

Titre: Solution des systèmes de contrôle de grandes dimensions basée sur les boucles de rétroaction dans la simulation des réseaux électriques
Title:

Auteur: Chuma Francis Mugombozi
Author:

Date: 2013

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Mugombozi, C. F. (2013). Solution des systèmes de contrôle de grandes dimensions basée sur les boucles de rétroaction dans la simulation des réseaux électriques [Thèse de doctorat, École Polytechnique de Montréal]. PolyPublie.
Citation: <https://publications.polymtl.ca/1235/>

 **Document en libre accès dans PolyPublie**
Open Access document in PolyPublie

URL de PolyPublie: <https://publications.polymtl.ca/1235/>
PolyPublie URL:

Directeurs de recherche: Jean Mahseredjian
Advisors:

Programme: génie électrique
Program:

UNIVERSITÉ DE MONTRÉAL

SOLUTION DES SYSTÈMES DE CONTRÔLE DE GRANDES DIMENSIONS
BASÉE SUR LES BOUCLES DE RÉTROACTION DANS LA SIMULATION
DES RÉSEAUX ÉLECTRIQUES

MUGOMBOZI CHUMA FRANCIS

DÉPARTEMENT DE GÉNIE ÉLECTRIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIAE DOCTOR
(GÉNIE ÉLECTRIQUE)

AOÛT 2013

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

SOLUTION DES SYSTÈMES DE CONTRÔLE DE GRANDES DIMENSIONS BASÉE SUR
LES BOUCLES DE RÉTROACTION DANS LA SIMULATION DES RÉSEAUX
ÉLECTRIQUES

présentée par : MUGOMBOZI Chuma Francis

en vue de l'obtention du diplôme de : Philosophiae Doctor

a été dûment acceptée par le jury d'examen constitué de :

M. KOCAR Ilhan, Ph.D., président

M. MAHSEREDJIAN Jean, Ph.D., membre et directeur de recherche

M. KARIMI Houshang, Ph.D., membre

M. LAGACÉ Pierre-Jean, Ph.D., membre

DÉDICACE

À la mémoire, à jamais féconde, de mon père.

À ma mère et à ses petits-enfants.

À nos bienfaiteurs.

REMERCIEMENTS

Je remercie mon directeur de recherche, prof. Jean Mahseredjian. Sa rigueur m'a incité à la perfection, au dépassement et m'a poussé à approfondir le sujet. Je remercie les membres de jury. Vos critiques ont amélioré ce travail.

Je remercie mes collègues de l'IREQ. Votre passion pour la recherche a été très inspirante. Merci particulier à Omar Saad pour ses multiples apports et critiques.

Je remercie aussi mes collègues de Polytechnique, spécialement Mouhamadou et Majid.

Merci très particulier à Etienne B. de Villemeur et Francine Bélanger. Nos discussions à teneur scientifique ou sur le plan humain, ont façonné et grandement influencé mon parcours tout au long de ce doctorat.

Merci aussi à toute ma famille. Enfin, Mama, j'ai soutenu !

Finalement, gros merci à mon épouse Delphine. Cette thèse est tienne aussi ! Chérie d'amour, rappelle-toi comment ton « troubad » a été maintes fois « foutu » et que tu l'as relevé.

RÉSUMÉ

La production de l'énergie électrique, tout comme son transport et son utilisation, exige des systèmes de contrôle complexes afin de réguler la puissance et la fréquence. Ces systèmes de contrôle doivent tenir compte, entre autres, de nouvelles sources d'énergie telles que l'énergie éolienne et de nouvelles technologies de lien d'interconnexion en courant continu à haute tension. Ces systèmes de contrôle doivent être capables de contrôler l'atteinte des consignes de puissance et de fréquence tout en tenant compte des changements dans la dynamique de la source d'énergie ainsi que des défauts et événements divers affectant le comportement du réseau électrique. Ce comportement doit être étudié, en particulier, dans les scénarios où les changements génèrent des transitoires électromagnétiques. Dans ces conditions, le niveau de détails exigé par les modèles a pour conséquence d'augmenter la complexité des systèmes de contrôle.

Par ailleurs, pendant la conception du réseau, pour des tests de calibration d'équipement ou même en phase d'exploitation du réseau, peuvent être requis des outils d'aide à la décision pour les ingénieurs, tels que l'estimation de l'énergie dissipée dans les parafoudres. Ces outils utilisent les données générées par la simulation. Ils comportent des fonctions complexes, nécessitant parfois une solution numérique. Ces fonctions sont associées aux systèmes de contrôle dans les programmes de simulation. La simulation sur ordinateur évolue par ailleurs dans un contexte de développement des contrôleurs numériques, d'algorithmes de calcul parallèle ou distribué et d'outils informatiques, notamment, de la technologie à multiprocesseurs. Dans ce contexte, il est indiqué de proposer des améliorations continues au niveau du solveur des équations de contrôle modélisant ces systèmes.

Cette thèse vise à améliorer la solution des systèmes de contrôle. Plus spécifiquement, elle vise à réduire le temps de simulation du réseau avec ses systèmes de contrôle, tout en maintenant la précision requise pour l'analyse de phénomènes transitoires électromagnétiques. Les modèles dynamiques des systèmes de contrôle sont représentés sous forme d'un système d'équations $\mathbf{ax}=\mathbf{b}$. Ces équations sont parfois non-linéaires, avec des boucles de rétroaction nécessitant la formation d'une Jacobienne et des traitements spéciaux avec les outils d'analyse de graphe.

L'hypothèse principale considérée ici consiste en ceci : une reformulation qui requiert moins d'équations et, donc, une taille réduite de la Jacobienne, est possible en considérant seulement les boucles de rétroaction. Cette reformulation offre de meilleures performances que la

représentation complète classique. L'adoption d'une représentation du système de contrôle sous forme de graphe orienté a été privilégiée. Une technique de détection des cycles dans ce graphe et une méthode de suppression de ces derniers ont été établies. Pour supprimer tout cycle dans le graphe, une condition s'applique, soit celle d'avoir toutes les valeurs propres nulles pour la matrice d'adjacence. La transformation du graphe de contrôle sans cycle a permis de formuler une représentation des équations aux éléments (points) de rétroaction (PR). Ceci a rendu possible la décomposition avec interconnexion aux points de rétroaction (DIPR) : il s'agit de la contribution principale de cette thèse, la nouvelle approche qu'elle propose.

Des méthodes de résolution des systèmes d'équations (non-linéaires) de contrôle applicables à un cadre plus étendu, ont été déployées pour la nouvelle approche DIPR. Cinq variantes de la nouvelle approche ont été illustrées. Il s'agit de la méthode de Newton de base (1), d'une méthode plus robuste que cette dernière, dite de Dogleg (2) ; d'une méthode à itération point-fixe (3). Celle-ci a été possible car la DIPR introduite permet une formulation des chemins de rétroaction tel qu'ils s'apparentent à un problème à point-fixe avec, comme variables, les seuls points de rétroaction. Il s'agit aussi d'une contribution originale de cette thèse. Une autre des méthodes repose sur une combinaison de la méthode à itérations à point-fixe et la méthode Dogleg (4) : la première est d'abord utilisée et si elle ne converge après 3 à 5 itérations, la deuxième est alors utilisée. Finalement, la dernière variante est une solution non-itérative (5), mais elle est limitée à certains cas car on y présume l'atteinte d'une solution après une seule itération.

Pour analyser la performance de la nouvelle approche, les variantes mentionnées ci-haut ont été appliquées sur des systèmes de contrôle représentatifs de cas couramment rencontrés. À cette fin, trois cas ont été étudiés. Le premier cas est une boucle à verrouillage de phase (« Phase-locked loop » (PLL)) ; le deuxième cas est celui d'une machine asynchrone modélisée par près d'une centaine d'équations ; finalement, le troisième est un réseau de très grande taille où 12 centrales éoliennes ont été intégrées au réseau test IEEE-39 barres. Dans tous les cas traités, les résultats avec la DIPR sont similaires à ceux obtenus par la méthode de référence considérée, implémentée dans le logiciel « Electromagnetic Transients Program-restructured version » (EMTP-RV). Les gains en temps de calcul, jusqu'à deux dans le meilleur des cas, ont été enregistrés dans la plupart des tests avec la méthode DIPR proposée. Des limitations ont été

analysées, à souligner que l'approche DIPR offre plusieurs variantes alternatives aux méthodes classiques de représentation complète de systèmes.

Pour une analyse approfondie de la nouvelle approche et pour proposer des pistes de recherche futures, la méthode DIPR a été comparée à d'autres méthodes de solution de système de contrôle. Deux comparaisons ont été menées. Une première a montré que, sur le pur plan des représentations des systèmes d'équations simultanées, la méthode de substitution de variables peut transformer une représentation complète classique en une représentation DIPR, à condition de ne pas substituer les variables PR et de suivre un ordre de substitution respectant la séquence topologique d'évaluation des blocs. Une deuxième comparaison est celle entre l'approche DIPR et la représentation par variables d'état. Le logiciel MATLAB/Simulink, lequel utilise une représentation par variables d'état, a été comparé à l'approche DIPR. Il ressort de l'analyse que l'étape d'évaluation des sorties de blocs dans l'approche avec représentation par variables d'état s'apparente à la solution DIPR pour la solution de la boucle algébrique, les contraintes algébriques étant comparables aux boucles de rétroaction. Une différence peut cependant surgir quand le nombre de contraintes algébriques du cas par variables d'état n'est pas le nombre optimal de coupures pour la méthode DIPR : pour le PLL, par exemple, MATLAB/Simulink a généré 3 contraintes alors que la DIPR en a généré 2. La méthode DIPR pourrait offrir certains avantages dans ce cas.

L'initialisation est un autre axe d'analyse pouvant susciter des travaux futurs. La manière dont la transformation DIPR modifie les propriétés des itérations de Newton a été analysée. Le théorème de Newton-Kantorovich, utilisant les limites sur les normes de la Jacobienne, a été appliqué. Les expressions des normes de la Jacobienne, ont été établies pour des cas génériques, sous forme analytique, à l'aide de graphe de Coates. Il ressort de l'analyse d'un cas simple que, pour les mêmes conditions initiales, les comportements du théorème de Newton-Kantorovich diffèrent dans les deux approches. Ces différences peuvent par ailleurs être plus prononcées dans le cas non-linéaire. Dans les applications illustrées l'initialisation a été faite par la méthode de Newton de base. Des travaux futurs seraient utiles à l'approfondissement de cet aspect et, éventuellement, permettraient de déduire des approches nouvelles d'initialisation des équations de contrôle.

Nonobstant les limitations évoquées et sans oublier les pistes d'amélioration que pourront mettre à jour les travaux futurs mentionnées plus haut (incluant des potentielles fonctionnalités à introduire), les retombées en gain de temps de simulation qu'a apportées cette thèse sont à noter.

ABSTRACT

The generation of electrical energy, as well as its transportation and consumption, requires complex control systems for the regulation of power and frequency. These control systems must take into account, among others, new energy sources such as wind energy and new technologies for interconnection by high voltage DC link. These control systems must be able to monitor and achieve such regulation in accordance with the dynamics of the energy source, faults and other events which may induce transients phenomena into the power network. Such transients conditions have to be analyzed using the most accurate and detailed hence, complex models of control system.

In addition, in the feasibility study phase, the calibration or the setup of equipment as well as in the operation of the power network, one may require decision aid tools for engineers. This includes, for instance, knowledge of energy dissipated into the arresters in transient analysis. These tools use simulation programs data as inputs and may require that complex functions be solved with numerical methods. These functions are part of control system in computer simulator. Moreover, the simulation evolves in a broader context of the development of digital controller, distributed and parallel high performance computing and rapid evolutions in computer (multiprocessor) technology. In such context, a continuing improvement of the control equations solver is welcomed.

This thesis aims to contribute to enhance the accuracy and efficiency in computation of control system equations in electromagnetic transients type simulation methods.

Control systems are modelled using $\mathbf{ax}=\mathbf{b}$ simultaneous system of equations. These equations are sometimes non-linear with feedback loops and thus require iterative Newton methods, including the formation of a Jacobian matrix and ordering as well as processing by graph theory tools. The proposed approach is based on the formulation of a reduced rank Jacobian matrix. The dimension is reduced up to the count of feedback loops. With this new approach, gains in computation speed are expected without compromising its accuracy when compared to classical full rank Jacobian matrix representation.

A directed graph representation is adopted and a proper approach for detecting and removing cycles within the graph is introduced. A condition of all zero eigenvalues of adjacency

matrix of the graph is adopted. The transformation of the graph of controls with no cycle permits a formulation of control equations for only feedback points. This yields a general feedback interconnection (GFBI) representation of control, which is the main contribution of this thesis.

Methods for solving (non-linear) equations of control systems were deployed into the new GFBI approach. Five variants of the new approach were illustrated including firstly, a basic Newton method (1), a more robust one, the Dogleg method (2) and a fixed-point iterations method (3). It was possible because the GFBI approach allows a formulation of fixed-point problem on output of feedback point blocks through feedback paths. Another method is a hybrid approach with fixed-point and dogleg (4). The switching condition between the two is pre-set iteration counts. Finally, a non-iterative variant (5) is proposed, but is limited to few classes of control given the assumption that single iteration yields an acceptable solution.

The presented approach is implemented in Electromagnetic Transient program EMTP-RV and tested on practical systems of various types and levels of complexity: the PLL, an asynchronous machine with 87 blocks reduced to 23 feedback equations by GFBI, and 12 wind power plants integrated to the IEEE-39 buses system. In all cases, similar results were obtained in comparison to the method of EMTP based on block linearization. Gains in computation time, up to two, in best case, are obtained with the proposed approach. However, some limitations are also noted and discussed.

Further analysis, which opens up avenues for future research includes comparison of the proposed approach against existing ones. With respect to the sole representation, it is shown that the proposed approach is equivalent to full classic representation of system of equations through a proper substitution process which complies with topological sequence and by skipping feedback variable identified by GFBI. Moreover, a second comparison with state space based approach, such as that in MATLAB/Simulink, shows that output evaluation step in state-space approach with algebraic constraints is similar to the GFBI. The algebraic constraints are similar to feedback variables. A difference may arise, however, when the number of algebraic constraints is not the optimal number of cuts for the GFBI method: for the PLL, for example, MATLAB/Simulink generated 3 constraints while the GFBI generated only 2. The GFBI method may offer some advantages in this case.

A last analysis axis prompted further work in initialization. It is shown that GFBI method may modifies the convergence properties of iterations of the Newton method. The Newton-Kantorovich theorem, using bounds on the norms of the Jacobian, has been applied to the proposed GFBI and classic full representation of control equations. The expressions of the Jacobian norms have been established for generic cases using Coates graph. It appears from the analysis of a simple case, for the same initial conditions, the behaviour of the Newton-Kantorovich theorem differs in both cases. These differences may also be more pronounced in the non-linear case. Further work would be useful to investigate this aspect and, eventually, pave the way to new initialization approaches.

Despite these limitations, not to mention areas for improvement in further work, one notes the contribution of this thesis to improve the gain of time on simulation for the solution of control systems.

TABLE DES MATIÈRES

DÉDICACE.....	III
REMERCIEMENTS	IV
RÉSUMÉ.....	V
ABSTRACT	IX
TABLE DES MATIÈRES	XII
LISTE DES TABLEAUX.....	XVI
LISTE DES FIGURES	XVIII
LISTE DES SIGLES ET ABRÉVIATIONS	XXV
LISTE DES ANNEXES	XXVII
INTRODUCTION.....	1
1. Problème étudié et buts poursuivis.....	1
2. Contributions apportées par cette thèse.....	3
3. Approche et évolution de la réalisation : méthodologie.....	3
4. Revue historique et bibliographique	5
5. Organisation de la thèse	10
CHAPITRE 1 GÉNÉRALITÉS SUR LA MODÉLISATION MATHÉMATIQUE DES SYSTÈMES DE CONTRÔLE.....	13
1.1 Introduction	13
1.2 Le système de contrôle et le réseau électrique	14
1.3 Types de blocs de contrôle et regroupement	18
1.3.1 Types de blocs.....	18
1.3.1.1 Blocs avec ou sans entrée et avec ou sans sortie.....	18
1.3.1.2 Blocs statiques.....	19

1.3.1.3	Blocs dynamiques avec termes d'histoire	20
1.3.1.4	Blocs avec coefficients constants ou variables	20
1.3.1.5	Blocs non-linéaires	20
1.3.1.6	Blocs « boîtes-noires »	21
1.3.1.7	Blocs discrets et blocs avec discontinuité	21
1.3.1.8	Tableau récapitulatif des types de blocs de contrôle	21
1.3.2	Regroupement topologique des blocs de contrôle	23
1.3.2.1	Groupe à action directe (« Feedforward »)	23
1.3.2.2	Groupe à rétroaction (« Feedback »)	23
1.4	Dynamique physique et modèles mathématiques	24
1.5	Quelques fonctionnalités de systèmes de contrôle	29
1.5.1	Modélisation des systèmes asservis	29
1.5.2	Fonctions spécifiques par usager pour les mesures et traitements divers	30
1.5.3	Modélisation d'un processus ou phénomène d'arc de disjoncteur	30
1.6	Conclusion	32
CHAPITRE 2	GRAPHES ET TRANSFORMATION AUX POINTS DE RÉTROACTION DES BOUCLES	34
2.1	Introduction	34
2.2	Propriétés du graphe : matrice d'adjacence et d'accessibilité	39
2.3	Exemple d'analyse topologique générique pour le cas de PLL	41
2.4	Suppression des cycles dans le graphe : valeurs propres de la matrice d'adjacence	44
2.4.1	But de la suppression des cycles	44
2.4.2	Analyse des boucles et cycles avec la matrice d'adjacence	45
2.5	Représentation des équations aux éléments de rétroaction des boucles	55
2.5.1	Aperçu de l'approche classique sur un cas simple de la Figure 2-7	55

2.5.2	Aperçu de la nouvelle approche sur le même cas simple de la Figure 2-7	57
2.6	Décomposition avec interconnexion aux points de rétroaction (DIPR).....	61
2.7	Conclusion.....	65
CHAPITRE 3 SOLUTION DES SYSTÈMES DE CONTRÔLE		66
3.1	Calculs préliminaires communs aux variantes des méthodes	66
3.1.1	Évaluation des fonctions objectives Φ du processus itératif.....	67
3.1.2	Évaluation de la matrice Jacobienne du processus itératif	68
3.2	Les variantes de la méthode de solution DIPR	71
3.2.1	Solution complète itérative.....	71
3.2.1.1	Méthode de Newton de base	71
3.2.1.2	Méthode Dogleg de la Routine HYBRJ1 de MINPACK.....	73
3.2.2	Solution par la méthode point-fixe.....	76
3.2.2.1	Formulation de la méthode point-fixe.....	76
3.2.2.2	Avantages et limitations de la méthode point-fixe.....	77
3.2.3	Solution non-itérative.....	81
3.2.3.1	Formulation de la solution non-itérative	81
3.2.3.2	Limitations de la méthode non-itérative.....	81
3.3	Conclusion.....	82
CHAPITRE 4 APPLICATIONS DE LA MÉTHODE PROPOSÉE		83
4.1	Système de contrôle avec boucles et nonlinéarités : cas du PLL.....	83
4.2	Machine asynchrone modélisée par équations algébriques.....	88
4.3	Système de grande taille : éoliennes de [1] intégrées au réseau IEEE 39-barres.....	90
4.4	Discussion de la performance de la nouvelle méthode	96
4.4.1	Précision et test de convergence de la solution	96

4.4.2	Gains en temps de calcul des différentes variantes	100
4.5	Conclusion.....	107
CHAPITRE 5 COMPARAISON AVEC LA REPRÉSENTATION PAR VARIABLES D'ÉTAT		109
5.1	Conformité des représentations complète et réduite par DIPR	109
5.2	Comparaison avec la représentation par variables d'état	120
5.2.1	Variables d'état, chacune dépendant d'un seul point de rétroaction	120
5.2.2	Variables d'état dépendant de plus d'un point de rétroaction	128
5.3	Conclusion.....	139
CHAPITRE 6 DÉFIS D'INITIALISATION ET TRAVAUX FUTURS		141
6.1	Motivation	141
6.2	Expressions de J et Φ dans les représentations complète et réduite par DIPR	144
6.2.1	Cas d'un seul point de rétroaction.....	144
6.2.2	Cas avec plusieurs points de rétroaction	168
6.3	Analyse de l'existence et l'unicité de la solution pour le processus de Newton.....	178
6.3.1	Théorème de Kantorovich.....	178
6.3.2	Application du théorème de Kantorovich : approches classique et DIPR	181
6.3.2.1	Formulation du théorème de Kantorovich pour le système de contrôle	181
6.3.2.2	Illustration du théorème de Kantorovich sur un système de contrôle simple ..	183
6.4	Conclusion.....	191
CONCLUSION		193
BIBLIOGRAPHIE		197
ANNEXES		201

LISTE DES TABLEAUX

Tableau 1.1: Récapitulatif des types de blocs	22
Tableau 2.1: Positions dans la LT pour les entrées des blocs candidats pour la coupure de la Figure 2-9	49
Tableau 3.1: Illustration de la précision de la Jacobienne numérique : erreurs relatives maximales sur l'évaluation de la Jacobienne numérique en comparaison avec la Jacobienne analytique pour les fonctions de la Figure 3-2.....	70
Tableau 4.1: Résultats de la nouvelle méthodologie appliquée au cas du PLL	84
Tableau 4.2: Caractéristiques des systèmes simulés : rétroactions, groupes et blocs non-linéaires	88
Tableau 4.3: Résultats de la nouvelle méthodologie sur une machine asynchrone entièrement modélisée par les équations algébriques	88
Tableau 4.4: Résultats de la nouvelle méthodologie sur le réseau IEEE-39 barres avec des centrales éoliennes.....	90
Tableau 4.5: Analyse d'un des groupes dans l'éolienne	102
Tableau 4.6: Nombres d'évaluation des sorties des blocs durant un processus-type d'itérations	106
Tableau 6.1: Résultats de l'inverse de la Jacobienne dans le système réduit par la DIPR : Jacobienne, déterminant et inverses des Jacobiennes.	175
Tableau 6.2: Résultats pour le déterminant de l'inverse de la Jacobienne pour le cas de quatre points de rétroaction définissant le système réduit par la DIPR $N_r = M = 4$	176
Tableau 6.3: Résultats de l'inverse de la Jacobienne dans le système réduit par la DIPR : Jacobienne, déterminant et inverses des Jacobiennes $N_r = 4$	177
Tableau 6.4: Comparaison de la Jacobienne et de son inverse dans la représentation complète et celle réduite par la DIPR pour analyse du théorème de Kantorovich.	184
Tableau 6.5: Comparaison de $\ \mathbf{J}^{-1}\Phi\ $ de la représentation complète et celle réduite par la DIPR pour l'analyse du théorème de Kantorovich.	185

Tableau 6.6: Les conditions du théorème de Kantorovich pour le cas simple avec blocs linéaires	186
Tableau 6.7: Application numérique sur les conditions du théorème de Kantorovich pour le cas simple avec blocs linéaires dont les gains sont : $[C_v] = [0.1, 0.2, 0.15, C_{Max}]$	187
Tableau 6.8: Les conditions du théorème de Kantorovich pour le cas simple avec un bloc non-linéaire	191

LISTE DES FIGURES

Figure 1-1: Schéma générique d'interaction réseau/système de contrôle	15
Figure 1-2: Réseau électrique pour étude des surtensions temporaires (tiré de [34])	16
Figure 1-3: Simulation d'un défaut proche en ligne avec un modèle d'arc de disjoncteur [34]	16
Figure 1-4: Sous-circuit du disjoncteur D dans le schéma de la Figure 1-3. Sur la phase a est modélisé l'arc selon les équations de Cassie-Mayr	16
Figure 1-5: Admittance contrôlée intervenant dans les équations du réseau présentées en (1.1) ..	17
Figure 1-6: Trois formes d'interactions entre le réseau et le système de contrôle via les variables dans (1.1)	17
Figure 1-7: Caractérisation des blocs selon la topologie : bloc sans entrée et bloc sans sortie	19
Figure 1-8: Bloc statique : fonction interne indépendante du temps t	19
Figure 1-9: Blocs somme et gain	20
Figure 1-10: Transformation du système continu pour obtenir le système discret	27
Figure 1-11: Portion du système de contrôle d'excitatrice de type EXST1 (IEEE)	29
Figure 1-12: Énergies évaluées dans le parafoudre pour les trois phases (a, b, c) dans la Figure 1-2	30
Figure 1-13: Diagramme du modèle d'arc Cassie-Mayr dans les exemples fournis dans [3]	32
Figure 2-1: Système de contrôle d'une boucle à verrouillage de la phase "Phase-locked loop" (PLL) tiré de [1]	36
Figure 2-2: Pictogramme du graphe orienté du PLL de la Figure 2-1	36
Figure 2-3: Schéma du PLL sous forme générique	38
Figure 2-4: Représentation de la matrice d'adjacence pour le système de contrôle du PLL	42
Figure 2-5: Représentation de la matrice d'accessibilité du PLL	43
Figure 2-6: Représentation de la matrice de connectivité pour le PLL	44
Figure 2-7: Schéma simple d'un diagramme bloc avec une rétroaction non-linéaire	45

Figure 2-8: Introduction des coupures dans le schéma simple d'un diagramme bloc avec une rétroaction non-linéaire : (a) une seule coupure (b) plusieurs coupures, une après chaque sortie de bloc.	46
Figure 2-9: Matrice d'adjacence de PLL avec indication des blocs à deux entrées dont les blocs-cibles sont ceux de EC_1	48
Figure 2-10: Procédure proposée pour établir l'ensemble optimal des coupures (oEC) garantissant l'absence de valeur propre non-nulle dans la matrice d'adjacence	50
Figure 2-11: Découpage des boucles de rétroaction pour le PLL en introduisant d'autres nœuds à la fonction 3 : 3, 3-1,3-2,3-3 et à la fonction 4 : 4,4-1,4-2, 4-3.....	51
Figure 2-12: Ré-listage des fonctions pour une nouvelle LT.....	51
Figure 2-13: Représentation de la matrice d'accessibilité R pour le diagramme de PLL modifié : points de rétroaction ouverts	53
Figure 2-14: Connexion par rétroaction : point de jonction pour la Figure 2-7,.....	58
Figure 2-15: Représentation classique du « general feedback interconnection » comparable à la DIPR proposée	61
Figure 2-16: Décomposition aux points de rétroaction : M coupes	62
Figure 3-1: Exemple simple de rétroaction avec fonctions non-linéaires (boîte-noire) à perturber	69
Figure 3-2: Fonctions non-linéaires de la boucle de rétroaction de la Figure 3-1	70
Figure 3-3: Variations des sorties des blocs PR pour le PLL dans une simulation avec un pas de temps de 1 μ s.....	79
Figure 3-4: Valeurs propres et norme spectrale de la Jacobienne des fonctions composées de chemin de rétroaction correspondantes aux solutions de la Figure 3-3 pour le PLL	80
Figure 4-1: Superpositions montrant la concordance parfaite des signaux d'amplitude et de sortie du PLL pour toutes les variantes du Tableau 4.1 et la référence de la solution itérative de [4]	85

Figure 4-2: Comptage d'itérations à chaque instant t pour les variantes itératives de la solution appliquées au PLL.....	86
Figure 4-3: Distribution cumulative des itérations à chaque pas de temps pour les variantes itératives de la solution appliquées au PLL.....	87
Figure 4-4: Agrandissement de quelques résultats de simulation de la machine asynchrone modélisée par les équations algébriques résolues par les méthodes du Tableau 4.3 et la méthode itérative de [4] :« CURRENT [EMT1]».....	89
Figure 4-5: Distributions cumulées des itérations pour le groupe de taille 26x26 (no 1102) dans la centrale éolienne de la barre 33, à proximité d'un défaut triphasé initié à 4.5 secondes sur le réseau IEEE 39-barres avec 12 centrales éoliennes	91
Figure 4-6: Exemple de résultats de simulation des signaux Vdc, P et Q sur la barre no 32 du réseau IEEE 39-barres avec 12 centrales éoliennes. [EMTP1] : solution de [4]; [EMTP2] : Newton non-itératif, [EMTP3] : DOGLEG itératif ; [EMTP4] : Itérations à point-fixe et [EMTP5] : Combinaison de DOGLEG et itérations à point-fixe.....	92
Figure 4-7: Sommes cumulées des itérations sur le groupe de taille 26x26 (no 1102) dans la centrale éolienne de la barre 33 à proximité d'un défaut triphasé initié à 4.5 secondes sur le réseau IEEE 39-barres avec 12 centrales éoliennes	93
Figure 4-8: Intégration de 12 centrales éoliennes sur le réseau IEEE 39-barres modélisé dans EMTP-RV	95
Figure 4-9: Superposition pour le PLL des itérations de la méthode point-fixe et des rayons spectraux de la Jacobienne des fonctions composées de chemins de rétroaction \mathbf{G}	101
Figure 4-10: Emplacement du groupe no 4 dans une des parties du système de contrôle de l'éolienne (voir le sous masque PI, en bas à droite).....	103
Figure 4-11: Le groupe no 4 (sous masque PI dans la Figure 4-10) dans le modèle d'éolienne dont la séquence des blocs est dans le Tableau 4.5	104
Figure 4-12: Variation de la norme spectrale en fonction du temps de simulation pour le groupe no 4 de l'éolienne installée à la barre 37 du système IEEE-39 barres	105

Figure 5-1: Décomposition générale aux points de rétroaction : cas des chemins de rétroaction indépendants.....	114
Figure 5-2: Décomposition générale aux points de rétroaction avant et après regroupement des chemins de rétroaction couplés.	116
Figure 5-3: Cas simple linéaire pour 3 variables d'état avec 3 boucles de rétroaction : similarité entre DIPR et représentation par matrices ABCD dans MATLAB/Simulink	121
Figure 5-4: Matrices ABCD pour le cas simple linéaire avec 3 variables d'état et 3 boucles de rétroaction. Linéarisation à 2 secondes avec un pas de 250 μ s.	122
Figure 5-5: Matrice Jacobienne dFdx et itérations pour le cas simple linéaire avec 3 variables d'état et 3 boucles de rétroaction. Linéarisation à 2 secondes avec un pas de 250 μ s.....	124
Figure 5-6: Matrice Jacobienne dFdx et itérations pour le cas simple linéaire avec 3 variables d'état et 3 boucles de rétroaction. Linéarisation à 2.00025 secondes avec un pas de 250 μ s	124
Figure 5-7: Listes d'exécution des méthodes dans MATLAB/Simulink pour le cas simple linéaire avec 3 variables d'état et 3 boucles de rétroaction (la mise à jour des états et le calcul des variables algébriques présentés en rouge).....	125
Figure 5-8: Représentation du PLL dans MATLAB/Simulink par variables d'état	128
Figure 5-9: Représentation par variable d'état pour le PLL linéarisé autour de 125ms pour un pas de temps de 5 μ s.	129
Figure 5-10: Boucle algébrique détectée par MATLAB/Simulink pour le PLL et l'ordre d'exécution des fonctions des blocs	130
Figure 5-11: Processus de la solution de la boucle algébrique dans MATLAB/Simulink pour le PLL autour de 0.125ms	131
Figure 5-12: Valeurs des états mises à jour : processus de la solution de la boucle algébrique dans MATLAB/Simulink pour le PLL autour de 0.125ms	132
Figure 5-13: Regroupement des blocs pour la fonction composée G_2 du PLL dans MATLAB/Simulink	132

Figure 5-14: Regroupement des blocs pour la fonction composée g_1 - du PLL dans MATLAB/Simulink	133
Figure 5-15: Fonction exprimant le regroupement des blocs pour la fonction composée G^2 du PLL dans MATLAB/Simulink.....	133
Figure 5-16: Fonction exprimant le regroupement des blocs pour la fonction composée g_1 - du PLL dans MATLAB/Simulink.....	134
Figure 5-17: Schéma Simulink du PLL modifié avec des fonctions exprimant le regroupement des blocs pour la fonction composée g_1 - et G_2 du PLL dans MATLAB/Simulink	134
Figure 5-18: Schéma Simulink du PLL modifié avec des fonctions exprimant le regroupement des blocs pour la fonction composée g_1 - et G_2 du PLL dans MATLAB/Simulink	135
Figure 5-19: Linéarisation du schéma MATLAB/Simulink du PLL modifié avec des fonctions exprimant le regroupement des blocs pour la fonction composée g_1 - et G_2 du PLL	136
Figure 5-20: Processus de solution de type DIPR appliqué au schéma Simulink avec regroupement des blocs pour la fonction composée g_1 - et G_2 du PLL dans MATLAB/Simulink	137
Figure 5-21: Jacobienne et processus de la solution similaire à la DIPR appliquée à MATLAB/Simulink pour la fonction composée g_1 - et G_2 du PLL dans MATLAB/Simulink	138
Figure 5-22: Solution avec les variables d'état appliquée à MATLAB/Simulink similaire à la DIPR.....	138
Figure 5-23: Les signaux des sorties o_1 et o_2 et d'entrée du PLL monté dans MATLAB/Simulink (identiques pour les solutions DIPR montées dans MATLAB/Simulink)	139
Figure 6-1: Système de contrôle simple pour analyse de l'existence et d'unicité de la solution pour système complet et réduit.....	144
Figure 6-2: Graphe de Coates pour les équations du système de contrôle en (6.10) réécrit en (6.13)	147
Figure 6-3: « Sous-graphes 1-facteur » du Graphe de Coates en la Figure 6-2 des équations du système de contrôle en (6.10).....	147

Figure 6-4: Graphe modifié pour la solution $\mathbf{J}\delta\mathbf{x} = \Phi$ pour un cas simple.....	150
Figure 6-5: Première des connexions dans le sous-graphe 1-facteur $H_{(N+1)i}$ dans (6.21) entre le nœud $N+1$ et 1 basé sur le graphe modifié pour la solution $\mathbf{J}\delta\mathbf{x} = \Phi$ montré en Figure 6-4 (les branches en pointillé sont inexistantes; les autres connexions sont dans la Figure 6-6)	151
Figure 6-6: Autres connexions dans le sous-graphe 1-facteur similaires à celles de la Figure 6-5	152
Figure 6-7: Système de contrôle simple transformé selon la DIPR pour l'analyse de l'existence et d'unicité de la solution pour système complet et réduit.....	158
Figure 6-8: Graphe de Coates du système de contrôle simple de la Figure 6-7 discrétisé selon (6.39)	158
Figure 6-9: Graphe de Coates des nœuds supprimés $\mathbf{gr}(\mathbf{N}-\mathbf{N}_r)$ du graphe $\mathbf{gr}(\mathbf{N})$ de la Figure 6-4: Graphe modifié pour la solution ... pour un cas simple vers le graphe $\mathbf{gr}(\mathbf{N}_r)$ de la Figure 6-8.....	159
Figure 6-10: Section du graphe $\mathbf{grD}((\mathbf{N}-\mathbf{N}_r) + \{5,5\})$: ajout du nœud 5 sur le graphe de Coates des nœuds supprimés $\mathbf{gr}(\mathbf{N}-\mathbf{N}_r)$	162
Figure 6-11: Sous-graphes 1-facteur $\mathbf{grDfa}((\mathbf{N}-\mathbf{N}_r) + \{i, j=i\} - \{x_5'\})$: suppression du nœud source 5 dans la section du graphe $\mathbf{grD}((\mathbf{N}-\mathbf{N}_r) + \{5,5\})$	162
Figure 6-12: Connexions dans les sous-graphes 1-facteur $\mathbf{grDfaX}(((\mathbf{N}-\mathbf{N}_r) + \{5,5\})(x_5', x_5''))$ entre les nœuds (x_5', x_5'') dans la section du graphe $\mathbf{grD}((\mathbf{N}-\mathbf{N}_r) + \{5,5\})$	163
Figure 6-13: Graphe de Coates $\mathbf{gr}(\mathbf{N}_r + \{\Phi\})$: Remplacement du gain équivalent après réduction du graphe selon la DIPR et ajout de la variable de la fonction objective $(\overset{R}{\mathbf{J}}\delta\mathbf{x} = \overset{R}{\Phi})$ comme en la Figure 6-4	164

Figure 6-14: Connexion 1-facteur $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5))$ de Φ vers δx_5 dans le graphe de Coates $\mathbf{gr}(\mathbf{N}_r + \{\Phi\})$	166
Figure 6-15: Graphe de Coates $\mathbf{gr}(\mathbf{N}_r)$ du système réduit par la DIPR, équivalent au sous-graphe 1-facteur $\mathbf{grfa}(\mathbf{N}_r)$	166
Figure 6-16: Graphe de Coates pour le cas de réduction DIPR avec plusieurs points de rétroaction $\mathbf{gr}(\mathbf{N}_r)$ fortement connectés.	169
Figure 6-17: Graphe de Coates pour le cas de réduction DIPR avec deux points de rétroaction $\mathbf{gr}(\mathbf{N}_r) = \mathbf{gr}(\{1, 2\})$	169
Figure 6-18: Deux sous-graphes 1-facteur $\mathbf{grfa}(\mathbf{N}_r) = \mathbf{grfa}(\{1, 2\})$ pour un cas générique de deux points de rétroaction $\mathbf{gr}(\mathbf{N}_r)$	171
Figure 6-19: Différents sous-graphes pour la connexion 1-facteur : a) Graphe augmenté $\mathbf{gr}(\mathbf{N}_r + \{\Phi\})$; b) et c) Sous-graphes connexion 1-facteur de l'ensemble $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_1))$; d et e) Sous-graphes connexion 1-facteur de l'ensemble $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_2))$	172
Figure 6-20: Graphes de Coates d'un système réduit à quatre points de rétroaction par la DIPR	176
Figure 6-21: Système simple dans deux modes représentation sur lequel sera illustré le théorème de Kantorovich	183
Figure 6-22: Résultat du théorème de Kantorovich sur un système simple avec $CI=0$ pour toutes les variables	188
Figure 6-23: Résultat du théorème de Kantorovich sur un système simple avec $CI=2$ pour toutes les variables	189

LISTE DES SIGLES ET ABRÉVIATIONS

\emptyset CFC	Ensemble vide pour les composantes fortement connectées
Υ	Fonction (ou opérateur) liant la tension et le courant
$\lambda(.)$	Norme spectrale
\otimes	Opération binaire ET
\oplus	Opération binaire OU
$[\]^T$	Transposition d'un vecteur ou d'une matrice $[\]$
$\tilde{\mathbf{P}}_{ij}$ ou $f_i \oslash f_j$	Collection d'indices du chemin de f_i vers f_j
$\tilde{\mathbf{R}}, \mathbf{R}$	Matrice d'accessibilité (« Reachability matrix »)
$\tilde{\mathbf{A}}, \mathbf{A}$	Matrice d'adjacence
<i>BFR</i>	Bloc de la fonction de rétroaction
CCHT	Courant continu à haute tension
CFC	Composante fortement connectée
DFS	Depth-First Search
DIPR	Décomposition avec interconnexion aux points de rétroaction
DLL	Dynamic Link Library
EC	Ensemble des coupures ne donnant aucun cycle dans le graphe
EMT	Transitoires électromagnétiques
EMTP-RV	Electromagnetic Transient Program-Restructured Version
f	Fonction décrivant le comportement d'un bloc de contrôle
\mathbf{I}	Matrice identité
i, \mathbf{i}	Courant électrique : scalaire et vecteur

LT	Liste topologique
M	Nombre de points de rétroaction
MEC	Maximal de EC
mEC	Minimal de EC
N	Nombre total de blocs dans le système
oEC	Optimal de EC
p.u	per unit
PLL	Phase-locked loop : Boucle à verrouillage de phase
PR	Point (bloc) de rétroaction
Q	Matrice de connectivité
TACS	Transient Analysis of Control Systems
TLM	Modèle de ligne de Transmission
TTR	Tension temporaire de rétablissement
u	Entrée d'un bloc de contrôle
v, \mathbf{v}	Tension électrique : scalaire et vecteur
y	Sortie d'un bloc de contrôle
Δu	Variation de l'entrée d'un bloc de contrôle
εu	Variation infinitésimale de l'entrée d'un bloc de contrôle
$\sigma (.)$	Substitution sur les variables (.)

LISTE DES ANNEXES

ANNEXE 1 – APERÇUS D’ALGORITHMES UTILISÉS	201
ANNEXE 1-1 Algorithme de « Depth-First Search ».....	201
ANNEXE 1-2 Algorithme pour le calcul de la matrice de “reachability” (avec Matlab).....	201
ANNEXE 1-3 Algorithme pour le calcul de “Strongly Connected Components” (SCC) (Matlab).....	202
ANNEXE 2 – SCRIPTS POUR L’ÉTUDE DES PROPRIÉTÉS CONTRACTANTES POUR LA MÉTHODE POINT-FIXE DU PLL	203
ANNEXE 3 – ANALYSE DE L’ÉQUIVALENCE ENTRE LA MÉTHODE DIPR ET MATLAB/SIMULINK AVEC LES ÉTATS DISCRETS	208
ANNEXE 4 – MACHINE ASYNCHRONE ENTIÈREMENT MODÉLISÉE PAR ÉQUATIONS ALGÈBRIQUES DANS EMTP-RV.....	211
ANNEXE 5 – SCRIPT POUR CALCULER LE PERMANENT D’UNE MATRICE CARRÉE	212

INTRODUCTION

1. Problème étudié et buts poursuivis

La production de l'énergie électrique, tout comme son transport et son utilisation, exigent des systèmes de contrôle complexes pour réguler la puissance et la fréquence. La régulation doit satisfaire aux exigences en énergie réactive nécessaire à l'équilibre en tension. Ces systèmes de contrôle doivent tenir compte des nouvelles sources d'énergie telle l'énergie éolienne et, aussi, des nouvelles technologies de lien en courant continu à haute tension (CCHT). Dans les centres de production ou dans les stations d'interconnexion, ces systèmes de contrôle doivent être capables de contrôler l'atteinte des consignes de puissance et de fréquence tenant compte des propriétés dynamiques de la source d'énergie lors de la diminution de la vitesse du vent, par exemple, et du comportement du réseau électrique. Ce dernier peut être soumis aux variations de la puissance : coupure soudaine d'un grand utilisateur et défauts divers, court-circuit, foudre, etc. La simulation sur ordinateur est utile pour étudier ces scénarios et, ainsi, pour mieux planifier les améliorations du réseau, l'ajout des éoliennes sur un réseau existant et les interconnexions entre réseaux de différentes zones, ou l'analyse et la réparation des cas de défaillances et de pannes rencontrées. Une part importante des scénarios à étudier consiste en changements soudains et sévères du niveau de la production ou de la demande, ou encore, de la foudre frappant le réseau. Ce dernier, par exemple, génère des transitoires électromagnétiques (EMT) pouvant occasionner de bris d'équipement entamant la sûreté des installations électriques, ou pire, une perte de la stabilité du réseau. L'étude du réseau doit être menée en simulant les conditions générant ces phénomènes transitoires électromagnétiques. De plus, dans le cas de ces phénomènes transitoires, le niveau de détail exigé pour les modèles des systèmes de contrôle est parfois tel qu'une grande complexité des systèmes est à considérer [1] [2].

Ces modèles sont représentés sous forme d'un système d'équations $\mathbf{ax}=\mathbf{b}$ (apparenté à la représentation nodale du réseau électrique de puissance) ou sous la forme par variables d'état. Dans les deux cas, pour un pas fixe, le modèle discret peut alors être représenté par les équations algébriques. Les méthodes de solution à analyser dans cette thèse supposent que les équations algébriques des modèles du système de contrôle sont formées. Les programmes d'analyse de type EMT utilisent justement ces équations algébriques.

Nous utiliserons, plus particulièrement, le programme d'analyse des transitoires électromagnétiques EMTP-RV : RV (« Electromagnetic Transient Program-Restructured Version ») [3] pour tester de nouvelles avenues et valider les algorithmes requis pour les exigences de la simulation des systèmes de contrôle dont les finalités ont été présentées ci-haut.

Dans les programmes de type EMTP, le système de contrôle est assemblé en diagramme-bloc. De plus, sa solution dans EMTP-RV requiert une matrice Jacobienne du système $\mathbf{ax}=\mathbf{b}$ mentionné plus haut. Les éléments de la Jacobienne sont évalués par linéarisation à partir de la solution au pas de temps précédent. Un processus d'itération de type Newton utilisant la Jacobienne est introduit selon les critères de précision voulue [4]. La solution de [4], actuellement fonctionnelle dans EMTP-RV, a amélioré plusieurs limitations précédentes telle la nécessité d'introduire un délai [5][6][7]. Plusieurs améliorations sont nécessaires.

Avec la généralisation de la possibilité d'utiliser le «Dynamic Link Library » (DLL) pour le système de contrôle il est nécessaire de rendre robuste et plus performante la solution des systèmes non-linéaires car, précisément, le DLL peut être vu comme une boîte-noire pouvant comporter des nonlinéarités à souhait. Les systèmes de contrôle complexes contiennent justement des DLL, soit pour reproduire des algorithmes spécifiques de contrôle, ou alors pour cacher à l'utilisateur certaines fonctionnalités en raison des propriétés intellectuelles.

De plus, les possibilités des boucles de rétroaction et la définition des équations personnalisées par l'utilisateur, suite aux tests sur terrain ou au comportement d'un système de contrôle durant l'opération, sont à considérer dans le modèle pratique de contrôle. On peut citer, par exemple, les modèles de la dynamique des centrales de production de la puissance et leurs contrôles validés par des tests sur terrain [8].

Tous ces modèles de contrôle requièrent un temps de calcul élevé en partie dû à la linéarisation requise pour former la Jacobienne. La simulation des certains modèles d'éolienne ou de lien CCHT requiert plusieurs heures pour une simulation de type transitoire électromagnétique (EMT) [1][9]. Ceci augmente les coûts d'analyse et d'étude des projets pour les utilitaires, manufacturiers et donc pour les usagers, utilisateurs finaux du réseau électrique.

Cette thèse vise à améliorer la solution des systèmes de contrôle. Plus spécifiquement elle vise à réduire le temps de simulation du réseau avec les systèmes de contrôle tout en maintenant

la précision requise pour l'analyse de phénomènes transitoires électromagnétiques. Les contributions spécifiques apportées par cette thèse sont présentées ci-bas.

2. Contributions apportées par cette thèse

Cette thèse apporte spécifiquement les contributions suivantes :

- 1) Une approche pour sélectionner le nombre optimal des nœuds dans le système de contrôle à partir desquels on reformule la représentation des équations ;
- 2) Une nouvelle formulation de la représentation d'équations avec une dimension réduite au niveau de la taille du système d'équations ;
- 3) Une approche d'analyse formelle des propriétés de la méthode point-fixe par analyse de la norme spectrale et, en général, la norme de la Jacobienne associée à la méthode à point-fixe ;
- 4) Une comparaison entre l'approche par solution des équations algébriques et celle par solution par variables d'état ;
- 5) Une illustration du théorème de Kantorovich sur un système de contrôle ;
- 6) Une représentation topologique des matrices Jacobienne et de leurs inverses en utilisant les outils de la théorie des graphes ;

Ces contributions ont été réalisées en différentes étapes. La méthodologie appliquée est présentée dans la section suivante.

Mentionnons qu'en plus des modules codés en FORTRAN, deux publications scientifiques ont été écrites [59][75].

3. Approche et évolution de la réalisation : méthodologie

Pour réduire le temps de calcul, nous analysons les possibilités de la réduction de la taille des équations du système de contrôle. Les blocs étant connectés tel que le sens de propagation du signal est unidirectionnel, on établit l'équivalent par représentation avec graphe orienté. Nous postulons donc qu'une représentation du système en taille réduite, si elle est adéquatement menée, peut en même temps maintenir la conformité de la solution et requérir moins de temps de

calcul qu'une représentation complète du système. Ceci est l'hypothèse principale de notre travail.

L'analyse topologique du système de contrôle est donc la première étape. La méthode « Depth First Search » (DFS) pour le parcours du graphe permet avec d'autres propriétés de la connectivité des blocs (la matrice d'adjacence, entre autres), d'identifier l'ordre dans la liste topologique des éléments et les boucles de rétroaction. Une méthode originale est introduite pour couper virtuellement les points de rétroaction et reformuler la représentation du système avec un nombre d'équations minimal selon les boucles dans le système. De fait, les tailles des matrices à résoudre dans le processus itératif de type Newton sont réduites. On réduit les fardeaux de représenter le système complet et des techniques pour la formation/mise à jour des grandes matrices.

Avec ce nombre minimal d'équations formées ci-haut, les méthodes de résolution performantes de type non-linéaire sont établies et plus encore, les caractéristiques de chaque groupe distinct sont isolées permettant de spécifier la méthode de résolution adéquate pour chaque groupe. Ceci permet par exemple d'assigner une méthode itérative, ou non, selon le cas : non-linéaire, linéaire ou même à point-fixe (ne nécessitant pas la formation de la Jacobienne) et ainsi d'augmenter la performance en temps de solution, selon la précision exigée. En effet, la formation de la Jacobienne nécessite l'évaluation de la dérivée de la sortie de chaque bloc par rapport à toutes ces entrées. Il faut donc appliquer une perturbation numérique de chaque entrée et évaluer la pente de la sortie en fonction de cette perturbation de l'entrée. Ceci exige beaucoup de temps. La méthode non-itérative quant à elle se limitera à une seule itération. On y présume que le pas de temps de la simulation est assez petit et les comportements des sorties des blocs sont assez lisses, tel que la solution soit acceptable après une seule itération. Cette variante de la solution est aussi appliquée dans [4].

Ces différentes variantes des méthodes de résolution sont alors formulées et appliquées sur différents cas pratiques de système de grande taille, incluant les éoliennes, pour en évaluer la performance. Mentionnons en terminant que la validation est aussi présentée. Elle inclut une discussion sur les limitations qui ouvre la voie à des futurs travaux. Dans la section ci-dessous nous présentons une revue de la littérature sur le sujet en comparant les différentes approches de résolution des équations de systèmes de contrôle.

4. Revue historique et bibliographique

La solution des équations de contrôle en réseau électrique n'a fait l'objet que de peu de publications, bien que la simulation des systèmes de contrôle soit une exigence pour les outils de simulation de régimes transitoires de réseaux électriques. Cela ne signifie pas pour autant que le domaine ne se soit pas développé.

Une première catégorie d'évolution des solutions des systèmes de contrôle est liée, entre autres, à la représentation des systèmes de contrôle, aux langages de programmation, aux aspects informatiques. Pour EMTP, cette catégorie d'évolution inclut les développements successifs autour du programme « Transient Analysis of Control Systems » (TACS) [5]. TACS a été développé pour la simulation des interactions dynamiques entre le système de contrôle et les composants de réseau électrique. Plus précisément, il était d'abord dédié à la modélisation des convertisseurs dans le CCHT puis il a été étendu pour d'autres équations de contrôle. Pour améliorer la flexibilité, une interface EMTP TACS-FORTRAN a été développée [6]. Il permet une modélisation plus facile en FORTRAN pour les composants de contrôle numériques. En 1994, [7] a amélioré TACS en enlevant les délais non nécessaires et en corrigeant certaines erreurs dans les termes d'histoire. Ceci permet de proposer une meilleure façon de traiter les données liées aux délais. Par ailleurs, les techniques d'initialisation furent améliorées. Au final, les usagers pouvaient ajouter leurs propres composants TACS codés en FORTRAN.

Dans les autres environnements de simulation de type EMT, des évolutions similaires ont été opérées. Pour MATLAB/Simulink [40], par exemple, les fonctions de contrôle peuvent être encapsulées dans des blocs en différents langages de programmation via des interfaces dédiés MATLAB, C ou FORTRAN. Ceci est précurseur de l'utilisation des liens de bibliothèques dynamiques (DLL) qui sera formalisée et intégrée dans [3].

En fait un des aspects de cette évolution est l'utilisation des DLL, tandis que l'autre aspect consiste en la génération des codes. Il s'agit de transformer la représentation syntaxique de blocs vers des modules exécutables directement intégrés dans le programme de la solution des contrôles. Ces derniers exigent des traitements informatiques particuliers pour reformuler les intrants des solutions des systèmes de contrôle. Les évolutions sur cet aspect "informatique" incluent aussi les améliorations d'interface usager et autres prétraitements des diagrammes des

modèles visant à détecter des simplifications possibles qui ne compromettent pas la validité du modèle [10].

Une deuxième catégorie d'évolution, qui rejoint cette thèse, est associée aux algorithmes de solution eux-mêmes. Dans les premiers moments d'évolution des outils de simulation, un délai d'un pas de temps de simulation était introduit pour casser la boucle algébrique et ainsi forcer une solution en une seule itération quand les blocs de contrôle sont en boucle, avec un des blocs non-linéaire. Ce genre d'astuces cause parfois des oscillations numériques et peut imposer des limitations au pas de temps de simulation, le contraignant à une petite valeur. Ceci rallonge le temps de simulation. La convergence de certains cas n'était parfois tout simplement pas garantie [4]. Les usagers devaient introduire systématiquement des délais dans les boucles, ce qui est encombrant et lourd pour les utilisateurs. Une difficulté plus contraignante encore limite cette méthode. Il s'agit du cas de deux blocs non-linéaires (ou plus) dans la boucle de rétroaction. Dans ce cas, introduire les délais ne garantit pas la convergence : les erreurs s'accumulent à chaque itération, le résultat final est dans une mauvaise région d'opération, ou alors, il y a « overflow » [4].

Pour résoudre ces lacunes, il a été proposé, dans [4], deux méthodes de solution basées sur la solution de type Newton avec une formation de la matrice Jacobienne par perturbation numérique. La solution de [4] est actuellement implémentée pour les systèmes de contrôle dans EMTP-RV [3].

Les coefficients de la matrice Jacobienne associés aux éléments non-linéaires sont mis à jour à partir du pas de temps passé et servent dans le processus itératif usuel de type Newton. La méthode d'itération complète et sa variante non-itérative ont le mérite d'éliminer la problématique d'ajout des délais rencontrée dans les versions précédentes.

La version non-itérative est seulement utilisée quand les comportements des blocs sont faiblement non-linéaires et les pas de temps, petits ; Sinon, la réponse retenue s'écarte de la vraie solution. La méthode itérative est aisément étendue aux modèles génériques personnalisés par l'utilisateur comme ceux du type « boîte-noire » sous forme DLL. La Jacobienne est aussi formée par perturbation numérique pour évaluer les dérivées de la sortie par rapport aux entrées à chaque itération. Cependant, le bloc « boîte-noire » devrait avoir un comportement non-linéaire raisonnablement simulable.

Ces développements pour EMTP-RV peuvent être mis en parallèle avec ceux d'autres logiciels de simulation des réseaux électriques de type EMT. Il s'agit, par exemple, des logiciels par représentation avec les variables d'état utilisée par MATLAB/Simulink. Pour ce dernier, comme pour EMTP-RV, les boucles avec éléments non-linéaires peuvent former une équation algébrique de type $f(x) = 0$ qui doit faire appel aux méthodes de solution des systèmes non-linéaires. Les développements des outils pour la solution des systèmes non-linéaires sont alors liés aux améliorations pour résoudre les équations algébriques. Nous présentons un petit survol très rapide sur les systèmes non-linéaires.

La problématique de la solution des modèles représentés comme des systèmes non-linéaires a été largement étudiée et a généré une abondante littérature scientifique et technique, mais il reste un domaine qui soulève beaucoup de défis. L'approche usuelle est d'utiliser des outils similaires à ceux utilisés pour un système linéaire après linéarisation à un point de fonctionnement donné. Dans [11], parmi d'autres, il est présenté un bon nombre de ces outils d'analyse, y compris le comportement qualitatif à proximité des points d'équilibre, etc.

En général, trois défis sont à relever : trouver la solution globale, assurer la robustesse de la convergence vers une solution locale et garantir ceci avec un choix varié des conditions initiales. En simulation de contrôle, nous nous restreindrons à établir une solution locale. On présumera, entre autres, que le pas de temps est réduit et que le comportement de blocs de contrôle est « correct », c'est-à-dire, qu'il retourne une sortie valable pour des « entrées valables ». On maintiendra aussi une perturbation numérique convenable et des conditions initiales les meilleures possible, consistantes avec la solution du système de contrôle. Produire le plus fidèlement les modèles physiques qui, par essence, selon [12], conduisent à des modèles mathématiques stables, sera de la responsabilité du concepteur du modèle (ou une portion du système de contrôle ou d'un contrôleur).

Par ailleurs, la simulation utilisant des valeurs discrètes dans le temps, nous présumerons donc que le système sera discrétisé et linéarisé selon les conditions mentionnées ci-haut. Les blocs de contrôle sont, soit exprimés via la méthode trapézoïdale reliant les entrées actuelles et les historiques des entrées et des sorties, soit via une perturbation numérique des entrées en un point donné autour duquel on dira que le modèle est linéarisé. La perturbation numérique permet une représentation du premier ordre du comportement du bloc. Il utilise la pente reliant la sortie

et l'entrée à proximité du point de fonctionnement. C'est une forme similaire à l'expansion en série de Taylor. La perturbation numérique se prête mieux pour les blocs de type boîte-noire puisque la configuration interne est inconnue.

Ce processus de perturbation a des implications par rapport aux défis présentés ci-haut.

Principalement nous devons nous assurer que la réponse en boucle ouverte du bloc à partir d'une entrée perturbée $u + \varepsilon u$ peut être adéquatement reliée à la réponse à l'entrée au prochain pas de temps $u + \Delta u$ pour chaque bloc et pour un ensemble de bloc groupé.

L'analyse de la solution de système de contrôle tenant compte du comportement entrée-sortie des blocs évoque l'analyse de stabilité pour le système non-linéaire pour l'approche dans l'espace des états. Dans [12], il est fourni un certain nombre de critères de stabilité de type Lyapunov. Plusieurs autres outils d'analyse sont disponibles pour ce type de problème. Parmi ceux-ci, mentionnons l'analyse de la stabilité entrée-sortie et, plus généralement, en présence de la rétroaction, une analyse des gains finis L-stable [13] découlant du théorème de petits-gains. D'autres notions utiles peuvent être puisées dans [11][12][14]-[21]. Une revue de l'évolution de la stabilité entrée-sortie a été rassemblée dans [22], couvrant quelques décennies de 1959 à 1985. La particularité et l'intérêt de l'analyse entrée-sortie et la stabilité consiste à lier le comportement du modèle physique et la solution du système de contrôle. En effet, du point de vue global, le comportement des systèmes de contrôle interfère sur celui du système tout entier. Les limitations spécifiques au type de solution, engendrant des oscillations numériques, pourraient induire des biais difficilement discernables sans une analyse approfondie du comportement de la stabilité. Finalement mentionnons qu'un système physiquement stable a tendance à admettre plus adéquatement un processus de solution moins sophistiqué, selon [12] [23]. C'est le cas d'une rétroaction négative avec un délai de pas de temps. Cet aspect n'est pas traité plus en profondeur dans la suite de la thèse. Nous l'évoquons pour placer le contraste par rapport à l'approche proposée et pour la délimitation du sujet de cette thèse. Terminons justement en évoquant un dernier axe d'évolution qui touche l'objet et une des contributions de cette thèse : il s'agit de la réduction de la taille des équations de modèle du système de contrôle.

Un autre axe d'évolution spécifique à l'approche proposée est la diminution de la taille du système d'équations. Diverses approches ont été appliquées pour diminuer la taille des équations. Lorsque les paramètres du système de contrôle sont connus pour isoler ceux à échelle rapide de

ceux à échelle lente, la technique de perturbation peut être appliquée avec succès. La plus utilisée est la technique de perturbation singulière. Nous rappelons le tutoriel de [24] qui introduit une perturbation singulière non linéaire de modélisation du transformateur ainsi que les équations du système de contrôle. Un autre travail intéressant par l'auteur de [24] est l'analyse appliquée aux systèmes électriques à grande échelle. Ils y ont réduit la complexité des systèmes de contrôle. Cette méthodologie a été principalement appliquée aux machines des centrales et à leurs systèmes de contrôle. Pourtant, dans tous ces cas, différentes échelles de temps devraient être connues ou dérivées et ainsi séparées. Il s'agit d'une limitation par rapport au bloc boîte-noire où nous n'avons pas tous les détails internes du bloc. Mentionnons, enfin, une méthode introduite par [25]. Il s'agit d'une méthode de solution utilisant la matrice de contingence. Cependant, la méthode est analytique et n'est donc pas indiquée pour les blocs définis par usager ou les boîtes-noires, ou tout au moins, nécessite une adaptation.

D'autres techniques liées à la théorie des graphes ont été appliquées à la fonction de transfert [26], mais nécessite un traitement hautement complexe pour les systèmes de contrôle génériques. On explore dans cette thèse comment prendre avantages sur les travaux connexes précédents présentés ci-haut.

L'approche par analyse topologique de graphe est explorée pour réduire la taille des équations. Le premier aspect est le DFS évoqué plus haut. Il est largement utilisé dans plusieurs domaines. Elle fournit par exemple la liste topologique des éléments de contrôle, la connectivité, ainsi que les prédécesseurs ou suiveurs dans l'ordre topologique. Ces derniers constituent la base des méthodes de réduction. Plusieurs algorithmes dans des langages de programmation variés sont disponibles [27][28]. Un deuxième aspect est la détermination des cycles dans un graphe. Les notions sur la matrice, d'adjacence, de connectivité, d'accessibilité (« reachability » en anglais) et sur les composantes fortement connectées sont largement utilisés, voir à ce sujet, les travaux de [23] dédiés aux systèmes de contrôle. Une forme particulière de représentation de la connectivité de graphe du système est celle de la connexion de rétroaction, connue sous le nom « feedback connexion » (voir entre autres, [11][19][23]). Pour ces derniers, ceci a été fait pour analyser les caractéristiques de propriétés des systèmes de contrôle. Des relations utiles développées par [23] aident à formuler la connexion de rétroaction avec les matrices d'adjacence, d'accessibilité, etc. Notre approche s'inspire de ce travail mais introduit les propriétés des valeurs

propres pour reformuler les équations comme pour [11][19][23] et plus précisément, les propriétés d'avoir toutes les valeurs propres nulles tel que rapporté par [29][30].

L'approche présentée se compare à celle d'élimination des nœuds [31][32]. Mais la différence est que les fonctions des blocs doivent être considérées linéaires. Les versions linéarisées sont comparables aux cas linéaires et ont les mêmes « signatures topologiques ». Car, en effet, pour une variable v l'existence d'une dérivée partielle par rapport à cette dernière, est clairement associée à la connectivité entre le bloc ayant la variable v comme sortie et le bloc pour lequel on établit la dérivée partielle, c'est-à-dire celle à linéariser. On utilisera ce fait pour une équivalence d'élimination des nœuds pour un cas linéaire et celle linéarisée autour d'un point spécifique.

Après ce dernier axe d'évolution de la solution de système de contrôle introduisant l'approche proposée, nous entamerons un premier chapitre sur les généralités sur les systèmes de contrôle. Pour clore cette introduction rappelons finalement l'agencement spécifique des différentes parties, sections et chapitres dans la section suivante.

5. Organisation de la thèse

Cette thèse comprend six chapitres, outre cette introduction et la conclusion.

Chapitre 1 : Le chapitre premier traite des généralités sur la modélisation mathématique des systèmes de contrôle. Il y est tour à tour traité des sections suivantes. La première est le système de contrôle et le réseau électrique. La deuxième est la description des types de blocs. Cinq critères pour classer les blocs sont présentés. La troisième section présente le regroupement des blocs de contrôle. Deux regroupements à action directe (« Feedforward ») et celle par rétroaction (« Feedback »). La quatrième section rappelle le lien entre la dynamique physique et les modèles mathématiques. La dernière section de ce premier chapitre présente très sommairement quelques fonctionnalités de système de contrôle.

Chapitre 2 : Le deuxième chapitre a pour objectif de présenter la transformation du système d'équations. Ceci est au cœur de la nouvelle approche. Il y sera traité des techniques basées sur la théorie des graphes. Les propriétés et définitions, telles la matrice d'adjacence et d'accessibilité, utilisés dans la suite, y sont rappelées. Aussi, une technique de détection et de suppression des cycles dans le graphe est présentée. Le chapitre termine par une représentation

des équations aux éléments de rétroaction des boucles, suivie par la décomposition du système sous forme d'interconnexion aux points de rétroaction (DIPR). Cette dernière fournit un système simultané d'équations qui est une des contributions spécifiques de cette thèse.

Chapitre 3 : Le chapitre trois se consacre à la solution des systèmes de contrôle dans la nouvelle approche présentée au chapitre deux. La solution a quatre variantes qui sont tour à tour présentées. Elles sont précédées par des calculs préliminaires communs : l'évaluation des fonctions objectives du processus itératif et l'évaluation de la matrice Jacobienne du processus itératif. Les quatre variantes sont sous deux catégories : itérative et non-itérative. Sous la première catégorie, on a :

La méthode de Newton de base ;

La méthode Dogleg de la Routine HYBRJ1 de MINPACK [50];

Et la solution par méthode point-fixe.

Et sous la deuxième catégorie, on a la solution non-itérative. Ces limitations sont aussi discutées.

Chapitre 4 : Ce quatrième chapitre illustre l'application de la méthode proposée sur différentes classes de contrôle : le PLL, une machine asynchrone modélisée par équations algébriques et, pour un cas de grande taille, un système avec éoliennes intégrées au réseau IEEE 39-barres. Une discussion sur la performance de la nouvelle méthode, la précision et le gain en temps de calcul des différentes variantes clôt ce chapitre.

Chapitre 5 et 6 : Le cinquième et le sixième chapitre approfondissent l'analyse de la nouvelle approche sous trois angles. Le premier angle est traité dans la première section du chapitre cinq. La représentation complète classique et la représentation réduite par DIPR sont comparées pour analyser la conformité de deux représentations. La technique de substitution bien connue dans la résolution des systèmes d'équations est utilisée. Le deuxième angle, dans la deuxième section du chapitre cinq, est la comparaison avec la représentation par variables d'état. Selon que les variables d'état dépendent d'un seul point ou de plusieurs points de rétroaction, la conformité dans les deux modes de représentation est montrée.

Enfin, le dernier angle d'analyse approfondie est présentée au chapitre six. Sa motivation est présentée en début de chapitre. Il s'agit brièvement de voir les comportements face aux conditions initiales dans la formulation proposée et dans la formulation complète. Les formes ou

allures des normes des matrices Jacobiennes et des normes de leurs inverses sont introduites. Pour établir ces normes, encore une fois, une approche par théorie des graphes, précisément par analyse de graphes de Coates et de Masson, est introduite. Finalement, le théorème de Kantorovich est utilisé pour analyser les changements des comportements pour les mêmes conditions initiales. Ceci projette l'un des travaux futurs qui, justement, pourrait traiter des mécanismes efficaces d'initialisation.

CHAPITRE 1 GÉNÉRALITÉS SUR LA MODÉLISATION MATHÉMATIQUE DES SYSTÈMES DE CONTRÔLE

Dans ce chapitre, il sera présenté un aperçu général de la modélisation mathématique des systèmes de contrôle composés des blocs élémentaires. Les différents types de blocs de contrôle sont aussi présentés. Il servira aussi à replacer le système de contrôle dans le contexte plus large de simulation des réseaux électriques. Les définitions et concepts de base y sont aussi présentés.

1.1 Introduction

Le terme « contrôle » en simulation des réseaux électriques est attribué aux composants qui ne modifient pas directement le couple courant électrique i et tension électrique v . De ce fait, ils ne sont pas associés directement aux équations qui lient i et v . Les lois d'Ohm généralisées, les lois de Kirchhoff et les équations nodales qui lient justement ces derniers ne comportent donc pas les composants de contrôle. De façon généralisée on peut écrire i en fonction de v ($i = Y(v)$) pour chacune des composantes, ou branche et nœuds associés, en réseau électrique [34]. En contrôle, les composants ont des entrées u et des sorties y (on écrit $y=f(u)$) pouvant être connectées aux équations de réseau d'une manière indépendante comme mesure (c'est-à-dire $y=f(v)$ ou $y=f(i)$). Ils peuvent aussi être connectés comme excitation via des sources contrôlées (c'est-à-dire $i=f(u)$ ou $v=f(u)$) intervenant dans l'équation dynamique $i = Y(v)$. Enfin, ils peuvent être associés au réseau, en modifiant les paramètres de la fonction (ou opérateur) Y de la modélisation du réseau électrique comme, par exemple, en actionnant le changement de topologie du réseau (interrupteur commandé) ou encore en encapsulant le comportement non-linéaire d'une composante.

Les fonctionnalités suivantes se placent dans la catégorie « contrôle » : les mesures, les équations mathématiques et les algorithmes divers traduisant un comportement spécifique durant la simulation ou intégrées dans un module dédié à l'aide de la décision, les commandes et contrôles des machines, des interrupteurs, etc.

Dans ce chapitre, on décrit les modèles mathématiques sous différents angles. Le premier est l'interaction entre le système de contrôle et le réseau électrique ; ensuite, des différents types de blocs : statiques, dynamiques avec termes d'histoire, discrets avec discontinuités, non-linéaires

et de « boîtes-noires ». Puis, du regroupement des blocs pour former des groupes à action directe ou à réaction. Une section suit pour introduire la modélisation dynamique elle-même, suivie par une illustration des quelques fonctionnalités des systèmes de contrôle.

1.2 Le système de contrôle et le réseau électrique

La simulation des systèmes de contrôle est couplée à celle du réseau électrique. Les modèles doivent alors être adéquatement établis pour traduire le phénomène physique que subissent ou qu'engendrent les systèmes de contrôle vis-à-vis du réseau électrique. Pour illustrer comment s'intègrent les contrôles dans une étude de réseau, partons des applications de simulation d'un réseau en mettant en lumière le besoin du système de contrôle requis. Nous prenons, à cet effet, deux exemples de *test d'équipement* : un premier exemple de surtension temporaire nécessitant une surveillance des parafoudres et un deuxième exemple de tension temporaire de rétablissement (TTR) d'un arc de disjoncteur (avec modèle Cassie-Mayr : voir les cas introductifs dans [3]). Ces deux exemples peuvent être menés précisément lors de la *planification des réseaux*. Il s'agit respectivement de la *simulation des pertes lignes ou des groupes* et d'une *étude de court-circuit*. Notez que d'autres applications de la simulation font intervenir les contrôles sous d'innombrables manières, notamment lors des *études de stabilité* : stabilité statique, de fréquence et de tension, connaissance du comportement dynamique, etc. Les détails de la classification peuvent être trouvés dans [2], entre autres.

Par ailleurs, ces deux exemples sont, précisément, des phénomènes transitoires électromagnétiques (EMT), respectivement, *défaut proche en ligne* et *étude des surtensions temporaires*. D'autres phénomènes transitoires sont, pour n'en citer que quelques uns, les coups de foudre et transitoires électromécaniques [34].

Les modèles adéquats du réseau doivent entre autres assurer une bonne représentation dans les gammes de variation de la fréquence des ondes transitoires. Dans les deux exemples de surtension temporaire et de TTR, les contenus fréquentiels à prendre en compte doivent être respectivement, 0.1 à 1 kHz, d'une part et de la fréquence fondamentale (50 ou 60Hz) à 20 kHz d'autre part. D'autres gammes des fréquences peuvent aller plus haut [34]:

- a. surtensions à front rapide : entre 10 kHz et 3 MHz ;
- b. surtensions à front très rapide : entre 100 kHz et 50 MHz.

La simulation des transitoires associés à ces phénomènes impose une modélisation du réseau en conséquence afin de refléter les actions et comportements de tous les équipements susceptibles d'être affectés ou ceux dont dépendent les phénomènes sous analyse.

Pour le réseau, les modèles des machines, des lignes, des parafoudres (caractéristiques non-linéaires) sont établis dans [3] et [34]. La simulation peut être faite en résolvant successivement, pour chacune des composantes, le système d'équations différentielles dans le domaine du temps [35][36] et assemblé en équations algébriques sous forme nodale augmentée [33][37].

Pour chaque branche j dans le réseau on a :

$$\mathbf{i}_j = \mathbf{y}_j \mathbf{v}_j + \bar{\mathbf{i}}_j, \quad (1.1)$$

où le vecteur des courants \mathbf{i}_j représente les injections de courant à chacun des nœuds du composant modélisé. La matrice \mathbf{y}_j est la matrice d'admittance nodale du composant générique. Le vecteur \mathbf{v}_j comprend les tensions des nœuds du composant et le vecteur $\bar{\mathbf{i}}_j$ représente les sources de courant internes du composant.

Le système de contrôle est connecté au réseau via les variables d'activation du réseau (sorties S des blocs de contrôle), plus précisément, des entrées des sources contrôlées ou des signaux contrôlables ainsi que des mesures du réseau (entrées E des blocs de contrôle). Les signaux E et S sont associés aux variables \mathbf{i}_j , \mathbf{v}_j et \mathbf{y}_j dans (1.1). Schématiquement, on aura donc le diagramme de la Figure 1-1.

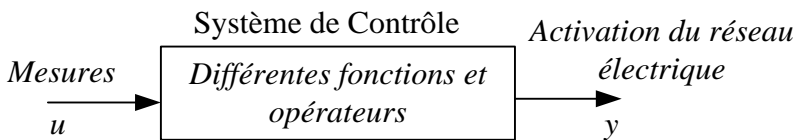


Figure 1-1: Schéma générique d'interaction réseau/système de contrôle

Illustrons cette description aux deux exemples choisis ci-haut. Les différents diagrammes des réseaux électriques se trouvent, plus bas, à la Figure 1-2 et à la Figure 1-3 pour, respectivement, l'étude de la surtension temporaire (énergie dissipée dans les parafoudres) et pour la coupure du courant de court-circuit (l'arc dans le disjoncteur avec le modèle Cassie-Mayr).

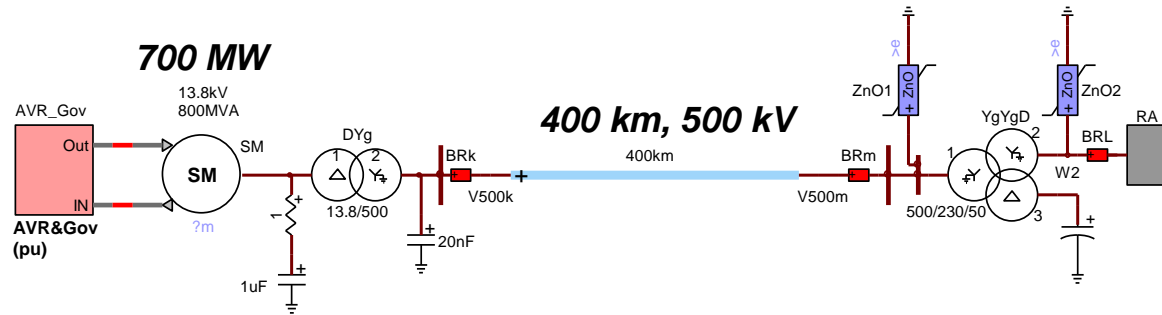


Figure 1-2: Réseau électrique pour étude des surtensions temporaires (tiré de [34])

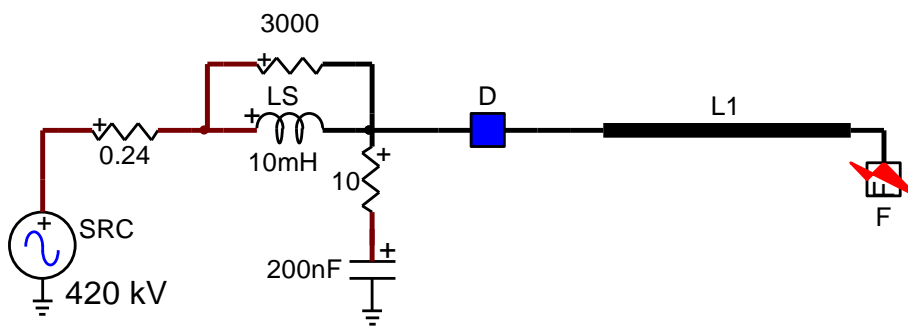


Figure 1-3: Simulation d'un défaut proche en ligne avec un modèle d'arc de disjoncteur [34]

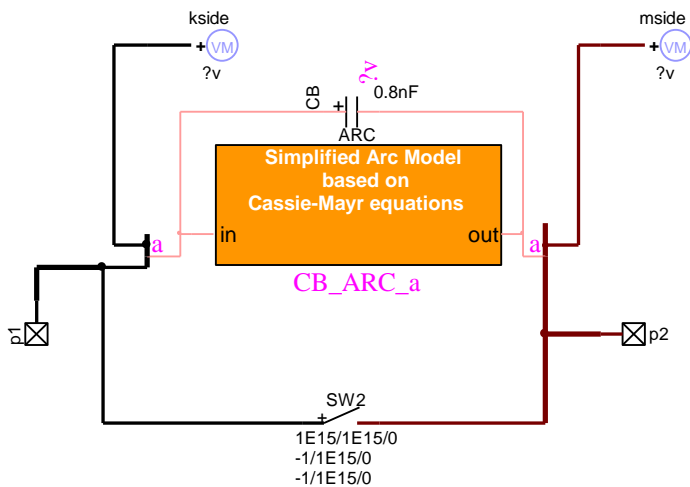


Figure 1-4: Sous-circuit du disjoncteur D dans le schéma de la Figure 1-3. Sur la phase a est modélisé l'arc selon les équations de Cassie-Mayr

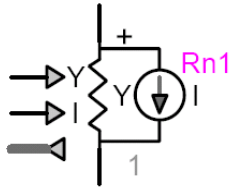


Figure 1-5: Admittance contrôlée intervenant dans les équations du réseau présentées en (1.1)

Les systèmes de contrôle intervenant dans les différents circuits électriques :

- a. Dans la Figure 1-2, on a, entre autres :
 - les contrôleurs de machines à l'extrême-gauche dans la Figure 1-2
 - les fonctions pour calculer les énergies aux parafoudres Zno1 et Zno2 dans la Figure 1-2, suite aux surtensions. En effet, les limites des énergies dissipées par ces derniers sont justement un critère de conception du réseau ;
- b. Dans la Figure 1-3, on a, entre autres :
 - les fonctions traduisant le modèle d'arc Cassie-Mayr dans le disjoncteur D dans la Figure 1-4. Ce dernier comporte un système d'équations différentielles qu'il faut intégrer par une méthode numérique appropriée. Il utilise, en rétroaction, les variables courant et tension du réseau (nécessité d'itérer au besoin entre la solution des contrôles et celle du réseau). Ce dernier donne à sa sortie une valeur de conductance intervenant dans la Figure 1-5

Schématiquement, pour ces 3 systèmes, les fonctions de contrôle et leurs interactions avec les signaux du réseau peuvent être présentées comme à la Figure 1-6.

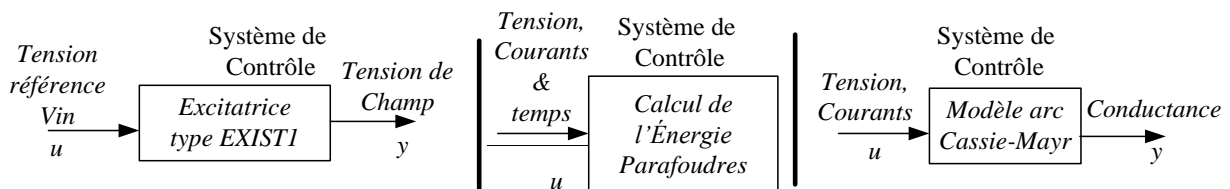


Figure 1-6: Trois formes d'interactions entre le réseau et le système de contrôle via les variables dans (1.1)

Les exemples simples ci-dessus illustrent quelques-unes des applications variées de contrôle :

- a. le mécanisme de contrôle pour les composants, avec les boucles classiques d'asservissement ;
- b. des équations usagers personnalisées variées dont les résultats peuvent d'ailleurs être des signaux de rétroaction d'un composant ou d'un mécanisme ;
- c. des modélisations complexes pouvant nécessiter des boucles de rétroaction pour reproduire des phénomènes physiques ; etc.

Bien que ces trois cas ne représentent qu'une infime partie des possibilités, on peut déjà noter différents types de blocs nécessaires et les différentes configurations possibles dans les systèmes de contrôle. Nous illustrons les types de bloc et leurs configurations ou connexions pour former les groupes dans la section qui suit.

1.3 Types de blocs de contrôle et regroupement

La diversité des fonctionnalités des systèmes de contrôle découlent des combinaisons et connexions variées possibles. Ces dernières elles-mêmes sont liées à la diversité des types de blocs et leurs caractéristiques. Une catégorisation bien indiquée peut comporter cinq critères : topologique (présence ou non d'entrées ou de sorties), dynamique (blocs statiques ou blocs dynamiques avec termes d'histoire), caractère continu ou discret, linéaire (blocs avec ou sans discontinuité, blocs non-linéaires et blocs de type « boîte-noire ») et, finalement, à coefficients constants ou variables. En pratique, un bloc aura les caractéristiques combinant chacun des critères ci-haut. Le modèle mathématique incorporera les quatre derniers critères, alors que l'effet du bloc dans le groupe dépendra de son modèle mathématique et du critère topologique, sa connexion aux autres blocs. Dans la section qui suit, les types de blocs sont présentés et dans la suivante, deux types de regroupement sont présentés : à action directe (« feedforward ») et à rétroaction (« feedback »).

1.3.1 Types de blocs

1.3.1.1 Blocs avec ou sans entrée et avec ou sans sortie

1.3.1.1.1 Blocs avec ou sans entrée

Les blocs avec entrée (multiples ou non) sont les blocs connectés à la sortie d'un autre bloc de contrôle. Les blocs sans entrée comprennent les blocs source de signal de contrôle. Pour chaque pas de temps et selon des paramètres définis, la sortie est déterminée à partir d'une fonction spécifiée pour le bloc. Typiquement, ces blocs servent entre autres à définir des fonctions spécialisées pour les courants et tensions d'excitation des systèmes électriques.



Figure 1-7: Caractérisation des blocs selon la topologie : bloc sans entrée et bloc sans sortie

Ce sont des blocs qui peuvent être le commencement d'un groupe des blocs inter-reliés, s'ils ont une sortie.

1.3.1.1.2 Blocs avec ou sans sortie

Les blocs avec sortie (multiples ou non) prolongent le parcours d'un signal. S'ils ont une entrée, alors ils ne peuvent être ni le début ni la fin d'un groupe, mais ils se retrouvent à l'intérieur d'un groupe pouvant être bouclés avec un chemin de retour. S'ils ont deux sorties ou plus, alors ils peuvent appartenir à plusieurs groupes. Les blocs sans sortie sont quant à eux essentiellement la fin d'un groupe ; Ou alors, ils servent à la mesure d'un signal.

1.3.1.2 Blocs statiques

Les blocs statiques sont ceux pour lesquels les paramètres de la fonction décrivant le lien entre la sortie et les entrées sont indépendants du temps.

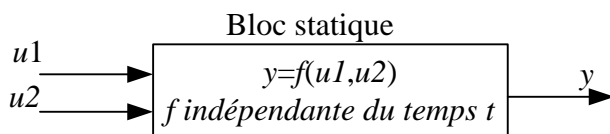


Figure 1-8: Bloc statique : fonction interne indépendante du temps t

Les exemples des blocs de ce type sont entre autres : le bloc gain, le bloc somme (et tous les blocs opérateurs : division, produit, etc.). Leurs fonctions f sont respectivement telles que :

$y = K \times u$ et $y = u1 + u2$ ou K est la valeur du gain. Ces deux blocs seront représentés par des symboles qui décrivent leur comportement comme ci-dessous :

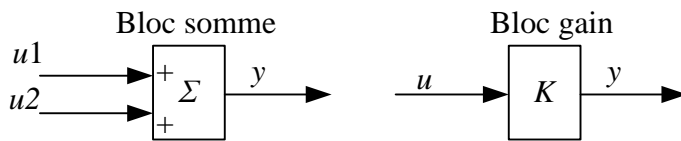


Figure 1-9: Blocs somme et gain

Notez qu'il n'y a aucune dépendance avec le temps, ni avec le pas de temps d'intégration de la simulation. Cependant, pour un pas de temps fixe, certains blocs auront un comportement statique, même si ces derniers dépendent du pas de temps. En effet, celui-ci peut être considéré constant durant toute la simulation. Il s'agit des blocs dérivés, par exemple. Leurs sorties sont des dérivés des entrées.

1.3.1.3 Blocs dynamiques avec termes d'histoire

Les blocs dynamiques se distinguent de ceux statiques car ils dépendent explicitement du temps, surtout du temps passé via les termes d'histoire. Dans cette catégorie on a les blocs Intégrale, les fonctions de transfert en « s » et en « z », les accumulateurs dépendant de la nouvelle valeur à chaque pas de temps, etc.

1.3.1.4 Blocs avec coefficients constants ou variables

Les fonctions des blocs avec coefficients constants ont des termes bien spécifiques dont les paramètres sont constants tout au long de la simulation. Il s'agit des gains et somme évoqués plus haut, par exemple, mais aussi des intégrales dont la discrétisation dépend du pas de calcul.

Par contre les blocs à coefficients variables présentent des termes pouvant changer grandement, variant du tout-au-rien. On peut citer un sélecteur d'entrée selon un signal d'activation quelconque.

1.3.1.5 Blocs non-linéaires

Les blocs non-linéaires sont, par définition classique, ceux pour lesquels la sortie n'est pas une fonction linéaire des entrées. Le principe de superposition ne s'applique pas en considérant deux entrées distinctes. La sortie n'est pas proportionnelle à l'entrée.

1.3.1.6 Blocs « boîtes-noires »

Les blocs boîtes-noires sont ceux pour lesquels le comportement de la sortie en fonction des entrées est inconnu et expressément caché à l'utilisateur. On devra donc, au moins, tenir compte du fait que ce comportement pourrait être non-linéaire.

1.3.1.7 Blocs discrets et blocs avec discontinuité

Les blocs discrets sont ceux pour lesquels on définit leurs valeurs à chacun des pas de temps discrets. Il s'agit, par exemple, de fonction en z .

Quand les valeurs d'un point de temps vers le suivant ou le plus proche du premier peuvent être définies d'une manière univoque, on peut établir une valeur de sortie pour tout incrément d'entrée, le bloc est à comportement continu. Sinon, c'est-à-dire si, pour deux points de temps contigus, il y a un saut des valeurs de la sortie du bloc, le bloc est discontinu. Une définition formelle des dérivées sur des telles fonctions donnent des critères pour déceler ce comportement. Le bloc « timer », par exemple, donne un comportement discontinu si on ne définit pas une rampe de variation de la sortie. Les autres blocs de cette catégorie sont le Tableau 1.1 sous la rubrique « immediate elements ».

En simulation de contrôle, en réseau électrique, on considérera que les fonctions des équations des blocs et le système d'équations des groupes découlant sont continus par morceaux, c'est-à-dire que les comportements sont analysés entre les points de temps pour lesquels il n'y a pas de discontinuité (saut). Les points de discontinuité eux-mêmes seront traités par des approximations et des techniques diverses.

1.3.1.8 Tableau récapitulatif des types de blocs de contrôle

D'autres types de blocs n'ont pas été évoqués. Dans le Tableau 1.1, il est rassemblé les blocs le plus utilisés dans [3]. Leurs types sont aussi présentés. La terminologie anglaise est conservée comme en [3]. Les connexions d'un certain nombre de ces blocs forment un groupe dédié à une tâche particulière. La solution se référera d'ailleurs à ce groupe spécifique. Le groupe est associé à la modélisation du comportement du système. Deux types de groupe sont à considérer. Ils sont présentés ci-bas.

Tableau 1.1: Récapitulatif des types de blocs

Bloc de contrôle	Type ou catégorie
signal scope	signal monitoring
constant signal	signal sources
simulation probe	
voltage probe	
power-system observable	
cos signal source	
function signal source	
ramp signal source	
step signal source	
triangular signal source	
signal gain	constant-coef static linear elements
signal sum	
mid-step dx/dt	
end-step dx/dt	
d2x/dt2	
signal(t0) holder	
signal gain with limits	constant-coef static linear elements with constraints
integral(xdt)	constant-coef dynamic linear elements
s function	
1/z function	
integral(xdt) with reset and/or limits	constant-coef dynamic linear elements with constraints
s function with reset and/or limits	
1/z function with reset and/or limits	
signal selector	variable-coef static linear elements
output selector	
signal comparator	static nonlinear immediate elements
zéro-hit detector	
rising-pulse detector	
falling-pulse detector	
constant delay	
variable delay	
discrete delay	
binary variable delay	
one-time-step delay	
function definition	
function	
table function	
truth table function	
direct signal sampler	dynamic nonlinear immediate elements
accumulating signal sampler	
minimum signal sampler	
maximum signal sampler	
signal holder	
rms meter	
frequency meter	
signal timer	event elements

1.3.2 Regroupement topologique des blocs de contrôle

Deux types de regroupement sont importants à noter dans l'analyse topologique de contrôle. Car la méthode de solution en dépend. Le premier est celui à *action directe*. On le désigne plus communément par le terme anglais « Feedforward ». Un deuxième groupe est à *rétroaction*. On le désigne par « Feedback » en anglais. Cette thèse se concentre plutôt sur le deuxième groupe, ceux à *rétroaction*. Les défis de calcul pour le cas comportant des nonlinéarités dans la boucle de rétroaction sont les plus grands. Le groupe à action directe est relativement plus facile. Les deux sont illustrés ci-bas.

1.3.2.1 Groupe à action directe (« Feedforward »)

Pour ce type de groupe des blocs de contrôle, les signaux à la sortie du groupe dépendent des évaluations successives séquentielles de chacun des blocs appartenant au groupe dans un ordre bien défini. Pour le résoudre, on applique une évaluation séquentielle des blocs. On peut donc parcourir ce dernier sans revenir plus d'une fois sur aucun des blocs.

1.3.2.2 Groupe à rétroaction (« Feedback »)

Le groupe à rétroaction, quant à lui, nécessite une réévaluation d'au moins un bloc quand on parcourt le groupe, bloc après bloc. Une définition faisant appel aux notions de la théorie de graphe désigne ce groupe comme étant une composante fortement connectée car, de n'importe quel bloc on peut atteindre tout autre bloc du groupe. Ceci sera pris en compte dans le chapitre suivant pour reformuler le groupe de contrôle. D'ailleurs mentionnons que dans la suite de thèse, le système de contrôle est considéré comme ayant une rétroaction. Sinon, il s'agirait de la solution séquentielle qui, comme mentionné plus haut, est relativement simple à calculer.

Pour terminer cette description des blocs et leur regroupement, soulignons que le groupe de blocs de contrôle découle du modèle mathématique car les connexions ne sont rien d'autre que les dépendances des équations de chacun des blocs pris individuellement. Il est donc associé au modèle physique. On survole ci-bas le lien entre la dynamique des phénomènes et le modèle mathématique des systèmes de contrôle, ou groupe de blocs de contrôle.

1.4 Dynamique physique et modèles mathématiques

Sans entrer dans les détails et spécificités de l'établissement des modèles de composantes, rappelons que la modélisation physique du comportement des systèmes face aux phénomènes dynamiques, dans la plupart des cas, peut être décrite par les modèles mathématiques sous forme d'équations différentielles de la forme (1.2) que nous empruntons de [38]:

$$\frac{d^{q_i}}{dt^{q_i}} y_i(t) = h_i \left(\begin{bmatrix} t, \\ y_1(t), \dot{y}_1(t), \dots, \frac{d^{q_1-1}}{dt^{q_1-1}} y_1(t), \\ y_2(t), \dot{y}_2(t), \dots, \frac{d^{q_2-1}}{dt^{q_2-1}} y_2(t), \\ \vdots \\ y_n(t), \dot{y}_n(t), \dots, \frac{d^{q_n-1}}{dt^{q_n-1}} y_n(t), \\ u_1(t), u_2(t), \dots, u_p(t) \end{bmatrix} \right), \quad (1.2)$$

où q_i est l'ordre de la dérivée pour la sortie y du bloc de contrôle i . Notez que dans le cas générique, on considère que ce q_i -ième ordre de la dérivée s'écrit en fonction des tous les ordres de la dérivée dans le système sous considération. Dans cette formulation générique, la fonction (ou opérateur) h_i spécifie justement les liens entre différentes dérivées et t est la variable de temps. On note que le temps est considéré continu. Il sera traité comme discret dans la suite de cette thèse en présumant qu'une transformation appropriée au niveau des modèles a été introduite. On considère enfin, plusieurs entrées u_i (par exemple, dans (1.2), on a un nombre maximal de p entrées).

On a donc au final un système des n équations différentielles couplées, d'ordre q_i , chacune.

Rappelons que cette représentation est générale. Elle s'applique aussi au réseau électrique pour inclure, entre autres, le comportement des composants électriques de base du réseau : la résistance, la capacité et l'inductance. Les dérivées du courant ou de la tension, d'ordre un ou deux, par rapport au temps, décrivent justement le comportement de ces composants. Ceci vaut aussi pour les lignes de transmission. Les équations de base, respectivement, pour la résistance R , l'inductance L , la capacité C sont :

$$i_R = \frac{1}{R} v(t), \quad (1.3)$$

$$v_L = L \frac{d}{dt} i, \quad (1.4)$$

$$i_C = C \frac{d}{dt} v, \text{ et} \quad (1.5)$$

Pour le modèle de la ligne de transmission (TLM « Transmission Line Model »), les équations sont :

$$\begin{cases} -\frac{\partial}{\partial z} v_{TLM} = L' \frac{\partial}{\partial t} i_{TLM} \\ -\frac{\partial}{\partial z} i_{TLM} = C' \frac{\partial}{\partial t} v_{TLM} \end{cases}, \quad (1.6)$$

où L' et C' sont les valeurs par unité de longueur, dans la direction longitudinale z de la ligne, de l'inductance et de la capacité du modèle de la ligne de transmission.

De plus, les lois des nœuds et des mailles servent à formuler les équations différentielles couplées quand on a plusieurs de ces composantes connectées. Par exemple, la somme de courant est nulle pour chaque nœud j ayant m branches :

$$i_{j,1} + i_{j,2} + \dots + i_{j,m} = 0. \quad (1.7)$$

En combinant les relations (1.3)-(1.7), on peut former une des équations du système en (1.2).

On peut déjà déceler la forme de l'équation (1.1) du réseau électrique sur laquelle nous avons introduit le lien avec le contrôle. Pour expliciter ceci, le système d'équations sera reformulé pour introduire la représentation des dérivées du premier ordre et une méthode d'intégration spécifique (trapézoïdale pour EMTP) débouchera sur des équations algébriques.

Pour le système de contrôle, une représentation similaire est possible, mais l'équation du bloc, pris individuellement, est souvent moins complexe. Par exemple, pour un bloc dérivée ou un bloc somme, on a respectivement les équations de type :

$$y_i(t) = \frac{d}{dt} u(t),$$

$$y_i(t) = \sum_{j=1}^p W_j u_j.$$

Considérant la connexion entre plusieurs blocs de contrôle, on forme encore fois le système d'équations (1.2).

Pour illustrer les formes finales des équations de modélisation, on applique une des techniques de résolution des équations différentielles. Un système d'ordre supérieur peut être représenté par un système de plusieurs équations différentielles d'ordre un. Si n_s est le nombre des variables supplémentaires intermédiaires pour exprimer les n_d dérivées d'ordre supérieur plus grand qu'un, alors, selon [38], (1.2) devient :

$$\frac{d}{dt} x_i(t) = f_i \left(\begin{bmatrix} t, \\ x_1(t), \\ x_2(t), \\ \vdots \\ \vdots \\ x_n(t), \\ u_1(t), u_2(t), \dots, u_p(t) \end{bmatrix} \right), \quad (1.8)$$

où les $x_i(t)$ sont les nouvelles variables et leur nombre total (intermédiaires supplémentaires et directes) est $n = n_s + n_d$.

Nous présumerons que cette représentation est toujours possible dans les modèles des systèmes de contrôle dans cette thèse. Notez que le système (1.8) peut être non linéaire. En notation vectorielle, on a :

$$\frac{d}{dt} \mathbf{x}(t) = \mathbf{f}[t, \mathbf{x}(t), \mathbf{u}(t)], \quad (1.9)$$

où \mathbf{f} est un ensemble-vecteur des fonctions ; T (dans $[]^T$) est le signe de transposition d'un vecteur) et $\mathbf{x}(t) = [x_1(t), \dots, x_n(t)]^T$ et $\mathbf{u}(t) = [u_1(t), \dots, u_p(t)]^T$.

La *simulation* d'un système de contrôle consiste à déterminer l'évolution, dans le temps des x et y de système ci-haut, suite aux valeurs u . Ce qui équivaut aussi à *calculer la solution* du modèle dans le domaine de temps. Un aspect à introduire est d'ores et déjà le fait que résoudre le modèle

dans le temps impose des moments distincts d'observation, c'est-à-dire un écoulement discret du temps. Les modèles équivalents seront donc les versions en temps discret de ceux en haut.

Pour refléter ce fait, conceptuellement, il est introduit une transformation bien connue dans la représentation de l'échantillonnage. Une formulation parmi tant d'autres est celle que nous empruntons de [39] :

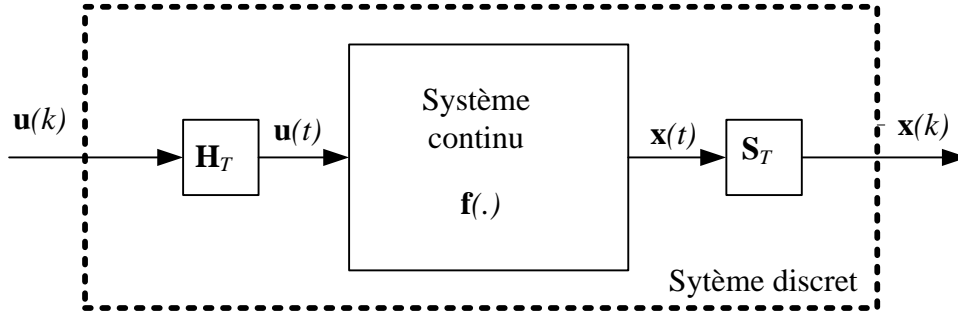


Figure 1-10: Transformation du système continu pour obtenir le système discret

Où les systèmes \mathbf{H}_T et \mathbf{S}_T associent respectivement $\mathbf{u}(t)$ à $\mathbf{u}(k)$ et $\mathbf{x}(t)$ à $\mathbf{x}(k)$

Le système à étudier dans le domaine discret est alors :

$$\mathbf{x}(k+1) = \tilde{\mathbf{f}}[\mathbf{x}(k), \mathbf{u}(k)], \quad (1.10)$$

où $\tilde{\mathbf{f}}$ est un ensemble-vecteur des fonctions dans le domaine discret associé à celui pour le domaine continu \mathbf{f} ; et :

$$\mathbf{x}(k) = [x_1(k), \dots, x_n(k)]^T \text{ et}$$

$$\mathbf{u}(k) = [u_1(k), \dots, u_p(k)]^T.$$

Notons qu'un cas particulier de la formulation en (1.10), est le cas où les fonctions \mathbf{f} sont telles que l'état actuel est déterminé par les états antérieurs et les entrées ; et que la sortie actuelle est déterminée par les états actuels et l'entrée actuelle. On a alors un système à états quasi-isolés. Les termes associés aux variables d'état peuvent être regroupés et découplés de ceux des entrées u . Alors, on peut avoir la représentation connue du système dans l'espace des variables d'état \mathbf{x} :

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k). \quad (1.11)$$

En fixant les variables d'intérêt comme sorties, on a, grâce à la linéarité :

$$y(k+1) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k), \quad (1.12)$$

où les matrices \mathbf{A} , \mathbf{B} , \mathbf{C} et \mathbf{D} sont caractéristiques du système modélisé.

Un autre élément à noter est que, pour un temps donné, les équations utilisent une entrée fixée \mathbf{u} , le processus d'itération ou de calcul devient donc indépendant de l'entrée \mathbf{u} .

Sans perte de la généralité, nous traiterons dans la suite dans cette thèse l'équation (1.10) comme le modèle du système de contrôle. Ceci est conforme avec la méthodologie de la solution EMTP qui est à pas fixe. Ainsi, nous allons volontairement omettre le « \sim ». Les blocs nécessitant la variation continue du temps seront discrétisés. Voici, pour rappel, en aperçu, la méthodologie de la modélisation propre à EMTP dans le domaine du temps. Nous illustrons ceci pour la relation décrivant le comportement d'une inductance dans le domaine du temps. La dérivée en temps continu pour l'inductance L est

$$v_L(t) = L \frac{d}{dt} i(t), \quad (1.13)$$

où $v_L(t)$ et $i(t)$ sont la tension et le courant instantané aux bornes de l'inductance L .

La description similaire à celle d'inductance donne pour un bloc de contrôle « intégral » :

$$y(t) = \int_t^{t+\Delta t} u(t), \quad (1.14)$$

En considérant l'équation aux différences centrées pour l'inductance (1.13) [35]:

$$\frac{v_L(t) - v_L(t - \Delta t)}{2} = L \frac{i(t) - i(t - \Delta t)}{\Delta t}. \quad (1.15)$$

L'équation du modèle de l'inductance devient :

$$i(t) = \frac{\Delta t}{2L} [v_{L1}(t) - v_{L2}(t)] + \underbrace{i(t - \Delta t) + \frac{\Delta t}{2L} [v_{L1}(t - \Delta t) - v_{L2}(t - \Delta t)]}_{\text{historiques}}, \quad (1.16)$$

où v_{L1} et v_{L2} sont les tensions aux deux bouts de l'inductance L .

Encore une fois, de la même manière, on a pour le bloc « intégral », avec la règle trapézoïdale d'intégration, une version discrétisée pour (1.14) :

$$y(t) = \frac{\Delta t}{2} u(t) + \underbrace{y(t - \Delta t) + \frac{\Delta t}{2} u(t - \Delta t)}_{\text{historiques}}. \quad (1.17)$$

Les équations (1.16) et (1.17) s'apparentent à (1.10), où la variable t devient k et $t - \Delta t$ devient $k - 1$. Un bloc intégral dont le modèle est (1.17) aura une entrée et une sortie. Ce bloc pourrait avoir des connexions avec d'autres blocs via son entrée et via les sorties d'autres blocs. Il se crée une dépendance qui engendre un système simultané d'équations. Ce système simultané aura différentes fonctionnalités, selon le besoin. Dans la section suivante, quelques fonctionnalités sont illustrées : systèmes asservis, fonctions propres de l'utilisateur pour le calcul des énergies de parafoudres et comportement d'un arc électrique.

1.5 Quelques fonctionnalités de systèmes de contrôle

1.5.1 Modélisation des systèmes asservis

L'asservissement est un domaine d'où, par nature, intervient le système de contrôle. Le terme système de contrôle s'applique en effet grandement aux systèmes d'asservissement. D'une manière générique, il s'agit de faire une régulation pour que certaines variables choisies dans le système restent autour des références fixées d'avance. On l'illustre dans la Figure 1-11 pour l'excitatrice faisant partie d'un système de contrôle des machines. En général deux types de paramètres intervenant dans ce genre de système : les paramètres du modèle (système à asservir) et les paramètres de régulation. Dans ce cas-ci, les constantes T_a , T_b , T_c et les gains K_a et K_f appartiennent à l'une ou l'autre de ces catégories.

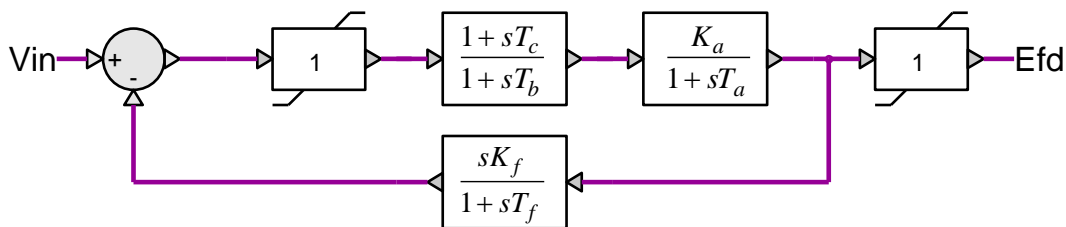


Figure 1-11: Portion du système de contrôle d'excitatrice de type EXST1 (IEEE)

1.5.2 Fonctions spécifiques par usager pour les mesures et traitements divers

D'innombrables fonctionnalités montées par usager sont possibles en contrôle. On ne peut pas prétendre en présenter ici une description générale. Elles servent, entre autres, comme outils d'analyse dans l'étude du réseau. Pour le besoin d'illustrer un exemple de modèle usager, mentionnons les fonctions pour calculer les énergies des parafoudres suite aux surtensions autour des transformateurs dans la Figure 1-2. En effet, les limites des énergies dissipées par ces derniers sont justement un critère de conception du réseau. Les fonctions calculant les grandeurs désirées peuvent être complexes. Dans ce cas-ci l'énergie dissipée E est fonction de la tension et du courant mesurés et aussi du temps à évaluer selon des standards bien définis :

$$E = f(v, i, t) \quad (1.18)$$

L'énergie E servirait à évaluer la limite du bon fonctionnement du parafoudre.

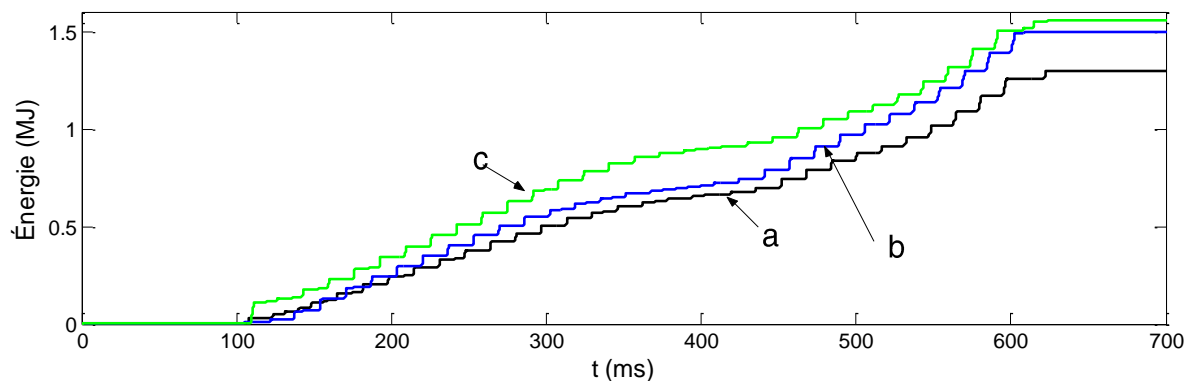


Figure 1-12: Énergies évaluées dans le parafoudre pour les trois phases (a, b, c) dans la Figure 1-2

1.5.3 Modélisation d'un processus ou phénomène d'arc de disjoncteur

Enfin, une dernière illustration montre la complexité de la programmation par usager. La limite où peut aller l'utilisateur n'est souvent pas suffisamment formellement indiquée. Un des modèles avec cet ordre de complexité est le cas des fonctions traduisant le modèle d'arc Cassie-Mayr dans le disjoncteur, évoqué plus haut. Ce dernier est un système d'équations différentielles qu'il faut intégrer et où intervient une boucle de rétroaction avec les variables courant I et tension U du réseau :

$$\frac{d}{dt}G_c = \frac{1}{\tau_c} \left[\frac{I^2}{U_c^2 G_c} - G_c \right], \quad (1.19)$$

$$\frac{d}{dt}G_m = \frac{1}{\tau_m} \left[\frac{I^2}{P_0} - G_m \right], \quad (1.20)$$

$$\frac{1}{G} = \frac{1}{G_m} + \frac{1}{G_c}, \quad (1.21)$$

avec :

G_c et G_m : conductances de l'arc (type Cassie (c) Mayr (m) respectivement), U : tension d'arc, I : courant d'arc, τ : constante de temps de l'arc, P_0 : puissance dissipée par le disjoncteur, U_0 : partie constante de la tension d'arc. Plus de détails sur ce modèle sont disponibles dans [76].

Une contrainte particulière des modèles de ce type est le fait qu'une de leurs variables dépend -ou est fonction- d'elle-même, de telle sorte que l'on ne peut pas écrire une fonction explicite de cette variable en fonction uniquement des autres variables. On est alors contraint d'écrire une formulation $f(x)=0$ pour cette variable particulière.

Ce type de fonction est une contrainte algébrique qui nécessite des itérations pour être résolue en considérant qu'elle peut être non-linéaire.

Ceci termine cette section introductive de la modélisation en ayant passé en revue, d'une manière très sommaire, quelques notions de base et quelques illustrations de fonctionnalité des systèmes de contrôle. Elle termine aussi ce chapitre introductif sur la généralité de la modélisation. Le chapitre suivant se concentrera sur la reformulation des groupes à rétroaction pour établir un système d'équations équivalent, mais à taille réduite.

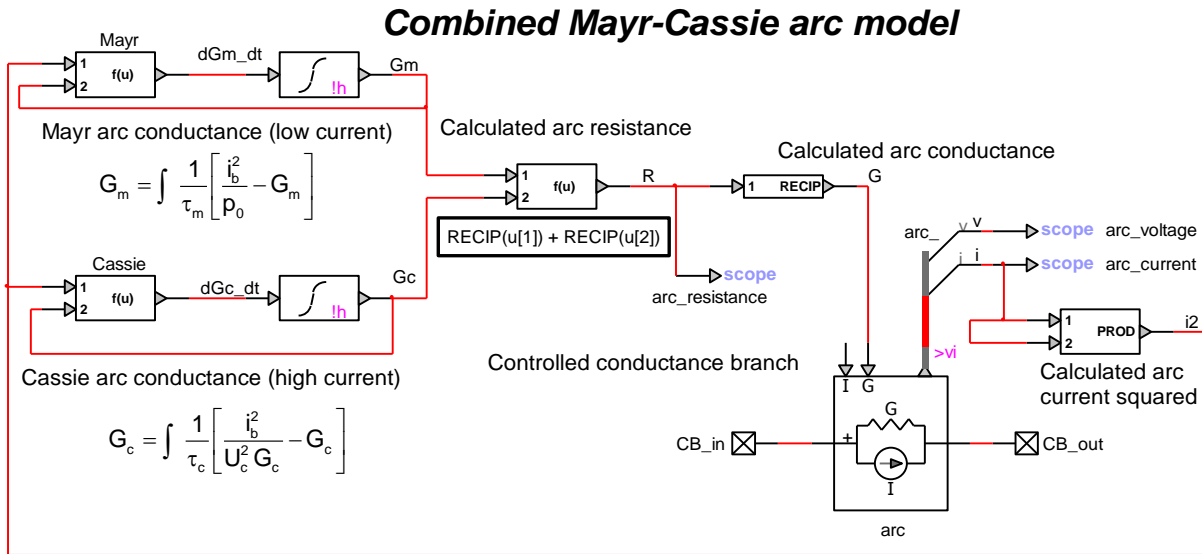


Figure 1-13: Diagramme du modèle d'arc Cassie-Mayr dans les exemples fournis dans [3]

1.6 Conclusion

Ce chapitre a présenté un aperçu général de la modélisation mathématique des systèmes de contrôle. Il a aussi aidé à replacer le système de contrôle dans le contexte beaucoup plus large de la simulation de réseaux électriques. Le fait qu'un système de contrôle ne modifie pas directement le couple courant-tension dans un modèle dynamique de réseau électrique est une de ces particularités. Les fonctions, traduisant le comportement dynamique de l'élément dans le système de contrôle, sont désignées par blocs de contrôle. Une catégorisation selon cinq critères a été adoptée pour décrire les blocs. Il s'agit du critère lié à la topologie (présence ou non d'entrée ou de sortie), du critère dynamique (blocs statiques ou blocs dynamiques avec termes d'histoire), du caractère continu ou discret, linéaire (blocs avec ou sans discontinuité, blocs non-linéaires et blocs de type « boîte-noire ») et, finalement, un critère lié au fait que les coefficients sont constants ou variables. Un tableau récapitulatif des types de bloc dans l'environnement EMTP-RV a été présenté. Les fonctions des blocs ont des interdépendances qui définissent aussi les connectivités des éléments. Ceci fait apparaître les regroupements de blocs qui peuvent être dédiés à une fonctionnalité particulière, telle un sous-système d'asservissement. Des groupes à action directe (« feedforward ») ou à rétroaction (« feedback ») découlent de ces interdépendances entre les fonctions des blocs.

Quelques fonctionnalités des systèmes de contrôle, parmi les milliers possibles, ont été présentées pour illustrer les applications de contrôle. Il s'agit de mesures et calculs variés, de systèmes d'asservissement et de modélisation d'arc de disjoncteur. À la lumière des quelques cas présentés, on peut conclure que la simulation de réseau pour les analyses d'intérêt devrait contenir des systèmes de contrôle. À mentionner, en terminant, que la solution des groupes à rétroaction est la plus difficile à calculer : elle fera l'objet de la suite de la thèse.

CHAPITRE 2 GRAPHE ET TRANSFORMATION AUX POINTS DE RÉTROACTION DES BOUCLES

Le but spécifique poursuivi dans ce chapitre est d'établir une nouvelle formulation des équations de contrôle résultant en une taille réduite. Celle-ci est établie dans la dernière section de ce chapitre. La procédure proposée est appliquée dans l'environnement EMTP. Les sections contenues dans ce chapitre sont : une introduction sur la représentation par graphe orienté, les propriétés du graphe (matrice d'adjacence et d'accessibilité (« reachability matrix »)), la technique de détection et suppression des cycles dans le graphe par inspection des valeurs propres de la matrice d'adjacence, la représentation des équations aux éléments de rétroaction des boucles et finalement, la décomposition avec interconnexion aux points de rétroaction.

2.1 Introduction

Comme décrit dans le chapitre précédent, le système de contrôle est modélisé par blocs connectés. Le diagramme bloc résultant peut être représenté comme un *graphe*. Étant donné que le signal circule dans une direction donnée, on a un *graphe orienté*. Les différents outils d'analyse de la théorie des graphes sont alors utilisés pour l'analyse topologique du système de contrôle. Les blocs seront considérés comme sommets du graphe et les connexions entre blocs seront les arcs du graphe.

Nous présumons que les prédécesseurs et successeurs de chacun des signaux ont été identifiés d'une manière ou d'une autre, en décodant les informations sur l'interface graphique et que nous pouvons rassembler les listes identifiant ces derniers. En effet, la plupart des outils de simulation génèrent des listes des prédécesseurs et successeurs des blocs [3][40].

En appliquant des techniques spécialisées de parcours de graphe, comme la « Depth-First Search technique (DFS) », sur les listes des prédécesseurs et successeurs, on obtient la liste topologique (LT) de la séquence des blocs de contrôle dans le sens de la circulation du signal. Cette séquence définit l'ordre d'évaluation des fonctions de sortie de chaque bloc. Les blocs *Constantes* et les blocs dont les sorties sont connues dans un temps donné n'ont pas de prédécesseur. Ils sont donc évalués en premier. Dans le processus de la DFS, on part du nœud qui

a le maximum des successeurs et récursivement, on visite chaque successeur et les successeurs de ce dernier (d'où le nom « recherche en profondeur ») jusqu'à ce que l'arbre soit complètement visité (nœud final) ; ou alors on fait une boucle pour revenir au nœud de départ : le nœud parent est revisité. Pour le nœud final, il n'y a plus d'enfant, c'est-à-dire aucun bloc dans la collection des successeurs et successeurs des successeurs. Si on détecte une boucle, le chemin parent-enfants-parent et la liste correspondante sont marqués. Plusieurs versions d'algorithmes de la DFS existent. Notons que les versions non-récurrentes offrent l'avantage de ne pas nécessiter des fonctions récursives sujettes à la méfiance de certains programmeurs informatiques à cause du risque de boucle infinie. Une bonne littérature sur ce sujet est disponible. On en fait mention ici pour illustrer conceptuellement l'analyse générique de la topologie des systèmes de contrôle. Parmi tant d'autres, on peut consulter [28] pour les notions de base ou approfondies sur le sujet. Un simple algorithme est présenté en annexe 1-2.

On va d'abord illustrer l'utilisation de la LT sur un cas concret par inspection. L'analyse sur des cas génériques sera faite ultérieurement.

Nous prenons le cas du système de contrôle d'une boucle à verrouillage de phase, mieux connu sous le terme anglais « Phase-Locked Loop » (PLL) présenté à la Figure 2-1. La LT peut être montée par inspection. Les premiers blocs de la liste sont le bloc source (entrée u en fonction cosinus) dont la valeur est connue à chaque nouveau pas de temps, ainsi que la valeur du bloc constante $C1$. La liste se complète par les autres blocs. Elle est présentée dans **F** :

$$\mathbf{F} = \{u; C1; \text{sum1}; P1; G2; G1; \text{Int1}, \text{sum2}; \text{sum3}; \text{Int2}; \text{SIN}; \text{COS}; P3; \text{Int3}; G3; P2\}. \quad (2.1)$$

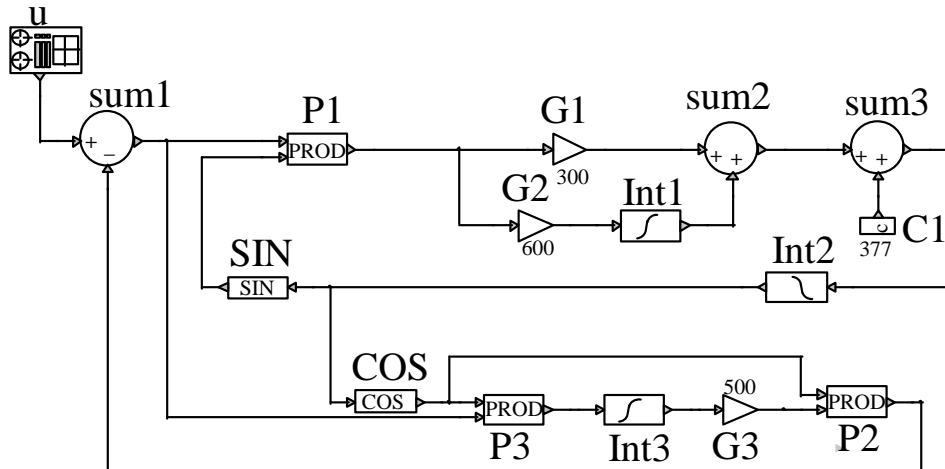


Figure 2-1: Système de contrôle d'une boucle à verrouillage de la phase "Phase-locked loop" (PLL) tiré de [1]

Le pictogramme du graphe orienté du PLL de la Figure 2-1 est présenté à la Figure 2-2.

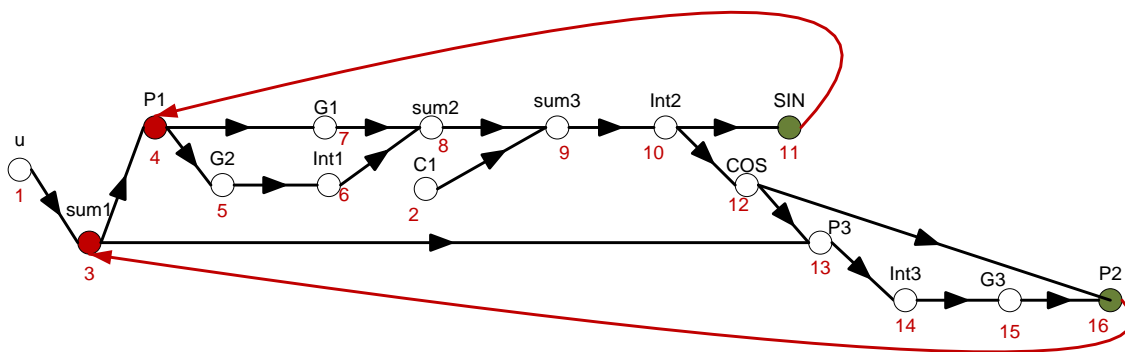


Figure 2-2: Pictogramme du graphe orienté du PLL de la Figure 2-1

On peut voir aisément que sum1 et P1, les fonctions d'indices troisième et quatrième respectivement, peuvent être considérés comme points de rétroaction : leurs entrées sont dépendantes des sorties via des chemins spécifiques à chacun. Par inspection, on a deux boucles et les listes des séquences **B1** et **B2** du chemin de rétroaction d'un point vers lui-même sont respectivement pour sum1 et P1 :

$$\mathbf{B1} = \{ G2; Int1; G1; sum2; sum3; Int2; SIN \}, \quad (2.2)$$

$$\mathbf{B2} = \{ P1; G2; Int1; G1; sum2; sum3; Int2; SIN; COS; P3; Int3; G3; P2 \}, \quad (2.3)$$

Les chemins de rétroaction mixtes entre P1 et sum1 à partir de la sortie du premier vers le second et vice-versa, sont respectivement :

$$\mathbf{B}_{P1 \rightarrow \text{sum1}} = \{G2; \text{Int1}; G1; \text{sum2}; \text{sum3}; \text{Int2}; \text{COS}; P3; \text{Int3}; G3; P2\}, \text{ et} \quad (2.4)$$

$$\mathbf{B}_{\text{sum1} \rightarrow P1} = \{1\}, \quad (2.5)$$

où $\{1\}$ dénote une fonction ou opérateur *identité* quand le signal est identiquement transféré entre deux points.

Pour ce simple cas, les séquences (2.2) à (2.5) ont été trouvées par inspection. Cependant, les outils de simulation doivent traiter des cas génériques représentant n'importe quelle topologie du système de contrôle. Pour cela, nous présentons les outils découlant de la théorie de graphe pour établir les listes de chemins de rétroaction. Nous reprenons le PLL et établissons les listes $\mathbf{B1}$, $\mathbf{B2}$, $\mathbf{B}_{P1 \rightarrow \text{sum1}}$ et $\mathbf{B}_{\text{sum1} \rightarrow P1}$ comme si c'était un cas générique.

Adoptant la notation de [23], le graphe orienté pour le PLL peut être représenté comme une paire (\mathbf{F}, \mathbf{L}) , où l'ensemble des fonctions \mathbf{F} dans (2.1) est modifié comme si on avait des fonctions génériques :

$$\mathbf{F} = \{f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, f_{15}, f_{16}\}, \quad (2.6)$$

où les indices de 1 à 16 représentent la séquence topologique des éléments, les fonctions de chacun des blocs sont identifiées dans la Figure 2-3.

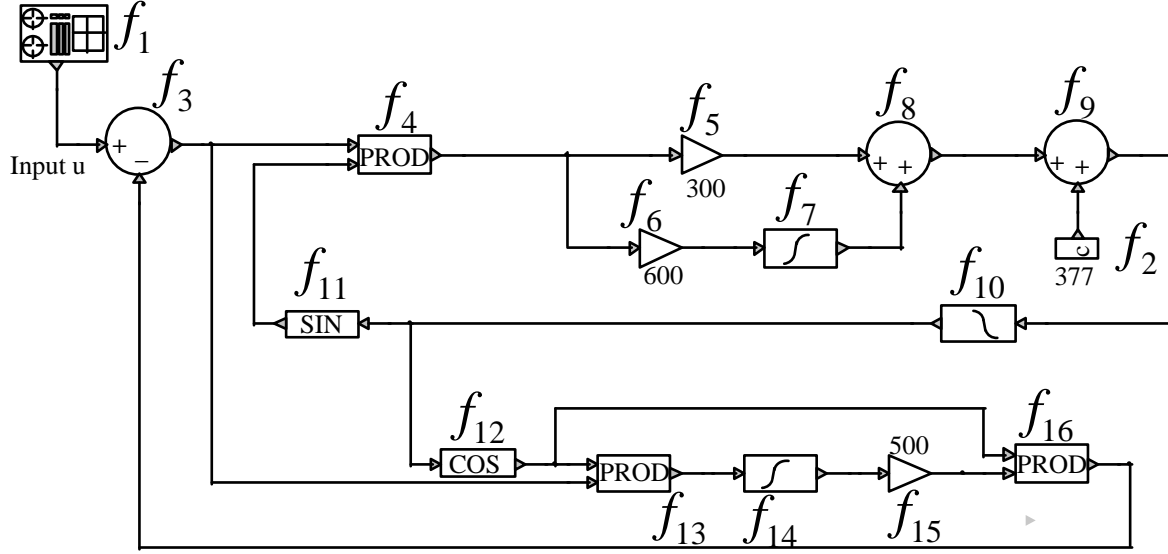


Figure 2-3: Schéma du PLL sous forme générique

\mathbf{F} est un ensemble fini des sommets ou nœuds dans le graphe et, comme indiqué, les arêtes sont les connexions entre deux sommets. Le sous-ensemble \mathbf{L} des paires d'éléments de \mathbf{F} , c'est-à-dire un ensemble $\mathbf{F} \times \mathbf{F}$ définissent une arête entre deux sommets.

Si la paire (f_i, f_j) est membre de \mathbf{L} , alors il y a une arête de f_i à f_j , c'est-à-dire que f_i est prédécesseur de f_j (ce qui est équivalent à f_j est le successeur de f_i).

Pour les seize fonctions des blocs du PLL, on a l'ensemble des arêtes :

$$\mathbf{L} = \left\{ \begin{aligned} &(f_1, f_3), (f_3, f_4), (f_4, f_5), (f_5, f_6), (f_4, f_7), (f_7, f_8), (f_6, f_8), \\ &(f_8, f_9), (f_2, f_9), (f_9, f_{10}), (f_{10}, f_{11}), (f_{10}, f_{12}), (f_{12}, f_{13}), (f_3, f_{13}), \\ &(f_{13}, f_{14}), (f_{14}, f_{15}), (f_{15}, f_{16}), (f_{12}, f_{16}), (f_{16}, f_3), (f_{11}, f_4) \end{aligned} \right\} \quad (2.7)$$

À la liste \mathbf{F} dans (2.6), qui répertorie tous les blocs de contrôle, et à liste \mathbf{L} dans (2.7) où on définit les paires des blocs connectés, on appliquera des techniques de décomposition des systèmes génériques de grande taille. Le but recherché est de le découper et de représenter un système réduit équivalent. La conformité de cette réduction sera validée et discutée dans les chapitres subséquents. On donne d'abord un rappel sur quelques définitions et propriétés des graphes sur lesquels s'appuieront les techniques de réduction.

2.2 Propriétés du graphe : matrice d'adjacence et d'accessibilité

Définition 1 : Étant donné un graphe orienté (F, L) , la matrice d'adjacence A est telle que ces éléments a_{ij} sont :

$$a_{ij} = \begin{cases} 1 & \text{si } (f_i, f_j) \in L \\ 0 & \text{si } (f_i, f_j) \notin L \end{cases}, \quad (2.8)$$

où L est la liste des paires de blocs connectés. Elle est présentée en (2.7)

Définition 2 : Étant donné un graphe orienté (F, L) , la matrice d'accessibilité R (pour « reachability ») spécifie si f_i est accessible à partir de f_j , on le notera $f_i R f_j$. On a alors les éléments r_{ij} , tels que :

$$r_{ij} = \begin{cases} 1 & \text{si } f_i R f_j \text{ ou si } i=j \\ 0 & \text{dans les autres cas} \end{cases}. \quad (2.9)$$

L'évaluation de la matrice d'adjacence A se fait aisément avec les opérations simples sur les listes, pour la matrice d'accessibilité R , on a, de [23] :

$$R = \left\{ \begin{array}{l} (I \oplus A) \otimes \\ (I \oplus A) \otimes \\ \vdots \\ (I \oplus A) \end{array} \right\} \begin{array}{l} \text{1ère} \\ \text{2ième} \\ \vdots \\ \text{S-ième opération } \otimes \end{array}, \quad (2.10)$$

où \oplus, \otimes , sont respectivement les opérations binaires OU et ET; I est la matrice d'identité de dimension $N \times N$ (N blocs dans le système); A est la matrice d'adjacence définie dans (2.8) et la S-ième opération ET est telle que $N-1 \leq S$. L'évaluation de (2.10) est montrée en annexe pour l'environnement Matlab voir la routine *mult_bool_mat(I, A)*.

Définition 3 : Étant donné un graphe orienté (F, L) , le chemin de longueur k de $f_i \in F$ à $f_j \in F$, est la séquence des sommets (c'est-à-dire, des opérateurs ou fonctions) $\{f_{i_0}, f_{i_1}, f_{i_2}, \dots, f_{i_k}\}$ avec $f_{i_0} = f_i$ et $f_{i_k} = f_j$ tel qu'il y a des arêtes successives de f_{i_p} à $f_{i_{p+1}}$ pour $p = 0, \dots, k-1$.

Définition 4 : Un graphe orienté (F, L) est fortement connecté si chaque paire de sommets (f_i, f_j) est fortement connectée, c'est-à-dire que pour chaque (f_i, f_j) dans le graphe, f_i est accessible de f_j ,

c'est-à-dire qu'il y a un chemin de f_i à f_j . Par ailleurs, si un système a des composants fortement connectés (CFC) qui sont distincts, alors il peut être découpé en des sous-systèmes. Cette analyse peut donc mener à définir des sous-systèmes de contrôle pouvant être calculés en parallèle car distincts et indépendants ; Ils peuvent aussi être calculés séquentiellement, si connectés à leurs bouts.

Notons qu'en pratique une matrice de connectivité \mathbf{Q} contient les CFC. Celle-ci est établie en mettant 1 pour chaque paire des sommets i et j qui sont fortement connectés. On a donc :

$$Q_{ij} = Q_{ji} = \begin{cases} 1 & \text{si la ligne } \mathbf{R}(i) \text{ est identique à la colonne } \mathbf{R}(j) \\ 0 & \text{dans les autres cas} \end{cases}, \quad (2.11)$$

La matrice d'accessibilité donne une indication sur la connectivité des blocs. Quelques manipulations sur la matrice nous donnent les chemins définis par les $\mathbf{B1}$, $\mathbf{B2}$, $\mathbf{B}_{P1 \rightarrow \text{sum1}}$ et $\mathbf{B}_{\text{sum1} \rightarrow P1}$ que nous voulons établir d'une manière générique. Cela revient à établir le chemin reliant deux blocs arbitraires d'indices i et j dans la collection des blocs comme en (2.6). Les notations suivantes sont à introduire :

- a. \mathbf{F}_I : les indices (entiers naturels) des fonctions dans \mathbf{F} , c'est-à-dire $\mathbf{F}_I = [1, 2, \dots, N]$;
- b. $\tilde{\mathbf{P}}_{ij} \equiv f_i \circ f_j$: Elle définit les indices du chemin de f_i vers f_j . C'est un sous-ensemble de \mathbf{F}_I ; et finalement,
- c. \mathbf{P}_{ij} : la liste des indices dans \mathbf{F} pour le chemin de i à j . C'est un sous-ensemble de \mathbf{F}_I référencé par $\tilde{\mathbf{P}}_{ij}$, tel qu'en notation MATLAB on a :

$$\mathbf{P}_{ij} = \mathbf{F}_I([\tilde{\mathbf{P}}_{ij}]). \quad (2.12)$$

Selon la définition de la matrice d'accessibilité, f_j et f_i sont accessibles de l'un vers l'autre si $\mathbf{R}(i,j)=1$;

Il est établi dans [23] que les indices des fonctions définissant le chemin entre les deux blocs sont :

$$\tilde{\mathbf{P}}_{ij} \equiv f_i \circ f_j = \text{Ligne } \mathbf{R}(i) \otimes \text{Colonne } \mathbf{R}(j)^T, \quad (2.13)$$

où \otimes est la multiplication binaire « ET » pour chacune des paires des éléments de la i -ième ligne dans la matrice \mathbf{R} et la j -ième colonne transposée de \mathbf{R} .

On achève en établissant les indices référencés dans la collection des fonctions \mathbf{F} . On a donc le sous-ensemble qui définit le chemin de i vers j noté $f_{\mathbf{p}_{ij}}$. Encore une fois, en notation MATLAB, on a :

$$f_{\mathbf{p}_{ij}} = \mathbf{F}([\mathbf{P}_{ij}]). \quad (2.14)$$

On utilisera dans cette thèse d'une manière similaire les notations suivantes pour la composition des fonctions : $f_{i,j}(\cdot) \equiv f_i \circ f_j(\cdot) \equiv f_i(f_j(\cdot))$ et pour la composition des compositions : $f_{i,\dots,j}(\cdot) \equiv f_{\mathbf{p}_{ij}}(\cdot)$.

Pour tout système de contrôle, une fois la matrice d'adjacence établie et celle d'accessibilité (comme, respectivement, dans (2.8) et (2.9) et en portant (2.13) dans (2.12) puis avec (2.14)), nous pouvons établir les listes $\mathbf{B1}$, $\mathbf{B2}$, $\mathbf{B}_{\mathbf{P1} \rightarrow \text{sum1}}$ et $\mathbf{B}_{\text{sum1} \rightarrow \mathbf{P1}}$ du PLL d'une manière générique. Ceci est illustré dans l'exemple ci-dessous.

2.3 Exemple d'analyse topologique générique pour le cas de PLL

Les matrices d'adjacence et d'accessibilité ont été établies à partir de (2.8) et (2.10). Elles sont illustrées par les points pour les valeurs non-nulles. Les valeurs non-nulles pour \mathbf{A} sont celles aux indices répertoriés dans \mathbf{L} indiquant les arêtes du graphe : $\mathbf{A}_{1,3}$; $\mathbf{A}_{3,4}$; $\mathbf{A}_{4,5}$; $\mathbf{A}_{5,6}$; $\mathbf{A}_{4,7}$; $\mathbf{A}_{7,8}$; $\mathbf{A}_{6,8}$; $\mathbf{A}_{8,9}$; $\mathbf{A}_{2,9}$; $\mathbf{A}_{9,10}$; $\mathbf{A}_{10,11}$; $\mathbf{A}_{10,12}$; $\mathbf{A}_{12,13}$; $\mathbf{A}_{3,13}$; $\mathbf{A}_{13,14}$; $\mathbf{A}_{14,15}$; $\mathbf{A}_{15,16}$; $\mathbf{A}_{12,16}$; $\mathbf{A}_{16,3}$; $\mathbf{A}_{11,4}$.

On voit, par exemple, que sur la Figure 2-4 le point (3,1) indique que le bloc 1 sur l'axe y est connecté au bloc 3 sur l'axe x .

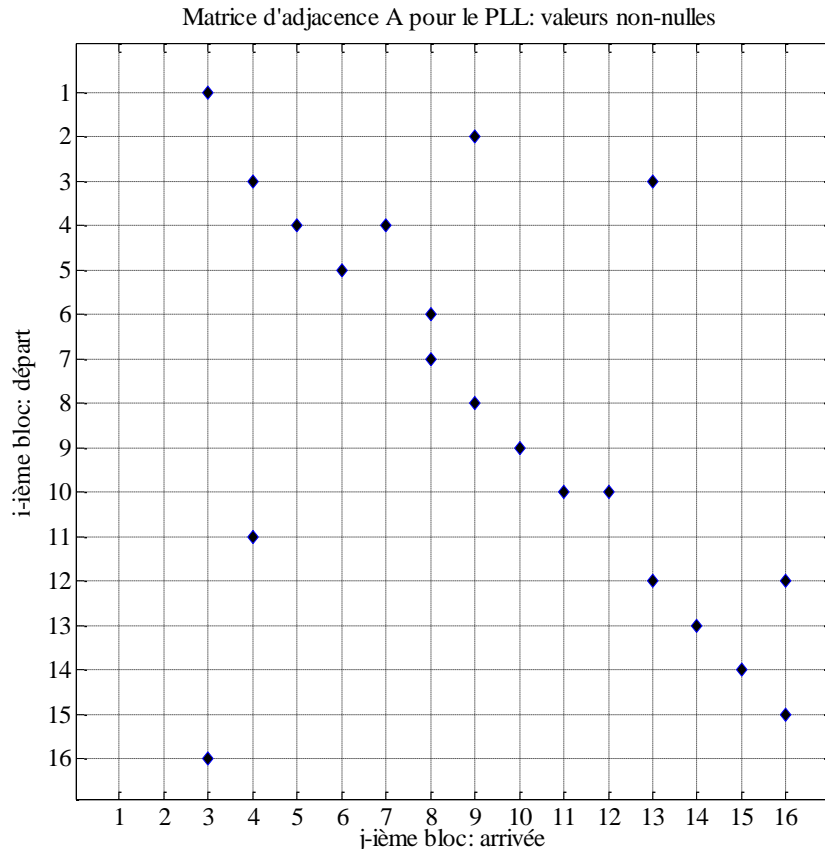


Figure 2-4: Représentation de la matrice d'adjacence pour le système de contrôle du PLL

Les calculs de la matrice d'accessibilité conduisent à la Figure 2-5 ci-dessous. On y voit, qu'à part les deux blocs 1 et 2, tous les autres blocs sont accessibles à partir de n'importe quel autre bloc. Par exemple, partant du bloc 1, on peut atteindre les blocs 3 à 16 car pour l'axe y correspondant à f1, on a des valeurs non-nulles pour y égales à 3 jusqu'à 16. L'inverse n'est pas vrai. Les blocs 1 et 2 entre eux ne sont pas accessibles car l'élément de coordonnée (1,2) (ou (2,1)) est nul.

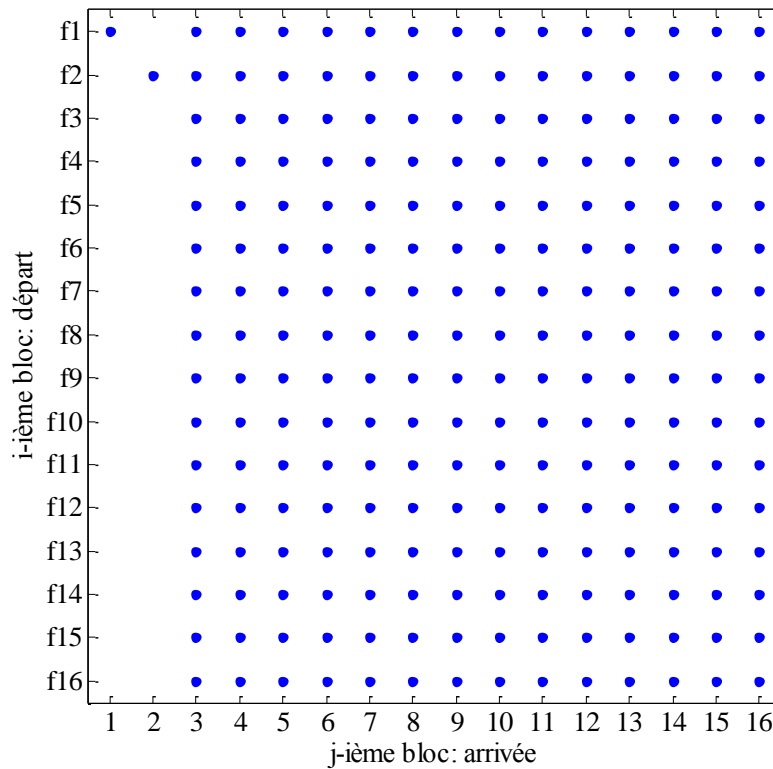


Figure 2-5: Représentation de la matrice d'accessibilité du PLL

La matrice de la connectivité établie par (2.11) donne les CFC (composantes fortement connectées) pour le PLL dans la Figure 2-6. On peut voir notamment que la partie inférieure gauche constitue un CFC. Il s'agit du sous-ensemble des blocs de f_3 à f_{16} . Ceci implique donc qu'une modification à l'entrée de n'importe quel bloc va influencer les autres blocs. Noter par ailleurs que, comme appréhendé, les blocs f_1 et f_2 ne sont accessibles à partir d'aucun autre bloc. Si, de plus, ces derniers sont connus durant le pas de temps de simulation, ils n'influenceront pas le calcul itératif de l'ensemble du système de contrôle. Cette brève description introduit déjà des aspects plus complexes des algorithmes de la solution de contrôle que nous présenterons dans les chapitres ultérieurs.

La section suivante présente une des contributions de cette thèse. Les boucles de rétroaction y sont isolées pour nous permettre de reformuler le système d'équation.

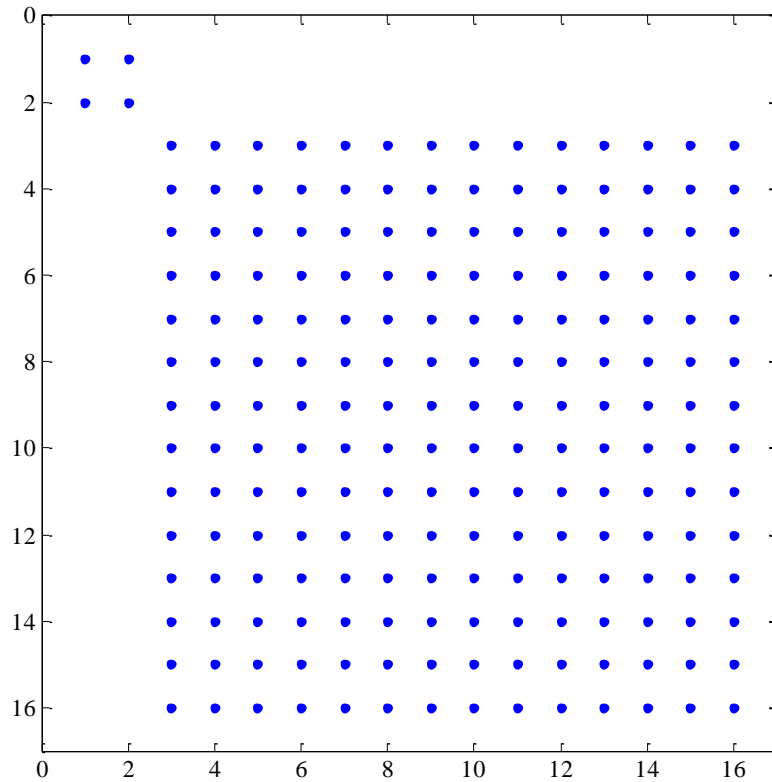


Figure 2-6: Représentation de la matrice de connectivité pour le PLL

2.4 Suppression des cycles dans le graphe : valeurs propres de la matrice d'adjacence

2.4.1 But de la suppression des cycles

La représentation du système de contrôle, en tenant compte de toutes les connectivités dans l'établissement des équations, génère le système d'équations de la taille correspondante au nombre de blocs formant le système. Nous voulons prendre une autre approche. Nous postulons qu'*une représentation, avec une taille réduite du système d'équation, est possible si on peut virtuellement supprimer les cycles dans le graphe, mais garder les contraintes associées aux points de découpage*. Nous montrerons l'équivalence entre la représentation réduite et le cas complet. Les avantages de cette représentation constituent une des contributions de cette thèse. Ils seront analysés dans les chapitres ultérieurs. Cependant, une technique de découpage est nécessaire pour supprimer tous les cycles dans le graphe (qui sont, pour rappel, les boucles de

rétroaction) dans le système de contrôle. Elle sera présentée dans la section suivante. Nous commencerons par introduire l'analyse des cycles dans le graphe du système de contrôle.

2.4.2 Analyse des boucles et cycles avec la matrice d'adjacence

La connectivité des blocs, et donc, la boucle de rétroaction sont détectables dans la matrice de connectivité. Si il n'y a aucune boucle alors le nombre des composantes fortement connectées est nul, on le notera par $\emptyset\text{CFC}$. Or, la connectivité est elle-même directement associée à la matrice d'accessibilité et donc, à la matrice d'adjacence (voir les relations (2.8) et (2.10), puis (2.11)). Celles-ci sont de type (1,0) : elles ne comportent que des zéros ou un (zéro quand il n'y a pas de connexion entre bloc et un, quand il y en a). Plusieurs outils d'analyse sont spécifiquement dédiés pour les matrices-(1,0). D'une manière explicite, on sait, par exemple, que la présence de cycle dans un graphe orienté est exprimée dans le corollaire que nous empruntons de [29][30] :

Il n'existe pas de cycle dans un graphe *si et seulement si* toutes les valeurs propres de la matrice d'adjacence sont nulles. On le note :

$$\emptyset\text{CFC} \Leftrightarrow \forall \text{ Valeurs propres de } \mathbf{A} \text{ sont nulles.} \quad (2.15)$$

Notons que s'il n'y a pas de cycle dans le graphe, selon la proposition 53 de [23], alors il n'y a *aucune* CFC (avec la notation plus haut : $\emptyset\text{CFC}$).

On peut à titre d'illustration vérifier la condition d'existence de boucle pour un cas simple d'un système de contrôle à une seule boucle de rétroaction. On considère le schéma d'un système de contrôle simple dans la Figure 2-7.

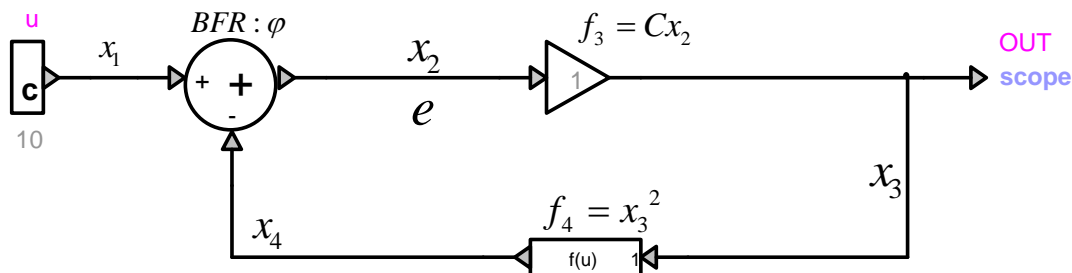


Figure 2-7: Schéma simple d'un diagramme bloc avec une rétroaction non-linéaire

Une inspection rapide dans la Figure 2-7 montre qu'il y a un cycle via la fonction de somme $BFR : \varphi$ (Bloc de la fonction de rétroaction φ). On transforme ce système (voir la Figure 2-8) en introduisant deux options de coupure dont la finalité sera présentée dans la section suivante.

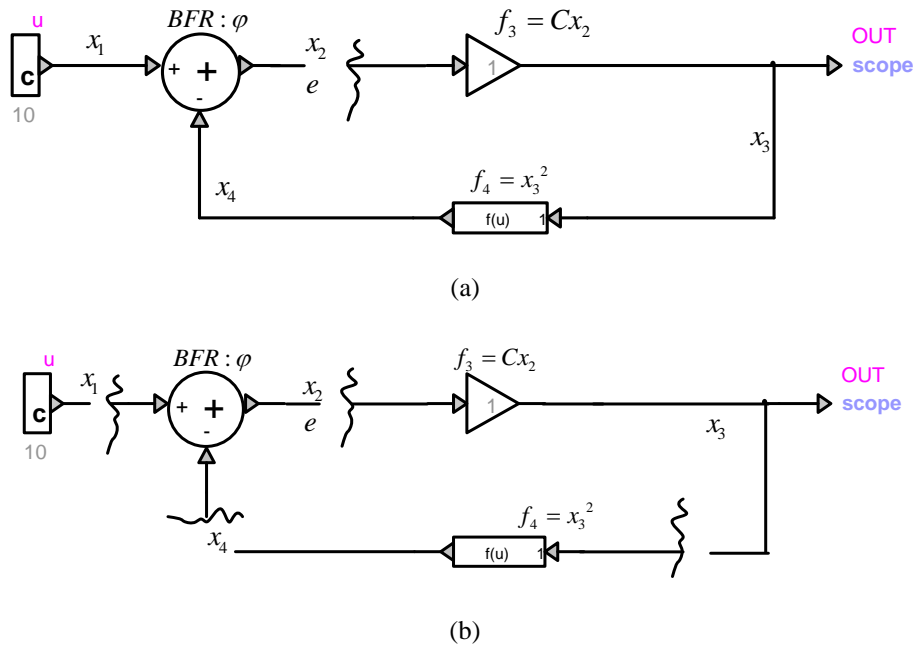


Figure 2-8: Introduction des coupures dans le schéma simple d'un diagramme bloc avec une rétroaction non-linéaire : (a) une seule coupure (b) plusieurs coupures, une après chaque sortie de bloc.

La coupure dans la Figure 2-8 (a) assure la suppression du cycle, tout comme les coupures à toutes les sorties de blocs dans la Figure 2-8 (b) suppriment évidemment le cycle. Les matrices d'adjacence, pour le cas de base et les deux cas coupés, sont respectivement, considérant la LT suivante :

$$\mathbf{F} = [u, \varphi, f_3, f_4], \quad (2.16)$$

$$\mathbf{A}_{Figure\ 2-8} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \mathbf{A}_{Figure\ 2-8(a)} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \mathbf{A}_{Figure\ 2-8(b)} = (\mathbf{0}), \text{ ainsi que leurs valeurs}$$

propres associées sont respectivement $[0, -0.5 \pm 0.866i, 1], [0, 0, 0, 0]$ et $[0, 0, 0, 0]$.

Cependant les deux dernières coupures ne donnent pas les mêmes avantages pour l'objectif exprimé auparavant quant à la réduction de la taille des équations. D'où les notations suivantes :

- a. EC : ensemble des coupures supprimant tout cycle dans le graphe ;
- b. mEC : ensemble minimal des coupures supprimant tout cycle dans le graphe ;
- c. MEC : ensemble maximal des coupures supprimant tout cycle dans le graphe ;
- d. oEC : ensemble optimal des coupures supprimant tout cycle dans le graphe ;

En effet, on a mentionné que le nombre de coupures assurant la suppression totale des boucles est ce qu'on recherche. Il est alors évident que l'idéal est d'obtenir le mEC. Cependant le coût associé à établir ce dernier peut être très grand en rapport avec le gain potentiel de la réduction du système d'équations. On optera alors pour oEC dans ce cas. L'établissement de la MEC, quant à elle, est d'intérêt pour la comparaison avec le cas de formation de système d'équations en considérant chacun des blocs. Car justement, le même nombre d'équations est formé, dans le cas MEC et dans le cas usuel de représentation complète du système en considérant les liens entre chaque bloc et tous les autres blocs. Ceci est utilisé dans la section 5.1.

La même analyse s'applique aussi sur le PLL. La matrice d'adjacence du graphe associé à Figure 2-3 donne 16 valeurs propres : $[0; 0; 1.2686; 0.6411 \pm 0.8630i; 0.1570 \pm 0.9191i; -0.5837 \pm 0.7605i; -0.8487 \pm 0.3652i; 0.0001; 0 \pm 0.0001i; -0.0001; 0]$. Les valeurs non-nulles sont présentes : en effet, il y a des cycles dans le graphe, notamment le chemin $f_4, f_5, f_7, f_8, f_9, f_{10}, f_{11}$ et encore f_4 . Différents ensembles de coupures assurent la non-présence de cycles. Intuitivement, les points de coupe sont ceux qui ont plus d'une entrée car susceptibles d'avoir un retour de boucle. Il s'agit entre autres d'ensembles suivants : EC_1 = [3,4,8,9,13,16], EC_2 = [3,4] et EC_3 = [4,16]. Trouver le mEC parmi ces EC exige une inspection approfondie. Par exemple, on peut voir que les blocs 4, 5, 6, 7, 8 et 9 appartiennent à un sous-groupe à action directe « *feedforward* » dont l'entrée est le 4 est la sortie est le bloc 9. On considère le bloc constant f2 comme étant

En général l'établissement d'oEC d'une manière générique n'est pas facile car il y a une infinité de possibilités. S'il y a M blocs candidats à la coupure, le nombre des tests est M -factorielle.

Tableau 2.1: Positions dans la LT pour les entrées des blocs candidats pour la coupure de la Figure 2-9

1. Blocs marqués dans la Figure 2-9 ayant plus d'une entrée	2. Indices de l'entrée correspondant dans la LT	3. Le bloc est-il bon candidat pour être un point de coupure : c'est-à-dire colonne 2 > colonne 1 dans ce tableau
3	1	OUI
	16	
4	3	OUI
	11	
8	6	NON
	7	
9	8	NON
	2	
13	12	NON
	3	
16	15	NON
	12	

La procédure pour établir l'oEC d'une manière générique est la suivante :

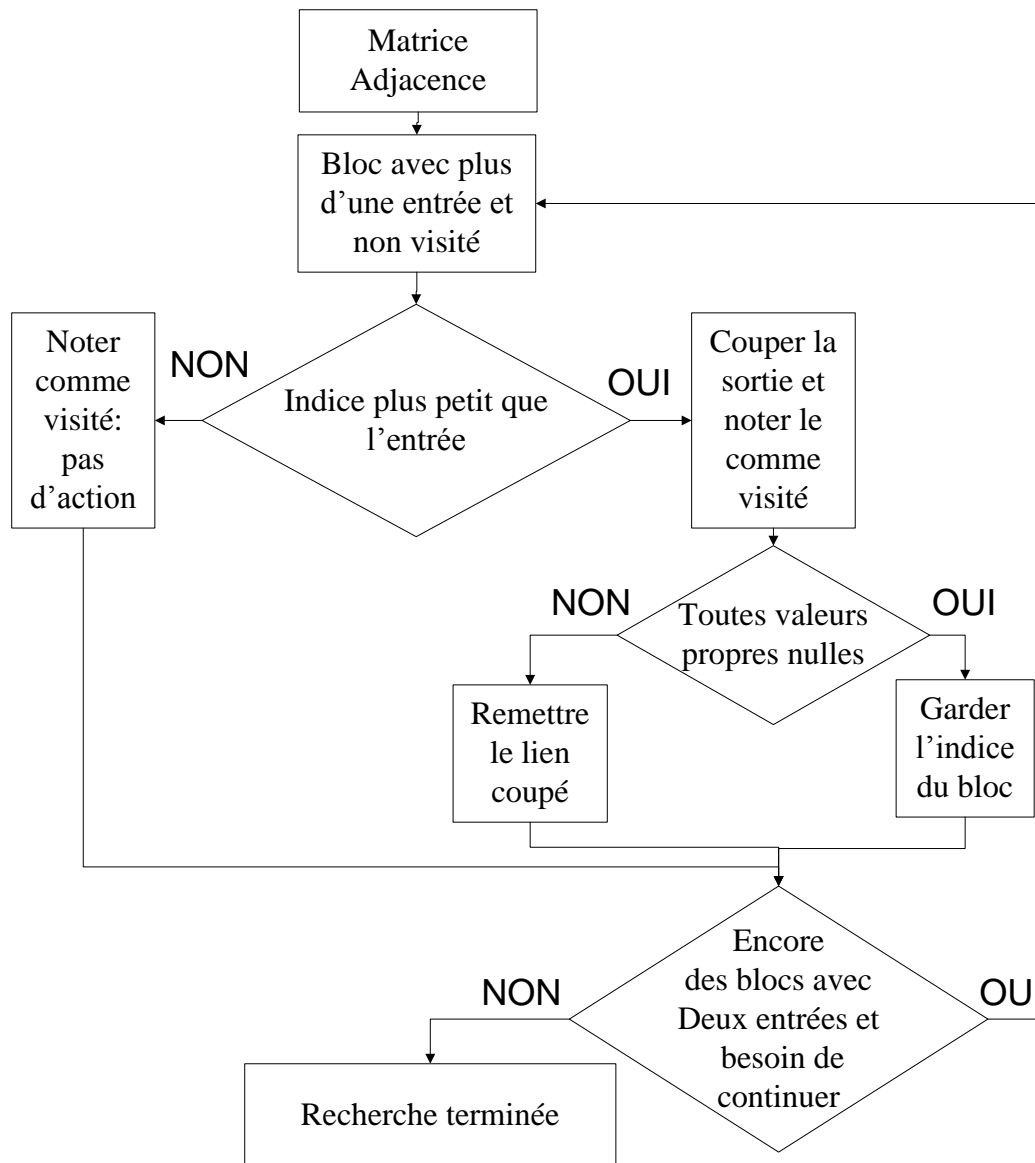


Figure 2-10: Procédure proposée pour établir l'ensemble optimal des coupures (oEC) garantissant l'absence de valeur propre non-nulle dans la matrice d'adjacence

On applique cet algorithme au PLL, où les positions dans la LT des blocs à deux entrées 3, 4, 8, 9, 13 et 16 sont présentées dans le Tableau 2.1. Les blocs f_3 et f_4 sont de bons candidats pour couper les rétroactions car les indices de leurs entrées (f_{16} et f_{11}) sont supérieurs à leurs propres indices 3 et 4. En coupant f_3 et f_4 il ne reste plus aucune valeur propre de la matrice qui ne soit non-nulle. Alors $\{f_3, f_4\}$ est l'oEC pour ce PLL. On montrera dans la section suivante les équations montées en considérant seulement ces deux points de rétroaction. Comme on l'a

postulé plus haut, la réduction de la taille du système d'équations sera alors de 14x14 à un système 2x2.

Les opérations de découpage virtuel introduites ci-haut sont en adéquation avec les définitions 1 à 4 dans la section 2.2. On montre ci-dessous les résultats à différentes étapes. Le découpage virtuel modifie la liste des blocs en introduisant au point de coupe trois blocs virtuels : bloc-mesure et deux blocs-source (voir, par exemple, pour la coupure aux fonctions 3 et 4). On y introduit : 3-1,3-2 et 3-3 et 4-1,4-2 et 4-3 pour respectivement sum1-a, sum1-b et sum1-c et P1-a, P1-b, P1-c. La nouvelle liste est alors (2.17) pour le diagramme en la Figure 2-11 :

$$\mathbf{F}_{\text{decoup}} = \left\{ \begin{array}{l} u; C1; \text{sum1}; \mathbf{\text{sum1} - b; \text{sum1} - c; P1; \mathbf{P1 - b; P1 - c;} \\ G2; G1; \text{Int1}; \text{sum2}; \text{sum3}; \text{Int2}; \text{SIN}; \mathbf{P1 - a; COS;} \\ P3; \text{Int3}; G3; P2; \mathbf{\text{sum1} - a;} \end{array} \right\}. \quad (2.17)$$

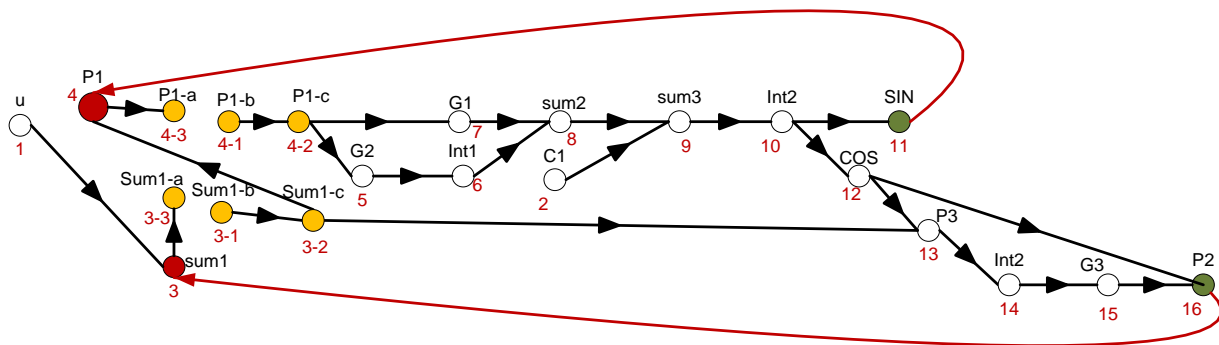


Figure 2-11: Découpage des boucles de rétroaction pour le PLL en introduisant d'autres nœuds à la fonction 3 : 3-1,3-2,3-3 et à la fonction 4 : 4-1,4-2, 4-3

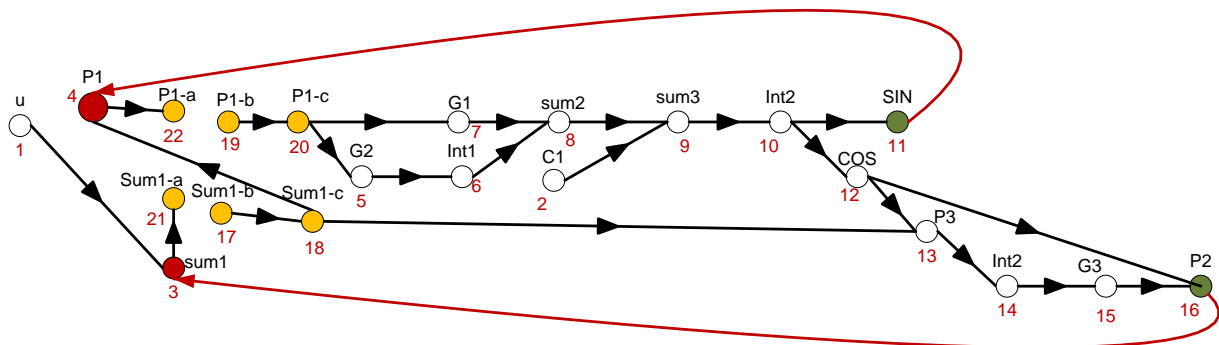


Figure 2-12: Ré-listage des fonctions pour une nouvelle LT

Les arêtes associées au graphe orienté sont alors :

$$\mathbf{L_decoup} = \left\{ \begin{array}{l} (f_1, f_3), (f_{17}, f_{18}), (f_{18}, f_4), (f_{19}, f_{20}), (f_{20}, f_5), (f_5, f_6), (f_{20}, f_7), (f_7, f_8), (f_6, f_8), \\ (f_8, f_9), (f_2, f_9), (f_9, f_{10}), (f_{10}, f_{11}), (f_{10}, f_{12}), (f_{12}, f_{13}), (f_3, f_{13}), (f_{13}, f_{14}), \\ (f_{14}, f_{15}), (f_{15}, f_{16}), (f_{12}, f_{16}), (f_{16}, f_3), (f_{11}, f_4), (f_3, f_{21}), (f_4, f_{22}) \end{array} \right\} \quad (2.18)$$

On en déduit les nouvelles matrices d'adjacence après découpage :

- En ouvrant au point de rétroaction à la fonction 3 et en connectant les blocs virtuels 3-1, 3-2 et 3-3 :

Ligne A(3) = 0 ;

A(17,18) = 1 ;

A(18,4) = 1 ;

A(18,13) = 1 ;

A (3,21) = 1 ;

- En ouvrant au point de rétroaction à la fonction 4 et en connectant les blocs virtuels 4-1, 4-2, 4-3 :

Ligne A(4) = 0 ;

A(19,20) = 1 ;

A(20,5) = 1 ;

A(20,7) = 1 ;

A (4,22) = 1 ;

- En n'assignant aucun suiveur pour les nœuds virtuels 22 et 21, (P1-a et sum1-a) :

Ligne A (22) = 0 ;

Ligne A (21) = 0.

Le graphe orienté modifié (sans cycle) donne la matrice d'accessibilité dans la Figure 2-13 d'où il est clair que certains blocs (nœuds) ne sont pas accessibles à partir d'autres blocs. La composante fortement connectée n'est plus, par ailleurs, disponible.

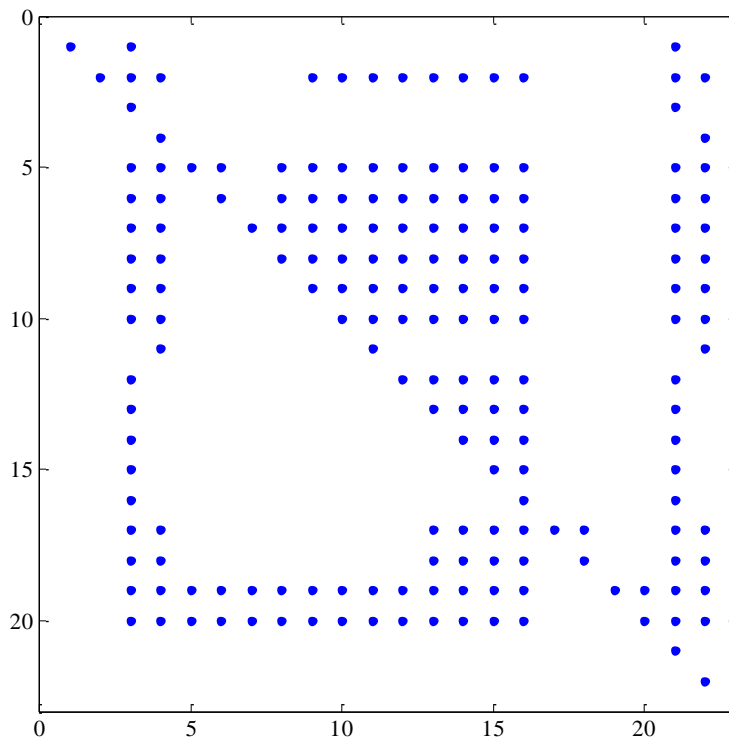


Figure 2-13: Représentation de la matrice d'accessibilité \mathbf{R} pour le diagramme de PLL modifié : points de rétroaction ouverts

Les chemins à partir desquels les sorties de points de rétroaction affectent les entrées de ces mêmes points génèrent l'*erreur*. Nous noterons la variable de sortie de blocs de rétroaction par “e” pour un seul bloc et par “ \mathbf{e} ” pour l'ensemble de points de rétroaction.

Dans le graphe du PLL dans la Figure 2-12, les sommets 17 et 19, les équivalents de l'oEC $\{3,4\}$ avant le découpage, et les sommets 21 et 22 sont dédiés à l'observation de la rétroaction, soit le vecteur d'erreurs \mathbf{e} .

On applique alors les calculs (2.13) et (2.14) pour établir les listes similaires à celles de (2.2) à (2.5). Sous forme compacte, on veut avoir :

$$\begin{bmatrix} \tilde{\mathbf{P}}_{17,21} \\ \tilde{\mathbf{P}}_{17,22} \\ \tilde{\mathbf{P}}_{19,21} \\ \tilde{\mathbf{P}}_{19,22} \end{bmatrix} = \left[\text{Ligne } \mathbf{R}([17 \ 19]) \otimes \text{colonne } \mathbf{R}([21 \ 22])^T \right], \quad (2.19)$$

où \mathbf{R} est la matrice d'accessibilité en la Figure 2-13.

$\mathbf{P}_{17,21} = [\dots, 13, 14, 15, 16, 3, 21]$ et $\mathbf{P}_{19,21} = [\dots, 13, 14, 15, 16, 3, 21]$. Ceci indique que le chemin de 19 à 21 et de 17 à 21 s'entrecroisent ou sont couplés. Ils doivent être combinés en conservant le bon ordre défini en (2.21). Les chemins finaux sont :

$$\begin{bmatrix} \mathbf{P}_{-,21} \\ \mathbf{P}_{17,22} \\ \mathbf{P}_{19,22} \end{bmatrix} = \begin{bmatrix} 19 & 20 & 5 & 6 & 7 & 8 & 9 & 10 & 12 & 17 & 18 & 13 & 14 & 15 & 16 & 3 & 21 \\ & & & & & & & & & 17 & 18 & 4 & 22 \\ 19 & 20 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 4 & 22 \end{bmatrix} \quad (2.23)$$

Ces chemins serviront à définir les fonctions modifiant les variables de la rétroaction \mathbf{e} . Ce qui est le but recherché dans la section suivante. Notez que les sommets virtuellement introduits ont les valeurs, soit du nœud 3 ou du nœud 4, selon les étapes de la solution. Ils découlent tous du nœud 3 ou 4. Finalement, dans la section qui suit, on présentera une procédure pratique du parcours de la LT pour avoir une équivalence des séquences définies dans (2.23). À ce stade, avec les chemins ou composition des fonctions donnant les entrées des points de rétroaction en fonction de leurs sorties, on peut maintenant présenter le système équivalent réduit que nous proposons.

2.5 Représentation des équations aux éléments de rétroaction des boucles

L'approche proposée est d'écrire seulement les équations liant les entrées des points de rétroaction en fonction de leurs sorties, si on a garanti préalablement que toutes les boucles aient été prises en compte. On l'a vu dans la section précédente, il a fallu s'assurer que toutes les valeurs propres de la matrice d'adjacence sont nulles pour le système avec les points de rétroaction coupés. Les sections précédentes ont servi à établir cela.

L'avantage que nous recherchons est de limiter le nombre d'équations au seul nombre des points de rétroaction. Revenons au cas simple du système de la Figure 2-7 pour illustrer cette nouvelle approche.

2.5.1 Aperçu de l'approche classique sur un cas simple de la Figure 2-7

Commençons par présenter l'approche classique où l'équation entrée-sortie de chacun des blocs est utilisée pour formuler le système d'équations. Par exemple, dans [4], la taille du système simultané d'équations est celle du nombre de blocs de contrôle. En effet, à moins d'utiliser certaines options de regroupement des blocs, similaire, à certains égards, à l'approche

proposée, tel que dans [40], il faut considérer tous les blocs dans la formulation de la solution simultanée.

Les conséquences de ceci sont que pour [4] par exemple, le processus de solution de Newton utilisant la Jacobienne souffre de la nécessité d'évaluer plusieurs équations et plusieurs fois quand il y a des blocs non-linéaires. Rappelons rapidement comment évolue un processus d'itération de type Newton sur le cas simple de la Figure 2-7 pour illustrer ceci.

Le système a quatre équations représentant les relations entre les réponses des blocs. Dans la forme usuelle on écrit :

$$\mathbf{a}\mathbf{x} = \mathbf{b}, \quad (2.24)$$

où le vecteur \mathbf{x} est le vecteur de variables (les sorties des blocs) ; \mathbf{a} contient les relations pour la partie linéaire du système et \mathbf{b} est le vecteur pour la relation NL et les valeurs connues dans l'itération. Cette itération de type Newton utilise une matrice Jacobienne \mathbf{J} de taille $N \times N$ ($N=4$ inconnus) à chaque pas de temps. Les pentes de toutes les fonctions NL des blocs (voir $\mathbf{J}_{4,3}$) sont calculées pour assembler la matrice Jacobienne à la k -ième itération. Rappelons la notation classique de la fonction objective $\Phi^{(k)} = \mathbf{a}^{(k)}\mathbf{x}^{(k)} - \mathbf{b}^{(k)} = 0$ et donc, chacun des éléments de la Jacobienne associée est $\mathbf{J}_{i,j}^{(k)} = \frac{\partial}{\partial x_j} \Phi^{(k)}_i$. On a la relation usuelle $\mathbf{J}^{(k)}(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = -\Phi^{(k)}$ du processus de Newton qui s'écrit :

$$\mathbf{J}^{(k)}(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = -\mathbf{a}^{(k)}\mathbf{x}^{(k)} + \mathbf{b}^{(k)}, \quad (2.25)$$

où \mathbf{J} , \mathbf{a} , \mathbf{x} and \mathbf{b} sont respectivement :

$$\mathbf{J}^{(k)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 1 \\ 0 & -C & 1 & 0 \\ 0 & 0 & -\frac{\partial}{\partial x_3} x_3^2 & 1 \end{pmatrix}^{(k)}, \mathbf{a}^{(k)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & -1 \\ 0 & -C & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad (2.26)$$

$$\mathbf{x} = [x_1 \ x_2 \ x_3 \ x_4]^T, \text{ et } \mathbf{b} = [u \ 0 \ 0 \ x_3^2]^T.$$

Ces pentes des fonctions NL (ici une seule : $f_4(x_3) = x_3^2$), doivent être évaluées avec la perturbation numérique au moins à chaque nouveau temps, pire, à chaque itération dans le même

temps. En général, on aura pour n blocs NL n perturbations numériques. Si, par exemple, on a des boîtes-noires pour tous les blocs, tous les éléments de la Jacobienne doivent être évalués par la perturbation numérique. Ces modifications vont exiger, durant le processus itératif, une nouvelle factorisation pour calculer (2.25) (pour rappel, les inversions directes des matrices sont déconseillées).

Il est clair qu'en ajoutant un bloc dans la boucle on accroît la taille du système. Pire, si en plus il s'agit d'un nouveau bloc NL, on accroît la taille du système d'équations ainsi que l'exigence supplémentaire de mise à jour de pentes dans \mathbf{J} par une perturbation numérique. Dans le contexte de l'augmentation de la complexité des systèmes de contrôle et de la sophistication des capacités de modélisation par usager (utilisation des boîtes-noires ou de DLL par exemple), la solution utilisant la représentation des équations du système avec tous les blocs peut s'avérer coûteux en temps de calcul dans la simulation du réseau électrique. Ceci a été rapporté, entre autres, mais seulement dans une audience restreinte d'experts, en modélisation et simulation des éoliennes pour certains utilitaires et manufacturiers. La nouvelle approche dans la sous-section suivante présente une approche différente.

2.5.2 Aperçu de la nouvelle approche sur le même cas simple de la Figure 2-7

La contribution de cette thèse est de proposer une solution simultanée des systèmes de contrôle en considérant seulement les points de rétroaction au lieu de considérer tous les blocs de contrôle. La taille du système est celle de l'ensemble optimal de coupures (oEC) à former en rendant le graphe représentant le système sans cycle (boucle).

Par exemple, dans la Figure 2-7, une seule coupure au point de rétroaction donne une seule équation associée à cette coupure à la fonction φ . Ainsi, on réduit le système de taille 4×4 . Par ailleurs, au lieu de représenter tout le système par une matrice de taille toujours plus grande selon l'ajout de blocs, comme le système est un graphe orienté, il suffit de garder les listes séquentielles topologiques et celles pour différentes opérations pour établir la Jacobienne et les différentes réponses des blocs. D'autres intérêts sont aussi associés à la possibilité d'analyse de chemin de rétroaction pour l'étude de la stabilité de l'opération d'un contrôleur ou encore, pour les propriétés associées à la formulation de la méthode des itérations à point-fixe. Ceci sera montré plus loin. La formulation de la nouvelle approche de solution peut être illustrée en la Figure 2-14 pour le cas simple de la Figure 2-7.

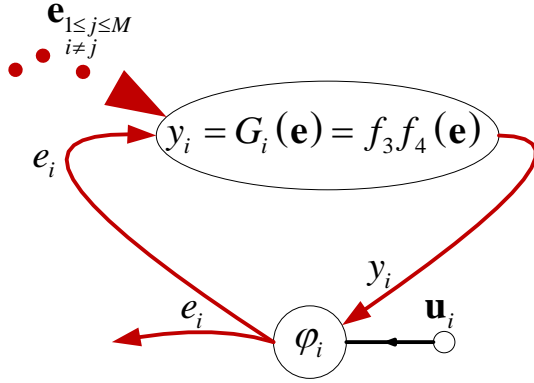


Figure 2-14: Connexion par rétroaction : point de jonction pour la Figure 2-7,

Le système de contrôle est décomposé aux points de rétroaction, ici la fonction Somme φ . Ces points connectent les entrées \mathbf{u} et les signaux d'interaction venant des sorties des fonctions associées aux points de rétroaction \mathbf{e} . Ces signaux d'interaction sont justement les fonctions composées du chemin de rétroaction sur \mathbf{e} , comme illustré dans la Figure 2-14, où nous avons adopté les notations suivantes :

φ : Fonction du bloc au point jonction de rétroaction ;

f : Fonction d'un bloc quelconque (générique) ;

\mathbf{u} : Vecteur des entrées externes dans les boucles du système ;

\mathbf{e} : Vecteur des sorties aux blocs point-jonction de rétroaction ;

\mathbf{y} : Vecteurs des signaux d'interaction des fonctions composées sur le chemin de rétroaction ;

G : Fonction composée (ou opérateur) pour les signaux d'interaction. Ses entrées sont les sorties \mathbf{e} (pour toutes les fonctions de rétroaction φ), en d'autres mots, c'est le chemin de rétroaction.

Dans la Figure 2-7, on a un seul point de rétroaction : $i = 1$ dans la Figure 2-14 et donc les autres $\mathbf{e}_{1 \leq j \leq M, i \neq j} = 0$. On l'a introduit pour anticiper le cas général où il y a plusieurs points de rétroaction et donc plusieurs chemins, c'est-à-dire $\mathbf{e} = [e_1, \dots, e_M]$ pour M boucles de rétroaction. Le découpage virtuel et la contrainte associée au point de rétroaction donne :

$$e = \varphi(u, y) = u - f_4(f_3(e)), \quad (2.27)$$

où u et e sont respectivement l'entrée et la sortie du bloc φ et où la fonction composée $f_4(f_3(e)) = G(e) = y$ est le chemin de rétroaction.

La solution est d'égaliser les deux membres de (2.27), c'est-à-dire :

$$\Phi = e - [u - f_4(f_3(e))] = 0. \quad (2.28)$$

Ou, pour revenir aux variables x ($e \equiv x_2$), (2.28) s'écrit aussi :

$$\Phi = x_2 - [u - (Cx_2)^2] = 0 \quad (2.29)$$

L'inconnu dans cette équation est unique, à savoir x_2 , au lieu de toutes les autres sorties comme dans le cas classique présenté plus haut, soit $\mathbf{x} = [x_1 \ x_2 \ x_3 \ x_4]^T$ dans (2.26). Le processus itératif de Newton s'en trouve aussi allégé. La k -ième itération pour cette fonction NL est :

$$J^{(k)} \Delta x_2^{(k+1)} = (-\Phi^{(k)}), \quad (2.30)$$

où Φ est la fonction définie dans (2.29). La Jacobienne est trouvée par perturbation numérique par une valeur très petite δ_{x_2} tel que :

$$J^{(k)} = \frac{1}{\delta_{x_2}^{(k)}} \left[\Phi(x_2^{(k)} + \delta_{x_2}^{(k)}) - \Phi(x_2^{(k)}) \right]. \quad (2.31)$$

Nous présumerons une Jacobienne établie par perturbation numérique pour tous les cas dans toute cette thèse. Une discussion sur la perturbation est présentée dans le Chapitre 3.

Une fois la solution du processus itératif de Newton trouvée pour la sortie x_2 (qui devient donc connue, comme les constantes et les entrées au temps considéré), d'autres variables sont évaluées séquentiellement selon la séquence appropriée, ce qui est pour ce cas, selon (2.16) :

$$(x_1, x_2) \rightarrow x_3 = f_3(x_2) \rightarrow x_4 = f_4(x_3). \quad (2.32)$$

On voit donc qu'avec cette approche, le nombre d'équations est égal au nombre de points de rétroaction, peu importe le nombre de blocs dans le chemin de la boucle. De plus, pour former la Jacobienne, on a à faire la perturbation des sorties de blocs aux points de rétroaction seulement. Mentionnons qu'on suppose que les blocs dans le chemin se comportent bien, en donnant une réponse appropriée pour une entrée appropriée. Une itération supplémentaire est

introduite si on atteint les limites d'un bloc limiteur, ou bloc saturation, ou au croisement de zéro dans un bloc de détection.

Nous avons postulé que cette nouvelle approche apporte des avantages, notamment dans la ré-factorisation de la solution et pour assembler la matrice Jacobienne par perturbation numérique. Dans l'approche classique, chaque bloc NL est individuellement perturbé. Le gain maximum avec cette nouvelle approche apparaît donc quand un système a un nombre maximum de blocs NL et un nombre réduit des points de rétroaction (le plus petit oEC). De plus, avec précaution on peut traiter n'importe quelle combinaison de fonctions composées dans le chemin de la boucle. Finalement, les propriétés, au mieux contractantes, pouvant découler sur le chemin de rétroaction sous forme d'un problème d'itérations à point-fixe, peuvent être analysées, et, le cas échéant prendre avantage de ces propriétés améliorant les dites itérations. En effet, cette formulation rend plus aisé l'isolation du chemin de boucle générique. Certaines classes des systèmes de contrôle pourraient être indiquées pour être résolues par la méthode point-fixe. Nous présenterons une variante de cette approche dans le chapitre suivant. Pour construire cela, on introduira une décomposition avec interconnexion aux points de rétroaction (DIPR) comme celle de la Figure 2-14 pour visualiser, entre autres, les propriétés du problème à point-fixe.

Cependant, certaines limites subsistent, spécialement, le coût de l'évaluation séquentielle des blocs. De plus, les chemins de boucle de rétroaction peuvent être fortement NLs. Ceci impose une procédure robuste pour traiter ce genre des fonctions. La méthode de Newton de base peut montrer des limites pour converger. D'autres propriétés sont supposées, notamment que la représentation est telle que le système est *bien posé* (« well-posed »). Plus précisément, on suppose que pour un ensemble d'entrées, les solutions pour les sorties aux points de rétroaction existent et sont uniques.

Les conditions de « well-posedness » sont largement discutées dans [12][21] [23] et [41], et en particulier la formulation de la décomposition avec interconnexion aux points de rétroaction (« general feedback interconnection »), ce qui est précisément l'approche adoptée ici. Voir entre autres la Figure 2-14. On présente maintenant le cas général dans la section suivante en se basant sur ce cas simple d'une seule coupure.

2.6 Décomposition avec interconnexion aux points de rétroaction (DIPR)

La représentation du système de contrôle par DIPR est une approche appliquée principalement pour la caractérisation des propriétés de stabilité du contrôle. On s'y réfère en anglais par le terme « general feedback interconnection ». Sous la forme la plus connue dans le domaine de contrôle, la DIPR est montrée à la Figure 2-15, mais avec deux entrées. Une manipulation sera nécessaire pour avoir exactement la formulation de la DIPR que nous proposons.

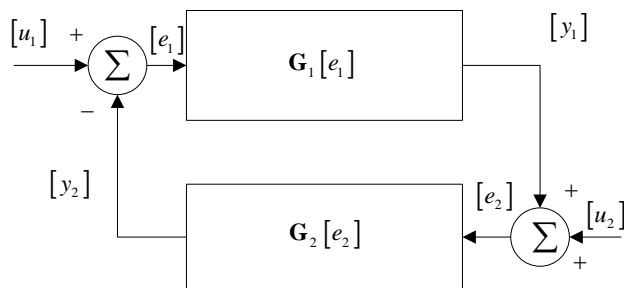


Figure 2-15: Représentation classique du « general feedback interconnection » comparable à la DIPR proposée

La DIPR est appliquée sur le système découpé virtuellement. On rappelle ici que ce découpage virtuel n'introduit aucune modification du modèle.

On choisissant M coupes de l'oEC garantissant aucun cycle dans le graphe du contrôle on a la généralisation de la Figure 2-14 illustrée dans la Figure 2-16.

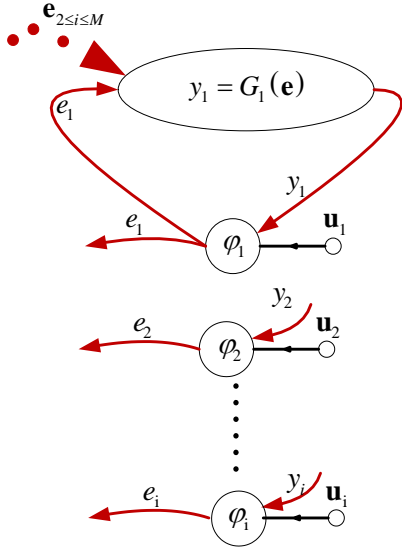


Figure 2-16: Décomposition aux points de rétroaction : M coupes

La représentation du DIPR en Figure 2-16 est équivalente à celle du cas classique de la Figure 2-15 en contractant les opérateurs G_1 et G_2 et en ne considérant qu'une seule entrée. Les éléments d'analyse sur la Figure 2-15 rapportés par [11][12][21][23] et [41] peuvent s'appliquer sur la DIPR de la Figure 2-16 .

Le système d'équations simultanées similaires à (2.27) sont maintenant à plusieurs variables, soient $i = 1 \dots M$ points de rétroaction (PR) :

$$\begin{cases} \mathbf{e} = \boldsymbol{\varphi}(\mathbf{U}, \mathbf{y}) \\ \mathbf{y} = \mathbf{G}(\mathbf{e}) \end{cases}, \quad (2.33)$$

La fonction composée pour la i -ième boucle de rétroaction est G_i :

$$G_i(e) = \psi_i \circ f_{p1} \circ f_{p2} \dots (\mathbf{e}), \quad (2.34)$$

où ψ_i dans (2.34) est une fonction (possiblement NL) qui est introduite quand la jonction n'est pas un bloc de sommation (voir, par exemple le bloc produit f_4 dans le PLL en la Figure 2-3). Dans les autres cas, ψ_i est une fonction identité comme le cas de la somme (f_3) dans la même Figure 2-3. Cette astuce ne sert en réalité qu'à montrer conceptuellement l'équivalence avec la Figure 2-15. Les calculs n'en sont aucunement modifiés. Noter aussi qu'une des entrées u_i et nulle comme il sera exprimé dans (2.37).

Par ailleurs \mathbf{p} , sur lequel on a omis, par abus de notation, le sous-indice relatif à i dans (2.34), est l'ensemble de la collection des indices des fonctions dans la liste topologique qui forment les chemins de toutes les sorties des blocs aux PR vers le i -ième PR. Si $f_3f_4f_5$ sont les trois fonctions sur le chemin i , on aura $\mathbf{p}_i=[3,4,5]$. On a donc, selon la notation de (2.34) (avec l'abus mentionné), $\mathbf{p}_1=3$, $\mathbf{p}_2=4$ et $\mathbf{p}_3=5$.

Nous avons montré comment établir les séquences de chemins de retour utilisant les opérations binaires sur la matrice d'adjacence et d'accessibilité. Pour les conformer à G_i , il suffit maintenant de combiner les indices comme on l'a fait en établissant (2.23) par rassemblement des indices de (2.22). En fait, on combine tous les chemins de boucle de rétroaction aboutissant sur un PR. Par exemple, sur le PLL, les résultats en (2.23) sont modifiés pour combiner les deux derniers ensembles d'indices qui aboutissent vers le nœud virtuel 22 ceci donne comme pour le nœud 21 :

$$\begin{bmatrix} \mathbf{P}_{-,21} \\ \mathbf{P}_{-,22} \end{bmatrix} = \begin{bmatrix} 19 & 20 & 5 & 6 & 7 & 8 & 9 & 10 & 12 & 17 & 18 & 13 & 14 & 15 & 16 & 3 & 21 \\ & 19 & 20 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 17 & 18 & 4 & 22 \end{bmatrix}, \quad (2.35)$$

où on a tenu compte, encore une fois, de la séquence en (2.21) :

$$\mathbf{F}_s = [1 \ 2 \ 19 \ 20 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10 \ 12 \ 11 \ 17 \ 18 \ 4 \ 22 \ 13 \ 14 \ 15 \ 16 \ 3 \ 21]$$

Les indices deviennent maintenant en omettant les nœuds virtuellement introduits (17, 18, 19, 20, 21 et 22) et les fonctions correspondantes aux blocs PR 3 et 4 sont aussi omises si elles ne sont de sommation. Ainsi, la fonction bloc PR 4 est remplacée par $\psi_2 \equiv \text{PROD}$, comme annoncé dans (2.34) :

$$\begin{cases} G_1(\mathbf{e}) = f_{\mathbf{p}_1}(\mathbf{e}) = f_{16,15,14,13,12,10,9,8,7,6,5}(\mathbf{e}) \\ G_2(\mathbf{e}) = f_{\mathbf{p}_2}(\mathbf{e}) = \psi_2(f_{11,10,9,8,7,6,5}(\mathbf{e})) \end{cases}, \quad (2.36)$$

où les indices "1" et "2" dénotent les blocs PR (somme f_3 et produit f_4).

Remarquons que les indices \mathbf{p}_i incluent celui associé à ψ_i . Cette astuce, bien qu'alourdisant la notation, nous sert, comme mentionné, à se formaliser avec la représentation classique en Figure 2-15. En réalité, la fonction composée résultante de la rétroaction inclut toutes les fonctions composées pour avoir $\varphi_i(\mathbf{u}, G_i(\mathbf{e}))$ pour tous points de rétroaction.

La DIPR sur le PLL à résoudre est (l'évaluation séquentielle des autres variables suivra) :

$$\begin{cases} e_1 = u - G_1(\mathbf{e}) \\ e_2 = 0 + G_2(\mathbf{e}) \end{cases}, \quad (2.37)$$

où les fonctions G_1 et G_2 sont dans (2.36). Par exemple, l'évaluation de la fonction composée $G_2(e_1, e_2)$ dans (2.37) conduit à l'expression analytique suivante (qui est en réalité discrétisée pour l'intégration comme évoqué dans la section 1.2:

$$G_2 = PROD\left(e_1, \sin\left\{\int\left[f_2 + \left(f_5(e_2) + \int f_6(e_2)dt\right)\right]dt\right\}\right), \quad (2.38)$$

où f_2 est une constante et f_5 et f_6 sont des gains.

D'une manière générique, on peut maintenant écrire le système d'équations simultanées multi-variable (de dimension M) dans la formulation DIPR que nous proposons :

$$\Phi_i = e_i - \varphi_i(\mathbf{u}, G_i(\mathbf{e})) = 0, \quad i = 1 \dots M, \quad (2.39)$$

ou de manière condensée :

$$\Phi = \mathbf{e} - \boldsymbol{\varphi}(\mathbf{U}, \mathbf{G}(\mathbf{e})) = 0, \quad (2.40)$$

où M est le nombre des coupures dans l'oEC ayant garanti la suppression de toutes les boucles de rétroaction. Les fonctions dans \mathbf{G} sont les chemins de rétroaction arrivant à l'entrée de chaque PR partant des sorties de toutes les PR et \mathbf{e} , \mathbf{U} et $\boldsymbol{\varphi}$ sont respectivement les vecteurs de sorties des blocs aux PR, les entrées externes et les fonctions de bloc aux PR. La fonction Φ_i est la fonction à minimiser. Les méthodes de solution classiques seront appliquées au système (2.39) dans le Chapitre 3. Notons déjà comment se dessine la formulation du problème à point-fixe sur la variable \mathbf{e} dans (2.40), en réécrivant cette dernière comme :

$$\mathbf{e} = \boldsymbol{\varphi}(\mathbf{U}, \mathbf{G}(\mathbf{e})). \quad (2.41)$$

On a mentionné que les propriétés de « well-posedness » du système en (2.39) ne seront pas approfondies. On a supposé en effet que le système est bien posé selon les conditions et l'analyse discutées par [12][21][23] et [41] sur le système similaire au nôtre, en la Figure 2-15. Cette hypothèse implique que tous les blocs « se comportent bien » (en somme, ils retournent une « bonne » sortie pour une entrée « raisonnable ») et plus important encore, les dérivées sont

disponibles pour chaque bloc. Les points de discontinuité seront donc traités séparément. Ces questions et celles d'existence et d'unicité de la solution sont néanmoins cruciales pour tout système, surtout non-linéaire. Nous n'en discuterons pas plus dans cette thèse.

2.7 Conclusion

Le but recherché dans ce chapitre a été d'établir une nouvelle formulation résultant en une taille réduite des équations de contrôle en utilisant les outils de la théorie de graphe.

En effet, le système de contrôle a été représenté par son équivalent en graphe. Étant donné le sens unique de circulation du signal de contrôle, un graphe orienté a été introduit. La modification du graphe s'appuie, entre autres, sur la connaissance de l'ordre topologique de succession des blocs. Pour établir cet ordre, une technique de parcours est nécessaire. Celle qui a été utilisée dans cette thèse, actuellement fonctionnelle dans les algorithmes du logiciel EMTP-RV, est celle de la recherche en profondeur DFS (« Depth-First-Search »).

Une brève introduction portant sur la matrice d'adjacence et d'accessibilité « reachability matrix » et sur la matrice de connectivité de la représentation des systèmes de contrôle par graphe orienté a été présentée.

Une technique de détection des cycles et une méthode de suppression de ces derniers ont été établies. Pour supprimer tout cycle dans le graphe, une condition appliquée est celle d'avoir toutes les valeurs propres de la matrice d'adjacence nulles.

La transformation du graphe de contrôle sans cycle a permis de formuler une représentation des équations aux éléments de rétroaction qui induisent les cycles (ou boucles). Ceci a rendu possible la décomposition avec interconnexion aux points de rétroaction (DIPR) : il s'agit de la contribution principale de cette thèse. Il faut souligner que les transformations introduites ne modifient aucunement les modèles de contrôle. Les coupures virtuelles ne servent qu'à la formation des équations. Les méthodes de solution des systèmes d'équations sont appliquées au chapitre suivant.

CHAPITRE 3 SOLUTION DES SYSTÈMES DE CONTRÔLE

Suite à l'établissement de la nouvelle formulation dans le chapitre précédent, on peut maintenant introduire la méthode de solution. Ce chapitre comporte deux sections. Une première section traite des éléments communs aux différentes variantes de la solution. Il s'agit de la fonction objective et de la Jacobienne. Une deuxième section présente les variantes de la méthode de solution dans la nouvelle formulation : la solution complète itérative, la solution d'itérations à point-fixe et la solution non-itérative. Les limitations de chacune des variantes sont aussi discutées.

3.1 Calculs préliminaires communs aux variantes des méthodes

La solution des systèmes de contrôle dans la formulation que nous avons introduite consiste à résoudre le système simultanée en (2.41). En référant à la théorie générale sur la solution des systèmes (non-linéaires), on en déduit la fonction objective Φ à minimiser suivante :

$$\Phi = \mathbf{e} - \boldsymbol{\varphi}(\mathbf{U}, \mathbf{G}(\mathbf{e})), \quad (3.1)$$

où des entrées indépendantes \mathbf{U} sont constantes durant le processus d'itération dans un pas de temps donné. Les fonctions dans Φ dépendent donc uniquement des variables \mathbf{e} aux sorties des points de rétroaction. Le problème peut être mis sous la forme :

$$\Phi(\mathbf{e}) = 0. \quad (3.2)$$

Ainsi présentée, la solution simultanée de M équations associées aux points de rétroaction dans (3.1) s'insère dans le cadre plus large des solutions des systèmes abondamment discutées dans plusieurs ouvrages. Parmi ceux-ci, on réfère à [42][43][44][45][46][47][48][49]. Les variantes de la solution que nous appliquerons sont l'itération point-fixe, les itérations Newton de base, ainsi que la méthode de mise à jour des itérations par DOGLEG. Pour ce dernier, nous utiliserons la librairie MINPACK [46] en langage FORTRAN. D'autres langages de programmation peuvent être utilisés.

Cependant, une particularité est à considérer pour la nouvelle méthode, par rapport au cadre général. Il s'agit notamment du fait que l'on doit parcourir les boucles (évaluation des

fonctions composées) et que nous considérons des boîtes-noires (non-linéaires) dans la boucle. À cause de cela, nous présentons les détails d'un processus d'évaluation des fonctions Φ et de la matrice Jacobienne. Ces deux éléments sont en effet, les plus importants dans tout processus itératif de type Newton.

3.1.1 Évaluation des fonctions objectives Φ du processus itératif

Avant de présenter comment évaluer les fonctions Φ , notons que la combinaison de toutes séquences aboutissant à tous les points de rétroaction correspond à l'évaluation de tous les blocs selon la séquence d'exécution découlant de la liste topologique, mais en sautant les blocs des points de rétroaction. Leurs sorties \mathbf{e} restent donc constantes. En d'autres termes c'est l'évaluation de $\mathbf{G}(\mathbf{e})$ de la partie droite de (3.1). Cet aspect, très utile au plan algorithmique, est explicité dans le chapitre 5 (voir, entre autres, (5.7), (5.8) et (5.16)) en utilisant la technique de *substitution des variables*. Ainsi donc, les étapes de la procédure d'évaluation d'une des fonctions objectives Φ_i est la suivante :

1. Fixer les valeurs aux PR \mathbf{e} avec la valeur initiale pour le premier pas de temps ou à partir de la solution du pas précédent : $\mathbf{e} = \mathbf{e}_{init}$ ou encore à partir de l'itération précédente ;
2. Évaluer toutes les fonctions des blocs selon la liste d'exécution établie selon la liste topologique, mais sauter les fonctions aux PR (c'est-à-dire, comme mentionné les \mathbf{e} restent constantes $\mathbf{e} = \mathbf{e}_{init}$). Ainsi, on évalue G_i ;
3. Évaluer les sorties des blocs PR en utilisant les constantes supplémentaires \mathbf{U} : $\mathbf{e} = \varphi_i(\mathbf{U}, G_i(\mathbf{e}_{init}))$ où pour rappel, la fonction φ_i est la fonction du bloc PR ;
4. Évaluer finalement les fonctions objectives pour tous les PR indexés par i :

$$\Phi_i = \mathbf{e}_{init} - \varphi_i(\mathbf{U}, G_i(\mathbf{e}_{init})) \quad .$$

À partir de ces fonctions objectives, on déduit la matrice Jacobienne associée pour toutes les étapes du processus de solution.

3.1.2 Évaluation de la matrice Jacobienne du processus itératif

La matrice Jacobienne \mathbf{J}_k pour le système des M points de rétroaction, à la k -ième itération est :

$$\mathbf{J}_k = \left. \frac{\partial \Phi(\mathbf{e})}{\partial \mathbf{e}} \right|_k \quad (3.3)$$

Cette expression analytique est trouvée par calcul numérique avec la technique de perturbation numérique. Deux valeurs de la fonction objective sont calculées : avant la perturbation $\Phi(e)$ et après la perturbation $\Phi(e + \delta e)$. On appliquera donc le calcul suivant pour chacune des fonctions PR pour évaluer les éléments dans la matrice Jacobienne (3.3) :

$$J^{ij} = \frac{\Phi_i(e_{init} + \delta e_j) - \Phi_i(e_{init})}{\delta e_j}, i = 1 \dots M, j = 1 \dots M, \quad (3.4)$$

où δe_j est la valeur de perturbation du j -ième bloc PR. Les $M \times M$ éléments sont évalués comme suit :

1. Évaluer les fonctions objectives pour chaque bloc PR (selon la procédure à la section 3.1.1 ci-haut), pour les valeurs de départ des sorties \mathbf{e} de PR :

$$\Phi_i = \mathbf{e}_{init} - \varphi_i(\mathbf{U}, G_i(\mathbf{e}_{init}))$$

2. Perturber tour à tour chacun des M blocs PR et évaluer la nouvelle fonction objective :

$$\begin{aligned} \Phi_i(\mathbf{e}_{init} + \delta e_1) &= \mathbf{e}_{init} + \delta e_1 - \varphi_i(\mathbf{U}, G_i(\mathbf{e}_{init} + \delta e_1)) \\ \Phi_i(\mathbf{e}_{init} + \delta e_2) &= \mathbf{e}_{init} + \delta e_2 - \varphi_i(\mathbf{U}, G_i(\mathbf{e}_{init} + \delta e_2)) \\ &\dots \\ \Phi_i(\mathbf{e}_{init} + \delta e_j) &= \mathbf{e}_{init} + \delta e_j - \varphi_i(\mathbf{U}, G_i(\mathbf{e}_{init} + \delta e_j)) \\ &\dots \\ \Phi_i(\mathbf{e}_{init} + \delta e_M) &= \mathbf{e}_{init} + \delta e_M - \varphi_i(\mathbf{U}, G_i(\mathbf{e}_{init} + \delta e_M)), \\ i &= 1 \dots M \end{aligned} \quad (3.5)$$

3. Évaluer les éléments $M \times M$ de la matrice Jacobienne comme en (3.4).

La valeur de perturbation δe_j doit être petite pour représenter adéquatement la dérivée de la réponse du bloc. Une valeur adéquate est reliée à la “précision machine” de la représentation des nombres de la plate-forme de calcul et doit aussi tenir compte de la plage de variation de différents \mathbf{e} . Dans les programmes en FORTRAN sur lesquels on a illustré la méthode, cette valeur a été calculée par la relation suivante (voir aussi MINPACK [46] et [50]) :

$$\delta e_i = \max(10^{-12}, 10^{-8} |e_i|). \quad (3.6)$$

Notons qu’à titre de comparaison, la valeur de perturbation dans MATLAB/Simulink est $\delta e_i = 10^{-5} + 10^{-8} |e_i|$. Par ailleurs, cette dernière est aussi associée à la version numérique du fameux théorème de Kantorovich sur la convergence du système d’équations [47]. Nous présumerons que la valeur de perturbation sera toujours telle que les dérivées numériques des fonctions de blocs seront quasiment égales aux dérivées analytiques dans le contexte d’un temps discret. En général, ceci n’est pas une forte contrainte. Par exemple, un simple système dans la Figure 3-1 avec deux chemins de rétroaction non-linéaires est analysé pour le cas de la Jacobienne analytique et numérique. Les erreurs relatives maximum sur les éléments de la matrice Jacobienne se situent en deçà de 0.01% pour tous les cas tel que présenté dans le Tableau 3.1.

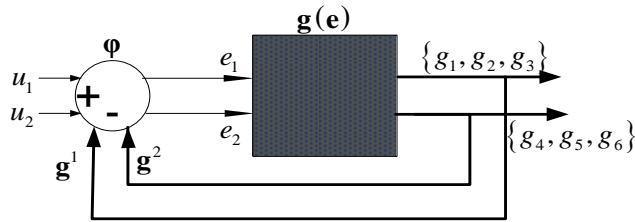


Figure 3-1: Exemple simple de rétroaction avec fonctions non-linéaires (boîte-noire) à perturber

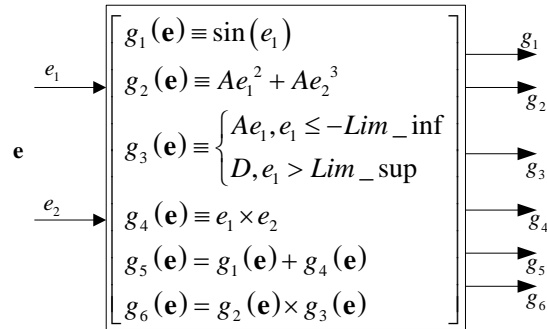


Figure 3-2: Fonctions non-linéaires de la boucle de rétroaction de la Figure 3-1

Terminons cette section sur l'évaluation de la Jacobienne par perturbation numérique en mentionnant, encore une fois, qu'en tenant compte de la possibilité des boîtes-noires, on s'impose qu'au moins sa fonction, décrivant le comportement de la réponse de la boîte-noire, est continue par morceau et que le bloc se comporte bien pour une certaine plage des valeurs de son entrée. Sinon, la dérivabilité de la fonction du bloc, et donc, l'existence et l'unicité de la solution pour tout le système où ce type de blocs est intégré ne seront pas garanties. On fait donc l'hypothèse que la boîte-noire retourne « une bonne sortie pour une bonne entrée », spécifiquement quand on ajoute des perturbations mineures aux entrées. Les cas problématiques peuvent apparaître là où la dérivée peut être bien grande, introduisant une forme de discontinuité. Des traitements spécifiques sont introduits dans ce genre des cas (voir, par exemple, le cas des limiteurs pour la fonction g_3). Nous n'en discuterons pas plus dans le cadre de ce travail.

Tableau 3.1: Illustration de la précision de la Jacobienne numérique : erreurs relatives maximales sur l'évaluation de la Jacobienne numérique en comparaison avec la Jacobienne analytique pour les fonctions de la Figure 3-2

g_i	e_1	e_2	$\text{Err}\left(\left.\frac{\partial g_i}{\partial \mathbf{e}}\right _{num.}, \left.\frac{\partial g_i}{\partial \mathbf{e}}\right _{ana.}\right)$
g_1	[-10,10]	[-10,10]	$\leq 10^{-10}\%$
g_2			$\leq 10^{-5}\%$
g_3			$\leq 10^{-14}\%$
g_4			$\leq 10^{-14}\%$
g_5			$\leq 10^{-8}\%$
g_6			$\leq 10^{-2}\%$

La fonction objective et la matrice Jacobienne étant définies, les méthodes de solution de type Newton peuvent à leur tour être entièrement établies. Dans la section suivante, les variantes du processus itératif de la solution à chaque pas de temps sont introduites.

3.2 Les variantes de la méthode de solution DIPR

3.2.1 Solution complète itérative

La solution des systèmes de contrôle dans la formulation que nous proposons se résume à trouver la meilleure approximation possible du système de M équations simultanées (dans le cas général, non-linéaires) présentées plus haut, que nous reprenons ci-bas :

$$\Phi(\mathbf{e}) = 0. \quad (3.7)$$

Les valeurs les plus approchées du vecteur \mathbf{e} , qui sont les racines de cette équation, peuvent être trouvées par itérations, par un processus bien connu désigné par la méthode de Newton. Ce problème est très vaste et dépasse le cadre de cette thèse. Les caractéristiques des fonctions Φ influencent beaucoup la convergence des itérations de Newton, notamment la dérivabilité de ces dernières en fonction des inconnues \mathbf{e} . Cette méthode sera donc référée comme méthode de Newton de base dans la suite de cette thèse. Étant donnée la complexité des fonctions Φ (pour rappel, elles comportent des fonctions composées le long des chemins de rétroaction), on doit établir aussi des méthodes plus robustes qui vont assurer la minimisation des fonctions Φ par des itérations successives. Leur mise à jour garantit une convergence dans une région de confiance (« Trust region ») selon certaines conditions, dont quelques unes sont discutées dans le dernier chapitre. Rappelons très brièvement deux processus, un de base et un autre avancé, pour résoudre (3.7).

3.2.1.1 Méthode de Newton de base

Cette méthode est basée sur l'approximation du premier ordre des fonctions Φ , parmi d'innombrables références on peut consulter [51]. Du développement en série de Taylor on peut écrire autour d'une valeur donnée du vecteur \mathbf{e} :

$$\Phi(\mathbf{e} + \Delta\mathbf{e}) = \sum_{j=0}^{\infty} \left\{ \frac{1}{j!} \left(\sum_{i=1}^M \Delta\mathbf{e}_i \frac{\partial}{\partial \mathbf{e}_i} \right)^j \Phi(\mathbf{e}) \right\} = 0, \text{ c'est-à-dire pour les 3 premiers termes :}$$

$$\Phi(\mathbf{e} + \Delta\mathbf{e}) = \Phi(\mathbf{e}) + \Phi'(\mathbf{e})(\Delta\mathbf{e}) + \frac{1}{2}\Phi''(\mathbf{e})(\Delta\mathbf{e})^2 + \dots = 0$$

En négligeant les dérivées secondes, on a

$$\Phi(\mathbf{e} + \Delta\mathbf{e}) \approx \Phi(\mathbf{e}) + \Phi'(\mathbf{e})(\Delta\mathbf{e}) = 0.$$

On peut alors déduire la valeur de l'accroissement :

$$\Phi'(\mathbf{e})(\Delta\mathbf{e}) = -\Phi(\mathbf{e}).$$

La mise à jour de la k -ième à la $k+1$ -ième itération est alors :

$$\mathbf{e}^{k+1} = \mathbf{e}^k + \Delta\mathbf{e}^k = \mathbf{e}^k - \frac{\Phi(\mathbf{e}^k)}{\Phi'(\mathbf{e}^k)}, \quad (3.8)$$

où $\Phi(\mathbf{e}^k)$ et $\Phi'(\mathbf{e}^k)$ sont respectivement la fonction objective et la Jacobienne à la k -ième itération. Les procédures pour les calculer dans la formulation DIPR se trouvent respectivement aux sections 3.1.1 et 3.2.1.

La méthode d'itérations de Newton de base (Newton) dans la formulation DIPR peut être résumée comme suit :

1. Fixer les valeurs aux points de rétroaction PR \mathbf{e} , avec la valeur initiale définie pour le premier pas de temps ou à partir de la solution du pas précédent : $\mathbf{e}^0 = \mathbf{e}_{init}$;
2. Évaluer toutes les fonctions objectives selon la procédure à la section 3.1.1;
3. Évaluer la matrice Jacobienne selon la procédure à la section 3.1.2;
4. Si on n'a pas convergé, continuer avec la prochaine itération $k+1$ en mettant à jour les inconnues selon (3.8);
5. Revenir à l'étape 2 tant que l'on n'a pas encore convergé.

Le critère de convergence n'est pas si simple à établir. En effet, comme indiqué, cette méthode nécessite une attention particulière à évaluer les fonctions Φ , à cause du fait qu'il s'agit des fonctions composées le long des boucles de rétroaction (pour rappel, il s'agit des fonctions \mathbf{G} , voir entre autres (2.36) ou (2.38) pour le PLL). Nous avons implémenté un double critère inspiré de la routine MINPACK :

Le premier s'écrit comme suit :

$$|e_i^{k+1} - e_i^k| \leq e_{TOL}, \quad i = 1, \dots, M, \quad (3.9)$$

où e_{TOL} est la tolérance pour les sorties des blocs PR.

Quant au deuxième, il est inspiré du « F-convergence » de [50] (voir la discussion dans la section 4.4.1) :

$$\left| \frac{\|\Phi^k\| - \|\Phi^{k+1}\|}{\|\Phi^k\|} \right| \leq F_{TOL}, \quad (3.10)$$

où la tolérance F_{TOL} est la tolérance sur la fonction objective.

Une limite de cette méthode est que la mise à jour est laissée à la merci de la valeur évaluée de la Jacobienne ou plus précisément au rapport entre les fonctions objectives et cette dernière. Cette mise à jour peut conduire à plusieurs limitations, incluant l'éloignement vers la solution supposée ou des comportements oscillatoires entre des valeurs qui ne sont pas acceptables comme solution et rejetées par les critères de convergence.

Pour palier à ces limitations de la méthode de Newton de base et pour tenir compte de la complexité des fonctions Φ mentionnée plus haut, une deuxième variante de la méthode de solution de type itératif un peu plus robuste a été introduite. Nous en présentons brièvement la description dans la section suivante.

3.2.1.2 Méthode Dogleg de la Routine HYBRJ1 de MINPACK

Comme on vient de le mentionner, la méthode de Newton de base peut présenter des limitations qui sont exacerbées à cause de la complexité des fonctions objectives. Des méthodes plus robustes s'avèrent nécessaires.

Nous avons implémenté une variante basée sur la famille des méthodes que l'on doit successivement à Broyden, Powell et implémentées dans la librairie MINPACK [46][50][52][53]. Plusieurs variantes sont disponibles dans les routines dont les détails sont disponibles dans [50]. Nous avons utilisé la routine HYBRJ1 où les fonctions objective et la matrice Jacobienne doivent être fournies à la routine. Celles-ci sont respectivement calculées selon la procédure présentée à la section 3.1.1 et à la section 3.2.1. Cette routine utilise les paramètres internes de la

convergence. Un affinement supplémentaire serait possible avec la routine HYBRJ, similaire au premier. Par ailleurs, une autre famille des variantes de MINPACK pour résoudre (3.7) calcule la Jacobienne par une procédure quasi-similaire à celle que nous avons adaptée au point 3 de la section 3.1.2. Il d'agit des routines HYBRD1 avec paramètres internes de convergence. Elle est associée à HYBRD, où on peut, encore une fois, affiner ces derniers. Pour faire un lien avec d'autres routines utilisées dans d'autres programmes de simulation de contrôle, terminons cette description de MINPACK en mentionnant qu'une autre formulation serait possible selon la minimisation au sens de moindres carrés. On devrait, pour ces cas, reformuler les fonctions objectives comme :

$$\min \left\{ \sum_{i=1}^M (\Phi_i(\mathbf{e}_{aug}))^2 \right\}. \quad (3.11)$$

Cette option serait utile, par exemple, si on avait des variables supplémentaires à ajouter tel que :

$$\mathbf{e}_{aug} = [\mathbf{e} \ v_1, \dots, v_Q], \quad (3.12)$$

où on a Q inconnus supplémentaires à ajouter dans les équations. Ceci pourrait être le cas d'ajouts des variables d'état en plus des variables des sorties aux points de coupure \mathbf{e} . Les routines pour résoudre seront alors basées sur les méthodes Levenberg-Marquadt [54][55].

Pour illustrer l'usage des routines de MINPACK, pouvant être adaptées à souhait, revenons sur un aspect le plus important de la routine HYBRJ1 que nous avons appliqué pour résoudre (3.7) en comparaison avec la méthode Newton de base.

La routine HYBRJ1 prend en entrée les fonctions pour calculer toutes les Φ et la Jacobienne \mathbf{J} .

La différence par rapport à la méthode Newton de base est la mise à jour des itérations. Plus précisément on assure que la mise à jour reste dans une certaine région autour de la valeur attendue comme solution. Cette procédure est nommée DOGLEG et elle est appliquée pour les routines HYBRJ, HYBRD, HYBRJ1 et HYBRD1 évoquées plus haut.

L'algorithme est comme ci-dessous [50] :

1. Fixer les valeurs aux points de rétroaction PR \mathbf{e} avec la valeur initiale pour le premier pas de temps ou à partir de la solution du pas précédent : $\mathbf{e}^0 = \mathbf{e}_{init}$;

2. Évaluer toutes les fonctions objectives Φ^k selon la procédure à la section 3.1.1;
3. Évaluer la matrice Jacobienne \mathbf{J}^k selon la procédure à la section 3.1.2.;
4. Si on n'a pas convergé, continuer avec la prochaine itération $k+1$ en mettant à jour les inconnues selon le point 5 suivant ;
5. Solutionner le problème de minimisation pour déterminer $\Delta \mathbf{e}$:

$$\min \{ \|\Phi^k + \mathbf{J}^k \Delta \mathbf{e}\| : \|\mathbf{D} \Delta \mathbf{e}\| \leq \Delta m \}, \quad (3.13)$$

où \mathbf{D} et Δm sont respectivement une matrice de mise à l'échelle et la valeur maximum de pas de changement entre deux itérations. Ces deux paramètres sont fixés à l'intérieur de l'algorithme. La norme $\| \cdot \|$ est la norme euclidienne.

La mise à jour est, encore une fois :

$$\mathbf{e}^{k+1} = \mathbf{e}^k + \Delta \mathbf{e}^k ;$$

6. Revenir à l'étape 2 tant que l'on n'a pas encore convergé.

Le critère de convergence combine le test d'égalité basée sur la précision machine et le test de variation de la fonction objective. Il est présenté dans les équations (3.9) et (3.10) que nous reprenons ici :

$$|e_i^{k+1} - e_i^k| \leq e_{TOL}, \quad i = 1, \dots, M,$$

où e_{TOL} est la tolérance pour les sorties des blocs PR et,

$$\left| \frac{\|\Phi^k\| - \|\Phi^{k+1}\|}{\|\Phi^k\|} \right| \leq F_{TOL},$$

où F_{TOL} est la tolérance sur la fonction objective.

En principe, la méthode Dogleg est plus robuste que celle de Newton de base [46]. Les deux méthodes sont, par contre, coûteuses à cause de l'évaluation de la Jacobienne. Une alternative est l'utilisation de la méthode point-fixe. Elle est dans la section suivante.

3.2.2 Solution par la méthode point-fixe

3.2.2.1 Formulation de la méthode point-fixe

Une particularité intéressante de la formulation des équations en DIPR est que les relations entre toutes les sorties des blocs PR \mathbf{e} et leurs entrées via les différents chemins de rétroaction $\varphi_i(G_i(\mathbf{e}))$ sont uniquement fonction des mêmes sorties des blocs et des constantes durant le processus d'itération. Ainsi donc, les itérations successives s'apparentent à celles à point-fixe. Deux itérations k et $k+1$ sont reliées comme un problème à point-fixe de plusieurs variables découlant de (2.41),

$$\mathbf{e}^{k+1} = \boldsymbol{\varphi}(\mathbf{G}(\mathbf{e}^k)), \quad (3.14)$$

où, pour rappel, les fonctions $\boldsymbol{\varphi}$ et \mathbf{G} sont respectivement les fonctions des blocs PR et les fonctions composées des chemins de rétroaction. On a omis les constantes durant les itérations. Les signaux d'entrées sont aussi omis car ils ne changent pas durant les itérations.

La méthode point-fixe dans la formulation DIPR peut être résumée comme suit :

1. Fixer les valeurs aux points de rétroaction PR \mathbf{e} avec la valeur initiale pour le premier pas de temps ou à partir de la solution du pas précédent : $\mathbf{e}^0 = \mathbf{e}_{init}$;
2. Évaluer toutes les fonctions des blocs selon la liste d'exécution établie suivant la liste topologique, mais sauter les fonctions aux PR (c'est-à-dire, \mathbf{e} restent constantes : $\mathbf{e}^0 = \mathbf{e}_{init}$). Ainsi on évalue \mathbf{G}^0 ;
3. Évaluer les nouvelles valeurs de la sortie de blocs PR: $\mathbf{e}^1 = \boldsymbol{\varphi}(\mathbf{G}(\mathbf{e}^0))$;
4. Vérifier la convergence selon un critère choisi ; Si on a convergé, arrêter ;
5. Si on n'a pas convergé, continuer avec la prochaine itération $k+1$: $\mathbf{e}^{k+1} = \boldsymbol{\varphi}(\mathbf{G}(\mathbf{e}^k))$
6. Revenir à l'étape 4.

Le critère de convergence combine le test d'égalité basée sur la précision machine et le test de variation de la fonction objective : $\Phi^{k+1} = \mathbf{e}^{k+1} - \boldsymbol{\varphi}(\mathbf{G}(\mathbf{e}^k))$. Il est présenté dans les équations (3.9) et (3.10).

3.2.2.2 Avantages et limitations de la méthode point-fixe

Un des avantages de la méthode point-fixe est le fait qu'elle ne nécessite pas de matrice Jacobienne et différents traitements pour établir la mise à jour des inconnus durant le processus itératif. Les opérations coûteuses de perturbation numérique pour établir cette matrice Jacobienne, ni les opérations remplaçant l'inversion de cette dernière ne sont donc requises. Il y a logiquement des gains à attendre de cette méthode. Néanmoins, on sait que la convergence de cette méthode n'est pas assurée dans tous les cas. Certains cas convergent facilement avec une seule itération. Il s'agit des cas comportant un délai dans la boucle de rétroaction :

$$\mathbf{e}^{k+1} = \boldsymbol{\varphi}(\mathbf{G}(\mathbf{e}^k)) = \boldsymbol{\varphi}(\mathbf{G}(\mathbf{e}^1)).$$

D'autres cas convergent avec différents degrés de rapidité selon les caractéristiques du chemin de rétroaction : les fonctions composées \mathbf{G} . On sait, dans la théorie des itérations à point-fixe, que la convergence est liée aux propriétés *contractantes* des fonctions \mathbf{G} . Parmi plusieurs références on peut consulter une collection en ligne [56] ou [11][57][58].

Pour le cas d'une seule variable, la condition de convergence d'itérations à point-fixe est liée à la dérivée d'ordre un, aux points de la valeur de point-fixe. C'est-à-dire que, considérant la formulation DIPR en (3.14), on a pour une seule variable :

$$\left| \frac{\partial \varphi(G(e))}{\partial e} \right| < 1 \quad (3.15)$$

En réseau électrique les systèmes de contrôle ne remplissent évidemment pas toujours cette condition. Notez par contre que, quand il y a un délai dans la boucle on a :

$$\frac{\partial \varphi(G(e))}{\partial e} = 0 < 1 \quad (3.16)$$

Les connaissances de propriétés contractantes des fonctions \mathbf{G} peuvent donc grandement faciliter le choix de méthode de solution, mais ceci n'est pas aisé en général. Un cas simple de la Figure 2-7 a une fonction $\varphi(e) = 10 - (e)^2$ dont la dérivée ne remplit pas la condition (3.15); en fait, il n'est pas résolu avec succès par la méthode point-fixe. On vérifie que

$$\left| \frac{\partial \varphi(e)}{\partial e} \right|_{e=\hat{e}} = |-2\hat{e}| \geq 1, \text{ dans le voisinage de la solution } \hat{e} = 2.701.$$

Pour les cas des systèmes à plusieurs variables, la condition de convergence de la méthode point-fixe est liée à la norme de la matrice des dérivées des fonctions sur lesquelles on performe les itérations. Il s'agit de la matrice Jacobienne $\boldsymbol{\varphi}'(\mathbf{G}(\mathbf{e}))$. Formellement, donc, la condition est [11][58]:

$$\left\| \begin{array}{ccccc} \frac{\partial \varphi_1(\mathbf{G}(\mathbf{e}))}{\partial e_1} & \dots & \frac{\partial \varphi_1(\mathbf{G}(\mathbf{e}))}{\partial e_i} & \dots & \frac{\partial \varphi_1(\mathbf{G}(\mathbf{e}))}{\partial e_M} \\ \vdots & \ddots & \vdots & \dots & \vdots \\ \frac{\partial \varphi_i(\mathbf{G}(\mathbf{e}))}{\partial e_1} & \dots & \frac{\partial \varphi_i(\mathbf{G}(\mathbf{e}))}{\partial e_i} & \dots & \frac{\partial \varphi_i(\mathbf{G}(\mathbf{e}))}{\partial e_M} \\ \vdots & \dots & \vdots & \ddots & \vdots \\ \frac{\partial \varphi_M(\mathbf{G}(\mathbf{e}))}{\partial e_1} & \dots & \frac{\partial \varphi_M(\mathbf{G}(\mathbf{e}))}{\partial e_i} & \dots & \frac{\partial \varphi_M(\mathbf{G}(\mathbf{e}))}{\partial e_M} \end{array} \right\|_{\mathbf{e}=\hat{\mathbf{e}}} < 1, \quad (3.17)$$

où $\|\cdot\|_{\mathbf{e}=\hat{\mathbf{e}}}$ est une norme appropriée évaluée au point-fixe $\mathbf{e}=\hat{\mathbf{e}}$. Nous avons pris la norme spectrale, que nous notons $\lambda(\cdot)$, pour les tests du chapitre quatre. On aura donc pour la solution $\hat{\mathbf{e}}$ aux blocs PR [58]:

$$\lambda(\boldsymbol{\varphi}'(\mathbf{G}(\hat{\mathbf{e}}))) < 1. \quad (3.18)$$

Pour illustration, reprenons l'exemple du PLL abondamment analysé dans les chapitres antérieurs. Les fonctions composées de deux chemins de rétroaction ont été établis comme :

$$G_1 = \cos \left\{ \int \left[f_2 + \left(f_5(e_2) + \int f_6(e_2) dt \right) \right] dt \right\} \times \dots \quad (3.19)$$

$$f_{15} \left(\int \left[e_1 \times \cos \left\{ \int \left[f_2 + \left(f_5(e_2) + \int f_6(e_2) dt \right) \right] dt \right\} \right] dt \right)$$

$$G_2 = e_1 \times \sin \left\{ \int \left[f_2 + \left(f_5(e_2) + \int f_6(e_2) dt \right) \right] dt \right\}, \quad (3.20)$$

où f_2, f_5, f_6 et f_{15} sont respectivement le bloc Constante de 377 et blocs gains de 300, 600 et 500.

Les fonctions pour lesquelles il faut établir les propriétés contractantes sont :

$$\begin{cases} \varphi_1 = u - G_1(e_1, e_2) \\ \varphi_2 = G_2(e_1, e_2) \end{cases} \quad (3.21)$$

La matrice Jacobienne à tester est :

$$\Phi' = \begin{bmatrix} \frac{\partial}{\partial e_1} G_1(e_1, e_2) & \frac{\partial}{\partial e_2} G_1(e_1, e_2) \\ \frac{\partial}{\partial e_1} G_2(e_1, e_2) & \frac{\partial}{\partial e_2} G_2(e_1, e_2) \end{bmatrix}. \quad (3.22)$$

La norme spectrale de cette Jacobienne est établie en annexe par les outils de calcul symbolique.

Si on admet que les sorties de blocs PR varient selon le résultat de la simulation à la Figure 3-3, on peut suivre la variation de la norme spectrale dans la Figure 3-4. On sait alors que la méthode point-fixe ne pourra pas bien performer sur le PLL dans la zone où la norme spectrale n'est pas inférieure à un. Les résultats de la simulation seront présentés dans la section 4.1, suivra ensuite une discussion de cette performance à la section 4.4.2

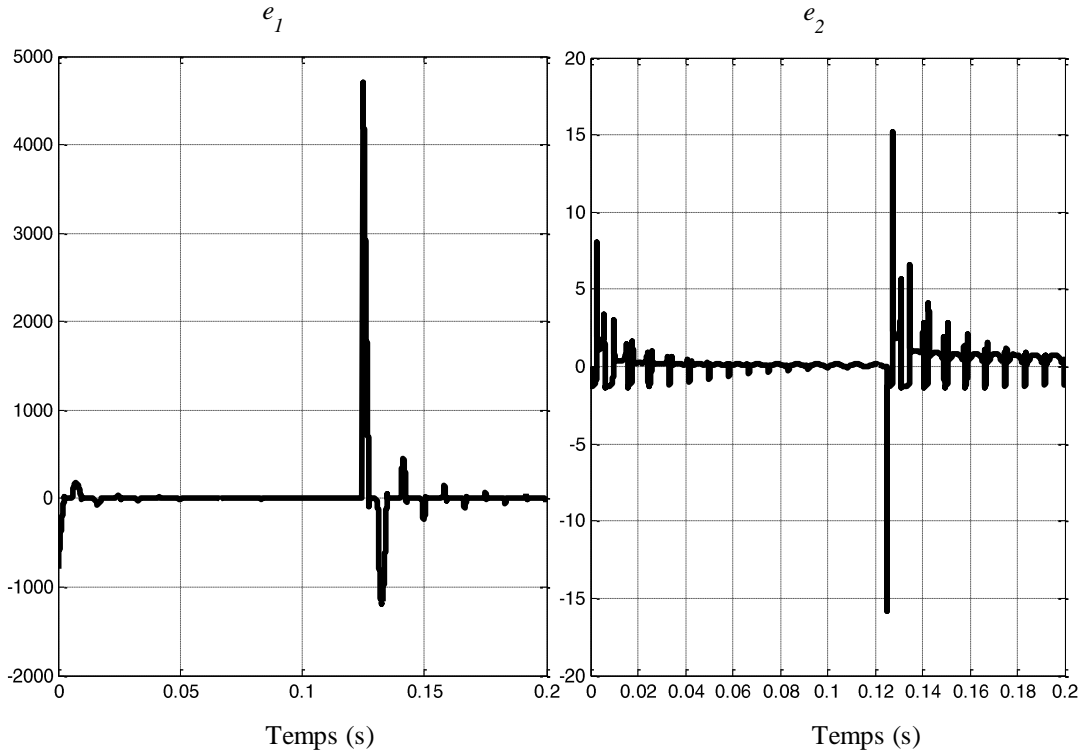


Figure 3-3: Variations des sorties des blocs PR pour le PLL dans une simulation avec un pas de temps de 1 μ s.

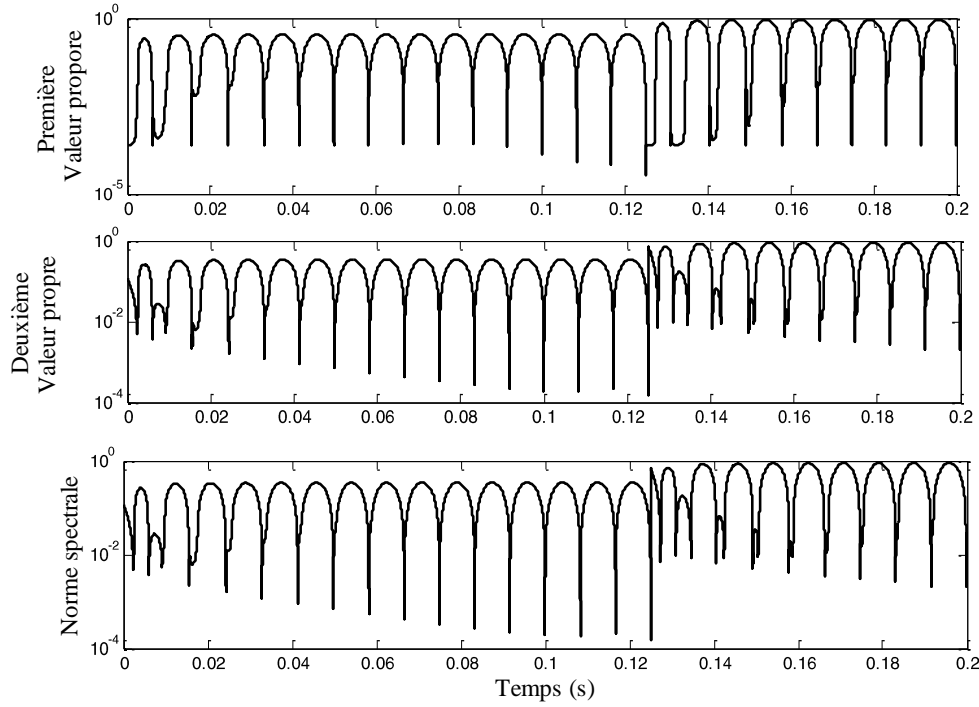


Figure 3-4: Valeurs propres et norme spectrale de la Jacobienne des fonctions composées de chemin de rétroaction correspondantes aux solutions de la Figure 3-3 pour le PLL

En gardant en vue cette limitation de la méthode point-fixe, nous avons implémenté une méthode de solution calibrée avec un test de convergence convenable sur l'erreur relative de la fonction objective et la précision d'égalité pour les inconnues \mathbf{e}^{k+1} et \mathbf{e}^k . Elle sera appliquée sur des systèmes de contrôle couramment utilisés. On l'a testé, entre autres, sur les modèles d'éoliennes de [1]. La propriété de la norme de la Jacobienne étant évaluée avec le point-fixe, qui est la solution recherchée, elle est difficile à établir car il faut préalablement connaître la solution avant d'utiliser la méthode. Il est commode d'établir, par des méthodes spécialisées, ces normes de la Jacobienne. Nous avons proposé une approche dans ce sens. Les détails peuvent être trouvés dans [59].

Une dernière variante ne requiert pas les itérations comme la méthode de Newton ou les itérations à point-fixe. Elle est présentée ci-bas.

3.2.3 Solution non-itérative

3.2.3.1 Formulation de la solution non-itérative

Pour une certaine classe des systèmes de contrôle, considérant un pas de temps de calcul réduit et des variations lisses des entrées tout comme des fonctions non-linéaires, on peut se limiter à une seule itération, mais tout au moins, refaire la linéarisation à chaque nouveau pas de temps. Cette approche est similaire à celle de [4].

Encore une fois, en considérant la linéarisation sur toute la boucle, c'est-à-dire des fonctions $G_i(\cdot)$ établies plus haut, on a une variante non-itérative de la solution avec la formulation DIPR.

3.2.3.2 Limitations de la méthode non-itérative

Une condition minimale est à observer pour cette variante non-itérative de la solution. Les dérivées de la fonction objective Φ , et donc, de chacun des blocs pris individuellement doivent être quasi-constantes autour du point de temps. En d'autres termes, les nonlinéarités doivent être moins sévères sans quoi les erreurs peuvent s'accumuler et conduire à une fausse solution. Ceci est une limitation de la méthode. En effet, cette variante est exacte pour le cas parfaitement linéaire : plus le degré de nonlinearité augmente plus elle devient mauvaise. Comme en [4], elle sera utilisée avec précaution, par exemple en validant ces solutions avec celles d'une méthode itérative. Il est recommandé à l'utilisateur de cette variante de la solution de vérifier le bon comportement des blocs et d'évaluer le degré de nonlinearité des blocs en fonction du pas de temps choisi.

Terminons en rappelant que l'initialisation au temps $t = 0$ se fait avec une méthode itérative. Pour les applications du chapitre qui suit, la méthode itérative complète de [4] est appliquée à $t = 0$. Justement, les applications pratiques de toutes les variantes sont illustrées sur quelques systèmes. Le premier est un système de contrôle avec nonlinéarités dans les boucles, tel le PLL présenté plus haut. Le deuxième système est une machine asynchrone modélisée par les équations algébriques. Enfin, le troisième système est un cas de grande taille. Plusieurs groupes d'éoliennes sont intégrés au réseau électrique test de IEEE 39-barres.

3.3 Conclusion

Ce chapitre visait à appliquer les méthodes de résolution des systèmes d'équations (non-linéaires) de contrôle. Les méthodes suivantes ont été déployées pour la nouvelle approche DIPR : Newton de base, Dogleg (mise à jour des itérations par Dogleg), itérations à point-fixe et variantes non-itératives de la méthode de Newton de base et de celle de Dogleg.

Étant donné que la fonction objective et la Jacobienne, communes à plusieurs variantes sont requises, elles ont été présentées une seule fois pour toutes les variantes. L'établissement des chemins de rétroaction définissant les fonctions d'itérations à point-fixe constitue une des contributions majeures de l'approche DIPR. Le critère de convergence exigé pour cette dernière (comme pour toutes les méthodes itératives) remplit l'exigence de précision pour la solution dans EMTP.

Les applications de ces variantes sur des cas couramment rencontrés sont présentées et discutées au chapitre suivant.

CHAPITRE 4 APPLICATIONS DE LA MÉTHODE PROPOSÉE

Les différentes variantes de la nouvelle méthode DIPR introduites à la section 3.2 sont appliquées sur des systèmes de contrôle représentatifs des cas rencontrés en pratique. Un premier cas est un système de contrôle avec boucle de rétroaction comportant des nonlinéarités. Le PLL introduit au Chapitre 2 appartient à cette catégorie. Il existe plusieurs variantes de PLL et elles sont toutes d'une relative complexité, d'où un intérêt à y tester l'approche proposée. Un deuxième cas est une machine asynchrone modélisée par plusieurs équations algébriques formant un système de dimension 87×87 . Finalement, un troisième cas est un réseau de très grande taille avec des milliers d'équations découlant de 12 centrales éoliennes récemment modélisées pour les applications d'intégration de l'énergie éolienne dans le réseau électrique. Ces cas illustratifs servent, d'une part, à valider la formulation des équations avec cette nouvelle approche et d'autre part, ils permettent l'analyse de performance en terme de gain en réduction du temps de simulation par rapport aux méthodes actuellement disponibles. Nous utilisons l'environnement d'EMTP-RV avec les méthodes introduites dans [4] comme référence. Pour rappel, la taille du système d'équations est directement proportionnelle au nombre des blocs dans chacun des groupes indépendants dans [4], alors que pour la méthode DIPR introduite ici, la taille des équations dépend du nombre des points de rétroaction dans le groupe indépendant. Le groupe indépendant des blocs est celui comprenant des blocs devant être résolus simultanément : il s'agit de la fameuse composante fortement connectée. Les autres signaux dans ce groupe doivent être connus. Terminons cette brève introduction des applications en mentionnant que pour chacune des variantes, nous avons augmenté le pas de temps de simulation jusqu'au point où, soit la méthode diverge ou alors les résultats s'écartent de la référence présentée dans [4]. Ainsi donc, l'avantage relatif de chaque variante est exploité pour mesurer le gain en temps de simulation.

4.1 Système de contrôle avec boucles et nonlinéarités : cas du PLL

Un premier cas résolu par la nouvelle approche a été le PLL largement analysé dans le chapitre traitant de la théorie des graphes. Le PLL est utilisé, entre autres, pour les mécanismes de mise à marche et d'arrêt (commutation) des interrupteurs commandés. Ils sont largement utilisés dans l'électronique de puissance : systèmes de commande des machines, systèmes

d'interconnexion à courant continu à haute tension (CCHT) et systèmes de commande des éoliennes et des panneaux solaires, pour n'en citer que quelques uns.

Le PLL doit être bien calculé car il est crucial pour ces systèmes, sinon les solutions de la simulation des systèmes complets seront loin des réponses attendues et pourront potentiellement engendrer des “faux” casse-têtes pour les concepteurs. Le cas simulé est celui de la Figure 2-1. Le temps total de simulation est 200ms durant lequel il y a une montée sur l'amplitude à 125ms, exactement comme en [4]. Les courbes des résultats de la nouvelle méthode et de la méthode de représentation complète de [4] sont identiques. Par ailleurs, la nouvelle méthode basée sur la mise à jour des itérations par DOGLEG continue de converger même à 40 μ s alors que d'autres variantes de la méthode n'y parviennent pas. Dans le Tableau 4.1, sont présentés les gains en temps de calcul obtenus par rapport à la méthode itérative [4] en s'assurant de la concordance des signaux entre cette dernière et les variantes de la nouvelle méthode. Pour rappel, les variantes de la nouvelle méthodologie appliquées pour ce cas sont :

- Itérations à point-fixe : ne nécessitent pas de Jacobienne, on itère jusqu'à l'atteinte de la tolérance.
- Dogleg non-itératif : la méthode Dogleg est limitée à une seule itération. Notez qu'il y a quelques calculs supplémentaires, par rapport au Newton de base, pour ajuster la mise à jour des itérations ;
- Newton non-itératif : la méthode de Newton de base est limitée à une seule itération ;
- Newton itératif et Dogleg itératif : les itérations sont répétées jusqu'à atteindre la tolérance voulue.

Tableau 4.1: Résultats de la nouvelle méthodologie appliquée au cas du PLL

Variante de la nouvelle méthodologie DIPR	Gain en temps de calcul CPU (%)	Tolérance absolue	Pas de temps de simulation (μs)	Valeur moyenne des itérations
Itérations à point-fixe	-95	1E-05	1	9.9
DOGLEG non-itératif	-7	n/a	15	n/a
Newton non-itératif	4	n/a	15	n/a
Newton itératif	33	1E-05	30	2.28
DOGLEG itératif	92	1E-05	40	3.30

La méthode itérative DOGLEG donne un gain de 92%. Le pas de simulation de la méthode de référence est de $20\mu\text{s}$. Notons que la méthode point-fixe ne converge pas pour ce PLL, sauf pour un pas réduit de $1\mu\text{s}$.

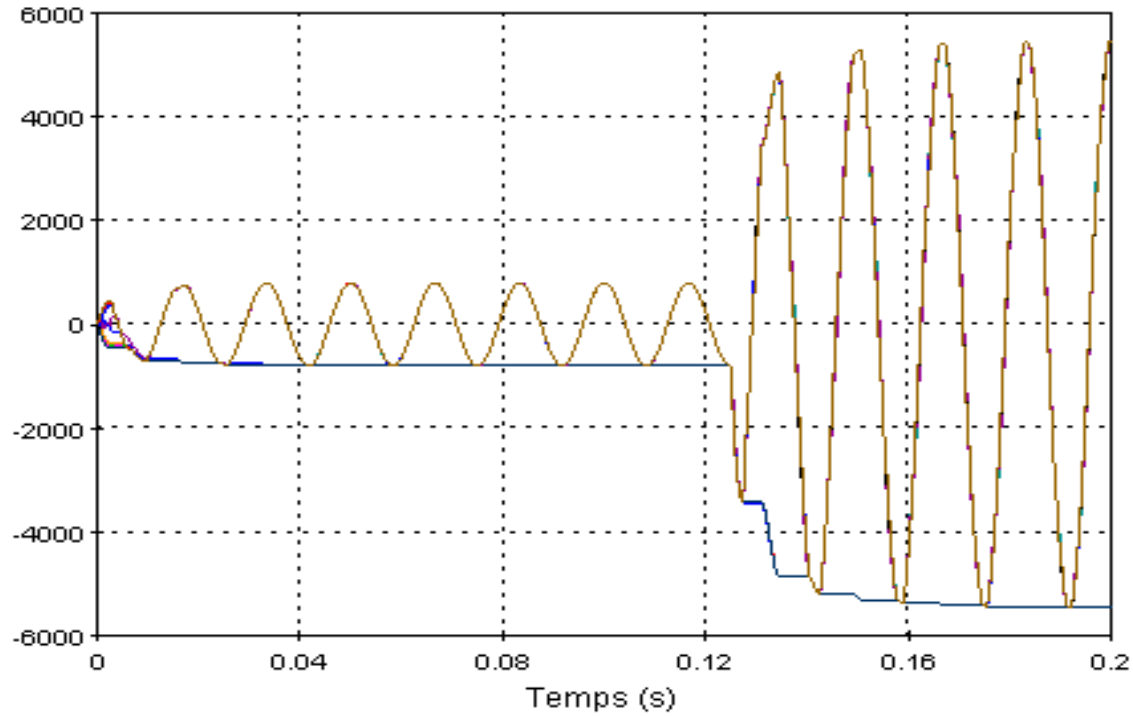


Figure 4-1: Superpositions montrant la concordance parfaite des signaux d'amplitude et de sortie du PLL pour toutes les variantes du Tableau 4.1 et la référence de la solution itérative de [4]

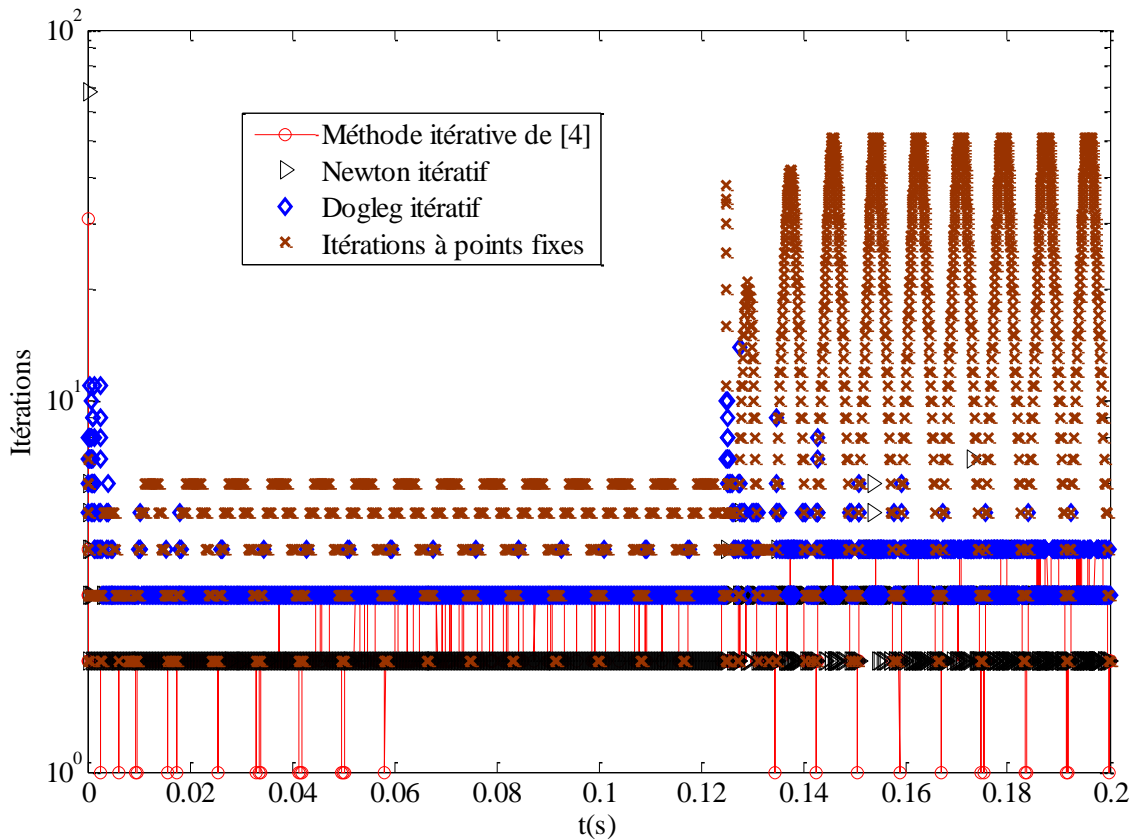


Figure 4-2: Comptage d'itérations à chaque instant t pour les variantes itératives de la solution appliquées au PLL.

La Figure 4-1 montre que, respectivement, la sortie du PLL, devant suivre le signal d'entrée, et l'amplitude coïncident parfaitement entre toutes les méthodes. La même observation est vérifiée pour les autres signaux. De plus, l'analyse de la performance se basant sur le nombre d'itérations menant à la convergence de chacune des méthodes est présentée à la Figure 4-2 et à la Figure 4-3. Pour toutes les méthodes, l'instant de la discontinuité à 125ms et la période d'initialisation requièrent plus d'itérations que pour les autres instants de simulation.

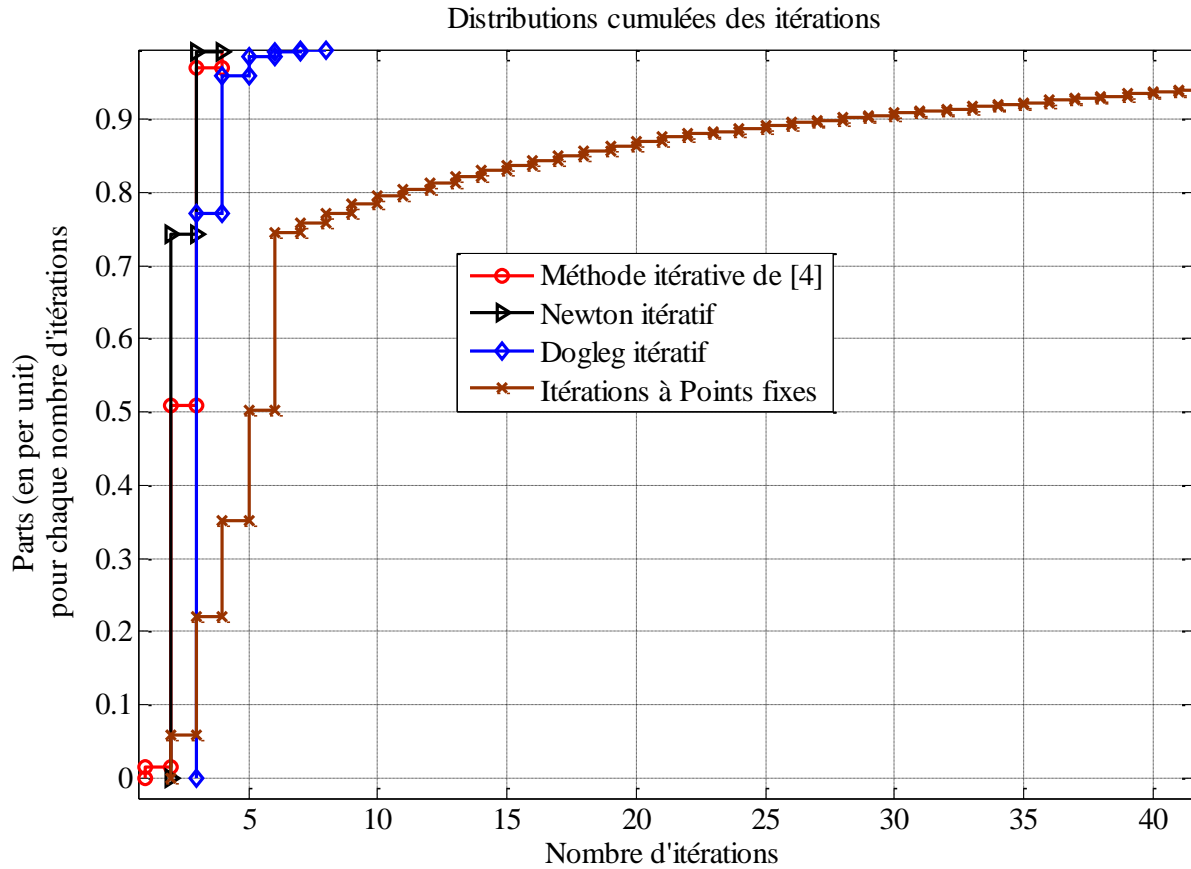


Figure 4-3: Distribution cumulative des itérations à chaque pas de temps pour les variantes itératives de la solution appliquées au PLL

Dans la Figure 4-2, on observe que la méthode point-fixe requiert plus d'itérations, spécialement quand les signaux se rapprochent de zéro. La distribution cumulée des itérations dans la Figure 4-3 montre que pour toutes les méthodes sauf celle de point-fixe, près de 80 % de nombres d'itérations se situent en bas de quatre. Pour la méthode point-fixe, 80 % du nombre d'itérations sont inférieurs à dix. La méthode DOGLEG itératif requiert un peu plus d'itérations que la méthode de Newton itératif. Ceci est dû au fait que pour le cas présenté, les pas de temps de simulation varie et monte jusqu'à 40 μ s pour la méthode DOGLEG. Pour rappel, le gain en temps de calcul est le plus grand, soit proche de deux (92% moins de temps que le cas de référence).

Tableau 4.2: Caractéristiques des systèmes simulés : rétroactions, groupes et blocs non-linéaires

	Numéro de groupes sur lesquels faut itérer	Nombre de blocs PR (M)	Nombre total de blocs par groupe (N)	Nombre de blocs non-linéaires	Ratio N/M
PLL	1	2	14	5	7.0
Machine asynchrone	1	23	87	55	3.78
Centrale éolienne	1	2	7	4	3.5
	2	3	26	15	8.67
	3,4,5,6,7,8	1	9	7	9

4.2 Machine asynchrone modélisée par équations algébriques

Une machine asynchrone a été entièrement modélisée par les équations algébriques, comme en [4]. Tel que présenté dans le Tableau 4.2, le modèle comporte 87 blocs de contrôle dont 55 blocs non-linéaires. Les procédures d'identification des blocs PR de la section 2.4 donnent 23 blocs PR pour le modèle de la machine en garantissant aucune valeur propre non-nulle dans la matrice d'adjacence du graphe associé.

Tableau 4.3: Résultats de la nouvelle méthodologie sur une machine asynchrone entièrement modélisée par les équations algébriques

Variante de la nouvelle méthodologie DIPR	Gain en temps de calcul CPU (%)	Tolérance absolue	Pas de temps de simulation (μ s)	Valeur moyenne des itérations
Itérations à point-fixe	-14.5	1E-5	200	n/a
DOGLEG non-itératif	-23	n/a	200	n/a
Newton non-itératif	-27.20	n/a	200	n/a

Les résultats de la simulation avec les variantes de la nouvelle méthode sont présentés dans le Tableau 4.3. On n'y présente pas les méthodes itératives car pour les pas de temps de simulation considérés, les méthodes non-itératives sont suffisamment performantes jusqu'à 300 μ s. On peut voir déjà certaines limitations de la méthode que nous proposons. La méthode de référence pour la comparaison est la méthode non-itérative de [4] avec un pas de temps de simulation de 200 μ s. Son temps de CPU est légèrement inférieur à celui des toutes les autres

variantes, incluant la méthode point-fixe qui ne requiert pas la formation d'une Jacobienne. Les raisons sont discutées plus loin. On peut observer une concordance entre les solutions des différentes méthodes. Les couples électromagnétiques et le courant au stator sont présentés dans la Figure 4-4.

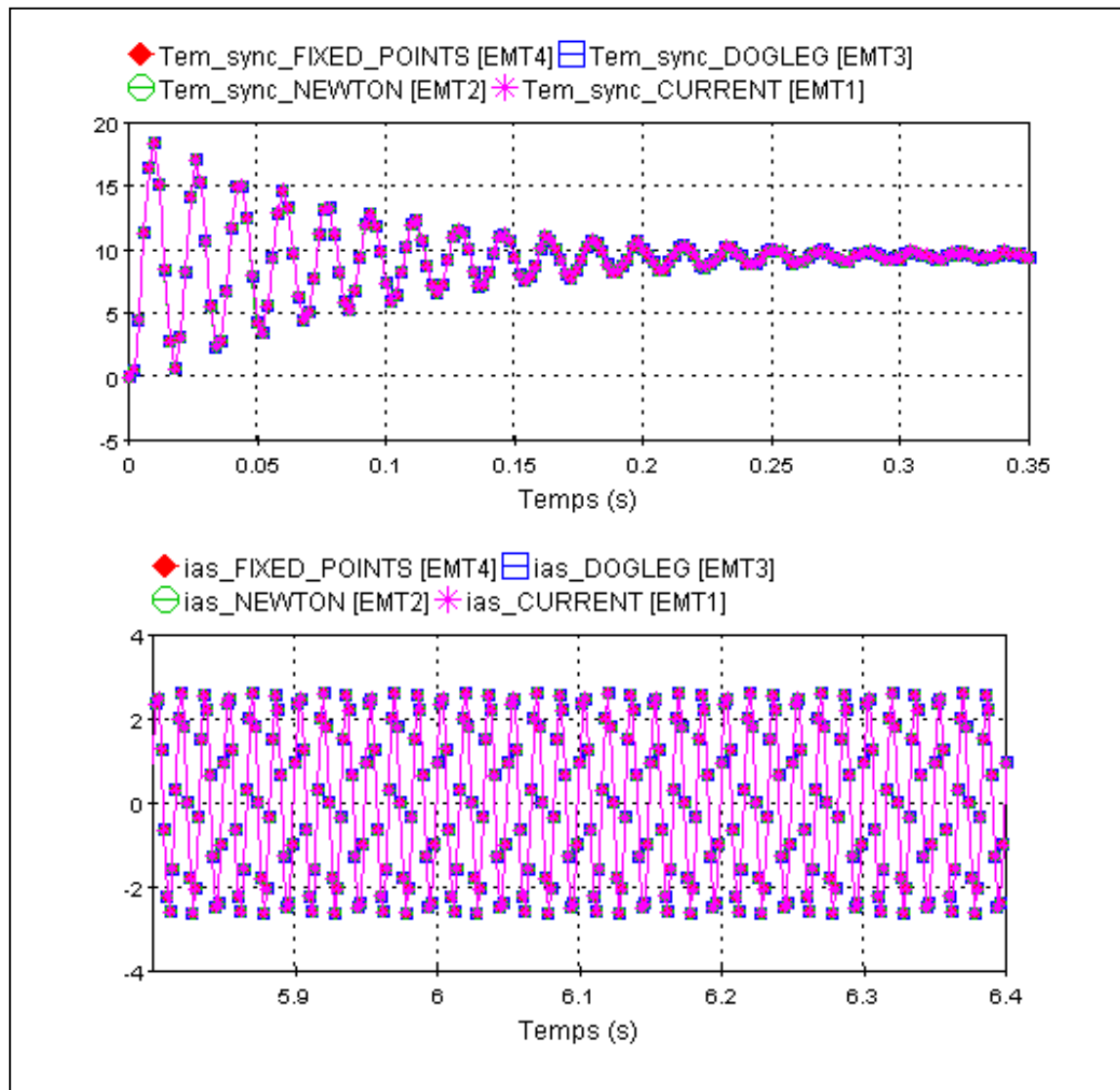


Figure 4-4: Agrandissement de quelques résultats de simulation de la machine asynchrone modélisée par les équations algébriques résolues par les méthodes du Tableau 4.3 et la méthode itérative de [4] :« CURRENT [EMT1]»

4.3 Système de grande taille : éoliennes de [1] intégrées au réseau IEEE 39-barres

Le modèle moyen ("Average Model" : AVM) d'éolienne a été utilisé pour former un modèle agrégé d'une centrale éolienne. Le modèle considéré est celui monté dans [1] avec les modifications introduites dans [60] pour former des systèmes d'équation de très grande taille dans EMTP-RV. Chaque éolienne a sept groupes nécessitant des itérations dans le processus de solution à chaque pas de temps (« groupes itératifs »). Les nombres de blocs PR sont respectivement 7, 26 et 9 pour les 3 types de groupes. Un des groupes itératifs est répété six fois dans chacune des éoliennes. Les blocs non-linéaires sont respectivement au nombre de 4, 15 et 7 dans les 3 types de groupes itératifs. Par ailleurs, chaque éolienne possède 9 autres groupes ne nécessitant pas d'itérations (« groupes non-itératifs »). Les 9 groupes totalisent 443 blocs et pour ces groupes, les sorties de blocs sont calculées séquentiellement selon la liste topologique dont on a parlé plus haut.

Notez qu'une variante hybride combinant deux méthodes a été testée sur ce cas. Dans cette dernière, la méthode point-fixe est utilisée en premier avec une tolérance sur la valeur absolue de $1E-12$ à $1E-5$ mais avec une limitation à 3 itérations à point-fixe. S'il n'y a pas de convergence, la méthode DOGLEG itératif est appelée.

Tableau 4.4: Résultats de la nouvelle méthodologie sur le réseau IEEE-39 barres avec des centrales éoliennes

Variante de la nouvelle méthodologie DIPR	Gain en temps de calcul CPU (%)	Tolérance absolue	Pas de temps de simulation (μ s)	Valeur moyenne des itérations	
				Points Fixes	DOGLEG
Newton non-itératif	40.6	n/a	25	n/a	
DOGLEG non-itératif	35.0	n/a	25	n/a	
Combinaison DOGLEG et Point-fixe (tol. $1E-12$)	52	$1E-12$	25	2	2.5
Itérations à point-fixe	71	$1E-8$	25	2	n/a
DOGLEG+Point-fixe	90	$1E-7$	25	2	2
Combinaison DOGLEG et Point-fixe (tol. $1E-5$)	92	$1E-5$	25	2	2

Un système de très grande taille est formé pour illustrer l'intégration de 12 centrales éoliennes, ayant chacune les caractéristiques ci-haut présentées, sur quatre des barres du réseau IEEE 39-barres. On le modifie par ailleurs en adjoignant des systèmes de contrôle complet aux machines génératrices des autres barres, comme en [60]. Au final, cela forme 1224 groupes des systèmes de contrôle dont 576 groupes itératifs. Le système est présenté à la Figure 4-8.

Les performances des variantes de la nouvelle méthode DIPR sont résumées dans le Tableau 4.4. Notez les différences des performances pour différentes valeurs de la tolérance absolue. Notez aussi les améliorations particulières de la combinaison de la méthode à itérations à point-fixe et la méthode DOGLEG. Avec cette variante hybride, on a enregistré les gains variant de 50 à 90% selon la valeur de la tolérance. Notez aussi que la méthode point-fixe performe bien avec des gains de près de 70% en temps de calcul. La simulation a été de 10 secondes.

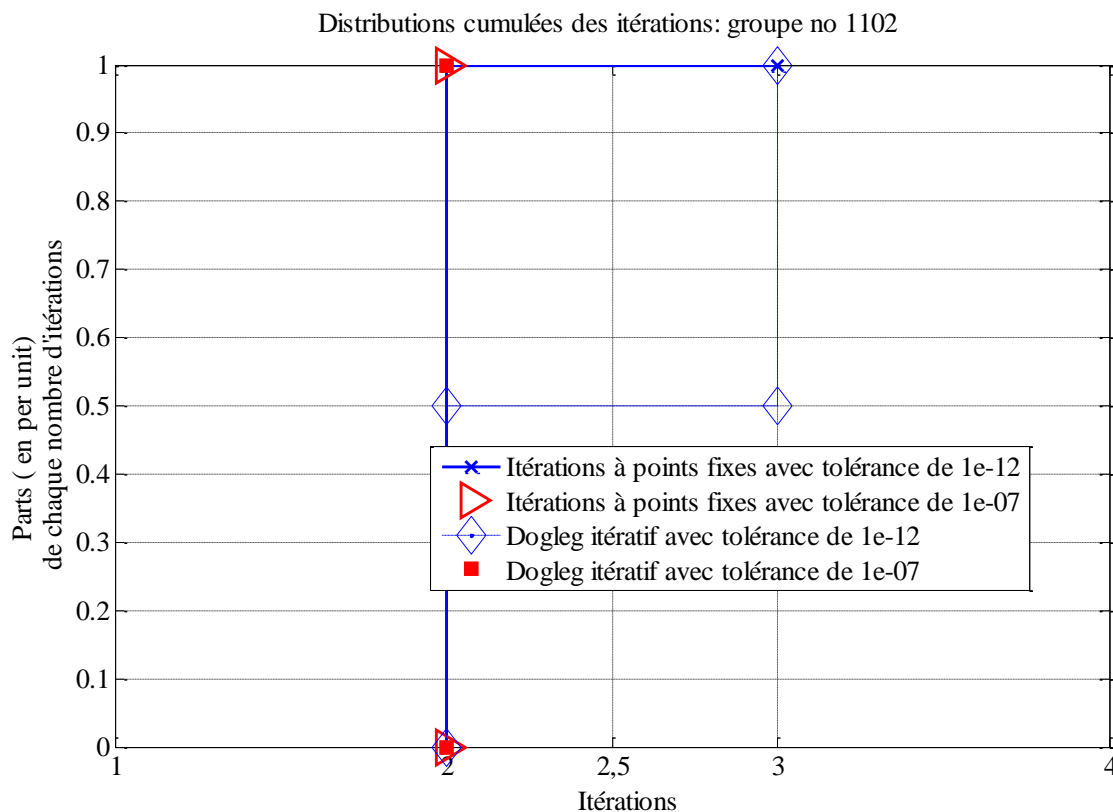


Figure 4-5: Distributions cumulées des itérations pour le groupe de taille 26x26 (no 1102) dans la centrale éolienne de la barre 33, à proximité d'un défaut triphasé initié à 4.5 secondes sur le réseau IEEE 39-barres avec 12 centrales éoliennes

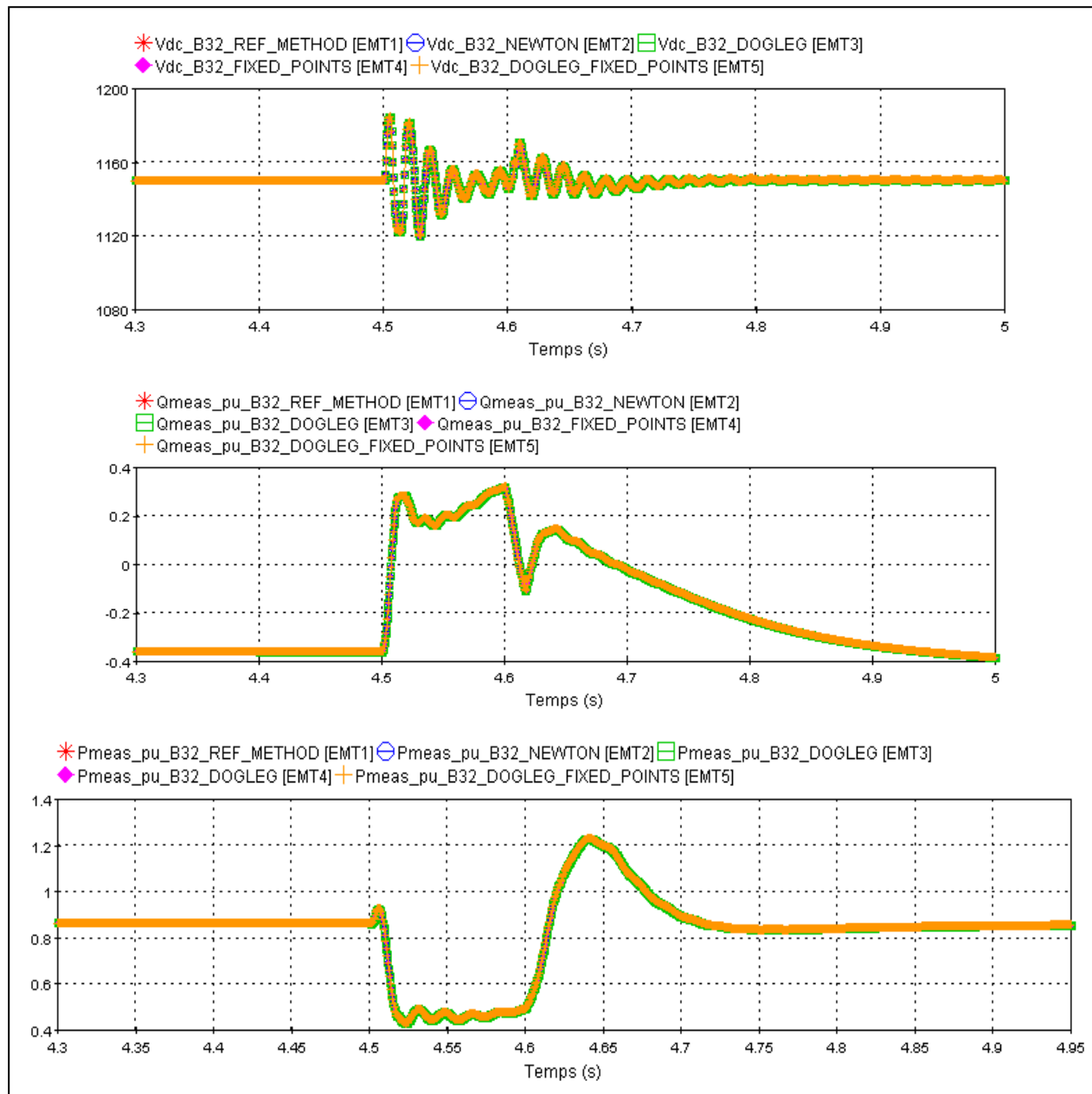


Figure 4-6: Exemple de résultats de simulation des signaux V_{dc} , P et Q sur la barre no 32 du réseau IEEE 39-barres avec 12 centrales éoliennes. [EMTP1] : solution de [4]; [EMTP2] : Newton non-itératif, [EMTP3] : DOGLEG itératif; [EMTP4] : Itérations à point-fixe et [EMTP5] : Combinaison de DOGLEG et itérations à point-fixe.

La concordance de signaux pour toutes les variantes en comparaison de la méthode de référence est montrée dans la Figure 4-6 pour les tensions continues V_{dc} sur la barre CC de l'éolienne, ainsi que pour les puissances actives P et réactives Q injectées par la centrale éolienne

sur la barre du réseau. La même concordance des signaux a été observée pour plusieurs dizaines d'autres signaux.

Pour le besoin d'analyse transitoire, un défaut triphasé d'environ 100 ms est initié à 4.5 secondes. Une analyse du processus itératif montre que le nombre maximum d'itérations est de 3. C'est dans le cas de la tolérance de 1E-12 qu'il a été le plus élevé aussi bien pour la méthode point-fixe que pour la méthode DOGLEG.

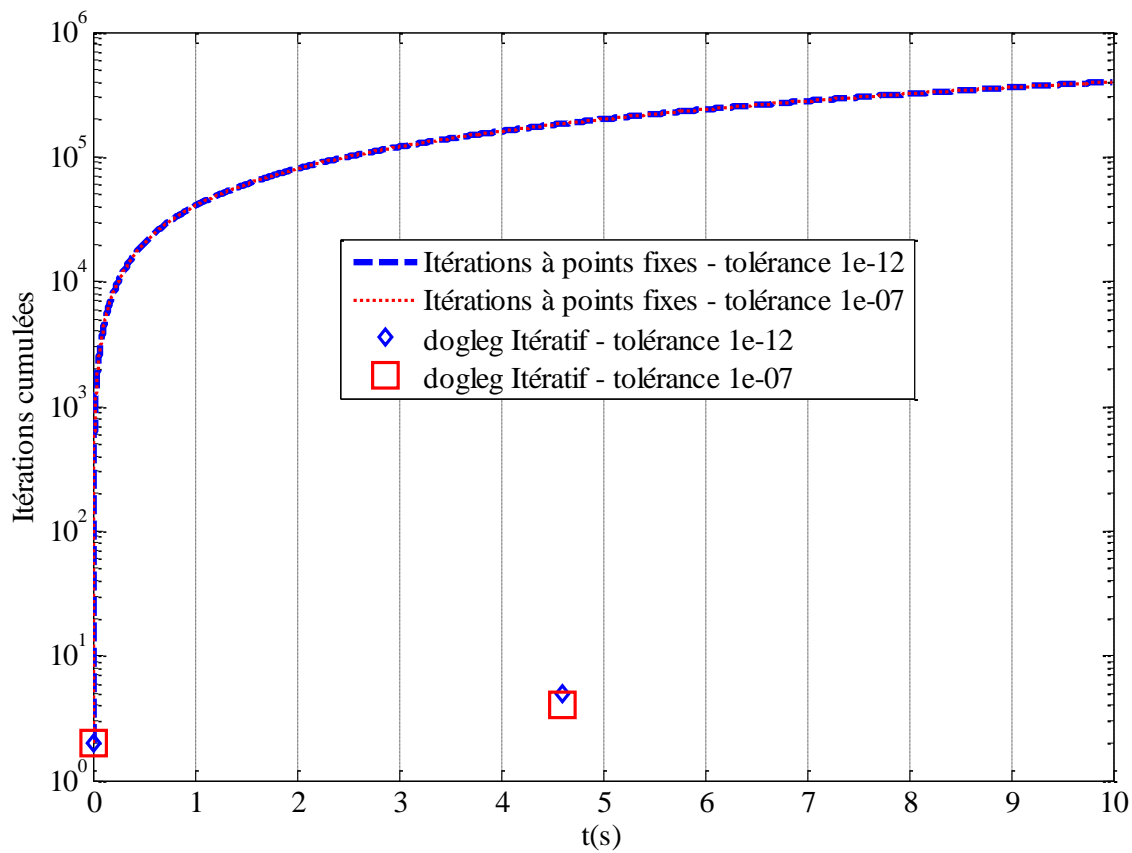


Figure 4-7: Sommes cumulées des itérations sur le groupe de taille 26x26 (no 1102) dans la centrale éolienne de la barre 33 à proximité d'un défaut triphasé initié à 4.5 secondes sur le réseau IEEE 39-barres avec 12 centrales éoliennes

En effet, sur la Figure 4-5 on note que pour tous les cas de la tolérance absolue 1E-7 on a eu seulement 2 itérations avant de converger. Ceci suggère ainsi que la valeur de la tolérance peut affecter la performance, sans toutefois induire une amélioration significative dans la solution. Car pour tous les cas que nous présentons pour comparaison, on a assuré des faibles erreurs entre la

référence et la variante étudiée. Finalement, observons d'une part, dans la Figure 4-7 que la méthode DOGLEG a été utilisée à l'initialisation et pendant la perturbation du défaut à 4.5 secondes pour les deux cas de tolérance absolue $1E-7$ et $1E-2$, ce dernier exigeant légèrement plus d'itérations (précisément une itération de plus en observant la Figure 4-5). D'autre part, pour les itérations à point-fixe, on voit l'impact de l'initialisation sur le nombre d'itérations. La courbe cumulée de ce dernier, en la Figure 4-5, a, de loin, une pente plus grande à l'initialisation pour les deux méthodes de point-fixe.

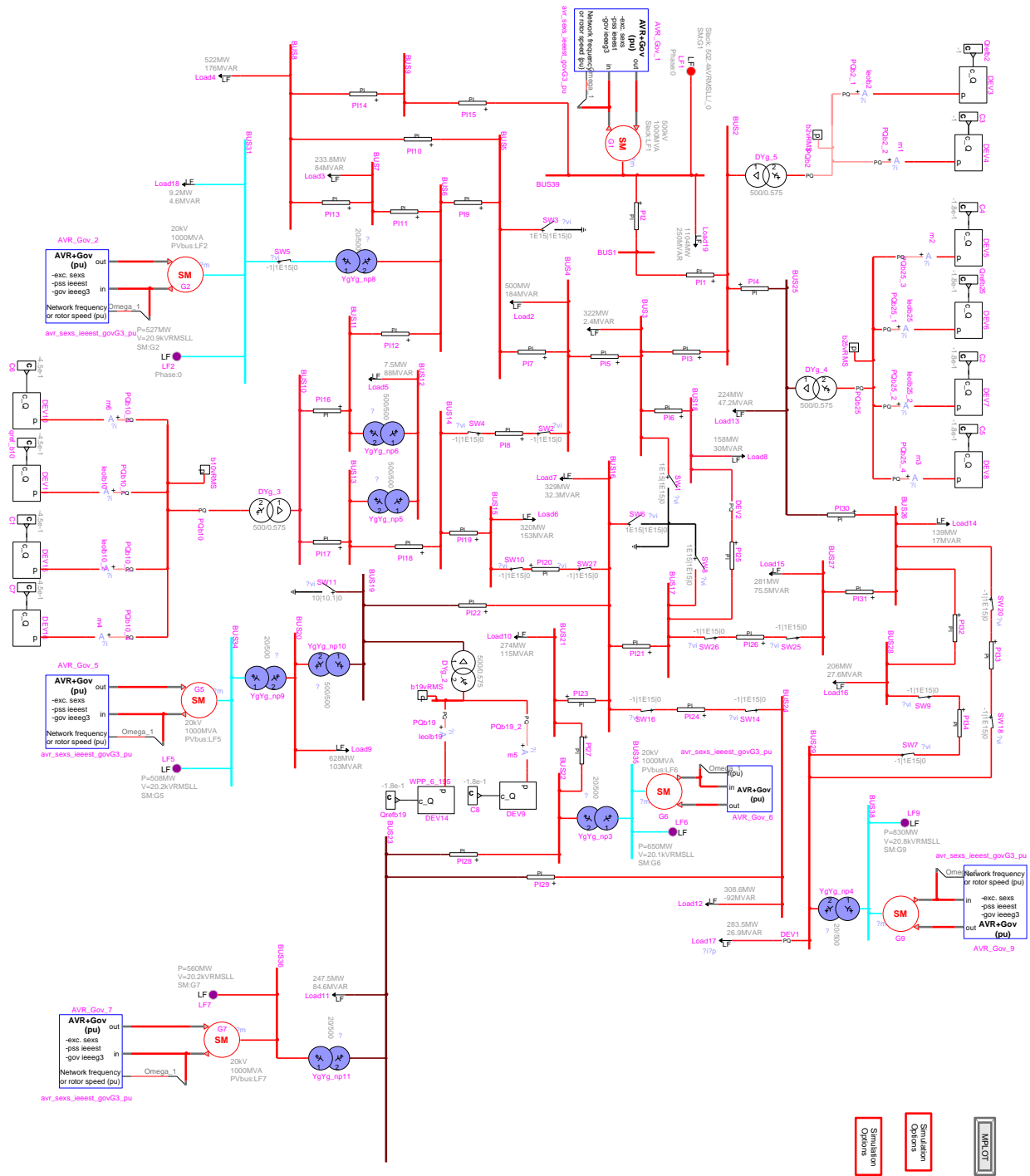


Figure 4-8: Intégration de 12 centrales éoliennes sur le réseau IEEE 39-barres modélisé dans EMTP-RV

4.4 Discussion de la performance de la nouvelle méthode

4.4.1 Précision et test de convergence de la solution

En général, pour tout système d'équations simultanées, la précision de la solution approximée est fonction de la tolérance choisie dans le processus itératif. Cette inévitable limitation a un effet aussi bien sur la solution de référence que sur celle de la nouvelle méthode. Cependant, selon le test de convergence adopté, certaines différences peuvent apparaître. En général, établir le critère de convergence n'est pas simple. En pratique, il faut combiner les tests basés sur la valeur absolue et sur la valeur relative. Par ailleurs, les appliquer sur plusieurs grandeurs dont la fonction objective, les écarts des variables inconnues (par exemple les \mathbf{e} dans la nouvelle méthode proposée) entre les itérations successives ou les estimations de la vraie solution. D'autres approches vont jusqu'à faire le test sur les changements aux éléments de la Jacobienne. Plusieurs variantes sont utilisées car aucune option ne peut être optimale pour tous les cas. En effet, ceci dépend du système à résoudre. Par exemple, il faut parfois introduire des paramètres de mise à l'échelle (« scaling »). C'est ce qui est introduit dans [46] et montré dans [50]. Les tests résultants nommés « X-convergence », « F-convergence », basés respectivement sur les valeurs des inconnues et la fonction objective sont disponibles :

$$\Delta \leq D \times e_{TOL} \times \|\mathbf{e}\|, \quad (4.1)$$

$$\left| \frac{\|\Phi\| - \|\Phi_+\|}{\|\Phi\|} \right| \leq F_{TOL}, \quad (4.2)$$

où Δ est tel que $|D(\mathbf{e}_{k+1} - \mathbf{e}_k)| \leq \Delta$ avec D étant un facteur de mise à l'échelle pour mitiger les disparités entre les valeurs des inconnues \mathbf{e} ; et où e_{TOL} et F_{TOL} sont respectivement les valeurs de tolérance sur les inconnues à évaluer et sur les fonctions objectives. Finalement, $\|\cdot\|$ est la norme euclidienne.

Nous avons utilisé pour les applications présentées au Chapitre 4 les tests « X-convergence », dans (4.1) ou « F-convergence » dans (4.2), alors que pour la méthode de référence tirée de [4], le test utilisé est du type « X-convergence ». Comme le système est représenté en tenant compte de tous les blocs du système, toutes les sorties des blocs sont testées dans [4]. Les deux tests « X-convergence » diffèrent quant à la prise en compte de la propagation

possible d'erreurs dans l'évaluation séquentielle finale des sorties des blocs dans la nouvelle méthode. L'explication est donnée dans le paragraphe ci-bas.

Considérons que les deux méthodes ont convergé suite au test de la forme « X-convergence ». Pour la nouvelle méthode, les erreurs sur les résultats des valeurs des inconnues sont $\Delta \mathbf{e}$ et on veut évaluer la propagation possible d'erreurs sur le n -ième bloc de contrôle dans la boucle représentée par la fonction \mathbf{G} . Soit la fonction composée prenant les valeurs de \mathbf{e} en entrée. Supposons qu'elle est évaluée jusqu'au n -ième bloc sur lequel nous voulons évaluer la précision du résultat de la sortie. Notons la séquence des fonctions menant au n -ième bloc, en fonction des sorties \mathbf{e} , par \mathbf{G}_n . Si le processus itératif a convergé vers les résultats \mathbf{e} alors on a pour ce n -ième bloc du chemin de rétroaction la valeur de sortie avec la nouvelle méthode :

$$x_n(\mathbf{e}) \pm \Delta x_n^e = \mathbf{G}_n(\mathbf{e} \pm \Delta \mathbf{e}), \quad (4.3)$$

où $x_n(\mathbf{e}) = \mathbf{G}_n(\mathbf{e})$ serait la vraie valeur attendue, alors qu'avec la méthode de référence, on a le résultat de la sortie garantie par le critère de convergence de la valeur absolue inférieure à Δx_n , donc on a :

$$|\bar{x}_n - x_n| \leq \Delta x_n, \quad (4.4)$$

où \bar{x}_n est la vraie valeur attendue pour le n -ième bloc.

Il faudrait garantir tout au moins que les deux critères conduisent à une valeur finale dans les limites inférieure et supérieure, respectivement, de la plus petite valeur et de la plus grande valeur de la tolérance. La valeur de la tolérance $\Delta \mathbf{e}$ dans la nouvelle méthode doit être telle que :

$$\begin{aligned}
-\Delta x_n &\leq -|\bar{x}_n - x_n| \\
-\Delta x_n &\leq -|\bar{x}_n - x_n| \leq |\bar{x}_n - x_n| \leq \Delta x_n \\
-\Delta x_n + \bar{x}_n &\leq -|\bar{x}_n - x_n| + \bar{x}_n \leq |\bar{x}_n - x_n| + \bar{x}_n \leq \Delta x_n + \bar{x}_n \\
\text{si on admet que } \bar{x}_n &\leq x_n \\
-\Delta x_n + \bar{x}_n &\leq -\bar{x}_n + x_n + \bar{x}_n \leq \bar{x}_n - x_n + \bar{x}_n \leq \Delta x_n + \bar{x}_n \\
-\Delta x_n + \bar{x}_n &\leq x_n \leq 2\bar{x}_n - x_n \leq \Delta x_n + \bar{x}_n \\
-\Delta x_n + \bar{x}_n &\leq x_n(\mathbf{e}) - \Delta x_n^e \leq x_n \leq x_n(\mathbf{e}) + \Delta x_n^e \leq \Delta x_n + \bar{x}_n \\
\text{Si par contre, on admet l'inverse, on aura } x_n &\leq \bar{x}_n \\
-\Delta x_n &\leq -|\bar{x}_n - x_n| \leq |\bar{x}_n - x_n| \leq \Delta x_n \\
-\Delta x_n + \bar{x}_n &\leq -|\bar{x}_n - x_n| + \bar{x}_n \leq |\bar{x}_n - x_n| + \bar{x}_n \leq \Delta x_n + \bar{x}_n \\
-\Delta x_n + \bar{x}_n &\leq -\bar{x}_n - x_n + \bar{x}_n \leq -\bar{x}_n + x_n + \bar{x}_n \leq \Delta x_n + \bar{x}_n \\
-\Delta x_n + \bar{x}_n &\leq 2\bar{x}_n - x_n \leq x_n \leq \Delta x_n + \bar{x}_n \\
-\Delta x_n + \bar{x}_n &\leq x_n(\mathbf{e}) - \Delta x_n^e \leq x_n \leq x_n(\mathbf{e}) + \Delta x_n^e \leq \Delta x_n + \bar{x}_n
\end{aligned}$$

On a donc pour les deux cas :

$$-\Delta x_n + \bar{x}_n \leq x_n(\mathbf{e}) - \Delta x_n^e \leq x_n \leq x_n(\mathbf{e}) + \Delta x_n^e \leq \Delta x_n + \bar{x}_n \quad (4.5)$$

Si on admet par hypothèse que les fonctions composées $G_n(\mathbf{e})$ donnent la valeur $x_n(\mathbf{e})$ attendue, c'est-à-dire si l'approche DIPR est quasi-identique à celle de [4], alors minimalement, on doit garantir que :

$$\Delta x_n^e \leq \Delta x_n \quad (4.6)$$

L'erreur globale vue au n -ième bloc doit être, tout au plus, égale à celle qui serait introduite en établissant le critère de convergence directement sur le bloc. Or, cette dernière dépend de la fonction composée du chemin vers le n -ième bloc et de l'erreur absolue $\Delta \mathbf{e}$. Celle-ci est établie pour la convergence des variables aux PR. Prenons un cas simple d'une seule boucle de rétroaction avec n blocs en cascade. On sait que la dérivée de fonction composée donne [62][63] :

$$\Delta x_n^e = \prod_i^n \frac{\partial}{\partial e} f_i(\mathbf{e}) \times \Delta e, \quad (4.7)$$

où les n fonctions successives f_i définissent la fonction composée $G_n(\mathbf{e}) = f_n(f(\dots f_1(\mathbf{e})))$.

Si, par exemple, 3 gains de 30 sont en cascade et que l'erreur absolue est $\Delta e = eps = 2.220446049250313e-016$, alors l'erreur équivalente due, Δx_3^e , à l'entrée du premier gain, pour la variable x_3 est :

$$\Delta x_3^e = 30 \times 30 \times 30 \times 2.220446049250313e-016 = 5.995204332975845e-012.$$

On voit donc que la tolérance absolue change d'une manière drastique.

Pour un cas général avec M boucles de rétroaction découplées (indépendantes), on a [62][63] :

$$\Delta x_n^e = \sum_j^M \prod_i^{N_j} \frac{\partial}{\partial e_j} f_{ij}(\mathbf{e}) \times \Delta e_j, \quad (4.8)$$

où les N_j fonctions successives $f_{ij}(\mathbf{e})$ définissent la fonction composée $G_j(\mathbf{e}) = f_{N_j}(f_{N_j-1}(\dots f_{1_j}(\mathbf{e})))$.

Si on considère le cas de blocs à comportement linéaire, on aura une somme d'erreurs dépendant directement du « gain » des blocs. Pour les blocs non-linéaires, on a une somme des termes dépendant des dérivées de blocs. Il est alors évident que le critère de tolérance avec l'erreur $\Delta \mathbf{e}$ doit être plus contraignante dans la nouvelle méthode pour tenir compte de la propagation de l'erreur jusqu'au n -ième bloc pour garantir la condition (4.6).

Dans le Tableau 4.1 et Tableau 4.4 les méthodes itératives ont été résolues avec les tolérances de 1E-12 à 1E-5 pour e_{TOL} et F_{TOL} dans (4.1), (4.2). Si on admet la convergence avec une tolérance de 1E-12 pour $\Delta \mathbf{e}$, la précision obtenue pour les x_n serait donc légèrement inférieure à celle correspondante pour \mathbf{e} à la tolérance de 1E-12. Une hypothèse importante étant encore une fois que les dérivées de blocs sont bornées à des valeurs raisonnables, tel que les erreurs ne soient pas amplifiées à l'infini. Encore une fois, cette exigence est aussi requise pour assurer l'existence et l'unicité même de la solution [47][61].

Il y a un autre aspect à considérer dans cette discussion sur la convergence et la précision en rapport avec la variante non-itérative. Celle-ci, comme mentionné plus haut, est limitée pour des pas de temps réduits ou des systèmes avec pentes de nonlinéarités très réduites. La précision de ces derniers est à évaluer en changeant le pas de temps puis en retenant ceux pour lequel l'écart vis-à-vis du cas avec les plus petits pas possibles est moindre, mais qui a l'avantage de

réduire le temps global de la simulation. Cette tâche n'est toutefois pas aisée. Elle suppose une bonne connaissance des phénomènes simulés. Dans le Tableau 4.3 où ces méthodes non-itératives ont été utilisées, on s'est assuré la concordance entre les pas de $50\mu\text{s}$ et ceux de $250\mu\text{s}$. Comme les deux étaient concordants, les résultats avec $250\mu\text{s}$ ont été aussi retenus.

Terminons en mentionnant que les erreurs entre les deux méthodes ont été, dans tous les cas présentés, très faibles, de l'ordre d'un et d'un demi pourcent. Par contre, l'évaluation des gains en temps de calcul n'a pas été aisée car les pas étaient parfois différents entre les différentes méthodes. Nous discutons, justement, dans la section suivante, des différents gains obtenus avec cette nouvelle méthode pour chacune des variantes.

4.4.2 Gains en temps de calcul des différentes variantes

Des résultats d'illustrations, on peut observer que :

La performance de la méthode point-fixe est à noter pour certains cas présentés ici. Ceci suggère que cette méthode pourrait être de plus en plus utilisée. En effet, pour une certaine classe de systèmes de contrôle, on peut avoir les conditions favorables où les boucles algébriques ont été volontairement coupées, par un délai, par exemple, et/ou le pas de calcul est tellement petit que le comportement global du système reste acceptable avec la solution point-fixe. Cette méthode ne peut pas cependant s'appliquer à tous les cas. En effet, on a vu sa faible performance pour le cas du PLL. On sait que ceci est dû à la mauvaise observance des conditions propices de convergence avec la méthode point-fixe, comme présenté dans la Figure 3-4.

En combinant les graphiques de nombres d'itérations et celui de la norme spectrale de la Jacobienne, on observe dans la Figure 4-2 que, plus la norme spectrale est élevée, plus il faudrait plus d'itérations. À noter que dans la Figure 4-9, la norme spectrale a été analytiquement établie à partir des équations analytiques du PLL avec les intégrateurs discrétisés par la méthode de Tustin. Les termes d'histoire ainsi que les valeurs aux points de rétroaction intervenant dans le calcul de la norme spectrale ont été simulés dans Simulink. Les premiers ont été décalés d'un pas de temps pour correspondre aux termes d'histoire des entrées et sorties des intégrateurs discrétisés. Les itérations, quant à elles, sont celles calculées en implémentant la méthode point-fixe. Dans les deux cas, le pas de temps de simulation est de $1\mu\text{s}$. Cette comparaison fournit qualitativement l'explication de la faiblesse de la méthode point-fixe sur le PLL. On voit, par exemple, que la

norme spectrale relativement faible, entre les instants 0 et 125 ms, se traduit par une dizaine d'itérations, alors que plus la norme spectrale s'approche de un, plus le nombre d'itérations augmente, jusqu'à monter à environ 50 quand la norme spectrale est presque de un.

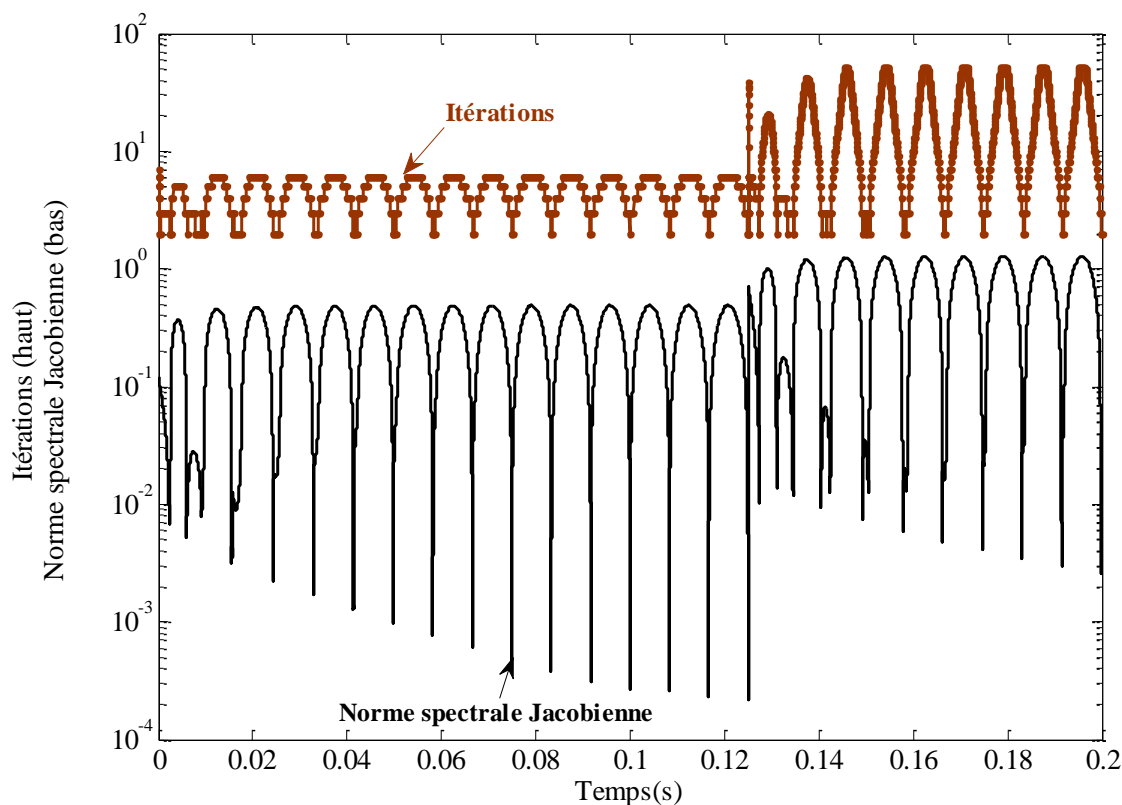


Figure 4-9: Superposition pour le PLL des itérations de la méthode point-fixe et des rayons spectraux de la Jacobienne des fonctions composées de chemins de rétroaction **G**.

À titre de comparaison, analysons maintenant un des groupes des éoliennes qui a été bien solutionné par la méthode point-fixe. Nous prenons le groupe no 4. L'ordre d'exécution, ainsi que les noms des blocs associés à ce groupe, sont dans le Tableau 4.5. Il comprend 9 blocs et une seule coupure enlève tout cycle dans le graphe de ce dernier.

Tableau 4.5: Analyse d'un des groupes dans l'éolienne

Ordre	Numéro d'identification et appartenance au groupe	Numéro d'identification et nom du bloc de contrôle
1	sEQgRP(4)%LIST(1) 251	INSTANCE_NAME(251) 'Fm3\$-983 '
2	sEQgRP(4)%LIST(2) 253	INSTANCE_NAME(253) 'Fm5\$-983 '
3	sEQgRP(4)%LIST(3) 257	INSTANCE_NAME(257) 'Fm9\$-983 '
4	sEQgRP(4)%LIST(4) 252	INSTANCE_NAME(252) 'Fm4\$-983 '
5	sEQgRP(4)%LIST(5) 256	INSTANCE_NAME(256) 'Fm8\$-983 '
6	sEQgRP(4)%LIST(6) 258	INSTANCE_NAME(258) 'Fm10\$-983 '
7	sEQgRP(4)%LIST(7) 104	INSTANCE_NAME(104) 'sum1\$-983 '
8	sEQgRP(4)%LIST(8) 179	INSTANCE_NAME(179) 'Sel1\$-983 '
9	sEQgRP(4)%LIST(9) 170	INSTANCE_NAME(170) 'Int2\$-983 '
sEQgRP(4)%FFB_INDEX(1) 251		

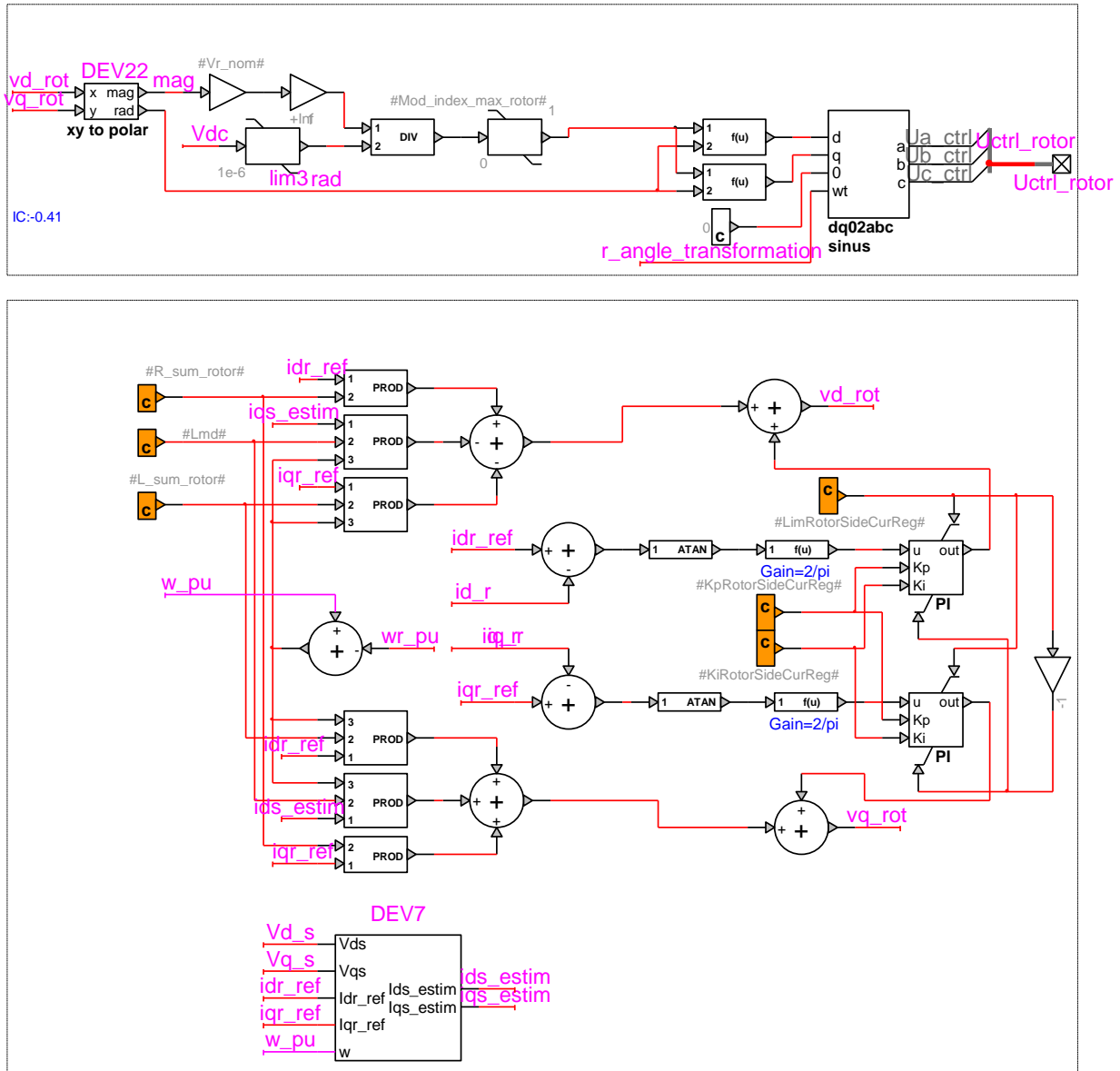


Figure 4-10: Emplacement du groupe no 4 dans une des parties du système de contrôle de l'éolienne (voir le sous masque PI, en bas à droite)

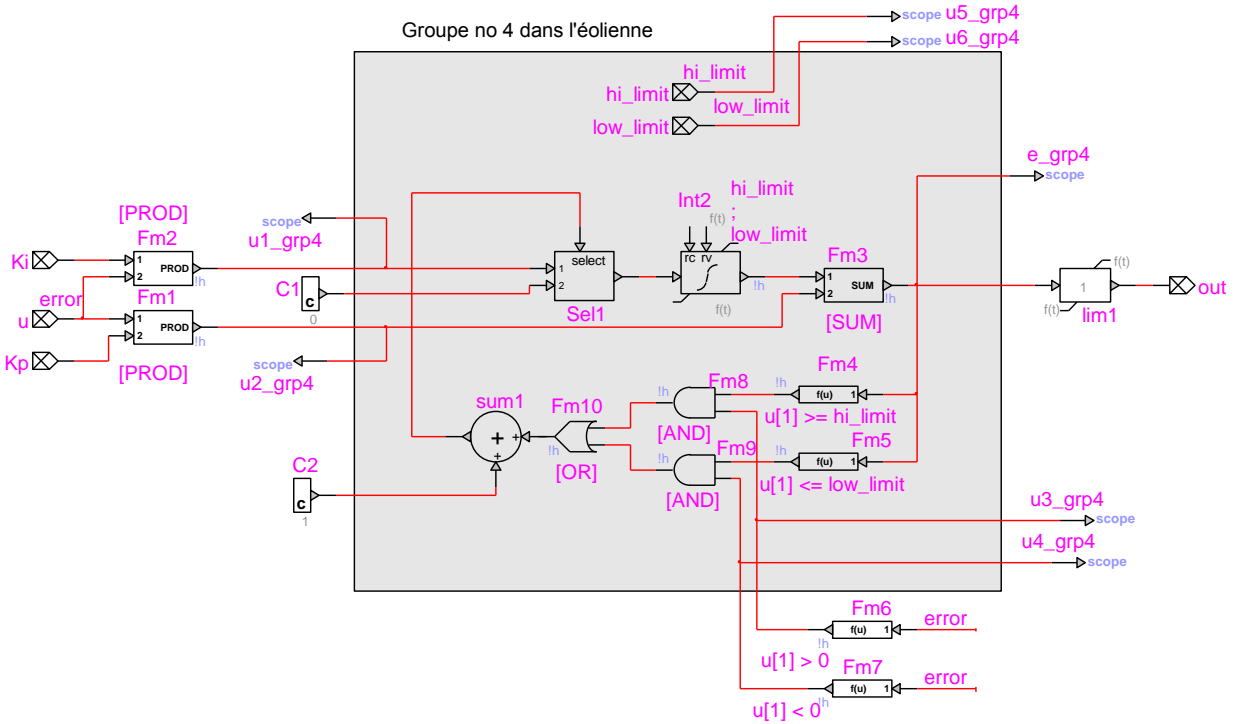


Figure 4-11: Le groupe no 4 (sous masque PI dans la Figure 4-10) dans le modèle d'éolienne dont la séquence des blocs est dans le Tableau 4.5

Le calcul de la norme spectrale, reproduit en annexe donne

$$\lambda = \left| \left(c1 \times (u5 \times u3 + u6 \times u4) - u1 \times (u5 \times u3 + u6 \times u4) \right) \times e + c1 \times (c2 - 1) - u1 \times (c2 - 2) \right|, \quad (4.9)$$

où les différentes valeurs sont, encore une fois, prises dans la simulation, notamment la valeur au seul PR du groupe, soit le bloc 'Fm3\$-983' dans le Tableau 4.5 et noté e_grp4 dans la Figure 4-11. Avec quelques manipulations (voir le script disponible en annexe), les entrées du groupe sont présentées dans la Figure 4-11. On a omis le suffixe « _grp4 » dans (4.9). Le résultat est présenté à la Figure 4-12.

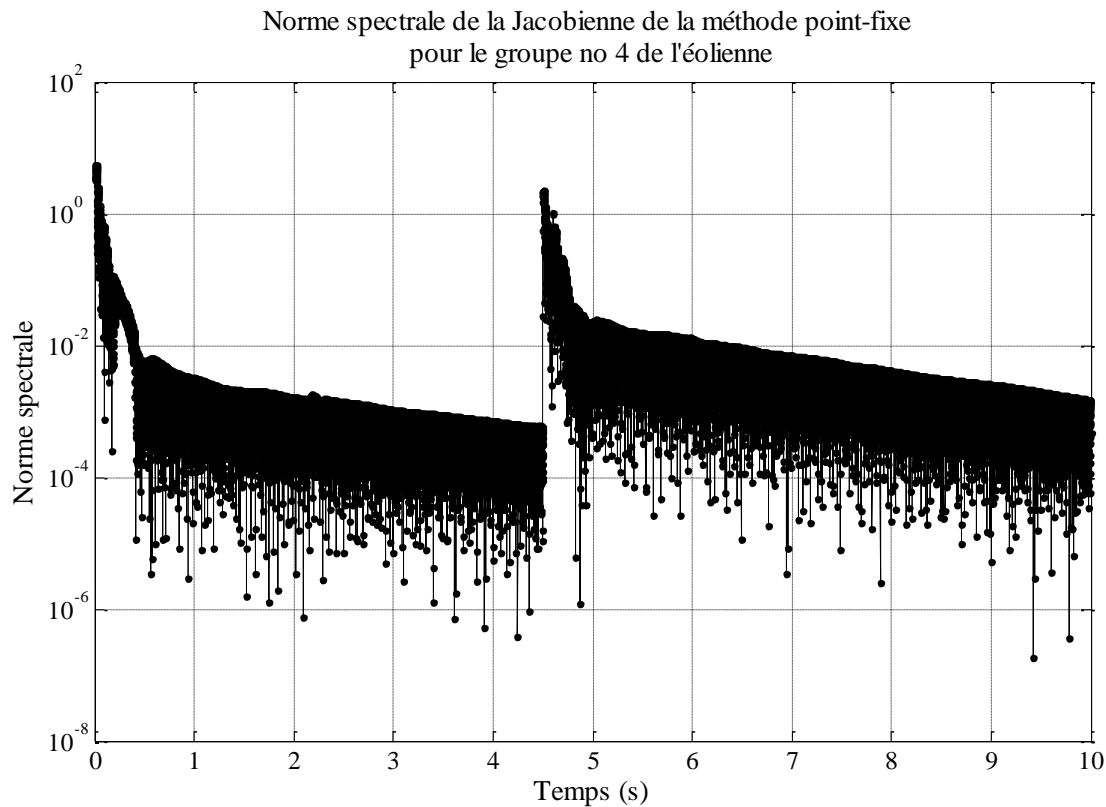


Figure 4-12: Variation de la norme spectrale en fonction du temps de simulation pour le groupe no 4 de l'éolienne installée à la barre 37 du système IEEE-39 barres

Par ailleurs, il faut se rappeler que les systèmes de contrôle sont de plus en plus digitaux, donc, discrets. Ils peuvent donc contenir des délais naturels pour la transmission de données ou pour la transformation des signaux de commande. Dans de tels cas, comme il a été indiqué, les propriétés contractantes assurent une norme inférieure à un pour le processus d'itérations à point-fixe. Cependant, comme on l'a appliqué avec la méthode combinant les itérations à point-fixe et la méthode DOGLEG, un critère de convergence doit être fixé pour répondre aux nécessités de grande précision des logiciels comme EMTP-RV pour répondre à l'analyse émise dans [4].

Le cas de la simulation de la machine électrique a montré les moins bonnes améliorations en gain de calcul, en comparaison à d'autres cas simulés. Ceci est dû, d'une part, au ratio du nombre des blocs sur le nombre des points de rétroactions PR (ratio N/M), et, d'autre part, au ratio de nombre des blocs non-linéaires et le nombre des blocs des points de rétroactions PR. En

comparant ce cas de la machine au cas de la centrale éolienne, dans le Tableau 4.2, ces ratios sont de (3.8, 2.4) pour la machine et de (8.7, 5) pour le deuxième groupe dans le modèle d'éoliennes.

Tableau 4.6: Nombres d'évaluation des sorties des blocs durant un processus-type d'itérations

Évaluation des sorties des blocs de contrôle	Mise à jour de la matrice Jacobienne $N \times M$ du Tableau 4.2	Calcul de la fonction objective	Total pour une itération	Total pour une convergence estimée à 4 itérations	
				Mise à jour de la Jacobienne	Méthode point-fixe
PLL	28	14	42	168	56
Machine asynchrone	2001	87	2088	8352	348
Centrale éolienne (pour les 3 types de groupe)	14	7	233	932	348
	78	26			
	6x9	6x9			

L'observation ci-dessus met en lumière un facteur important qui nuit à la bonne performance de la nouvelle méthode. Si le nombre des points de rétroaction est élevé et qu'il y a beaucoup de blocs dans le groupe, la mise à jour de la Jacobienne et les évaluations de la fonction objective (à chaque itération) requièrent plusieurs évaluations de sorties de blocs. Ceci est montré dans le Tableau 4.6. On y voit clairement que le processus itératif pour la machine nécessite le plus grand nombre d'évaluations des blocs. Donc, un coût plus grand en calcul avec la nouvelle méthode. Pour mitiger ceci, il serait très avantageux d'envisager des évaluations parallèles des éléments de la Jacobienne présentées dans l'équation (3.5). De plus, d'autres façons d'améliorer le coût d'évaluation des fonctions de sorties des blocs et d'utiliser l'agrégation des blocs si c'est possible (utilisée aussi dans MATLAB/Simulink), mais quand les signaux intermédiaires ne sont pas requis. Finalement les techniques similaires aux formules de Faà-Di-Bruno [62][63] peuvent être explorées pour le calcul de la dérivée numérique pour la Jacobienne via les fonctions composées le long du chemin de la boucle de rétroaction.

Nous allons clore ce chapitre en faisant un lien entre les chapitres suivants et la discussion sur la précision de la solution de la nouvelle approche avec celle de [4]. Comme mentionné, la tolérance absolue appliquée comme critère de convergence pour le cas de [4] devait être ajustée pour intégrer le fait que si la solution converge par la nouvelle approche DIPR, l'erreur absolue aux points PR augmente proportionnellement aux gains dans le chemin de rétroaction pour les variables à calculer séquentiellement. Un autre défi est alors de savoir si les conditions initiales

s'altèrent d'une approche à l'autre. L'analyse de cela est faite dans la dernière section du dernier chapitre. On aura recours au fameux théorème de Kantorovich. Dans le chapitre qui suit, deux autres aspects liés à la concordance de la solution dans les deux méthodologies sont discutés. Il s'agit de l'équivalence entre la formulation DIPR et la méthode de substitution des variables d'une part, et, d'autre part, de l'équivalence avec la représentation par les variables d'état.

4.5 Conclusion

Le but visé de ce chapitre était d'appliquer les variantes de la nouvelle méthode DIPR à des systèmes de contrôle représentatifs de cas couramment rencontrés. À cette fin, trois cas ont été étudiés. Le premier cas est le PLL, décrit au Chapitre 2; le deuxième cas est une machine asynchrone modélisée par un système d'équations de dimension 87×87 ; et finalement, le troisième cas, de très grande taille (environ 50000 équations), où 12 centrales éoliennes ont été intégrées au réseau test IEEE-39 barres. Dans tous les cas traités, les résultats avec la DIPR sont similaires à ceux obtenus par la méthode de référence considérée, soit pour chaque cas (itératif ou pas), la méthode correspondante dans [4]. Les gains en temps de calcul sont évalués en augmentant le pas de simulation jusqu'à la limite acceptable pour chaque variante de la méthode appliquée au cas considéré. Cette approche correspond au point de vue de l'utilisateur qui cherche à réduire son temps de simulation sans compromettre la viabilité des résultats. Ainsi, l'avantage relatif de chaque variante est exploité pour mesurer le gain en temps de simulation.

Les gains en temps de calcul, atteignant parfois 2, ont été enregistrés dans la plupart des cas pour une des variantes de la méthode DIPR proposée. Les limitations de certaines variantes ont été analysées. Par exemple, la performance de la nouvelle approche diminue quand le ratio du nombre des blocs PR sur tous les autres blocs (soit, M/N) augmente. Ceci a été analysé en détails sur la machine. La méthode DIPR est potentiellement plus avantageuse quand ce ratio est le plus bas car la taille de la Jacobienne est plus petite.

Pour pallier à la contrainte consistant à former ces Jacobiennes de grande taille, à chaque pas de simulation, la méthode point-fixe a été combinée à celle de Dogleg itératif. Avec un même critère de convergence pour les deux méthodes, la précision n'est pas compromise. Les avantages de la rapidité pour la première et de la robustesse pour la seconde sont exploités. Cette méthode hybride a été appliquée pour la solution des centrales éoliennes dans le réseau IEEE-39 barres.

Son gain en temps de simulation a été de près de 75%, dans certains cas. Il convient de mentionner que cette combinaison peut ne pas améliorer certains cas : le parcours des boucles, surtout si elles sont nombreuses, reste coûteux. C'est, encore fois, le cas de la machine où 23 boucles sont nécessaires. Toute méthode itérative appliquée sur ces derniers engendre un coût en calcul plus grand que la méthode non-itérative de [4]. Des améliorations pourront être apportées à l'avenir, entre autres, pour introduire des évaluations parallèles des fonctions objectives et des Jacobiennes, ou encore, d'autres aspects, tels que discuté dans les chapitres suivants.

Malgré ces cas limites, à souligner que l'approche DIPR offre plusieurs variantes alternatives aux méthodes classiques de représentation complète de système. À noter encore fois la formulation des boucles de rétroaction sous forme de problème à point-fixe.

CHAPITRE 5 COMPARAISON AVEC LA REPRÉSENTATION PAR VARIABLES D'ÉTAT

Dans ce chapitre, on compare la méthode DIPR à d'autres méthodes de solution de contrôle. On commence d'abord par la comparaison avec la méthode utilisant la représentation complète classique (telle qu'introduite dans [4]), mais en se concentrant cette fois-ci sur l'équivalence entre les deux en appliquant la substitution des variables, alors que dans les chapitres précédents il s'agissait de la comparaison des résultats et l'analyse des performances de deux méthodes. Mentionnons ici que représenter un système d'équations dans une formulation ou une autre ne devrait pas changer le résultat si les deux représentations sont conformes. Étant donné qu'une méthode de solution présente plus d'avantages dans une représentation qu'une autre, alors la convergence peut être plus difficile pour une représentation en comparaison d'une autre. On a vu en effet, entre autres, que la tolérance se propage différemment quand elle est appliquée sur quelques variables seulement ou sur toutes les variables. Cette analyse se veut donc un approfondissement de l'approche proposée. Un autre approfondissement consiste en la comparaison avec la représentation par variables d'état. Ceci est fait dans la deuxième section de ce chapitre.

5.1 Conformité des représentations complète et réduite par DIPR

Nous avons établi une représentation générale du système en (2.33) conforme avec la Figure 2-16 sous forme DIPR. On peut la reprendre ici avec les variables sous forme vectorielle :

$$\begin{cases} \mathbf{e} = \boldsymbol{\varphi}(\mathbf{u}, \mathbf{y}) \\ \mathbf{y} = \mathbf{G}(\mathbf{e}) \end{cases} \quad (5.1)$$

Nous avons mentionné dans la section 2.5 et, spécialement, dans l'analyse des coupures des boucles de la Figure 2-8 (b) qu'avec la coupure maximale (c'est-à-dire, avec toutes les sorties coupées), notée MEC, on a justement un cas équivalent au cas classique, c'est-à-dire, un cas où nous représentons toutes les relations entrée-sortie des blocs. Il est évident que la formulation DIPR d'un tel cas est équivalente au cas classique. En effet, pour la Figure 2-8 (b) on a les chemins des boucles G à une seule variable :

$$\left\{ \begin{array}{l} G_1(\mathbf{e}) = f_{\mathbf{p}_1}(\mathbf{e}) = f_1(\mathbf{e}) = 0 \\ G_2(\mathbf{e}) = f_{\mathbf{p}_2}(\mathbf{e}) = f_3(\mathbf{e}) = e_1 + e_4 \\ G_3(\mathbf{e}) = f_{\mathbf{p}_3}(\mathbf{e}) = f_3(\mathbf{e}) = 1 \times e_2 \\ G_4(\mathbf{e}) = f_{\mathbf{p}_4}(\mathbf{e}) = f_4(\mathbf{e}) = (e_3)^2 \end{array} \right. , \quad (5.2)$$

où chaque chemin est maintenant constitué d'un seul bloc celui précédent le bloc PR que nous considérons. Les équations à résoudre sont maintenant :

$$\left\{ \begin{array}{l} e_1 = 10 - G_1(\mathbf{e}) \\ e_2 = 0 + G_2(\mathbf{e}) \\ e_3 = 0 + G_3(\mathbf{e}) \\ e_4 = 0 + G_4(\mathbf{e}) \end{array} \right. , \quad (5.3)$$

où les valeurs de G_i sont en (5.2) et où Φ , apparaissant dans (5.1), est un ensemble des fonctions SOMME. Clairement, on a la forme (5.1) en posant :

$$\begin{aligned} \mathbf{u} &= [10, 0, 0, 0] \\ \mathbf{y} &= [-G_1(\cdot), G_2(\cdot), G_3(\cdot), G_4(\cdot)] \end{aligned} \quad (5.4)$$

En outre, le système composé avec (5.3) et (5.2) est équivalent à l'écriture des relations pour chacun de blocs (avec $\mathbf{e} = [x_1, x_2, x_3, x_4]$), car on a :

$$\left\{ \begin{array}{l} x_1 = 10 - G_1(x_1, x_2, x_3, x_4) = 10 \\ x_2 = 0 + G_2(x_1, x_2, x_3, x_4) = x_1 + x_4 \\ x_3 = 0 + G_3(x_1, x_2, x_3, x_4) = x_2 \\ x_4 = 0 + G_4(x_1, x_2, x_3, x_4) = x_3^2 \end{array} \right. , \quad (5.5)$$

La prochaine étape est maintenant de voir la conformité entre le système réduit et ceux en (5.3) ou (5.5). On a vu qu'il fallait couper en un seul point pour supprimer la boucle et écrire la seule équation résultante. Ceci a donné, en coupant en x_2 :

$$x_2 = \varphi(u, y) = u - f_4(f_3(x_2)), \quad (5.6)$$

Dans (5.5), en portant la troisième équation dans la quatrième, puis le tout dans la deuxième, mais aussi en portant la première dans la même deuxième, on a le résultat de (5.6). Ainsi, les deux formulations sont identiques. L'opération de substitution effectuée sur ce cas illustratif (inutile d'en présenter ici les détails car elle est bien connue pour résoudre le système) est aisément généralisée en remarquant qu'il s'agit en fait de l'équation du chemin de rétroaction pour la seule PR, notée e c'est-à-dire :

$$G_1(e) = f(e) = f_{\mathbf{p}(1), \mathbf{p}(2), \dots, \mathbf{p}(j) \dots \mathbf{p}(q-1), \mathbf{p}(Q)}(e), \quad (5.7)$$

où on a Q blocs dans le chemin de la boucle de rétroaction et leurs indices sont dans le vecteur \mathbf{p} . Par exemple, dans (5.6), $\mathbf{p} = [3, 4]$ et on écrit :

$$e = \varphi(u, y) = u - G_1(e) = u - \underbrace{f_4(f_3(e))}_{\text{substitution}}, \quad (5.8)$$

Pour le cas avec plus d'une fonction bloc PR, on applique la même procédure mais en ne substituant que les variables qui ne sont pas les sorties d'un bloc PR. De plus, cette substitution doit suivre l'ordre de la séquence topologique. On voit clairement que ceci revient à déterminer le deuxième sous-système d'équations en (5.1) qu'on reprend ici :

$$\mathbf{y} = \mathbf{G}(\mathbf{e}), \quad (5.9)$$

où, à gauche, ne restent que les variables des blocs PR. D'une manière formelle, la preuve peut se décliner comme dans les lignes qui suivent. On introduit l'opération de substitution sur tout système complet comme en (5.5), en supposant que cette substitution est valide, c'est-à-dire que chaque nouvelle représentation de la sortie d'un bloc utilise les sorties des autres blocs préalablement connues. On présume que cette opération aboutit à un système équivalent à celui non-réduit. Il suffit alors de montrer comment la représentation DIPR est une substitution valide sur un système complet, et donc, son résultat donne un système réduit équivalent au système complet pour un bon nombre de classes des systèmes.

Considérons un système d'équations d'un système de contrôle générique avec N variables. Les fonctions exprimant l'entrée et la sortie du bloc sont notées f_j et sont disposées selon un ordre donné (pas nécessairement selon la séquence topologique). Essayons de caractériser l'opération de substitution qui réduit la représentation du système, pas nécessairement jusqu'au

nombre minimal des variables, tout en garantissant l'équivalence avec le système initial. Notons cette opération de substitution σ sur un système d'équations quelconque. Plus précisément, on note $\sigma_{1,2,\dots,k,\dots,Q}$, la substitution à l'issue de laquelle Q variables du système sont écrites en fonction d'autres variables. Sur un système S de la forme générale :

$$\begin{cases} x_1 = f_1(x_1, x_2, \dots, x_i, \dots, x_{i+1}, \dots, x_{i+M}, \dots, x_j, \dots, x_N, \{C_l^1\}) \\ x_2 = f_2(x_1, x_2, \dots, x_i, \dots, x_{i+1}, \dots, x_{i+M}, \dots, x_j, \dots, x_N, \{C_l^2\}) \\ \vdots \\ x_j = f_j(x_1, x_2, \dots, x_i, \dots, x_{i+1}, \dots, x_{i+M}, \dots, x_j, \dots, x_N, \{C_l^j\}) \\ \vdots \\ x_N = f_N(x_1, x_2, \dots, x_i, \dots, x_{i+1}, \dots, x_{i+M}, \dots, x_j, \dots, x_N, \{C_l^N\}) \end{cases}, \quad (5.10)$$

et sur laquelle on applique la substitution, c'est-à-dire, symboliquement $\sigma_{1,2,\dots,k,\dots,Q}\langle S \rangle$, où Q variables sont substituées, M variables restantes (c'est-à-dire $Q = N - M$) non-substituées tout comme toutes les c_j constantes $\{C_l^j, l=1, \dots, c_j\}$, intervenant dans la fonction du bloc j . On

obtient le système $\overset{R}{S}$ réduit de la taille N à la taille M :

$$\begin{cases} x_i = f_i(x_i, x_{i+1}, \dots, x_{i+M}, \{C_l^i\}) \\ x_{i+1} = f_{i+1}(x_i, x_{i+1}, \dots, x_{i+M}, \{C_l^{i+1}\}) \\ x_{i+2} = f_{i+2}(x_i, x_{i+1}, \dots, x_{i+M}, \{C_l^{i+2}\}) \\ \vdots = \vdots \\ x_{i+M-1} = f_{i+M-1}(x_i, x_{i+1}, \dots, x_{i+M}, \{C_l^{i+M-1}\}) \\ x_M = f_M(x_i, x_{i+1}, \dots, x_{i+M}, \{C_l^M\}) \end{cases}. \quad (5.11)$$

Il est évident que $\overset{R}{S}$ est équivalent au premier système S car seules la substitution des variables a été faite. Il faut par contre y adjoindre l'opération inverse de substitution pour retrouver les Q autres variables. Notez que, jusqu'à présent, on n'a pas précisé encore l'ordre de la substitution. Notons cet ordre choisi comme :

$$\mathbf{p} = \{p1, p2, \dots, pQ\}. \quad (5.12)$$

L'exigence sous-entendue est telle que cet ordre ait assuré que la réduction par substitution soit possible, et donc, que S^R soit équivalent à S . Précisons que le nombre M a été arbitraire et n'est pas nécessairement minimal : on le considérera optimal. En effet, il pourrait y avoir d'autres substitutions possibles. Choisissons maintenant M égale au nombre des coupures oEC. Pour rappel, il s'agit du nombre optimal des coupures qui garantit un nombre nul de cycle dans le système.

On notera la substitution ainsi spécifiée par $\sigma_{1,2,\dots,k,\dots,Q}^{oEC}$, l'ordre n'étant toujours pas spécifié. Choisissons, justement, un ordre précis, celui qui découle de la relation (2.13) ayant été appliquée sur le PLL pour donner (2.22). Considérons la forme obtenue après regroupement pour toutes les séquences aboutissant à chacun des blocs PR de l'oEC. En d'autres termes, ce sont les séquences de la forme (2.35) appliquées dans (2.36). Notons $\sigma_{\emptyset(1,2,\dots,k,\dots,Q)}^{oEC-M}$ cette opération des substitutions où l'on maintient M variables équivalentes aux coupes de l'oEC et où les séquences (2.13) regroupées aboutissent aux M variables. On a établi que si la substitution est valide alors le système réduit est équivalent au système complet. Il suffit alors de montrer que la substitution $\sigma_{\emptyset(1,2,\dots,k,\dots,Q)}^{oEC-M}$ est valide : on aura ainsi l'équivalence de la méthode de DIPR et l'approche classique de représentation du système complet.

On montrera cela en deux étapes : le cas où chaque chemin de rétroaction est indépendant des autres, c'est-à-dire il n'y a pas de couplage entre les chemins de rétroaction et le cas où les chemins sont couplés comme pour le PLL traité précédemment.

Pour le premier cas, on peut réécrire la représentation sous une forme condensée en la Figure 5-1, où on a M coupes d'oEC et chacun des blocs PR a un ensemble d'entrées \mathbf{u} .

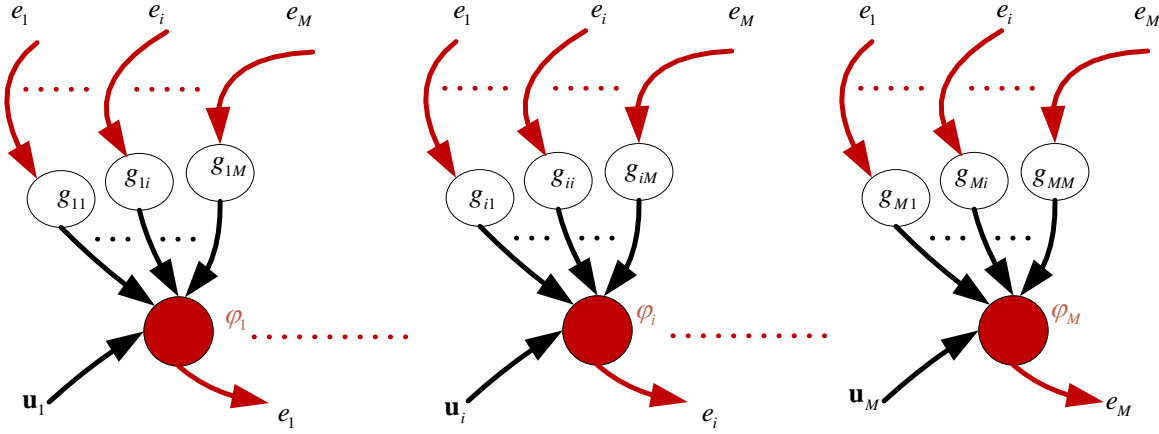


Figure 5-1: Décomposition générale aux points de rétroaction : cas des chemins de rétroaction indépendants

La fonction au PR φ est à plusieurs variables, dans le cas général, avec les chemins de tous les PR : e_1, \dots, e_M . Les chemins de rétroaction indépendants sont les g_{ik} formant un opérateur tel que les équations sont :

$$\begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_i \\ \vdots \\ e_M \end{bmatrix} = \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_i \\ \vdots \\ \varphi_M \end{bmatrix} \circ \left(\begin{bmatrix} g_{11}(\mathbf{e}) & g_{12}(\mathbf{e}) & \cdots & g_{1i}(\mathbf{e}) & \cdots & g_{1M}(\mathbf{e}) \\ g_{21}(\mathbf{e}) & g_{22}(\mathbf{e}) & \cdots & g_{2i}(\mathbf{e}) & \cdots & g_{2M}(\mathbf{e}) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ g_{i1}(\mathbf{e}) & g_{i2}(\mathbf{e}) & \cdots & g_{ii}(\mathbf{e}) & \cdots & g_{iM}(\mathbf{e}) \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ g_{M1}(\mathbf{e}) & g_{M2}(\mathbf{e}) & \cdots & g_{Mi}(\mathbf{e}) & \cdots & g_{MM}(\mathbf{e}) \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_i \\ \vdots \\ \mathbf{u}_M \end{bmatrix} \right), \quad (5.13)$$

où $\varphi_i \circ (.)$ indique que la fonction φ_i a les arguments $(.)$. Ces chemins de rétroaction ont été trouvés en faisant (2.13), (2.12) et (2.14), pour rappel, il s'agissait du chemin entre deux blocs, comme celui entre les blocs P1 et sum1 du PLL et exprimé en (2.4). Plus encore, on se rappelle que ce chemin a été défini en traitant l'accessibilité successive des blocs dans le chemin. Il devient alors évident que la substitution est possible. Précisément, elle s'exprimerait comme ceci pour le i -ième bloc PR :

$$\left\{ \begin{array}{l} g_{i1}(e_1) = \underset{\mathbf{p}i1}{f}(e_1) = f_{\mathbf{p}11(1), \mathbf{p}11(2), \dots, \mathbf{p}11(j), \dots, \mathbf{p}11(Q11-1), \mathbf{p}11(Q11)}(e_1) \\ g_{i2}(e_2) = \underset{\mathbf{p}i2}{f}(e_2) = f_{\mathbf{p}12(1), \mathbf{p}12(2), \dots, \mathbf{p}12(j), \dots, \mathbf{p}12(Q12-1), \mathbf{p}12(Q12)}(e_2) \\ \vdots \\ g_{ii}(e_i) = \underset{\mathbf{p}ii}{f}(e_i) = f_{\mathbf{p}11(1), \mathbf{p}11(2), \dots, \mathbf{p}11(j), \dots, \mathbf{p}11(Q11-1), \mathbf{p}11(Qii)}(e_i) \\ \vdots \\ g_{iM}(e_M) = \underset{\mathbf{p}iM}{f}(e_M) = f_{\mathbf{p}11(1), \mathbf{p}11(2), \dots, \mathbf{p}11(j), \dots, \mathbf{p}11(Q11-1), \mathbf{p}11(QiM)}(e_M) \\ e_i = \varphi_i(g_{i1}(e_1), g_{i2}(e_2), \dots, g_{ii}(e_i), \dots, g_{iM}(e_M)) \end{array} \right. , \quad (5.14)$$

où Qik est le nombre des blocs rencontrés dans les chemins i vers k , $\mathbf{p}ik$ est précisément la séquence du chemin de rétroaction établie selon les relations de la matrice d'accessibilité entre les blocs i et k (les fameuses (2.13), (2.12) et (2.14)) qui sont combinées pour donner :

$$\mathbf{p}_{ik} = \mathbf{F}(\mathbf{F}_i(\text{Ligne } \mathbf{R}(i) \otimes \text{Colonne } \mathbf{R}(k)^T)), \quad (5.15)$$

où la matrice \mathbf{R} définit l'accessibilité, \mathbf{F}_i une énumération d'entiers (1 : N) et \mathbf{F} les indices des blocs. Chaque fonction composée découlant de ces séquences d'accessibilité est clairement une substitution valide entre les différentes variables commençant par chaque sortie d'un bloc PR vers chacune des entrées des blocs PR. Ainsi donc, on a les opérations de substitution $\sigma_{\varphi(\mathbf{p}_{ik})}^{oEC-M}$ qui forment le système réduit en (5.13) à partir du système complet (5.10). D'où l'équivalence entre la nouvelle formulation DIPR et le système classique de représentation complète du système de contrôle pour le cas de chemins de rétroaction indépendants.

Maintenant, on analyse l'autre cas où on a des chemins de rétroaction couplés. La validité de la substitution n'est pas évidente car il faut, pour chaque rétroaction aboutissant au bloc PR, considérer plusieurs chemins venant de sorties des blocs PR et se combinant (voir la Figure 5-2). C'est justement cette combinaison que nous devons valider. Dans la Figure 5-2, on a les sous-ensembles des séquences : $g_{i1}, \dots, g_{ii}, \dots, g_{iM}$, $G^{i1}, \dots, G^{ii}, \dots, G^{iK}$ qui sont regroupés à droite de la figure en G_i pour chacun de i -ième bloc PR.

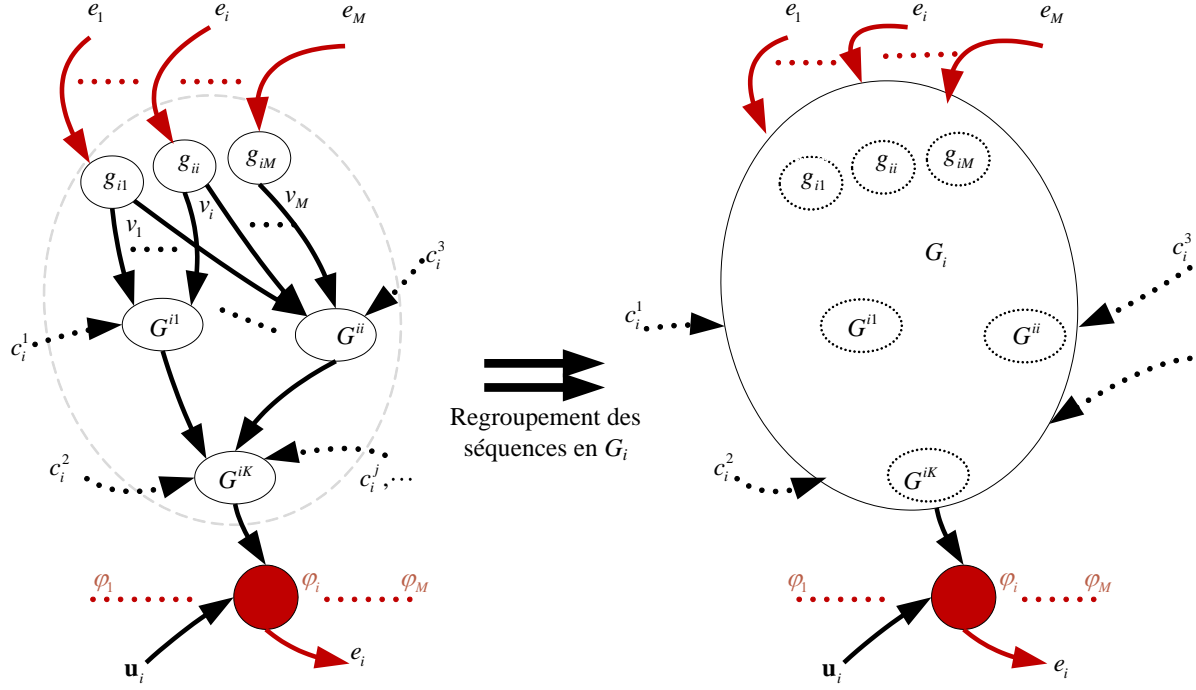


Figure 5-2: Décomposition générale aux points de rétroaction avant et après regroupement des chemins de rétroaction couplés.

Formellement, introduisons une notation :

$$G_i = LT \left\langle \bigcup_{j=1}^M \mathbf{p}_{ij} \right\rangle, \quad (5.16)$$

où $LT \langle . \rangle$ est un arrangement selon la liste topologique, $\bigcup_{j=1}^M \mathbf{p}_{ij}$ est une union des ensembles \mathbf{p}_{ij} .

Ces derniers sont, encore une fois, évalués selon (5.15). Ce regroupement dans la Figure 5-2 a déjà été introduit en l'illustrant sur le PLL dans (2.35). La séquence regroupée G_i comporte des éléments indépendants d'un chemin g_{ij} et des éléments communs G^{ij} . Avec les séquences g_{ij} , on peut compléter une étape intermédiaire en introduisant les variables v_i , (voir en la Figure 5-2), à gauche :

$$\begin{cases} v_{i1} = g_{i1}(e_1) \\ v_{i2} = g_{i2}(e_2) \\ \vdots \\ v_{ii} = g_{ii}(e_i) \\ \vdots \\ v_{iM} = g_{iM}(e_M) \end{cases} . \quad (5.17)$$

Enfin, l'étape finale consiste à écrire pour les séquences couplées les relations entrées-sorties selon la LT du graphe. Soit pour le i -ième bloc et pour chacune des séquences dans les sous-ensembles G^{ij} , on a :

$$\begin{array}{l} w_{i1} = G^{i1}(v_{i1}, v_{i2}, \dots, v_{ii}, \dots, v_{iM}, \{C_i^l\}) \\ \downarrow \\ w_{i2} = G^{i2}(w_{i1}, v_{i1}, v_{i2}, \dots, v_{ii}, \dots, v_{iM}, \{C_i^l\}) \\ \downarrow \\ \vdots \\ w_{ij} = G^{ij}(w_{i1}, w_{i2}, \dots, w_{ij-1}, v_{i1}, v_{i2}, \dots, v_{ii}, \dots, v_{iM}, \{C_i^l\}) \\ \vdots \\ w_{iK} = G^{iK}(w_{i1}, w_{i2}, \dots, w_{ij}, \dots, w_{iK-1}, v_{i1}, v_{i2}, \dots, v_{ii}, \dots, v_{iM}, \{C_i^l\}) \\ y_i = w_{iK} \end{array} \quad (5.18)$$

Selon la sequence
de la LT

↓

En portant toutes les relations autres que la dernière dans (5.18), selon l'ordre inverse de la LT, et en portant les fonctions des séquences de chemins non-couplés de (5.17), on peut établir clairement l'équation de tous les chemins de rétroaction. Cette opération est illustrée dans (5.19) :

$$\begin{array}{l}
w_{iK} = G^{iK}(\mathbf{w}_{1:iK-1}, \mathbf{v}, \{C_i^l\}) \\
\downarrow \\
w_{iK} = G^{iK}(w_{i1}, w_{i2}, \dots, w_{ij}, \dots, G^{iK-1}(\mathbf{w}_{1:iK-2}, \mathbf{v}), \mathbf{v}, \{C_i^l\}) \\
\downarrow \\
\vdots \\
w_{iK} = G^{iK}(w_{i1}, w_{i2}, \dots, G^{ij}(\mathbf{w}_{1:ij-1}, \mathbf{v}), \dots, G^{iK-1}(\mathbf{v}), \mathbf{v}, \{C_i^l\}) \\
\downarrow \\
\text{Substitution } \vdots \\
w_{iK} = G^{iK}(w_{i1}, w_{i2}, G^{i3}(\mathbf{w}_{1:i2}, \mathbf{v}), \dots, G^{ij}(\mathbf{v}), \dots, G^{iK-1}(\mathbf{v}), \mathbf{v}, \{C_i^l\}) \\
\downarrow \\
w_{iK} = G^{iK}(w_{i1}, G^{i2}(\mathbf{w}_{1:i1}, \mathbf{v}), G^{i3}(\mathbf{v}), \dots, G^{ij}(\mathbf{v}), \dots, G^{iK-1}(\mathbf{v}), \mathbf{v}, \{C_i^l\}) \\
\downarrow \\
w_{iK} = G^{iK}(G^{i1}(\mathbf{v}), G^{i2}(\mathbf{v}), G^{i3}(\mathbf{v}), \dots, G^{ij}(\mathbf{v}), \dots, G^{iK-1}(\mathbf{v}), \mathbf{v}, \{C_i^l\}) \\
y_i = w_{iK}
\end{array} \quad , \quad (5.19)$$

où le vecteur $\mathbf{w}_{1:it} = [w_{i1}, \dots, w_{it}]$ et le vecteur \mathbf{v} est tel que :

$$\mathbf{v} = [v_{i1}, v_{i2}, \dots, v_{ii}, \dots, v_{iM}] = [g_{i1}(e_1), g_{i2}(e_2), \dots, g_{ii}(e_i), \dots, g_{iM}(e_M)] \quad (5.20)$$

En portant (5.20) dans (5.19), on a :

$$y_i = G^{iK} \left(\begin{array}{l} G^{i1}(g_{i1}(e_1), g_{i2}(e_2), \dots, g_{ii}(e_i), \dots, g_{iM}(e_M)), \\ G^{i2}(g_{i1}(e_1), g_{i2}(e_2), \dots, g_{ii}(e_i), \dots, g_{iM}(e_M)), \\ G^{i3}(g_{i1}(e_1), g_{i2}(e_2), \dots, g_{ii}(e_i), \dots, g_{iM}(e_M)) \\ \dots, \\ G^{ij}(g_{i1}(e_1), g_{i2}(e_2), \dots, g_{ii}(e_i), \dots, g_{iM}(e_M)), \\ \dots, \\ G^{iK-1}(g_{i1}(e_1), g_{i2}(e_2), \dots, g_{ii}(e_i), \dots, g_{iM}(e_M)), \\ (g_{i1}(e_1), g_{i2}(e_2), \dots, g_{ii}(e_i), \dots, g_{iM}(e_M)), \{C_i^l\} \end{array} \right), \quad (5.21)$$

que l'on peut noter :

$$y_i = G_i(g_{i1}(e_1), g_{i2}(e_2), \dots, g_{ii}(e_i), \dots, g_{iM}(e_M)) \quad (5.22)$$

ou encore ;

$$y_i = G_i(e_1, e_2, \dots, e_i, \dots, e_M), \quad (5.23)$$

Car les séquences des fonctions composées g_{ij} sont découplées comme on peut le voir à gauche dans la Figure 5-2.

Il suffit maintenant de voir qu'en respectant l'hypothèse selon laquelle toutes les séquences suivent la LT, on peut sans crainte substituer de bloc à bloc les entrées-sorties de la chaîne, en commençant par les variables dans G_i dans (5.23).

Établissant (5.23) à partir des substitutions dans le système classique et en combinant cette dernière avec les équations des blocs PR on forme donc (5.1) qui est le système réduit par DIPR.

En récapitulant, on a montré successivement que pour trois cas possibles d'un seul chemin de rétroaction, de plusieurs chemins indépendants et de plusieurs chemins dépendants (couplés), l'opération de substitution de toutes variables selon la LT à l'exception de celles associés aux blocs de point de rétroaction est conforme à la représentation du système sous forme classique complète. Mentionnons que pour le cas où on a aucune boucle de rétroaction, les sorties successives des blocs sont directement évaluées séquentiellement en commençant par les variables connues. On n'a pas traité de ce cas simple dans cette thèse. D'autres cas où les sorties de blocs en fonction des entrées sont discontinues ou fortement non-linéaires nécessiteront des précautions particulières dans l'évaluation des chemins de la boucle de rétroaction.

Bien que le système formulé par la DIPR puisse découler des opérations de substitution sur une formulation de [4], il est montré dans le dernier chapitre que, les caractéristiques du processus itératif de Newton (avec ses conditions initiales choisies telles que postulé par le théorème de Kantorovich) peuvent différer pour les deux approches. Ce constat peut d'ailleurs s'appliquer dans un contexte plus large. Il est, par exemple rappelé aussi, dans [66], que réécrire un système d'équation dans une forme ou une autre, conforme à une formulation donnée, peut résulter en des différences non négligeables quant aux performances attendues par une méthode de solution sur les deux systèmes. Cet aspect est un angle d'approfondissement de l'analyse de l'approche DIPR.

Un autre angle d'approfondissement est la comparaison ou le lien entre la DIPR et la représentation par variables d'état. Il constitue l'objet de la section suivante.

5.2 Comparaison avec la représentation par variables d'état

La représentation DIPR adoptée ici et la représentation usuelle des systèmes de contrôle dans l'espace des états ont une certaine équivalence, mais aussi des particularités propres à chacune. Cette discussion réfère aussi bien à la comparaison entre une représentation complète de la forme (5.10) et celle par variables d'état.

Avec le passage en temps discret, on a eu les équations :

$$\mathbf{x}(k+1) = \mathbf{f}(k, \mathbf{x}(k), \mathbf{u}), \quad (5.24)$$

Si \mathbf{f} est linéaire (ou, peut être linéarisé autour d'un instant k) et qu'en plus elle est indépendante du temps. Ceci donnera :

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{a}\mathbf{x}(k) + \mathbf{b}\mathbf{u}(k) \\ \mathbf{y}(k+1) = \mathbf{c}\mathbf{x}(k) + \mathbf{d}\mathbf{u}(k) \end{cases} \quad (5.25)$$

avec \mathbf{y} , le vecteur des sorties choisies. Notez que son évaluation nécessite la solution simultanée d'un ensemble d'équations faisant intervenir tous les blocs inter-reliés ; et avec k , qui est une mesure d'écoulement du temps (c'est le pas).

5.2.1 Variables d'état, chacune dépendant d'un seul point de rétroaction

Un premier cas est celui pour lequel les variables d'état sont associées aux points de rétroaction des boucles, c'est-à-dire qu'ils sont explicitement fonction de seuls inconnus PR (c'est-à-dire qu'elles sont dans le vecteur \mathbf{e}) dans le système simultané d'équations formulé par DIPR. Ceci est le cas de la Figure 5-3.

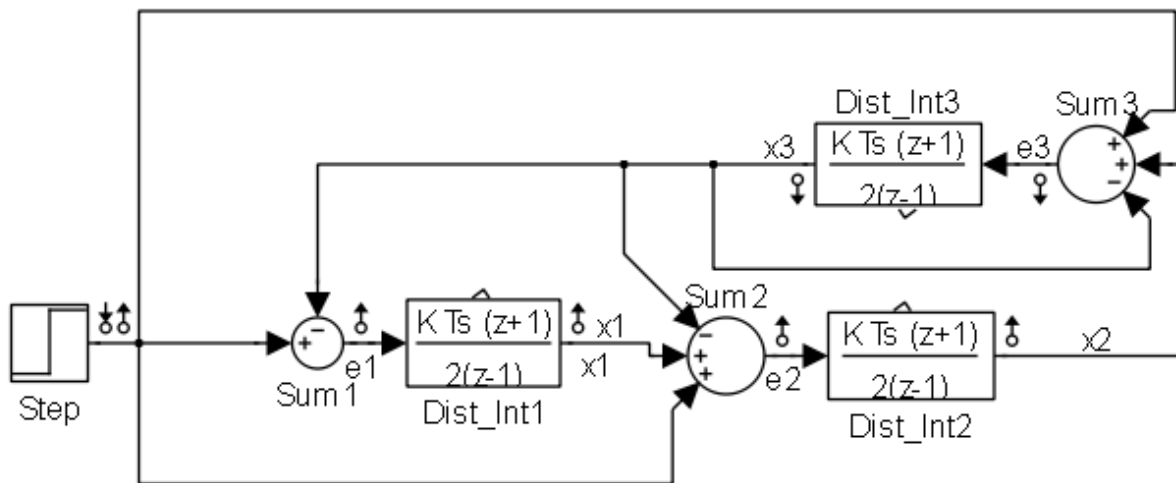


Figure 5-3: Cas simple linéaire pour 3 variables d'état avec 3 boucles de rétroaction : similarité entre DIPR et représentation par matrices ABCD dans MATLAB/Simulink

En mode discret, on introduit l'intégration trapézoïdale pour les fonctions intégrales, soit pour les blocs Dist_Int1, Dist_Int2 et Dist_Int3 de la Figure 5-3. Les variables d'état du système sont les sorties de ces trois blocs car elles dépendent directement des termes d'histoire, à cause de l'intégration trapézoïdale. Ils sont représentés par la transformée en Z dans la Figure 5-3.

La solution dans l'espace des états requiert une linéarisation à chaque instant. À 2 secondes pour

un pas de temps de $250\mu s$, c'est-à-dire $k = \frac{t}{\Delta t} = \frac{2s}{250 \times 10^{-6}s} = 8000$, on a les matrices en la

Figure 5-4 déduites par l'outil d'analyse linéaire dans MATLAB/Simulink. Il y a une boucle algébrique avec trois variables associées aux blocs Sum2, Sum3 et Sum1 (dans cet ordre) détectée par MATLAB/Simulink. On peut vérifier que les coupures à ces derniers assurent zéro cycle dans le graphe du système. Elles peuvent donc servir à la formulation DIPR. On nomme d'ailleurs ces variables des PR : e1, e2 et e3.

```
>> Model_sys
```

a(k)=

	Dist_Int1	Dist_Int2	Dist_Int3
Dist_Int1	1	-3.125e-008	-0.00025
Dist_Int2	0.00025	1	-0.00025
Dist_Int3	3.125e-008	0.00025	0.9998

b(k) =

	Step (pt. 1)
Dist_Int1	0.00025
Dist_Int2	0.00025
Dist_Int3	0.00025

c(k) =

	Dist_Int1	Dist_Int2	Dist_Int3
Dist_Int1 (p	1	-1.562e-008	-0.000125
Dist_Int2 (p	0.000125	1	-0.000125
Dist_Int3 (p	1.562e-008	0.000125	0.9999
Sum1 (pt. 1)	-1.562e-008	-0.000125	-0.9999
Sum2 (pt. 1)	1	-0.000125	-1
Sum3 (pt. 1)	0.000125	0.9999	-1
Step (pt. 1)	0	0	0

d(k) =

	Step (pt. 1)
Dist_Int1 (p	0.000125
Dist_Int2 (p	0.000125
Dist_Int3 (p	0.000125
Sum1 (pt. 1)	0.9999
Sum2 (pt. 1)	1
Sum3 (pt. 1)	1
Step (pt. 1)	1

Sampling time: 0.00025
Discrete-time model.

Figure 5-4: Matrices ABCD pour le cas simple linéaire avec 3 variables d'état et 3 boucles de rétroaction. Linéarisation à 2 secondes avec un pas de 250 μ s.

Les formulations de la boucle algébrique dans l'espace des états et celle DIPR donne le même système d'équations simultanées pour le cas simple de la Figure 5-3 pour le temps continu :

$$\begin{cases} e2 = -\left(\int e3\right) + \left(\int e1\right) + u \\ e3 = -\left(\int e3\right) + \left(\int e2\right) + u . \\ e1 = -\left(\int e3\right) + u \end{cases} \quad (5.26)$$

Avec la discrétisation, les fonctions intégrales par la méthode trapézoïdale modifiée dans Simulink [40] comme :

$$sortie = sortie_histoire + \left(\frac{\Delta T}{2}\right) \times Entrée_histoire + \left(\frac{\Delta T}{2}\right) \times Entrée , \quad (5.27)$$

où ΔT est le pas de temps (noté aussi T_s « sampling time »)

On déduit la Jacobienne du système (5.26) à un point k avec $\mathbf{e}(k) = [e2, e3, e1]_k$, cela donne :

$$\mathbf{J} = \begin{pmatrix} 1 & \frac{\Delta T}{2} & -\frac{\Delta T}{2} \\ -\frac{\Delta T}{2} & 1 + \frac{\Delta T}{2} & 0 \\ 0 & \frac{\Delta T}{2} & 1 \end{pmatrix}. \quad (5.28)$$

La Jacobienne numérique (utilisant une perturbation numérique) correspondante à $k = 8000$ est calculée en mode débogage de MATLAB/Simulink (voir dFdX) dans la Figure 5-5.

Le processus de solution de la boucle algébrique avec les méthodes de type Newton (que l'on a utilisées aux Chapitre 3 et Chapitre 4) est entièrement défini par la Jacobienne et les équations des contraintes aux blocs PR, comme en (5.26).

S'il y a d'autres blocs dans le système, ils sont inclus dans le processus de solution par évaluation de leur sortie selon la séquence de la liste d'exécution des méthodes des blocs dans MATLAB/Simulink (voir la Figure 5-7) ou selon l'évaluation séquentielle d'après les listes topologiques de chaque chemin de rétroaction dans la méthode DIPR.

```

dFdx = [
    1                                0.0001249995333550583 -0.0001249303570769729
    -0.0001249826089941018         1.000124999533355                    0
    0                                0.0001249995333550583                    1]
z[0] = 5.404074856948878 g[0] = 1.519380177228413e-010
z[1] = 8.748681733395751 g[1] = 3.61879415322619e-011
z[2] = 0.3346474227782339 g[2] = -3.910205492729801e-013
Successfully solved loop in 1 iterations.
z[0] = 5.404074856948878 g[0] = 1.519380177228413e-010
z[1] = 8.748681733395751 g[1] = 3.61879415322619e-011
z[2] = 0.3346474227782339 g[2] = -3.910205492729801e-013
%-----%
[TM = 2 ] simple_3_etats_lineaire.Outputs.Major
(sldebug @31): >>

```

Figure 5-5: Matrice Jacobienne dFdx et itérations pour le cas simple linéaire avec 3 variables d'état et 3 boucles de rétroaction. Linéarisation à 2 secondes avec un pas de 250µs

On représente maintenant les équations d'état classiques en écrivant cette fois le système (5.26) avec les variables d'état x_1 , x_2 et x_3 .

Notez, d'une part, que dFdx est exactement égale à **J**. Et d'autre part, en principe, les éléments de la Jacobienne sont constants car l'intégrateur discrétisé est linéaire. À différents temps, ils sont légèrement différents. En effet, pour le point de temps suivant, à $2s+250\mu s$, on a le nouveau dFdx dans la Figure 5-6.

```

dFdx = [
    1                                0.0001250046634887812 -0.0001249454059598024
    -0.0001250091945288326         1.000125004663489                    0
    0                                0.0001250046634887812                    1]
z[0] = 5.401971179498124 g[0] = 1.233004809364502e-010
z[1] = 8.747845423204103 g[1] = -1.544364636174578e-011
z[2] = 0.3324603568879505 g[2] = 3.89999144090325e-012
Successfully solved loop in 1 iterations.
z[0] = 5.401971179498124 g[0] = 1.233004809364502e-010
z[1] = 8.747845423204103 g[1] = -1.544364636174578e-011
z[2] = 0.3324603568879505 g[2] = 3.89999144090325e-012
%-----%
[TM = 2.00025 ] simple_3_etats_lineaire.Outputs.Major

```

Figure 5-6: Matrice Jacobienne dFdx et itérations pour le cas simple linéaire avec 3 variables d'état et 3 boucles de rétroaction. Linéarisation à 2.00025 secondes avec un pas de 250µs


```

[TM = 2          ] simple_3_etats_lineaire_dessin.Outputs.Major
(sldebug @26): >> elist gcs

RootSystem.Start 'simple_3_etats_lineaire_dessin':
  0:1 DiscreteIntegrator.Start 'simple_3_etats_lineaire_dessin/Dist_Int3'
  0:2 DiscreteIntegrator.Start 'simple_3_etats_lineaire_dessin/Dist_Int1'
  0:3 Sum.Start 'simple_3_etats_lineaire_dessin/Sum2' [ algVar ]
  0:4 DiscreteIntegrator.Start 'simple_3_etats_lineaire_dessin/Dist_Int2'
  0:5 Sum.Start 'simple_3_etats_lineaire_dessin/Sum3' [ algVar ]
  0:6 Sum.Start 'simple_3_etats_lineaire_dessin/Sum1' [ algVar ]

RootSystem.Initialize 'simple_3_etats_lineaire_dessin':
  0:1 DiscreteIntegrator.Initialize 'simple_3_etats_lineaire_dessin/Dist_Int3'
  0:2 DiscreteIntegrator.Initialize 'simple_3_etats_lineaire_dessin/Dist_Int1'
  0:4 DiscreteIntegrator.Initialize 'simple_3_etats_lineaire_dessin/Dist_Int2'

RootSystem.Enable 'simple_3_etats_lineaire_dessin':
  0:1 DiscreteIntegrator.Enable 'simple_3_etats_lineaire_dessin/Dist_Int3'
  0:2 DiscreteIntegrator.Enable 'simple_3_etats_lineaire_dessin/Dist_Int1'
  0:4 DiscreteIntegrator.Enable 'simple_3_etats_lineaire_dessin/Dist_Int2'

RootSystem.Outputs 'simple_3_etats_lineaire_dessin' [tid=0,1] :
  0:0 Step.Outputs 'simple_3_etats_lineaire_dessin/u' [tid=0]
  0:1 DiscreteIntegrator.Outputs 'simple_3_etats_lineaire_dessin/Dist_Int3' [tid=1]
  0:2 DiscreteIntegrator.Outputs 'simple_3_etats_lineaire_dessin/Dist_Int1' [tid=1]
  0:3 Sum.Outputs 'simple_3_etats_lineaire_dessin/Sum2' [ algVar ] [tid=0]
  0:4 DiscreteIntegrator.Outputs 'simple_3_etats_lineaire_dessin/Dist_Int2' [tid=1]
  0:5 Sum.Outputs 'simple_3_etats_lineaire_dessin/Sum3' [ algVar ] [tid=0]
  0:6 Sum.Outputs 'simple_3_etats_lineaire_dessin/Sum1' [ algVar ] [tid=0]

RootSystem.Update 'simple_3_etats_lineaire_dessin' [tid=0,1] :
  0:1 DiscreteIntegrator.Update 'simple_3_etats_lineaire_dessin/Dist_Int3' [tid=1]
  0:2 DiscreteIntegrator.Update 'simple_3_etats_lineaire_dessin/Dist_Int1' [tid=1]
  0:4 DiscreteIntegrator.Update 'simple_3_etats_lineaire_dessin/Dist_Int2' [tid=1]

(sldebug @26): >>

```

Figure 5-7: Listes d'exécution des méthodes dans MATLAB/Simulink pour le cas simple linéaire avec 3 variables d'état et 3 boucles de rétroaction (la mise à jour des états et le calcul des variables algébriques présentés en rouge)

Cette différence découle de la définition de la discrétisation et de la mise à jour particulière des variables. En effet, pour les blocs intégrateurs discrétisés via cette méthode, voici le processus qu'utilise MATLAB/Simulink [40]:

« Step » k :

$$y(k) = x(k) + K * \Delta T / 2 * u(k)$$

$$x(k+1) = y(k) + K * \Delta T / 2 * u(k)$$

Les équations de calcul de sortie y et la mise à jour des variables d'état sont légèrement modifiés pour prendre la forme suivante :

$$\begin{aligned} y1(k) &= x1(k) - \frac{\Delta T}{2} y3(k) + \frac{\Delta T}{2} u(k) \\ y2(k) &= x2(k) + \frac{\Delta T}{2} y2(k) - \frac{\Delta T}{2} y3(k) + \frac{\Delta T}{2} u(k) \\ y3(k) &= x3(k) + \frac{\Delta T}{2} y2(k) - \frac{\Delta T}{2} y3(k) + \frac{\Delta T}{2} u(k) \end{aligned} \quad (5.29)$$

En substituant les sorties, jusqu'à un certain ordre (on a arrêté pour obtenir une précision de $1e-6$), et n'écrire que les relations avec les variables d'état, on a, en omettant l'indice k pour le pas de temps discret :

$$\begin{aligned} y1 &= \left(1 - \frac{\Delta T^3}{8} + \frac{\Delta T^4}{16}\right) x1 + \left(-\frac{\Delta T^2}{4} + \frac{T^3}{8}\right) x2 + \left(\frac{\Delta T}{2} - \frac{\Delta T^2}{4} + \frac{\Delta T^4}{16}\right) x3 \\ &+ \left(\frac{\Delta T}{2} - \frac{\Delta T^2}{4}\right) u + \frac{\Delta T^4}{16} (e1 + e3) \\ y2 &= \left(\frac{\Delta T}{2} - \frac{\Delta T^3}{8} + \frac{\Delta T^5}{32}\right) x1 + \left(1 - \frac{\Delta T^2}{4} + \frac{\Delta T^4}{16}\right) x2 + \left(-\frac{\Delta T}{2} + \frac{\Delta T^3}{8} + \frac{\Delta T^4}{16} + \frac{\Delta T^5}{32}\right) x3 \\ &+ \left(\frac{\Delta T}{2} - \frac{\Delta T^3}{8}\right) u + \left(\frac{\Delta T^4}{8} + \frac{\Delta T^5}{16}\right) (e1 + e3) \\ y3 &= \left(\frac{\Delta T^2}{4} + \frac{\Delta T^3}{8}\right) x1 + \left(\frac{\Delta T}{2} - \frac{\Delta T}{4}\right) x2 + \left(1 - \frac{\Delta T}{2} - \frac{\Delta T^3}{8}\right) x3 \\ &+ \left(\frac{\Delta T}{2}\right) u - \frac{\Delta T^3}{8} (e1 + e3) \end{aligned} \quad (5.30)$$

La mise à jour des variables d'état devient alors :

$$\begin{aligned}
x1(k+1) &= y1(k) + \frac{\Delta T}{2}(u(k) - y3(k)) \\
x2(k+1) &= y2(k) + \frac{\Delta T}{2}(y2(k) - y3(k) + u(k)) \\
x3(k+1) &= y3(k) + \frac{\Delta T}{2}(y2(k) - y3(k) + u(k))
\end{aligned} \tag{5.31}$$

Avec la définition du temps discret, on a eu en (1.10):

$$\mathbf{x}(k+1) = \tilde{\mathbf{f}}[\mathbf{x}(k), \mathbf{u}(k)]. \tag{5.32}$$

Les sorties en fonction des variables d'état actuelles sont :

$$\mathbf{y}(k) = \mathbf{g}[\mathbf{x}(k), \mathbf{u}(k)]. \tag{5.33}$$

On déduit les matrices ABCD usuelles en se conformant à la définition [40] :

$$J_{ABCD} = \begin{pmatrix} \frac{\partial \mathbf{f}}{\partial \mathbf{x}} & \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \\ \frac{\partial \mathbf{g}}{\partial \mathbf{x}} & \frac{\partial \mathbf{g}}{\partial \mathbf{u}} \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix}, \tag{5.34}$$

où pour la fonction continue \mathbf{f} on substitue la version discrétisée en (5.32).

Pour retrouver les résultats de la discrétisation dans la Figure 5-4, on construit (5.33) avec (5.30) et (5.32) par (5.32), dans laquelle on a porté (5.30) pour n'avoir que les variables d'état actuelles. Les résultats sont en annexe. Les matrices ABCD sont identiques pour les lignes disponibles pour le calcul numérique et la discrétisation par MATLAB/Simulink.

On voit cependant qu'à part ce processus de mise à jour des variables d'état et le calcul des sorties, tel que montré en suivant les étapes de la solution MATLAB/Simulink dans la Figure 5-7, la solution de la boucle algébrique est équivalente à celle de la DGIPR. Pour ce cas simple quoique bien instructif, il s'agit d'équations en (5.26). Toutes les autres variables sont évaluées selon la séquence des listes topologiques.

Pour le cas où les variables d'état sont fonction de plusieurs chemins de rétroaction, la similarité n'est pas difficile à montrer. Un cas illustratif est montré ci-bas.

5.2.2 Variables d'état dépendant de plus d'un point de rétroaction

Un deuxième cas est celui où des variables d'état ne sont pas explicitement dépendantes d'un seul point de rétroaction. La solution de la boucle algébrique implique donc un calcul implicite de toutes les variables incluant les variables d'état.

Pour montrer les similarités et différences entre cette approche par variables d'état et celle introduite ici, nous traitons le cas de PLL présenté plus haut. En mode discret, on introduit l'intégration trapézoïdale pour les blocs Int1, Int2 et Int3, soit respectivement Dis_int1, Dis_int2 et Dis_int3. Les variables d'état du système sont les sorties de ces trois blocs car elles dépendent directement des termes d'histoire, à cause de l'intégration trapézoïdale. Ils sont représentés par la transformée en Z dans la Figure 5-8.

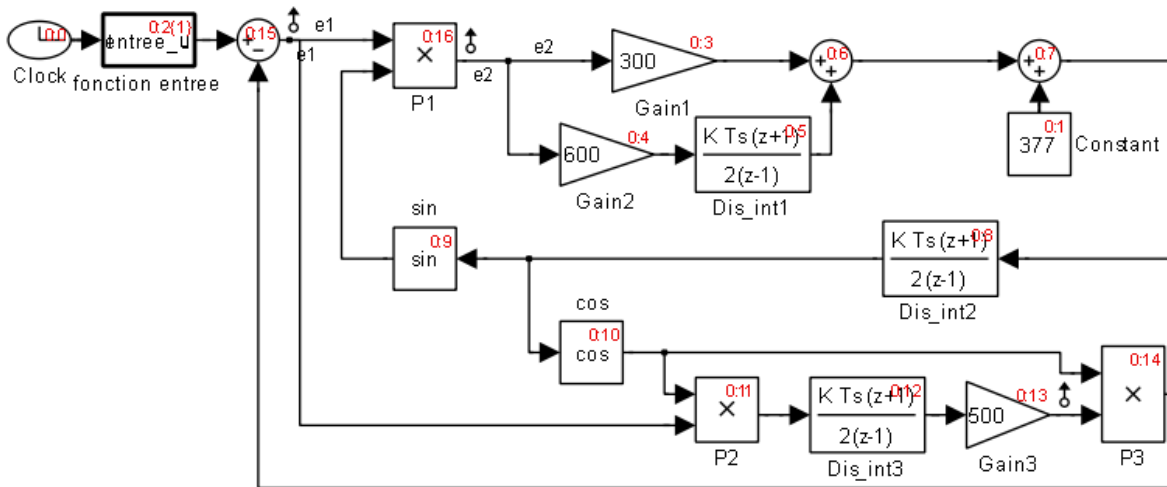


Figure 5-8: Représentation du PLL dans MATLAB/Simulink par variables d'état

La linéarisation au point k autour de 125ms ($k = \frac{t}{\Delta t} = \frac{125\text{ms}}{0.005\text{ms}} = 25000$) donne les

matrices $\mathbf{a}(k)$, $\mathbf{b}(k)$, $\mathbf{c}(k)$ et $\mathbf{d}(k)$ dans la Figure 5-9. Notez l'ordre d'exécution, en format 0:xx sur chaque bloc dans la Figure 5-8, obtenu par MATLAB/Simulink. Il s'apparente à la séquence similaire obtenue par EMTP, où les blocs PR (0:15 et 0:16) sont résolus en dernier d'une manière simultanée car on a une boucle algébrique dans MATLAB/Simulink ou un système simultané d'équations (à deux variables) dans la solution DIPR.

```

>> Model_sys

a =
      Dis_int1      Dis_int2      Dis_int3
Dis_int1         1      -0.1667      -0.4243
Dis_int2  4.792e-006      0.9167      -0.2122
Dis_int3 -2.122e-009 -0.0008487      0.9978

b =
      Sine Wave (p
Dis_int1      0.0008929
Dis_int2      0.0004465
Dis_int3      4.549e-006

c =
      Dis_int1      Dis_int2      Dis_int3
Gain3 (pt. 1) -5.304e-007      -0.2122      499.5
P1 (pt. 1)    -0.0001389      -55.56      -141.4
P3 (pt. 1)    0.0004465      178.6      454.9
PLL_simulink -0.0004465      -178.6      -454.9

d =
      Sine Wave (p
Gain3 (pt. 1)      0.001137
P1 (pt. 1)         0.2976
P3 (pt. 1)         0.04273
PLL_simulink       0.9573

Sampling time: 5e-006
Discrete-time model.

```

Figure 5-9: Représentation par variable d'état pour le PLL linéarisé autour de 125ms pour un pas de temps de 5 μ s.

Comme pour le cas simple, MATLAB/Simulink calcule la contrainte algébrique puis met à jour les variables d'état à chaque pas d'itération.

En mode "debug", on peut en effet voir différentes informations inhérentes à la solution. On a par exemple la boucle algébrique comportant les blocs listés dans la Figure 5-10.

```
(sldebug @47): >> ashow gcb
```

Blocks in Algebraic loop 0#1

- PLL_simulink_disc_etude_execution/Gain1
- PLL_simulink_disc_etude_execution/Gain2
- PLL_simulink_disc_etude_execution/Dis_int1
- PLL_simulink_disc_etude_execution/Sum2
- PLL_simulink_disc_etude_execution/Sum1
- PLL_simulink_disc_etude_execution/Dis_int2
- PLL_simulink_disc_etude_execution/sin
- PLL_simulink_disc_etude_execution/cos
- PLL_simulink_disc_etude_execution/P2 (**algebraic variable**)
- PLL_simulink_disc_etude_execution/Dis_int3
- PLL_simulink_disc_etude_execution/Gain3
- PLL_simulink_disc_etude_execution/P3
- PLL_simulink_disc_etude_execution/Sum (**algebraic variable**)
- PLL_simulink_disc_etude_execution/P1 (**algebraic variable**)

Figure 5-10: Boucle algébrique détectée par MATLAB/Simulink pour le PLL et l'ordre d'exécution des fonctions des blocs

Les étapes de la solution MATLAB/Simulink de la boucle algébrique sont montrées dans la Figure 5-11. Étant donné que les blocs connus n'ont pas à être évalués, on commence par le bloc Gain1 et on suit l'ordre en la Figure 5-10. On achève en calculant les variables algébriques associés à la boucle. Notez la Jacobienne ($dFdx$) dans la Figure 5-11 associée aux variables algébriques.

```
[TM=0.125495      ] Solving algebraic loop starting with block 0:3 'PLL_simulink_disc_etude_execution/Gain1'
```

```
z[0] =  -208.6905795351418  g[0] =  0.0001314904265257155
```

```
z[1] =  -208.6907985365179  g[1] =   9.664939110042525
```

```
z[2] =   0.3023360649411386  g[2] =   0.4934523966307993
```

```
...
```

```
z[0] =  -208.6905795351418  g[0] =  0.0001314904271794148
```

```
z[1] =  -208.6907985365179  g[1] =   9.664939110040024
```

```
z[2] =   0.3023360604359802  g[2] =   0.4934523914198106
```

```
dFdx = [
```

```
    1                -0.9999995795787929   -0.0001451001844938496
```

```
    0.00125000494680903  0.9999999999999999   0.0005551659232808158
```

```
    0                -0.0009157870461561067   1.156507010750265 ]
```

```
z[0] =  -218.3434019026775  g[0] = -9.459250236432126e-006
```

```
z[1] =  -218.3434304521337  g[1] =  4.298266745195178e-005
```

```
z[2] =  -0.1319822290522365  g[2] = -0.003149386817630828
```

```
...
```

```
z[2] =  -0.129259081682684  g[2] =  1.972502631086837e-005
```

```
z[0] =  -218.3434360908533  g[0] =  2.842170943040401e-012
```

```
z[1] =  -218.3434743615067  g[1] = -1.080024958355352e-011
```

```
z[2] =  -0.1292760307727655  g[2] =  6.000502872360869e-011
```

```
z[0] =  -218.3434360908453  g[0] =                0
```

```
z[1] =  -218.3434743614959  g[1] =                0
```

```
z[2] =  -0.1292760308243196  g[2] = -8.640310689145281e-013
```

```
Successfully solved loop in 5 iterations.
```

```
z[0] =  -218.3434360908453  g[0] =                0
```

```
z[1] =  -218.3434743614959  g[1] =                0
```

```
z[2] =  -0.1292760308243196  g[2] = -8.640310689145281e-013
```

```
%-----%
```

```
[TM = 0.1255      ] PLL_simulink_disc_etude_execution.Outputs.Major
```

Figure 5-11: Processus de la solution de la boucle algébrique dans MATLAB/Simulink pour le PLL autour de 0.125ms

À l'issue de la solution de la boucle algébrique, les variables d'état et autres variables sont mises à jour.

```
(sldebug @47): >> states
```

Discrete States:

Idx	Value	(system:block:element Name 'BlockName')
0	2.4384043170523286	(0:5:0 DSTATE 'PLL_simulink_disc_etude_execution/Dis_int1')
1	48.694863776760357	(0:8:0 DSTATE 'PLL_simulink_disc_etude_execution/Dis_int2')
2	-1.5997586728366362	(0:12:0 DSTATE 'PLL_simulink_disc_etude_execution/Dis_int3')

Figure 5-12: Valeurs des états mises à jour : processus de la solution de la boucle algébrique dans MATLAB/Simulink pour le PLL autour de 0.125ms

La solution du PLL trouvée via les variables d'état est identique à celle formulée par DIPR (voir la Figure 5-23), cependant le nombre d'équations diffère. Une équation de plus impliquant le bloc P3 apparaît dans le cas de MATLAB/Simulink. Pour analyser ceci, modifions le PLL dans MATLAB/Simulink pour isoler les fonctions des chemins de boucles. Regroupons ainsi les fonctions autour de la boucle algébrique comme les fonctions g et G de la représentation DIPR.

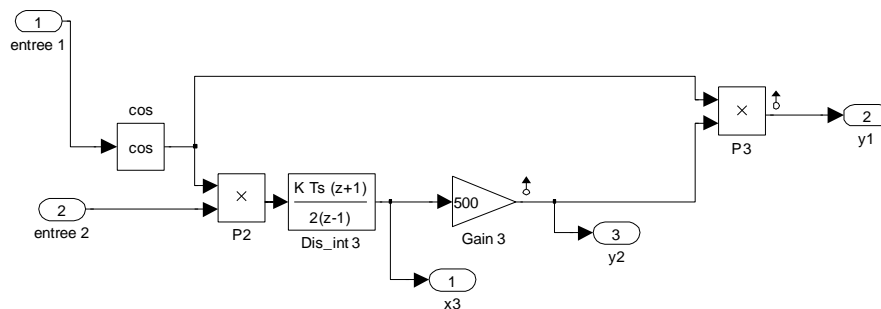


Figure 5-13: Regroupement des blocs pour la fonction composée G_2 du PLL dans MATLAB/Simulink

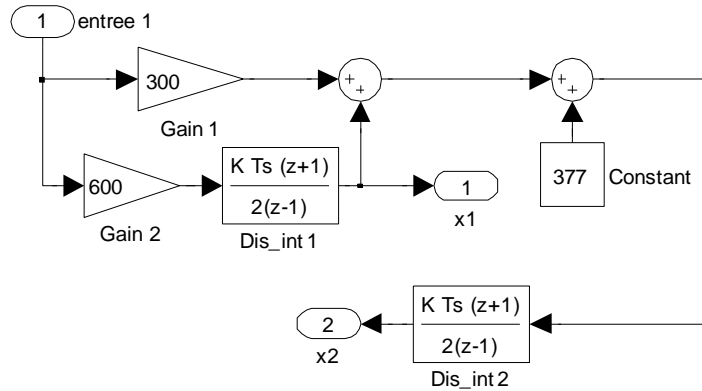


Figure 5-14: Regroupement des blocs pour la fonction composée g_1 - du PLL dans MATLAB/Simulink

Les regroupements des fonctions dans la Figure 5-13 et dans la Figure 5-14 sont codés pour tenir compte de la composition des fonctions. Le code est dans la Figure 5-15 et dans la Figure 5-16, respectivement.

```
function [x3, o1, o2] = G2(entree1,entree2,histoires, Ts)
% This block supports the Embedded MATLAB subset.
% See the help menu for details.
h_entree1 = histoires(1);
h_entree2 = histoires(2);
h_x3 = histoires(3);
%x1 = h_x1 +(Ts/2)*600* h_entree1 +(Ts/2)*600* entree1;
v = cos(entree1); vh = cos(h_entree1);
w = v*entree2; wh = vh*h_entree2;
x3 = h_x3 + (Ts/2)*( wh + w);
o2 =500*x3;
o1 = v*o2;
```

Figure 5-15: Fonction exprimant le regroupement des blocs pour la fonction composée G^2 du PLL dans MATLAB/Simulink

```
function [x1, x2] = g_1(entree1,histoires, Ts)
% This block supports the Embedded MATLAB subset.
% See the help menu for details.
h_entree1 = histoires(1);
h_x1 = histoires(2);
h_x2= histoires(3);
x1 = h_x1 + (Ts/2)*600* h_entree1 + (Ts/2)*600* entree1;
x2 = h_x2 + (Ts/2)*( 300* h_entree1 + h_x1 + 377 + 300*entree1 + x1 + 377);
```

Found algebraic loop containing:

'PLL_simulink_disc_matlab_func_DIPR_jump_input2_funcG/fonction composee g2-'

'PLL_simulink_disc_matlab_func_DIPR_jump_input2_funcG/G1'

'PLL_simulink_disc_matlab_func_DIPR_jump_input2_funcG/fonction composée G2'

'PLL_simulink_disc_matlab_func_DIPR_jump_input2_funcG/Sum' (algebraic variable)

'PLL_simulink_disc_matlab_func_DIPR_jump_input2_funcG/P1' (algebraic variable)

This algebraic loop may be resolved when the subsystem option 'Minimize algebraic loop occurrences' is selected on some or all of the atomic and enabled subsystems in this algebraic loop.

%-----%

[TM = 0] PLL_simulink_disc_matlab_func_DIPR_jump_input2_funcG.Simulate

(sldebug @0): >>

Figure 5-18: Schéma Simulink du PLL modifié avec des fonctions exprimant le regroupement des blocs pour la fonction composée g1- et G2 du PLL dans MATLAB/Simulink

Encore une fois, la linéarisation du modèle autour de 125ms donne les matrices dans la Figure 5-19.

Model_sys						
a =						
	histoires_e2	histoires_e2	histoires_e2	histoires_x2	histoires_x2	histoires_x2
histoires_e2	-0.375	-0.0025	-500	6.94e-009	-6.678e-010	0.3501
histoires_e2	0.0009376	1	-0.7499	1.041e-011	-1.002e-012	0.0005252
histoires_e2	0.0004688	3.125e-006	0.625	5.205e-012	-5.009e-013	0.0002626
histoires_x2	0.0004688	3.125e-006	0.625	5.205e-012	-5.009e-013	0.0002626
histoires_x2	0.375	0.0025	500	-6.94e-009	6.678e-010	-0.3501
histoires_x2	1.062e-009	7.083e-012	1.417e-006	1.982e-008	-1.906e-009	1
b =						
	fonction ent					
histoires_e2	-0.625					
histoires_e2	-0.0009376					
histoires_e2	-0.0004688					
histoires_x2	-0.0004688					
histoires_x2	0.625					
histoires_x2	1.746e-009					
c =						
	histoires_e2	histoires_e2	histoires_e2	histoires_x2	histoires_x2	histoires_x2
e2	-0.375	-0.0025	-500	6.94e-009	-6.678e-010	0.3501
e1	0.375	0.0025	500	-6.94e-009	6.678e-010	-0.3501
fonction com	0.0009376	1	-0.7499	1.041e-011	-1.002e-012	0.0005252
fonction com	0.0004688	3.125e-006	0.625	5.205e-012	-5.009e-013	0.0002626
fonction com	1.062e-009	7.083e-012	1.417e-006	1.982e-008	-1.906e-009	1
fonction com	-0.375	-0.0025	-500	6.94e-009	-6.678e-010	0.3501
fonction com	5.312e-007	3.541e-009	0.0007083	9.911e-006	-9.531e-007	500
d =						
	fonction ent					
e2	-0.625					
e1	0.625					
fonction com	-0.0009376					
fonction com	-0.0004688					
fonction com	1.746e-009					
fonction com	0.375					
fonction com	8.731e-007					
Sampling time: 5e-006						
Discrete-time model.						

Figure 5-19: Linéarisation du schéma MATLAB/Simulink du PLL modifié avec des fonctions exprimant le regroupement des blocs pour la fonction composée g_1 - et G_2 du PLL

La liste d'ordre d'exécution des méthodes est montrée dans la Figure 5-20. Notez-y, en bas, que les états sont maintenant associés aux délais nécessitant les termes d'histoire (voir aussi dans la Figure 5-19 en haut).

```
(sdebug @62): >> elist gcs

RootSystem.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution':
0:0 Clock.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Clock'
0:1 AtomicStateflowSubSystem.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction entree'
0:2 UnitDelay.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/histoires_x2_e1_x3/dly3'
0:3 Constant.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Ts1'
0:4 UnitDelay.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/histoires_e2_x1_x2/dly3'
0:5 Constant.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Ts'
0:6 AtomicStateflowSubSystem.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composee g2-'
0:8 AtomicStateflowSubSystem.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composée G2'
0:9 Sum.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Sum' [ algVar ]
0:10 Product.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/P1' [ algVar ]
0:11 Scope.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Etats'
0:12 Scope.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/e1_e2'
0:13 Scope.Start 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/o1_o2_input'

RootSystem.Initialize 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution':
0:1 AtomicStateflowSubSystem.Initialize 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction entree'
0:2 UnitDelay.Initialize 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/histoires_x2_e1_x3/dly3'
0:4 UnitDelay.Initialize 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/histoires_e2_x1_x2/dly3'
0:6 AtomicStateflowSubSystem.Initialize 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composee g2-'
0:8 AtomicStateflowSubSystem.Initialize 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composée G2'

RootSystem.Enable 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution':
0:1 AtomicStateflowSubSystem.Enable 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction entree'
0:6 AtomicStateflowSubSystem.Enable 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composee g2-'
0:8 AtomicStateflowSubSystem.Enable 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composée G2'

RootSystem.Disable 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution':
0:1 AtomicStateflowSubSystem.Disable 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction entree'
0:6 AtomicStateflowSubSystem.Disable 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composee g2-'
0:8 AtomicStateflowSubSystem.Disable 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composée G2'

RootSystem.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution' [tid=0,1] :
0:0 Clock.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Clock' [tid=0]
0:1 AtomicStateflowSubSystem.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction entree' [tid=0]
0:2 UnitDelay.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/histoires_x2_e1_x3/dly3' [tid=1]
0:3 Constant.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Ts1' [tid=1]
0:4 UnitDelay.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/histoires_e2_x1_x2/dly3' [tid=1]
0:5 Constant.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Ts' [tid=1]
0:6 AtomicStateflowSubSystem.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composee g2-' [tid=0]
0:7 Trigonometry.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/G1' [tid=0]
0:8 AtomicStateflowSubSystem.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/fonction composée G2' [tid=0]
0:9 Sum.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Sum' [ algVar ] [tid=0]
0:10 Product.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/P1' [ algVar ] [tid=0]
0:11 Scope.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/Etats' [tid=1]
0:12 Scope.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/e1_e2' [tid=1]
0:13 Scope.Outputs 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/o1_o2_input' [tid=1]

RootSystem.Update 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution' [tid=0,1] :
0:2 UnitDelay.Update 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/histoires_x2_e1_x3/dly3' [tid=1]
0:4 UnitDelay.Update 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/histoires_e2_x1_x2/dly3' [tid=1]
```

Figure 5-20: Processus de solution de type DIPR appliqué au schéma Simulink avec regroupement des blocs pour la fonction composée g₁. et G₂ du PLL dans MATLAB/Simulink

```

                                dFdx = [
                                1 -0.6013209883995235
                                0.999999375976844 1]
z[0] = 0.007928511543268282 g[0] = 1.768498038984268e-010
z[1] = -0.00792850656805659 g[1] = -4.059252933785729e-016
z[0] = 0.007928511543268282 g[0] = 1.768498038984268e-010
z[1] = -0.00792850656805659 g[1] = -4.059252933785729e-016
z[0] = 0.007928511432729777 g[0] = -1.890368070411874e-012
z[1] = -0.007928506457517748 g[1] = -1.734723475976807e-018
z[0] = 0.007928511433898841 g[0] = 4.962150482179517e-012
z[1] = -0.007928506458686811 g[1] = 0
        Successfully solved loop in 3 iterations.
z[0] = 0.007928511432729777 g[0] = -1.890368070411874e-012
z[1] = -0.007928506457517748 g[1] = -1.734723475976807e-018
z[0] = 0.007928511432729777 g[0] = -1.890368070411874e-012
z[1] = -0.007928506457517748 g[1] = -1.734723475976807e-018
%-----%
[TM = 0.125005 ] PLL_simulink_disc_matlab_func_DGiPR_etude_execution.Outputs.Major

```

Figure 5-21: Jacobienne et processus de la solution similaire à la DIPR appliquée à MATLAB/Simulink pour la fonction composée g_1 - et G_2 du PLL dans MATLAB/Simulink

```

Discrete States:
Idx Value (system:block:element Name 'BlockName')
0 48.695806406531062 (0:2:0 DSTATE 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/
histoires_x2_e1_x3/dly3')
0 0.0079285114327297766 (0:2:1)
0 -1.5997585014320093 (0:2:2)
1 -0.0079285064575177477 (0:4:0 DSTATE 'PLL_simulink_disc_matlab_func_DGiPR_etude_execution/
histoires_e2_x1_x2/dly3')
1 2.4354405433437387 (0:4:1)
1 48.695806406531062 (0:4:2)

```

Figure 5-22: Solution avec les variables d'état appliquée à MATLAB/Simulink similaire à la DIPR

Dans la Figure 5-23, les deux solutions DIPR et celle par MATLAB/Simulink sont superposées. Cependant, bien que les solutions trouvées par les deux approches soient identiques, les particularités de chacune peuvent induire des défis différents dans le processus de solution de type Newton. On note, par ailleurs, que trois équations algébriques ont été formées avant regroupement des blocs de la Figure 5-17, soit une équation de plus pour la solution MATLAB/Simulink. Rappelons d'ailleurs le fait que, pour la DIPR aussi, on pourrait être appelé à former un système légèrement plus grand en optant pour le oEC au lieu du mEC, soit un

ensemble des coupes optimal au lieu de minimal. Dans ce cas de PLL, la DIPR aurait un avantage par rapport à la solution Simulink car elle a moins d'équations.

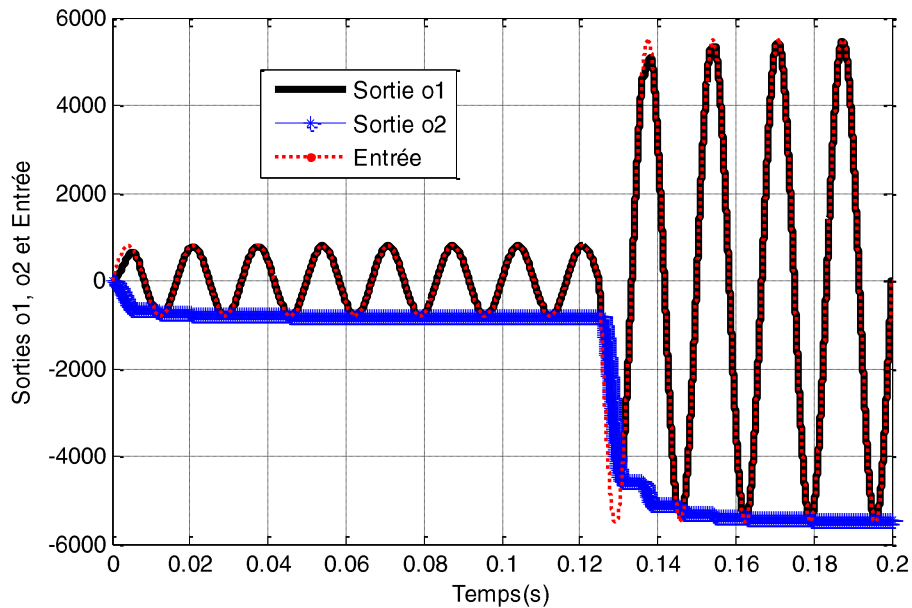


Figure 5-23: Les signaux des sorties o1 et o2 et d'entrée du PLL monté dans MATLAB/Simulink (identiques pour les solutions DIPR montées dans MATLAB/Simulink)

Par ailleurs, élargissant cette discussion, mentionnons que l'initialisation dans les différentes approches induit des défis différents. On traite justement de cet aspect dans le dernier chapitre.

5.3 Conclusion

Ce chapitre a consisté à comparer la méthode DIPR à d'autres méthodes de solution de système de contrôle. Deux comparaisons ont été faites.

Une première a montré que, sur le pur plan des représentations des systèmes d'équations simultanées, la méthode de substitution de variables peut transformer une représentation complète classique (telle qu'introduite dans [4]) en une représentation DIPR, à condition de ne pas substituer les variables PR et de suivre un ordre de substitution respectant la séquence topologique d'évaluation des blocs, telle que fournie par une méthode de type DFS. Il a été rappelé, d'une part, que cette comparaison concerne seulement l'équivalence de représentation alors que les chapitres précédents traitaient, eux, de la précision et de la performance de deux

approches. D'autre part, et à plus forte raison, l'équivalence de représentation ne garantit pas la même performance quand une méthode est appliquée à deux approches équivalentes de représentation du même système.

Une deuxième comparaison est celle entre l'approche DIPR et la représentation par variables d'état. Le logiciel MATLAB/Simulink, lequel utilise une représentation par variables d'état, a été comparé à l'approche DIPR dans le cas où les variables d'état équivalent aux sorties des blocs PR et dans le cas où existent des variables d'état qui ne soient pas des variables aux PR. Les analyses ont été illustrées sur le PLL abordé dans les chapitres précédents. Il ressort de l'analyse que l'étape d'évaluation des sorties de blocs dans l'approche avec représentation par variables d'état s'apparente à la solution DIPR pour la solution de la boucle algébrique, les contraintes algébriques étant comparables aux boucles de rétroaction. Une différence peut cependant surgir quand le nombre de ces contraintes (et donc la taille de la Jacobienne) dans le cas par variables d'état n'est pas le nombre optimal des coupures (c'est-à-dire, le nombre de PR). Pour le PLL par exemple, MATLAB/Simulink a généré 3 contraintes alors que la DIPR en a généré 2. La méthode DIPR pourrait avoir certains avantages. La solution finale cependant est identique car, entre autres, on a vu que le nombre de coupures peut être supérieur au nombre minimal mais dans les deux cas, il faut, virtuellement, qu'aucun cycle ne subsiste dans le graphe représentant le système de contrôle.

Les deux angles de cette analyse permettent d'approfondir la compréhension de l'approche DIPR et peuvent susciter des pistes d'améliorations futures.

CHAPITRE 6 DÉFIS D'INITIALISATION ET TRAVAUX FUTURS

6.1 Motivation

Dans les applications du chapitre quatre, la méthode d'initialisation de [4] a été utilisée au temps $t=0$ avec la nouvelle approche DIPR. Pour rappel, l'approche de [4] est une méthode de Newton (de base) appliquée sur une représentation complète du système d'équations, c'est-à-dire, pour lequel une équation est formée pour chaque bloc de contrôle. Par ailleurs, on a montré une légère altération des critères de convergence, quand on « propage », par évaluation séquentielle, le résultat final des PR sur les autres blocs de la boucle, dans l'approche DIPR (voir la discussion à la section 4.4.1). Clairement, des défis d'initialisation sont à relever.

Ce chapitre veut approfondir l'analyse de ces défis. Précisément, dans cette section, nous nous proposons d'analyser ce qui suit :

- i) Par hypothèse, on suppose que le système de contrôle représenté sous la forme complète est résolu avec succès par les itérations de la méthode de Newton ; Plus précisément, nous entendons par là que sa Jacobienne est non-singulière et nous avons fait le choix des valeurs initiales qui assurent l'existence et l'unicité de la solution dans un voisinage autour des valeurs initiales ;
- ii) Partant des hypothèses ci-haut, comment la transformation GIPR modifie-t-elle les propriétés des itérations de Newton ?

En effet, on sait que certaines caractéristiques du système modélisé donnent une mesure de la performance des itérations de Newton. Il s'agit : de la fonction objective Φ , de la Jacobienne \mathbf{J} et de son inverse \mathbf{J}^{-1} , du vecteur des valeurs initiales $\mathbf{e}^{k=0}$, des produits des inverses de la Jacobienne et de la fonction objective $\mathbf{J}^{-1}\Phi$ et des dérivées des Jacobiennes \mathbf{J}' (matrices Hessiennes). En d'autres termes, l'analyse veut identifier comment changent ces différents éléments par la transformation DIPR pour affecter les conditions d'existence de la solution. Plus encore, ces éléments ont un lien avec les propriétés entrée-sortie impliquées dans l'estimation de la stabilité du système. Cependant, ce lien ne sera pas traité dans le cadre de ce travail. Les caractéristiques Φ , \mathbf{J} , \mathbf{J}^{-1} , $\mathbf{e}^{k=0}$, $\mathbf{J}^{-1}\Phi$ et \mathbf{J}' sont les paramètres des différents

théorèmes sur l'existence et l'unicité de la solution associés à i) et ii). Nous appliquerons précisément le théorème de Newton-Kantorovich et leurs variantes pour cet aspect.

En d'autres termes, nous devons établir une expression générique de la transformation des matrices et vecteurs Φ , \mathbf{J} , \mathbf{J}^{-1} , $\mathbf{e}^{k=0}$, $\mathbf{J}^{-1}\Phi$ et \mathbf{J}' . Nous recourons encore fois aux outils d'analyse du graphe des équations. Au final, le but est d'établir comment changent les normes de ces matrices et vecteurs durant la transformation.

Notons que cette analyse offre des pistes pour une détermination formelle des valeurs initiales, aussi bien pour la méthode à représentation complète que pour la solution avec la méthode réduite DIPR.

Avant de débiter rappelons donc les notions de norme d'un vecteur, norme induite d'une matrice et de mesures d'une matrice :

a. Normes l_1, l_2, l_∞ d'un vecteur $\mathbf{e} = [e_1, e_2, \dots, e_j, \dots, e_M]$, avec la terminologie de [64] :

- Norme $-l_1$ ou norme des moindres résidus absolus :

$$\|\mathbf{e}\|_1 = \sum_{j=1}^M |e_j| \quad (6.1)$$

- Norme l_∞ ou norme de supremum :

$$\|\mathbf{e}\|_\infty = \max_{j=1 \dots M} |e_j| \quad (6.2)$$

- Norme l_2 ou norme euclidienne ou de moindres-carrés :

$$\|\mathbf{e}\|_2 = \sqrt{\sum_{j=1}^M |e_j|^2} \quad (6.3)$$

On note aussi les inégalités suivantes entre les bornes des normes :

$$\|\mathbf{e}\|_1 \geq \|\mathbf{e}\|_2 \geq \|\mathbf{e}\|_\infty,$$

$$\|\mathbf{e}\|_1 \leq \sqrt{M} \|\mathbf{e}\|_2,$$

$$\|\mathbf{e}\|_2 \leq \sqrt{M} \|\mathbf{e}\|_\infty.$$

Ainsi donc, l'utilisation d'une ou d'une autre norme ne change pas le résultat de l'analyse, le choix de la norme tenant compte uniquement de la facilité à établir les bornes supérieures et inférieures.

b. Normes d'une matrice \mathbf{J} : la norme matricielle découle de celle d'un vecteur et lui est induite. D'où :

○ Norme l_1 de \mathbf{J} :

$$\|\mathbf{J}\|_1 = \max_{1 \leq j \leq M} (\|\mathbf{J}_j\|_1) = \max_{1 \leq j \leq M} \sum_{i=1}^M |\mathbf{J}_{i,j}|, \quad (6.4)$$

où l'évaluation est faite sur tous les vecteurs lignes de \mathbf{J} selon (6.1)

○ Norme l_∞ de \mathbf{J} :

$$\|\mathbf{J}\|_\infty = \max_{1 \leq i \leq M} (\|\mathbf{J}_i\|_1) = \max_{1 \leq i \leq M} \sum_{j=1}^M |\mathbf{J}_{i,j}|, \quad (6.5)$$

où l'évaluation est faite sur tous les vecteurs colonnes de \mathbf{J} selon (6.1).

○ Norme l_2 de \mathbf{J} (ou norme spectrale car \mathbf{J} est une matrice carrée) :

$$\|\mathbf{J}\|_2 = \sqrt{\lambda \max(\mathbf{J}^T \mathbf{J})}, \quad (6.6)$$

où $\lambda \max(\cdot)$ est le maximum des valeurs propres de (\cdot) et \mathbf{J}^T est la transposée de \mathbf{J} .

c. Mesure d'une matrice :

La mesure μ de la matrice Jacobienne \mathbf{J} des fonctions Φ est tel que [16]:

$$\mu(\mathbf{J}) = \lim_{\theta \rightarrow 0} \frac{\|\mathbf{I} + \theta \mathbf{J}\| - 1}{\theta}, \quad (6.7)$$

où la norme induite $\|\cdot\|$ dans (6.7) dépend de la norme utilisée pour l'analyse et θ est une constante. Notez que μ peut être négative, contrairement aux normes.

Les différentes normes introduites ci-haut sont appliquées dans la section suivante. Nous établissons les formes génériques de \mathbf{J} , \mathbf{J}^{-1} pour les deux types de formulation complète et réduite par la DIPR.

6.2 Expressions de \mathbf{J} et Φ dans les représentations complète et réduite par DIPR

6.2.1 Cas d'un seul point de rétroaction

Pour des besoins de clarté, on choisit d'abord le cas simple d'une seule coupure au PR dans le système de contrôle comme dans la Figure 6-1 :

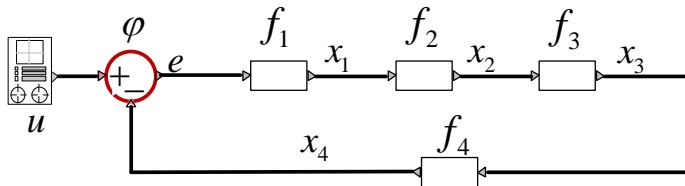


Figure 6-1: Système de contrôle simple pour analyse de l'existence et d'unicité de la solution pour système complet et réduit

Le système d'équations de ce système dans la représentation complète est :

$$\begin{aligned}
 x_1 &= f_1(e) \\
 x_2 &= f_2(x_1) \\
 x_3 &= f_3(x_2) \\
 x_4 &= f_4(x_3) \\
 e &= u - x_4
 \end{aligned} \tag{6.8}$$

Le processus de Newton pour le résoudre est la séquence d'itérations k successives que nous avons introduites plus haut, de la forme suivante :

$$\mathbf{x}^k = \mathbf{x}^{k-1} - \mathbf{J}^{-1} \Phi|^{k-1}, \tag{6.9}$$

où $\mathbf{x} = [x_1, x_2, x_3, x_4, e]^T$ et le vecteur des fonctions objectives Φ et la Jacobienne \mathbf{J} sont respectivement :

$$\Phi \equiv \begin{cases} x_1 - f_1(e) = 0 \\ x_2 - f_2(x_1) = 0 \\ x_3 - f_3(x_2) = 0 \text{ et} \\ x_4 - f_4(x_3) = 0 \\ e - u + x_4 = 0 \end{cases} \quad \mathbf{J} = \begin{pmatrix} 1 & 0 & 0 & 0 & -\frac{\partial}{\partial e} f_1 \\ -\frac{\partial}{\partial x_1} f_2 & 1 & 0 & 0 & 0 \\ 0 & -\frac{\partial}{\partial x_2} f_3 & 1 & 0 & 0 \\ 0 & 0 & -\frac{\partial}{\partial x_3} f_4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \quad (6.10)$$

Avec N blocs dans ce cas d'unique chemin de rétroaction, on aura une forme quasi bi-diagonale inférieure pour \mathbf{J} :

$$\Phi \equiv \begin{cases} x_1 - f_1(e) = 0 \\ x_2 - f_2(x_1) = 0 \\ \vdots \\ x_j - f_j(x_{j-1}) = 0 \\ \vdots \\ x_{N-1} - f_{N-1}(x_{N-2}) = 0 \\ e - u \pm x_{N-1} = 0 \end{cases} \quad \text{et} \quad \mathbf{J} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & * \\ * & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & * & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & * & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & * & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & * & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}, \quad (6.11)$$

où les $*$ représentent les $N-1$ dérivées des fonctions des blocs dans l'ordre topologique du parcours du chemin de rétroaction.

L'inverse de la matrice Jacobienne sera de la forme suivante pour 5 blocs :

$$\Delta_{\mathbf{J}} \mathbf{J}^{-1} = \begin{pmatrix} 1 & -\frac{\partial}{\partial} f_3 \frac{\partial}{\partial} f_4 \frac{\partial}{\partial} f_1 & -\frac{\partial}{\partial} f_4 \frac{\partial}{\partial} f_1 & -\frac{\partial}{\partial} f_1 & \frac{\partial}{\partial} f_1 \\ \frac{\partial}{\partial} f_2 & 1 & -\frac{\partial}{\partial} f_2 \frac{\partial}{\partial} f_1 \frac{\partial}{\partial} f_4 & -\frac{\partial}{\partial} f_2 \frac{\partial}{\partial} f_1 & \frac{\partial}{\partial} f_2 \frac{\partial}{\partial} f_1 \\ \frac{\partial}{\partial} f_3 \frac{\partial}{\partial} f_2 & \frac{\partial}{\partial} f_3 & 1 & -\frac{\partial}{\partial} f_3 \frac{\partial}{\partial} f_2 \frac{\partial}{\partial} f_1 & \frac{\partial}{\partial} f_3 \frac{\partial}{\partial} f_2 \frac{\partial}{\partial} f_1 \\ \frac{\partial}{\partial} f_4 \frac{\partial}{\partial} f_3 \frac{\partial}{\partial} f_2 & \frac{\partial}{\partial} f_4 \frac{\partial}{\partial} f_3 & \frac{\partial}{\partial} f_4 & 1 & \frac{\partial}{\partial} f_4 \frac{\partial}{\partial} f_3 \frac{\partial}{\partial} f_2 \frac{\partial}{\partial} f_1 \\ -\frac{\partial}{\partial} f_2 \frac{\partial}{\partial} f_3 \frac{\partial}{\partial} f_4 & -\frac{\partial}{\partial} f_3 \frac{\partial}{\partial} f_4 & -\frac{\partial}{\partial} f_4 & -1 & 1 \end{pmatrix}, \quad (6.12)$$

où $\Delta_{\mathbf{J}}$ est le déterminant du système d'équation en (6.11)

On sait déduire la forme générale du déterminant Δ_J par les outils de calcul symbolique. De plus, sa valeur peut être déduite de la formule de Mason [31][32] en transformant le système comme un graphe orienté dans lequel on met en évidence les fonctions des blocs, contrairement à l'analyse du Chapitre 2, en le représentant comme les arcs. Les variables (inconnues) étant les sommets du graphe. On montre, dans les lignes qui suivent, comment cette forme générale de la Jacobienne et de son inverse peut être déduite d'une manière générique.

Pour y arriver, on se basera sur les développements et théorèmes de [32], entre autres.

Établissons d'abord la correspondance entre la représentation des équations et une représentation sous forme de graphe de Coates. Le graphe de Coates d'un système d'équations linéaires de la forme (6.11) où les fonctions des blocs sont linéaires se déduit en réécrivant chaque équation (cas linéaire) dans la forme :

$$\sum_{i=1}^N C_{ji} x_i = C u_j \quad (j = 1, 2, \dots, N), \quad (6.13)$$

où C_{ji} est la fonction (un gain dans le cas linéaire) entre les nœuds i et j , $-C u_j$ est la fonction (encore une fois, un gain dans le cas linéaire) de la branche de la source u_j vers le nœud j , et les x_i sont les inconnues. Notez que pour le cas non-linéaire on considérera la version linéarisée. En clair, ce sera la Jacobienne du système pour lequel on a à tout instant :

$$\mathbf{x}^{k-1} - \mathbf{x}^k = \mathbf{J}^{-1} \mathbf{\Phi} \big|^{k-1}. \quad (6.14)$$

Dans ce cas non-linéaire, on considérera les éléments de la Jacobienne comme étant les gains de branches. Cette approche est similaire à celle de la page 122 de [31].

Notons aussi que la règle de Mason généralise le gain entre deux nœuds en définissant le « transfert » d'un bloc j vers i par « transmittance » $T_{i \rightarrow j}$ qui, par définition, est la dérivée partielle de x_i relativement au j -ième entrée. Ceci sera utile pour des cas comportant plusieurs chemins de rétroaction et des nonlinéarités. Nous représenterons d'une manière équivalente les gains des branches et les dérivées partielles d'un nœud vers un autre. Sans perte de généralité, le graphe de Coates sera de la forme suivante pour le cas simple en (6.11) :

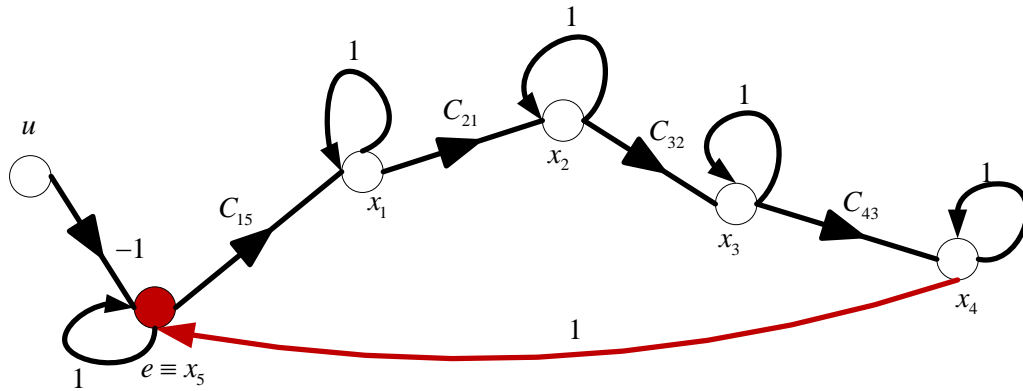


Figure 6-2: Graphe de Coates pour les équations du système de contrôle en (6.10) réécrit en (6.13)

Pour déduire d'une manière générique, les expressions plus haut, introduisons les quelques définitions suivantes : sous-graphes 1-facteur d'un graphe de Coates, matrice de connectivité et permanent d'une matrice.

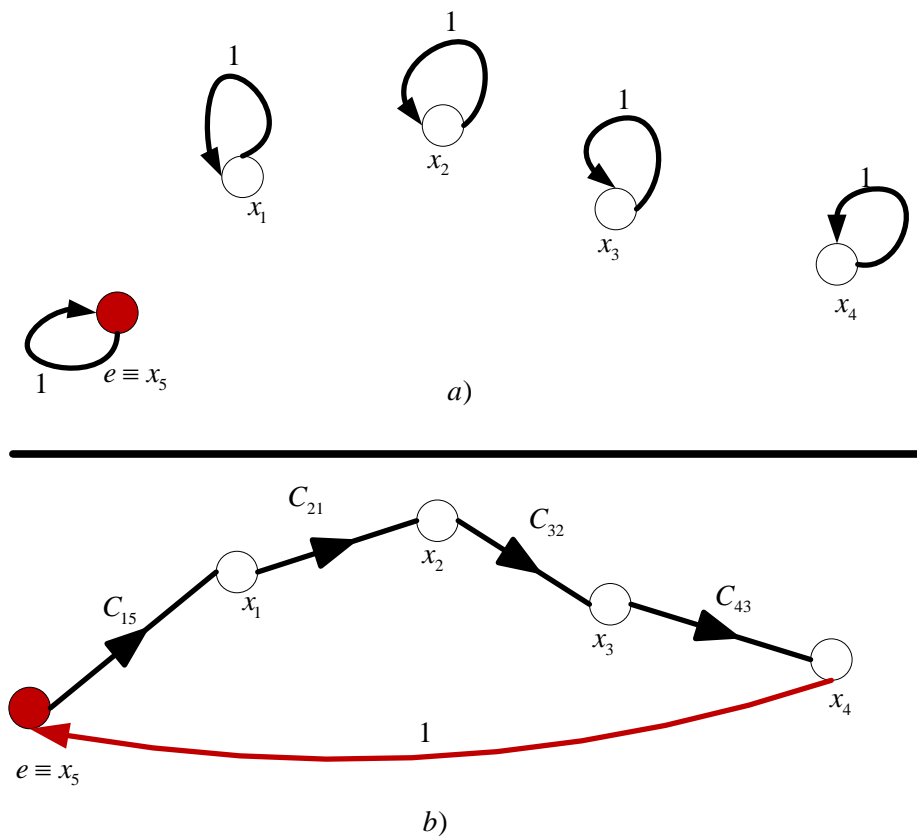


Figure 6-3: « Sous-graphes 1-facteur » du Graphe de Coates en la Figure 6-2 des équations du système de contrôle en (6.10)

On forme, de ce graphe, des sous-graphes 1-facteur, illustrés dans la Figure 6-3 comme suit [32] : i) Garder tous les nœuds en place sauf les nœuds sources ; ii) N'associer qu'une seule branche entrant dans le nœud et une seule branche sortant du nœud. On dira alors que les sous-graphes sont de degré un ; ii) Répéter ce processus tant qu'on peut former des sous-graphes 1-facteur différents. Dans le cas simple traité on a seulement deux sous-graphes 1-facteur. Le nombre total des sous-graphes 1-facteur peut d'ailleurs être déterminé comme suit [32] : i) Écrire la matrice $Con(\mathbf{gr})$ pour la connectivité du graphe de Coates ; ii) Calculer son permanent, $permanent(Con(\mathbf{gr}))$; iii) Le nombre des sous-graphes 1-facteur est égal à la valeur trouvée de $permanent(Con(\mathbf{gr}))$.

La matrice de connectivité étant définie comme l'adjacence, on met toutefois à l'élément (i, j) la valeur de nombre de branches quittant le nœud i vers le nœud j , en lieu et place de un. Le permanent, quant à lui, se calcule comme le déterminant, mais sans ajouter le signe négatif sur aucun de termes. Quand on a au maximum une branche reliant deux nœuds, ce qui est le cas des systèmes de contrôle, on a la relation suivante entre la matrice d'adjacence et la connectivité :

$$Con(\mathbf{gr}) = \mathbf{A}^T + \mathbf{I}. \quad (6.15)$$

Un algorithme, dû à [65], est présenté en annexe sous forme MATLAB pour calculer le permanent d'une matrice carrée sous forme récursive. Une commande *linalg::per* récemment introduite dans l'outil muPAD de MATLAB (voir version 2009 et plus), évalue le permanent de la matrice [40].

Encore une fois, en appliquant sur le cas simple, on a :

$$permanent(Con(\mathbf{gr})) = permanent \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} = 2 \quad (6.16)$$

On vérifie donc que selon (6.16), on ne peut construire que deux sous-graphes dans la Figure 6-3.

On peut maintenant donner l'expression générale du déterminant de la matrice.

On sait en fait que, se basant sur la règle de Mason (voir en annexe de [31], notamment), et en la page 169 de [32] et se basant aussi sur la transformation en graphe de Coates [32][66][67] et le développement indépendant dans [66], on a les relations suivantes générales pour N blocs :

$$\Delta_{\mathbf{J}} = (-1)^N \sum_{\substack{h \\ i=1}}^{\#h} (-1)^{\#Bhi} \prod_j \frac{\partial}{\partial x_{j-1}} f_j, \quad (6.17)$$

où h est l'ensemble des sous-graphes 1-facteur du graphe de Coates, $\#h$ est le nombre de sous-graphes 1-facteur ; $\#Bhi$ est le nombre des boucles dans le i -ième sous-graphe considéré ; Ni est le nombre de branches (gains, fonctions dérivées) se retrouvant dans le sous-graphe et à multiplier dans le produit PI.

En appliquant cette formule pour le cas simple du sous-graphe dans la Figure 6-3, on a ;

$$\Delta_{\mathbf{J}} = (-1)^5 \left(\underbrace{(-1)^5 (1 \times 1 \times 1 \times 1 \times 1)}_{\text{Premier sous-graphe 1-facteur}} + \underbrace{(-1)^1 \left(\left(-\frac{\partial}{\partial_3} f_4 \right) \left(-\frac{\partial}{\partial_{x_2}} f_3 \right) \left(-\frac{\partial}{\partial_{x_1}} f_2 \right) \left(-\frac{\partial}{\partial_e} f_1 \right) \right)}_{\text{Deuxième sous-graphe 1-facteur}} \right), \quad (6.18)$$

$$\Delta_{\mathbf{J}} = 1 + \frac{\partial}{\partial_3} f_4 \frac{\partial}{\partial_{x_2}} f_3 \frac{\partial}{\partial_{x_1}} f_2 \frac{\partial}{\partial_e} f_1 \times 1, \quad (6.19)$$

où, comme on a indiqué, les dérivées peuvent être les constantes pour le cas linéaires, on a en réalité :

$$\Delta_{\mathbf{J}} = 1 + C_{43} C_{32} C_{21} C_{15} \quad (6.20)$$

Pour déterminer l'expression générique de la matrice Jacobienne, nous utilisons l'analogie de la solution linéarisée en (6.14) en appliquant encore fois l'analyse du graphe de Coates de [32] qui serait associé à ces équations linéarisées et l'expression de la solution $\delta \mathbf{x} = \mathbf{x}^{k-1} - \mathbf{x}^k$. En effet, tout comme la règle de Mason, on a la solution pour chacune des variables autour du point de linéarisation :

Du théorème 3.3 de [32], page 149, on aura une forme de la solution suivante :

$$\delta x_i = \frac{\sum_{H_{(N+1)i}} (-1)^{BH} \prod H_{(N+1)i}}{\sum_{h'} (-1)^{B'h} \prod h'}, \quad (6.21)$$

où $H_{(N+1)i}$ et BH sont respectivement la *connexion du sous-graphe 1-facteur* (que nous définirons ci-bas) et le nombre des boucles dans cette dernière et ceci pour le graphe d'une matrice modifiée en ajoutant le vecteur de la fonction Φ à droite de la matrice \mathbf{J} et des zéros sur la ligne d'en bas (ainsi, on a le $(N+1)$ -ième variable) ; h' et $B'h$ sont respectivement l'ensemble sous-graphe 1-facteur et le nombre de boucles. La connexion du sous-graphe 1-facteur entre deux nœuds, ici $(N+1)$ et i , sont les sous-graphes dans lequel on ne garde que les branches connectant les nœuds $(N+1)$ et i et les nœuds disjoints à tout nœud dans le chemin $(N+1)$ et i , mais pour lesquels on peut former une boucle. Et finalement \prod_- est, encore une fois, le produit des gains (dérivées partielles des branches).

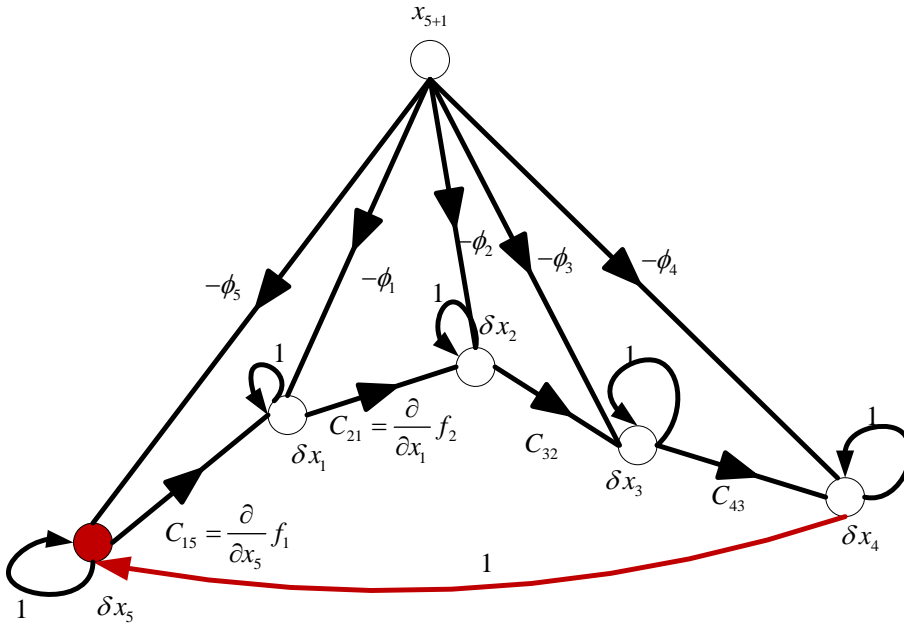


Figure 6-4: Graphe modifié pour la solution $\mathbf{J}\delta\mathbf{x} = \Phi$ pour un cas simple

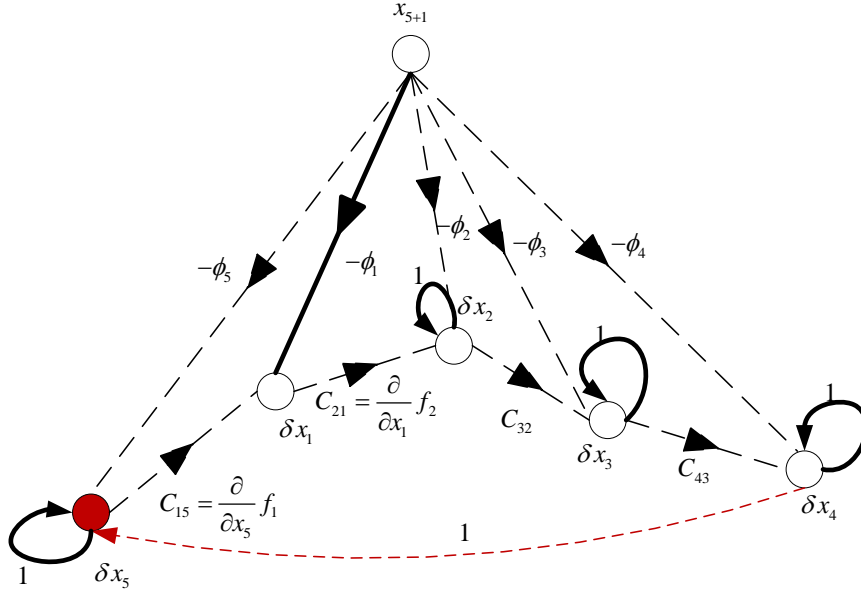


Figure 6-5: Première des connexions dans le sous-graphe 1-facteur $H_{(N+1)i}$ dans (6.21) entre le nœud $N+1$ et 1 basé sur le graphe modifié pour la solution $\mathbf{J}\delta\mathbf{x} = \Phi$ montré en Figure 6-4 (les branches en pointillé sont inexistantes; les autres connexions sont dans la Figure 6-6)

Le premier terme dans la sommation (6.21), qu'on notera ζ_1 car associé au lien du nœud source de la fonction ϕ vers le nœud 1, découlant de la Figure 6-5 est :

$$\zeta_1 = (-1)^4 \prod H_{(5+1)1} = (-1)^4 (-\phi_1) \times 1 \times 1 \times 1 \times 1, \quad (6.22)$$

où on a donc $BH = 4$ car on a quatre boucles aux nœuds $\delta x_2, \delta x_3, \delta x_4$ et δx_5 . Les gains sont quant à eux de $-\phi_1$ pour la branche $(5+1) \rightarrow 1$ et tous les autres gains à considérer sont l'unité dans les branches propres bouclées : $5 \rightarrow 5$, $2 \rightarrow 2$, $3 \rightarrow 3$ et $4 \rightarrow 4$.

Par la même procédure, on établit les quatre autres termes restants $\zeta_{2,3,4,5}$. Rappelons d'ailleurs que le nombre des termes à considérer (ici 5 termes) peut être évalué par le calcul du permanent de la matrice associée au graphe de Coates montré en Figure 6-4 dans lequel on a supprimé la colonne $(5+1)$ et 1. On vérifiera bien cette relation après avoir établi les quatre termes restants. Les connexions de sous-graphes pour chacun de ces termes est dans la Figure 6-6.

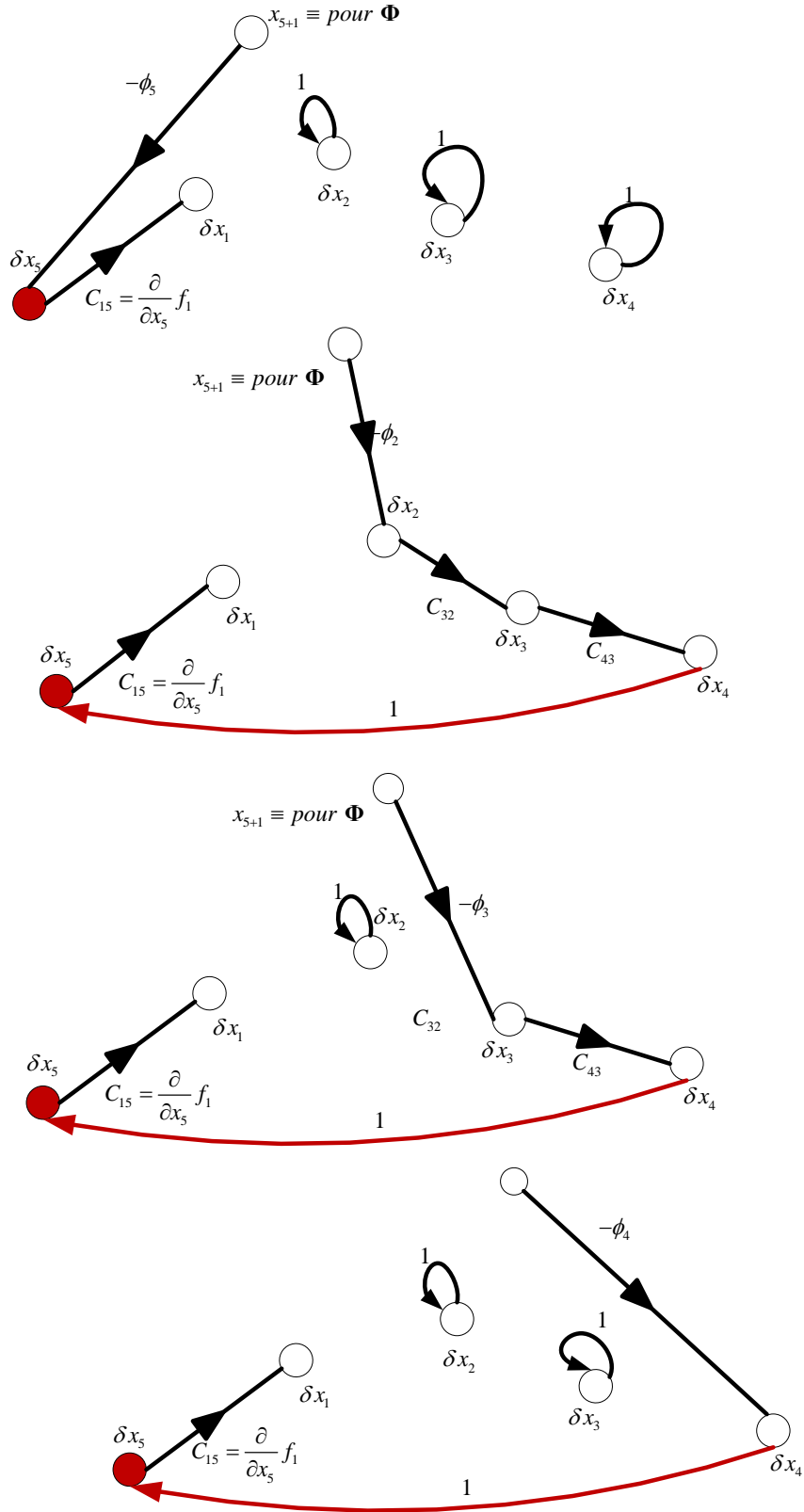


Figure 6-6: Autres connexions dans le sous-graphe 1-facteur similaires à celles de la Figure 6-5

Du bas en haut, on a :

$$\zeta_5 = (-1)^3 \prod_{H_{(5+1)1}} = (-1)^3 \underbrace{(-\phi_5) \times -\frac{\partial}{\partial x_5} f_1}_{\text{Connexion directe}} \times \underbrace{1 \times 1 \times 1}_{\text{Boucles sur branches propres de 2,3 et 4}}, \quad (6.23)$$

$$\zeta_2 = (-1)^0 \prod_{H_{(5+1)1}} = (-1)^0 \underbrace{(-\phi_2) \times -\frac{\partial}{\partial x_2} f_3 \times -\frac{\partial}{\partial x_3} f_4 \times \frac{\partial}{\partial x_4} f_5 \times \frac{\partial}{\partial x_5} f_1}_{\text{Connexion directe}}, \quad (6.24)$$

$$\zeta_3 = (-1)^1 \prod_{H_{(5+1)1}} = (-1)^1 \underbrace{(-\phi_3) \times -\frac{\partial}{\partial x_3} f_4 \times -\frac{\partial}{\partial x_4} f_5 \times -\frac{\partial}{\partial x_5} f_1}_{\text{Connexion directe}} \times \underbrace{1}_{\text{Boucle sur branche propre de 2}}, \quad (6.25)$$

$$\zeta_4 = (-1)^2 \prod_{H_{(5+1)1}} = (-1)^2 \underbrace{(-\phi_4) \times \frac{\partial}{\partial x_4} f_5 \times -\frac{\partial}{\partial x_5} f_1}_{\text{Connexion directe}} \times \underbrace{1 \times 1}_{\text{Boucles sur branche propre de 2 et 3}}. \quad (6.26)$$

Réécrivons l'équation (6.21) pour clairement faire voir l'inverse de la Jacobienne :

$$\delta x_i = \frac{\sum_{j=1}^{H_{(N+1)i}} \text{permanent}(Con(\mathbf{gr}H_{(N+1)i})) \zeta_j}{\sum_{h'} (-1)^{L'h} \prod h'}, \quad (6.27)$$

où $\text{permanent}(Con(\mathbf{gr}H_{(N+1)i}))$ est le permanent de la matrice de connectivité $Con(\mathbf{gr}H_{(N+1)i})$ du système d'équation augmenté du vecteur Φ , comme introduit en bas de (6.21) et pour lequel on a le graphe de Coates $\mathbf{gr}H_{(N+1)i}$ en la Figure 6-4.

On a :

$$Con(\mathbf{gr}H_{(N+1)i}) = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & \mathbf{5+1} \end{matrix} \\ \begin{matrix} \mathbf{1} \\ 2 \\ 3 \\ 4 \\ 5 \\ (5+1) \end{matrix} & \begin{pmatrix} \cancel{x} & \cancel{x} & \cancel{x} & \cancel{x} & \cancel{x} & \cancel{x} \\ 0 & 1 & 1 & 0 & 0 & \cancel{x} \\ 0 & 0 & 1 & 1 & 0 & \cancel{x} \\ 0 & 0 & 0 & 1 & 1 & \cancel{x} \\ 1 & 0 & 0 & 0 & 1 & \cancel{x} \\ 1 & 1 & 1 & 1 & 1 & \cancel{x} \end{pmatrix} \end{matrix} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}, \quad (6.28)$$

où le signe \times indique que l'élément est supprimé comme toute la ligne ou toute la colonne.

On vérifie encore fois, selon l'algorithme en annexe basé sur [65], que le permanent est :

$$\text{permanent}\left(\text{Con}\left(\mathbf{gr}H_{(N+1)i}\right)\right)=5. \quad (6.29)$$

Mentionnons aussi que cette expression (6.27) est similaire à celle de l'inverse de la matrice par utilisation des cofacteurs et la solution associée à chaque variable [32], page 149 :

$$\delta x_i = \frac{\sum_{j=1}^N \Phi_j(\Delta_j ij)}{\Delta_j}, \quad (6.30)$$

où $(\Delta_j ij)$ est le cofacteur de l'élément (i, j) dans Δ_j .

Terminons en réécrivant maintenant (6.27) en isolant les facteurs associés à la fonction Φ_j pour chaque variable et en se rappelant que, pour chaque inconnue, la solution est de la forme :

$$\delta x_j = \sum \mathbf{J}^{-1}(i, j) \times \phi_i.$$

On voit donc aisément que les éléments de la matrice inverse de la Jacobienne \mathbf{J}^{-1} sont de la forme :

$$\mathbf{J}^{-1}(i, j) \Delta_j = \frac{\zeta_{ji}}{-\phi_j} = \frac{(-1)^{BH} \prod H_{(s+1)l}}{-\phi_j} = (-1)^{BH} \underbrace{\prod \frac{\partial}{\partial x_-} f_-}_{\text{Connexion directe (n+1)} \rightarrow j} \times \prod 1 \quad ,$$

Boucles sur branches propres

où BH est le nombre de boucles dans le sous-graphe.

On aura donc la forme suivante pour l'inverse de la matrice Jacobienne :

$$\mathbf{J}^{-1} = \frac{1}{\Delta_{\mathbf{J}}} \begin{pmatrix} 1 & \alpha_{12} \prod_j^{P_{12}} \frac{\partial}{\partial \cdot} f_j & \cdots & \cdots & \alpha_{1N} \prod_j^{P_{1N}} \frac{\partial}{\partial \cdot} f_j \\ \alpha_{21} \prod_j^{P_{21}} \frac{\partial}{\partial \cdot} f_j & 1 & \ddots & \ddots & \alpha_{2N} \prod_j^{P_{2N}} \frac{\partial}{\partial \cdot} f_j \\ \vdots & \ddots & 1 & \vdots & \vdots \\ \alpha_{N-1,1} \prod_j^{P_{N-1,1}} \frac{\partial}{\partial \cdot} f_j & \alpha_{N-1,2} \prod_j^{P_{N-1,2}} \frac{\partial}{\partial \cdot} f_j & \ddots & 1 & \alpha_{N-1,N} \prod_j^{P_{N-1,N}} \frac{\partial}{\partial \cdot} f_j \\ \alpha_{N1} \prod_j^{P_{N1}} \frac{\partial}{\partial \cdot} f_j & \alpha_{N2} \prod_j^{P_{N2}} \frac{\partial}{\partial \cdot} f_j & \cdots & -1 & 1 \end{pmatrix}, \quad (6.31)$$

où P_{ij} est entre 1 et N . Son expression exacte découle de la formule exprimant l'inverse d'une matrice à l'aide des cofacteurs dans (6.30), qui est similaire au numérateur dans (6.21). De plus les facteurs α_{ij} dépendent de la signature des permutations ou comme montré plus haut, leurs liens avec les nombres des boucles dans les connexions de sous-graphes 1-facteur. Mentionnons en terminant que cette forme de l'inverse d'une matrice peut aussi s'établir pour le cas d'un seul PR par la matrice adjointe $\text{Adj}\mathbf{J}$:

$$\Delta_{\mathbf{J}} \mathbf{J}^{-1} = \text{Adj}\mathbf{J} = \sum_{i=1}^{N-1} \beta_i (\mathbf{J})^i, \quad (6.32)$$

où $\mu_i = f(\beta_i)$ est un terme dépendant des coefficients de Bernoulli :

$$\beta_i = \frac{N!}{i \times (N-i)!} \quad (6.33)$$

Dans le cas où les fonctions des blocs ont des dérivées constantes, par exemple, des gains C_i , la Jacobienne analytique est :

$$\mathbf{J} = \begin{pmatrix} 1 & 0 & 0 & 0 & -C_1 \\ -C_2 & 1 & 0 & 0 & 0 \\ 0 & -C_3 & 1 & 0 & 0 \\ 0 & 0 & -C_4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \quad (6.34)$$

La procédure montrée plus haut donne ainsi, pour la première ligne de l'inverse de (6.34) :

$$\begin{aligned}
\Delta_{\mathbf{J}} \mathbf{J}^{-1}([1, 2, 3, 4, 5], 1) &= \begin{bmatrix} (-1)^4 \times 1 \times 1 \times 1 \times 1 \\ (-1)^0 \left(-\frac{\partial}{\partial x_2} f_3 \right) \left(-\frac{\partial}{\partial x_3} f_4 \right) \frac{\partial}{\partial x_4} f_5 \left(-\frac{\partial}{\partial x_5} f_1 \right) \\ (-1)^1 \left(-\frac{\partial}{\partial x_3} f_4 \right) \frac{\partial}{\partial x_4} f_5 \left(-\frac{\partial}{\partial x_5} f_1 \right) \times 1 \\ (-1)^2 \left(\frac{\partial}{\partial x_4} f_5 \right) \left(-\frac{\partial}{\partial x_5} f_1 \right) \times 1 \times 1 \\ (-1)^3 \left(-\frac{\partial}{\partial x_5} f_1 \right) \times 1 \times 1 \times 1 \end{bmatrix}^T, \\
\Delta_{\mathbf{J}} \mathbf{J}^{-1}([1, 2, 3, 4, 5], 1) &= \begin{bmatrix} 1 \\ -\left(\frac{\partial}{\partial x_2} f_3 \right) \left(\frac{\partial}{\partial x_3} f_4 \right) \frac{\partial}{\partial x_4} f_5 \left(\frac{\partial}{\partial x_5} f_1 \right) \\ -\left(\frac{\partial}{\partial x_3} f_4 \right) \frac{\partial}{\partial x_4} f_5 \left(\frac{\partial}{\partial x_5} f_1 \right) \times 1 \\ -\left(\frac{\partial}{\partial x_4} f_5 \right) \left(\frac{\partial}{\partial x_5} f_1 \right) \times 1 \times 1 \\ \left(\frac{\partial}{\partial x_5} f_1 \right) \times 1 \times 1 \times 1 \end{bmatrix}^T. \tag{6.35}
\end{aligned}$$

Ce qui se réduit, dans le cas des blocs constants, à :

$$\Delta_{\mathbf{J}} \mathbf{J}^{-1}([1, 2, 3, 4, 5], 1) = \begin{bmatrix} 1 \\ -(C_3)(C_4)C_5(C_1) \\ -(C_4)C_5(C_1) \\ -(C_5)(C_1) \\ (C_1) \end{bmatrix}^T, \tag{6.36}$$

où, on sait que $C_5=1$. Le déterminant a été calculé en (6.20). Notez qu'on abuse de la représentation en omettant le deuxième indice. Dans ce cas, une seule branche (autre que propre) aboutit sur le nœud. On garde seulement le nom du nœud.

Le calcul symbolique de l'inverse de (6.34) est :

$$\mathbf{J}^{-1} = \frac{1}{1 + C_4 C_3 C_2 C_1} \begin{pmatrix} 1 & -C_3 C_4 C_1 & -C_4 C_1 & -C_1 & C_1 \\ C_2 & 1 & -C_2 C_1 C_4 & -C_2 C_1 & C_2 C_1 \\ C_3 C_2 & C_3 & 1 & -C_3 C_2 C_1 & C_3 C_2 C_1 \\ C_4 C_3 C_2 & C_4 C_3 & C_4 & 1 & C_4 C_3 C_2 C_1 \\ -C_2 C_3 C_4 & -C_3 C_4 & -C_4 & -1 & 1 \end{pmatrix}. \quad (6.37)$$

On voit très bien que le calcul symbolique en (6.37) ainsi que l'analyse topologique menant à (6.36) donnent les mêmes éléments pour l'inverse de la Jacobienne pour la première ligne. On peut aisément montrer l'équivalence pour les autres lignes. Notons que ceci s'applique à une seule boucle de rétroaction. Cette analyse peut d'ailleurs être généralisée pour les autres cas. En effet, les « connexions de sous-graphes 1-facteur entre deux nœuds » et les « sous-graphes 1-facteur » sont génériques d'une part, et, d'autre part, la formule (6.21) n'est pas limitée aux seuls graphes uni-cycles. Le nombre des termes et produits dans les sous-graphes seront néanmoins largement plus élevés.

Nous venons donc d'établir la forme générale (analytique) de la matrice Jacobienne et son inverse. Pour rappel, le but est d'estimer les normes de ces dernières pour des conditions initiales choisies au début d'itérations de type Newton. En deuxième lieu, nous verrons comment ces normes changent avec la transformation DIPR. Passons justement à cette transformation pour réduire le système.

Considérons maintenant le même système transformé par le DIPR en utilisant une seule coupe en e , le système est celui montré en Figure 6-7. Le système d'équations de ce système de contrôle dans la représentation DIPR est :

$$e = u - g(e) = u - f_4 f_3 f_2 f_1(e). \quad (6.38)$$

Le processus de Newton pour résoudre ce système réduit est :

$$\mathbf{x}^k = \mathbf{x}^{k-1} - \mathbf{J}^{-1} \mathbf{\Phi}^k, \quad (6.39)$$

où $\mathbf{x}^k = e$ et le vecteur de la seule fonction objective $\mathbf{\Phi}^k$ et la Jacobienne \mathbf{J}^k sont respectivement :

$$\mathbf{\Phi}^k \equiv e - u + f_4 f_3 f_2 f_1(e) = 0 \text{ et :}$$

$$\mathbf{J}^R = 1 + \frac{\partial}{\partial e} f_1 \prod \frac{\partial}{\partial x_{j-1}} f_j = \frac{\partial}{\partial e} f_1 \frac{\partial}{\partial x_1} f_2 \frac{\partial}{\partial x_1} f_3 \frac{\partial}{\partial x_1} f_4, \quad (6.40)$$

$$\mathbf{J}^R = 1 + C_4 C_3 C_2 C_1, \quad (6.41)$$

et, finalement,

$$\mathbf{J}^{R^{-1}} = \frac{1}{1 + C_4 C_3 C_2 C_1} \quad (6.42)$$

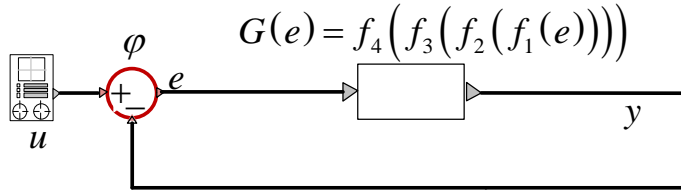


Figure 6-7: Système de contrôle simple transformé selon la DIPR pour l'analyse de l'existence et d'unicité de la solution pour système complet et réduit

On peut établir les expressions (6.41) et (6.42) par analyse topologique comme dans le cas du système complet traité ci-haut. Le graphe de Coates associé à la transformation DIPR et la solution discrétisée est présentée dans la Figure 6-8 :

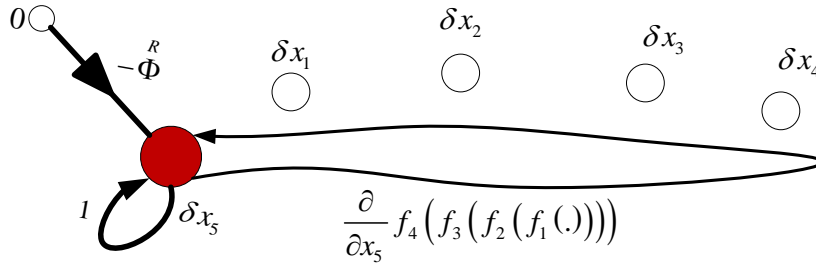


Figure 6-8: Graphe de Coates du système de contrôle simple de la Figure 6-7 discrétisé selon (6.39)

Il faut remarquer que la transformation DIPR s'apparente à l'élimination des nœuds et des branches dans le graphe et au remplacement, par une branche équivalente, des branches supprimées.

Dans les lignes qui suivent nous allons caractériser formellement cette opération pour le cas simple de la Figure 6-4 menant à la Figure 6-8. Le but étant d'établir comment les deux

inverses de la Jacobienne sont liées, c'est-à-dire les liens entre (6.42) et (6.37). On fera abstraction du fait que ce lien peut être fait par inspection dans ce cas simple car la procédure est générique pour inclure plusieurs points de rétroaction.

Quelques notations avant de continuer. Pour la suite, nous notons $\mathbf{gr}(\mathbf{N})$ le graphe des nœuds contenus dans l'ensemble \mathbf{N} comportant N nœuds. Le graphe réduit contient moins des nœuds que le graphe $\mathbf{gr}(\mathbf{N})$; soit \mathbf{N}_r , l'ensemble des nœuds restant après élimination des nœuds dans \mathbf{N} . On notera par $\mathbf{gr}(\mathbf{N}-\mathbf{N}_r)$, le graphe des nœuds supprimés. Éliminer un nœud (ou une branche) revient à supprimer toute branche (nœud) partant ou aboutissant à ce nœud (branche). Pour le cas simple sous étude, on a le graphe $\mathbf{gr}(\mathbf{N}-\mathbf{N}_r)$ dans la Figure 6-9 où on ne garde que le seul nœud représentant la rétroaction, la fameuse variable au point de rétroaction e adopté plus tôt dans cette thèse.

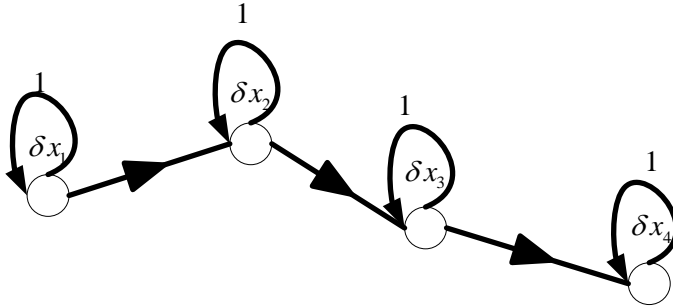


Figure 6-9: Graphe de Coates des nœuds supprimés $\mathbf{gr}(\mathbf{N}-\mathbf{N}_r)$ du graphe $\mathbf{gr}(\mathbf{N})$ de la Figure 6-4: Graphe modifié pour la solution ___ pour un cas simple vers le graphe $\mathbf{gr}(\mathbf{N}_r)$ de la Figure 6-8

On peut maintenant écrire la fonction composée du chemin de rétroaction, c'est-à-dire la branche équivalente aux branches supprimées du graphe complet vers le graphe réduit. On la déduit des résultats de [32], pages 156 et 154.

La branche équivalente C_{ij}^R entre des nœuds choisis comme points de rétroaction, soit la branche de i vers j est, voir relation 3.28 dans [32] :

$$C_{ij}^R = \frac{(-1)^\alpha}{K_c} \sum_{\mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))}^{\text{per}(\text{Con}(\mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))))} (-1)^{B_{\mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))}} \prod \mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j)), \quad (6.43)$$

où, on a :

$\mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))$: Graphes connexions 1-facteur entre les nœuds i et j du graphe $\mathbf{gr}((\mathbf{N}-\mathbf{N}_r)+\{i,j\})$, voir sa définition dans l'explication en bas de (6.21) et les illustrations dans la Figure 6-6;

$\mathbf{gr}((\mathbf{N}-\mathbf{N}_r)+\{i,j\})$: Graphe des nœuds supprimés auquel on rajoute les nœuds i et j

α : Nombre des nœuds dans le graphe $\mathbf{gr}((\mathbf{N}-\mathbf{N}_r)+\{i,j\})$;

$B\mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))$: Nombre des boucles dans le graphe $\mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))$. Incluant les boucles propres sur les nœuds ;

$\prod \mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))$: Produit des gains des branches sur les $B\mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))$ boucles (pour rappel, ces gains sont équivalents aux dérivées partielles) ;

$per\left(\text{Con}\left(\mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))\right)\right)$: Permanent de la matrice de connectivité de $\mathbf{grfaX}((\mathbf{N}-\mathbf{N}_r)+\{i,j\},(i,j))$. Pour rappel on supprime la i -ième ligne et la j -ième colonne de la matrice de connectivité de $\mathbf{gr}((\mathbf{N}-\mathbf{N}_r)+\{i,j\})$.

Finalement K_c est le déterminant du graphe $\mathbf{gr}(\mathbf{N}-\mathbf{N}_r)$ des nœuds supprimés comme celui en la Figure 6-9. On a la relation suivante pour faire le lien avec la notation adoptée plus haut :

$$K_c = \Delta_{\mathbf{gr}(\mathbf{N}-\mathbf{N}_r)} \quad (6.44)$$

Une deuxième façon d'établir la relation (6.43), celle que nous illustrerons, utilise la transmission de graphe du nœud i vers le nœud j . Introduisons cette notion. La transmission d'un graphe $\mathbf{gr}(\mathbf{N})$ de i (de degré un) vers j , que nous notons $Tr(\mathbf{gr}(\mathbf{N})i \rightarrow j)$ est défini comme [32], page 154 :

$$Tr(\mathbf{gr}(\mathbf{N}) i \rightarrow j) = \frac{\sum_{\mathbf{grfaX}((\mathbf{N}), (i, j))}^{per(\text{Con}(\mathbf{grfaX}((\mathbf{N}), (i, j))))} (-1)^{B_{\mathbf{grfaX}((\mathbf{N}), (i, j))}} \prod \mathbf{grfaX}((\mathbf{N}), (i, j))}{\sum_{\mathbf{grfa}(\mathbf{N}-\{i\})}^{per(\text{Con}(\mathbf{gr}(\mathbf{N}-\{i\})))} (-1)^{B_{\mathbf{grfa}(\mathbf{N}-\{i\})}} \prod \mathbf{grfa}(\mathbf{N}-\{i\})} \quad (6.45)$$

On achève l'établissement des branches après élimination des nœuds $\mathbf{N}-\mathbf{N}_r$ en construisant deux sections des graphes particulières pour chaque branche équivalente :

- a. Si $i=j$: on remplace le nœud i par deux nœuds i' et i'' pour avoir les mêmes branches qui partaient de i dans le graphe $\mathbf{gr}((\mathbf{N}-\mathbf{N}_r) + \{i, j=i\})$, elles partent maintenant de i' , et, celles qui arrivaient sur i dans ce même graphe $\mathbf{gr}((\mathbf{N}-\mathbf{N}_r) + \{i, j=i\})$, elles arrivent maintenant sur i'' . De plus, on ajoute une boucle propre de gain -1 sur i'' . (On illustrera ceci sur le cas sous étude). Notons ce graphe par $\mathbf{grD}((\mathbf{N}-\mathbf{N}_r) + \{i, j=i\})$, « D » indiquant que le nœud i est divisé (virtuellement) en deux ;
- b. Si $i \neq j$, on enlève toutes les branches terminant en i et toutes les branches partant de j . De plus, on ajoute une boucle propre de gain -1 sur j . On forme cette fois-ci le graphe $\mathbf{grD}((\mathbf{N}-\mathbf{N}_r) + \{i, j \neq i\})$.

D'une manière équivalente à (6.43) on a la branche après réduction $\overset{R}{C}_{ij}$ entre des nœuds choisis comme points de rétroaction [32], page 156-157 :

$$\overset{R}{C}_{ij} = \begin{cases} Tr(\mathbf{grD}((\mathbf{N}-\mathbf{N}_r) + \{i, j \neq i\}) i \rightarrow j) & i \neq j \\ Tr(\mathbf{grD}((\mathbf{N}-\mathbf{N}_r) + \{i, j=i\}) i' \rightarrow i'') & i = j \end{cases}, \quad (6.46)$$

avec $Tr(.)$ défini dans (6.45) et les sections des graphes $\mathbf{grD}((\mathbf{N}-\mathbf{N}_r) + \{i, j\})$ définis plus haut dans a et b. Appliquons cette analyse au cas simple traité plus haut :

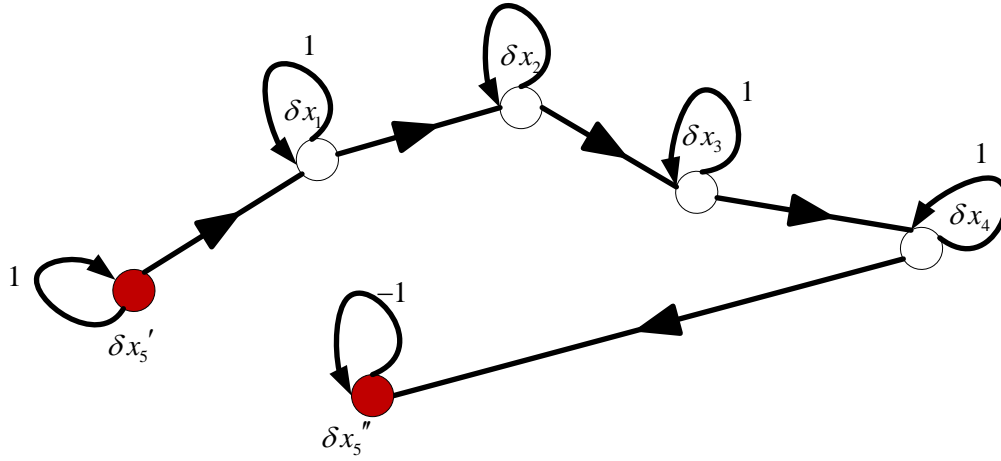


Figure 6-10: Section du graphe $\mathbf{grD}((\mathbf{N}-\mathbf{N}_r)+\{5,5\})$: ajout du nœud 5 sur le graphe de Coates des nœuds supprimés $\mathbf{gr}(\mathbf{N}-\mathbf{N}_r)$

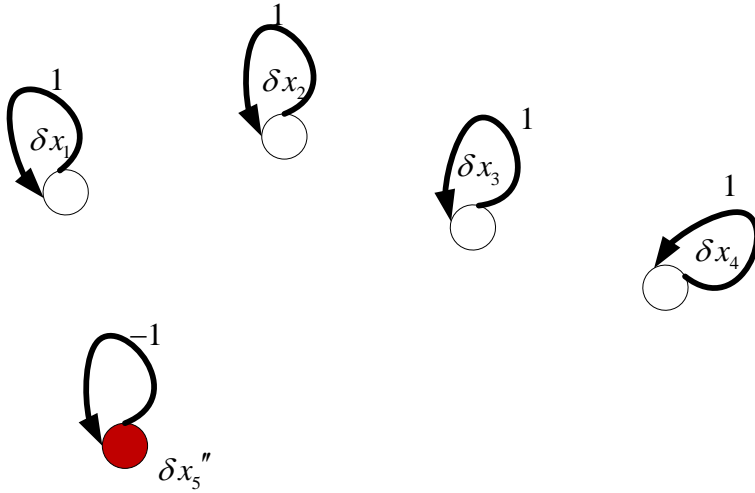


Figure 6-11: Sous-graphes 1-facteur $\mathbf{grDfa}((\mathbf{N}-\mathbf{N}_r)+\{i, j=i\}-\{x_5'\})$: suppression du nœud source 5 dans la section du graphe $\mathbf{grD}((\mathbf{N}-\mathbf{N}_r)+\{5,5\})$

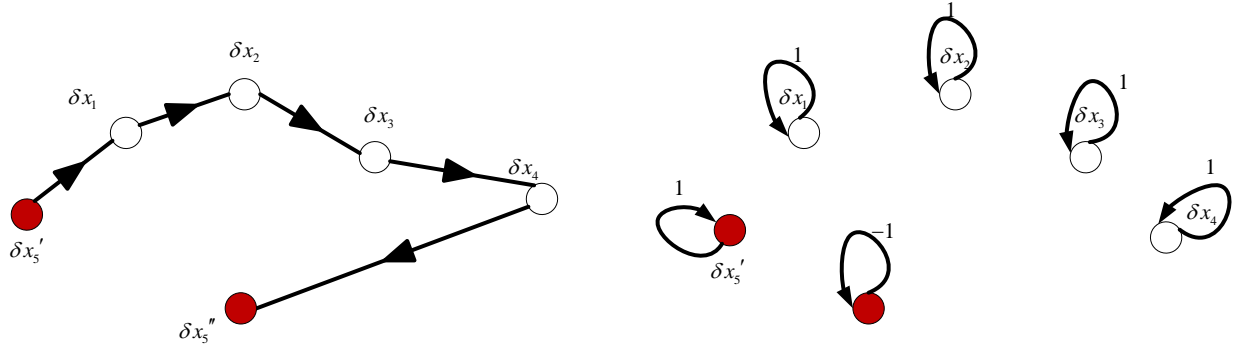


Figure 6-12: Connexions dans les sous-graphes 1-facteur $\mathbf{grDfaX}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5,5\}\right)(x_5',x_5'')\right)$ entre les nœuds (x_5',x_5'') dans la section du graphe $\mathbf{grD}\left((\mathbf{N}-\mathbf{N}_r)+\{5,5\}\right)$

En se basant sur la définition de la transmission entre deux nœuds de graphe donnée en (6.45) :

$$\begin{aligned}
 C_{55}^R &= Tr\left(\mathbf{grD}\left((\mathbf{N}-\mathbf{N}_r)+\{i,j=i\}\right)i' \rightarrow i''\right) \\
 &= \frac{\sum_{\substack{\text{per}\left(\text{Con}\left(\mathbf{grDfaX}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right),(5',5'')\right)\right)\right) \\ \mathbf{grDfaX}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right),(5',5'')\right)}}{\sum_{\substack{\text{per}\left(\text{Con}\left(\mathbf{grD}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right)-\{5'\}\right)\right)\right) \\ \mathbf{grDfa}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right)-\{5'\}\right)}} Bn \prod \mathbf{grDfaX}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right),(5',5'')\right), \quad (6.47)
 \end{aligned}$$

avec :

$$\begin{aligned}
 Bn &= (-1)^{B\mathbf{grDfaX}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right),(5',5'')\right)} \\
 Bd &= (-1)^{B\mathbf{grDfa}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right)-\{5'\}\right)} ,
 \end{aligned}$$

où, les sous graphes 1-facteur $\mathbf{grDfa}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right)-\{5'\}\right)$ et les connexions associées $\mathbf{grDfaX}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right),(5',5'')\right)$ sont respectivement dans la Figure 6-11 et dans la Figure 6-12. On y déduit notamment :

- Le nombre de sous-graphes 1-facteur par le calcul du permanent de la matrice de connectivité associée :

$$\text{per}\left(\text{Con}\left(\mathbf{grD}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right)-\{5'\}\right)\right)\right)=1$$

- Le nombre de sous-graphes dans la connexion 1-facteur du permanent de la matrice de connectivité associée :

$$\text{per}\left(\text{Con}\left(\text{grDfaX}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right), (5',5'')\right)\right)\right)=1$$

- Pour les boucles du seul sous-graphe 1-facteur, on a :

$$\text{BgrDfa}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right)-\{5'\}\right)=[5]$$

- Pour les boucles du seul sous-graphe où on a la connexion $5' \rightarrow 5''$, on a :

$$\text{BgrDfaX}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right), (5',5'')\right)=[0]. \text{ C'est-à-dire, aucune boucle.}$$

On a donc (6.47):

$${}^R C_{55} = \frac{\sum_{\text{grDfaX}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right), (5',5'')\right)}^1 (-1)^0 [C_1 \times C_2 \times C_3 \times C_4 \times C_5]}{\sum_{\text{grDfa}\left(\left((\mathbf{N}-\mathbf{N}_r)+\{5',5''\}\right)-\{5'\}\right)}^1 (-1)^5 [-1 \times 1 \times 1 \times 1]} = C_1 C_2 C_3 C_4 C_5, \quad (6.48)$$

où les ${}^R C_{ij}$ sont les gains du nœud i vers les nœuds j ou encore la dérivée partielle selon i de la fonction qui est associée au nœud j . Le graphe réduit qui représente la formulation DIPR est maintenant entièrement défini. On porte (6.48) dans la Figure 6-8 :

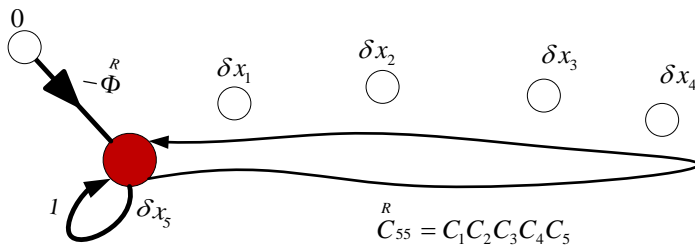


Figure 6-13: Graphe de Coates $\text{gr}(\mathbf{N}_r + \{\Phi\})$: Remplacement du gain équivalent après réduction du graphe selon la DIPR et ajout de la variable de la fonction objective $\left({}^R \mathbf{J} \delta \mathbf{x} = {}^R \Phi\right)$ comme en la Figure 6-4

En réécrivant la seule équation associée à ce graphe de Coates pour le système linéarisé, on retrouve la matrice Jacobienne après réduction par la DIPR, laquelle est semblable à (6.41) établie par inspection :

$$\delta x_5 + C_1 C_2 C_3 C_4 C_5 \delta x_5 - \overset{R}{\Phi} = 0 \quad (6.49)$$

$$\overset{R}{\mathbf{J}} = 1 + C_1 C_2 C_3 C_4 C_5 \quad (6.50)$$

Les mêmes outils peuvent aider à établir la forme de l'inverse de la Jacobienne comme on l'a fait pour le cas complet, en utilisant le lien entre l'inverse de la matrice, le déterminant et les cofacteurs. On a établi ce lien en associant (6.21) et (6.30). La même procédure donne pour le calcul de la seule variable δx_5 :

$$\delta x_5 = \frac{\sum_{\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5))} (-1)^{B_{\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5))}} \prod \mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5))}{\sum_{\mathbf{grfa}(\mathbf{N}_r)} (-1)^{B_{\mathbf{grfa}(\mathbf{N}_r)}} \prod \mathbf{grfa}(\mathbf{N}_r)}, \quad (6.51)$$

où :

- $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5))$ est l'ensemble des connexions 1-facteur du nœud $\{0\}$ des fonctions Φ (noté $\{\Phi\}$) vers le nœud δx_5 dans le sous-graphe $\mathbf{gr}(\mathbf{N}_r + \{\Phi\})$;
- $\mathbf{grfa}(\mathbf{N}_r)$ est l'ensemble des sous-graphes 1-facteur de $\mathbf{gr}(\mathbf{N}_r)$; et, pour rappel,
- $\prod \mathbf{gr}_-$ est le produit des gains associés aux branches des sous-graphes \mathbf{gr}_- considéré.

Les sous-graphes du graphe de la Figure 6-13 intervenant dans (6.51) sont repris dans la Figure 6-14 ci-dessous :

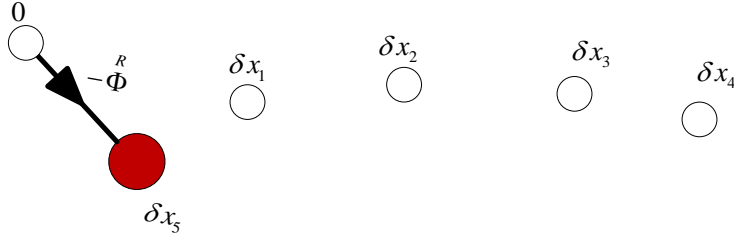


Figure 6-14: Connexion 1-facteur $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5))$ de Φ vers δx_5 dans le graphe de Coates $\mathbf{gr}(\mathbf{N}_r + \{\Phi\})$

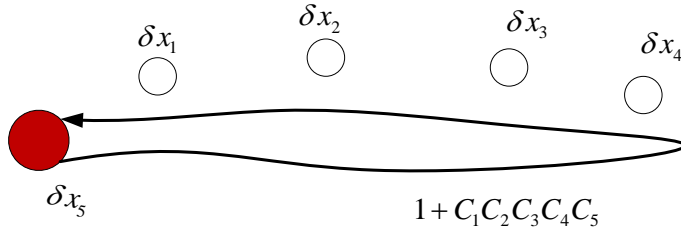


Figure 6-15: Graphe de Coates $\mathbf{gr}(\mathbf{N}_r)$ du système réduit par la DIPR, équivalent au sous-graphe 1-facteur $\mathbf{grfa}(\mathbf{N}_r)$

Se basant sur les sous-graphes de la Figure 6-14 et de la Figure 6-15, on a les relations suivantes :

$$\left\{ \begin{array}{l} \text{permanent}\left(\text{con}\left(\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5))\right)\right) = 1 \\ \mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5)) = \{0, \delta x_5\} - \{\text{boucle propre sur } \delta x_5\} \\ B\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5)) = 0 \\ \prod \mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_5)) = -\overset{R}{\Phi} \\ \text{permanent}\left(\text{con}\left(\mathbf{gr}(\mathbf{N}_r)\right)\right) = 1 \\ \mathbf{grfa}(\mathbf{N}_r) = \{\delta x_5\} \\ B\mathbf{grfa}(\mathbf{N}_r) = 1 \\ \prod \mathbf{grfa}(\mathbf{N}_r) = 1 + C_1 C_2 C_3 C_4 C_5 \end{array} \right. . \quad (6.52)$$

Portant (6.52) dans (6.51), on a pour l'inconnue δx_5 :

$$\delta x_5 = \frac{(-1)^0 \binom{R}{-\Phi}}{(-1)^1 (1 + C_{55}^R)} = \frac{(-1)^0 \binom{R}{-\Phi}}{(-1)^1 (1 + C_1 C_2 C_3 C_4 C_5)} \quad (6.53)$$

On achève le calcul de l'inverse de la Jacobienne $\binom{R}{\mathbf{J}}^{-1}$ en considérant que l'équation discrétisée est (voir plus haut en (6.14)) :

$$\delta x_5 = \binom{R}{\mathbf{J}}^{-1} \binom{R}{\Phi}. \quad (6.54)$$

On déduit en isolant $\binom{R}{\mathbf{J}}^{-1}$ dans la combinaison de (6.54) et de (6.53) ;

$$\binom{R}{\mathbf{J}}^{-1} = \frac{1}{(1 + C_1 C_2 C_3 C_4 C_5)}. \quad (6.55)$$

On retrouve alors l'inverse de la Jacobienne du système réduit par analyse du graphe identique à la valeur trouvée par inspection du système d'équations en (6.42) avec $C_5 = 1$.

Notez encore que ce résultat peut être trouvé par le déterminant et le cofacteur de la matrice de chacun des éléments. De plus, la forme générique de (6.55) est :

$$\mathbf{J}^{-1}(i, j) = \frac{\binom{\Delta_R}{\mathbf{J}}^{ij}}{\Delta_R} = \frac{\sum_{\Phi, \delta x_j} (-1)^{B_{\text{grfaX}}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_j, \dots, \delta x_5)) - 1} \prod_{\text{gain} \neq \Phi_j} \text{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_j, \dots, \delta x_i))}{\sum_{\text{grfa}(\mathbf{N}_r)} (-1)^{B_{\text{grfa}}(\mathbf{N}_r)} \prod \text{grfa}(\mathbf{N}_r)}, \quad (6.56)$$

où Δ_R et $\binom{\Delta_R}{\mathbf{J}}^{ij}$ ($\Delta_{\mathbf{J}}^{ij}$) sont respectivement le déterminant du système réduit et le cofacteur (i, j) de ce dernier. Où, $\text{grfa}(\mathbf{N}_r)$ représentent les sous-graphes 1-facteur dans le graphe réduit $\text{gr}(\mathbf{N}_r)$ et, chacun des sous-graphes $\text{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_j, \dots, \delta x_i))$ est la connexion 1-facteur de Φ vers δx_i qui comporte δx_j dans le chemin direct. En d'autres termes, δx_j n'est pas dans une boucle faisant partie du sous-graphe 1-facteur. Dans les sous-graphes de la Figure 6-6, on peut bien voir que le parcours direct de la connexion passe sur δx_j . Le gain associé à Φ_j n'est cependant pas inclus dans le produit des gains comme, indiqué dans

l'expression $\prod_{\text{gain} \neq \Phi_j} \mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_j, \dots, \delta x_i))$. La sommation $\sum_{\Phi, \delta x_j}$ se fait sur les sous-graphes ayant la combinaison des nœuds $\Phi, \delta x_j$ dans le chemin de la connexion vers δx_i .

La relation (6.56) nous introduit justement dans le cas général avec plusieurs points de rétroaction. Ce cas est traité dans la section suivante. Les formes génériques des matrices \mathbf{J} , \mathbf{J}^{-1} pour le cas d'une seule boucle ci-haut tout comme pour le cas suivant de plusieurs boucles, ouvriront la possibilité d'exprimer les bornes des normes de ces matrices indispensables dans le théorème de Kantorovich.

6.2.2 Cas avec plusieurs points de rétroaction

Quand on a plusieurs points de rétroaction, il faut établir pour chaque branche le gain (composition des dérivées partielles) par les équations de type (6.46). Le nombre de branches possibles à déterminer est $N_r \times N_r$, dont $(N_r \times N_r) - N_r$ gains mutuels de type $\overset{R}{C}_{ij}$ et N_r gains propres de type $\overset{R}{C}_{ii}$. La Figure 6-16 présente le graphe de Coates avec N_r points de rétroaction. Les branches discontinues sont celles du i -ième nœud et doivent partir des autres $N_r - 1$ nœuds, ou y arriver. Clairement, chaque nœud aura $N_r - 1$ branches entrantes et $N_r - 1$ sortantes dans le cas général où la matrice Jacobienne est pleine.

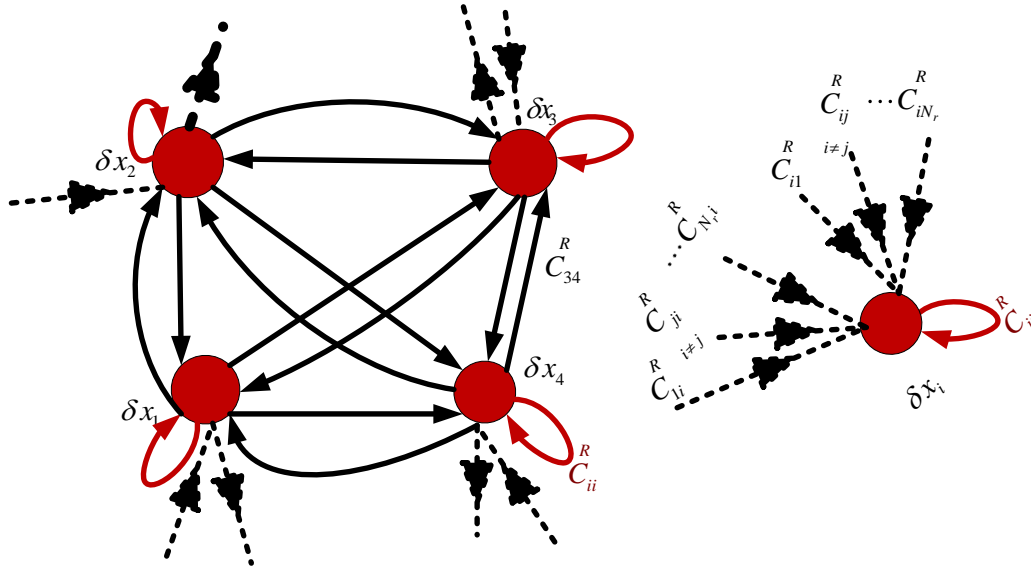


Figure 6-16: Graphe de Coates pour le cas de réduction DIPR avec plusieurs points de rétroaction $\mathbf{gr}(\mathbf{N}_r)$ fortement connectés.

La procédure établie pour le cas précédent simple s'applique aussi pour ce cas plus complexe. Nous résumons en donnant la forme de la Jacobienne pour de tels cas. Pour plus de clarté, nous présentons le résultat d'un cas de deux points de rétroaction à la Figure 6-17. Notez d'ailleurs que pour tout nombre des points de rétroaction il faut établir les branches équivalentes propres et mutuelles, exactement pour le cas de deux points de rétroaction. Le gain de la boucle propre est noté ici C_{ii}^R , mais il faut lire $1 + C_{ii}^R$ pour avoir une représentation conforme au graphe de Coates.

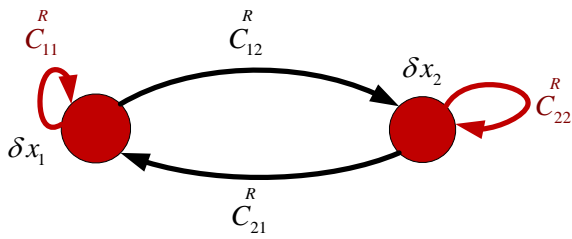


Figure 6-17: Graphe de Coates pour le cas de réduction DIPR avec deux points de rétroaction $\mathbf{gr}(\mathbf{N}_r) = \mathbf{gr}(\{1, 2\})$.

Les deux sous-graphes 1-facteur de $\mathbf{grfa}(\mathbf{N}_r)$ sont dans a) et b) de la Figure 6-18. En effet, à partir du calcul de permanent de la matrice de connectivité, on a vu que le nombre des sous-graphes que nous notons $\# \mathbf{grfa}(\mathbf{N}_r)$ vaut :

$$\# \mathbf{grfa}(\mathbf{N}_r) = \text{permanent}(\text{con}(\mathbf{gr}(\mathbf{N}_r))). \quad (6.57)$$

Observons aussi que si on considère que tous les nœuds possèdent une et une seule branche partant d'un autre nœud, alors, pour chaque paire de nœuds, on a deux branches distinctes. Le graphe est donc fortement connecté. On peut atteindre chaque nœud par n'importe quel nœud. La matrice Jacobienne sera donc pleine (en comparaison à la matrice creuse du cas complet). Avec cette hypothèse, par ailleurs indiquée pour établir les bornes supérieures des normes définies en (6.4), nous avons la matrice de connectivité remplie comme suit :

$$\text{con}(\mathbf{gr}(\mathbf{N}_r)) = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \ddots & 1 \\ \vdots & \ddots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}^{N_r \times N_r} \quad (6.58)$$

Pour lesquelles on aura la relation :

$$\# \mathbf{grfa}(\mathbf{N}_r) = \text{permanent} \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \ddots & 1 \\ \vdots & \ddots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}^{N_r \times N_r} = \text{factoriel}(N_r), \quad (6.59)$$

où N_r est le nombre des nœuds dans le graphe réduit. Pour rappel, il s'agit du nombre de points de rétroaction. Noter qu'on a adopté le symbole $()^{N_r \times N_r}$ quand la matrice est de dimension $N_r \times N_r$. En revenant à la notation du développement dans le chapitre précédent, on a $N_r = M$.

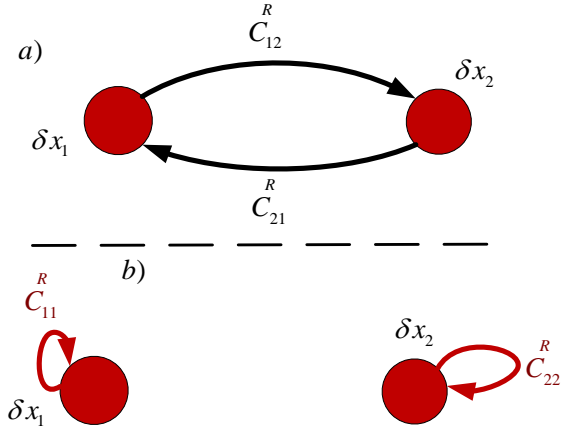


Figure 6-18: Deux sous-graphes 1-facteur $\mathbf{grfa}(\mathbf{N}_r) = \mathbf{grfa}(\{1, 2\})$ pour un cas générique de deux points de rétroaction $\mathbf{gr}(\mathbf{N}_r)$.

On vérifie qu'en appliquant (6.59) sur le graphe de la Figure 6-17 on a deux sous-graphes dans la Figure 6-18 :

$$\#\mathbf{grfa}(\{1, 2\}) = \text{permanent} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \text{factoriel}(2) = 2 \quad (6.60)$$

Les termes associés à ces deux sous-graphes interviennent dans le dénominateur des éléments de l'inverse de la matrice Jacobienne en (6.56). Analysons maintenant les termes du numérateur en (6.56) à partir des connexions 1-facteur entre l'élément Φ et chaque nœud. On a donc le graphe $\mathbf{gr}(\mathbf{N}_r + \{\Phi\})$ et les sous-graphes $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_1))$ et $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_2))$ dans, respectivement, a), b) et c) , d) et f) dans la Figure 6-19.

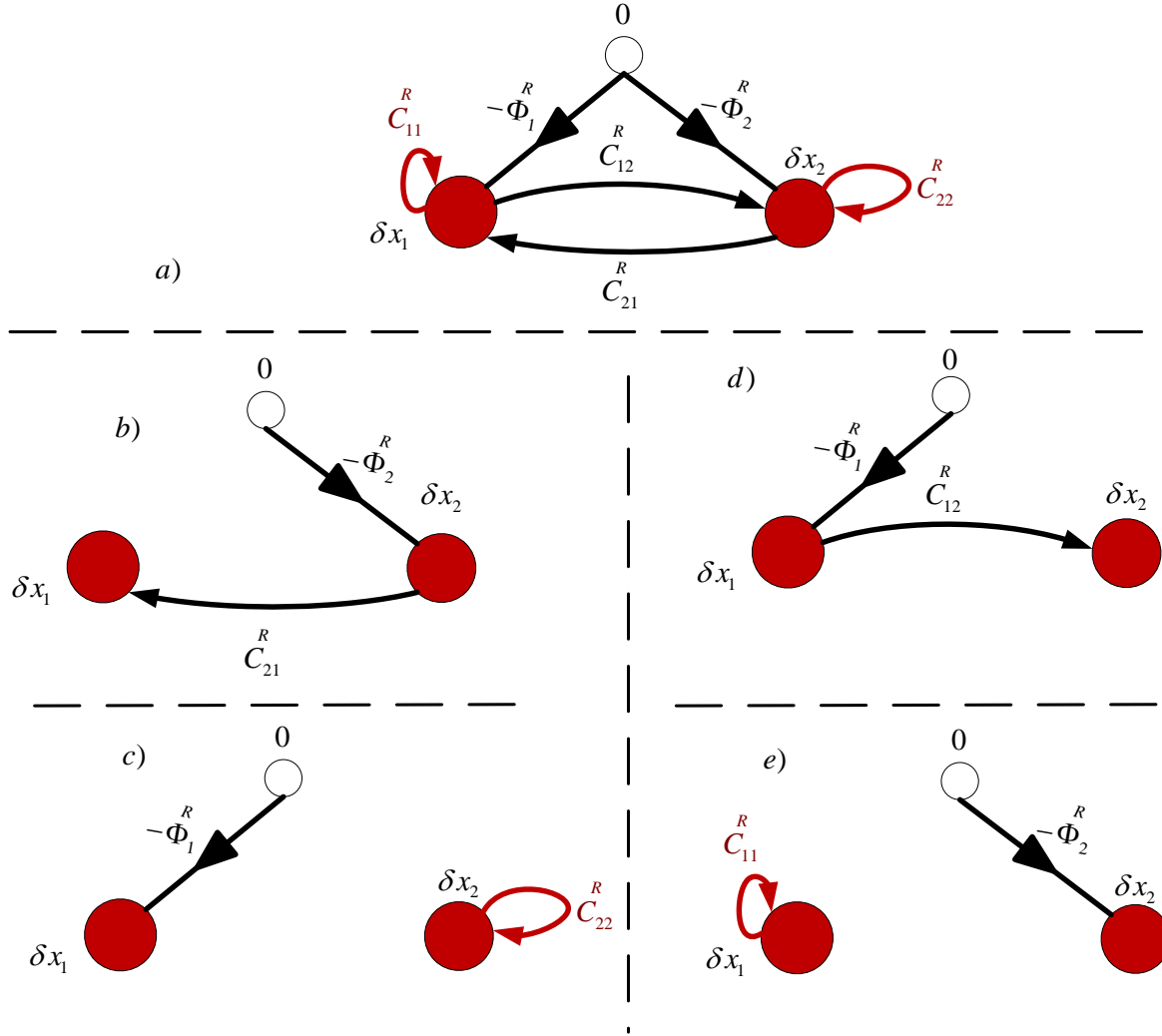


Figure 6-19: Différents sous-graphes pour la connexion 1-facteur : a) Graphe augmenté $\mathbf{gr}(\mathbf{N}_r + \{\Phi\})$; b) et c) Sous-graphes connexion 1-facteur de l'ensemble $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_1))$; d et e) Sous-graphes connexion 1-facteur de l'ensemble $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_2))$.

On vérifie encore une fois, via le calcul de permanent de la matrice de connectivité pour les connexions 1-facteur, qu'on a bien deux sous-graphes dans chacune des connexions 1-facteur de $\Phi \rightarrow \delta x_i$. Cette fois-ci, la matrice à considérer est différente, comme on l'a introduit dans (6.28). Pour rappel, on tient compte de la connexion $\Phi \rightarrow \delta x_i$ dans la matrice de connectivité en

supprimant la ligne Φ et la colonne de δx_i dans la matrice du graphe $\mathbf{gr}(\mathbf{N}_r + \{\Phi\})$. On transforme (6.58) pour former la matrice de connectivité $\mathbf{gr}(\mathbf{N}_r + \{\Phi\})$ comme suit :

$$\text{con}(\mathbf{gr}(\mathbf{N}_r + \{\Phi\})) = \begin{matrix} 1^{er} \\ 2^{ième} \\ \vdots \\ N_r^{Nième} \\ \Phi : (N_r + 1)^{th} \end{matrix} \begin{pmatrix} 1 & 1 & \cdots & 1 & 0 \\ 1 & 1 & \ddots & 1 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 1 & 1 & \cdots & 1 & 0 \\ 1 & 1 & \cdots & 1 & 0 \end{pmatrix}^{(N_r+1) \times (N_r+1)} \quad (6.61)$$

La matrice de connectivité pour la connexion 1-facteur est alors la suivante :

$$\text{con}(\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_i))) = \begin{matrix} 1^{er} \\ i^{ième} \\ \vdots \\ N_r^{Nième} \\ \Phi : (N_r + 1)^{th} \end{matrix} \begin{pmatrix} 1 & 1 & \cdots & 1 & \text{X} \\ \text{X} & \text{X} & \times & \text{X} & \text{X} \\ \vdots & \ddots & \ddots & \vdots & \text{X} \\ 1 & 1 & \cdots & 1 & \text{X} \\ 1 & 1 & \cdots & 1 & \text{X} \end{pmatrix}^{((N_r+1)-1) \times ((N_r+1)-1)} \quad (6.62)$$

où, encore fois, X indique que l'élément est supprimé.

On applique les relations ci-haut pour le cas de réduction de deux variables pour avoir :

$$\# \mathbf{grfaX}(\{1, 2\} + \{\Phi\}, (\Phi, \delta x_1)) = \text{permanent} \begin{pmatrix} \text{X} & \text{X} & \text{X} \\ 1 & 1 & \text{X} \\ 1 & 1 & \text{X} \end{pmatrix}^{(3-1) \times (3-1)} = \text{factoriel}(3-1) = 2 \quad (6.63)$$

$$\# \mathbf{grfaX}(\{1, 2\} + \{\Phi\}, (\Phi, \delta x_2)) = \text{permanent} \begin{pmatrix} 1 & 1 & \text{X} \\ \text{X} & \text{X} & \text{X} \\ 1 & 1 & \text{X} \end{pmatrix}^{(3-1) \times (3-1)} = \text{factoriel}(3-1) = 2 \quad (6.64)$$

Les deux sous-graphes pour $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_1))$ et $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_2))$ dans respectivement b) et c) et d) et f) dans la Figure 6-19 sont ainsi déterminés par les relations du type (6.63). Les termes dans la sommation du numérateur de (6.56) sont aussi déterminés. Il reste à considérer les gains sur les connexions et sur les boucles dans chacun des sous-graphes.

Étant donné que par hypothèse, on a considéré le cas « fortement connecté », on a pour :

- chacun des termes du dénominateur (c'est-à-dire, le déterminant de la Jacobienne), \mathbf{N}_r facteurs qui sont des gains dans l'un des graphes $\mathbf{grfa}(\mathbf{N}_r)$ considéré,
- chacun des termes du numérateur aura $\mathbf{N}_r - 1$ facteurs pour la connexion de Φ au nœud δx_i . En effet, si on considère la connexion passant par δx_j juste après Φ , on l'a noté $\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_j, \dots, \delta x_i))$ plus haut, alors on aura nC_B gains associés à l'une des $B\mathbf{grfaX}(\mathbf{N}_r + \{\Phi\}, (\Phi, \delta x_j, \dots, \delta x_i))$ boucles et $\mathbf{N}_r - 1 - nC_B$ gains associés aux « marches » dans le graphe entre δx_j et δx_j .

On peut donc établir maintenant la forme générique de l'inverse de la Jacobienne réduite dans le cas d'un graphe réduit fortement connecté :

$$\left(\mathbf{J}^R\right)^{-1}_{ij} = \frac{\sum_1^{\text{factoriel}(\mathbf{N}_r-1)} (-1)^{B\mathbf{grfaX}(\mathbf{N}_r+\{\Phi\},(\Phi,\delta x_j,\dots,\delta x_i))-1} \left(\underbrace{C_- \times C_- \times \dots}_{nC_B}\right) \left(\underbrace{C_- \times C_- \times \dots}_{\mathbf{N}_r-1-nC_B}\right)}{\sum_1^{\text{factoriel}(\mathbf{N}_r)} (-1)^{B\mathbf{grfa}(\mathbf{N}_r)} \left(\underbrace{C_- \times C_- \times \dots}_{B\mathbf{grfa}(\mathbf{N}_r)}\right) \left(\underbrace{C_- \times C_- \times \dots}_{\mathbf{N}_r-B\mathbf{grfa}(\mathbf{N}_r)}\right)}, \quad (6.65)$$

où les C_- sont des gains équivalents trouvés après réduction du graphe $\mathbf{gr}(\mathbf{N})$ au graphe $\mathbf{gr}(\mathbf{N}_r)$ où les nœuds dans l'ensemble \mathbf{N}_r sont les points de rétroaction établis par DIPR. Pour rappel, ces C_- s sont de la forme (voir aussi (6.45) et l'explication subséquente) :

$$C_{ij}^R = \frac{\sum_{\mathbf{grfaX}((\mathbf{N}), (i,j))}^{\text{per}(\text{Con}(\mathbf{grfaX}((\mathbf{N}), (i,j))))} (-1)^{B\mathbf{grfaX}((\mathbf{N}), (i,j))} \prod \mathbf{grfaX}((\mathbf{N}), (i,j))}{\sum_{\mathbf{grfa}(\mathbf{N}-\{i\})}^{\text{per}(\text{Con}(\mathbf{gr}(\mathbf{N}-\{i\})))} (-1)^{B\mathbf{grfa}(\mathbf{N}-\{i\})} \prod \mathbf{grfa}(\mathbf{N}-\{i\})}. \quad (6.66)$$

Tous les autres termes dans (6.65) ont été définis plus hauts et pour n'insister que sur quelques uns, on a :

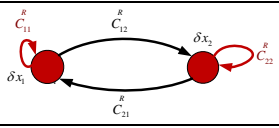
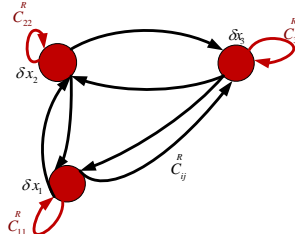
$\left(\mathbf{J}^R\right)^{-1}_{ij}$: Élément à la i -ième ligne et à la j -ième colonne dans l'inverse de la matrice de la Jacobienne du système réduit par la DIPR ;

N_r : L'ensemble des N_r nœuds (blocs de contrôle) retenus dans le graphe réduit (équivalent aux M points de rétroaction de la procédure DIPR sur le système de contrôle) ;

nC_B : nombre des gains apparaissant des $BgrfaX(N_r + \{\Phi\}, (\Phi, \delta x_j, \dots, \delta x_i))$ boucles du sous-graphe $grfaX(N_r + \{\Phi\}, (\Phi, \delta x_j, \dots, \delta x_i))$. Notez que le gain associé aux branches non-disjointes à δx_j n'y apparaîtront pas car δx_j n'est dans aucune des boucles de $grfaX(N_r + \{\Phi\}, (\Phi, \delta x_j, \dots, \delta x_i))$, par hypothèse.

Pour illustrer ce processus, on considère les résultats de réduction pour N_r respectivement deux trois et quatre points de rétroactions (voir du Tableau 6.1 au Tableau 6.3 et la Figure 6-20, où pour simplicité, nous notons C_{ij} en lieu et place de $\overset{R}{C}_{ij}$ adopté auparavant).

Tableau 6.1: Résultats de l'inverse de la Jacobienne dans le système réduit par la DIPR : Jacobienne, déterminant et inverses des Jacobiennes.

N_r	Graphe de Coates	$\det(\overset{R}{J})$	i -ième ligne de $(\overset{R}{J})^{-1} \times \det(\overset{R}{J})$	
2		C11C22 -C12C21	1	[C22, -C12]
			2	[-C21, C11]
3		C11C22C33 -C11C23C32 -C21C12C33 +C21C13C32 +C31C12C23 -C31C13C22	1	[C22C33-C23C32, -C12C33+C13C32, C12C23-C13C22]
			2	[-C21C33+C23C31, C11C33-C13C31, -C11C23+C13C21]
			3	[C21C32-C22C31, -C11C32+C12C31, C11C22-C12C21]

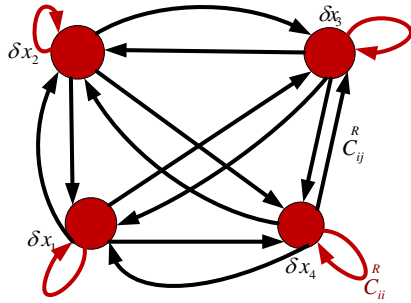


Figure 6-20: Graphes de Coates d'un système réduit à quatre points de rétroaction par la DIPR

Tableau 6.2: Résultats pour le déterminant de l'inverse de la Jacobienne pour le cas de quatre points de rétroaction définissant le système réduit par la DIPR $N_r = M = 4$.

$\det(\mathbf{J}^R)$
$C_{11}C_{22}C_{33}C_{44}-C_{11}C_{22}C_{34}C_{43}-C_{11}C_{32}C_{23}C_{44}+C_{11}C_{32}C_{24}C_{43}$ $+C_{11}C_{42}C_{23}C_{34}-C_{11}C_{42}C_{24}C_{33}-C_{21}C_{12}C_{33}C_{44}+C_{21}C_{12}C_{34}C_{43}$ $+C_{21}C_{32}C_{13}C_{44}-C_{21}C_{32}C_{14}C_{43}-C_{21}C_{42}C_{13}C_{34}+C_{21}C_{42}C_{14}C_{33}$ $+C_{31}C_{12}C_{23}C_{44}-C_{31}C_{12}C_{24}C_{43}-C_{31}C_{22}C_{13}C_{44}+C_{31}C_{22}C_{14}C_{43}$ $+C_{31}C_{42}C_{13}C_{24}-C_{31}C_{42}C_{14}C_{23}-C_{41}C_{12}C_{23}C_{34}+C_{41}C_{12}C_{24}C_{33}$ $+C_{41}C_{22}C_{13}C_{34}-C_{41}C_{22}C_{14}C_{33}-C_{41}C_{32}C_{13}C_{24}+C_{41}C_{32}C_{14}C_{23}$

Tableau 6.3: Résultats de l'inverse de la Jacobienne dans le système réduit par la DIPR : Jacobienne, déterminant et inverses des Jacobiennes $N_r = 4$.

i -ième ligne	$\left(\mathbf{J}^R\right)^{-1} \times \det\left(\mathbf{J}^R\right)$
1	[C22C33C44-C22C34C43-C32C23C44+C32C24C43+C42C23C34-C42C24C33, -C12C33C44+C12C34C43+C32C13C44-C32C14C43-C42C13C34+C42C14C33, C12C23C44-C12C24C43-C22C13C44+C22C14C43+C42C13C24-C42C14C23, -C12C23C34+C12C24C33+C22C13C34-C22C14C33-C32C13C24+C32C14C23]
2	[-C21C33C44+C21C34C43+C31C23C44-C31C24C43-C41C23C34+C41C24C33, C11C33C44-C11C34C43-C31C13C44+C31C14C43+C41C13C34-C41C14C33, -C11C23C44+C11C24C43+C21C13C44-C21C14C43-C41C13C24+C41C14C23, C11C23C34-C11C24C33-C21C13C34+C21C14C33+C31C13C24-C31C14C23]
3	[-C21C33C44+C21C34C43+C31C23C44-C31C24C43-C41C23C34+C41C24C33, C11C33C44-C11C34C43-C31C13C44+C31C14C43+C41C13C34-C41C14C33, -C11C23C44+C11C24C43+C21C13C44-C21C14C43-C41C13C24+C41C14C23, C11C23C34-C11C24C33-C21C13C34+C21C14C33+C31C13C24-C31C14C23]
4	[-C21C33C44+C21C34C43+C31C23C44-C31C24C43-C41C23C34+C41C24C33, C11C33C44-C11C34C43-C31C13C44+C31C14C43+C41C13C34-C41C14C33, -C11C23C44+C11C24C43+C21C13C44-C21C14C43-C41C13C24+C41C14C23, C11C23C34-C11C24C33-C21C13C34+C21C14C33+C31C13C24-C31C14C23]

Nous venons d'établir comment se transforme la matrice Jacobienne et son inverse avec la réduction par DIPR, sous certaines hypothèses et pour une classe des problèmes. Notons par exemple que, d'une part, on a eu à considérer que tous les points de rétroaction dans le graphe réduit sont connectés entre eux et que d'autre part, la dérivée composée sur le chemin de rétroaction, l'équivalent de $\overset{R}{C}_{ij}$ (ou par abus de notation C_{ij} du Tableau 6.1 au Tableau 6.3), est représenté selon les produits des dérivées de chacun des blocs (voir par exemple (6.48) pour $\overset{R}{C}_{55}$ du cas simple traité plus haut). Nous pouvons maintenant déduire les liens entre différentes conditions du théorème de Kantorovich. Pour rappel, le but de la section 6.2 était de formaliser les normes des matrices Jacobiennes et leurs inverses.

6.3 Analyse de l'existence et l'unicité de la solution pour le processus de Newton

6.3.1 Théorème de Kantorovich

On suppose qu'un système avec représentation complète comme en [4] converge sur l'intervalle approprié pour un processus de Newton pour lequel le choix des paramètres tels que les valeurs initiales et valeurs de perturbation (pour l'évaluation numérique de la Jacobienne) ont été adéquatement fixées. On se demande, tenant compte de ces hypothèses, si la transformation DIPR appliquée à ce cas convergeant de la représentation complète de [4], va « hériter » à son tour de ces « bonnes conditions » de convergence du processus de Newton. Sinon, c'est-à-dire si les bonnes conditions de convergence « ne sont pas transférées », quelles sont les conditions associées à l'équivalence des propriétés dans les deux modes de représentation des systèmes. Pour être plus rigoureux, on devrait étendre la même question à la problématique de l'unicité de la solution dans l'intervalle équivalent à celui de la formulation complète. Mais on se restreindra à l'analyse de la convergence. Les conditions de convergence peuvent être étendues à celles de l'unicité de la solution.

Comme on pose par hypothèse que ce processus converge, on peut y appliquer les théorèmes de Kantorovich (désigné aussi comme Newton-Kantorovich) ou leurs variantes [47][57][61][64][68][69][70]. Justement, l'unicité de solution, non traitée ici comme indiqué plus haut, peut être analysée via le théorème de Kantorovich. Voici comment s'énonce ce théorème.

Si une fonction Φ est doublement différentiable dans un domaine convexe D_0 , inclus dans un domaine D plus large sur lequel on définit la fonction Φ , c'est-à-dire avec notation mathématique usuelle : $\Phi: D \subseteq \mathbf{X} \rightarrow \mathbf{V}$. Si on suppose que, pour une certaine valeur initiale : $\mathbf{x}^{k=0} \in D_0$, et on pose aussi l'inverse de la Jacobienne $\mathbf{J}(\mathbf{x}^{k=0})^{-1}$ existe et de plus, elle est bornée comme

$$\|\mathbf{J}(\mathbf{x}^{k=0})^{-1}\| \leq \beta; \quad (6.67)$$

et que la dérivée seconde est bornée :

$$\|\mathbf{J}'(\mathbf{x})\| \leq \kappa \quad (6.68)$$

Et que, par ailleurs, les conditions suivantes sont posées :

$$\left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \boldsymbol{\Phi}(\mathbf{x}^{k=0}) \right\| \leq \eta, \quad (6.69)$$

$$\kappa\beta\eta \leq \frac{1}{2}, \quad (6.70)$$

Alors :

$$\text{i) les itérations : } \mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{J}(\mathbf{x}^k)^{-1} \boldsymbol{\Phi}(\mathbf{x}^k), \quad k=0,1,2,\dots$$

sont bien définies, les différentes valeurs $\mathbf{x}_k \in \lambda$, à l'intérieur de :

$$\bar{\lambda} = \bar{\lambda} \left(\mathbf{x}^{k=0}, \frac{2\eta}{1 + \sqrt{1 - 2\eta\beta\kappa}} \right), \quad (6.71)$$

et $k \geq 0$ et on a la séquence des $\{\mathbf{x}_k\}$ qui convergent vers la solution $\hat{\mathbf{x}} \in \bar{\lambda}$;

ii) Et aussi que la solution est unique dans :

$$\tilde{\lambda} = \begin{cases} \lambda = \lambda \left(\mathbf{x}^{k=0}, 1 + \sqrt{1 - 2\eta\beta\kappa} / \beta\kappa \right) \cap D_0 & \text{si } 2\eta\beta\kappa < 1 \\ \bar{\lambda} = \bar{\lambda} \left(\mathbf{x}^{k=0}, 1 + \sqrt{1 - 2\eta\beta\kappa} / \beta\kappa \right) \cap D_0 & \text{si } 2\eta\beta\kappa = 1 \end{cases} \quad (6.72)$$

Nous utiliserons le théorème de Kantorovich répondant aux conditions (6.67) à (6.71). Notons que la détermination des bornes de matrice Hessienne est difficile dans le cas multidimensionnel [64]. Une des variantes qui se prête le mieux à ce cas est déduit des résultats de [57] [71] [72], rapporté par [61], pages 4-3. Il s'agit d'une formulation du théorème de Kantorovich sans dérivée seconde, où les changements suivants ont été introduits :

- La condition (6.68) sur la dérivée seconde devient :

$$\left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} (\mathbf{J}(\mathbf{u}) - \mathbf{J}(\mathbf{v})) \right\| \leq \kappa \|\mathbf{u} - \mathbf{v}\|, \quad \forall \mathbf{u}, \mathbf{v} \in D_0, \quad (6.73)$$

Ou mieux encore, selon l'observation récemment présentée [70][73] :

$$\left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} (\mathbf{J}(\mathbf{u}) - \mathbf{J}(\mathbf{x}^{k=0})) \right\| \leq \kappa^0 \|\mathbf{u} - \mathbf{x}^{k=0}\|, \quad \forall \mathbf{u} \in D_0, \kappa^0 \leq \kappa, \quad (6.74)$$

Notez donc l'avantage de se passer de la détermination d'une matrice Hessienne de tenseur d'ordre 3 (de dimension $N \times N \times N$) qu'exigerait (6.68).

- La condition (6.70) devient :

$$\kappa\eta \leq \frac{1}{2}, \quad (6.75)$$

- L'intervalle où les itérations successives sont confinées devient :

$$\bar{\lambda} = \bar{\lambda} \left(\mathbf{x}_1, \frac{2\eta}{1 + \sqrt{1 - 2\eta\kappa}} - \eta \right) \subseteq D_0, \quad (6.76)$$

Alors :

$$\text{iii) les itérations : } \mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{J}(\mathbf{x}_k)^{-1} \Phi(\mathbf{x}_k), \quad k=0,1,2,\dots$$

sont bien définies, les différentes valeurs $\mathbf{x}_k \in \lambda$, intérieur de (6.76) et $k \geq 0$ et les $\{\mathbf{x}_k\}$ convergent vers la solution $\hat{\mathbf{x}} \in \bar{\lambda}$, voir (6.76),

- iv) Et la solution est unique dans :

$$\tilde{\lambda} = \begin{cases} \lambda = \lambda \left(\mathbf{x}^{k=0}, 1 + \sqrt{1 - 2\eta\kappa} / \kappa \right) \cap D_0 & \text{si } 2\eta\kappa < 1 \\ \bar{\lambda} = \bar{\lambda} \left(\mathbf{x}^{k=0}, 1 + \sqrt{1 - 2\eta\kappa} / \kappa \right) \cap D_0 & \text{si } 2\eta\kappa = 1 \end{cases} \quad (6.77)$$

- Et finalement, on a le théorème de Newton–Mysovskii si une condition plus forte que (6.67), soit :

$$\|\mathbf{J}(\mathbf{x})^{-1}\| \leq \beta, \quad (6.78)$$

est respectée, alors la condition (6.70) pour la convergence est :

$$\kappa\eta \leq 2, \quad (6.79)$$

Nous pouvons maintenant appliquer le théorème de Kantorovich ou une de ses variantes relaxées comme celle respectant les conditions (6.73) à (6.76). En plus de la condition (6.69), on fait les hypothèses que Φ est différentiable et de l'existence de l'inverse de la Jacobienne $\mathbf{J}(\mathbf{x}_0)^{-1}$.

L'implication de ce dernier sur la transformation complète vers celle réduite DIPR est présentée dans la section suivante.

6.3.2 Application du théorème de Kantorovich : approches classique et DIPR

6.3.2.1 Formulation du théorème de Kantorovich pour le système de contrôle

Nous arrivons presque au but poursuivi. Pour rappel, il s'agit de poser les bonnes conditions de convergence pour le cas complet et ensuite d'analyser comment ces conditions de convergence s'altèrent (ou s'améliorent) avec la transformation DIPR.

En récapitulant, voici les conditions découlant du théorème de Kantorovich introduit plus haut et l'intervalle de convergence des itérations $\bar{\lambda}$ (la même récapitulation peut aussi être trouvée dans [74] en considérant, pour la dernière relation, l'analyse de [72]):

$$\left. \begin{aligned} & \|\mathbf{J}(\mathbf{x}^{k=0})^{-1} \Phi(\mathbf{x}^{k=0})\| \leq \eta \\ & \|\mathbf{J}(\mathbf{x}^{k=0})^{-1} (\mathbf{J}(\mathbf{u}) - \mathbf{J}(\mathbf{v}))\| \leq \kappa \|\mathbf{u} - \mathbf{v}\|, \quad \forall \mathbf{u}, \mathbf{v} \in D_0 \\ & \|\mathbf{J}(\mathbf{x}^{k=0})^{-1} (\mathbf{J}(\mathbf{u}) - \mathbf{J}(\mathbf{x}^{k=0}))\| \leq \kappa^0 \|\mathbf{u} - \mathbf{x}^{k=0}\|, \quad \forall \mathbf{u} \in D_0, \kappa^0 \leq \kappa \\ & \kappa \eta \leq \frac{1}{2} \\ & \bar{\lambda} = \bar{\lambda} \left(\mathbf{x}_1, \frac{2\eta}{1 + \sqrt{1 - 2\eta\kappa}} - \eta \right) \subseteq D_0 \end{aligned} \right\}, \quad (6.80)$$

Le premier aspect du travail est de caractériser la Jacobienne dans la forme (6.10) où les éléments sont les dérivées partielles des fonctions des blocs et puis son inverse, exprimée dans (6.31).

Supposons que le système complet $\mathbf{F}(\mathbf{x}) = 0$ converge en respectant (6.80). On peut alors déduire que les normes de \mathbf{J}^{-1} et $\mathbf{J}^{-1}\Phi$ et leurs bornes :

$$\|\mathbf{J}^{-1}\| = \left\| \frac{1}{\det(\mathbf{gr}(\mathbf{N}))} \begin{pmatrix} \Delta_{11} & \cdot & \Delta_{1i} & \cdot & \Delta_{1N} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \Delta_{i1} & \cdot & \Delta_{ii} & \cdot & \Delta_{iN} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \Delta_{N1} & \cdot & \Delta_{Ni} & \cdot & \Delta_{NN} \end{pmatrix} \right\| = \frac{1}{|\det(\mathbf{gr}(\mathbf{N}))|} \max_{i=1 \dots N} \left(\sum_{j=1}^N |\Delta_{ij}| \right) \leq \beta, \quad (6.81)$$

où le déterminant du graphe $\det(\mathbf{gr}(\mathbf{N}))$ du système complet à N blocs ainsi que les cofacteurs Δ_{ij} ont été définis dans la section précédente :

$$\begin{aligned}\det(\mathbf{gr}(\mathbf{N})) &= \left((-1)^N\right)^{\text{per}(\text{con}(\mathbf{grfa}(\mathbf{N})))} \sum_{j=1} (-1)^{B\mathbf{grfa}\mathbf{N}} \prod(\mathbf{grfa}(\mathbf{N})) \\ \Delta_{ii} &= \left((-1)^{N-1}\right)^{\text{per}(\text{con}(\mathbf{grfa}(\mathbf{N}-\{i\})))} \sum_{j=1} (-1)^{B\mathbf{grfa}(\mathbf{N}-\{i\})} \prod(\mathbf{grfa}(\mathbf{N}-\{i\})) \quad , \\ \Delta_{ij} &= \left((-1)^{N-1}\right)^{\text{per}(\text{con}(\mathbf{grfaX}(\mathbf{N},\{i,j\})))} \sum_{j=1} (-1)^{B\mathbf{grfaX}(\mathbf{N},\{i,j\})} \prod(\mathbf{grfaX}(\mathbf{N},\{i,j\}))\end{aligned}$$

où le permanent $\text{per}()$, la connectivité $\text{con}()$, les sous graphes 1-facteur $\mathbf{grfa}(\mathbf{N}-\{i\})$ du graphe de N nœuds en éliminant le i -ième nœud et les sous-graphes connexion 1-facteur $\mathbf{grfaX}(\mathbf{N},\{i,j\})$ dans $\mathbf{gr}(\mathbf{N})$ pour deux nœuds $\{i,j\}$, ont été empruntés de [32] et expliqués plus haut. Le terme $B\mathbf{gr}_-$ est le nombre des boucles dans le graphe \mathbf{gr}_- .

Et puis :

$$\left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \Phi(\mathbf{x}^{k=0}) \right\| = \frac{1}{|\det(\mathbf{gr}(\mathbf{N}))|} \max_{i=1 \dots N} \left(\sum_{j=1}^N |\Delta_{ij} \Phi_j| \right)^{k=0} \leq \eta \quad (6.82)$$

En rappelant que dans la mise à jour des itérations de Newton on a, pour chacune de variable i :

$$\left| \delta x_i^{k=1} \right| = \left| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \Phi(\mathbf{x}^{k=0}) \right|_i = \frac{1}{|\det(\mathbf{gr}(\mathbf{N}))|} \sum_{j=1}^N |\Delta_{ij} \Phi_j|, \text{ on remplace (6.82) par :}$$

$$\left\| \mathbf{J}(\mathbf{x}_0)^{-1} \Phi(\mathbf{x}_0) \right\| = \max_{i=1 \dots N} (|\delta x_i|) \leq \eta \quad (6.83)$$

Pour établir la borne pour la norme dans (6.73), remarquons que :

$$\left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} (\mathbf{J}(\mathbf{u}) - \mathbf{J}(\mathbf{v})) \right\| \leq \kappa \|\mathbf{u} - \mathbf{v}\| \Rightarrow \left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{u}) - \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{v}) \right\| \leq \kappa \|\mathbf{u} - \mathbf{v}\|,$$

et en vertu de la relation triangulaire sur les normes :

$$\left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{u}) \right\| - \left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{v}) \right\| \leq \left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{u}) - \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{v}) \right\| \leq \kappa \|\mathbf{u} - \mathbf{v}\|.$$

Puis qu'en vertu des propriétés de la valeur absolue, le terme à l'extrême gauche et celui à l'extrême droite sont liés comme :

$$-\kappa \|\mathbf{u} - \mathbf{v}\| \leq \left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{u}) \right\| - \left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{v}) \right\| \leq \kappa \|\mathbf{u} - \mathbf{v}\|.$$

En considérant la deuxième inégalité on a :

$$\left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{u}) \right\| - \left\| \mathbf{J}(\mathbf{x}^{k=0})^{-1} \mathbf{J}(\mathbf{v}) \right\| \leq \kappa \|\mathbf{u} - \mathbf{v}\|. \quad (6.84)$$

Finalement les deux autres relations du théorème de Kantorovich sont :

$$\frac{\kappa}{\left| \det(\mathbf{gr}(\mathbf{N})) \right|} \max_{i=1 \dots N} \left(\sum_{j=1}^N |\Delta_{ij} \Phi_j| \right)^{k=0} \leq \kappa \eta \leq \frac{1}{2} \quad (6.85)$$

$$\bar{\lambda} = \bar{\lambda} \left(\mathbf{x}_0 + \left(\frac{1}{\left| \det(\mathbf{gr}(\mathbf{N})) \right|} \max_{i=1 \dots N} \left(\sum_{j=1}^N |\Delta_{ij} \Phi_j| \right)^{k=0} \right), \frac{2\eta}{1 + \sqrt{1 - 2\eta\kappa}} - \eta \right) \subseteq D_0 \quad (6.86)$$

Les relations (6.81) à (6.86) posées, on doit maintenant faire le lien avec les relations similaires dans le système transformé par la DIPR.

6.3.2.2 Illustration du théorème de Kantorovich sur un système de contrôle simple

On illustre en appliquant sur les systèmes de plusieurs blocs dans une seule boucle comme en la Figure 6-1. Il a été largement analysé dans les paragraphes antérieurs. Son schéma est repris ci-dessous. La seule boucle comprend 4 blocs que nous considérons avec une dérivée seconde constante, de sorte que l'inégalité de type (6.68) est satisfaite. Nous apposons, à côté de la représentation complète, l'équivalent de la représentation réduite par la transformation DIPR :

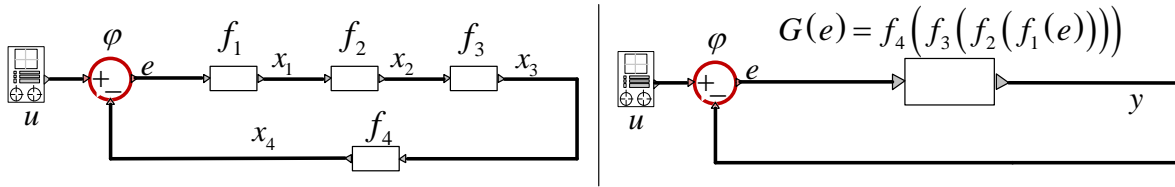


Figure 6-21: Système simple dans deux modes représentation sur lequel sera illustré le théorème de Kantorovich

Les Jacobiennes \mathbf{J} et leurs inverses \mathbf{J}^{-1} et les produits $\mathbf{J}^{-1}\Phi$ sont présentés dans le Tableau 6.4 et dans le Tableau 6.5 respectivement, avec les normes associées.

Tableau 6.4: Comparaison de la Jacobienne et de son inverse dans la représentation complète et celle réduite par la DIPR pour analyse du théorème de Kantorovich.

	Représentation complète	Transformation DIPR
1	$\mathbf{J} = \begin{pmatrix} 1 & 0 & 0 & 0 & C_1 \\ C_2 & 1 & 0 & 0 & 0 \\ 0 & C_3 & 1 & 0 & 0 \\ 0 & 0 & C_4 & 1 & 0 \\ 0 & 0 & 0 & -C_5 & 1 \end{pmatrix}$	$\mathbf{J}^R = 1 + C_5 C_4 C_3 C_2 C_1$
2	$\mathbf{J}^{-1} = \frac{1}{1 + C_5 C_4 C_3 C_2 C_1} \times$ $\begin{pmatrix} 1 & -C_5 C_3 C_4 C_1 & -C_4 C_5 C_1 & -C_1 C_5 & C_1 \\ C_2 & 1 & -C_2 C_5 C_1 C_4 & -C_2 C_5 C_1 & C_2 C_1 \\ C_3 C_2 & C_3 & 1 & -C_3 C_2 C_5 C_1 & C_3 C_2 C_1 \\ C_4 C_3 C_2 & C_4 C_3 & C_4 & 1 & C_4 C_3 C_2 C_1 \\ -C_5 C_2 C_3 C_4 & -C_5 C_3 C_4 & -C_5 C_4 & -C_5 & 1 \end{pmatrix}$ <p>où C_v est la dérivée de la v-ième fonction de bloc f_v et les $\prod C_v$ sont les éléments de \mathbf{J}^{-1}_{ij} qui sont les cofacteurs Δ_{ij} dans (6.81)</p>	$\mathbf{J}^{R^{-1}} = \frac{1}{1 + C_5 C_4 C_3 C_2 C_1}$
3	$\ \mathbf{J}\ = \max_{i=1 \dots N} \left(\sum_{j=1}^N \mathbf{J}_{ij} \right)$ $= \max(1 + C_1 , 1 + C_2 , 1 + C_3 , 1 + C_4 , 1 + C_5)$	$\ \mathbf{J}\ ^R = \max_{i=1 \dots M} \left(\sum_{j=1}^M \mathbf{J}_{ij}^R \right)$ $= 1 + C_1 C_2 C_3 C_4 C_5 $
4	$\ \mathbf{J}^{-1}\ = \frac{1}{ 1 + C_5 C_4 C_3 C_2 C_1 } Q_{\max}$ $Q_{\max} = \max \begin{pmatrix} 1 + C_2 + C_3 C_2 + C_4 C_3 C_2 + -C_5 C_2 C_3 C_4 , \\ 1 + C_3 + C_4 C_3 + -C_5 C_3 C_4 + -C_5 C_3 C_4 C_1 , \\ 1 + C_4 + -C_5 C_4 + -C_4 C_5 C_1 + -C_2 C_5 C_1 C_4 , \\ 1 + -C_5 + -C_1 C_5 + -C_2 C_5 C_1 + -C_3 C_2 C_5 C_1 , \\ 1 + C_1 + C_2 C_1 + C_3 C_2 C_1 + C_4 C_3 C_2 C_1 \end{pmatrix}$	$\ \mathbf{J}^{R^{-1}}\ $ $= \frac{1}{ 1 + C_5 C_4 C_3 C_2 C_1 } \quad \text{a}$ $\leq \frac{\ \mathbf{J}^{-1}\ }{Q_{\max}} \leq \ \mathbf{J}^{-1}\ $ <p>vec $Q_{\max} \geq 1$</p>

Tableau 6.5: Comparaison de $\|\mathbf{J}^{-1}\Phi\|$ de la représentation complète et celle réduite par la DIPR pour l'analyse du théorème de Kantorovich.

	Représentation complète	Transformation DIPR
1	$\ \mathbf{J}^{-1}\Phi\ = \frac{1}{ 1 + C_5 C_4 C_3 C_2 C_1 } T \max$ $T \max = \left(\begin{array}{l} \Phi_1 + C_2 \Phi_2 + C_3 C_2 \Phi_3 + C_4 C_3 C_2 \Phi_4 + -C_5 C_2 C_3 C_4 \Phi_5 \\ \Phi_2 + C_3 \Phi_3 + C_4 C_3 \Phi_4 + -C_5 C_3 C_4 \Phi_5 + -C_5 C_3 C_4 C_1 \Phi_1 \\ \Phi_3 + C_4 \Phi_4 + -C_5 C_4 \Phi_5 + -C_4 C_5 C_1 \Phi_1 + -C_2 C_5 C_1 C_4 \Phi_2 \\ \Phi_4 + -C_5 \Phi_5 + -C_1 C_5 \Phi_1 + -C_2 C_5 C_1 \Phi_2 + -C_3 C_2 C_5 C_1 \Phi_3 \\ \Phi_5 + C_1 \Phi_1 + C_2 C_1 \Phi_2 + C_3 C_2 C_1 \Phi_3 + C_4 C_3 C_2 C_1 \Phi_4 \end{array} \right)$ <p>où la fonction vecteur est $\Phi = [\Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5]^{\text{Transposé}}$. Pour rappel : $\Phi_i = x_i - f_i(\mathbf{x})$. Notez que l'ordonnancement des termes et des facteurs obéit à une permutation circulaire et de toutes les combinaisons possibles. On sait aussi que le signe dépend de nombre des boucles, c'est-à-dire des blocs n'apparaissant pas dans le terme $\prod C_v$.</p>	$\ \mathbf{J}^{-1} \Phi\ = \frac{1}{ 1 + C_5 C_4 C_3 C_2 C_1 } \Phi$ $\leq \frac{\ \mathbf{J}^{-1}\Phi\ }{T \max} \leq \ \mathbf{J}^{-1}\Phi\ $ <p>car $\frac{\Phi}{T \max} \leq 1$, où $T \max$ est calculé comme pour la représentation complète ci-à-gauche.</p>

On peut établir alors les conditions du théorème pour les deux cas et comparer le comportement par rapport aux conditions initiales. Notez que \mathbf{J} , \mathbf{J}^{-1} et $\mathbf{J}^{-1}\Phi$ sont dans la forme « topologique » générique pour une seule boucle. Les éléments C_v peuvent être constants ou pas, c'est-à-dire que le terme C_v pour le v -ième bloc peut comprendre aussi bien des valeurs dépendant du point de linéarisation c'est-à-dire du pas ou de l'itération précédents. De sorte, qu'en général, on aura :

$$C_v = \frac{\partial}{\partial x_{\text{entrées}}} f_v = l(x_{\text{entrées}}) \times \text{Gain}, \quad (6.87)$$

où $l(.)$ est une fonction qui donne la mesure de la linéarité de la fonction du bloc

Nous choisissons un cas simple pour mieux illustrer l'analyse. On suppose que les dérivées sont constantes, c'est-à-dire des blocs sont à fonction linéaire. On a donc $l(x_{\text{entrées}}) = 1$ dans (6.87).

Les différentes conditions du théorème de Kantorovich sont représentées dans le Tableau 6.6.

Tableau 6.6: Les conditions du théorème de Kantorovich pour le cas simple avec blocs linéaires

	Représentation complète	Transformation DIPR
1	Étant donné que les dérivées premières sont constantes, la borne des dérivées secondes tel que : $\ \mathbf{J}'(\mathbf{x})\ \leq \kappa$	Pour la même raison que celle à gauche : $\ \mathbf{J}'^R(\mathbf{x})\ \leq \kappa$
2	$\ \mathbf{J}(\mathbf{x}^{k=0})^{-1}\ \leq \beta$	$\ \mathbf{J}^R(\mathbf{x}^{k=0})^{-1}\ \leq \frac{\beta}{Q_{\max}}$
3	$\ \mathbf{J}(\mathbf{x}^{k=0})^{-1} \Phi(\mathbf{x}^{k=0})\ \leq \eta$	$\ \mathbf{J}^R(\mathbf{x}^{k=0})^{-1} \Phi^R(\mathbf{x}^{k=0})\ \leq \frac{\eta}{T_{\max}} \Phi^R \leq \eta$ car $T_{\max} \geq \Phi^R$. En effet on a $T_{\max} = \sum \left(\Phi_i + \sum_{\substack{j=1 \dots N \\ j \neq i}} \prod C_v \Phi_j \right) \geq \Phi_i$, et on a aussi $\Phi^R(\mathbf{x}^{k=0}) = \Phi_5(\mathbf{x}^{k=0})$ pour les mêmes conditions initiales
4	$\kappa \beta \eta \leq \frac{1}{2}$	$\kappa \frac{\beta}{Q_{\max}} \frac{\eta}{T_{\max}} \leq \frac{1}{2}$
5	$\bar{\lambda} = \bar{\lambda} \left(\mathbf{x}^{k=0}, \frac{2\eta}{1 + \sqrt{1 - 2\eta\beta\kappa}} \right)$	$\bar{\lambda} = \bar{\lambda} \left(e^{k=0}, \frac{2 \frac{\eta}{T_{\max}}}{1 + \sqrt{1 - 2 \frac{\eta\beta}{T_{\max} Q_{\max}} \kappa}} \right)$ $\bar{\lambda} = \bar{\lambda} \left(e^{k=0}, \frac{2 T_{\max} (Q_{\max})^2 \eta}{(T_{\max} Q_{\max})^2 + \sqrt{T_{\max} Q_{\max} - 2\eta\beta\kappa}} \right)$
6	$\bar{\lambda} = \bar{\lambda}(\mathbf{x}^{k=0}, 2\eta), \kappa = 0$	$\bar{\lambda} = \bar{\lambda} \left(e^{k=0}, 2 \frac{\eta}{T_{\max}} \right), \kappa = 0$

Illustrons ce résultat sur une application numérique. Soit les gains du premier au quatrième bloc dans la Figure 6-21 dans l'ordre $[C_v] = [0.1, 0.2, 0.15, C_{Max}]$, où $C_{Max} = [0.75, 25, 50]$ pour avoir différentes valeurs des normes.

Tableau 6.7: Application numérique sur les conditions du théorème de Kantorovich pour le cas simple avec blocs linéaires dont les gains sont : $[C_v] = [0.1, 0.2, 0.15, C_{Max}]$.

C_{Max}	0.75	25	50
$\ J(\mathbf{x}^{k=0})^{-1}\ $	2.5842	50.2326	93.0435
β vérifiant $\ J(\mathbf{x}^{k=0})^{-1}\ \leq \beta$	2.5842+EPS	50.2326+EPS	93.0435+EPS
Q_{max}	2.5900	54.0000	107.0000
$\frac{\beta}{Q_{max}}$	0.9978	0.9302	0.8696
$\ J^R(\mathbf{x}^{k=0})^{-1}\ \leq \frac{\beta}{Q_{max}}$	0.9978	0.9302	0.8696
$\ J(\mathbf{x}^{k=0})^{-1} \Phi(\mathbf{x}^{k=0})\ $	1.1227 u	1.1144 u	1.1070 u
η avec $\ J(\mathbf{x}^{k=0})^{-1} \Phi(\mathbf{x}^{k=0})\ \leq \eta$	1.1227 u +EPS	1.1144 u +EPS	1.1070 u +EPS
T_{max}	1.1253	1.1980	1.2730
$\ J^R(\mathbf{x}^{k=0})^{-1} \Phi^R(\mathbf{x}^{k=0})\ $	0.9978 u	0.9302 u	0.8696 u
$\frac{\eta}{T_{max}}$ vérifiant $\ J^R(\mathbf{x}^{k=0})^{-1} \Phi^R(\mathbf{x}^{k=0})\ \leq \frac{\eta}{T_{max}}$	0.9978 u +EPS	0.9302 u +EPS	0.8696 u +EPS
Complète : $\bar{\lambda} = \bar{\lambda}(\mathbf{x}^{k=0}, 2\eta), \kappa = 0$	2.2454 u +EPS	2.2288 u +EPS	2.2140 u +EPS
DIPR : $\bar{\lambda} = \bar{\lambda}\left(e^{k=0}, 2\frac{\eta}{T_{max}}\right), \kappa = 0$	1.9956 u +EPS	1.8604 u +EPS	1.7392 u +EPS

La solution pour les valeurs des entrées u de -20 à 20 et avec les conditions $\mathbf{x}^{k=0} = 0$ est comparée à la borne supérieure de l'intervalle $\bar{\lambda} = \bar{\lambda}(\mathbf{x}^{k=0} = 0, 2\eta), \kappa = 0$ et la borne inférieure correspondante. Les résultats sont dans la Figure 6-22.

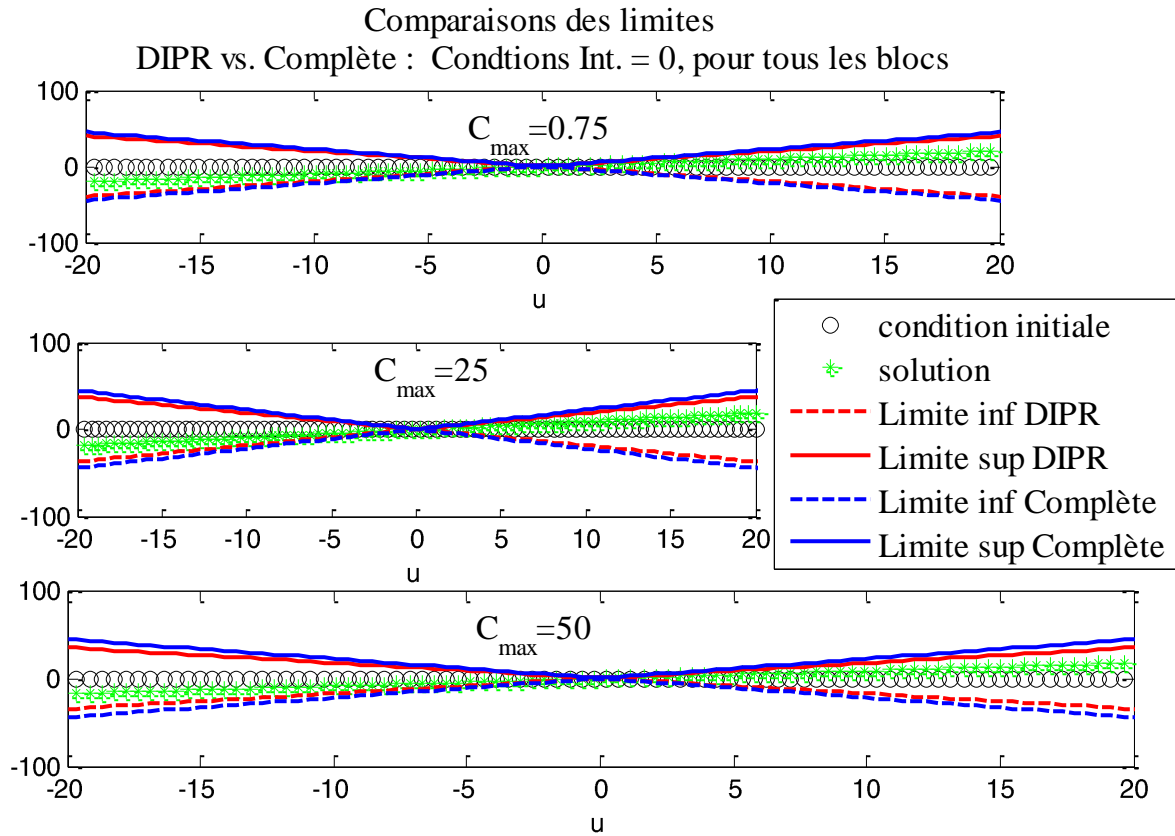


Figure 6-22: Résultat du théorème de Kantorovich sur un système simple avec CI=0 pour toutes les variables

De la même manière, avec les conditions $\mathbf{x}^{k=0} = 2$, les bornes supérieures de l'intervalle $\bar{\lambda} = \bar{\lambda}(\mathbf{x}^{k=0} = 1, 2\eta), \kappa = 0$, sont dans la Figure 6-23

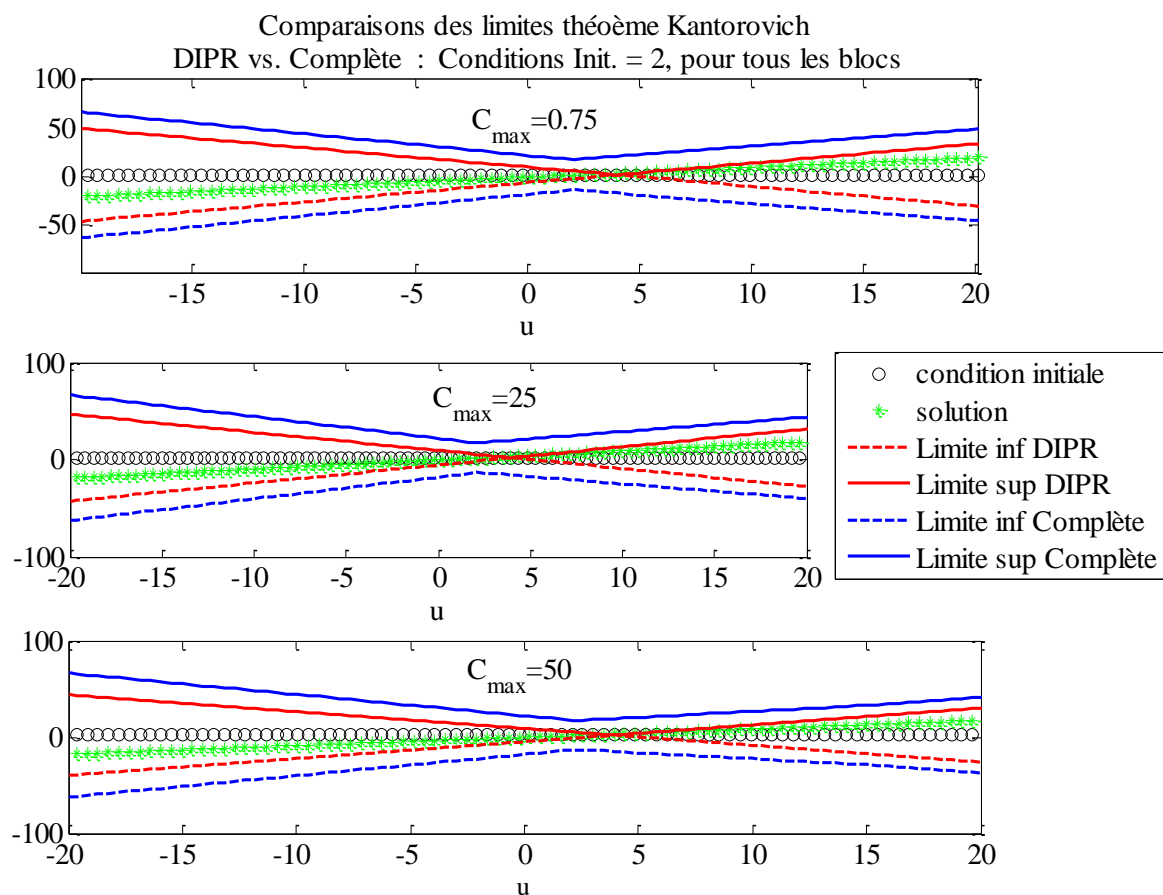


Figure 6-23: Résultat du théorème de Kantorovich sur un système simple avec CI=2 pour toutes les variables

Dans la Figure 6-22 et dans la Figure 6-23, on peut voir comment changent les bornes du théorème de Kantorovich prédites dans le Tableau 6.6 et dans le Tableau 6.7, où EPS est un incrément infinitésimal et u est la variable d'entrée. En fonction de l'entrée u , d'une part, les bornes sont différentes et, d'autre part, la solution change avec chaque entrée. Les valeurs de cette dernière ont varié de -20 à 20. On note que pour la DIPR, la zone en rouge est incluse dans celle de la méthode complète en bleue. Il est évident que certaines conditions qui seraient incluses dans la zone en bleue de la méthode complète sortiraient de la zone rouge pour la méthode réduite par la DIPR. Cependant les avantages à avoir une zone réduite pourraient aussi être étudiés.

Finalement, en comparant les deux cas illustrés, on voit qu'entre les valeurs de conditions initiales de 0 et de 2 les tailles de zone de convergence changent. Pour CI=0, les deux méthodes sont presque identiques.

Sans refaire toute l'analyse pour un cas complexe, par exemple non-linéaire, on peut déceler que les conditions initiales vont avoir un impact différent sur la performance. On établit seulement les expressions des bornes de la zone de convergence comme dans Tableau 6.6.

Supposons que le quatrième bloc f_4 soit non-linéaire. Précisément on pose que :

$$C_4 = \frac{\partial}{\partial x_3} f_4 = l(x_3) \times \underline{C}_4, \text{ où la fonction } l(.) \text{ mesurant la linéarité du bloc est } l(x_3) = 2x_3.$$

On a donc :

$$f_4 = \underline{C}_4 x_3^2 \tag{6.88}$$

Les conditions du théorème de Kantorovich changent par rapport au cas linéaire. Le résumé récapitulatif est dans le Tableau 6.8.

Tableau 6.8: Les conditions du théorème de Kantorovich pour le cas simple avec un bloc non-linéaire

	Représentation complète	Transformation DIPR
1	$\ \mathbf{J}'(\mathbf{x})\ \leq 2, \kappa = 2$	$\ \mathbf{J}'(\mathbf{x})\ \leq \kappa, \kappa = 2$
2	$\ \mathbf{J}(\mathbf{x}^{k=0})^{-1}\ \leq \beta$	$\ \mathbf{J}(\mathbf{x}^{k=0})^{-1}\ \leq \frac{\beta}{Q_{\max}}$
3	$\ \mathbf{J}(\mathbf{x}^{k=0})^{-1} \Phi(\mathbf{x}^{k=0})\ \leq \eta$	$\ \mathbf{J}(\mathbf{x}^{k=0})^{-1} \Phi(\mathbf{x}^{k=0})\ \leq \frac{\eta}{T_{\max}} \Phi \leq \eta \text{ car } T_{\max} \geq \Phi.$ <p>En effet on a $T_{\max} = \sum \left(\Phi_i + \sum_{\substack{j=1 \dots N \\ j \neq i}} \prod C_v \Phi_j \right) \geq \Phi_i$</p> <p>Et on a aussi $\Phi(\mathbf{x}^{k=0}) = \Phi_s(\mathbf{x}^{k=0})$ pour les mêmes conditions initiales</p>
4	$\beta\eta \leq \frac{1}{4}$	$\frac{\beta}{Q_{\max}} \frac{\eta}{T_{\max}} \leq \frac{1}{4}$
5	$\bar{\lambda} = \bar{\lambda} \left(\mathbf{x}^{k=0}, \frac{2\eta}{1 + \sqrt{1 - 4\eta\beta}} \right)$	$\bar{\lambda} = \bar{\lambda} \left(e^{k=0}, \frac{2 \frac{\eta}{T_{\max}}}{1 + \sqrt{1 - 4 \frac{\eta\beta}{T_{\max} Q_{\max}}}} \right)$ $\bar{\lambda} = \bar{\lambda} \left(e^{k=0}, \frac{2 T_{\max} (Q_{\max})^2 \eta}{(T_{\max} Q_{\max})^2 + \sqrt{T_{\max} Q_{\max} - 4\eta\beta}} \right)$

Avec une certaine nonlinearité dans la boucle de rétroaction les zones de convergence prédites par le théorème de Kantorovich sont différentes pour les deux méthodes (voir le Tableau 6.8). Ceci suggère donc la nécessité des travaux futurs pour approfondir cet aspect de la méthode proposée.

6.4 Conclusion

Le but recherché dans ce dernier chapitre a consisté à analyser la manière dont la transformation DIPR modifie les propriétés des itérations de Newton, en supposant que le système de contrôle représenté sous la forme complète soit résolu avec succès avec un choix des valeurs initiales convenables. Le théorème de Newton-Kantorovich, utilisant les limites sur les normes de la Jacobienne, a été appliqué. Les expressions des normes de la Jacobienne, introduites

brièvement au début du chapitre, ont été établies pour un cas générique, sous forme analytique. Il convient de mentionner que les équivalents numériques de ces normes peuvent aisément être appliqués. La représentation de graphe de Coates de la Jacobienne et sa réduction aux seuls nœuds PR a fait ressortir les changements que la transformation DIPR induits aux normes de la Jacobienne et aux autres expressions sur lesquelles la Jacobienne intervient. Il ressort de l'analyse d'un cas simple, que, pour les mêmes conditions initiales, les comportements du théorème de Newton-Kantorovich diffèrent entre la représentation classique complète et la représentation DIPR. Ces changements peuvent par ailleurs être plus prononcés dans le cas non-linéaire. Des travaux futurs seraient utiles à l'approfondissement de cet aspect et, éventuellement, permettraient à déduire des approches nouvelles d'initialisation des équations de contrôle.

CONCLUSION

Cette thèse a visé à améliorer la solution des systèmes de contrôle. Les méthodes proposées ont aidé à réduire le temps de simulation du réseau avec ses systèmes de contrôle, tout en maintenant la précision requise pour l'analyse de phénomènes transitoires électromagnétiques.

Après un aperçu général de la modélisation mathématique des systèmes de contrôle, une catégorisation selon le critère topologique fait ressortir un des regroupements de blocs de contrôle parmi les plus difficiles à résoudre : il s'agit du groupe à rétroaction, lequel représente le système comme un graphe orienté.

L'hypothèse principale considérée a consisté en ceci : une reformulation avec moins d'équations et, donc, une taille réduite de la Jacobienne, est possible en considérant seulement les boucles de rétroaction ; de plus, elle permet de meilleures performances que la représentation complète classique. Les outils de la théorie de graphe ont été utilisés pour établir, entre autres, l'ordre topologique des blocs et différentes caractéristiques de la connectivité. Pour l'ordre topologique, la méthode de parcours du graphe en profondeur (« Depth-First-Search ») a été utilisée. Pour la connectivité, la matrice d'adjacence, d'accessibilité et les composantes fortement connectées permettent d'analyser les cycles dans le graphe. Une technique de détection des cycles et une méthode de suppression de ces derniers ont été établies. Pour supprimer tout cycle dans le graphe, une condition s'applique, soit celle d'avoir toutes les valeurs propres nulles pour la matrice d'adjacence. La transformation du graphe de contrôle sans cycle a permis de formuler une représentation des équations aux éléments de rétroaction. Ceci a rendu possible la décomposition avec interconnexion aux points de rétroaction (DIPR) : il s'agit de la contribution principale de cette thèse, la nouvelle approche qu'elle propose.

Des méthodes de résolution des systèmes d'équations (non-linéaires) de contrôle, qui s'appliquent dans le cadre général, ont été déployées pour la nouvelle approche DIPR. Quatre variantes de la nouvelle approche ont été illustrées. La première est la méthode de Newton de base, dans laquelle la mise à jour entre les itérations se fait selon la solution linéarisée à l'itération précédente. La deuxième variante, plus robuste que la méthode de Newton de base, permet la mise à jour des itérations selon la méthode dite de Dogleg. Elle ajuste les incréments pour garantir que les itérations restent dans une zone de confiance. La troisième variante est une méthode point-fixe. Elle est possible car la DIPR introduite permet une formulation des chemins

de rétroaction tels qu'ils s'apparentent à un problème à point-fixe avec, comme variables, les seuls points de rétroaction. De plus, les propriétés contractantes mesurant la qualité des itérations à point-fixe peuvent être analysées pour les chemins de rétroaction établis via la DIPR. Cet aspect est aussi une contribution originale de cette thèse. Finalement, une quatrième variante repose dans une solution non-itérative. Contrairement aux trois premières, aucun critère de convergence n'est établi pour cette variante non-itérative. Sa précision et, donc, son application sont restreintes à certaines classes de système de contrôle. L'avantage qu'elle offre, soit celui de ne pas recalculer, ni la Jacobienne, ni la fonction objective, est à prendre en compte dans le cas où le contexte le permet.

Pour analyser la performance de la nouvelle approche, les variantes ci-haut ont été appliquées sur des systèmes de contrôle représentatifs de cas couramment rencontrés. À cette fin, trois cas ont été étudiés. Le premier cas est le PLL ; le deuxième cas est une machine asynchrone modélisée par un système d'équations de dimension 87×87 ; et finalement, le troisième cas est un réseau de très grande taille où 12 centrales éoliennes ont été intégrées au réseau test IEEE-39 barres. Dans tous les cas traités, les résultats avec la DIPR sont similaires à ceux obtenus par la méthode de référence considérée, implémentée dans EMTP. Les gains en temps de calcul ont été enregistrés dans la plupart des cas pour une des variantes de la méthode DIPR proposé : ils atteignent parfois 2. Les limitations de certaines variantes ont toutefois dû être analysées. Par exemple, la performance de la nouvelle approche diminue quand le ratio du nombre des blocs PR, sur tous les autres blocs augmente. Ceci a été analysé en détail sur la machine. La méthode DIPR est potentiellement plus avantageuse quand ce ratio est le plus bas car la taille de la Jacobienne est plus petite.

Pour pallier à la contrainte consistant à former ces Jacobiennes de grande taille, à chaque pas de simulation, la méthode point-fixe a été combinée à celle de Dogleg itératif. Ainsi, avec un même critère de convergence pour les deux méthodes, la précision n'est pas compromise. Les avantages conférés par la rapidité, pour la première, et par la robustesse, pour la seconde, sont exploités. Cette méthode hybride a été appliquée pour la solution des centrales éoliennes dans le réseau IEEE-39 barres. Son gain en temps de simulation a été de près de 75%, dans certains cas. Malgré des cas limites, à souligner que l'approche DIPR offre plusieurs variantes alternatives aux méthodes classiques de représentation complète de systèmes.

Pour une analyse approfondie de la nouvelle approche et pour ouvrir des pistes de recherche futures, la méthode DIPR a été comparée à d'autres méthodes de solution de système de contrôle. Deux comparaisons ont été faites. Une première a montré que, sur le pur plan des représentations des systèmes d'équations simultanées, la méthode de substitution de variables peut transformer une représentation complète classique en une représentation DIPR, à condition de ne pas substituer les variables PR et de suivre un ordre de substitution respectant la séquence topologique d'évaluation des blocs, telle que fournie par une méthode de type DFS. Une deuxième comparaison est celle entre l'approche DIPR et la représentation par variables d'état. Le logiciel MATLAB/Simulink, lequel utilise une représentation par variables d'état, a été comparé à l'approche DIPR pour deux cas, soit celui où les variables d'état équivalentes aux sorties des blocs PR et celui où il y a des variables d'état qui ne soient pas des variables aux PR. Les analyses ont été illustrées sur le PLL. Il ressort de l'analyse que l'étape d'évaluation des sorties de blocs dans l'approche avec représentation par variables d'état s'apparente à la solution DIPR pour la solution de la boucle algébrique, les contraintes algébriques étant comparables aux boucles de rétroaction. Une différence peut cependant surgir quand le nombre des contraintes (et, donc, la taille de la Jacobienne) du cas par variables d'état n'est pas le nombre optimal de coupures (c'est-à-dire, le nombre de PR), pour la méthode DIPR. Pour le PLL, par exemple, MATLAB/Simulink a généré 3 contraintes alors que la DIPR en a généré 2. La méthode DIPR pourrait avoir certains avantages. La solution finale, cependant, est identique car, entre autres, on a établi que le nombre de coupures peut être supérieur au nombre minimal : dans les deux cas, cependant, il faut qu'aucun cycle ne subsiste dans le graphe représentant le système de contrôle.

Un autre axe d'analyse pouvant suscité des travaux futurs est l'initialisation. Elle est bien connue comme étant un de défis de la solution des systèmes non-linéaires, en général, et celui de contrôle, en particulier. A été analysée la manière dont la transformation DIPR modifie les propriétés des itérations de Newton, en supposant que le système de contrôle représenté sous la forme complète soit résolu avec succès, avec un choix des valeurs initiales convenables. Le théorème de Newton-Kantorovich, utilisant les limites sur les normes de la Jacobienne, a été appliqué. Les expressions des normes de la Jacobienne, ont été établies pour un cas générique, sous forme analytique. Il convient de mentionner que les équivalents numériques de ces normes peuvent aisément être appliqués. La représentation de graphe de Coates de la Jacobienne et sa réduction aux seuls nœuds PR a fait ressortir les changements que la transformation DIPR induit

aux normes de la Jacobienne et aux autres expressions sur lesquelles la Jacobienne intervient. Il ressort de l'analyse d'un cas simple que, pour les mêmes conditions initiales, les comportements du théorème de Newton-Kantorovich diffèrent selon la représentation classique complète ou la représentation DIPR. Ces changements peuvent, par ailleurs, être plus prononcés dans le cas non-linéaire. Cette thèse apporte au moins deux autres contributions : une illustration du théorème de Kantorovich sur un système de contrôle, de même qu'une représentation topologique des matrices Jacobiennes et de leurs inverses en utilisant les outils de la théorie des graphes. Des travaux futurs seraient utiles à l'approfondissement de cet aspect et, éventuellement, permettraient de déduire des approches nouvelles d'initialisation des équations de contrôle.

Nonobstant ces limitations, sans oublier les pistes d'améliorations avec les travaux futurs mentionnées plus haut incluant aussi les nouvelles fonctionnalités potentiellement à introduire, à noter les retombées en temps de simulation qu'a apportées cette thèse.

BIBLIOGRAPHIE

- [1] R. Gagnon, G. Turmel, C. Larose, J. Brochu, G. Sybille, M. Fecteau, "Large-Scale Real-Time Simulation of Wind Power Plants into Hydro-Québec Power System," Presented at the 9th International Workshop on Large-Scale Integration of Wind Power into Power Systems as well as on Transmission Networks for Offshore Wind Power Plants, 18-19 October 2010, Quebec-city, Québec, Canada
- [2] M. Stubbe, M. Jerosolimski, B. Meyer, Outils de simulation dynamique des réseaux électriques, *Revue Techniques de l'Ingénieur*, Dossier 4120, 10 nov. 1998
- [3] DCG-EMTP (Development coordination group of EMTP) Version EMTP-RV 2.1, Electromagnetic Transients Program. [Online]. Available: <http://www.emtp.com>
- [4] J. Mahseredjian, L. Dubé, Ming Zhou, S. Denetière, G. Joos, "Simultaneous solution of control system equations in EMTP," *IEEE Transactions on Power Systems*, Vol. 21, No. 1, Feb. 2006, pp. 117 – 124
- [5] L. Dubé, H.W. Dommel, "Simulation of control systems in an electromagnetic transients program with TACS," *Power Industry Computer Applications Conference*, 1977, pp. 266-271.
- [6] L. X. Bui, G. Morin, S. Casoria, J. Reeve, "EMTP TACS-FORTRAN Interface Development for Digital Control Modeling," *IEEE Transactions on Power Systems*, Vol. 7, No. 1, Feb. 1992, pp. 314-319.
- [7] R. H. Lasseter, J. Zhou, "TACS Enhancements for the Electromagnetic Transient Program," *IEEE Transactions on Power Systems*, Vol. 9, No. 2, May 1994, pp. 736-742
- [8] D. Kosterev, "Hydro Turbine-Governor Model Validation in Pacific Northwest," *IEEE Transactions on Power Systems*, Vol. 19, No. 2, May 2004, pp. 1144-490.
- [9] L. D. Bellomo, Modelling of Wind Parks for Slow and Fast Transients, Ph.D. thesis, 2010, École polytechnique de Montréal
- [10] P. J. Mosterman, J.E. Cioffi, Automated approach to resolving artificial algebraic loops, The Mathworks, US Patent no.: US 7,167,817 B2, Jan. 23, 2007
- [11] H. K. Khalil, *Nonlinear systems*, Third edition, Prentice hall, 2002.
- [12] G. Zames, "Realizability Condition for Nonlinear Feedback Systems" *IEEE Transactions on Circuits Theory*, [legacy, pre - 1988], Volume 11, Issue 2, Jun 1964 Page(s):186 – 194.
- [13] F. F. Wu, C. Desoer, "Global Inverse Function Theorem," *IEEE Transactions on circuit theory*, March 1972
- [14] A. Isidori, *Nonlinear control systems II*, London: Springer, c1999. --, ISBN 1852331887
- [15] I. W. Sandberg, "Necessary and Sufficient Conditions for the Global Invertibility of Certain Nonlinear Operators That Arise in the Analysis of Networks," *IEEE Transactions on circuit theory*, vol. ct-18, No. 2, March 1971,
- [16] C. Desoer, H. Haneda, "The measure of a matrix as a tool to analyze computer algorithms for circuit analysis," *IEEE Transactions on circuit theory*, Vol. 19, No 5, September 1972.
- [17] C. J. Harris, J. M. E. Valença, *The stability of input-output dynamical systems*, London, Angletterre: Academic Press, c1983.
- [18] J. C Willens, *The Analysis of Feedback Systems*, Cambridge, Mass. The MIT Press, c1971
- [19] C. Desoer, W. W. Chang, "The Feedback Interconnection of Multivariable Systems: Simplifying Theorems for Stability," *Proceedings of the IEEE*, Vol. 64, no 1, January 1976.

- [20] V. Fronion, S. Monaco, D. Normand-Cyrot, "Asymptotic Properties of Incrementally Stable Systems," IEEE transaction on automatic control, Vol. 41, No. 5, May 1996.
- [21] C. A. Desoer, M. Vidyasagar, Feedback systems: input-output properties, electrical sciences series, New York: Academic Press, 1975, ISBN 0122120507.
- [22] G. Zames, "Input-output feedback stability and robustness 1959-85".IEEE Control Systems Magazine, Vol. 16, No.3, June 1996, pp. 61 - 66
- [23] M. Vidyasagar, "Input-output analysis of large-scale interconnected systems : decomposition, well-posedness, and stability," Lecture notes in control and information sciences, Berlin, New York : Springer-Verlag, 1981, ISBN: 0387105018 (U.S.), 3540105018 (Germany).
- [24] P. V. Kokotovic, J. J. Allemong, J. R. Winkelman and J. H. Chow, " Singular perturbation and iterative separation of time scale" , 1980, automatica, 16, pp 23-33
- [25] M. Koga, Y. Tsutsui, and J. Yabuuchi, "Java Simulation Platform for Control System based on Block Diagram," Proceedings of the 2006 IEEE, Conference on Computer Aided Control Systems Design, Munich, Germany, October 4-6, 2006, pp. 2304-09.
- [26] Ü. Kotta, E. Pawluszewicz, S. Nõmm, "Reducibility and reduction of discrete time nonlinear control systems: comparison of two approaches," Proceedings 2003 international conference on physics and Control, 2003, St. Petersburg, Russia.
- [27] T. H. Cormen, C. E. Leiserson, Ronald L. Rivest, and Clifford Stein, Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill, 2001. ISBN 0-262-03293-7. Section 22.3: Depth-first search, pp.540–549
- [28] T. A. Davis, Direct method for sparse linear system, SIAM, Philadelphia, Sept. 2006. Part of the SIAM Book series of the fundamentals of algorithms
- [29] B. D. McKay, F. E. Oggie, G. F. Royle, N. J. A. Sloane, I. M. Wanless, H. S. Wilf, "Acyclic Digraphs and Eigenvalues of (0; 1)-Matrices," Journal of Integer Sequences, Vol. 7 (2004),
- [30] D. M. Cvetkovic, M. Doob and H. Sachs, Spectra of Graphs, third ed., Barth, Heidelberg, 1995
- [31] E. J. Henley, R.A. Williams, Graph theory in modern engineering, Academic press 1973
- [32] W. K. Chen, Applied graph theory American Elsevier ISBN 0 444 101055
- [33] J. Mahseredjian, S. Denetière, L. Dubé, B. Khodabakhchian, L. Gérin-Lajoie. "On a new approach for the simulation of transients in power systems," Proceedings of International Conference on Power Systems Transients, IPST 2005 in Montréal, 19-23 juin 2005.
- [34] B. Khodabakhchian, J. Mahseredjian, A. Xémard, Régimes transitoires dans les réseaux électriques, Revue Techniques de l'Ingénieur, Dossier 082,10 nov. 2007
- [35] H. W. Dommel, Electromagnetic Transients Program Reference Manual, (EMTP Theory Book), Portland, Oregon, 1986
- [36] IEEE working group 15.08.09 Tutorial on Modeling and analysis of system transients using digital programs, IEEE PES Special Publication, A. M. Gole, J. Martinez-Velasco, A. J. F. Keri Ed., IEEE operations center, P.O. Box 1331, 445 Hoes Lane, Piscataway, NJ, USA,1998
- [37] J. Mahseredjian, Régimes transitoires électromagnétiques: simulation, Revue Techniques de l'Ingénieur, Dossier 4130, 10 févr. 2008
- [38] M. Vidyasagar, "Nonlinear system analysis," Network series, Robert W. Newcomb, editor, prentice-hall, 1978
- [39] M. Bensoubaya, "Sur la stabilité et la stabilisation des systèmes non-linéaires discrets", Thèse de doctorat présentée à l'université de Metz, 25 juin 1997
- [40] The MathWorks Inc, Software Package for Modeling and Analyzing Dynamic Systems. Simulink, [Online]. Available: <http://www.mathworks.com>.

- [41] T. Fujisawa, E. S. Kuh, "Some Results on Existence and Uniqueness of Solutions of Nonlinear Networks," IEEE Transactions on circuit theory, vol. cr-18, No. 5, Sept. 1971
- [42] A. Ralston, P. Rabinowitz, A first course in numerical analysis, New York: McGraw-Hill, c1978.
- [43] P. Rabinowitz, Numerical methods for nonlinear algebraic equations, London; New York: Gordon and Breach Science Publishers, [1970].
- [44] R. Wait, The numerical solution of algebraic equation, Chichester Ed. [Eng.], New York: Wiley, c1979.
- [45] Y. Sawaragi, H. Nakayama, T. Tanino, Theory of multiobjective optimization, Mathematics in science and engineering; v. 176, Orlando: Academic Press, 1985.
- [46] Minpack library: <http://www.netlib.org/minpack/> consulted 10th Dec. 2012.
- [47] J. E. Dennis, On the convergence of Newton-like Methods, in Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz, ed., Gordon and Breach Science Publishers, [1970].
- [48] Y. Collette, P. Siarry, Multiobjective optimization: principles and case studies, Berlin: Springer, c2004.
- [49] J. Nocedal, S. J. Wright, Numerical Optimization, Springer series in Operation Research, New York: Springer, c2006, ISBN 0387303030.
- [50] J. J More, G. S. Burton, K. Hillstrom, User guide for MINPACK-1, Argonne National Labs Report ANL-80-74, Argonne, Illinois, 1980.
- [51] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, "Chapter 9. Root Finding and Nonlinear Sets of Equations". Numerical Recipes: The Art of Scientific Computing (3rd ed.). (2007). New York: Cambridge University Press. ISBN 978-0-521-88068-8. [available online consulted 15 October 2012 <http://apps.nrbook.com/empanel/index.html#pg=442>]
- [52] M. J. D. Powell, A hybrid method for nonlinear equations, in Numerical Methods for Nonlinear Algebraic Equations, 1970, P. Rabinowitz, ed. , Gordon and Breach
- [53] C. G. Broyden, "A class of methods for solving nonlinear simultaneous equations," 1965, Math. comp. 19, 577-593
- [54] K. Levenberg, "A method for the solution of certain nonlinear problems in least squares," 1944, Quart. Appl. Math. 2, 164-168
- [55] D. W. Marquadt, "An algorithm for least-squares estimation of nonlinear parameters," 1963, SIAM J. Appl. Math. 11, 431-441
- [56] Journal of fixed points and applications, available online, consulted 15 October 2012: <http://www.fixedpointtheoryandapplications.com/>
- [57] J. M. Ortega, W. C. Rheinboldt, Iterative solution of nonlinear equations in several variables, Philadelphia, Pa. : Society for Industrial and Applied Mathematics, 2000 [En ligne] Permalink: <http://dx.doi.org/10.1137/1.9780898719468>
- [58] P. J. Olver, "Numerical solution of algebraic systems," in AIMS Lecture Notes on Numerical Analysis, 2006
- [59] C. F. Mugombozi, J. Mahseredjian, O. Saad, "On Fixed-point Iterations for the Solution of Control Equations in Power Systems Transients," IPST 2013, Vancouver, Canada
- [60] C. F. Mugombozi, O. Saad, T. Roudier, V. Morissette, R. Gagnon, G. Sybille, "Co-Simulation avec EMTP et MATLAB/SPS Appliquée à l'Intégration à Grande Échelle des Centrales Éoliennes," International Council on Large Electric Systems (CIGRE) Canada Conference, Montréal, septembre 2012
- [61] T. Yamamoto, "Historical developments in convergence analysis for Newton's and Newton-like methods," Journal of Computational and Applied Mathematics 124 (2000)

- [62] S. Stefano, The formula of Faà Di Bruno, reprinted from the American Mathematical Monthly, Vol. 87, No 10, December 1980, pp. 805-809
- [63] M. Hardy, Combinatorics of partial derivatives, The electronic journal of combinatorics, vol 13, no1, 2006; <http://www.combinatorics.org/ojs/index.php/eljc/issue/archive> consulted 2012-10-14
- [64] J. E. Dennis and R. B. Schnabel (1983). Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Society for Industrial and applied Mathematics, Philadelphia
- [65] B. Shirver, P. J. Eberlein, R. D. Dixon, "Permanent function of a square matrix I and II, " Comm. Assoc. Computing Machinery, vol. 12, 1969 , Algorithm 361
- [66] C. A. Desoer, "The Optimum Formula for the Gain of a Flow Graph or a Simple Derivation of Coates' Formula, " Proceedings of the IRE, 1959
- [67] R. Diestel Graph theory, Springer-Verlag, Heidelberg Graduate Texts in Mathematics, Volume 173, ISBN 978-3-642-14278-9 July 2010
- [68] M. Wu, "A convergence theorem for the Newton-like methods under some kind of weak Lipschitz conditions," Journal of Mathematical Analysis and Applications, v 339, n 2, p 1425-31, 15 March 2008
- [69] W. Shen, C. Li, "Kantorovich-type convergence criterion for inexact Newton methods," Applied Numerical Mathematics, v 59, n 7, p 1599-1611, July 2009
- [70] L. U. Uko, I. K. Argyros, "A weak Kantorovich existence theorem for the solution of nonlinear equations," Journal of Mathematical Analysis and Applications, Vol. 342, Issue 2, 15 June 2008, Pages 909-914,
- [71] P. Deulphard, G. Heindl, "Affine invariant convergence theorems for Newton's method and extensions to related methods," SIAM J. NUMER. ANAL. Vol. 16, No. 1, February 1979
- [72] A. M. Ostrowski, Solution of equations in Euclidean and Banach Space, Academic press, New York, 1973
- [73] I. K. Argyros, S. Hilout, "Weaker conditions of Newton's method, " Journal of complexity 28, (2012) 364-387
- [74] T. Yamamoto, Error bounds for Newton's iterates derived from Kantorovich theorem," Numer. Math. 48, 91-98 (1986), Springer-Verlag
- [75] C. F. Mugombozi, J. Mahseredjian, O. Saad, "Efficient computation of feedback-based control system equations for electromagnetic transients". En soumission à IEEE transactions on Power delivery, 2013
- [76] J. A. Martinez, J. Mahseredjian and B. Khodabakhchian, "Parameter determination for Model System Transients- Part VI: Circuit Breakers" , IEEE Transactions on Power delivery, vol. 20, no 3, July 2005, pp 2079-2085

ANNEXE 1 – Aperçus d’algorithmes utilisés

ANNEXE 1-1 Algorithme de « Depth-First Search »

```
algorithm dft( $f_i$ )  
  visit( $f_i$ )  
  FOR each  $f_j$  such that  $(f_i, f_j)$  is an edge DO  
    IF  $f_j$  was not visited yet THEN  
      dft( $f_j$ )
```

ANNEXE 1-2 Algorithme pour le calcul de la matrice de “reachability” (avec Matlab)

```
function matR= multbool_mat (A,B)  
n = size(B,2);  
for ii = 1: n  
  for jj = 1:n  
    sumcum = 0;  
    for ik = 1:n  
      sumcum = or(sumcum, and(A(ii,ik),B(ik, jj)));  
    end  
    matR(ii,jj) = sumcum ;  
  end  
end  
end
```

ANNEXE 1-3 Algorithme pour le calcul de “Strongly Connected Components” (SCC) (Matlab)

```
N = size(matR,1); % taille de la matrice d'accessibilité ( reachability matrix R)
matR_scc = zéros(N,N); % définition des dimensions de la matrice des composants fortement connectés (CFC ou
% SCC) en anglais
for ii= 1:N
    for jj = 1:N
        if matR(ii,jj) == matR(jj,ii)
            matR_scc(ii,jj) = 1;
        end
    end
end
end
```

ANNEXE 2 – Scripts pour l'étude des propriétés contractantes pour la méthode point- fixe du PLL

```

syms e1 e2 u hi1 hi2 hi3 ho1 ho2 ho3 g1 g2 g3 k Ts

% in_int2 = g1*e2 + (g2*e2*Ts/2 + hi1*Ts/2 +ho1)+k;
% int2 = in_int2*Ts/2 + hi2*Ts/2 +ho2;
% G2 = e1*sin(int2);
% in_int3 = e1* cos (int2);
% G1 = cos(int2) *g3*(in_int3*Ts/2 + hi3*Ts/2 +ho3);
% Jacobienne_FP_PLL_analytique = jacobian ( [u-G1, G2],[e1, e2]);
% g1 = 300; g2 = 600; g3 = 500; k = 377;
% Jacobienne_FP_PLL_analytique_val = eval(Jacobienne_FP_PLL_analytique);
% % calcul de valeurs propres
% % eig_FP_Jac = eig (Jacobienne_FP_PLL_analytique);
% % eig_FP_Jac_eval = simple(eval( eig_FP_Jac ));

Ts = 1e-6;
sim('PLL_simulink_inputEMTP.mdl')
results_simul_PLL_1us =
[ScopeData_e1.signals.values(:),ScopeData_e2.signals.values(:),ScopeData_hi1.signals.values(:),ScopeData_hi2.signals.values(:),ScopeData_hi3.signals.values(:),ScopeData_ho1.signals.values(:),ScopeData_ho2.signals.values(:),ScopeData_ho3.signals.values(:)];
tsimul=ScopeData_e1.time;save results_simul_PLL_1us

%load results_simul_PLL_1us ; % results_simul_PLL_1us et tsimul
e1 = results_simul_PLL_1us(:,1);
e2 = results_simul_PLL_1us(:,2);
lenT = size(results_simul_PLL_1us,1);
%reconstitution variables d'histoire du PLL:
hi1 = [0; results_simul_PLL_1us(1:lenT-1,3)];
hi2 = [0; results_simul_PLL_1us(1:lenT-1,4)];

```

```

hi3 = [0; results_simul_PLL_1us(1:lenT-1,5)];

ho1 = [0; results_simul_PLL_1us(1:lenT-1,6)];
ho2 = [0; results_simul_PLL_1us(1:lenT-1,7)];
ho3 = [0; results_simul_PLL_1us(1:lenT-1,8)];

jac_col1 = [ -250.*Ts.*cos(ho2 + (Ts.*hi2)./2 + (Ts.*(300.*e2 + ho1 + 300.*Ts.*e2 + (Ts.*hi1)./2 + 377))./2).^2, 250.*Ts.*sin(ho2 +
(Ts.*hi2)./2 + (Ts.*(300.*e2 + ho1 + 300.*Ts.*e2 + (Ts.*hi1)./2 + 377))./2).*(300.*Ts + 300).*(ho3 + (Ts.*hi3)./2 + (Ts.*e1.*cos(ho2
+ (Ts.*hi2)./2 + (Ts.*(300.*e2 + ho1 + 300.*Ts.*e2 + (Ts.*hi1)./2 + 377))./2))./2) + 125.*Ts.^2.*e1.*cos(ho2 + (Ts.*hi2)./2 +
(Ts.*(300.*e2 + ho1 + 300.*Ts.*e2 + (Ts.*hi1)./2 + 377))./2).*sin(ho2 + (Ts.*hi2)./2 + (Ts.*(300.*e2 + ho1 + 300.*Ts.*e2 +
(Ts.*hi1)./2 + 377))./2).*(300.*Ts + 300)];
jac_col2 = [
sin(ho2 + (Ts.*hi2)./2 + (Ts.*(300.*e2 + ho1 + 300.*Ts.*e2 + (Ts.*hi1)./2 + 377))./2),
(Ts.*e1.*cos(ho2 + (Ts.*hi2)./2 + (Ts.*(300.*e2 + ho1 + 300.*Ts.*e2 + (Ts.*hi1)./2 + 377))./2).*(300.*Ts + 300))./2];

for ii = 1:numel(eig_FP_Jac_val_t(:,1))
    jacobienne_PLL_analytique(:,1) = jac_col1(ii,:);
    jacobienne_PLL_analytique(:,2) = jac_col2(ii,:);
    norm2(ii) = norm(jacobienne_PLL_analytique,2); % euclidienne ( spectrale)
    normF(ii) = norm(jacobienne_PLL_analytique,'fro'); % frobenius
    norm1(ii) = norm(jacobienne_PLL_analytique,1); % maximal
    norm_inf(ii) = norm(jacobienne_PLL_analytique,'inf'); % infinie
end

% calcul des valeurs propres de la Jacobienne:
%norm_spectral
norm_spectral = norm2;
%norm2S = norm(eigjacobienne_PLL_analytique'*jacobienne_PLL_analytique,2);
figure;
subplot(1,2,1)
plot(tsimul,e1,'b')
title('e_1 pour le PLL résolu avec Point-fixe: 1us')
subplot(1,2,2)
plot(tsimul,e2,'b'),
title('e_2 pour le PLL résolu avec Point-fixe :1us')

```



```

saveas(gcf,'e1_e2_du_PLL_pointsFixes222','fig');

figure;
subplot(3,1,1)
semilogy(tsimul,abs(eig_FP_Jac_val_t(:,1)),'b')
title('Valeur propre no 1 pour Jacobienne Point-fixe du PLL')
subplot(3,1,2)
semilogy(tsimul,abs(eig_FP_Jac_val_t(:,2)),'b'),
title('Valeur propre no 2 pour Jacobienne Point-fixe du PLL')
subplot(3,1,3)
semilogy(tsimul,norm_spectral,'b')
title('Norme spectrale pour Jacobienne Point-fixe du PLL l')
saveas(gcf,'Valeurs_propres_nomr2_Jacobienne_PLL_pointsFixes','fig');

%recharger la figure des itérations établies par le nouveau code appliqué
%sur le PLL
uiopen('G:\doctorat2012\these\matlab\iters_vs_time_PLL_PF.fig',1)
hold on
%ajouter la norme spectrale
semilogy(tsimul,norm_spectral,'k'), title('Itérations et Norme spectrale en fonction du temps de simulation pour Point-fixe')
saveas(gcf,'Norm2FP_vs_itération_PLL_pointsFixes','fig');hold off;

% Comparaison des différentes façons de calculer la norme
%recharger la figure des itérations établies par le nouveau code appliqué
%sur le PLL
uiopen('G:\doctorat2012\these\matlab\iters_vs_time_PLL_PF.fig',1)
hold on
%ajouter les norm: une norme spectrale
semilogy(tsimul,norm_spectral,'k'), title('Itérations et Normes de la Jacobienne en fonction du temps de simulation pour Point-fixe')
semilogy(tsimul,norm1,'r-')
semilogy(tsimul,norm_inf,'g.')
semilogy(tsimul,norm2,'b-')
semilogy(tsimul,normF,'y.')
legend ( 'iterations','norm-spectral','1-norm','inf-norm','2-norm','F-norm')

```

```
saveas(gcf,'NormJaco_vs_itération_PLL_pointsFixes','fig');
save normPLLtest norm2 normF norm1 norm_inf jac_col1 jac_col2
```

```
% analyse groupe no4 dans eolienne:
```

```
%syms u1_grp4 u2_grp4 u3_grp4 u4_grp4 u5_grp4
syms u1 u2 u3 u4 u5 u6 e c2 c1
```

```
fm4 = e*u5;
fm5 = e*u6;
fm8 = fm4*u3;
fm9 = fm5*u4;
fm10 = fm8+fm9;
sum1 = fm10 + c2;
sel1 = c1.*(sum1-2+1) - u1.*(sum1-1-1);
fm3 = int ( sel1, e);
jac_phi_func_e = jacobian ( u2 + fm3,e);
% jac_phi_func_e =
% c1*(e*u5*u3+e*u6*u4+c2-1) - u1*(e*u5*u3+e*u6*u4+c2-2)
jac_phi_func_e = collect(jac_phi_func_e, e);
% (c1*(u5*u3+u6*u4)-u1*(u5*u3+u6*u4))*e+c1*(c2-1)-u1*(c2-2);
c1 = 0; c2 = 1;
jac_phi_func_e_eval = eval(jac_phi_func_e);
% jac_phi_func_e_eval =
% -u1*(u5*u3+u6*u4)*e+u1
% solution du groupe pour e:
fm3_eval = eval(fm3);
sol_e = solve ( fm3_eval - e, e);
% sol_e =
% 0
% 2*(u1-1)/u1/(u5*u3+u6*u4)
e = sol_e(1);
jac_phi_func_e_eval_1 = eval(jac_phi_func_e_eval);
% jac_phi_func_e_eval_1 =
```

```
% u1
e = sol_e(2);
jac_phi_func_e_eval_2 = eval(jac_phi_func_e_eval);
% jac_phi_func_e_eval_2 =
% -u1+2
% condition contraction :  $-1 < -u1+2 < 1$ 
% c'est-à-dire:  $-3 < -u1 < 3$  ou encore  $-3 < -u1*Ki < 3$ 
% ou Ki est #KiRotorSideCurReg# et vaut 8
```

ANNEXE 3 – Analyse de l'équivalence entre la méthode DIPR et MATLAB/Simulink avec les états discrets

```

% script pour la comparaison de la méthode DiPR et MATLAB/Simulink
% évaluation de la contrainte algébrique, des sorties des blocs et mise à
% jour des états dans le cas du modèle simple linéaire à 3 états avec les
% intégrateurs discrétisés via la méthode trapézoïdale (Tustin)
%% les variables
syms e1 e2 e3 u x1 x2 x3 y1 y2 y3 Ts hist1 hist2 hist3
%% Jacobienne pour calculer les boucles algébriques équivalentes aux
%% coupures DiPR
jacobian_CA_Tcontinu = jacobian([e2 - u - int(e1)+int(e3); e3 - u - int(e2)+int(e3); e1 - u+int(e3)], [e2, e3, e1]);
jacobian_CA_Tdiscret = jacobian([e2 - u - (hist1+e1*Ts/2)+(hist3+e3*Ts/2); e3 - u - (hist2+e2*Ts/2)+(hist3+e3*Ts/2); e1 -
u+(hist3+e3*Ts/2)], [e2, e3, e1]);
% jacobian_CA_Tdiscret =
% [    1,  1/2*Ts, -1/2*Ts]
% [ -1/2*Ts, 1+1/2*Ts,    0]
% [    0,  1/2*Ts,    1]
%% Évaluation des sorties y3, y1 et y2 avec  $y_i(k) = x_i(k) + (Ts/2)*input_i(k)$ 
y3 = (Ts^2/4 - Ts^3/8)*x1 + (Ts/2 - Ts^2/4)*x2 + (1 - Ts/2 - Ts^3/8)*x3 + (Ts/2)*u - Ts^3*(e1 + e3);
y1 = x1 + (Ts/2)*u - (Ts/2)*y3; y1 = collect(y1, [x1, x2, x3, u]);
%%
mat_C_dYdX = jacobian([y1; y2; y3], [x1, x2, x3]);
mat_D_dYdu = jacobian([y1; y2; y3], [u]);
%% Mise à jour des états
syms x1k x2k x3k
x1_kp1 = y1 + (Ts/2)*(u - y3);
x2_kp1 = y2 + (Ts/2)*(u + y1 - y3);
x3_kp1 = y3 + (Ts/2)*(u + y2 - y3);
mat_A_dX_kp1dXk = jacobian([x1_kp1; x2_kp1; x3_kp1], [x1, x2, x3]);

```

```
mat_B_dX_kp1dXk = jacobian([x1_kp1; x2_kp1 ; x3_kp1],[u]);
```

```
Ts = 50e-6;
```

```
matA_eval = eval(mat_A_dX_kp1dXk)
```

```
matB_eval = eval(mat_B_dX_kp1dXk )
```

```
matC_eval = eval(mat_C_dYdX)
```

```
matD_eval = eval(mat_D_dYdu)
```

```
Ts = 250e-6;
```

```
matA_eval_250us= eval(mat_A_dX_kp1dXk)
```

```
matB_eval_250us= eval(mat_B_dX_kp1dXk )
```

```
matC_eval_250us= eval(mat_C_dYdX)
```

```
matD_eval_250us = eval(mat_D_dYdu)
```

```
% matA_eval =
```

```
% 9.999999999999687e-001 -1.249968750000000e-009 -4.999874999999922e-005
```

```
% 4.999999996875000e-005 9.999999987499999e-001 -4.999999996874921e-005
```

```
% 1.249968750000000e-009 4.999875000000000e-005 9.999499999999842e-001
```

```
% matB_eval =
```

```
% 4.999875000000000e-005
```

```
% 4.999999996875000e-005
```

```
% 4.999999999999961e-005
```

```
% matC_eval =
```

```
% 9.999999999999844e-001 -6.249843750000001e-010 -2.499937499999961e-005
```

```
% 2.499999998437500e-005 9.99999993750000e-001 -2.499999998437461e-005
```

```
% 6.249843750000000e-010 2.499937500000000e-005 9.999749999999843e-001
```

```
% matD_eval =
```

```
% 2.499937500000000e-005
```

```
% 2.499999998437500e-005
```

```
% 2.500000000000000e-005
```

```
% matA_eval_250us =
```

```
% 9.999999999960942e-001 -3.124609375000000e-008 -2.499687499995117e-004
```

```
% 2.499999960937501e-004 9.999999687500005e-001 -2.499999960932617e-004
```

```
% 3.124609375000000e-008 2.499687500000000e-004 9.997499999980474e-001
```

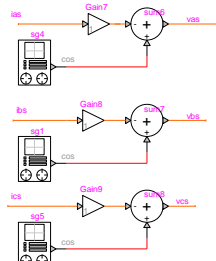
```
% matB_eval_250us =
```

```
% 2.499687500000000e-004
% 2.499999960937500e-004
% 2.49999999997558e-004
% matC_eval_250us =
% 9.99999999980471e-001 -1.562304687500000e-008 -1.249843749997559e-004
% 1.249999980468750e-004 9.999999843750003e-001 -1.249999980466308e-004
% 1.562304687500000e-008 1.249843750000000e-004 9.998749999980469e-001
% matD_eval_250us =
% 1.249843750000000e-004
% 1.249999980468750e-004
% 1.250000000000000e-004
```

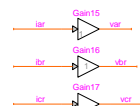
ANNEXE 4 – Machine asynchrone entièrement modélisée par équations algébriques dans EMTP-RV

Control/power conversion and per unit conversion

Stator Connection with a resistance and a source

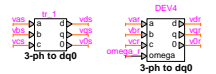


Rotor Connection with a resistance

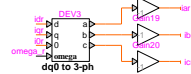


Park's transformations

abc to dq0



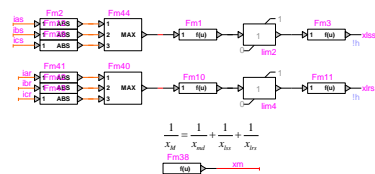
dq0 to abc



Saturation of the leakage reactances

$$DF = \frac{2}{\pi} \left[\sin^{-1} \left(\frac{I_m}{I} \right) + \left(\frac{I_m}{I} \right) \sqrt{1 - \left(\frac{I_m}{I} \right)^2} \right] \text{ where } I_m < I \text{ else } DF = 1$$

$$x_{ls} = \frac{x_{ls0}}{2} (1 + DF)$$



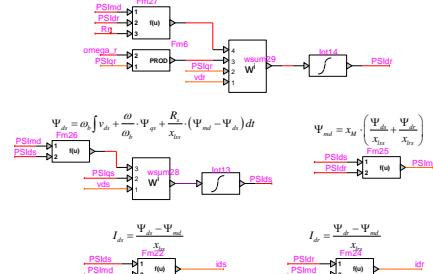
Fixed rotor resistance



Asynchronous Machine

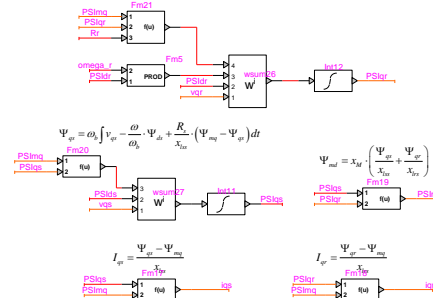
D axis equations

$$\Psi_d = \omega_s \int v_d + \left(\frac{\omega - \omega_s}{\omega_s} \right) \cdot \Psi_q + \frac{R_s}{x_{ls}} (\Psi_{sd} - \Psi_d) dt$$



Q axis equations

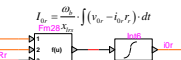
$$\Psi_q = \omega_s \int v_q - \left(\frac{\omega - \omega_s}{\omega_s} \right) \cdot \Psi_d + \frac{R_s}{x_{ls}} (\Psi_{sq} - \Psi_q) dt$$



0 axis equations

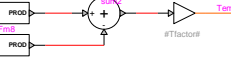
$$I_{\phi} = \frac{\omega_s}{x_{ls}} \int (v_{\phi} - i_{\phi} r_{\phi}) dt$$

$$I_{\phi} = \frac{\omega_s}{x_{ls}} \int (v_{\phi} - i_{\phi} r_{\phi}) dt$$



The torque equation

$$T_{em} = \frac{3P}{4\omega_s} (\Psi_d i_q - \Psi_q i_d)$$



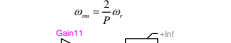
The rotor angular speed



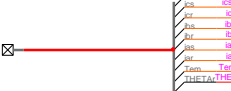
The rotor angle



The rotor speed



Mesures



Gain2S-999
Gain3S-999
Gain7S-999
Gain8S-999
Gain9S-999
Gain10S-999
Gain11S-999
Gain12S-999
Gain13S-999
Gain14S-999
Gain15S-999
Gain16S-999
Gain17S-999
Gain18S-999
Gain19S-999
Gain20S-999

limits
MPLLOT

ANNEXE 5 – Script pour calculer le permanent d'une matrice carrée

```

function y=permanent(A)
%PERMANENT(A) Calculate the permanent of a square matrix A, which is
%defined as the analog of determinant where the signs of each term in
%summation was removed.
%Example, the matrix
%A=[1 2
% 3 4]
% is p(A)=1x4+2x3=10.
%written by C.Xu, Nov.,2008, Hangzhou,China. All rights reserved.
[m,n]=size(A);
%Restrict A to be square
if (m~=n)
    error('A must be square');
end
if n==1
    y=A;
else
    for k=2:n
        P=ones(1,k);
        for i=1:k
            SubA=A([1:k-1],[1:i-1 i+1:k]);
            P(i)=permanent(SubA);
        end
        y=A(k,1:k)*P';
    end
end
end

```