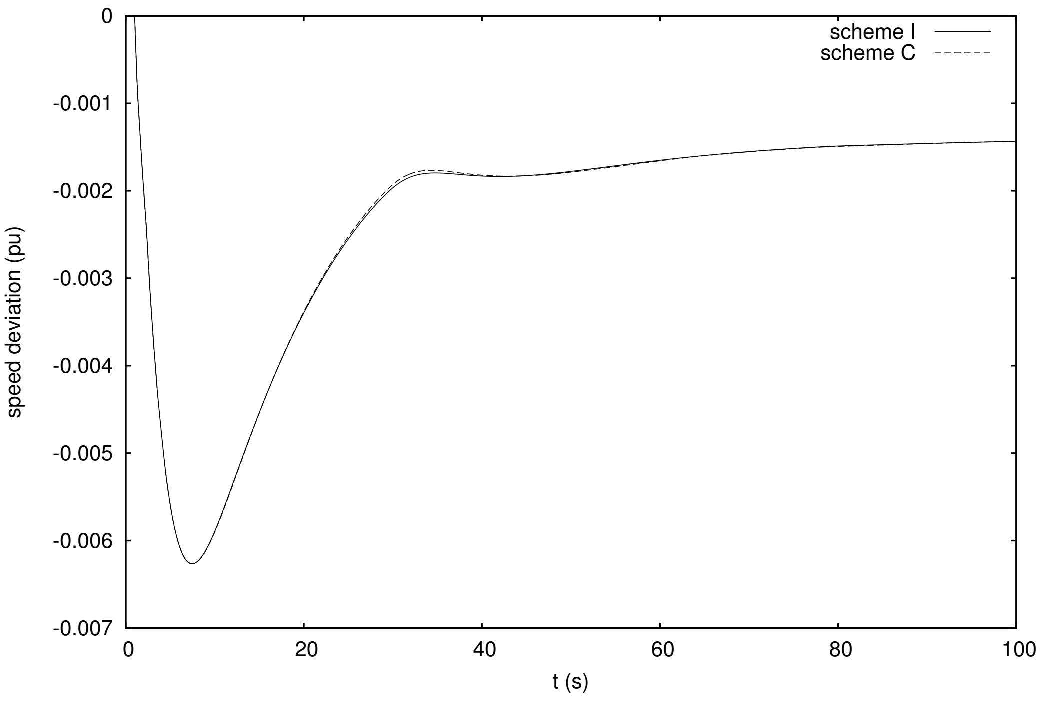


Figure 7.9: Case Q2: deviation of COI speed

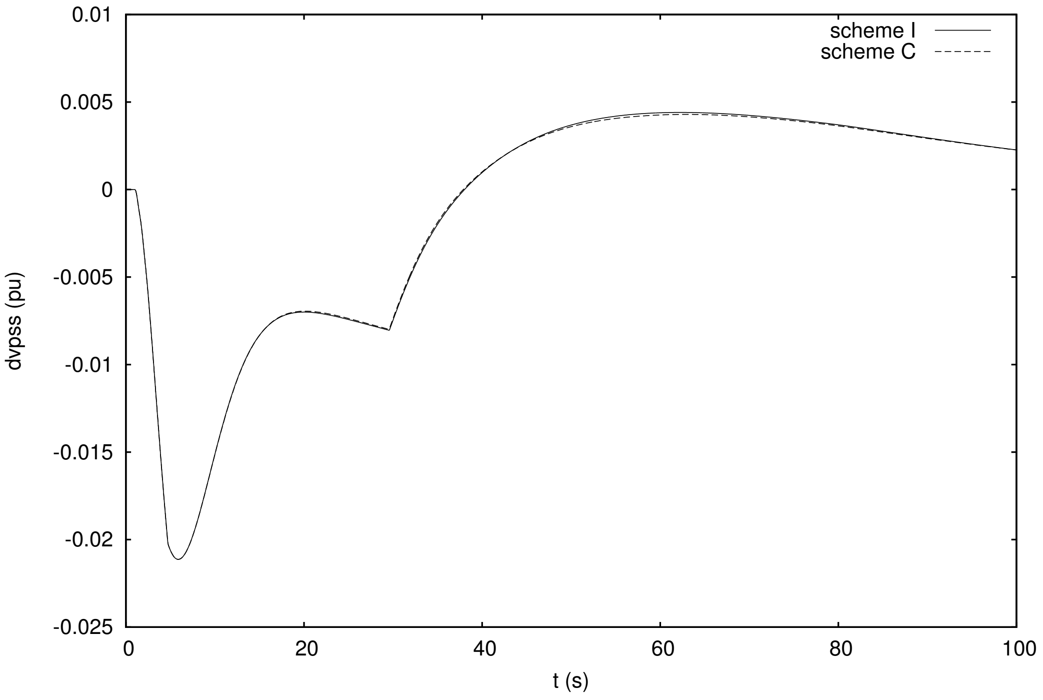


Figure 7.10: Case Q2: PSS signal of generator 294

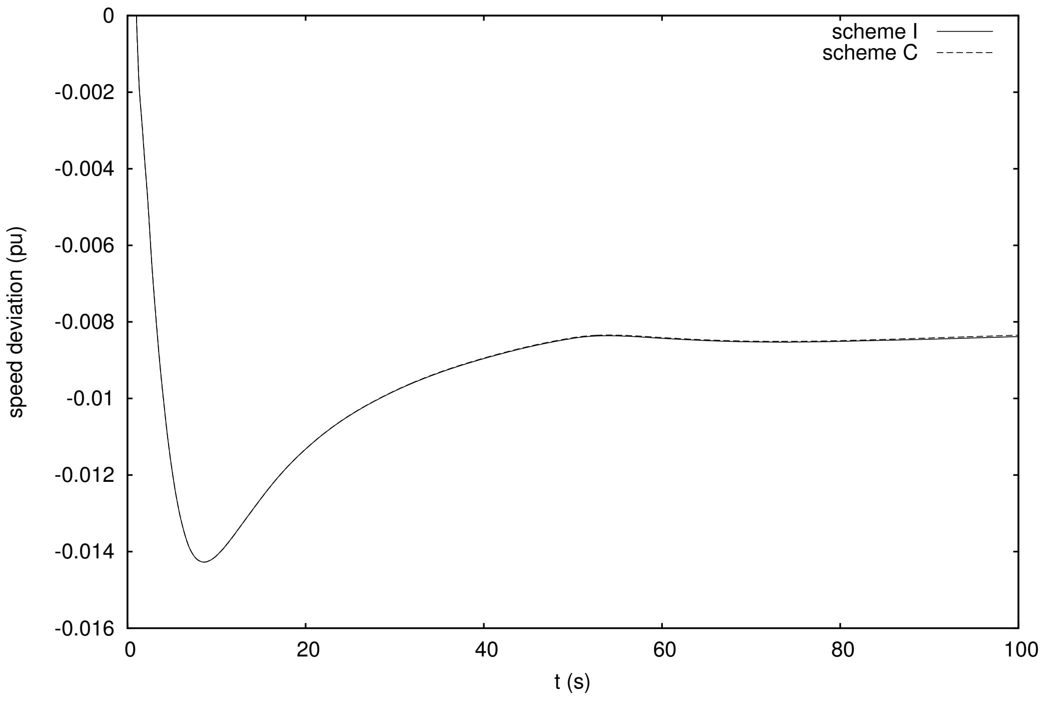


Figure 7.11: Case Q3: deviation of COI speed

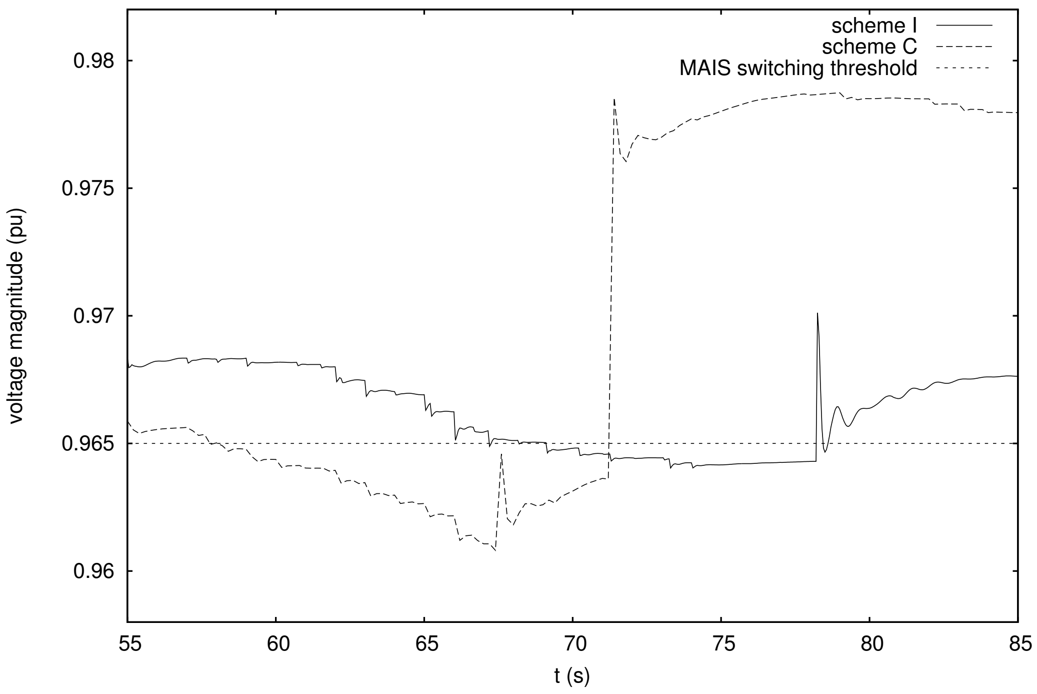


Figure 7.12: Case Q4: zoom of the voltage evolution at transmission bus 755

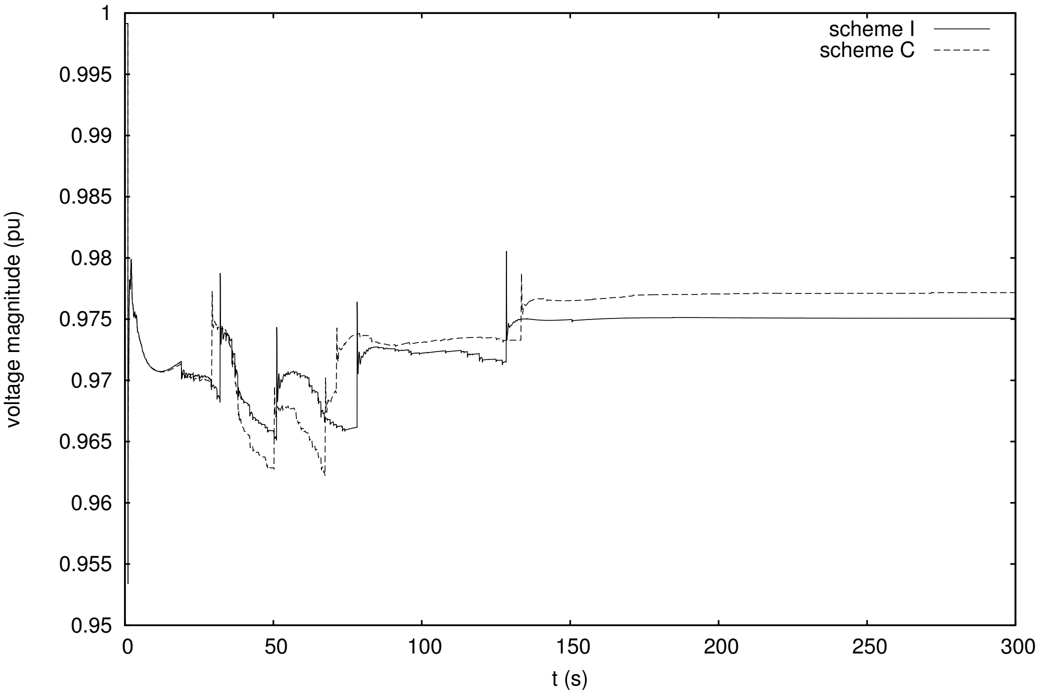


Figure 7.13: Case Q4: voltage at transmission bus 702

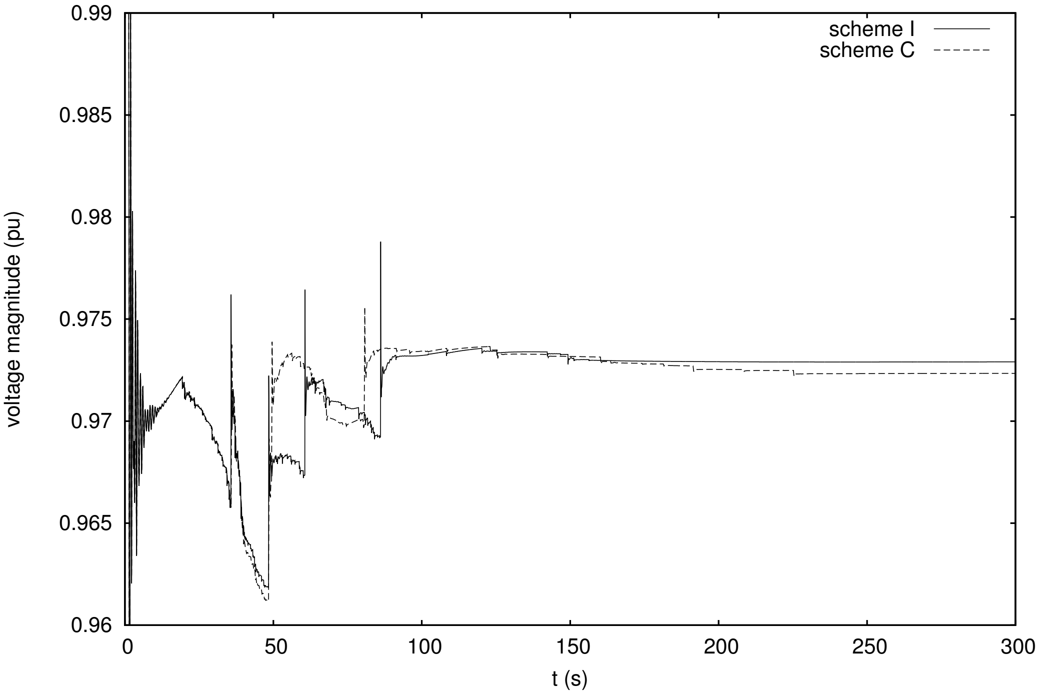


Figure 7.14: Case Q5: zoom of the voltage evolution at transmission bus 702

activations. Again, the discrepancy due to a marginal switch of such a large-impact discrete device is deemed fully acceptable.

7.5.6 Computational effort

Table 7.4 summarizes the gain in computing time which is obtained with scheme C compared to scheme I, while preserving an acceptable level of accuracy. The performance of schemes T, A and L are repeated as well to give a general view over all the proposed techniques. The combination of decomposition, localization and time-averaging brings a significant speed-up ratio in the simulation of long-term cases (i.e. Q2, Q3, Q4 and Q5), which ranges from 4 to 7 with respect to the benchmark.

Table 7.4: CPU times (s)

Case	Scheme I	Scheme T	Scheme A	Scheme L $\epsilon_L = 0.001$	Scheme C
Q1	11.11		5.93	6.04	5.93
Q2	35.84	10.53	11.91	11.06	4.76
Q3	27.89	9.91	13.16	12.19	5.40
Q4	60.16	23.79	30.55	28.51	10.63
Q5	61.34		34.36	32.80	13.80

7.6 Combined simulation results on PEGASE system

This section shows the results obtained with the combined simulation of the PEGASE test system, in all the scenari described in Section 2.6. The outputs have been obtained with a dishonest Newton scheme, combining detailed simulation with localization and time-averaging techniques (denoted "C" in the plots). More details about this and the other solvers are given in Appendix A.5. In this scheme, after a user-imposed disturbance at $t = t_d$, the step will be increased gradually up to its maximum value according to Table 7.5.

7.6.1 Cases P1, P2 and P3

The voltage evolutions at bus F0322411 are shown in Figs. 7.15, 7.16, 7.17 and 7.18 for the three scenari. The approximation brought by combined simulation yields very satisfactory results.

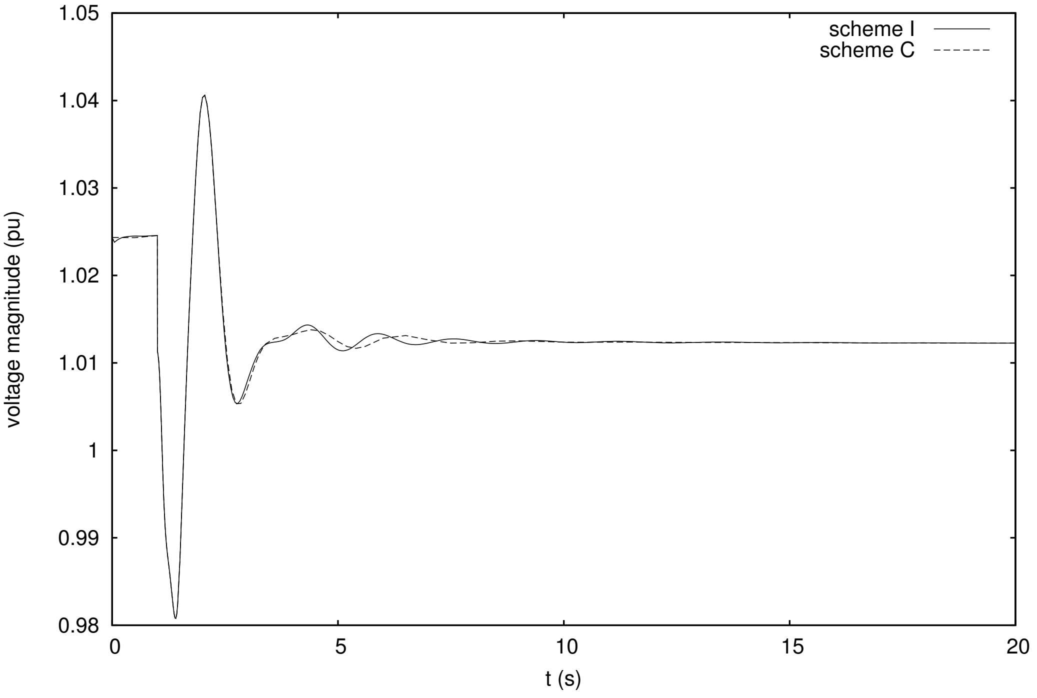


Figure 7.15: Case P1: voltage at transmission bus F0322411

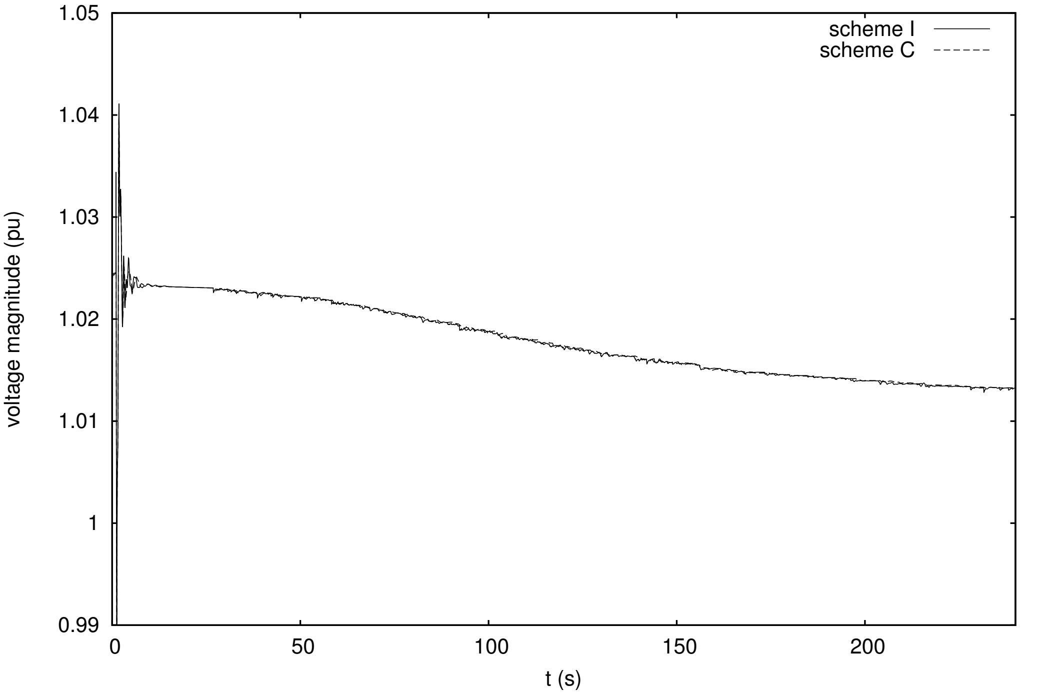


Figure 7.16: Case P2: voltage at transmission bus F0322411

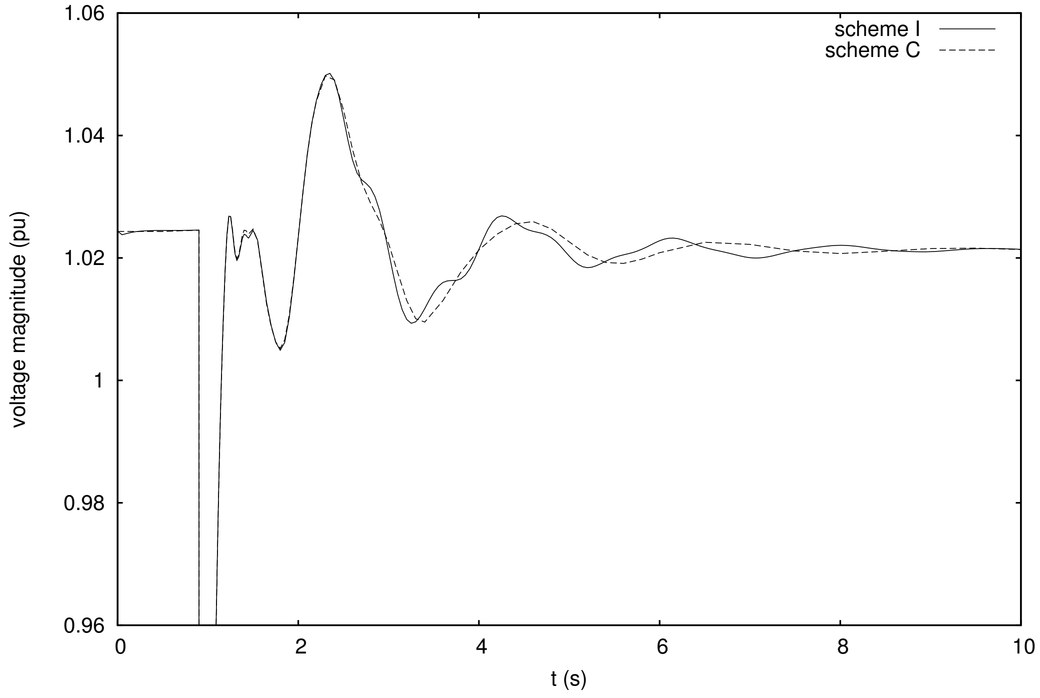


Figure 7.17: Case P3: zoom of the short-term voltage evolution at transmission bus F0322411

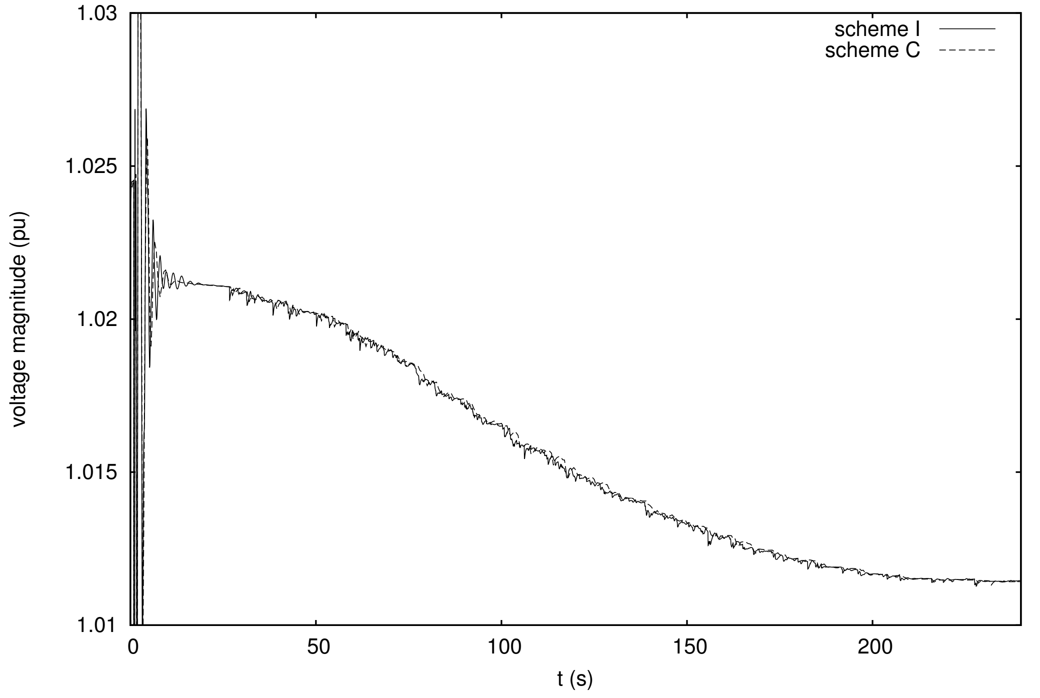


Figure 7.18: Case P3: zoom of the long-term voltage evolution at transmission bus F0322411

Table 7.5: Time step size and integration formulae used

time interval	detailed simulation	combined simulation
$[t_d ; t_d + 0.001]$ s	0.001 s (BEM, $\epsilon_L = 0$)	0.001 s (BEM, $\epsilon_L = 0$)
$[t_d + 0.001 ; t_d + 0.010]$ s	0.009 s (BEM, $\epsilon_L = 0$)	0.009 s (BEM, $\epsilon_L = 0$)
$[t_d + 0.010 ; t_d + 0.170]$ s	0.010 s (BDF2, $\epsilon_L = 0$)	0.010 s (BDF2, $\epsilon_L = 0$)
$[t_d + 0.170 ; t_d + 0.450]$ s	0.020 s (BDF2, $\epsilon_L = 0$)	0.020 s (BDF2, $\epsilon_L = 0$)
$[t_d + 0.450 ; t_d + 1.200]$ s	0.050 s (BDF2, $\epsilon_L = 0$)	0.050 s (BDF2, $\epsilon_L = 0$)
$[t_d + 1.200 ; t_d + 2.400]$ s	same as above	0.100 s (BDF2, $\epsilon_L = 0$)
$[t_d + 2.400 ; t_d + 5.000]$ s	same as above	0.200 s (BDF2, $\epsilon_L = 0$)
$[t_d + 5.000 ; +\infty]$ s	same as above	0.500 s (BDF2, $\epsilon_L = 0.01$)

7.6.2 Computational effort

Tables 7.6 and 7.7 detail the computing times obtained and the number of operations performed in case P1 using the benchmark scheme I and scheme C, respectively. The performance of schemes T, A and L are repeated as well to give a general view over all the proposed techniques. Similar results for Case P2 are given in Tables 7.8 and 7.9, for Case P3 in Tables 7.10 and 7.11. All the tables show that the respective gains gain brought by time-averaging and localization can be successfully combined.

Figures 7.19 and 7.20 show the plots of the elapsed time vs. the simulation time. The bisectrix $y = x$ represents a simulation as fast as real-time (1:1 ratio between elapsed and simulation time), while a curve below the diagonal corresponds to simulation faster than real-time. Apart from the initial transients, scheme C is indeed faster than real-time in both cases. The other schemes are plotted to give a complete view of the presented techniques.

7.7 Chapter conclusion

This chapter hosts the final contribution of this thesis, which is a combination of detailed simulation and accuracy relaxation techniques applied to the decomposed dishonest Newton solution: to our knowledge, the reported gains in CPU time are unprecedented in long-term dynamic simulation of power systems of the size of PEGASE, on single-core computers, while avoiding to develop a simplified model. The gain obtained in the short-term scenario is also significant, thanks mainly to the techniques detailed in Chapter 4.

In the PEGASE system the gain of scheme C ranges between 6 and 16, while in Hydro-Québec system is between 2 and 7, with respect to scheme I. A solver approaching real-time and retaining a satisfactory accuracy is within reach thanks to the presented techniques.

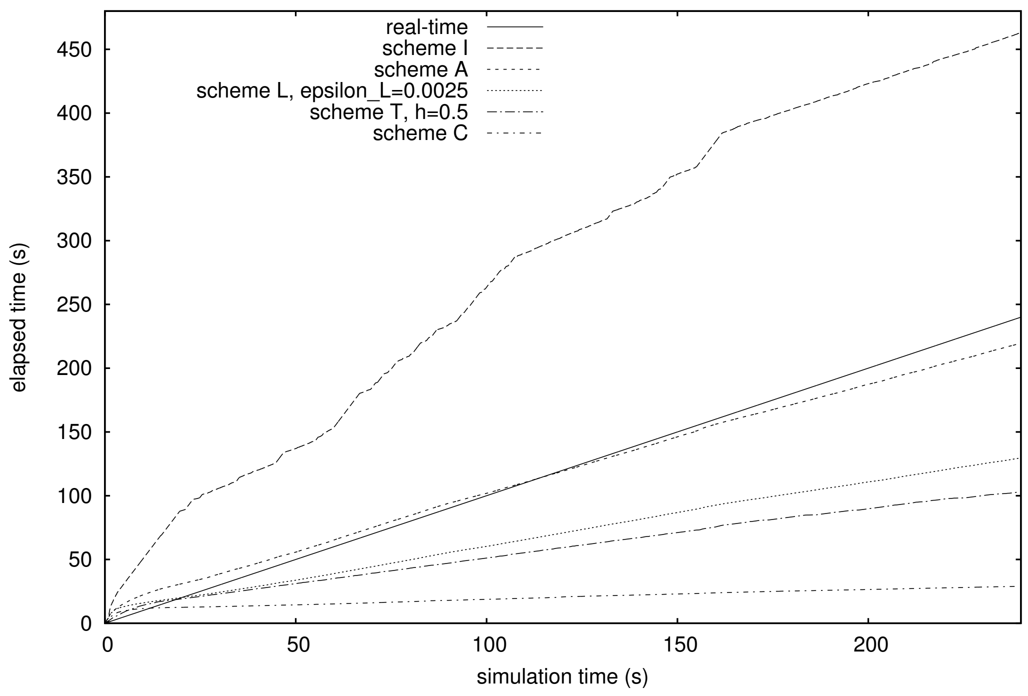


Figure 7.19: Case P2: elapsed time vs. simulation time

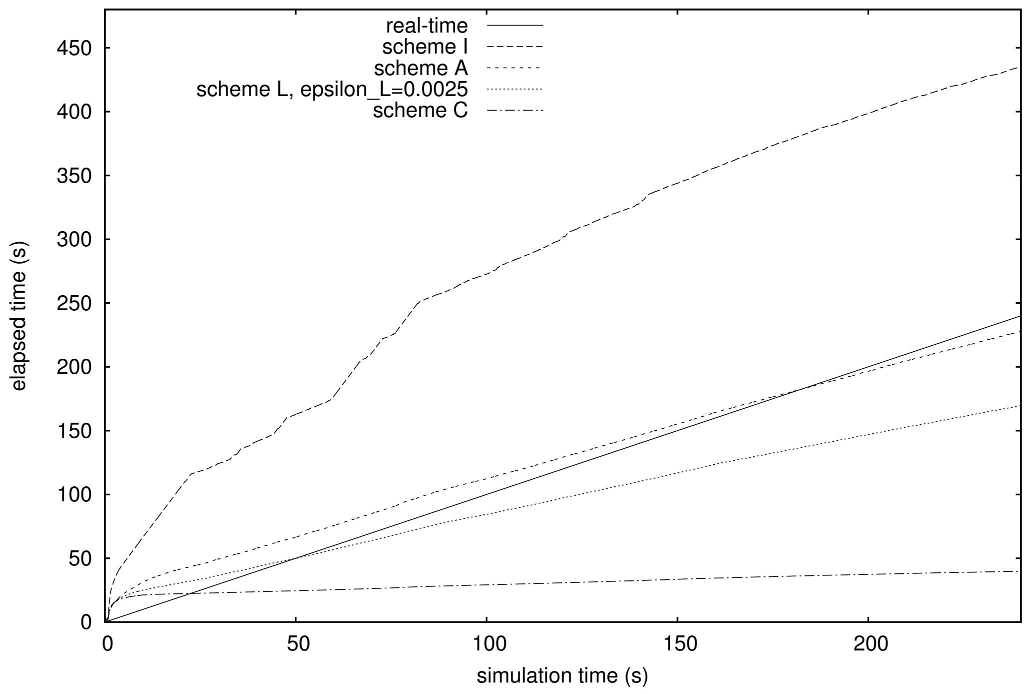


Figure 7.20: Case P3: elapsed time vs. simulation time

Table 7.6: Case P1: CPU times (s)

	Scheme I	Scheme T $h = 0.5$ s		Scheme A	Scheme L $\epsilon_L = 0.0025$	Scheme C
total	90.29	21.13	total	32.81	24.45	15.28
factorizing $\mathbf{J}$	57.51	13.90	factorizing $\tilde{\mathbf{D}}$	0.69	0.76	0.68
solving for $[\mathbf{\Delta V} \;\; \mathbf{\Delta x}]^T$	11.19	2.78	solving for $\mathbf{\Delta V}$	0.82	1.06	0.46
evaluating $[\mathbf{g} \;\; \mathbf{f}]^{\mathbf{T}}$	13.11	2.67	evaluating $\mathbf{g}$	1.97	2.07	0.58
			factorizing $\mathbf{A}_i$ *	2.84	1.23	2.94
			solving for $\mathbf{\Delta x}_i$ *	6.51	4.66	2.89
			evaluating $\mathbf{f}_i$ *	14.75	10.28	5.46

* $i = 1, \ldots, n$

Table 7.7: Case P1: Number of operations

	Scheme I	Scheme T $h = 0.5$ s		Scheme A	Scheme L $\epsilon_L = 0.0025$	Scheme C
factorizing $\mathbf{J}$	446	109	factorizing $\tilde{\mathbf{D}}$	25	26	22
solving for $[\mathbf{\Delta V} \;\; \mathbf{\Delta x}]^T$	1393	355	solving for $\mathbf{\Delta V}$	1031	1009	474
evaluating $[\mathbf{g} \;\; \mathbf{f}]^{\mathbf{T}}$	1891	410	evaluating $\mathbf{g}$	2326	2429	883
			factorizing $\mathbf{A}_i$ *	26	15	27
			solving for $\mathbf{\Delta x}_i$ *	792	478	310
			evaluating $\mathbf{f}_i$ *	2321	1443	847

* average per injector

Table 7.8: Case P2: CPU times (s)

	Scheme I	Scheme T $h = 0.5$ s		Scheme A	Scheme L $\epsilon_L = 0.0025$	Scheme C
total	470.00	105.89	total	220.21	132.41	30.07
factorizing $\mathbf{J}$	207.58	68.80	factorizing $\tilde{\mathbf{D}}$	0.75	0.96	0.75
solving for $[\mathbf{\Delta V} \ \ \mathbf{\Delta x}]^T$	90.00	13.86	solving for $\mathbf{\Delta V}$	7.42	8.25	1.41
evaluating $[\mathbf{g} \ \ \mathbf{f}]^\mathbf{T}$	111.86	13.38	evaluating $\mathbf{g}$	14.04	14.66	2.11
			factorizing $\mathbf{A}_i$ ⋆	2.51	1.25	2.11
			solving for $\mathbf{\Delta x}_i$ ⋆	41.53	20.73	6.05
			evaluating $\mathbf{f}_i$ ⋆	117.25	54.76	12.31

⋆ $i = 1, \ldots, n$

Table 7.9: Case P2: Number of operations

	Scheme I	Scheme T $h = 0.5$ s		Scheme A	Scheme L $\epsilon_L = 0.0025$	Scheme C
factorizing $\mathbf{J}$	1597	511	factorizing $\tilde{\mathbf{D}}$	27	32	24
solving for $[\mathbf{\Delta V} \ \ \mathbf{\Delta x}]^T$	11106	1704	solving for $\mathbf{\Delta V}$	8514	8476	1631
evaluating $[\mathbf{g} \ \ \mathbf{f}]^\mathbf{T}$	15988	2196	evaluating $\mathbf{g}$	17160	18310	2753
			factorizing $\mathbf{A}_i$ ⋆	22	16	20
			solving for $\mathbf{\Delta x}_i$ ⋆	5634	2089	685
			evaluating $\mathbf{f}_i$ ⋆	19044	7192	1842

⋆ average per injector

Table 7.10: Case P3: CPU times (s)

	Scheme I		Scheme A	Scheme L $\epsilon_L = 0.0025$	Scheme C
total	438.21	total	232.56	174.35	41.29
factorizing $\mathbf{J}$	179.40	factorizing $\tilde{\mathbf{D}}$	0.93	1.03	0.94
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	88.93	solving for $\mathbf{\Delta V}$	8.19	7.83	1.80
evaluating $[\mathbf{g}\ \ \mathbf{f}]^{\mathbf{T}}$	111.34	evaluating $\mathbf{g}$	13.91	14.32	2.53
		factorizing $\mathbf{A}_i$ *	4.49	2.56	4.65
		solving for $\mathbf{\Delta x}_i$ *	47.35	29.69	8.64
		evaluating $\mathbf{f}_i$ *	119.44	83.23	16.04

* $i = 1, \ldots, n$

Table 7.11: Case P3: Number of operations

	Scheme I		Scheme A	Scheme L $\epsilon_L = 0.0025$	Scheme C
factorizing $\mathbf{J}$	1374	factorizing $\tilde{\mathbf{D}}$	32	34	28
solving for $[\mathbf{\Delta V}\ \ \mathbf{\Delta x}]^T$	10975	solving for $\mathbf{\Delta V}$	8556	8362	1887
evaluating $[\mathbf{g}\ \ \mathbf{f}]^{\mathbf{T}}$	15946	evaluating $\mathbf{g}$	17020	17337	3165
		factorizing $\mathbf{A}_i$ *	40	27	43
		solving for $\mathbf{\Delta x}_i$ *	5757	3429	893
		evaluating $\mathbf{f}_i$ *	19285	11572	2509

* average per injector

Chapter 8

General conclusion

The value of an idea
lies in the using of it [1]

8.1 Summary of work and main contributions

In this work methods to speed up time-domain simulations of large-scale power systems have been proposed, a typical application being dynamic security assessment. We summarize hereafter the main contributions.

The advantages of a non-causal equation-based hybrid modelling have been stressed. It allows intuitive writing of both physical constraints and control equations, regardless whether they are algebraic or differential, continuous or discrete. Since no causality is imposed, an equation can dynamically change from differential to algebraic and viceversa. This feature has been exploited in the treatment of state events. Also, an explicit center-of-inertia reference frame has been proposed for use in long-term simulations.

The heart of the proposed techniques is a decomposition of the linear equations solved at each Newton iteration, exploiting the bordered block-diagonal structure of the Jacobian matrix. Algorithms have been proposed to minimize the number of partial Jacobian updates and re-factorizations, and perform less iterations on components with lower dynamic

[1]Thomas Alva Edison, in an interview given to E. Hubbard, published in [HB28].

response. The accuracy and convergence properties of the original Newton method are preserved at this stage.

Further speed-ups have been obtained by relaxing accuracy to an extent compatible with applications such as dynamic security assessment. The overall approach offers the important advantage of using the same model as detailed simulations. Hence, model reduction or simplification is not required; instead, the gain in efficiency comes from the relaxable accuracy solver used to simulate the reference model.

Accuracy has been first relaxed in space, by localizing the system response. Localization exploits the fact that, in a large-scale system, most of disturbances give rise to localized effects, i.e. some system components do not participate much in the dynamic response. Latent components are not solved, but replaced by a static linear model relating injected current to bus voltage.

Accuracy has been also relaxed in time by numerically averaging the system response. Time-averaging is obtained by using a step size which is large with respect to the dynamics that are deemed negligible for the application of concern. To this purpose, attention has been paid to the handling of discrete dynamics, performed without identifying the exact instant of the discrete transition, but with an *ex post* treatment of discrete events.

Our final recommendation to achieve faster simulations, the focus being on the long-term response of the system to the initiating disturbance, is a combination of detailed dynamic simulation in the short-term with relaxation of accuracy in the long-term by means of localization and time averaging. The advantage of the proposed method lies in the seamless integration between detailed short-term simulation and relaxed-accuracy long-term simulation, one disadvantage being hyper-stability, which may lead to overlooking instability of oscillations. Tests performed on large-scale systems indicate that very significant reductions in computing time can be obtained when combining the techniques of decomposition, localization and time-averaging, which are the most salient contributions of this thesis.

The results presented in this report were obtained with a mock-up developed at the University of Liège. A wide variety of grids has been considered, namely (i) the Nordic32 system, an academic test system, (ii) the Hydro-Québec system, a real system with 11774 differential-algebraic states, and (iii) within the context of a European project, the PEGASE system, whose size is unprecedented with a total number of 146239 differential-algebraic states.

Most of the corresponding algorithms have been subsequently implemented in a dynamic simulation prototype derived from EUROSTAG [SBDB89] and, for some of them, in a dispatch training simulator prototype based on FAST-DTS [GCA^{+}00].

8.2 Directions for future work

The presented algorithms may be expanded and further improved along the following directions:

- mitigating or bounding the risks coming from the hyper-stability of the presently used integration methods (namely backward differentiation formulae);
- investigating heuristics for automatic switching between detailed, short-term simulation and relaxed-accuracy, long-term simulation;
- investigating techniques for estimating and controlling of the simulation error, for instance through numerical estimations based on Newton iteration count;
- extending the decomposed formulation to multi-terminal HVDC networks with the introduction of internal DC buses into a mixed AC/DC network;
- using the proposed techniques to assess the feasibility of a real implementation of dynamic security assessment, for instance to determine secure operation limits [VVC06].

The work performed in this thesis has also served as a basis for extensions that are being investigated in collaboration with other researchers of the University of Liège:

- using multi-layer domain decomposition algorithms, such as sub-tree reduction method, in order to include distribution networks, whose impact on power system dynamics is likely to increase with the installation of dispersed generation[1];
- including detailed power electronics models into dynamic simulation under the phasor approximation within the proposed injector-network domain decomposition BBD framework, using multi-rate techniques to interface the currents computed with an

[1]See P. Aristidou, D. Fabozzi, and T. Van Cutsem. A Schur complement method for DAE systems in power system simulation. *Accepted for presentation in 21st International Conference on Domain Decomposition Methods*, Rennes, France, June 2012.

electromagnetic transients program with the main transmission network represented in the phasor domain[1];

- parallelizing the computations dealing with injectors for use on multi-core processor architectures, which are becoming increasingly common even in off-the-shelf hardware, and are likely to be the most effective near-future way of increasing computational power[2].

[1]See F. Plumier, D. Fabozzi, C. Geuzaine, and T. Van Cutsem. Combining full transients and phasor approximation models in power system time simulation. *Accepted for presentation in 21st International Conference on Domain Decomposition Methods*, Rennes, France, June 2012.

[2]See P. Aristidou, D. Fabozzi, and T. Van Cutsem. Dynamic simulations of large-scale power systems on parallel architectures using a domain decomposition method. *Submitted for publication in IEEE Transactions on Parallel and Distributed Systems*, 2012.

Appendices

Der liebe Gott steckt im Detail [1]

A.1 Explicit reference frame [FVC11b]

The continuous part of the dynamic model of an electric power system considered in stability studies can be written as shown in Chapter 2 and repeated here for clarity:

$$\mathbf{0} = \mathbf{D}\,\mathbf{V} - \mathbf{C}\,\mathbf{x} \tag{A.1}$$

$$\boldsymbol{\Gamma}(\mathbf{z})\,\dot{\mathbf{x}} = \boldsymbol{\phi}(\mathbf{x},\mathbf{z},\mathbf{V}) \tag{A.2}$$

Equations (A.1) relate to the network, whose response is assumed instantaneous under the phasor approximation. It is appropriate to project these complex equations onto two orthogonal axes, denoted x and y, respectively. Equations (A.2) deal with a variety of short- and long-term phenomena and controls, with corresponding state variables in $\mathbf{x}$. Among them, the motion of the rotor of i-th synchronous machine ($i = 1,\ldots,m$) is described by:

$$\dot{\theta}_i = \omega_i \tag{A.3}$$

$$M_i\dot{\omega}_i = T_{mi} - T_{ei} \tag{A.4}$$

[1] *God is in the detail.* Apocryphal, attributed to art historian Abraham Moritz (Aby) Warburg among others. Also used in its dual form: *der Teufel* ("devil") *steckt im Detail.*

where M_i is an inertia coefficient, T_{mi} (resp. T_{ei}) is the turbine mechanical (resp. generator electromagnetic) torque, ω_i is the rotor speed and θ_i the rotor position with respect to a fixed reference.

Apart from the need to have all phase angles referring to the same axis, the choice of the latter is free. In fact, all variables in $\mathbf{x}$ except angles are unaffected by this choice. The x and y axes should be chosen having in mind computational efficiency.

A standard practice is to choose axes that rotate at the nominal angular frequency $\omega_o = 2\pi f_o$, where f_o is the nominal frequency of the system. The rotor angle of the i-th machine, referred to the x-axis is:

$$\delta_i = \theta_i - (\omega_o t + C) \tag{A.5}$$

where C is an arbitrary constant. This yields:

$$\dot{\delta}_i = \omega_i - \omega_o \tag{A.6}$$

This equation is substituted to (A.3), while (A.4) is unchanged.

While appropriate for short-term simulations (e.g. for transient stability analysis), the above reference suffers from a major drawback in long-term studies. Indeed, after a disturbance affecting the power balance, the system settles at a new angular frequency ω and, with respect to the reference axes, all phasors rotate at the angular speed $\omega - \omega_o$. Hence, although the system settles at a new equilibrium, the δ_i variable increase linearly with time and the components of $\mathbf{v}_x, \mathbf{v}_y, \mathbf{i}_x$ and $\mathbf{i}_y$ oscillate with a period $T = 2\pi|\omega - \omega_o|^{-1}$. This non-linear behaviour requires frequent updates of the Jacobian matrix and increases the number of iterations of the Newton method used to solve the implicit equations. It also requires the time step to remain small compared to T, in order to track those oscillations and/or avoid numerical instability.

The above problem can be solved by somewhat linking the references axes to the rotor motion. This is the idea underlying the Center-Of-Inertia (COI) reference frame, which has been used in direct transient stability analysis methods [TS72] and in industrial simulation software (e.g. [SBDB89]). The COI angle and its time derivative are defined respectively as:

$$\theta_{coi} = \frac{1}{M_T} \sum_{i=1}^{m} M_i \theta_i + K \tag{A.7}$$

$$\omega_{coi} = \frac{1}{M_T} \sum_{i=1}^{m} M_i \omega_i \tag{A.8}$$

where $M_T = \sum_{i=1}^{m} M_i$ is the total inertia and K is an arbitrary constant. The x and y axes now turn at the angular speed ω_{coi}. The rotor angle of the i-th machine, referred to the

x-axis is:

$$\delta_i = \theta_i - \theta_{coi} \tag{A.9}$$

from which one easily derives:

$$\dot{\delta}_i = \dot{\theta}_i - \dot{\theta}_{coi} = \omega_i - \omega_{coi} \tag{A.10}$$

This equation replaces (A.3). Thus, in an m-machine system, there are m equations of the type (A.10) involving m state variables δ_i and the $(m+1)$-th variable ω_{coi}. The latter is included in $\mathbf{y}$ and is balanced by (A.8), an algebraic equation handled by the Newton method together with (A.1). Note that (A.8) can be replaced by $\sum_{i=1}^{m} M_i \delta_i = 0$.

Clearly, if the system settles at a new equilibrium with frequency f, all machines rotate at the angular speed $2\pi f$ and so does the x axis. Hence, $\mathbf{v}_x, \mathbf{v}_y, \mathbf{i}_x$ and $\mathbf{i}_y$ become constant.

This result still holds true when substituting to the right hand side of (A.8) a linear combination $\sum_{i=1}^{m} a_i \omega_i$ with $\sum_{i=1}^{m} a_i = 1$, and in particular a single reference speed. The COI frame of reference, however, is appealing as it provides the average system angular frequency. The latter is appropriately used in frequency-dependant load models as well as in corrective damping terms $D_i(\omega_i - \omega_{coi})$ added to (A.4).

Still, this COI reference suffers from drawbacks. First, the presence of ω_{coi} in each equation (A.10) and in the load models produces a dense column in the Jacobian matrix, while the presence of all ω_i's in (A.8) produces a "dense" row. Next, in case of network split, since one COI has to be considered in each connected subnetwork, the structure of the Jacobian is significantly modified by the added variables and equations. Finally, the algebraic equation (A.8) does not allow to take advantage of the block bordered diagonal structure of the Jacobian, as detailed in Chapter 4.

We propose an explicit approach which is free from these drawbacks. The idea is to make the x axis coincide with a *previous* position of the COI. In the context of numerical simulation, when integrating the equations from $t-h$ to t (where h is the time step), the phasors are projected onto axes that correspond to the COI position at $t-h$. At that time, the COI speed is $\omega_{coi}(t-h)$. The angle and speed of the i-th machine, referred to that axis become:

$$\delta_i = \theta_i - \theta_{coi}(t-h) \tag{A.11}$$

$$\dot{\delta}_i = \omega_i - \omega_{coi}(t-h) \tag{A.12}$$

Note that the COI considered in (A.10) is "implicit" in the sense that it involves the rotor speeds at the same time t, while the one considered in (A.12) is "explicit" in so far as it refers to past, already known speed values. In (A.12), $\omega_{coi}(t-h)$ is just a number, not a variable. Equation (A.8) is not handled together with the other equations. It is used, *after* convergence of the Newton iterations, to determine the value of $\omega_{coi}(t)$ from the just computed rotor speeds $\omega_i(t)$. This updated value of ω_{coi} is used in (A.12) when proceeding with the next integration step.

By so doing, the dense row and column are no longer present in the Jacobian, while network splits do not require any restructuring of the Jacobian.

The advantages of the COI reference frame are also preserved, because a slightly delayed COI angle is as good as the exact COI in making $\mathbf{v}_x, \mathbf{v}_y, \mathbf{i}_x$ and $\mathbf{i}_y$ very little dependent on the system frequency. One could argue that using a slightly delayed ω_{coi} in the frequency dependent load models causes some inaccuracy; however, using ω_{coi} instead of the true angular frequency at the load bus already involves some approximation.

To show the effectiveness of the method we consider a variant of the system used in [FVC09], subject to a branch outage. All responses have been computed using the Trapezoidal method with $h = 0.01$ s.

The long-term evolution of a bus voltage magnitude is shown in Fig. A.1. The system responds to an overexcitation limiter acting at $t \simeq 50$ s and a transformer load tap changer moving by 10 steps, before hitting its limit.

The evolution of ω_{coi} is shown in Fig. A.2. The overall increase is due to the load sensitivity to the dropping voltage. As can be seen, the system settles at equilibrium after $t \simeq 120$ s.

Apart from imperceptible numerical inaccuracies (within the solver tolerances), ω_{coi} is the same, whichever angle reference is used, since the ω_i's are also the same.

Figure A.3 shows with dotted line the evolution of the v_x component of the same bus voltage, when using the angle reference as in (A.5). As expected, v_x undergoes pronounced oscillations, even after the frequency has returned to steady state. The other two curves have been obtained with the angle references defined by (A.10) and (A.12), respectively. The corresponding reference axes differ by the angle $\int_{t-h}^{t} \omega_{coi}(u)\, du = \theta_{coi}(t) - \theta_{coi}(t-h)$, which is small for $h = 0.01$ s, and hence makes the curves indistinguishable. More importantly, these two curves are much smoother than the first one.

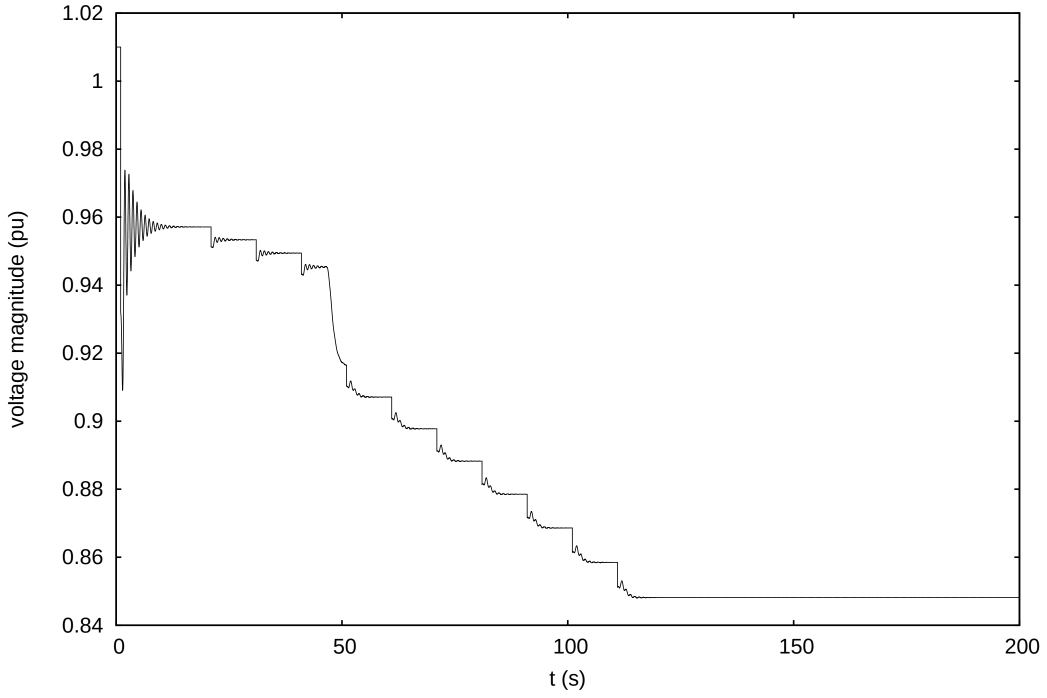


Figure A.1: Evolution of voltage

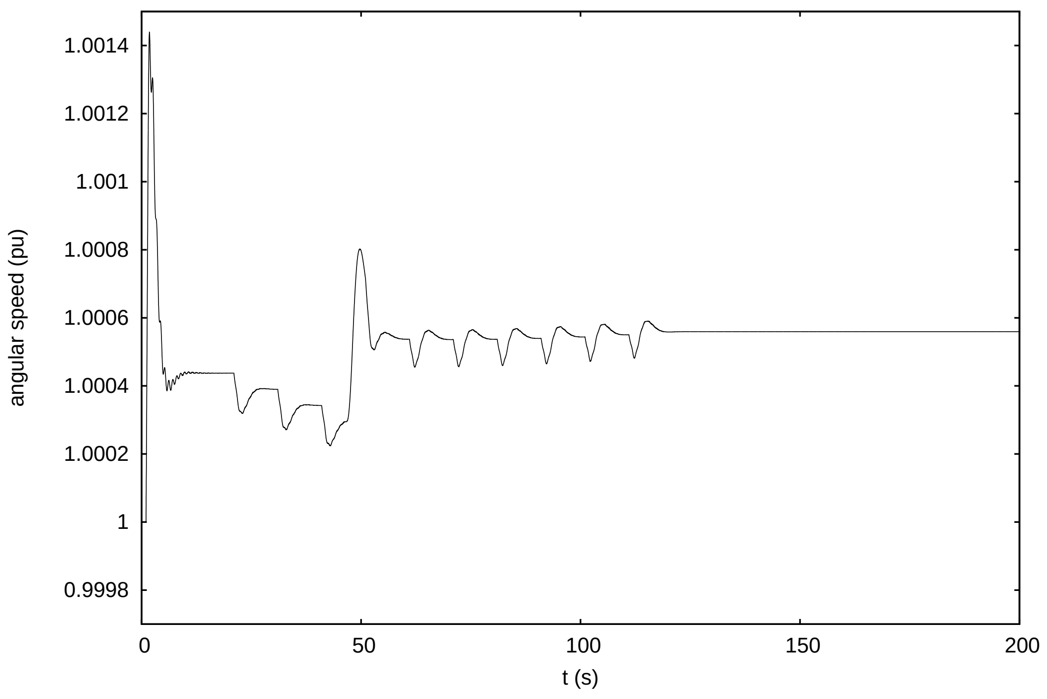


Figure A.2: Evolution of COI angular speed

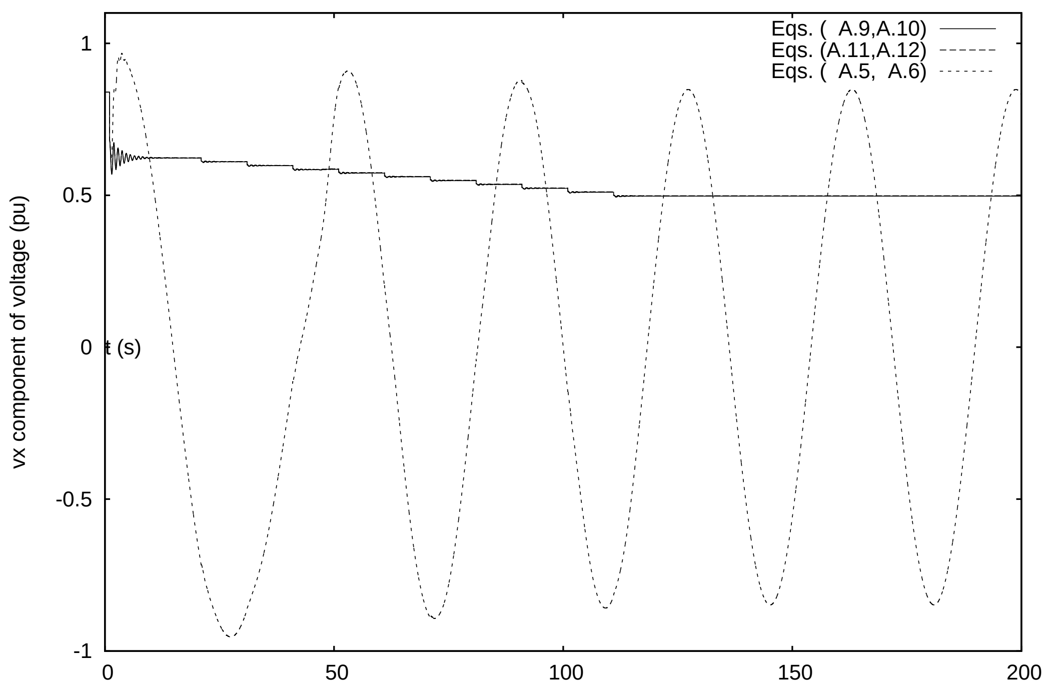


Figure A.3: Evolution of a v_x component

A.2 Synchronous machine model

The advantages of equation-based over input-output modelling mentioned in Section 2.2 can be clarified with a power system example: the synchronous machine model including saturation effects. In order to have a single model, whatever the number of rotor windings, we use "model switches", i.e. integer parameters such as $S_{d1} = 1$ if there is a damper winding $d1$, $S_{d1} = 0$ otherwise, and similarly for $q1$ and $q2$.

The table below shows usual models and their corresponding values of switches:

model	switches
detailed, round rotor	$S_{d1} = 1$, $S_{q1} = 1$, $S_{q2} = 1$
detailed, salient pole	$S_{d1} = 1$, $S_{q1} = 1$, $S_{q2} = 0$
simplified, no damper	$S_{d1} = 0$, $S_{q1} = 0$, $S_{q2} = 0$

Using the equal-mutual-flux-linkage per unit system, the relationship between magnetic flux linkages and currents can be written as:

$$\begin{bmatrix} \psi_d \\ \psi_f \\ \psi_{d1} \end{bmatrix} = \begin{bmatrix} L_\ell + M_d & M_d & S_{d1}M_d \\ M_d & L_{\ell f} + M_d & S_{d1}M_d \\ S_{d1}M_d & S_{d1}M_d & L_{\ell d1} + S_{d1}M_d \end{bmatrix} \begin{bmatrix} i_d \\ i_f \\ i_{d1} \end{bmatrix}$$

$$\begin{bmatrix} \psi_q \\ \psi_{q1} \\ \psi_{q2} \end{bmatrix} = \begin{bmatrix} L_\ell + M_q & S_{q1}M_q & S_{q2}M_q \\ S_{q1}M_q & L_{\ell q1} + S_{q1}M_q & S_{q2}M_q \\ S_{q2}M_q & S_{q2}M_q & L_{\ell q2} + S_{q2}M_q \end{bmatrix} \begin{bmatrix} i_q \\ i_{q1} \\ i_{q2} \end{bmatrix}$$

The d and q components of the air-gap flux are given by:

$$\psi_{ad} = M_d(i_d + i_f + S_{d1}i_{d1}) \quad (A.13)$$

$$\psi_{aq} = M_q(i_q + S_{q1}i_{q1} + S_{q2}i_{q2}) \quad (A.14)$$

From the above equations, and taking into account that the S variables can take values in $\{0, 1\}$ only, one easily obtains:

$$\psi_d = L_\ell i_d + \psi_{ad} \quad (A.15)$$

$$\psi_f = L_{\ell f} i_f + \psi_{ad} \quad (A.16)$$

$$\psi_{d1} = L_{\ell d1} i_{d1} + S_{d1}\psi_{ad} \quad (A.17)$$

$$\psi_q = L_\ell i_q + \psi_{aq} \quad (A.18)$$

$$\psi_{q1} = L_{\ell q1} i_{q1} + S_{q1}\psi_{aq} \quad (A.19)$$

$$\psi_{q2} = L_{\ell q2} i_{q2} + S_{q2}\psi_{aq} \quad (A.20)$$

Using (A.16, A.17, A.19, A.20), the rotor currents are obtained from flux linkages as:

$$i_f = \frac{\psi_f - \psi_{ad}}{L_{\ell f}} \tag{A.21}$$

$$i_{d1} = \frac{\psi_{d1} - S_{d1}\psi_{ad}}{L_{\ell d1}} \tag{A.22}$$

$$i_{q1} = \frac{\psi_{q1} - S_{q1}\psi_{aq}}{L_{\ell q1}} \tag{A.23}$$

$$i_{q2} = \frac{\psi_{q2} - S_{q2}\psi_{aq}}{L_{\ell q2}} \tag{A.24}$$

Let M_d^u and M_q^u be the *unsaturated* direct- and quadrature-axis mutual inductances, related to their corresponding saturated values M_d and M_q by:

$$M_d = \frac{M_d^u}{1 + m\left(\sqrt{\psi_{ad}^2 + \psi_{aq}^2}\right)^n} \tag{A.25}$$

$$M_q = \frac{M_q^u}{1 + m\left(\sqrt{\psi_{ad}^2 + \psi_{aq}^2}\right)^n} \tag{A.26}$$

where m and n relate to the material saturation characteristics.

Replacing M_d by the above expression and i_f and i_{d1} by (A.21, A.22), respectively, the expression (A.13) of the d-axis air-gap flux becomes:

$$\psi_{ad} = \frac{M_d^u}{1 + m\left(\sqrt{\psi_{ad}^2 + \psi_{aq}^2}\right)^n}(i_d + \frac{\psi_f - \psi_{ad}}{L_{\ell f}} + S_{d1}\frac{\psi_{d1} - S_{d1}\psi_{ad}}{L_{\ell d1}})$$

Rearranging terms yields the algebraic equation[1]:

$$\psi_{ad}\left(\frac{1 + m\left(\sqrt{\psi_{ad}^2 + \psi_{aq}^2}\right)^n}{M_d^u} + \frac{1}{L_{\ell f}} + \frac{S_{d1}}{L_{\ell d1}}\right) - i_d - \frac{1}{L_{\ell f}}\psi_f - \frac{S_{d1}}{L_{\ell d1}}\psi_{d1} = 0 \tag{A.27}$$

Similarly we obtain for the q axis:

$$\psi_{aq}\left(\frac{1 + m\left(\sqrt{\psi_{ad}^2 + \psi_{aq}^2}\right)^n}{M_q^u} + \frac{S_{q1}}{L_{\ell q1}} + \frac{S_{q2}}{L_{\ell q2}}\right) - i_q - \frac{S_{q1}}{L_{\ell q1}}\psi_{q1} - \frac{S_{q2}}{L_{\ell q2}}\psi_{q2} = 0 \tag{A.28}$$

The d and q components of the stator voltage relate to their x and y components through:

$$\begin{pmatrix} v_d \\ v_q \end{pmatrix} = \begin{pmatrix} -\sin\delta & \cos\delta \\ \cos\delta & \sin\delta \end{pmatrix} \begin{pmatrix} v_x \\ v_y \end{pmatrix} \tag{A.29}$$

[1]noting that $S_{d1}^2 = S_{d1}$.

and similarly for the current:

$$\begin{pmatrix} i_d \\ i_q \end{pmatrix} = \begin{pmatrix} -\sin\delta & \cos\delta \\ \cos\delta & \sin\delta \end{pmatrix} \begin{pmatrix} i_x \\ i_y \end{pmatrix} \tag{A.30}$$

where δ is the rotor angle (angle between q axis and the reference detailed in Appendix A.1).

Eqs. (A.27, A.28) become respectively:

$$\psi_{ad}\left(\frac{1+m(\sqrt{\psi_{ad}^2+\psi_{aq}^2})^n}{M_d^u}+\frac{1}{L_{\ell f}}+\frac{S_{d1}}{L_{\ell d1}}\right)+\sin\delta\, i_x-\cos\delta\, i_y-\frac{1}{L_{\ell f}}\psi_f-\frac{S_{d1}}{L_{\ell d1}}\psi_{d1}=0 \tag{A.31}$$

Similarly we obtain for the q axis:

$$\psi_{aq}\left(\frac{1+m(\sqrt{\psi_{ad}^2+\psi_{aq}^2})^n}{M_q^u}+\frac{S_{q1}}{L_{\ell q1}}+\frac{S_{q2}}{L_{\ell q2}}\right)-\cos\delta\, i_x-\sin\delta\, i_y-\frac{S_{q1}}{L_{\ell q1}}\psi_{q1}-\frac{S_{q2}}{L_{\ell q2}}\psi_{q2}=0 \tag{A.32}$$

The original Park equations are written as:

$$v_d = -R_a i_d - \omega_N \psi_q \tag{A.33}$$

$$v_q = -R_a i_q + \omega_N \psi_d \tag{A.34}$$

$$\frac{d\psi_f}{dt} = \omega_N (K_f v_f - R_f i_f) \tag{A.35}$$

$$\frac{d\psi_{d1}}{dt} = -\omega_N R_{d1} i_{d1} \tag{A.36}$$

$$\frac{d\psi_{q1}}{dt} = -\omega_N R_{q1} i_{q1} \tag{A.37}$$

$$\frac{d\psi_{q2}}{dt} = -\omega_N R_{q2} i_{q2} \tag{A.38}$$

The stator equations are transformed as follows. Eqs. (A.30) are used to express v_d, v_q, i_d and i_q as functions of v_x, v_y, i_x, i_y, while Eqs. (A.15, A.18) are used to involve ψ_{ad} and $\psi_a q$. This yields for the d axis:

$$\begin{aligned} v_d &= -R_a(-\sin\delta\, i_x+\cos\delta\, i_y)-\omega_N(L_\ell i_q+\psi_{aq}) \\ &= -R_a(-\sin\delta\, i_x+\cos\delta\, i_y)-\omega_N L_\ell(\cos\delta\, i_x+\sin\delta\, i_y)-\omega_N\psi_{aq} \end{aligned}$$

and finally:

$$0=\sin\delta v_x-\cos\delta v_y+(R_a\sin\delta-\omega_N L_\ell\cos\delta)i_x-(R_a\cos\delta+\omega_N L_\ell\sin\delta)i_y-\omega_N\psi_{aq} \tag{A.39}$$

For the q axis we have:

$$\begin{aligned} v_q &= -R_a(\cos\delta\, i_x+\sin\delta\, i_y)+\omega_N(L_\ell i_d+\psi_{ad}) \\ &= -R_a(\cos\delta\, i_x+\sin\delta\, i_y)+\omega_N L_\ell(-\sin\delta\, i_x+\cos\delta\, i_y)+\omega_N\psi_{ad} \end{aligned}$$

and finally:

$$0 = -\cos\delta v_x - \sin\delta v_y - (R_a\cos\delta + \omega_N L_\ell \sin\delta)i_x - (R_a\sin\delta - \omega_N L_\ell\cos\delta)i_y + \omega_N\psi_{ad} \tag{A.40}$$

The rotor equations are transformed by substituting the expressions (A.21, A.22, A.23, A.24) for the rotor currents, thereby obtaining:

$$\frac{d\psi_f}{dt} = \omega_N(K_f v_f - R_f\frac{\psi_f - \psi_{ad}}{L_{\ell f}}) \tag{A.41}$$

$$\frac{d\psi_{d1}}{dt} = -\omega_N R_{d1}\frac{\psi_{d1} - S_{d1}\psi_{ad}}{L_{\ell d1}} \tag{A.42}$$

$$\frac{d\psi_{q1}}{dt} = -\omega_N R_{q1}\frac{\psi_{q1} - S_{q1}\psi_{aq}}{L_{\ell q1}} \tag{A.43}$$

$$\frac{d\psi_{q2}}{dt} = -\omega_N R_{q2}\frac{\psi_{q2} - S_{q2}\psi_{aq}}{L_{\ell q2}} \tag{A.44}$$

The equations of the rotor motion are:

$$\frac{1}{\omega_N}\frac{d\delta}{dt} = \omega - \omega_{coi} \tag{A.45}$$

$$2H\frac{d\omega}{dt} = K_m T_m - T_e - D(\omega - \omega_{coi}) \tag{A.46}$$

in which the expression of the electromagnetic torque T_e is obtained as follows:

$$\begin{aligned} T_e &= \psi_d i_q - \psi_q i_d = (L_\ell i_d + \psi_{ad})i_q - (L_\ell i_q + \psi_{aq})i_d = \psi_{ad}i_q - \psi_{aq}i_d = \\ &= \psi_{ad}(\cos\delta\, i_x + \sin\delta\, i_y) - \psi_{aq}(-\sin\delta\, i_x + \cos\delta\, i_y) \end{aligned} \tag{A.47}$$

In this model it is very convenient to retain in the variable set the i_x and i_y current components and the (d- and q-axis) air gap fluxes ψ_{ad} and ψ_{aq}. The 10 state variables are thus: i_x, i_y, ψ_{ad}, ψ_{aq}, ψ_f, ψ_{d1}, ψ_{q1}, ψ_{q2}, δ and ω. They are balanced by 10 equations:

- 2 algebraic equations of the type (2.2) (stator equations): (A.39, A.40)
- 2 algebraic equations of the type (2.3) (saturation equations) : (A.31, A.32)
- 6 differential equations of the type (2.4) (rotor equations): (A.41, A.42, A.43, A.44, A.45, A.46)

In fact, by resorting to the incidence matrix of the system, it can be shown that the block formed by the equations (A.39, A.40),A.31, A.32) and the variables i_x, i_y, ψ_{ad} and ψ_{aq} constitutes an algebraic loop, in so far as the corresponding incidence matrix cannot be put in a lower triangular form, i.e. a causality cannot be established [CK06], thus the corresponding equations cannot be described by an input-output modelling.

A.3 Restorative load model

The advantages of hybrid equation-based over input-output modelling outlined in Section 2.3 can be clarified with a power system example: the model of the restorative load which has been used in this thesis.

A.3.1 Continuous part of the model

Consider first an admittance load with the following relationships between power and voltage:

$$P = P_0 \left(\frac{V}{V_0}\right)^2 x_p = (V_x^2 + V_y^2)\frac{P_0}{V_0^2} x_p \tag{A.48}$$

$$Q = Q_0 \left(\frac{V}{V_0}\right)^2 x_q = (V_x^2 + V_y^2)\frac{Q_0}{V_0^2} x_q \tag{A.49}$$

where V_0, P_0 and Q_0 refer to the initial values of voltage V, active power P and reactive power Q, and the role of x_p and x_q will be explained in the sequel.

Substituting P and Q with their expressions in terms of voltage (V_x, V_y) and current (i_x, i_y) in Cartesian coordinates yields:

$$V_x i_x + V_y i_y = (V_x^2 + V_y^2)\frac{P_0}{V_0^2} x_p \tag{A.50}$$

$$V_y i_x - V_x i_y = (V_x^2 + V_y^2)\frac{Q_0}{V_0^2} x_q \tag{A.51}$$

Solving (A.50-A.51) with respect to i_x and i_y yields, after some manipulations:

$$i_x = V_x \frac{P_0}{V_0^2} x_p + V_y \frac{Q_0}{V_0^2} x_q \tag{A.52}$$

$$i_y = V_y \frac{P_0}{V_0^2} x_p - V_x \frac{Q_0}{V_0^2} x_q \tag{A.53}$$

The load power restoration process makes x_p and x_q vary according to [VCV08]:

$$\dot{x}_p = \frac{\left(\frac{V}{V_0}\right)^\alpha - x_p \left(\frac{V}{V_0}\right)^2}{T} \quad \text{with} \quad \check{x}_p < x_p < \hat{x}_p \tag{A.54}$$

$$\dot{x}_q = \frac{\left(\frac{V}{V_0}\right)^\beta - x_q \left(\frac{V}{V_0}\right)^2}{T} \quad \text{with} \quad \check{x}_q < x_q < \hat{x}_q \tag{A.55}$$

After a disturbance, for instance a short circuit, the load behaves instantaneously as an admittance (load exponent 2) but after some time, dictated by the time constant T, it restores to an exponential model with exponents α and β.

Indeed, by considering that the dynamics (A.54-A.55) has reached equilibrium, one has:

$$x_p = \frac{\left(\frac{V}{V_0}\right)^{\alpha}}{\left(\frac{V}{V_0}\right)^{2}} \tag{A.56}$$

$$x_q = \frac{\left(\frac{V}{V_0}\right)^{\beta}}{\left(\frac{V}{V_0}\right)^{2}} \tag{A.57}$$

and substituting into (A.48-A.49) yields

$$P = P_0\left(\frac{V}{V_0}\right)^{\alpha} \tag{A.58}$$

$$Q = Q_0\left(\frac{V}{V_0}\right)^{\beta} \tag{A.59}$$

which confirms the above statement.

This model has three advantages. First, it can help convergence when simulating a fault: in the very first moment after the fault inception, loads behave linearly. In this case, T can be set to a low value such as $T = 0.05$ s. Next, the above model can be used to depict some fast restoration observed in practice [VCV08], in which case T can be chosen in the order of a few hundred milliseconds. Finally, the lower limiting of x_p and x_q makes the model switch to constant admittance when the voltage V settles below some value V_{min}, as sketched in Fig. A.4 for active power. Indeed, under limit $x_p = \breve{x}_p$ and $x_q = \breve{x}_q$, the model (A.48-A.49) coincides with that of a constant admittance load.

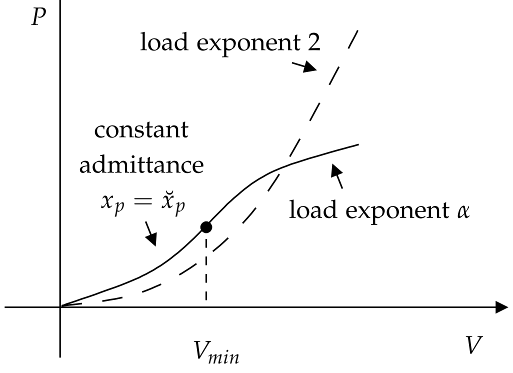


Figure A.4: Static and dynamic active power characteristics

It is easily shown that $\check{x}_p$ and $\check{x}_q$ relate to V_{min} through:

$$\check{x}_p = \left(\frac{V_{min}}{V_0}\right)^{\alpha-2} \tag{A.60}$$

$$\check{x}_q = \left(\frac{V_{min}}{V_0}\right)^{\beta-2} \tag{A.61}$$

A.3.2 Hybrid part of the model

The bounds on x_p and x_q are easily transformed into a model of the type (2.12):

$$(1-z_p^2)\dot{x}_p = (1-z_p^2)\frac{\left(\frac{V}{V_0}\right)^{\alpha} - x_p\left(\frac{V}{V_0}\right)^2}{T_p} + z_p^2 x_p - \sigma(z_p)\hat{x}_p - \sigma(-z_p)\check{x}_p \tag{A.62}$$

$$(1-z_q^2)\dot{x}_q = (1-z_q^2)\frac{\left(\frac{V}{V_0}\right)^{\beta} - x_q\left(\frac{V}{V_0}\right)^2}{T_q} + z_q^2 x_q - \sigma(z_q)\hat{x}_q - \sigma(-z_q)\check{x}_q \tag{A.63}$$

where function σ is equal to 1 if its argument is positive, 0 otherwise. The jump conditions are:

$$\begin{array}{rlll} z_p(t_j^+) = 1 & \text{if} & z_p(t_j^-) = 0 & \text{and } x_p > \hat{x}_p \\ z_p(t_j^+) = -1 & \text{if} & z_p(t_j^-) = 0 & \text{and } x_p < \check{x}_p \\ z_p(t_j^+) = 0 & \text{if} & z_p(t_j^-) = \pm 1 & \text{and } z_p\dot{x}_p < 0 \end{array} \tag{A.64}$$

$$\begin{array}{rlll} z_q(t_j^+) = 1 & \text{if} & z_q(t_j^-) = 0 & \text{and } x_q > \hat{x}_q \\ z_q(t_j^+) = -1 & \text{if} & z_q(t_j^-) = 0 & \text{and } x_q < \check{x}_q \\ z_q(t_j^+) = 0 & \text{if} & z_q(t_j^-) = \pm 1 & \text{and } z_q\dot{x}_q < 0 \end{array} \tag{A.65}$$

where $\dot{x}_p$ is computed from (A.54) and $\dot{x}_q$ from (A.55).

The discrete variables z_p and z_q take values in the finite set $\{-1, 0, 1\}$. Equations (A.62) and (A.63) can thus be decomposed with simple substitutions into the three flows corresponding to each value in the finite set $\{-1, 0, 1\}$.

Indeed for $z_p = 1$ it holds:

$$0 = x_p - \hat{x}_p$$

which simply forces x_p to its upper limit $\hat{x}_p$. For $z_p = -1$, the lower limit $\check{x}_p$ is enforced. In a similar way, the upper $\hat{x}_q$ and lower $\check{x}_q$ limits are enforced on x_q.

A.3.3 Overall model

The four continuous variables i_x, i_y, x_p, x_q and the two discrete variables z_p, z_q are balanced by:

- 2 algebraic equations of the type (2.2) (stator equations): (A.52, A.53)
- 2 algebraic-differential equations of the type (2.3, 2.4) (restoration equations) : (A.62, A.63)
- 2 discrete transition equations of the type (2.5) (jump conditions) : (A.64, A.65)

In fact, the two algebraic-differential equations of the type (2.3, 2.4) are here written in the compact form (A.62, A.63) thanks to the hybrid equation-based formulation (2.12), that allows a flow to switch from differential to algebraic and viceversa. An equivalent block-diagram modelling formulation would not be as straightforward, insofar as a differential-algebraic switch could modify the incidence matrix of the system possibly leading to an algebraic loop problem.

A.4 Computation of BDF coefficients

The polynomial that interpolates $p+1$ points of $\mathbf{w}$ may be written in Newton's form as:

$$\begin{aligned}\boldsymbol{\phi}(t) &= \mathbf{w}(t_{j-p}) + \mathbf{w}[t_{j-p}, t_{j-p+1}](t-t_{j-p}) + \dots \\ &+ \mathbf{w}[t_{j-p}, \dots, t_{j-1}, t_j](t-t_{j-p})\dots(t-t_{j-1})\end{aligned} \tag{A.66}$$

where the divided differences are defined recursively as:

$$\mathbf{w}(t_i) = \mathbf{w}[t_i] \tag{A.67}$$

$$\mathbf{w}[t_{i-k}, t_{i-k+1}, \dots, t_{i-1}, t_i] = \frac{\mathbf{w}[t_{i-k+1}, \dots, t_i] - \mathbf{w}[t_{i-k}, \dots, t_{i-1}]}{t_i - t_{i-k}} \tag{A.68}$$

The p-order BDF method is obtained by imposing Eq. (3.5), hereafter repeated for covenience:

$$\dot{\boldsymbol{\phi}}(t_j) = \dot{\mathbf{w}}(t_j) \tag{A.69}$$

where t_j is the instant where the value of $\mathbf{w}$ has to be computed, while its values at the previous p points are known.

BEM can be easily derived with this procedure, by setting $p = 1$:

$$\boldsymbol{\phi}(t) = \mathbf{w}(t_{j-1}) + \mathbf{w}[t_{j-1}, t_j](t-t_{j-1}) = \mathbf{w}(t_{j-1}) + \frac{\mathbf{w}(t_j) - \mathbf{w}(t_{j-1})}{t_j - t_{j-1}}(t - t_{j-1}) \tag{A.70}$$

Taking the time derivative of (A.70) and substituting in (A.69) yields:

$$\dot{\mathbf{w}}(t_j) = \frac{\mathbf{w}(t_j) - \mathbf{w}(t_{j-1})}{t_j - t_{j-1}}$$

The classical BEM formula (3.2) is then obtained by isolating $\mathbf{w}(t_j)$ and setting $h = t_j - t_{j-1}$:

$$\mathbf{w}(t_j) = \mathbf{w}(t_{j-1}) + h\dot{\mathbf{w}}(t_j) \tag{A.71}$$

In order to obtain the BDF2 coefficients, let us consider 3 interpolation points, i.e. $p = 2$:

$$\begin{aligned}\boldsymbol{\phi}(t) &= \mathbf{w}(t_{j-2}) + \mathbf{w}[t_{j-2}, t_{j-1}](t-t_{j-2}) + \mathbf{w}[t_{j-2}, t_{j-1}, t_j](t-t_{j-2})(t-t_{j-1}) \\ &= \mathbf{w}(t_{j-2}) + \frac{\mathbf{w}(t_{j-1}) - \mathbf{w}(t_{j-2})}{t_{j-1} - t_{j-2}}(t-t_{j-2}) + \frac{\mathbf{w}[t_{j-1}, t_j] - \mathbf{w}[t_{j-2}, t_{j-1}]}{t_j - t_{j-2}}(t-t_{j-2})(t-t_{j-1}) \\ &= \mathbf{w}(t_{j-2}) + \frac{\mathbf{w}(t_{j-1}) - \mathbf{w}(t_{j-2})}{t_{j-1} - t_{j-2}}(t-t_{j-2}) \\ &+ \left(\frac{\mathbf{w}(t_j) - \mathbf{w}(t_{j-1})}{(t_j - t_{j-1})(t_j - t_{j-2})} - \frac{\mathbf{w}(t_{j-1}) - \mathbf{w}(t_{j-2})}{(t_{j-1} - t_{j-2})(t_j - t_{j-2})}\right)(t-t_{j-2})(t-t_{j-1})\end{aligned} \tag{A.72}$$

Taking the time derivative of (A.72) yields:

$$\begin{aligned}\dot{\boldsymbol{\phi}}(t) &= \frac{\mathbf{w}(t_{j-1})-\mathbf{w}(t_{j-2})}{t_{j-1}-t_{j-2}} \\ &+ \left(\frac{\mathbf{w}(t_j)-\mathbf{w}(t_{j-1})}{(t_j-t_{j-1})(t_j-t_{j-2})} - \frac{\mathbf{w}(t_{j-1})-\mathbf{w}(t_{j-2})}{(t_{j-1}-t_{j-2})(t_j-t_{j-2})}\right)(2t-t_{j-1}-t_{j-2})\end{aligned} \tag{A.73}$$

which is then evaluated in t_j:

$$\begin{aligned}\dot{\boldsymbol{\phi}}(t_j) &= \frac{\mathbf{w}(t_{j-1})-\mathbf{w}(t_{j-2})}{t_{j-1}-t_{j-2}} \\ &+ \left(\frac{\mathbf{w}(t_j)-\mathbf{w}(t_{j-1})}{(t_j-t_{j-1})(t_j-t_{j-2})} - \frac{\mathbf{w}(t_{j-1})-\mathbf{w}(t_{j-2})}{(t_{j-1}-t_{j-2})(t_j-t_{j-2})}\right)(2t_j-t_{j-1}-t_{j-2})\end{aligned} \tag{A.74}$$

Without loss of generality, the notation in (A.74) can be simplified by introducing $h = t_{j-1}-t_{j-2}$ and $\xi h = t_j - t_{j-1}$:

$$\begin{aligned}\dot{\boldsymbol{\phi}}(t_j) &= \frac{\mathbf{w}(t_{j-1})-\mathbf{w}(t_{j-2})}{h} + \left(\frac{\mathbf{w}(t_j)-\mathbf{w}(t_{j-1})}{(\xi h)(h+\xi h)} - \frac{\mathbf{w}(t_{j-1})-\mathbf{w}(t_{j-2})}{(h)(h+\xi h)}\right)(h+2\xi h) \\ &= \frac{\mathbf{w}(t_{j-1})-\mathbf{w}(t_{j-2})}{h} + \left(\frac{\mathbf{w}(t_j)-\mathbf{w}(t_{j-1})}{(\xi h)(1+\xi)} - \frac{\mathbf{w}(t_{j-1})-\mathbf{w}(t_{j-2})}{(h)(1+\xi)}\right)(1+2\xi) \\ &= \frac{1}{h}\left(\mathbf{w}(t_j)\frac{1+2\xi}{\xi(1+\xi)} + \mathbf{w}(t_{j-1})\left(1-\frac{1+2\xi+\xi+2\xi^2}{\xi(1+\xi)}\right) + \mathbf{w}(t_{j-2})\left(\frac{1+2\xi}{1+\xi}-1\right)\right) \\ &= \frac{1}{h}\left(\mathbf{w}(t_j)\frac{1+2\xi}{\xi(1+\xi)} - \mathbf{w}(t_{j-1})\frac{1+2\xi+\xi^2}{\xi(1+\xi)} + \mathbf{w}(t_{j-2})\frac{\xi}{1+\xi}\right) \\ &= \frac{1}{h}\left(\mathbf{w}(t_j)\frac{1+2\xi}{\xi(1+\xi)} - \mathbf{w}(t_{j-1})\frac{1+\xi}{\xi} + \mathbf{w}(t_{j-2})\frac{\xi}{1+\xi}\right)\end{aligned} \tag{A.75}$$

By substituting (A.69) into (A.75) we obtain

$$\dot{\mathbf{w}}(t_j) = \frac{1}{h}\left(\mathbf{w}(t_j)\frac{1+2\xi}{\xi(1+\xi)} - \mathbf{w}(t_{j-1})\frac{1+\xi}{\xi} + \mathbf{w}(t_{j-2})\frac{\xi}{1+\xi}\right)$$

which can be rearranged and fitted to the classical BDF form (3.4) by isolating $\mathbf{w}(t_j)$:

$$\mathbf{w}(t_j) = \frac{(1+\xi)^2}{1+2\xi}\mathbf{w}(t_{j-1}) - \frac{\xi^2}{1+2\xi}\mathbf{w}(t_{j-2}) + h\frac{\xi(1+\xi)}{1+2\xi}\dot{\mathbf{w}}(t_j) \tag{A.76}$$

$$= \sum_{\ell=1}^{2}\gamma_\ell\,\mathbf{w}(t_{j-\ell}) + h\,\beta\,\dot{\mathbf{w}}(t_j) \tag{A.77}$$

where

$$\gamma_1 = \frac{(1+\xi)^2}{1+2\xi} \tag{A.78}$$

$$\gamma_2 = -\frac{\xi^2}{1+2\xi} \tag{A.79}$$

$$\beta = \frac{\xi(1+\xi)}{1+2\xi} \tag{A.80}$$

It can be easily verified that, by imposing $\xi = 1$, i.e. $h = t_{j-1}-t_{j-2} = t_j - t_{j-1}$, the fixed-step BDF2 coefficients ($\gamma_1 = \frac{4}{3}$,$\gamma_2 = -\frac{1}{3}$,$\beta = \frac{2}{3}$) shown in Table 3.1 are obtained.

A.5 Implementation aspects

The program used for the simulation results shown in this thesis, named RAMSES[1] has been developed since 2008, at the University of Liège, jointly by the author and Prof. Thierry Van Cutsem and, since 2011, with the help of Petros Aristidou. Most of the software is written in Fortran 95 language.

The partial derivatives relative to network and synchronous machine equations are computed analytically, while those relative to the machine controllers as well as the other injectors are evaluated by finite differences.

In the integrated schemes (I and T), the `ma41` sparse solver from Harwell [hsl12] is used to deal with the system of equations (2.11) and (3.6), considered as a whole. The Jacobian matrix factorization is dealt with in a dishonest way, following a classical strategy that consists in updating the Jacobian after major discrete changes (thus excluding LTC moves, for instance) and in general after E iterations are performed with the same factors without achieving convergence. E has been set to 3, which has been found to offer a good compromise between the burden of updating the factors and the burden of performing too many Newton iterations.

The schemes based on BBD decomposition (A, L and C) use the `DGETR` BLAS routines [LHKK79] for dense matrices to deal with the injector systems (4.9) and the `ma37` sparse solver from Harwell [hsl12] to deal with the network system (4.11). The Jacobian matrix factorization is again dishonest, and different strategies are adopted for the injectors and for the network. The network matrix is refactorized after major discrete changes (LTC moves excluded, for instance), and after E iterations are performed with the same factors without achieving convergence: in this case E is set to a higher value, i.e. 5, than in the integrated case, because this matrix is almost constant in between topological changes; indeed it would be constant if it was not for the corrections $-\tilde{\mathbf{C}}_i\mathbf{B}_i$ (see Section 4.2). The injector systems are factorized: (i) after a step size variation, (ii) after a discrete change affecting the specific injector, and (iii) if the ratio between one injector mismatch (4.9) at iterations $k+1$ and k is higher than a given threshold r at a given iteration. Setting $r = 0.75$ proved to be a good compromise.

The CPU times have been measured on a standard laptop with the following characteristics: Intel Core i7-2630QM CPU, 2.9 GHz, 8 GB RAM, running under Ubuntu Linux 11.04,

[1]Acronym for "Relaxable Accuracy Multithreaded Simulator of Electric power Systems".

using one core. The code profiling results have been obtained with Intel VTune Amplifier XE on the same platform, and do not take into account the time spent in data reading and loading.

Bibliography

Ὁ βίος βραχύς, ἡ δὲ τέχνη μακρή[1]

[ABJ94] J. Y. Astic, A. Bihain, and M. Jerosolimski. The mixed Adams-BDF variable step size algorithm to simulate transient and long term phenomena in power systems. *IEEE Trans. Power Syst.*, 9(2):929–935, May 1994. 50, 124

[ANL^{+}12] U. D. Annakkage, N. K. C. Nair, Y. Liang, A. M. Gole, V. Dinavahi, B. Gustavsen, T. Noda, H. Ghasemi, A. Monti, M. Matar, R. Iravani, and J. A. Martinez. Dynamic System Equivalents: A Survey of Available Techniques. *IEEE Trans. Power Delivery*, 27(1):411–420, Jan. 2012. 96

[AP98] U. M. Ascher and L. R. Petzold. *Computer methods for ordinary differential equations and differential-algebraic equations*. Miscellaneous Titles in Applied Mathematics Series. Society for Industrial and Applied Mathematics, 1998. 22, 23, 26, 42, 44, 46, 124

[AST83] O. Alsac, B. Stott, and W. F. Tinney. Sparsity-Oriented Compensation Methods for Modified Network Solutions. *IEEE Trans. Power Apparatus and Systems*, PAS-102(5):1050–1060, May 1983. 60

[1] *The lyf so short, the craft so long to lerne* [sic]. Hippocrates of Cos in *Aphorisms*. Also known in its Latin form *ars longa, vita brevis*. Free translation in Middle English by Geoffrey Chaucer in *The Parliament of Fowles*.

[BET91] R. Bacher, G. C. Ejebe, and W. F. Tinney. Approximate sparse vector techniques for power network solutions. *IEEE Trans. Power Syst.*, 6(1):420–428, Feb. 1991. 22, 96

[BGH72] R. K. Brayton, F. G. Gustavson, and G. D. Hachtel. A new efficient algorithm for solving differential-algebraic systems using implicit backward differentiation formulas. *Proceedings of the IEEE*, 60(1):98–108, Jan. 1972. 43

[BL89] V. Brandwajn and M. G. Lauby. Complete bounding method for AC contingency screening. *IEEE Trans. Power Syst.*, 4(2):724–729, May 1989. 96

[Boe77] Boeing Computer Services, Inc. *Power System Dynamic Analysis Phase I*. EPRI EL-484 Project 670-1, Final report. EPRI, 1977. 21, 58

[Bos03] A. Bose. *Power System Stability: New Opportunities for Control*. Book Chapter in *Stability and Control of Dynamical Systems and Applications*, D. Liu and P. J. Antsaklis, Editors. Birkhäuser, 2003. 57

[Bra88] V. Brandwajn. Efficient bounding method for linear contingency analysis. *IEEE Trans. Power Syst.*, 3(1):38–43, Feb. 1988. 96

[Bra93] V. Brandwajn. Localization Concepts in (In)-Security Analysis. In *Proc. 1993 Athens Power Tech Conf.*, volume 1, pages 10–15, Sept. 1993. 22, 95

[BT89] R. Bacher and W. F. Tinney. Faster local power flow solutions: the zero mismatch approach. *IEEE Trans. Power Syst.*, 4(4):1345–1354, Nov. 1989. 22, 96

[BTS96] S. Bernard, G. Trudel, and G. Scott. A 735 kV shunt reactors automatic switching system for Hydro-Québec network. *IEEE Trans. Power Syst.*, 11(4):2024–2030, Nov. 1996. 35

[BW98] D. Baraff and A. Witkin. Large steps in cloth simulation. In *Proc. 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH ’98, pages 43–54, New York, NY, USA, 1998. ACM. 22, 125

[CB86] S. M. Chan and V. Brandwajn. Partial Matrix Refactorization. *IEEE Trans. Power Syst.*, 1(1):193–199, Feb. 1986. 60

[CC94] M. L. Crow and J. G. Chen. The multirate method for simulation of power system dynamics. *IEEE Trans. Power Syst.*, 9(3):1684–1690, Aug. 1994. 20, 59

[CC96] M. L. Crow and J. G. Chen. The multirate simulation of FACTS devices in power system dynamics. *IEEE Trans. Power Syst.*, 11(1):376–382, Feb. 1996. 59

[CC08] J. Chen and M. L. Crow. A variable partitioning strategy for the multirate method in power systems. *IEEE Trans. Power Syst.*, 23(2):259–266, May 2008. 22, 59, 97, 123

[Cho82] J. H. Chow. *Time-scale modeling of dynamic networks with applications to power systems*. Lecture notes in control and information sciences. Springer-Verlag, 1982. 96

[CI90] M. L. Crow and M. Ilic. The parallel implementation of the waveform relaxation method for transient stability simulations. *IEEE Trans. Power Syst.*, 5(3):922–932, Aug. 1990. 59

[CK06] F. E. Cellier and E. Kofman. *Continuous system simulation*. Springer, 2006. 22, 26, 27, 28, 31, 44, 49, 50, 52, 124, 186

[CZBT91] J. S. Chai, N. Zhu, A. Bose, and D. J. Tylavsky. Parallel Newton type methods for power system stability analysis using local and shared memory multiprocessors. *IEEE Trans. Power Syst.*, 6(4):1539–1545, Nov 1991. 20

[DS72] H. W. Dommel and N. Sato. Fast Transient Stability Soultions. *IEEE Trans. Power Apparatus and Systems*, PAS-91(4):1643–1650, July 1972. 46

[EFV07] A. Efrat, Q. Fan, and S. Venkatasubramanian. Curve Matching, Time Warping, and Light Fields: New Algorithms for Computing Similarity between Curves. *Journal of Mathematical Imaging and Vision*, 27(3):203–216, Apr. 2007. 54

[EPT92] G. C. Ejebe, R. F. Paliza, and W. F. Tinney. An adaptive localization method for real-time security analysis. *IEEE Trans. Power Syst.*, 7(2):777–783, May 1992. 22, 96

[FCPVC11] D. Fabozzi, A. S. Chieh, P. Panciatici, and T. Van Cutsem. On simplified handling of state events in time-domain simulation. In *Proc. 17th Power System Computation Conference (PSCC)*, pages 1–9, Aug. 2011. 23, 24, 126, 128

[FGWVC09] D. Fabozzi, M. Glavic, L. Wehenkel, and T. Van Cutsem. Security assessment by multiple transmission system operators exchanging sensitivity and tie-line power flow information. In *Proc. 2009 IEEE Bucharest PowerTech conf.*, pages 1 –8, June 2009. 96

[FMT12] C. Fu, J. D. McCalley, and J. Tong. A numerical solver design for extended-term time-domain simulation. *IEEE Trans. Power Syst.*, 2012. IEEE Early Access. 50

[FP78] J. Fong and C. Pottle. Parallel Processing of Power System Analysis Problems Via Simple Parallel Microcomputer Structures. *IEEE Trans. Power Apparatus and Systems*, PAS-97(5):1834–1841, Sept. 1978. 58

[FVC09] D. Fabozzi and T. Van Cutsem. Simplified time-domain simulation of detailed long-term dynamic models. In *Proc. 2009 IEEE PES General Meeting*, pages 1–8, July 2009. 22, 23, 24, 49, 124, 125, 129, 130, 180

[FVC10] D. Fabozzi and T. Van Cutsem. Localization and latency concepts applied to time simulation of large power systems. In *Proc. 2010 Bulk Power System Dynamics and Control - VIII IREP Symposium*, pages 1–14, Aug. 2010. 24, 101

[FVC11a] D. Fabozzi and T. Van Cutsem. Assessing the proximity of time evolutions through dynamic time warping. *IET Generation, Transmission & Distribution*, 5(12):1268–1276, Dec. 2011. 24, 54, 56

[FVC11b] D. Fabozzi and T. Van Cutsem. On angle references in long-term time-domain simulations. *IEEE Trans. Power Syst.*, 26(1):483–484, Feb. 2011. 24, 28, 177

[FX06] D. Z. Fang and Y. Xiaodong. A new method for fast dynamic simulation of power systems. *IEEE Trans. Power Syst.*, 21(2):619–628, May 2006. 59

[Gal84] F. D. Galiana. Bound Estimates of the Severity of Line Outages in Power System Contingency Analysis and Ranking. *IEEE Trans. Power Apparatus and Systems*, PAS-103(9):2612–2624, Sept. 1984. 96

[gC07] CIGRE Working group C4.601. *Review of On-line Dynamic Security Assessment Tools and Techniques.* Technical Brochure 325. CIGRE, 2007. 20

[GCA$^+$00] S. Gissinger, P. Chaumes, J.-P. Antoine, A. Bihain, and M. Stubbe. Advanced dispatcher training simulator. *Computer Applications in Power, IEEE*, 13(2):25–30, Apr. 2000. 175

[Gea71] C. W. Gear. *Numerical initial value problems in ordinary differential equations*. Prentice-Hall series in automatic computation. Prentice-Hall, 1971. 43, 49, 129

[GLVC05] M. E. Grenier, D. Lefebvre, and T. Van Cutsem. Quasi steady-state models for long-term voltage and frequency dynamics simulation. In *Power Tech, 2005 IEEE Russia*, pages 1–8, June 2005. 122

[GP78] A. J. Germond and R. Podmore. Dynamic Aggregation of Generating Unit Models. *IEEE Trans. Power Apparatus and Systems*, PAS-97(4):1060–1069, July 1978. 96

[HB28] E. Hubbard and F. Bann. *Little Journeys to the Homes of the Great*, volume 1. W.H. Wise & Co., 1928. 173

[HBJVN77] W. L. Hatcher, F. M. Brasch Jr., and J. E. Van Ness. A feasibility study for the solution of transient stability problems by multiprocessor structures. *IEEE Trans. Power Apparatus and Systems*, 96(6):1789–1797, Nov. 1977. 21, 58

[HP00] I. A. Hiskens and M. A. Pai. Trajectory sensitivity analysis of hybrid systems. *IEEE Trans. Circuits and Systems I: Fundamental Theory and Applications*, 47(2):204–220, Feb. 2000. 27, 30, 126

[hsl12] *HSL. A collection of Fortran codes for large scale scientific computation.* http://www.hsl.rl.ac.uk. accessed May 2012. 193

[HSV81] G. D. Hachtel and A. L. Sangiovanni-Vincentelli. A survey of third-generation simulation techniques. *Proceedings of the IEEE*, 69(10):1264–1280, Oct. 1981. 22, 95

[IEE92] IEEE Task Force of Power System Engineering Committee. Parallel processing in power systems computation. *IEEE Trans. Power Syst.*, 7(2):629–638, May 1992. 47

[ISCP87] M. Ilic′-Spong, M. L. Crow, and M. A. Pai. Transient Stability Simulation by Waveform Relaxation Methods. *IEEE Trans. Power Syst.*, 2(4):943–949, Nov. 1987. 20, 59

[JMD09] V. Jalili-Marandi and V. Dinavahi. Instantaneous Relaxation-Based Real-Time Transient Stability Simulation. *IEEE Trans. Power Syst.*, 24(3):1327–1336, Aug. 2009. 20, 59

[JMD10] V. Jalili-Marandi and V. Dinavahi. SIMD-Based Large-Scale Transient Stability Simulation on the Graphics Processing Unit. *IEEE Trans. Power Syst.*, 25(3):1589–1599, Aug. 2010. 20

[KL10] Y. Kong and S. Liu. Power system transient stability assessment based on geometric features of disturbed trajectory and DTW. In *Proc. 6th International Conference on Natural Computation (ICNC 2010)*, pages 485–489, Aug. 2010. 54

[KMC08] S. K. Khaitan, J. D. McCalley, and Q. Chen. Multifrontal Solver for Online Power System Time-Domain Simulation. *IEEE Trans. Power Syst.*, 23(4):1727–1737, Nov. 2008. 21

[KOO$^+$93] A. Kurita, H. Okubo, K. Oki, S. Agematsu, D. B. Klapper, N. W. Miller, W. W. Price Jr., J. J. Sanchez-Gasca, K. A. Wirgau, and T. D. Younkins. Multiple time-scale power system dynamic simulation. *IEEE Trans. Power Syst.*, 8(1):216–223, Feb. 1993. 23, 150

[KP01] E. J. Keogh and M. J. Pazzani. Derivative Dynamic Time Warping. In *First SIAM International Conference on Data Mining (SDM'2001*, pages 1–11, Apr. 2001. 54

[Kun94] P. Kundur. *Power system stability and control*. The EPRI power system engineering series. McGraw-Hill, 1994. 19, 21, 27

[Lar04] M. Larsson. ObjectStab-an educational tool for power system stability studies. *IEEE Trans. Power Syst.*, 19(1):56–63, Feb. 2004. 28

[LHKK79] C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh. Basic Linear Algebra Subprograms for Fortran Usage. *ACM Trans. Math. Softw.*, 5(3):308–323, Sept. 1979. 193

[LRLVC01] L. Loud, P. Rousseaux, D. Lefebvre, and T. Van Cutsem. A time-scale decomposition-based simulation tool for voltage stability analysis. In *Power Tech Proceedings, 2001 IEEE Porto*, volume 2, page 6 pp. vol.2, Sept. 2001. 151

[LSBTC90] M. La Scala, A. Bose, D. J. Tylavsky, and J. S. Chai. A highly parallel method for transient stability analysis. *IEEE Trans. Power Syst.*, 5(4):1439–1446, Nov. 1990. 20, 59

[LSBTT90] M. La Scala, M. Brucoli, F. Torelli, and M. Trovato. A Gauss-Jacobi-Block-Newton method for parallel transient stability analysis [of power systems]. *IEEE Trans. Power Syst.*, 5(4):1168–1177, Nov. 1990. 59

[Lyg04] J. Lygeros. *Lecture Notes on Hybrid Systems*. ENSIETA 2-6/2/2004. Department of Electrical and Computer Engineering, University of Patras, 2004. 31, 49

[MDGS79] A. Monticelli, S. Deckmann, A. Garcia, and B. Stott. Real-Time External Equivalents for Static Security Analysis. *IEEE Trans. Power Apparatus and Systems*, PAS-98(2):498–508, Mar. 1979. 96

[MFL$^+$05] A. Monti, H. Figueroa, S. Lentijo, X. Wu, and R. Dougal. Interface Issues in Hardware-in-the-Loop Simulation. In *Electric Ship Technologies Symposium, 2005 IEEE*, pages 39–45, July 2005. 128

[Mil12] F. Milano. Extraneous instabilities arising in power systems with non-synchronous distributed energy resources. *International Journal of Electrical Power & Energy Systems*, 34(1):174–176, 2012. 27

[ML89] J. R. Marti and J. Lin. Suppression of numerical oscillations in the EMTP power systems. *IEEE Trans. Power Syst.*, 4(2):739–747, May 1989. 23, 125

[MR06] F. Magoulès and F.-X. Roux. Lagrangian formulation of domain decomposition methods: A unified theory. *Applied Mathematical Modelling*, 30(7):593–615, 2006. 20, 59

[Nor61] A. Nordsieck. *On numerical integration of ordinary differential equations*. Univ. of Illinois, Coordinated Science Laboratory, 1961. 43

[Ogr94] J. Ogrodzki. *Circuit simulation methods and algorithms*. Electronic engineering systems series. CRC Press, 1994. 21, 58

[PC11] P. Panciatici and A. S. Chieh. Equation-based hybrid modeling of power systems for time-domain simulation. In *Proc. 2011 IEEE PES General Meeting*, pages 1–9, July 2011. 29

[PCLG$^+$11] F. Pruvost, T. Cadeau, P. Laurent-Gengoux, F. Magoulès, F.-X. Bouchez, and B. Haut. Numerical accelerations for power systems transient stability simulations. In *Proc. 17th Power System Computation Conference (PSCC)*, pages 1–8, Aug. 2011. 20

[peg12] *PEGASE project*. http://www.fp7-pegase.eu. accessed May 2012. 37

[RBFEVC12] R. Ramírez-Betancour, C. R. Fuerte-Esquivel, and T. Van Cutsem. A two-time scale simulation for dynamic analysis of power systems. *Electric Power Systems Research*, 83(1):185–195, 2012. 151

[RM09] J. Rommes and N. Martins. Exploiting structure in large-scale electrical circuit and power system problems. *Linear Algebra and its Applications*, 431(3-4):318–333, 2009. Special Issue in honor of Henk van der Vorst. 58

[RSVH79] N. B. G. Rabbat, A. L. Sangiovanni-Vincentelli, and H. Y. Hsieh. A multilevel Newton algorithm with macromodeling and latency for the analysis of large-scale nonlinear circuits in the time domain. *IEEE Trans. on Circuits and Systems*, 26(9):733–741, Sept. 1979. 97

[SBDB89] M. Stubbe, A. Bihain, J. Deuse, and J. C. Baader. STAG-a new unified software program for the study of the dynamic behaviour of electrical power systems. *IEEE Trans. Power Syst.*, 4(1):129–138, Feb. 1989. 175, 178

[SC78] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 26(1):43–49, Feb. 1978. 54, 55

[SGDPP95] J.J. Sanchez-Gasca, R. D'Aquila, W.W. Price, and J.J. Paserba. Variable time step, implicit integration for extended-term power system dynamic simulation. In *Power Industry Computer Application Conference, 1995 IEEE*, pages 183–189, May 1995. 50

[SLM$^+$00] K. Strunz, L. Linares, J. R. Marti, O. Huet, and X. Lombard. Efficient and accurate representation of asynchronous network structure changing phenomena in digital real time simulators. *IEEE Trans. Power Syst.*, 15(2):586–592, May 2000. 128

[Sto79] B. Stott. Power system dynamic response calculations. *Proceedings of the IEEE*, 67(2):219–241, Feb. 1979. 49

[Str04] K. Strunz. Flexible Numerical Integration for Efficient Representation of Switching in Real Time Electromagnetic Transients Simulation. *IEEE Trans. on Power Delivery*, 19(3):1276–1283, July 2004. 128

[Stu95] M. Stubbe. Long-Term Dynamics - Phase II. *Report of CIGRE Task Force 38.02.08*, Jan. 1995. 32

[SVM07] J. A. Sanders, F. Verhulst, and J. A. Murdock. *Averaging methods in nonlinear dynamical systems*. Applied mathematical sciences. Springer, 2007. 22, 123

[TBC85] W. F. Tinney, V. Brandwajn, and S. M. Chan. Sparse Vector Methods. *IEEE Trans. Power Apparatus and Systems*, PAS-104(2):295–301, Feb. 1985. 22, 96

[TBS99] G. Trudel, S. Bernard, and G. Scott. Hydro-Québec's defence plan against extreme contingencies. *IEEE Trans. Power Syst.*, 14(3):958–966, Aug. 1999. 35

[TGQS09] P. Tormene, T. Giorgino, S. Quaglini, and M. Stefanelli. Matching incomplete time series with dynamic time warping: an algorithm and an application to post-stroke rehabilitation. *Artificial Intelligence in Medicine*, 45(1):11–34, Jan. 2009. 54

[Tin79] W. F. Tinney. *This Week's Citation Classics*, 18:30, Apr. 1979. 5

[TS72] C. J. Tavora and O. J. M. Smith. Characterization of Equilibrium and Stability in Power Systems. *IEEE Trans. Power Apparatus and Systems*, PAS-91(3):1127–1130, May 1972. 27, 178

[TW67] W. F. Tinney and J. W. Walker. Direct solutions of sparse network equations by optimally ordered triangular factorization. *Proceedings of the IEEE*, 55(11):1801–1809, Nov. 1967. 46

[TW05] A. Toselli and O. B. Widlund. *Domain Decomposition Methods - Algorithms and Theory*. Springer series in computational mathematics. Springer, 2005. 97

[VCGL06] T. Van Cutsem, M. E. Grenier, and D. Lefebvre. Combined detailed and quasi steady-state time simulations for large-disturbance analysis. *International Journal of Electrical Power & Energy Systems*, 28(9):634–642, 2006. Selection of Papers from 15th Power Systems Computation Conference, 2005. 151

[VCJMP95] T. Van Cutsem, Y. Jacquemart, J. N. Marquet, and P. Pruvot. A comprehensive analysis of mid-term voltage stability. *IEEE Trans. Power Syst.*, 10(3):1173–1182, Aug. 1995. 122

[VCM97] T. Van Cutsem and R. Mailhot. Validation of a fast voltage stability analysis method on the Hydro-Québec system. *IEEE Trans. Power Syst.*, 12(1):282–292, Feb. 1997. 122

[VCV08] T. Van Cutsem and C. Vournas. *Voltage Stability of Electric Power Systems*. Power Electronics and Power Systems. Springer, 2008. 21, 122, 135, 187, 188

[Vin68] T. K. Vintsyuk. Speech discrimination by dynamic programming. *Cybernetics*, 4(1):52–57, 1968. Russian Kibernetika 4(1):81-88 (1968). 54, 55

[VVC06] C. Vournas and T. Van Cutsem. On-line voltage security assessment. In *Real-time stability in power systems*, pages 119–146. Springer, 2006. 175

[War49] J. B. Ward. Equivalent Circuits for Power-Flow Studies. *Transactions of the American Institute of Electrical Engineers*, 68(1):373–382, July 1949. 95

[WC11] K. Wang and M. L. Crow. Numerical simulation of Stochastic Differential Algebraic Equations for power system transient stability with random loads. In *Proc. 2011 IEEE PES General Meeting*, pages 1–8, July 2011. 22, 124

[WM83] F. F. Wu and A. Monticelli. Critical review of external network modelling for online security analysis. *International Journal of Electrical Power & Energy Systems*, 5(4):222–235, Oct. 1983. Special Issue Control Centres. 96

[YAGESS04] A. M. Youssef, T. K. Abdel-Galil, E. F. El-Saadany, and M. M. A. Salama. Disturbance classification utilizing dynamic time warping classifier. *IEEE Trans. on Power Delivery*, 19(1):272 – 278, Jan. 2004. 54

[YES81] Y. Yu and M. A. El-Sharkawi. Estimation of External Dynamic Equivalents of a Thirteen-Machine System. *IEEE Trans. Power Apparatus and Systems*, PAS-100(3):1324–1332, Mar. 1981. 96

[ZA05] Y. Zhou and V. Ajjarapu. A novel approach to trace time-domain trajectories of power systems in multiple time scales. *IEEE Trans. Power Syst.*, 20(1):149–155, Feb. 2005. 151

[Zha05] F. Zhang. *The Schur complement and its applications*. Numerical methods and algorithms. Springer Science, 2005. 21, 63

[ZWP80] J. Zaborszky, K.-W. Whang, and K. Prasad. Fast Contingency Evaluation Using Concentric Relaxation. *IEEE Trans. Power Apparatus and Systems*, PAS-99(1):28–36, Jan. 1980. 96