



AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur. Ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite encourt une poursuite pénale.

Contact : ddoc-theses-contact@univ-lorraine.fr

LIENS

Code de la Propriété Intellectuelle. articles L 122. 4

Code de la Propriété Intellectuelle. articles L 335.2- L 335.10

http://www.cfcopies.com/V2/leg/leg_droi.php

<http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm>

Pilotage des cycles limites dans les systèmes dynamiques hybrides. Application aux alimentations électriques statiques

THÈSE

présentée et soutenue publiquement le 6 Février, 2009

pour l'obtention du

Doctorat de l'Institut National Polytechnique de Lorraine

Spécialité Automatique et Traitement du signal

par

Diego Alejandro Patiño Guevara

Composition du jury

<i>Président :</i>	Jean Marie Rétif	Professeur, INSA, Lyon
<i>Rapporteurs :</i>	Jean Buisson Mazen Alamir	Professeur, SUPELEC, Rennes Directeur de Recherche au CNRS Gipsa-lab, INP Grenoble
<i>Examineurs :</i>	Jean Pierre Barbot Ulf Jonsson	Professeur, ENSEA, Cergy-Pontoise Professeur, Royal Institute of Technology (KTH), Stockholm, Suède
	Pierre Riedinger Claude Iung	Maître de Conférences, INP Lorraine (co-directeur de thèse) Professeur, INP Lorraine (directeur de thèse)



Centre de Recherche en Automatique de Nancy
UMR 7039 Nancy-Université – CNRS

2, avenue de la forêt de Haye 54516 Vandœuvre-lès-Nancy
Tél.+33 (0)3 83 59 59 59 Fax +33 (0)3 83 59 56 44

Mis en page avec la classe thloria.

Remerciements

Les travaux de recherche présentés dans cet mémoire ont été effectués au Centre de Recherche en Automatique de Nancy (CRAN) une Unité Mixte de Recherche Nancy Université - INPL - CNRS - UMR 7039. Je remercie mon directeur de thèse, Monsieur le professeur Claude Iung pour ses remarques constructives toujours intelligentes. A mon codirecteur de thèse, Monsieur le maître de conférences Pierre Riediger pour son enthousiasme, son soutien, ses conseils et sa patience qui m'ont grandement aidé à mener à bien mes travaux de thèse. Pierre, un très grand merci pour tout !

Ces remerciements s'adressent également à M. Jean Buisson, professeur à SUPE-LEC Rennes, et à M. Mazen Alamir, directeur de Recherche au CNRS GIPSA-lab, INP Grenoble, pour m'avoir fait l'honneur de rapporter cette thèse.

Ma reconnaissance à M. Jean Marie Retif, professeur à l'INSA, Lyon, qui m'a fait l'honneur de présider mon jury ainsi qu'à M. Jean Pierre Barbot, professeur à l'ENSEA en Cergy-Pontoise et à M. Ulf Jonsson, Professeur au Royal Institute of Technology, Stockholm, pour avoir accepté le rôle d'examineurs et pour leurs remarques qui m'ont aidé dans l'élaboration de la version finale de ce manuscrit.

Je souhaite remercier les membre du laboratoire CRAN pour leur aide, en particulier M. Jamal Daafouz, pour avoir suivi mon parcours en France depuis le début ainsi que Marc Junges, Radu Ranta, Samia Maza, Benoit Marx et Nicolae Brânzei pour leur sympathie. Bien sûr, aussi à J. M. Munier pour tout son aide dans la construction de la maquette.

Je ne peux pas oublier à mes collègues que j'ai connus à l'ENSEM : Sophie, Emilie, Ashraf, Julien, Julie, Yvon, Nadia et Pascal. A Laurentiu pour son soutien, son guide, son amitié et les discussions scientifiques, toujours très intéressantes.

Je tiens à remercier mes amis hispanophones. Agradezco a Yahir y a Sinuhé por acogerme cuando llegué sin conocerme anteriormente. Es algo que no voy a olvidar jamas. Igualmente agradezco a Hugo, Rebeca, Yolanda, Gilberto, Joel, Evaristo, Luis, Hernando, los disidentes (/') Flavia y la tía Leticia (se me olvidó alguien?). Agradezco a Ivan (grazie mille per tutto il mio caro amico, mio fratello). Viví momentos realmente inolvidables con todos ustedes que quedaran siempre grabados en mi corazón. Bien sûr, Annita, comment je ne peux pas te mentionner ?.

Aussi en Espagnol, je voudrais remercier à ma famille. Gracias a mis padres, Alejandro y Esperanza, por su amor y apoyo incondicional, siempre ha sido muy importante en mi vida. También a mis hermanos Francy y Ernesto por ser mis compañeros desde niño. Al Profesor René Meziat por estar siempre disponible, creer en mi y por marcar una guía. Por ultimo, gracias a Marcela por su paciencia

a los 8456 km de distancia, su comprensión, pero sobre todo por todo su apoyo y amor.

A mes parents : Alejandro et Esperanza.

A mes grandes mères qui ne sont plus là : Transito et Aura

A ma lumière et mon avenir toujours présente : Marcela.

Table des matières

Introduction Générale	1
Chapitre 1 Notions introductives	5
1.1 Introduction	5
1.2 Comportement asymptotique des systèmes non autonomes	7
1.2.1 Ensembles limites et invariants	8
1.2.2 Stabilité des cycles	10
1.2.2.1 Les fonctions de sensibilité	11
1.2.2.2 Les sections de Poincaré	12
1.2.2.3 Calcul pratique de stabilité	15
1.3 Systèmes hybrides et stabilité des cycles	16
1.3.1 Définitions. Séquences de temps et exécutions.	16
1.3.2 Résultats de stabilité de cycles pour les systèmes hybrides	20
1.3.2.1 Stabilité de Lyapunov	23
1.4 Problématique de commande des systèmes commutés cycliques . .	24
1.4.1 Classe d'applications : les convertisseurs statiques	25
1.4.2 Points de fonctionnement	26
1.4.3 Objectifs de la commande	27
1.5 Conclusions	28
Chapitre 2 Commande prédictive des systèmes commutés cycliques	29
2.1 Introduction	29
2.2 Principe de la commande prédictive	33
2.3 Commande prédictive pour les systèmes commutés cycliques . . .	35
2.3.1 Formulation du problème	35
2.3.2 Régime permanent x^∞	36

2.3.3	Stratégie de résolution du problème (2.5)	37
2.4	Résolution du problème de la commande prédictive	39
2.4.1	Régime permanent	39
2.4.1.1	Exemple numérique (Convertisseur buck-boost)	40
2.4.2	Optimisation des temps de commutation entre modes	41
2.4.2.1	Exemple numérique (Reprise du convertisseur buck-boost)	46
2.5	Construction de la boucle fermée	48
2.6	Conclusions et perspectives	49
 Chapitre 3 Commande optimale par retour d'état des systèmes com-		
mutés		51
3.1	Introduction	51
3.2	Rappels et Préliminaires	53
3.2.1	Conditions d'existence	55
3.2.2	Principe du minimum de Pontriaguine	56
3.2.3	Conditions de transversalité pour les systèmes commutés	58
3.3	Formulation du problème pour des systèmes cycliques	60
3.3.1	Application du principe de minimum	62
3.3.2	Type de solutions	63
3.4	Arcs singuliers	64
3.4.1	Points de fonctionnement et arcs singuliers	64
3.4.2	Une seule commande singulière ($r = 1$)	65
3.4.2.1	Résultats de base et condition nécessaire d'ordre deux	65
3.4.2.2	Une méthode pratique pour l'obtention des arcs singuliers pour $r = 1$	67
3.4.3	Cas multi-commande $r > 1$	72
3.4.3.1	Résultats de base et condition nécessaire d'ordre deux	72
3.4.3.2	Une méthode pratique pour l'obtention des arcs singuliers lorsque $r > 1$	74
3.5	Synthèse des trajectoires optimales	78
3.6	Exemple (Convertisseur buck-boost)	82
3.7	Conclusions et Perspectives	86

Chapitre 4 Application aux convertisseurs statiques	89
4.1 Introduction	89
4.2 Le convertisseur buck	91
4.2.1 Topologie	91
4.2.1.1 Modèle mathématique	92
4.2.1.2 Cahier des charges	93
4.2.2 Commande prédictive	93
4.2.2.1 Détermination du régime permanent	94
4.2.2.2 Construction de la boucle fermée	95
4.2.2.3 Résultat pratique et de simulation	96
4.2.3 Commande optimale	101
4.2.3.1 Recherche des arcs singuliers	101
4.2.3.2 Retour d'état	102
4.2.3.3 Résultats pratiques et de simulation	104
4.2.4 Quelques commentaires des résultats	104
4.3 Convertisseur multiniveaux	109
4.3.1 Topologie et état de l'art	109
4.3.1.1 Modèle mathématique	111
4.3.1.2 Cahier de charges	111
4.3.2 Commande prédictive	114
4.3.2.1 Détermination du régime permanent	114
4.3.2.2 Construction de la boucle fermée	115
4.3.2.3 Résultats pratiques et de simulation	116
4.3.3 Commande optimale	116
4.3.3.1 Détermination des arcs singuliers	120
4.3.3.2 Retour d'état	122
4.3.3.3 Résultats pratiques et de simulation	123
4.3.4 Quelques commentaires des résultats	123
4.4 Conclusion	127
Conclusion générale et perspectives	129
Annexe A Détermination des lois de commande avec Matlab	133
A.1 Commande prédictive	134
A.1.1 Calcul du régime permanent	134

A.1.2 Optimisation des temps de commutation	137
A.2 Commande optimale	138
Bibliographie	151

Introduction Générale

Cette thèse s'intéresse au pilotage des cycles limites pour une classe particulière de systèmes hybrides : les systèmes commutés cycliques. La thématique des systèmes hybrides est née du constat d'insuffisance des modèles dynamiques classiques pour décrire la complexité des comportements notamment lorsque des aspects événementiels interviennent dans l'évolution des processus continus. Une classe particulièrement importante de SDH en termes d'applications est formée par celle qui présente un régime permanent cyclique (moteur à explosion, convertisseur de puissance, etc.). Ces systèmes se caractérisent par des points de fonctionnement non auto-maintenable : il n'existe pas de commande qui maintienne le système sur le point de fonctionnement désiré. Le maintien n'est alors assuré qu'en valeur moyenne, en effectuant un cycle dans un voisinage du point de fonctionnement par commutation des différents sous systèmes.

Objectifs

L'établissement d'une loi de commutation pour cette classe de systèmes non linéaires doit répondre aux objectifs classiques de stabilité, de performance dynamique et de robustesse, mais doit également garantir la satisfaction de critères liés à la forme d'onde du régime permanent (harmonique, oscillations, etc.). A l'heure actuelle, peu de méthodes prennent en compte directement le caractère cyclique du système lors de la phase de synthèse.

Les travaux de cette thèse ont pour objectif de développer des méthodes génériques, performantes et robustes pour piloter cette classe de systèmes. Les algorithmes proposés doivent également pouvoir être implémentés en temps réels. On modélise le système commuté comme un système non - linéaire affine en la commande dont la grandeur de la commande (commutation) apparaît explicitement dans le modèle. Ce type de modélisation nous permet d'envisager deux types de synthèse : l'un à base de commande prédictive et l'autre à base de commande optimale. La commande prédictive est bien adaptée au problème de poursuite et de suivi de trajectoire. Elle sera donc privilégiée pour maintenir le système sur son cycle limite.

Les outils de la commande optimale permettent d'apporter des solutions performantes lors de grands mouvements des phases de ralliement. L'importance jouée par les arcs singuliers dans cette synthèse est clairement mise en évidence.

Ce travail est validé par une partie applicative importante sur des manipulations développées au CRAN et dans des laboratoires partenaires du réseau d'excellence européenne HYCON dans le cadre duquel s'est déroulé cette étude.

Organisation du mémoire

Ce mémoire est organisé en 4 chapitres qui se présentent comme suit :

Chapitre 1

Le premier chapitre est une introduction aux cycles et aux systèmes hybrides. Nous donnerons quelques rappels sur le comportement asymptotique des systèmes non linéaires et nous utiliserons les concepts d'ensembles limites et d'invariants. Ensuite, nous présenterons les systèmes dynamiques hybrides et quelques définitions qui nous seront utiles par la suite. Enfin, nous présentons la problématique de commande des systèmes commutés cycliques.

Chapitre 2

Dans le deuxième chapitre, une méthode de commande prédictive sera proposée. Cette approche sélectionne le cycle par l'utilisation d'un critère. Le choix de ce critère permet alors de limiter les oscillations du signal autour du point de fonctionnement. Le ralliement à ce cycle optimal s'effectue par une commande prédictive suivant deux stratégies en fonction de la distance à ce cycle. Les fonctions de sensibilité sont utilisées pour la synthèse des trajectoires optimales. Le respect des contraintes de temps de calcul imposées pour l'implémentation en temps réel est garanti par l'apprentissage de la commande par un réseau de neurones. Une relation simple entrée-sortie résultante définit alors une commande par retour d'état.

Chapitre 3

Ce chapitre est consacré à la recherche des lois de commutation par la théorie de la commande optimale. Il est montré que cette synthèse passe nécessairement pour l'étude d'un problème de commande optimale singulière. Ce problème réputé difficile se caractérise par une indétermination de la commande par le principe du minimum. Il est alors nécessaire d'utiliser des conditions d'ordre deux. Les travaux développés montrent que l'établissement formel des commandes singulières candidates peut être obtenu par l'utilisation conjointe de conditions d'ordre deux et d'un ensemble de conditions algébriques. Un algorithme de synthèse en temps rétrograde permet alors de générer l'ensemble des trajectoires optimales depuis les surfaces singulières sur une région de l'espace d'état donnée. La méthode proposée présente également l'avantage d'éviter le traditionnel problème "aux deux

bouts” pour des systèmes d’ordre faible. Cette synthèse se termine par une interpolation des solutions et produit une commande optimale par retour d’état gage de robustesse.

Chapitre 4

L’objectif du dernier chapitre est de valider ces travaux. Les deux approches ont été expérimentées sur des convertisseurs de puissance (Buck, boost, buck-boost, multiniveaux, ...). Ce chapitre donne les résultats obtenus en simulation et sur maquettes réelles situées à SUPELEC Rennes, à ETH Zurich et au CRAN. Les tests de robustesse montrent une excellente tenue vis-à-vis de variations paramétriques, de source et de charge.

Production scientifique

Les recherches présentées dans cette thèse figurent dans les références suivantes :

Revues

1. **D. PATINO**, P. RIEDINGER, C. IUNG. Practical optimal state feedback control law for continuous-time affine switched systems with cyclic steady state. *International Journal of control*, 2008, **Acceptée, à paraître**.
2. M. BAJA, **D. PATINO**, H. CORMERAIS, P. RIEDINGER, et. al. Hybrid control of a four-level three-cell dc-dc converter. *IEEE Transaction on Control Systems Technology*, 2008, **Soumis**.
3. S. MARIÉTHOZ, S. ALMÉER, A.G. BECCUTI, **D. PATINO**, et. al. Comparative Assessment of Alternative Control Techniques for Synchronous Step Down DC-DC Converters. *IEEE Transaction on Control Systems Technology*, 2008, **Soumis**

Revues électroniques

4. **D. PATINO**, P. RIEDINGER, C. IUNG . Une approche predictive pour la commande d’un convertisseur. *e-revue science & technologies de l’automatique*, Vol. 4, No. 4, 2007.

Conférences nationales et internationales avec comité de lecture

5. M. BAJA, **D. PATINO**, H. CORMORAIS P. RIEDINGER, J. BUISSON, Hybrid control methods for a Single Ended Primary Inductor Converter (SEPIC),

European Control Conference, 2009, **Soumis**.

6. **D. PATINO**, P. RIEDINGER, C. IUNG. Predictive control approach of a four-level three-cell DC-DC Converter. *IEEE Annual Conference of the Industrial Electronics Society*, Orlando, USA, Novembre, 2008.
7. **D. PATINO**, P. RIEDINGER, C. IUNG, Synthèse de commandes optimales pour les systèmes commutés cycliques : Applications aux convertisseurs de puissance, *Conférence Internationale Francophone d'Automatique*, Septembre 2008, Bucarest, Roumanie.
8. A. STEFAN, H. FUJIOKA **D. PATINO**, P. RIEDINGER, M. MORARI, et. al., Hybrid control Techniques for a switched-mode DC-DC Converter, Part I : The Step-Down Topology, Juillet 2007, New York, USA.
9. M. BAJA, **D. PATINO**, H. CORMORAIS P. RIEDINGER, J. BUISSON, Hybrid control of a three-level three-cell converter, *IEEE American Control Conference*, Juillet 2007, New York, USA.
10. **D. PATINO**, P. RIEDINGER, C. IUNG, Une approche predictive pour la commande d'un convertisseur multicellulaire, *Journées Doctorales et Nationales du GDR MACS*, Reims, France. Juillet, 2007.

Rapports

11. **D. PATINO**, P. RIEDINGER, et. al. Hybrid Control Techniques Applied to the Synchronous Step-Down DC-DC Converter. Rapport HYCON, 2006.
12. **D. PATINO**, P. RIEDINGER, et. al. Hybrid Control Techniques for a multilevel converter. Rapport HYCON, 2006.

Chapitre 1

Notions introductives

Résumé. Dans ce chapitre, nous présentons quelques notions utilisées pour l'étude et l'analyse du comportement asymptotique des systèmes dynamiques. Pour la classe des systèmes non-linéaires et non autonomes, les notions d'ensemble limite et leur déclinaison, en point d'équilibre, cycle et attracteur sont rappelées. L'étude des cycles et de leur stabilité au moyen des fonctions de sensibilité et des sections de Poincaré est évoquée et adaptée au cas des systèmes hybrides. A la fin de ce chapitre, nous présentons le problème de la commande d'un système hybride particulier dit commuté dont le point de fonctionnement recherché ne peut être considéré comme un point d'équilibre qu'en valeur moyenne. La régulation de ce type de système conduit alors à un régime cyclique.

Mots clés. Cycle limite, Fonctions de sensibilité, sections de Poincaré, Systèmes hybrides, Fonction de Lyapunov, point de fonctionnement, théorèmes de relaxation.

1.1 Introduction

Un grand nombre de phénomènes évolutifs issus de la physique s'écrivent sous la forme de systèmes d'équations différentielles ordinaires et forment une classe de "systèmes dynamiques" au sens décrit par Kalman [[KFA69](#)].

La première étape dans l'étude et l'analyse de ces systèmes commence typiquement par la recherche des équilibres, c'est-à-dire des solutions stationnaires ne présentant pas d'évolution temporelle. On dit qu'un équilibre est asymptotiquement stable si une petite perturbation de cet équilibre évolue dans le temps en restant petite et en convergeant vers zéro. Le système revient alors dans sa position d'équilibre. Ces équilibres ont donc de grandes chances d'être observés sur le système physique étudié ou sur la simulation numérique du modèle proposé.

L'étape suivante consiste à faire varier un ensemble de paramètres contrôlant le système. Ce qui est connu comme la commande du système. On regarde alors ce que deviennent les équilibres du modèle, en particulier ceux qui étaient stables avant de modifier les paramètres de commande. Lorsqu'en variant les valeurs de la commande, le système n'arrive pas à un point d'équilibre, mais à une trajectoire périodique, on dit que l'on est en présence d'un cycle limite.

Considérons l'exemple classique d'un oscillateur de Van der Pol naturel décrit par l'équation non-linéaire :

$$\frac{d^2x(t)}{dt^2} - \epsilon\omega_0 (1 - x^2(t)) \frac{dx(t)}{dt} + \omega_0^2 x(t) = 0$$

où $\omega_0 > 0$. ϵ est considéré comme la commande. Une analyse classique de ce système montre que l'équilibre n'est pas un point, mais une trajectoire unique et périodique (fonction du temps) lorsque $\epsilon \neq 0$. Voir la figure 1.1.

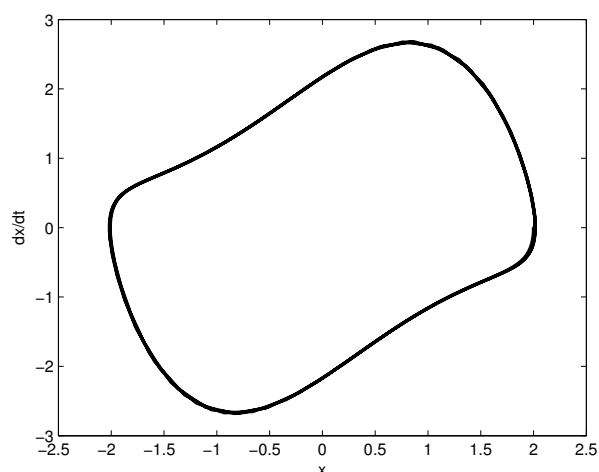


FIG. 1.1 – Cycle limite de l'oscillateur de Van-der-Pol pour $\omega_0 = 1$

Un bel exemple de système avec un régime permanent cyclique est illustré par le fonctionnement du coeur humain. En effet, le coeur ne reste jamais sur un point fixe. Le coeur se contracte (systole) et se relâche (diastole) continûment. Le fonctionnement normal du coeur est donc un mouvement cyclique [Law73].

Des nombreux systèmes dynamiques hybrides (SDH) présentent un régime permanent cyclique. Ils constituent une classe particulièrement importante car beaucoup d'applications entrent dans ce cadre. Par exemple, les moteurs thermiques des automobiles ou les convertisseurs de puissance (pour lesquelles le chapitre 4 est complètement dédié). Ces derniers sont des dispositifs chargés d'assurer une alimentation convenable à un appareil électrique par conversion et adaptation de la forme d'un signal électrique (tension / courant / fréquence) issu d'une

source. Cette modulation est obtenue en utilisant des technologies à base de semi-conducteurs dont les commutations de modes (ouverture et fermeture des interrupteurs) à fréquences élevées vont permettre de “découper” un courant ou une tension.

Ici, dans ce chapitre, nous voulons résumer quelques notions fondamentales liées aux systèmes avec un régime permanent cyclique. Nous donnerons quelques résultats importants pour les systèmes avec un régime permanent cyclique. En détail, nous verrons la définition de cycle. Nous rappellerons les outils existants pour analyser la stabilité des trajectoires cycliques sur des systèmes non-linéaires. Nous présenterons aussi la définition d’un système hybride et également quelques outils existants pour en analyser la stabilité lorsque le régime permanent est cyclique (stabilité du cycle). Finalement, nous présentons un modèle sous la forme d’un seul champ de vecteurs, discuterons ses points de fonctionnement et énoncerons un théorème (Théorème de densité) qui aidera à comprendre les résultats présentés plus tard, dans cet mémoire.

1.2 Comportement asymptotique des systèmes non autonomes

Considérons un système d’équations différentielles non linéaires variant dans le temps, défini dans $\mathbb{R}^n$ et représenté formellement par :

$$\begin{aligned}\dot{x}(t) &= F(x(t), t) \\ x(0) &= x_0\end{aligned}\tag{1.1}$$

où $x(t) \in \mathbb{R}^n$ est l’état du système, $t \in \mathbb{R}$, $F : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n$ et x_0 une condition initiale. Nous supposons que F vérifie les conditions nécessaires permettant de définir des solutions au sens de Carathéodory [LR07].

Nous supposons que les solutions sont globales dans le temps et noterons $x : \mathbb{R} \rightarrow \mathbb{R}^n$, une trajectoire associée au problème (1.1).

Dans toute le mémoire et dans ce chapitre en particulier, nous ne cherchons pas à nous placer dans les hypothèses les moins restrictives pour obtenir les résultats. En conséquence, on admettra que les conditions suffisantes sont remplies pour que les objets que nous utilisons aient un sens (continuité et dérivabilité en particulier).

Nous commencerons cette section avec une étude portant dans une large mesure sur le comportement asymptotique des trajectoires, en l’occurrence sur le comportement d’une trajectoire x lorsque t tend vers $+\infty$.

1.2.1 Ensembles limites et invariants

Voyons quelques définitions utiles pour la suite.

Définition 1 (*Point ω – limite et ensemble ω – limite* [SC64], [Per01])

Le point z est un point ω – limite d’une trajectoire donnée x pour le système dynamique (1.1) s’il existe une suite de temps,

$$t_0 < t_1 < \dots < t_n, \quad (1.2)$$

telle que $t_n \rightarrow +\infty$ et :

$$\lim_{n \rightarrow \infty} x(t_n) = z. \quad (1.3)$$

L’ensemble des points ω – limite de la trajectoire x est appelé ensemble ω – limite et il est noté $\Omega(x)$.

Définition 2 (*Point d’équilibre*) Un point $x_{eq} \in \mathbb{R}^n$ est dit un point d’équilibre du système (1.1) s’il existe un instant t_{eq} tel que :

$$F(x_{eq}, t) = 0 \quad \forall t \geq t_{eq}$$

Définition 3 (*Cycle périodique* [SC64], [Per01]) Une trajectoire x du système (1.1) est dite un cycle périodique si elle n’est pas réduite à un point et si il existe une constante $T_p > 0$ telle que

$$x(t + T_p) = x(t), \quad -\infty < t < +\infty \quad (1.4)$$

La constante T_p est la période du cycle. Nous dirons donc que x a une période T_p .

De cette définition, il est évident que si T_p est une période, alors pour tout entier positif k le nombre kT_p , ($k > 1$) est également une période.

Définissons la notion de presque-périodique :

Définition 4 (*Sous-ensemble dense* [SV85]) Un sous-ensemble S de $\mathbb{R}$ est dit relativement dense s’il existe un nombre positif L tel que $[a, a + L] \cap S \neq \emptyset$ pour tout $a \in \mathbb{R}$.

Un exemple classique est la suite des passages par zéro d’une sinusoïde perturbée par un jitter.

Définition 5 (*Cycle presque-periodique* [SV85])

1. Considérons une trajectoire unidimensionnelle x continue et un nombre positif ϵ ; $\tau(\epsilon)$ est un nombre de translation de x si :

$$\|x(t + \tau(\epsilon)) - x(t)\| \leq \epsilon \quad \forall t \in \mathbb{R}$$

2. La trajectoire $x(t)$ est appelée *presque-périodique* si pour tout $\epsilon > 0$, un ensemble relativement dense de nombres de translation $\tau(\epsilon)$ existe.

Une manière d'analyser ces trajectoires est de considérer la transformée de Fourier (spectre). Un cycle périodique est caractérisé par un spectre discret dont les rapports des fréquences sont rationnels. La figure 1.3 montre ce cas.

Les cycles quasi-périodiques englobent les cycles périodiques et des cycles dits "quasi-périodique" pour lesquels le spectre de la trajectoire est discret mais dans des rapports irrationnels. La figure 1.4 montre un attracteur quasi-périodique. Le mouvement de cette trajectoire est décrit par un tore.

On peut classifier les ensembles ω – limite associés à une trajectoire x en trois sous catégories :

- x converge vers un point particulier x_{eq} . Ce point est un point d'équilibre. D'après la définition 1, le lemme suivant est évident :

Lemme 6 [SC64] Si la trajectoire x du système (1.1) tend vers un point x_{eq} lorsque $t \rightarrow +\infty$, alors l'ensemble limite $\Omega(x) = x_{eq}$ et réciproquement.

La figure 1.2 montre un exemple de trajectoire lorsque x converge vers un point d'équilibre.

- x converge vers un cycle périodique ou presque périodique γ . Dans ce cas, si $\gamma \neq x$ (la notation $\gamma \neq x$ implique que le lieu géométrique associé à ces deux trajectoires n'est pas identiquement confondu), alors le cycle γ est un cycle limite.

Définition 7 (Cycle limite [SC64], [Per01]) Un cycle périodique ou presque périodique γ de (1.1) est appelé *cycle limite* s'il existe au moins une autre trajectoire x telle que $x \neq \gamma$ et $\Omega(x) = \gamma$.

- x converge vers un ensemble connexe, borné qui n'est ni réduit à un point, ni à un cycle. On entre dans le cadre de la théorie du chaos. Voir [PJS92], [Str94], [ASY97], [RMB06]. La figure 1.5 montre le cas d'un attracteur de Lorentz.

La théorie du chaos n'est pas simple et déborde largement de nos objectifs. En revanche, les autres ensembles limites vont être privilégiés par la suite comme objectifs à atteindre dans le pilotage des systèmes commutés.

Remarque 1 *Considérons le système :*

$$\dot{x}(t) = F(u(t), x(t), t)$$

où $u(t) \in \mathbb{R}^m$ est un vecteur composé par des entrées externes. Les définitions de point d'équilibre, cycle, cycle limite et attracteur étrange sont aussi valables si la fonction u est donnée.

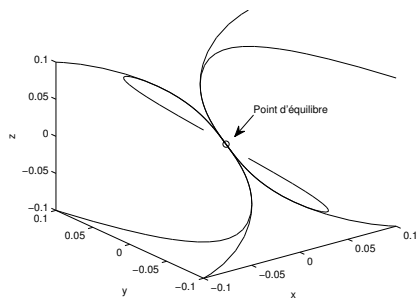


FIG. 1.2 – Point d'équilibre. L'ensemble $\Omega(x) = x_{eq}$.

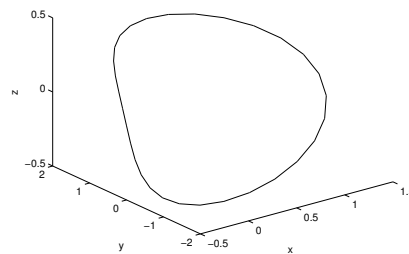


FIG. 1.3 – Cycle périodique ($x(t + T_p) = x(t)$). La trajectoire reste sur une orbite.

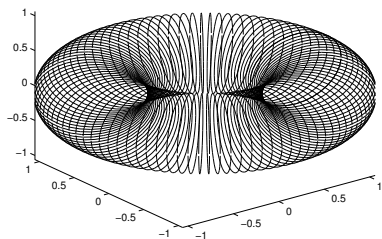


FIG. 1.4 – Cycle quasi-périodique. La trajectoire est sur un tore. Le mouvement a deux fréquences.

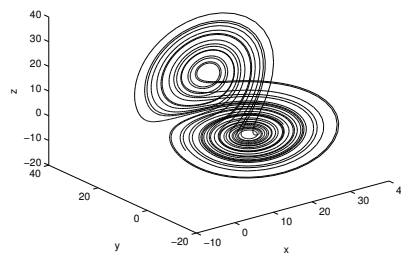


FIG. 1.5 – Attracteur étrange pour le système de Lorentz [Spa82].

1.2.2 Stabilité des cycles

D'une manière intuitive, la stabilité asymptotique d'un système est la propriété que possède un système de revenir à sa solution d'équilibre lorsqu'il en est écarté ponctuellement. En général, la stabilité d'une trajectoire est définie comme :

Définition 8 (*Stabilité d'une trajectoire* [SC64], [Kha01], [Law73]) Soit x une trajectoire de (1.1) associée à la condition initiale (x_0) . Cette trajectoire est dite *stable* si pour tout $\epsilon > 0$, il existe $\delta = \delta(\epsilon) > 0$ tel que pour tout $\bar{x}_0$ vérifiant

$$\|x_0 - \bar{x}_0\| \leq \delta, \text{ on a } \|x(t) - \bar{x}(t)\| \leq \epsilon \quad \text{pour } 0 \leq t \leq +\infty$$

si de plus,

$$\lim_{t \rightarrow +\infty} \|x(t) - \bar{x}(t)\| = 0$$

alors x est dite *asymptotiquement stable*.

Comme la définition précédente est une définition pour une trajectoire quelconque, elle s'applique aussi pour les cycles. Le but de cette partie est de mentionner les outils existants de base pour analyser la stabilité d'un cycle. Pour cela, considérons que x est un cycle du système (1.1). Nous présenterons de manière succincte les deux principales méthodes permettant d'étudier la stabilité d'un cycle périodique du système (1.1) : Les fonctions de sensibilité ou matrice de Monodromie et la section de Poincaré.

1.2.2.1 Les fonctions de sensibilité

Les fonctions de sensibilité permettent d'étudier l'influence d'une petite perturbation des conditions initiales, sur l'évolution temporelle des trajectoires du système (1.1). Pour une trajectoire périodique x , la stabilité de x peut être évaluée au bout d'une période en considérant l'écart entre le cycle x et la trajectoire issue d'une légère perturbation de la condition initiale. Cette approche est détaillée dans [Rei91], [Sey87], [Sey88]. Une application récente en robotique qui évalue la stabilité du cycle avec cette approche est donnée dans [MBSL05].

Soit $\bar{x}$ la trajectoire issue de la condition initiale $x_0 + \delta x_0$. L'évolution temporelle de l'écart entre les trajectoires est donnée par :

$$\delta x(t) = \bar{x}(t) - x(t)$$

En particulier, après une période, en effectuant un développement en série de Taylor autour de x_0 , on aboutit au premier ordre à :

$$\delta x(T_p) = \frac{\partial x(T_p)}{\partial x_0} \delta x_0$$

Nous constatons que l'écart entre les deux trajectoires à $t = T_p$ est lié à l'écart initial δx_0 par la matrice :

$$\frac{\partial x(T_p)}{\partial x_0} = \begin{bmatrix} \frac{\partial x^1}{\partial x_0^1} & \cdots & \frac{\partial x^1}{\partial x_0^n} \\ \vdots & & \vdots \\ \frac{\partial x^n}{\partial x_0^1} & \cdots & \frac{\partial x^n}{\partial x_0^n} \end{bmatrix} \quad (1.5)$$

où x^i et x_0^i ($i = 1, \dots, n$) représentent respectivement les i -ème composantes de x et x_0 .

Des propriétés de stabilité de la matrice (1.5), appelée fonction de sensibilité de x par rapport à l'état initial x_0 ou matrice de Monodromie, on déduira aisément celle de la trajectoire en inférant dans un voisinage de la trajectoire la croissance ou l'amortissement de la perturbation initiale.

Puisque x représente la trajectoire du (1.1) ayant x_0 pour condition initiale, x vérifie également ce système, soit :

$$\frac{dx}{dt} = F(x, t)$$

Différencier cette équation par rapport à la condition initiale x_0 donne alors :

$$\frac{d}{dt} \frac{\partial x}{\partial x_0} = \frac{\partial F(x, t)}{\partial x} \frac{\partial x}{\partial x_0} \quad (1.6)$$

La fonction de sensibilité de x par rapport à l'état initial est la même matrice que la matrice de transition d'état de (1.6), $\Phi(T_p, 0)$, où $\Phi(t, 0)$ vérifie :

$$\frac{d\Phi(t, 0)}{dt} = \frac{\partial F(t, x(t))}{\partial x} \Phi(t, 0), \quad \Phi(0, 0) = \mathbb{I} \quad (1.7)$$

La fonction de sensibilité, de période T_p , de x par rapport à l'état initial x_0 , notée $\mathbf{M} \in \mathbb{R}^n \times \mathbb{R}^n$ est définie par :

$$\mathbf{M} \equiv \Phi(T_p, 0) = \frac{\partial x(T_p)}{\partial x_0}$$

Enfin, la matrice $\mathbf{M}$ possède deux propriétés remarquables :

- $\forall k \in \mathbb{N}, \Phi(kT_p, 0) = \mathbf{M}^k$.
- La matrice de $\mathbf{M}$ possède toujours la valeur $\varrho = 1$ comme valeur propre.

La stabilité est donc déterminée par les valeurs propres de $\mathbf{M}$ qui ont un module différent de 1. Si toutes les valeurs propres sont dans le cercle unitaire le cycle est stable.

1.2.2.2 Les sections de Poincaré

La section de Poincaré est un outil fréquemment utilisé pour étudier les systèmes dynamiques et notamment la stabilité des orbites périodiques [Sey88], [JKN00], [Zhe07].

Au lieu de s'intéresser directement à x , comme il a été fait pour définir la fonction de sensibilité par rapport à l'état initial, nous allons considérer un ensemble de point $p_0, p_1, p_2, \dots$. Considérons donc la définition suivante :

Définition 9 (*Section de Poincaré [PM98]*) Soit une hypersurface Σ_p de dimension $n-1$ et une trajectoire x issue du système (1.1). Considérons l'ensemble des points $p_0, p_1, p_2, \dots$ aux instants $t_0 < t_1 < t_2 < \dots$ correspondant aux intersections successives de la trajectoire x avec Σ_p (en supposant qu'ils existent). Σ_p est appelé une section de Poincaré

On considère alors l'application $\mathbf{T} : \mathbb{R}^{n-1} \rightarrow \mathbb{R}^{n-1}$ transformant un point en son suivant sur la section de Poincaré. On a, pour une trajectoire quelconque, la relation de récurrence suivante :

$$p_k = \mathfrak{T}(p_{k-1}).$$

Pour un cycle périodique du système (1.1), il existe j fini tel que $\mathfrak{T}^j$ est équivalente à l'identité puisque la trajectoire se referme sur elle-même. p_0 est alors un point fixe de l'application $\mathbf{T} = \mathfrak{T}^j$:

$$p_0 = \mathbf{T}(p_0) = \mathbf{T}(\mathbf{T}(p_0)) = \dots$$

Choisissons l'hypersurface Σ_p contenant le point x_0 que l'on dénomme p_0 sur l'hypersurface. Les figures 1.6 et 1.7 représentent des trajectoires dans $\mathbb{R}^3$, l'hypersurface se réduit à un plan de $\mathbb{R}^2$. La figure 1.6 présente un cycle. Ce cycle coupe toujours l'hyperplan au même point p_0 . La section de Poincaré est alors réduite à l'unique point p_0 . La figure 1.7 présente la trajectoire d'une solution non périodique. Dans ce cas, la trajectoire ne suit pas une orbite fermée et coupe l'hypersurface en différents points p_0, p_1, p_2 , etc. Cet ensemble de points forme la section de Poincaré. Dans le cas où un cycle est quasi-périodique de fréquences f_1 et f_2 , la section de Poincaré est une courbe refermée sur elle-même.

Dans le même esprit que pour introduire la fonction de sensibilité $\mathbf{M}$, on peut étudier les écarts, au bout d'une itérée de l'application $\mathbf{T}$ entre les images du point périodique p_0 dont on aurait très légèrement modifié la position initiale sur la section de Poincaré et du point p_0 (voir Figure 1.8) :

$$\delta p = \mathbf{T}(p_0 + \delta p_0) - \mathbf{T}(p_0)$$

Puisque

$$\mathbf{T}(p_0 + \delta p_0) = \mathbf{T}(p_0) + \frac{\partial \mathbf{T}(p_0)}{\partial p_0} \delta p_0 + \mathcal{O}(\delta p_0^2)$$

il vient immédiatement :

$$\delta p = \frac{\partial \mathbf{T}(p_0)}{\partial p_0} \delta p_0 + \mathcal{O}(\delta p_0^2)$$

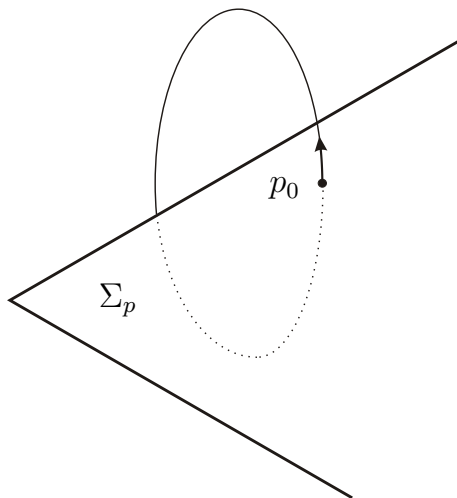


FIG. 1.6 – Section de Poincaré d'une solution périodique. Cycle.

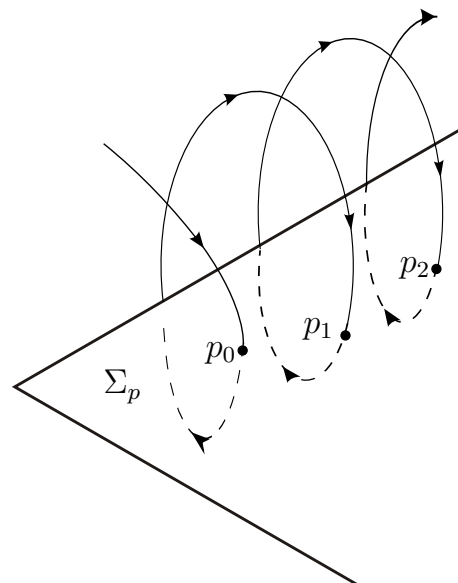


FIG. 1.7 – Section de Poincaré d'une solution non périodique

La matrice

$$\mathfrak{F} = \frac{\partial \mathbf{T}(p_0)}{\partial p_0} \quad (1.8)$$

est appelée matrice de Floquet.

Par la définition de stabilité 8, le cycle est stable si et seulement si $\|\delta p\| < \|\delta p_0\|$, c'est-à-dire :

$$\|\mathfrak{F}\delta p_0\| < \|\delta p_0\| \quad (1.9)$$

Soit ϱ_i , $i = 1, \dots, n - 1$ les valeurs propres de la matrice de Floquet $\mathfrak{F}$. La trajectoire périodique est :

- Asymptotiquement stable si toutes les valeurs propres ϱ_i ont un module strictement inférieur à 1.
- Asymptotiquement instable si au moins une valeur propre de $\mathfrak{F}$ a un module supérieur à 1.

La perte de stabilité de la solution périodique (du cycle) correspond à la traversée du cercle unitaire par une (ou plusieurs) valeur(s) propre(s) de la matrice de Floquet (voir figure 1.9). Lorsque deux valeurs propres complexes conjuguées $\alpha \pm i\beta$ traversent simultanément le cercle unitaire, nous obtenons une bifurcation de Hopf.

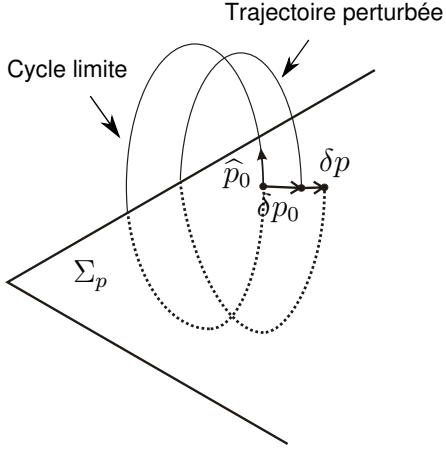


FIG. 1.8 – Cycle et la trajectoire perturbée

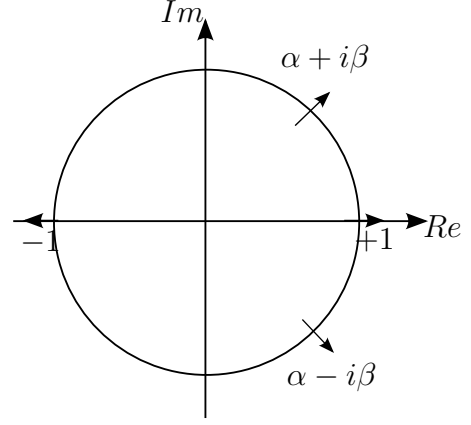


FIG. 1.9 – Différents croisements du cercle unitaire par les valeurs propres de $\mathbf{F}$

1.2.2.3 Calcul pratique de stabilité

Les sections précédentes ont mis en évidence deux manières différentes d’aborder le problème de stabilité locale d’une solution périodique ou plus exactement d’un cycle du système (1.1). Dans [Sey88], il est prouvé que ces deux méthodes sont finalement liées car :

$$\mathfrak{F} = \frac{\partial \mathbf{T}(p_0)}{\partial p_0} = \frac{\partial x(T_p; (t_0, p_0))}{\partial p_0} \quad (1.10)$$

où x est restreinte à $n - 1$ composantes de Σ_p . Comme nous l’avons déjà remarqué à la fin de la section 1.2.2.1, la matrice $\mathbf{M}$ définie en (1.5) a systématiquement une valeur propre égale à 1. En choisissant donc une base appropriée de représentation pour la matrice de $\mathbf{M}$, les $n - 1$ valeurs propres restantes correspondent à celles de la matrice de Floquet $\mathfrak{F}$ définie en (1.8).

Théorème 10 Soit x un cycle de période T_p pour le système (1.1), la matrice de sensibilité $\mathbf{M}$ est encore définie par la matrice de transition $\Phi(T_p, 0)$ où $\Phi(t, 0)$ est solution du problème matriciel aux valeurs initiales suivant :

$$\frac{d\Phi(t, 0)}{dt} = \frac{\partial F(x(t), t)}{\partial x} \Phi(t, 0), \quad \Phi(0, 0) = \mathbb{I} \quad (1.11)$$

Soit ϱ_i , $i = 1, \dots, n$ les valeurs propres de la matrice $\mathbf{M}$ où ϱ_n correspond à la valeur propre égale à 1 de la matrice de sensibilité. Alors,

- Le cycle est localement stable si pour tout $i = 1, \dots, n - 1$, $|\varrho_i| < 1$.
- Le cycle est localement instable s’il existe un i , $1 \leq i \leq n - 1$, tel que $|\varrho_i| > 1$.

1.3 Systèmes hybrides et stabilité des cycles

Dans la section précédente, nous avons passé en revue le comportement asymptotique d'un système non-linéaire variant dans le temps. Nous allons à présent nous intéresser à la classe des systèmes hybrides.

Commençons donc par donner quelques définitions et notions de base.

1.3.1 Définitions. Séquences de temps et exécutions.

D'une manière courte, nous pouvons dire que les systèmes hybrides sont des systèmes dynamiques faisant intervenir explicitement et simultanément des phénomènes ou des modèles de type dynamique continue et discrète.

Une définition plus complète et plus formelle est celle obtenue par l'extension de la théorie des automates à états finis.

Cette modélisation est modulaire. En effet, les règles de compositions entre automates s'appliquent permettant une description sous forme de modules [NOaSY93], [ACH⁺95], [SJSL00] [LJS⁺03].

Tout d'abord, notons $\mathcal{P}(U)$ l'ensemble des parties de U défini par :

$$\mathcal{P}(U) = \{A | A \subseteq U\}$$

Définition 11 (*Système hybride [SJSL00] [LJS⁺03]*) *Un système hybride est un 8-uplet*

$$\mathbf{H} = (\sigma, \mathcal{E}, x, \mathcal{D}, \mathcal{F}, \text{Init}, \mathcal{G}, \mathcal{S})$$

où

- σ est l'ensemble des variables discrètes de $\mathbf{H}$ (mode). $\sigma \in \mathcal{I} = \{1, 2, \dots, m\}$, $m \geq 1$ ($m \in \mathbb{N}$).
- $\mathcal{E} \subseteq \mathcal{I} \times \mathcal{I}$ est un ensemble d'arcs.
- x est l'ensemble des variables continues. $x \in \mathbf{X}$
- $\mathcal{D} = \{D_i : i \in \mathcal{I}\}$ est un ensemble de domaines de l'automate hybride $\mathbf{H}$. $\mathcal{D} : \mathcal{I} \rightarrow \mathcal{P}(\mathbf{X})$.
- $\mathcal{F} = \{F_i : i \in \mathcal{I}\}$ est un ensemble de champs de vecteurs tel que F_i est Lipschitz sur D_i , pour tout $i \in \mathcal{I}$.
- $\text{Init} \subseteq Q \times \mathbf{X}$ est l'ensemble des états initiaux
- $\mathcal{G} = \{G(e) : e \in \mathcal{E}\}$ est l'ensemble de gardes, où $e = (i, j) \in \mathcal{E}$. $\mathcal{G} : \mathcal{E} \rightarrow \mathcal{P}(\mathbf{X})$

- $\mathcal{S} = \{S(e, x) : e \in \mathcal{E}\}$ est un ensemble de resets, où $e = (i, j) \in \mathcal{E}$, S_e .
- $\mathcal{S} : \mathcal{E} \times \mathbf{X} \rightarrow \mathcal{P}(\mathbf{X})$

Hypothèse 1 *Le nombre d'état discret est fini. $\mathbf{X} = \mathbb{R}^n$, $n > 0$. Pour tout $e \in \mathcal{E}$, $G(e) \neq \emptyset$ et pour tout $x \in G(e)$, $S(e, x) \neq \emptyset$.*

L'interprétation de la définition précédente est : A partir d'un point initial qui vérifie *Init* et qui se trouve dans le domaine D_i , le système évolue avec une dynamique F_i tant que $x \in D_i$ (D_i est aussi appelé invariant du mode i). Si un point de la trajectoire est tel que $x \in G(i, j)$, alors un changement de mode est possible vers le mode j . L'état peut alors être discontinu (saut) lors du changement de mode et la valeur de l'état après le saut est donnée par $S((i, j), x)$.

Remarquons que les définitions des gardes et des invariants ne définissent en général pas une trajectoire unique (On dit que l'automate n'est pas déterministe). L'automate hybride permet la représentation d'un ensemble de fonctionnements possibles du SDH. Lorsque les conditions d'invariance et garde conduisent à une trajectoire unique (σ, x) , on dit que l'automate est déterministe. Un automate hybride non déterministe peut le devenir si on ajoute des règles pour commander les commutations quand elles sont possibles. Nous pouvons classer les commutations en deux classes :

1. Les commutations provoquées lorsque $x(t)$ soit de l'invariant D_i sont appelées commutations autonomes.
2. Les commutations provoquées pour une commande externe sont appelées commutations commandées.

Afin d'illustrer la définition 11, considérons un système composé par deux réservoirs et notons x_i le volume d'eau du réservoir i , v_i est un taux constant d'eau sortante du réservoir i , l_i est le niveau du volume désiré pour le réservoir i . Le flux d'eau entrant w ne peut qu'alimenter un seul réservoir. Autrement dit, les deux réservoirs n'ont pas de flux d'eau entrant en même temps. Le but de la commande est de maintenir le niveau de volume au-dessus de l_1 et l_2 par une stratégie qui commute le flux d'entrée w au premier réservoir lorsque $x_1 = l_1$ et au deuxième réservoir lorsque $x_2 = l_2$. Nous avons comme hypothèse que les volumes initiaux sont au-dessus de l_1 et l_2 .

La modélisation de ce système hybride est donc :

- $n = 2$, $m = 2$, $\mathcal{E} = \{(1, 2), (2, 1)\}$
- $x = \{x_1, x_2\}$, $\mathbf{X} = \mathbb{R}^2$
- $D_1 = \{x \in \mathbb{R}^n, l_1 \leq x_1 < \infty\}$, $D_2 = \{x \in \mathbb{R}^n, l_2 \leq x_2 < \infty\}$,

- $F_1 = (w - v_1, -v_2)^T$, $F_2 = (-v_1, w - v_2)^T$,
- $Init = \mathbf{X}$
- $G(1, 2) = \{x \in \mathbb{R}^n : x_2 = l_2\}$, $G(2, 1) = \{x \in \mathbb{R}^n : x_1 = l_1\}$
- $S((1, 2), x) = \{[x_1, l_2]^T\}$, $S((2, 1), x) = \{[l_1, x_2]^T\}$.

La figure 1.10 montre l'automate correspondant à ce système.

Une autre définition importante est celle de la séquence de temps. Elle va nous aider à définir la durée de chaque mode de l'automate hybride $\mathbf{H}$.

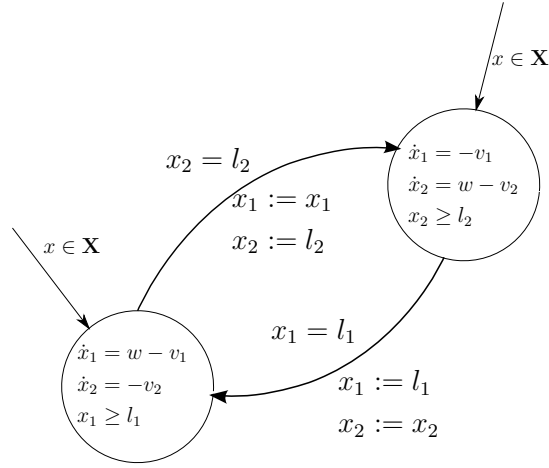


FIG. 1.10 – Automate associé à l'exemple de deux réservoirs d'eau

L'évolution des variables $\sigma(t)$ et $x(t)$ définissent une exécution de l'automate hybride (non unique lorsque l'automate n'est pas déterministe). Pour spécifier complètement une telle exécution, on définit les instants de commutation, l'évolution de l'état continu lorsque l'état discret est fixé, les règles de changement d'état discret et de recalage éventuel de l'état continu lors d'une commutation. Ceci conduit aux définitions suivantes :

Définition 12 (Séquence de temps) Une séquence de temps (finie ou infinie) est un ensemble ordonné $\mathcal{T} = \{t_j\}_{j=0}^s$ de temps tels que $t_{j+1} \geq t_j$ pour tout $j \geq 0$ si la séquence est infinie. Si elle est finie, alors $t_{j+1} \geq t_j$ pour tout $0 \leq j \leq s$.

Les instants t_j sont les temps de commutation. Si $\mathcal{T}$ est une séquence de temps, alors s est la taille de la séquence. Notons $\langle \mathcal{T} \rangle$ l'ensemble $\{0, 1, \dots, s\}$ si la séquence est finie, et $\{0, 1, 2, \dots\}$ si la séquence est infinie.

Définition 13 (Exécution d'un système hybride [SJS00], [LJS⁺03]) Une exécution d'un système hybride $\mathbf{H}$ est un triplet $(\mathcal{T}, \Omega, \mathcal{X})$, où

- $\mathcal{T}$ est la séquence des temps de commutation.

- $\Omega = \{\sigma_j\}_{j=0}^{s-1}$ est un ensemble d'applications appelé la séquence de modes tel que pour tout $j \in \langle \mathcal{T} \rangle$, $\sigma_j : [t_j, t_{j+1}] \rightarrow \mathcal{I}$.
- $\mathcal{X} = \{x^j : j \in \langle \mathcal{T} \rangle\}$, $x^j : [t_j, t_{j+1}] \rightarrow \mathbf{X}$ est un ensemble d'applications C^1 tel que :

1. $(\sigma_0, x^0) \in \text{Init}$
2. Pour tout $t \in [t_j, t_{j+1}]$, $x^j : [t_j, t_{j+1}] \rightarrow D_{\sigma_j}$ et,

$$\dot{x}^j(t) = F_{\sigma_j}(x^j(t)) \quad (1.12)$$

3. Pour tout $j \in \langle \mathcal{T} \rangle$, nous avons $e = (\sigma_j, \sigma_{j+1}) \in \mathcal{E}$, $x^j(t_{j+1}^-) \in G(\sigma_j, \sigma_{j+1})$, et $x^{j+1}(t_{j+1}^+) \in S(e, x^j(t_{j+1}^-))$.

La figure 1.11 montre une exécution d'un système hybride.

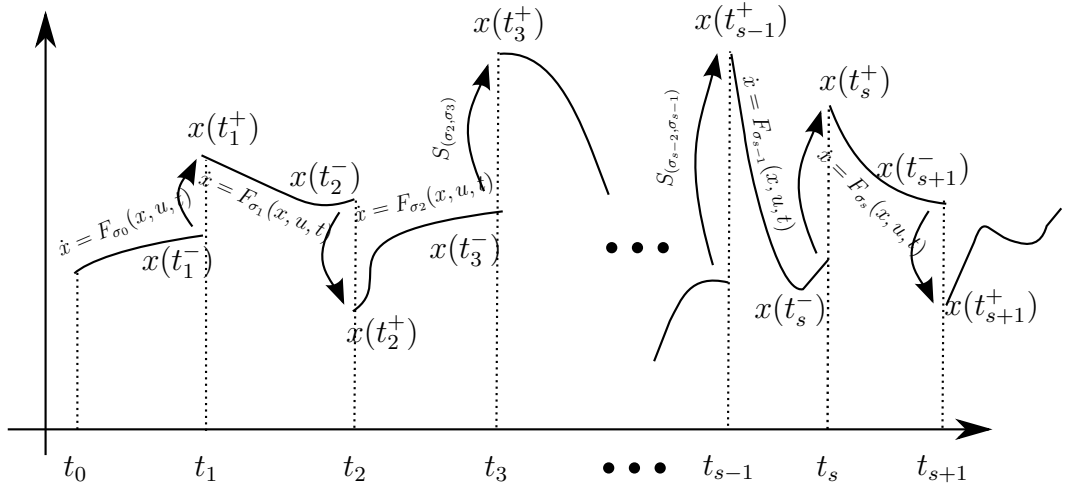


FIG. 1.11 – Exécution d'un système hybride

Notons $\mathcal{T}_f$ le temps total de l'exécution $(\mathcal{T}, \Omega, \mathcal{X})$ tel que $\mathcal{T}_f = t_s - t_0$.

Définition 14 (*Exécution infinie et zeno [SJS00]*) Une exécution est dite :

- *Infinie* si $s = \infty$ ou $\mathcal{T}_f = \infty$
- *Zeno* si $s = \infty$ et $\mathcal{T}_f < \infty$

Un système hybride peut en général être considéré comme un système non-linéaire variant dans le temps. Les solutions d'équilibre sont alors de type point d'équilibre, cycle, attracteur étrange (Voir section 1.2.1).

La définition 3 d'un cycle est assez générale et elle est toujours valable pour les systèmes hybrides. Cependant, son interprétation change un peu [SM01] : Un **cycle** de l'automate hybride $\mathbf{H}$ de période T_p apparaît s'il existe une exécution finie de l'automate hybride $\mathbf{H}$ telle que :

$$x(t_0) = x(t_s) \quad \mathcal{T}_f = T_p \quad (1.13)$$

Par la suite, on dénote l'exécution de l'automate hybride $\mathbf{H}$ qui produit le cycle comme $(\mathcal{T}^\infty, \Omega^\infty, \mathcal{X}^\infty)$ où $\mathcal{T}^\infty = \{t_j^\infty\}_{j=0}^{s^\infty}$ et $\Omega^\infty = \{\sigma_j^\infty\}_{j=0}^{s^\infty-1}$ sont respectivement la séquence de temps de commutation et la séquence de modes nécessaires pour obtenir un cycle. De plus, nous nous intéresserons aux valeurs de l'état aux instants de commutation plutôt qu'à son comportement sur un interval $[t_j, t_{j+1}]$, nous noterons également $\mathcal{X}^\infty = \{x_j^\infty\}_{j=0}^{s^\infty} = \{x(\mathcal{T}^\infty)\}$

Remarquons que dans la séquence de modes

$$\Omega = \{\sigma_0^\infty, \sigma_1^\infty, \dots, \sigma_j^\infty, \dots, \sigma_k^\infty, \dots, \sigma_{s-1}^\infty\}$$

il est possible que $\sigma_j^\infty = \sigma_k^\infty$, pour $j \neq k$. Autrement dit, l'exécution peut avoir le même mode plus d'une seule fois. Lorsque $\sigma_j \neq \sigma_k$ pour tout $j \neq k$, le cycle est dit fondamental.

1.3.2 Résultats de stabilité de cycles pour les systèmes hybrides

Nous abordons, ici, l'étude de la stabilité d'un cycle pour un système hybride. Nous verrons une formulation adaptée aux systèmes hybrides, des concepts de section de Poincaré et fonction de sensibilité. Nous utiliserons également l'outil "fonctions de Lyapunov" pour l'analyse de stabilité des cycles.

Les sections de Poincaré sont un outil fréquemment utilisé pour étudier la stabilité des cycles. Comme nous l'avons déjà mentionné, la section de Poincaré discrétise la trajectoire en considérant l'intersection de la trajectoire avec un hyperplan. Si le cycle est stable, la suite de points obtenue converge vers un point unique et diverge dans le cas contraire.

Cette méthode nécessite une continuité aux conditions initiales dans un voisinage de la trajectoire périodique. Nous ferons donc l'hypothèse que l'automate hybride est un automate hybride continu. Cette hypothèse dit que si une exécution commence proche d'une autre exécution, les deux exécutions restent toujours proches l'une de l'autre.

Notons l'ensemble de toutes les exécutions de $\mathbf{H}$ avec un état initial x_0 et un mode initial σ_0 comme $E_H(\sigma_0, x_0)$.

Définition 15 (*Système hybride continu [LJS⁺03, Définition III.3]*) On dit qu'un système hybride est continu en (σ_0, x_0) si pour toute exécution finie $(\mathcal{T}, \Omega, \mathcal{X}) \in E_H(\sigma_0, x_0)$, et pour tout $\epsilon > 0$, il existe $\delta > 0$, tel que i) pour tout $\tilde{x}_0 \in \text{Init}$, ii) $\|x_0 - \tilde{x}_0\| < \delta$ entraîne : il existe une autre exécution $(\mathcal{T}', \Omega', \mathcal{X}') \in E_H(\sigma_0, \tilde{x}_0)$ tel que $\|\mathcal{T} - \mathcal{T}'\| < \epsilon$, $\sigma_s = \sigma'_s$, $\|\tilde{x}(t'_s) - x(t'_s)\| < \epsilon$.

La principale difficulté est alors d'établir la fonction récurrente $\mathbf{T}$. De même le calcul des fonctions de sensibilité est également possible mais cette fois la nature des points des discontinuités (autonomes ou contrôlées) est à prendre en compte. En effet, on observe que la modification d'une condition initiale entraîne une modification des instants de commutation lorsque l'on considère des sauts autonomes.

Considérons à nouveau x une trajectoire périodique issue de l'évolution du système hybride $\mathbf{H}$ muni de la condition initiale $x(t_0) = x_0$ et soit $\mathbf{T} : \mathbb{R}^{n-1} \rightarrow \mathbb{R}^{n-1}$ la fonction récurrente associée à une surface de Poincaré. Une trajectoire cyclique périodique, x_0^∞ est alors un point fixe de l'application $\mathbf{T}$.

D'après (1.10), observons donc que pour déterminer la stabilité locale du cycle, il faut connaître les valeurs propres de la matrice formée par la fonction de sensibilité de $x(t_0 + T_p, t_0, x_0^\infty)$ à la condition initiale x_0^∞ (dont la restriction à $n-1$ composantes est équivalente au calcul de la matrice de Floquet). Si ces valeurs propres sont à l'intérieur du cercle unitaire, le cycle est stable. Malheureusement, ce calcul est loin d'être facile étant données les discontinuités du système hybride.

Pour mener ces calculs à bien, la définition très générale de l'automate hybride (Définition 11) est insuffisante. Les invariants, leurs frontières et les fonctions F doivent être précisés analytiquement.

En général, nous pouvons classer les différentes méthodes en trois :

1. Une approche où le modèle est écrit par des équations algébriques - différentielles - discrètes dépendants d'un paramètre (DAD, de l'anglais differential-algebraic-discrete). Dans ce modèle, le système hybride est écrit comme un système différentiel sous la forme d'un seul champ de vecteurs et les sauts (gards, resets, etc.) se transforment en contraintes de type algébriques. En effet [HP00], [His01], [HR07] utilisent un modèle DAD pour le calcul de la fonction de sensibilité d'un système hybride et par conséquence, analyser la stabilité du cycle. Une application intéressante sur les régulateurs automatiques de tension est donnée dans [RH05] et pour les robots dans [His01].

2. Une approche qui considère le système hybride sous la forme :

$$\dot{x}(t) = F_{\sigma_j}(x(t)) \quad t \in [t_j, t_{j+1}], \quad \forall j = 0, 1, \dots, \quad (1.14)$$

où chaque temps de commutation t_j de la séquence de temps $\mathcal{T}$ est donné par la contrainte :

$$g_{\sigma_j}(x(t^-), t^-) = 0 \quad (1.15)$$

$g_{\sigma_j} : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}$ est une fonction en C^1 . Observons que cet ensemble correspond aux gardes de la définition 11. Les resets sont représentés par des fonctions de saut $\varsigma_j : \mathbb{R}^n \rightarrow \mathbb{R}^n$ de la forme :

$$x(t_j^+) = \varsigma_j(x(t_j^-)) \quad (1.16)$$

Dans cette approche, les auteurs proposent de calculer la matrice de Floquet [Sey88], [PC89] avec la proposition suivante :

Proposition 16 *Soit un cycle de période T_p décrit par l'exécution $(\mathcal{T}^\infty, \Omega^\infty, \mathcal{X}^\infty)$. La matrice de Floquet $\mathfrak{F}$ est donnée par $\mathfrak{F} = \mathfrak{F}_{s^\infty-1} \mathfrak{F}_{s^\infty-2} \dots \mathfrak{F}_1 \mathfrak{F}_0$, où*

$$\mathfrak{F}_j = \left(\mathbb{I} - \frac{F_{\sigma_j^\infty}(x_j^\infty) \eta^T}{F_{\sigma_j^\infty}^T(x_j^\infty) \eta} \right) \Phi_j(t_{j+1}^\infty, t_j^\infty), \quad \forall j = 0, 1, \dots, s^\infty - 1 \quad (1.17)$$

η est un vecteur normal à la section de Poincaré Σ_p et $\Phi_j(t_{j+1}^\infty, t_j^\infty)$ est la matrice de transition de l'état x_j^∞ à x_{j+1}^∞ définie par

$$\begin{cases} \frac{d\Phi_j(t, t_j)}{dt} = \frac{\partial F_{\sigma_j}}{\partial x} \Phi_j(t, t_j), & t_{j+1} \leq t \leq t_j \\ \Phi_j(t_j, t_j) = \mathbb{I} \end{cases} \quad (1.18)$$

Le cycle est alors stable si les modules des valeurs propres de la matrice sont à l'intérieur du cercle unitaire.

La difficulté de cette approche vient évidemment de la connaissance a priori du cycle.

3. Une autre approche existante est celle présentée dans [FRL06] qui a comme avantage de permettre d'obtenir explicitement les temps de commutation (et donc le cycle), à partir de la connaissance a priori de la séquence des modes.

La modélisation du système hybride est réalisée par l'ensemble d'équations (1.14), (1.15) et (1.16). A partir d'une séquence de modes $\sigma = [\sigma_0, \sigma_1, \dots, \sigma_{s-1}]^T$ et des contraintes algébriques associées (1.15), $\mathfrak{G} = [g_{\sigma_0}, g_{\sigma_1}, \dots, g_{\sigma_s}]^T$, permettant les transitions entres ces modes, une détermination du cycle (i.e. la condition initiale x_0 et les temps de commutation) est obtenue rapidement grâce à un algorithme de Newton :

$$\begin{bmatrix} x(t_0) \\ \mathcal{T} \end{bmatrix}_{h+1} = \begin{bmatrix} x(t_0) \\ \mathcal{T} \end{bmatrix}_h - \mathcal{J}^{-1} \begin{bmatrix} x(t_s) - x(t_0) \\ \mathfrak{G} \end{bmatrix}_h \quad (1.19)$$

où h indique les itérations effectuées par l'algorithme et $\mathcal{J}$ désigne le jacobien,

$$\mathcal{J} = \begin{bmatrix} \frac{\partial x(t_s)}{\partial x(t_0)} - \mathbb{I} & \frac{\partial x(t_s)}{\partial \mathcal{T}} \\ \frac{\partial \mathfrak{G}}{\partial x(t_0)} & \frac{\partial \mathfrak{G}}{\partial \mathcal{T}} \end{bmatrix} \quad (1.20)$$

Cet algorithme utilise explicitement les fonctions de sensibilité vis-à-vis de x_0 et des instants de commutation.

Si l'algorithme de Newton converge, alors $x(t_0) \rightarrow x_0^\infty$, $\mathcal{T} \rightarrow \mathcal{T}^\infty$ et par conséquence, l'exécution $(\mathcal{T}^\infty, \Omega^\infty, \mathcal{X}^\infty)$ est facilement construite. En outre, la stabilité locale du cycle est directement déduite à partir des valeurs propres de la matrice $\mathcal{J}^\infty$ car les valeurs propres de $\mathcal{J}$ tendent vers les valeurs propres de $\mathfrak{F}$.

Il existe aussi d'autres approches pour calculer la fonction de sensibilité, mais toujours basées sur les mêmes idées et dédiées plutôt aux applications très spécifiques. Par exemple, [Gal05] montre la stabilité du cycle d'un circuit appelé Chua. [HSSR03], [EADMS07] montre une application pour un convertisseur multi-niveaux et [FA02] pour des convertisseurs qui utilisent comme commande une MLI.

1.3.2.1 Stabilité de Lyapunov

Les méthodes présentées dans la section 1.3.2 basées sur les fonctions de sensibilité et les sections de Poincaré vérifient la stabilité locale du cycle. Autrement dit, elles garantissent que le cycle est toujours stable pour une petite région dans son voisinage. Cependant, ce voisinage n'est pas connu. L'intérêt des approches de type Lyapunov est de chercher à déterminer la région dans l'espace d'état (bassin d'attraction du cycle) pour lesquelles on peut garantir la stabilité du cycle.

Dernièrement, il a eu un grand intérêt pour appliquer les mêmes concepts aux systèmes hybrides [JR98], [DRI02], [BNT04], [GC06]. Branicky dans [Bra98] définit la notion de fonction de Lyapunov multiple et donne des conditions de stabilité pour un système hybride. Cette notion a été reprise par divers auteurs pour définir un ensemble de conditions LMI qui sont suffisantes pour garantir la stabilité de ces systèmes [GC06], [LW04], [DRI02], [Joh02]. Dans [Bra98], il montre également une extension pour déterminer l'existence d'un cycle pour un système hybride avec 2 variables d'état. Pour les mêmes systèmes (2 variables d'état), [SJM02] a eu une idée similaire pour déterminer l'existence des cycles. Dans son approche, il est possible de différencier entre un point d'équilibre, un état zeno et un cycle limite.

La difficulté d'adapter les approches de Lyapunov à la stabilité des cycles est toujours un sujet de recherche. En effet, les approches existantes à l'heure actuelle ne considèrent que des hyperplans comme surfaces de commutation et aucune fonction de reset. Ce sont des hypothèses assez restrictives pour l'analyse de stabilité.

L'idée est donc : trouver une famille de LMI (Linear Matrix Inequalities) qui puisse déterminer des matrices de Lyapunov et établir la stabilité d'un cycle. A partir de cette idée, [RL00] propose une approche intéressante pour les systèmes commutés affines du type

$$\dot{x}(t) = A_{\sigma_j}x(t) + B_{\sigma_j}, \quad j = 1, 2, \dots$$

dont les surfaces de commutation sont des hyperplans, alors :

$$g_{\sigma_j}(t, x(t)) = C_{\sigma_j}x(t) + D_{\sigma_j}.$$

Cette approche prend comme hypothèse qu'une variation de la condition initiale implique une variation des temps de commutation $\mathcal{T}^\infty$. Alors, un intervalle du type $[t_j^\infty - \Delta_j^-, t_j^\infty + \Delta_j^+]$ est associé à chaque temps de commutation ($\forall j = 0, \dots, s^\infty$ où Δ_j^- et Δ_j^+ sont des valeurs constantes positives réelles). L'approche donne des conditions permettant de conclure si le cycle est toujours stable pour ces intervalles de temps.

Une autre approche intéressante est celle présentée dans [Gon03], il s'agit de trouver la région sur l'espace d'état où le cycle est toujours stable. Dans cette méthode, les résultats reposent également sur le principe de Lyapunov classique où les fonctions de Lyapunov sont évaluées sur *des sections d'impact* [GMD03]. Des conditions similaires à celles de [RL00] sont alors déduites afin de trouver les intervalles de temps et d'après ces intervalles, les régions sur l'espace d'état dans lesquelles le cycle est stable sont déduites sous la forme d'ellipsoïdes.

1.4 Problématique de commande des systèmes commutés cycliques

Nous nous intéressons à présent à une classe particulier des systèmes hybrides, dit systèmes commutés dont le comportement asymptotique issu d'une loi de commande est un cycle. Nous avons les définitions suivantes :

Définition 17 (*Système commuté*) *Un système commuté est un système hybride dont les commutations sont toujours commandées et il n'existe pas de sauts sur la variable continue. Autrement dit, $G(e) = \{x \in \mathbf{X}\}$ et $S(e, x) = x$ pour tout $e \in \mathcal{E}$.*

Définition 18 (*Système commuté cyclique*) *On appelle un système commuté cyclique un système commuté qui converge vers un cycle presque - périodique.*

De nombreuses applications réelles entrent dans la catégorie des systèmes commutés cycliques. Un bon exemple est les convertisseurs statiques.

1.4.1 Classe d'applications : les convertisseurs statiques

Les dispositifs électrotechniques utilisent généralement des semi-conducteurs capables d'opérer sur des laps de temps très courts un grand nombre de commutations. Ces commutations modifient la topologie des circuits et par conséquent leurs dynamiques. Nous pouvons donc considérer ces circuits comme une classe particulière de systèmes hybrides.

Quelques exemples qui nous serviront tout au long de ce mémoire et plus particulièrement au chapitre 4, sont montrés de la figure 1.12 à la figure 1.15.

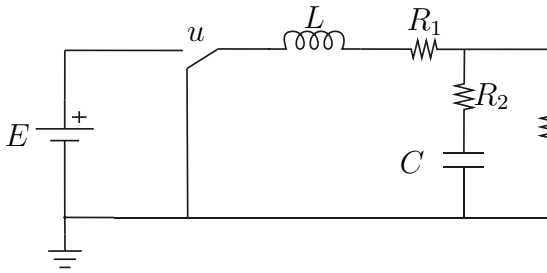


FIG. 1.12 – Convertisseur step-down

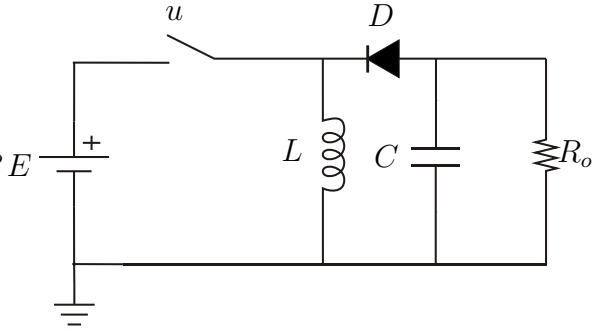


FIG. 1.13 – Convertisseur buck-boost

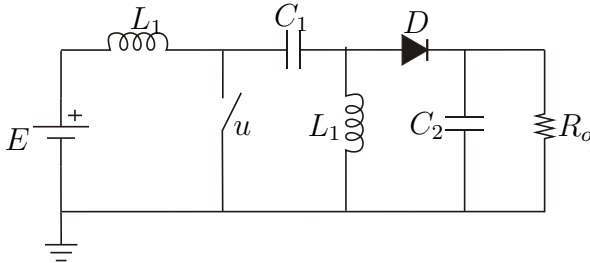


FIG. 1.14 – Convertisseur sepic

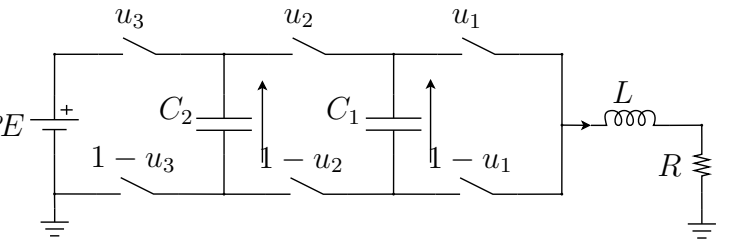


FIG. 1.15 – Convertisseur multiniveaux

Tous ces dispositifs peuvent être considérés comme des systèmes commutés (il n'y a pas de saut sur l'état). Nous nous limiterons donc à cette classe. Chaque mode, donné en fonction de la configuration des interrupteurs, est un système affine en l'état lorsque les éléments qui composent le circuit sont linéaires (Inductances, capacités, résistances, etc) :

$$\dot{x}(t) = F_{\sigma_j(x(t))} = A_{\sigma_j}x(t) + B_{\sigma_j}, \quad j = 1, 2, \dots \quad (1.21)$$

où $A_{\sigma_j} \in \mathbb{R}^{n \times n}$ est une matrice et B_{σ_j} un vecteur.

Prenons l'exemple du convertisseur de la figure 1.12, si $\sigma_j = 1$ (mode 1) le commutateur du circuit est ouvert, par contre si $\sigma_j = 2$ (mode 2) le commutateur

est fermé. Si on assigne à u , la valeur 0 lorsque le commutateur est ouvert et la valeur 1 lorsque le commutateur est fermé, alors le système peut s'écrire sous la forme :

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t), \quad u \in \{0, 1\} \quad (1.22)$$

Plus généralement, dans [BCR02] et [BCZ⁺04], il est démontré à l'aide de la modélisation sous forme de Bond Graph que pour des systèmes physiques avec des commutateurs considérés comme idéaux, cette classe de systèmes peut s'écrire sous la forme affine en la commande suivante :

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t) \quad (1.23)$$

où $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$, $g(x) = [g_1(x), \dots, g_r(x)] \in \mathbb{R}^{n \times r}$, $g_i : \mathbb{R}^n \rightarrow \mathbb{R}^n$, $i = \{1, \dots, r\}$, $u(t) \in U = \{0, 1\}^r$. Les champs f et g sont linéaires en x et la commande u est un vecteur de Booléens.

La représentation (1.23) a comme avantage de s'écrire sous la forme d'un seul champ de vecteurs. Les commutations sont directement liées à la commande u de manière explicite.

Par la suite, nous utiliserons en fonction de la facilité de manipulation et suivant le contexte l'un ou l'autre de ces deux modèles (1.21), (1.23).

Un des objectifs d'une loi de commande est de stabiliser les variables d'état sur un état d'équilibre x_{ref} . Observons d'après (1.21) que les points d'équilibres de chaque sous système n'ont aucune raison d'être communs. C'est une caractéristique de ces systèmes que nous formalisons à la prochaine section.

1.4.2 Points de fonctionnement

Considérons le modèle affine à la commande (1.23) et notons ces points de fonctionnement comme x_{ref} . Pour les systèmes exhibant un comportement cyclique en régime permanent, les points de fonctionnement sont généralement définis par la valeur moyenne de x sur le cycle. Afin d'obtenir l'ensemble des points de fonctionnement, on peut utiliser le modèle dynamique associé à l'état moyen $\bar{x}$ où $\bar{x}$ est donné par le produit de convolution :

$$\bar{x}(t) = \square_{T_p} * x(t) = \frac{1}{T_p} \int_{t-T_p}^t x(\tau) d\tau \quad (1.24)$$

avec T_p la période du cycle et $\square_{T_p}$ une fonction fenêtre rectangulaire.

Afin de déterminer le modèle dynamique de $\bar{x}$, une possibilité est de dériver (1.24) par rapport au temps t . Cependant, ce modèle dynamique est généralement inexploitable car fortement non-linéaire.

Une solution consiste à définir le modèle d'état moyen comme suit :

$$\dot{\tilde{x}}(t) = f(\tilde{x}(t)) + g(\tilde{x}(t))\bar{u}(t), \quad \bar{u} \in co(U) \quad (1.25)$$

où $co(U)$ est l'enveloppe convexe de U . Le modèle (1.25) est une approximation de la dynamique de $\bar{x}$ où $\bar{u}$ représente la valeur moyenne de u sur le cycle. Il est en effet prouvé que la valeur de $\tilde{x}$ tend vers la valeur de $\bar{x}$ et x lorsque T_p est petit par rapport à la dynamique du système [SV85] ($\tilde{x} \rightarrow \bar{x}$ et x si $T_p \rightarrow 0$).

Les points de fonctionnement sont alors définis comme les points d'équilibre du modèle d'état moyen i.e. :

$$X_{ref} = \{x_{ref} \in \mathbb{R}^n : f(x_{ref}) + g(x_{ref})u_{ref} = 0, u_{ref} \in co(U)\} \quad (1.26)$$

Par conséquence, il existe des points d'équilibre x_{ref} dont la commande $u_{ref} \in co(U) \setminus U$. Comme aucune commande u ne permet de maintenir le système sur la référence x_{ref} , lorsque $u_{ref} \in co(U) \setminus U$, le système ne peut être que maintenu dans un voisinage de la référence x_{ref} avec une commande proche en valeur moyenne de u_{ref} .

En effet, il est possible d'approcher x_{ref} d'aussi près que l'on veut avec une commande $u \in U$ en vertu des théorèmes de densité en temps infini [ISW03].

Théorème 19 (Voir [ISW03, Theorem 1].) *Les solutions au sens de Carathéodory de (1.23) sont denses dans l'ensemble des solutions obtenues en relaxant U à son enveloppe convexe $co(U)$ au sens suivant : Soient x une solution globale du système (1.23) avec un domaine de commande étendu à son enveloppe convexe $co(U)$ et partant de x_0 et $\varepsilon : [0, +\infty) \rightarrow (0, +\infty)$ une fonction continue. Alors il existe une solution ξ de (1.23) partant de $\xi_0 \in B(x_0, \varepsilon(0))$ telle que $\|\xi(t) - x(t)\| < \varepsilon(t)$ pour tout $t \in [0, +\infty)$.*

1.4.3 Objectifs de la commande

A la section 1.3.2, nous avons présenté un certain nombre de méthodes (sections de Poincaré, fonctions de sensibilité et fonction de Lyapunov) permettant de mener une analyse de stabilité sur un cycle. Ces méthodes supposent toutes plus au moins implicitement que ce cycle est connu ou presque (séquence des modes au minimum pour les fonctions de sensibilité).

Nous venons également de voir les raisons justifiant l'existence de comportement cyclique en régime permanent pour une classe de systèmes commutés. La

nature du point de fonctionnement et l'impossibilité de s'y maintenir étant les principales.

La problématique de la commande associée à ces systèmes est double : D'une part il faut définir une stratégie de ralliement vers le point de fonctionnement (régime transitoire) et d'autre part il faut s'y maintenir dans un voisinage (régime permanent). Pour ce second objectif, il est clair que de nombreux degrés de liberté existent sur la façon d'opérer. Au chapitre 2, nous privilégierons la recherche d'un cycle particulier établi sur la base d'un critère. Bien sûr des objectifs de robustesse sont à associer aux deux problématiques sus mentionnées.

1.5 Conclusions

Ce chapitre a d'abord présenté différentes techniques pour analyser la stabilité d'un cycle pour les systèmes non-autonomes. Ensuite, nous avons abordé le cas des systèmes hybrides. Une revue de méthodes existantes pour étudier la stabilité des cycles pour ce type de système est exposée. Nous avons également souligné la spécificité de la problématique de commande qui existe pour la sous classe des systèmes commutés.

Ces systèmes sont représentés par des modèle linéaire par morceaux (1.21) et (1.23).

L'étude des points de fonctionnement montre que la commande assurant le maintien sur un point d'équilibre n'appartient pas au domaine original de commande. Cependant, un résultat de densité en temps infini permet de garantir des solutions arbitrairement proches.

Dans les prochains chapitres, les travaux présentés abordent le problème du pilotage de ces systèmes et nous essayerons de proposer des méthodes systématiques pour commander ces systèmes commutés cycliques (1.21).

Chapitre 2

Commande prédictive des systèmes commutés cycliques

Résumé. Les lois de commande classiques pour les systèmes cycliques à base de modèle moyen sont satisfaisantes une fois le cycle atteint, elles permettent de régler le régime transitoire de manière plus ou moins satisfaisante vers un cycle dont le comportement en régime permanent est mal maîtrisé (forme d'onde, sous-harmonique, etc.). Ce chapitre présente une nouvelle approche prédictive qui permet d'atteindre un cycle optimal périodique, à partir de la référence moyenne sur l'état. Cette méthode offre la possibilité de mieux spécifier le régime permanent souhaité. La trajectoire optimale de ralliement est déterminée sur la base d'un critère et utilisée comme référence pour la synthèse d'une commande prédictive. L'implantation en temps réel est assurée par un réseau de neurones.

Mots clés. Cycle limite, fonctions de sensibilités, programmation non-linéaire, convertisseur buck-boost.

2.1 Introduction

Le concept de commande prédictive (MPC pour Model Predictive Control) s'appuie sur la possibilité de prédiction d'un modèle dynamique. Il peut se résumer par : "utiliser un modèle dynamique pour prédire le comportement futur du système compte tenu de la connaissance actuelle de l'état du système et en déduire la meilleure décision à appliquer au sens donné par un certain critère, tout en respectant un ensemble de contraintes sur l'état et la commande".

Les termes de critère et de contraintes sont familiers dans le langage de l'optimisation. Il s'agit donc d'une utilisation particulière qui s'adresse à l'optimisation des systèmes dynamiques. Cette idée apparaît déjà entre les lignes dans le cadre des travaux sur la commande optimale [Bel03].

Le terme “commande prédictive” ne fait pas référence à une commande spécifique, mais plutôt à un ensemble de méthodes suivant la nature et le type de modèle de représentation considéré. L’algorithme permettant d’élaborer la commande reste néanmoins le même indépendamment du modèle et des considérations faites sur le système.

Comme dans toutes les méthodes de commande, il existe des avantages et des inconvénients liés à ces idées [CB05]. Parmi les avantages de la commande prédictive, nous pouvons mentionner :

- Un concept intuitif ne nécessitant pas des connaissances approfondies en commande et une relative simplicité de réglage.
- Un caractère générique car elle peut être appliquée sur une grande variété de processus (à retard, à non minimum de phase, instable,...).
- Un caractère multi-variable par essence.
- Une méthode adaptée pour résoudre le problème de poursuite de trajectoires.
- La facilité de prendre en compte des contraintes sur l’état et également, des contraintes sur la variable de commande.

Et parmi les inconvénients :

- La charge de calcul. A chaque pas de temps, la méthode nécessite de résoudre un problème d’optimisation.
- La nécessité d’un modèle fiable du système car l’algorithme de commande est toujours basé sur ce modèle. La performance de la commande sera alors affectée par les différences entre le modèle et le système. C’est le plus grand inconvénient.

Les différents algorithmes de commande prédictive existants ont été développés en fonction du type de systèmes à commander, du bruit, du critère à minimiser et des contraintes. C’est avec les systèmes échantillonnés que la commande prédictive rencontre son premier succès avec les travaux de Cutler et Ramakter et leur méthode de commande à matrice dynamique (DMC). En effet, dans l’article [CR80] un modèle explicite est utilisé pour prédire l’effet futur des actions de commandes tout en optimisant une prédiction de l’erreur entre une référence et la sortie. Ces formulations ont donné les idées de base de la commande prédictive. En outre, puisque le principe est très simple, la méthode a eu un grand succès particulièrement dans les industries chimiques et pétrolières [GPM89].

Parallèlement, une autre ligne indépendante connue, comme la commande adaptative, commence à apparaître dans les années quatre vingt pour des systèmes mono variables. Un exemple de commande adaptative est la commande

développée par Peterka [Pet84]. Il s'agit de minimiser la valeur espérée d'un critère quadratique des sorties prédites sur un horizon donné.

Plus tard, avec un grand succès, Clarke fusionne les idées de la commande prédictive et de la commande adaptative dans la méthode de Commande Prédictive Généralisée (GPC) pour des modèles à base de fonctions de transfert [Cla88], [CMT87]. Cette méthode fournit une solution analytique et introduit un poids sur l'incrément de la commande dans le critère [CB05], [BD96]. Des considérations de bruits sont aussi prises en compte. Demircioglu et Gawthrop réalisent ensuite une extension de la méthode GPC pour des systèmes continus monovariables [Dem91] et puis, dans les années suivantes ils présentent une généralisation pour le cas multivariable [DG92], [GDSA98], [RAG99]. Leur méthode est connue comme la commande prédictive généralisée en temps continu (CGPC) où la seule différence avec le GPC est la manière de réaliser la prédiction. A l'heure actuelle, il existe des CGPC qui considèrent la saturation ainsi que des retards [DITH04].

La commande prédictive est aussi formalisée pour des systèmes modélisés dans l'espace d'état. Ceci permet une extension directe au cas des systèmes multivariables non linéaires et stochastiques. Un des avantages de la modélisation dans l'espace d'état est de pouvoir utiliser des techniques d'estimation et de filtrage de l'état pour réaliser la prédiction [LMG94]. La commande prédictive est donc interprétée comme une commande basée sur un estimateur d'état dont les paramètres de l'estimateur (pôles, matrices de gains, etc.) et les paramètres du régulateur (horizon de prédiction, matrices de poids, etc.) déterminent la stabilité, performance et robustesse du système en boucle fermée.

A l'heure actuelle ce type de formulation est toujours un sujet d'étude notamment lorsqu'on ajoute un filtre de Kalman [RN94], [HB04], [MRFA06].

A partir de la formulation dans l'espace d'état, des résultats importants concernant la robustesse vis-à-vis d'incertitude de modèle ont eu un grand intérêt scientifique. L'idée est de prendre en compte les paramètres incertains du système et d'obtenir une loi de commande prédictive dont le critère optimise la pire situation possible donnée par les paramètres incertains. Plusieurs approches qui utilisent des fonctions de Lyapunov ont été déjà développées dans ce domaine [Bem98], [BM99a], [BM99b], [MH07b].

Pour les systèmes commutés, la commande prédictive a déjà été utilisée [GPM05], [LK04], [Ala06], [CA05]. Ces approches emploient la programmation dynamique. L'approche de commande prédictive la plus employée pour trouver des exécutions minimisant est celle proposée par Bemporad et Morari [BM99a]. Ils adoptent une formulation MLD (Mixed Logical Dynamical) pour décrire le problème d'optimisation (2.5) où les expressions discrètes (la commande u) s'expriment comme des

contraintes algébriques par l’algèbre booléenne. En conséquence, des problèmes de programmation entière mixtes apparaissent (MIP). Ce type de méthodes utilise d’habitude la technique de “branch and bound” pour trouver la commande [HSB01]. Cependant, les inconvénients d’une formulation MLD sont, comme la plupart des méthodes de commande prédictive le temps de calcul. [BFM00], [BBM03] ont créé une méthode basée sur le calcul hors ligne d’une table et de son application en ligne pour pouvoir réduire le temps de calcul. Nous adoptons aussi l’idée du calcul hors-ligne pour la méthode présentée dans ce chapitre.

Au niveau pratique, il existe des nombreuses utilisations de la commande prédictive. Par exemple, [AAaSI07] montre une commande pour un moteur d’eau, [CCC⁺07] une commande d’un moteur turbo-chargé et [BB06], [SMQG07], [BPFM07] présentent des applications sur les convertisseurs statiques. Bien que la plupart de ces systèmes aient un caractère cyclique en régime permanent, aucune des méthodes n’a cherché à exploiter cette propriété.

L’objectif de ce chapitre est précisément de présenter une méthode de commande prédictive adaptée aux systèmes commutés ayant un régime permanent cyclique.

Nous débuterons en présentant le principe de la commande prédictive appliquée dans le cadre général des systèmes échantillonnés.

Nous abordons ensuite à la section 2.3, la formulation du problème de commande dans le cadre spécifique des systèmes commutés ayant un régime permanent cyclique. Nous proposons alors d’appliquer une stratégie différente de résolution en fonction de la distance à la référence permettant de distinguer le transitoire, du régime permanent.

La synthèse de la référence sous la forme d’un régime permanent cyclique est effectuée hors-ligne et peut être vue comme le résultat d’une commande en boucle ouverte. Cette référence utilisée comme trajectoire de ralliement pour la commande en boucle fermée, permet de contrôler efficacement le comportement du système autour du point de fonctionnement. Cette partie sera développée dans la section 2.4.

Lorsque le régime permanent est a priori connu, le problème peut se réduire en la détermination des séquences de temps de commutation pour pouvoir le rejoindre. Aussi dans la section 2.4, nous verrons une méthode qui nous permet de déterminer ces séquences à partir des fonctions de sensibilités et d’un algorithme de Gauss-Newton.

Néanmoins, au niveau de la mise en oeuvre, l'algorithme peut devenir très coûteux en temps de calcul. Pour pallier cet inconvénient, nous proposons d'interpoler par un réseau de neurones les solutions afin de déterminer une simple relation entrée (l'état) - sortie (la commande). La loi de commande par retour d'état résultante est alors facilement implémentable et peu coûteuse. Nous verrons cette partie dans la section 6.

A titre d'exemple, on applique les résultats obtenus sur le cas d'un convertisseur simple, le convertisseur buck-boost.

2.2 Principe de la commande prédictive

Par simplicité, nous considérons le cas d'un système non-linéaire échantillonné invariant dans le temps dont l'équation d'état est la suivante :

$$x_{k+1} = F(x_k, u_k) \quad (2.1)$$

où $x_k \in \mathbb{R}^n$ représente l'état du système à l'instant t_k , $F : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n$ est une fonction continue et $u_k \in \mathbb{R}$ est la commande appliquée à l'instant t_k .

Nous supposons que tous les états du système (2.1) sont mesurés. Nous ne prendrons pas en compte les bruits de mesure ainsi que d'éventuels bruits de quantification et incertitudes paramétriques.

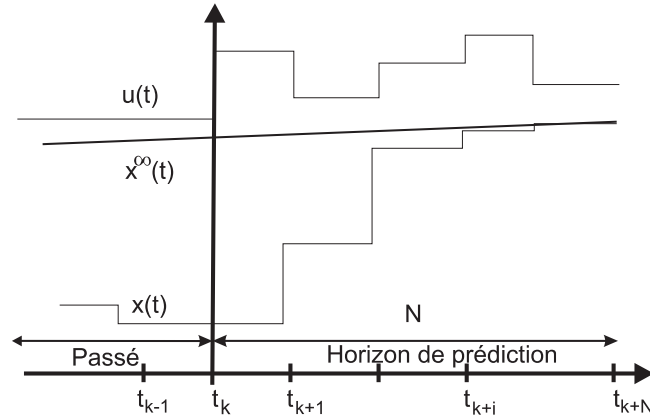


FIG. 2.1 – Diagramme temporel de la prédiction à horizon fini

L'algorithme permettant d'élaborer une commande prédictive se décompose de la manière suivante (Voir la Figure 2.1) :

1. **Prédiction** : A partir de la mesure de l'état x_k à l'instant t_k , et pour une suite de commandes donnée $u_k, u_{k+1}, \dots, u_{k+N-1}$, une prédiction de l'évolution future de l'état x sur un horizon de longueur N , est obtenue par la récurrence (2.1). Soient $x_{k+1}, \dots, x_{k+N}$ ces valeurs prédites.

2. **Optimisation de la commande** : On peut alors envisager à partir de l'estimation du point 1 de déterminer la suite de commande $u_k^*, u_{k+1}^*, \dots, u_{k+N}^*$ qui minimisent un critère J . Le choix du critère dépendra de l'application et du cahier des charges. Par exemple, prenons le cas particulier d'un suivi de trajectoire dont le critère est de type quadratique (celle-ci est un des critères les plus utilisés et forme la base de la méthode de Commande Prédictive Fonctionnelle [BD96]). La fonction à minimiser s'écrit sous la forme :

$$\min_{\{u_k, u_{k+1}, \dots, u_{k+N-1}\}} \sum_{i=1}^N (x_{k+i} - x_{k+i}^\infty)^T Q (x_{k+i} - x_{k+i}^\infty) \quad (2.2)$$

où x_k^∞ est la référence connue aux instants t_k , $Q = Q^T > 0$ est une matrice de poids et N est l'horizon de prédiction. Ces deux paramètres permettent de régler les performances de la commande résultante et sont donc considérés comme des paramètres de réglage.

3. **Mise à jour** : On applique uniquement la commande u_k^* , à l'instant t_k et on recommence le processus depuis le point 1 en décalant d'une unité la fenêtre de prédiction ($k \leftarrow k + 1$) et à partir d'une nouvelle mesure de l'état, x_{k+1} .

Les phases 1 à 3 sont plus au moins raffinées suivant les hypothèses faites sur le problème et les méthodes utilisées. Par exemple, la mesure de l'état complet x_k n'est pas nécessaire et on peut envisager l'utilisation d'un estimateur d'état à l'étape 1 pour estimer x_k comme un filtre de Kalman ou un filtre adaptatif de type Alien [FJSR08].

Le critère, on l'aura compris joue un rôle important dans le réglage de la dynamique. Il est fréquent de considérer un terme de pénalisation sur les incréments de commande à la fonction (2.2). La fonction à minimiser devient alors :

$$\min_{\{u_k, u_{k+1}, \dots, u_{k+N-1}\}} \sum_{i=1}^N (x_{k+i} - x_{k+i}^\infty)^T Q (x_{k+i} - x_{k+i}^\infty) + \sum_{i=1}^{N-1} \Delta u_{k+i}^T R \Delta u_{k+i} \quad (2.3)$$

où $R > 0$ est une matrice de poids et Δu_{k+i} l'incrément de commande défini par $\Delta u_{k+i} = u_{k+i} - u_{k+i-1}$ pour $i = 1, \dots, N - 1$. On utilise les incréments de commande Δu_k plutôt que les commande u_k car les termes de commande provoqueraient sur une référence constante un biais non-nul en régime permanent et par conséquent une erreur statique.

La formulation (2.3) est largement employée dans le cas où le système est mal identifié puisque l'absence de pénalisation peut entraîner une instabilité de

la boucle de commande (par exemple, avec des zéros mal identifiés). Cette formulation permet également d'éviter des phénomènes de grandes oscillations entre deux échantillons grâce à la pénalisation des variations fortes de commande.

Un autre aspect important de la commande prédictive est sa flexibilité pour prendre en compte des contraintes égalités ou inégalités comme fonctions de l'état, du temps et de la commande. D'une façon générale, ces contraintes peuvent s'exprimer sous la forme :

$$G(x_k, u_k, t_k) \geq 0 \quad (2.4)$$

Le problème d'optimisation (résolution de (2.2) ou (2.3) sous les contraintes (2.1)) déterminé au point 2 de l'algorithme précédent s'effectue en tenant compte de ces contraintes supplémentaires (2.4).

2.3 Formulation de problème de commande prédictive pour les systèmes commutés cycliques

Nous avons présenté le principe de la commande prédictive dans le cadre des systèmes échantillonnés modélisés par l'équation (2.1).

La problématique qui nous intéresse est l'application de cette méthode aux systèmes ayant un régime permanent cyclique dont le modèle s'écrit généralement sous la forme d'un système commuté.

2.3.1 Formulation du problème

D'après la formulation de la commande prédictive et la définition d'un système commuté affine, l'algorithme de commande prédictive nécessite de déterminer à chaque pas de temps une exécution optimale (la séquence de modes Ω , la séquence de temps $\mathcal{T}$ et l'état à chaque instant de commutation $\mathcal{X}$) qui minimise un critère donné.

Comme notre objectif est de privilégier un régime permanent cyclique, nous pouvons considérer un critère mesurant l'erreur quadratique aux instants de commutations $\mathcal{T}$ entre un cycle de référence périodique x^∞ et la trajectoire prédite sur un horizon de longueur de temps $T(=t_s)$.

Formellement, il s'agit d'un problème d'optimisation du type :

$$\begin{aligned} \min_{\mathcal{T}, \Omega, s} J(\mathcal{T}, \Omega, s) &= \min_{\mathcal{T}, \Omega, s} \sum_{k=1}^s (x_k - x_k^\infty)^T Q (x_k - x_k^\infty) \\ t.q. \quad \dot{x}(t) &= f(x(t)) + g(x(t))u(t), \quad u(t) \in \{0, 1\}^r \\ x(0) &= x_0 \end{aligned} \quad (2.5)$$

où $t \mapsto u(t)$ définit le mode actif à l'instant t et donc la séquence de modes Ω , x_k sont les valeurs de x aux instants de commutations. Rappelons que ces valeurs définissent l'ensemble $\mathcal{X}$ par la définition 13 du Chapitre I. x_k^∞ sont les valeurs de la référence aux instants de commutations. x_0 est l'état mesuré à l'instant initial et $\mathcal{Q} = \mathcal{Q}^T > 0$ est une matrice de poids. Dans cette formulation, T est l'horizon de prédiction. Comme précédemment, nous supposons l'absence du bruit de mesure et l'état complètement mesurable. Nous n'avons donc pas besoin d'un estimateur d'état pour évaluer x_0 .

Le terme de pénalisation sur les variations de la commande n'apparaît pas car la valeur de la commande indique un numéro de modèle et n'a pas de signification physique ou énergétique. On peut cependant envisager dans de futurs travaux, de pénaliser les changements de configurations pour limiter la fréquence de commutation.

2.3.2 Régime permanent x^∞

Avant de résoudre le problème (2.5), il est nécessaire d'établir un cycle de référence optimal x^∞ au sens donné par un critère.

Considérons le système

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t)$$

où les modes apparaissent explicitement sous la forme du vecteur u . Le but est d'obtenir la meilleure séquence de temps $\mathcal{T}^\infty$, séquence de modes Ω^∞ et la longueur s^∞ ($1 < s^\infty < s_{max}$), qui nous donne une forme d'onde spécifique. Afin de minimiser les oscillations autour du point de fonctionnement désiré, nous avons choisi un critère quadratique intégral autour de la valeur moyenne de référence $\bar{x}^\infty$:

$$J(\mathcal{T}^\infty, \Omega^\infty, s^\infty) = \min_{\mathcal{T}, \Omega, s} \int_0^{T_p} [x(t) - \bar{x}^\infty]^T \mathcal{Q} [x(t) - \bar{x}^\infty] dt \quad (2.6)$$

où T_p est la période du cycle. T_p n'est pas connu à l'avance mais est supposé borné

$$T_p < T_{p,max} \quad (2.7)$$

Le caractère cyclique de la trajectoire impose la contrainte :

$$x(0) = x(T_p) \quad (2.8)$$

Par ailleurs, on impose une durée minimum entre les commutations. Par exemple, dans un système type convertisseur statique, les composantes de u qui

représentent l'état des interrupteurs, ont la propriété d'avoir un temps minimum entre deux commutations. On considère les contraintes additionnelles suivantes :

$$\begin{aligned} \delta_k(t_i) &\geq t_{min}|u_k(t_j) - u_k(t_{j-1})| & \forall j = 1, \dots, s \\ \dot{\delta}_k(t_j) &= 1 & \forall k = 1, \dots, r \\ \delta_k(t_{j+1}) &= 0 & \text{si } |u_k(t_j) - u_k(t_{j-1})| \neq 0 \end{aligned} \quad (2.9)$$

L'équation (2.9) est une condition sur la durée minimum d'activation du mode σ_j où t_{min} est une constante. t_{min} représente le temps minimum entre deux instants d'activation du même interrupteur. δ_k est le temps accumulé à partir de la dernière activation de l'interrupteur. Cet ensemble de contraintes représente un intégrateur avec une condition de remise à zéro pour chaque interrupteur.

Rappelons que dans la séquence de modes $\Omega = \{\sigma_0, \sigma_1, \dots, \sigma_i, \dots, \sigma_k, \dots, \sigma_{s-1}\}$, il est possible que $\sigma_i = \sigma_k$ pour $i \neq k$.

Remarque 2 *D'autres critères peuvent être aussi considérés. Par exemple, afin de minimiser le contenu harmonique, nous pouvons changer la fonction à minimiser (2.6) par :*

$$J(\mathcal{T}^\infty, \Omega^\infty, s^\infty) = \min_{\mathcal{T}, \Omega, s} \int_0^{T_p} [x_w(t) - \bar{x}^\infty]^T \mathcal{Q} [x_w(t) - \bar{x}^\infty] dt \quad (2.10)$$

où x_w est la variable d'état filtrée. Autrement dit, la variable d'état n'est pas prise en compte dans le critère, mais plutôt la variable filtrée. [SMQG07] utilise cette idée pour réaliser la commande prédictive d'un convertisseur multiniveaux où le courant est filtré par un filtre passe-bande afin de concentrer le spectre du courant à une fréquence déterminée.

La résolution du problème (2.6) sous les contraintes (2.7)-(2.8) et (2.9) déterminent la trajectoire de référence x^∞ et sera traité à la section 2.4.1.

2.3.3 Stratégie de résolution du problème (2.5)

La stratégie appliquée pour trouver les solutions de (2.5) pour un ensemble de conditions initiales x_0 données dépend de la distance au cycle. On définit 2 stratégies :

- Stratégie de ralliement : Loin du voisinage du cycle optimal (Régime transitoire), i.e. lorsque la relation suivante est vérifiée :

$$(x_0 - \bar{x}^\infty)^T \mathcal{Q} (x_0 - \bar{x}^\infty) > \varepsilon \quad (2.11)$$

où $\bar{x}^\infty$ est la valeur moyenne du cycle et est généralement donnée par la valeur désirée du cahier de charge. ε est une constante qui détermine la

tolérance d'erreur afin de considérer le système en régime permanent ou en régime transitoire.

L'horizon de prédiction T et le nombre de mode s sur cet horizon sont fixés. Egalement, la référence $x^\infty(t)$ est fixée à une valeur constante qui est la valeur moyenne du cycle de référence (Voir la figure 2.2(a)). Ce problème est considéré comme un problème de commande classique avec une référence constante.

Le cycle n'est pas choisi comme référence pour le régime transitoire parce que la trajectoire du système est très loin du cycle. Une référence cyclique n'apporte donc pas d'information supplémentaire et par contre, rend le calcul de la commande plus difficile à cause de la variation temporelle.

Dans le régime transitoire, la séquence de temps de commutations $\mathcal{T}$ et la séquence des modes Ω sont optimisées.

- Stratégie de Poursuite : Dans le voisinage du cycle optimal (Régime permanent), nous vérifions l'inégalité suivante

$$(x_0 - \bar{x}^\infty)^T \mathcal{Q}(x_0 - \bar{x}^\infty) \leq \varepsilon \quad (2.12)$$

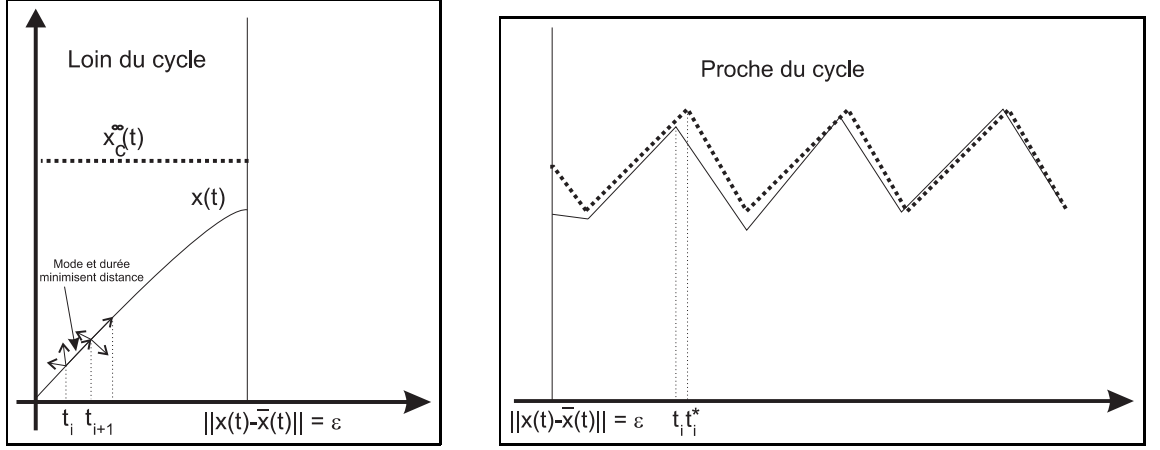
Seule la séquence de temps de commutation $\mathcal{T}$ est recherchée par le modèle prédictif car la séquence de modes Ω , le nombre de modes s et l'horizon de prédiction sont fixés à leurs valeurs de référence Ω^∞ , s^∞ et t_s^∞ .

La référence x^∞ est le cycle. Comme la trajectoire $x(t)$ est proche du cycle, ni les modes, ni la longueur de la séquence changent de valeurs. (Voir la figure 2.2(b)).

La gestion de la transition entre régime transitoire et régime permanent nécessite le recalage de la séquence optimale celle-ci étant définie à une permutation près sur l'ordre de la séquence. Supposons que le système passe du régime transitoire au régime permanent à l'instant $t = t_c$, avec l'état x_c nous devons déterminer la valeur initiale du cycle de référence à l'instant t_c . Ce choix s'effectue en choisissant parmi les valeurs du cycle aux instants de commutation x_k^∞ , la valeur la plus proche i.e. on choisit de prendre l'indice qui minimise la distance :

$$k_0 : \arg \left(\min_{k \in \{1, 2, \dots, s\}} (x_c - x_k^\infty)^T \mathcal{Q}(x_c - x_k^\infty) \right) \quad (2.13)$$

Une fois fixé k_0 , le cycle de référence est appliqué après recalage de la séquence dans l'ordre $x(t_{k_0}), x(t_{k_0+1}), \dots, x(t_{k_0+s-1}), x(t_{k_0}), x(t_{k_0+1}), \dots, x(t_{k_0+s-1}), x(t_{k_0}), \dots$



(a) Régime transitoire. Nous recherchons la séquence de temps de commutation $\mathcal{T}$, la séquence de modes Ω . La référence est la valeur moyenne d'un cycle.

(b) Régime permanent. Nous recherchons seulement la séquence de temps de commutations $\mathcal{T}$. Les modes sont fixés à la valeur de référence Ω^∞ ainsi que l'horizon de prédiction à la période du cycle.

FIG. 2.2 – Exemple d'une trajectoire commande $x(t)$ avec une méthode prédictive

2.4 Résolution du problème de la commande prédictive

2.4.1 Régime permanent

Nous trouvons la séquence de temps et la séquence de modes qui vérifient (2.6) de façon algorithmique. Plus exactement, fixons un ensemble de modes Ω et une valeur de s , alors le problème est de déterminer les instants de commutation. Nous adoptons l'algorithme suivant :

- Algorithme 20**
1. Pour s et Ω fixés, nous optimisons la fonction de coût $J(\mathcal{T}, \Omega, s)$ décrit dans l'équation (2.6) par rapport à la séquence de temps $\mathcal{T}$, sous les contraintes (2.7), (2.8), et (2.9)
 2. On reprend le point 1 avec un nouveau couple Ω et s jusqu'à épuisement des séquences possibles.
 3. On sélectionne la séquence de modes et la séquence de temps telles que la valeur $J(\mathcal{T}, \Omega, s)$ est minimum.

L'algorithme se termine car Ω est fini et s est borné en raison des temps minimum de commutation.

Généralement, une solution analytique au problème d'optimisation (2.6) sous les contraintes (2.8), (2.7) et (2.9) est impossible à obtenir. Nous ne pouvons

qu'appliquer des méthodes numériques d'optimisation non-linéaires. Ce type de méthodes montre d'habitude une bonne efficacité. La solution définit une exécution $(\mathcal{T}^\infty, \Omega^\infty, \mathcal{X}^\infty)$ et la longueur de l'exécution s^∞ .

Remarque 3 *La méthode peut être numériquement très lourde car nous devons résoudre le problème (2.10) $\sum_{s=1}^{s_{max}} r^s$ fois. Cette optimisation est faite hors ligne.*

2.4.1.1 Exemple numérique (Convertisseur buck-boost)

Considérons un convertisseur buck-boost en mode de conduction continue (CCM) dont la topologie est donnée sur la figure 2.3.

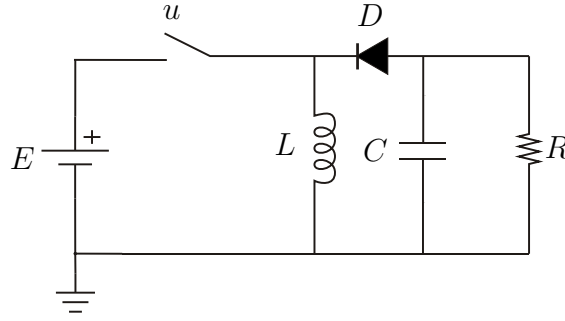


FIG. 2.3 – Convertisseur buck-boost

La dynamique de ce système dont les variables d'état sont le courant dans l'inductance i_L et la tension aux bornes de la capacitance v_c est déterminée par l'équation :

$$\dot{x} = \begin{cases} \mathcal{A}_1 x + \mathcal{B}_1 & \text{pour } u = 1 \text{ (commutateur fermé - Mode 1)} \\ \mathcal{A}_2 x + \mathcal{B}_2 & \text{pour } u = 0 \text{ (commutateur ouvert - Mode 2)} \end{cases}$$

où

$$\mathcal{A}_1 = \begin{bmatrix} 0 & 0 \\ 0 & -\frac{1}{RC} \end{bmatrix} \quad \mathcal{A}_2 = \begin{bmatrix} 0 & \frac{1}{L} \\ -\frac{1}{C} & -\frac{1}{RC} \end{bmatrix}$$

$$\mathcal{B}_1 = \left[\frac{E}{L}, 0 \right]^T \quad \mathcal{B}_2 = [0, 0]^T$$

où $R = L = C = E = 1$. Rappelons que le mode discontinu n'est pas pris en compte. En conséquence, les diodes n'ont pas d'influence sur la réponse du système.

$f(x)$ et $g(x)$ sont directement obtenus du modèle $f(x) = \mathcal{A}_2 x + \mathcal{B}_2 E$ et $g(x) = (\mathcal{A}_1 - \mathcal{A}_2)x + (\mathcal{B}_1 - \mathcal{B}_2)E$. Le critère à minimiser est les oscillations par rapport à une valeur moyenne $x_{ref} = [i_{l,ref}, v_{c,ref}]^T = [2, -1]^T$. La matrice de poids est $\mathcal{Q} = \text{diag}[1, 1]$. Nous fixons $T_{p,max} = 1$ s. ainsi que le nombre de modes maximum

$s_{max} = 2$. Le commutateur a un temps de commutation minimal de $t_{min} = 0.25s$. Pour cet exemple $r = 1$, donc il y a juste 4 séquences possibles. Après avoir résolu le problème d'optimisation, nous obtenons les séquences optimales :

$$\mathcal{T}^\infty = \{0, 0.2509, 0.5\}, \quad \Omega^\infty = \{1, 2\} \quad (2.14)$$

La figure 2.4 montre la trajectoire dans l'espace d'état (x_1, x_2) à gauche et à droite l'évolution en temps de chaque variable. Une fois le cycle de référence choisi, il

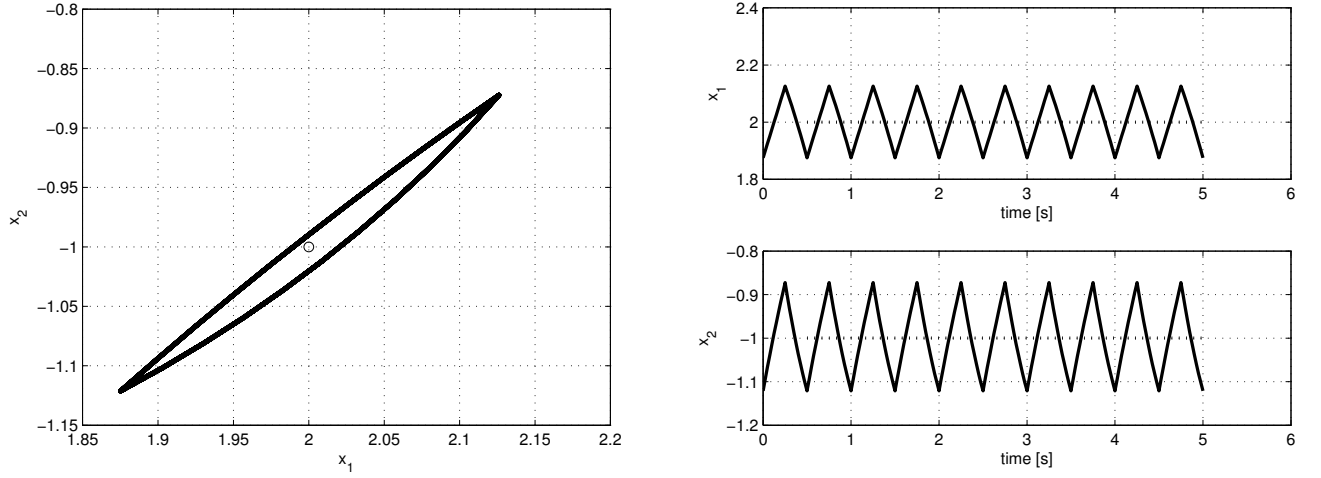


FIG. 2.4 – Cycle optimal pour un convertisseur buck-boost et un critère des oscillations minimum autour du $[2, -1]^T$. La figure de gauche montre le cycle dans l'espace d'état. La figure de droite montre l'évolution des variables d'état.

nous reste à déterminer la stratégie de ralliement. Nous intéressons donc à la poursuite d'un cycle.

2.4.2 Optimisation des temps de commutation entre modes

Tout au long de cette section, nous exposons la méthode employée pour optimiser les temps de commutations lorsque la séquence de modes est fixée. Cette méthode est utilisée quelle que soit la stratégie considérée (Régime permanent ou transitoire).

Il nous faut donc déterminer la séquence de temps de commutation $\mathcal{T}$ pour Ω et s fixés. Notons que lorsqu'il s'agit du régime permanent $\Omega = \Omega^\infty$ et $s = s^\infty$.

Il suffit de formaliser la recherche de $\mathcal{T}$ comme un problème d'optimisation concernant encore une fois un critère quadratique entre la trajectoire x et le régime permanent x^∞ mais cette fois, nous ne regardons qu'**aux instants de**

commutation. Cette idée est employée par la commande prédictive fonctionnelle (PFC) [BD96] où nous ne regardons que dans les “points de coïncidence”.

On pose alors le problème de la commande des temps de commutation entre les modes comme :

$$(\mathcal{P}_{\mathcal{T}}) \quad \left\{ \begin{array}{l} \text{Minimiser} \quad J(\mathcal{T}) = \sum_{i=1}^s (x_i - x_i^*)^T \mathcal{Q}(x_i - x_i^*) \\ \text{tel que :} \quad \begin{array}{l} \dot{x}(t) = f(x(t)) + g(x(t))u(t) \\ x(0) = x_0 \quad (\text{Condition initiale donné}) \\ t_s = T_p^\infty \quad (\text{Condition sur la période}). \end{array} \end{array} \right.$$

où $T_p^\infty \in \mathbb{R}$ est la période du cycle de référence et

$$x_i^* = \begin{cases} x^\infty(t_i^\infty) & \text{si régime permanent} \\ \bar{x}^\infty & \text{si régime transitoire} \end{cases} \quad (2.15)$$

Dans ce problème l'ordre de la séquence de référence est déterminé à partir de x_0 , comme dans (2.13).

Nous proposons une méthode basée sur l'algorithme de Gauss-Newton et la connaissance des fonctions de sensibilité afin de trouver la séquence de temps $\mathcal{T}$ qui vérifie $(\mathcal{P}_{\mathcal{T}})$. Les fonctions de sensibilité ont été utilisées pour le calcul de cycles [FRL06]. Nous utilisons cette approche et l'adaptions à un algorithme de Gauss-Newton qui cherche un minimum lorsqu'il n'y pas de contraintes.

Écrivons l'évolution de l'état comme fonction des instants de commutation :

$$x(t_i) = \Phi_i(t_i, t_{i-1})x(t_{i-1}) \quad \forall i = 1, \dots, s \quad (2.16)$$

où $\Phi_i(.,.)$ est la matrice de transition du système linéarisé associée au système

$$\dot{x} = f_i(t, x) \quad t_{i-1} \leq t \leq t_i \quad (2.17)$$

$\Phi_i(.,.)$ est définie par

$$\begin{cases} \frac{d}{dt}\Phi_i(t, t_{i-1}) = \frac{\partial f_i}{\partial x}\Phi_i(t, t_{i-1}) & t_{i-1} \leq t \leq t_i \\ \Phi_i(t_{i-1}, t_{i-1}) = I \end{cases} \quad (2.18)$$

Comme nous ne considérons que des systèmes linéaires (voire affines) par morceaux, l'équation d'évolution peut s'écrire :

$$\dot{x}(t) = A_{\sigma_i}x(t) \quad (2.19)$$

et la matrice de transition est :

$$\Phi_i(t_i, t_{i-1}) = e^{A_{\sigma_i}(t_i - t_{i-1})} \quad \forall i = 1, \dots, s \quad (2.20)$$

L'écart entre la trajectoire et le cycle de référence Δx aux instants $\mathcal{T}$ est donné par l'expression :

$$\Delta x(t_i) = x(t_i) - x_i^*, \quad \forall i = 1, \dots, s \quad (2.21)$$

avec x_i^* défini comme dans (2.15).

Comme la période du cycle est de valeur T_p^∞ , la valeur du temps $t_s = T_p^\infty$ et substituant (2.21) dans le problème $(\mathcal{P}_\mathcal{T})$, nous arrivons à un problème de minimisation sans contrainte :

$$\min_{\mathcal{T}} J(\mathcal{T}) = \min_{\mathcal{T} \setminus t_s} \Delta x(T_p^\infty)^T \mathcal{Q} \Delta x(T_p^\infty) + \sum_{i=1}^{s-1} \Delta x(t_i)^T \mathcal{Q} \Delta x(t_i) \quad (2.22)$$

Ce problème peut aussi s'écrire de manière matricielle comme :

$$\min_{\mathcal{T}} J(\mathcal{T}) = \min_{\mathcal{T} \setminus t_s} \Delta X(\mathcal{T})^T \mathbf{Q} \Delta X(\mathcal{T}) \quad (2.23)$$

où $\Delta X(\mathcal{T}) : \mathbb{R}^s \rightarrow \mathbb{R}^{ns}$ est un vecteur qui contient tous les écarts des états à chaque temps de commutation :

$$\Delta X(\mathcal{T}) = \begin{bmatrix} \Delta x(t_1) \\ \Delta x(t_2) \\ \vdots \\ \Delta x(t_{s-1}) \\ \Delta x(T_p^\infty) \end{bmatrix} \quad (2.24)$$

$\mathbf{Q} : \mathbb{R}^{ns \times ns}$ est une matrice diagonale constante composée par la matrice $\mathcal{Q}$ répétée s fois.

$$\mathbf{Q} = \begin{bmatrix} \mathcal{Q} & 0_{n \times n} & \dots & 0_{n \times n} \\ 0_{n \times n} & \mathcal{Q} & \dots & 0_{n \times n} \\ \vdots & \vdots & \ddots & 0_{n \times n} \\ 0_{n \times n} & 0_{n \times n} & \dots & \mathcal{Q} \end{bmatrix} \quad (2.25)$$

Notons τ la séquence de temps $\mathcal{T}$ privée de t_s ($\mathcal{T} \setminus t_s$). (2.23) est une minimisation quadratique et nous pouvons l'associer au problème de moindre carrés non-linéaire. Un algorithme de Gauss - Newton est donc très avantageux pour cette formulation.

Algorithme 21 (*Algorithme pour déterminer les instants de commutation - Solution au problème $(\mathcal{P}_\mathcal{T})$*).

Données d'entrées :

- Valeur de x_0 , taille de la séquence s , séquence de modes Ω .
- Valeurs initiales de la séquence de temps de commutation τ_{in} .
- Période du cycle T_p^∞
- Paramètre d'arrêt de l'algorithme $\nu \approx 0$ (Constante).

L'algorithme est formé des étapes suivantes :

1. Initialiser $h \leftarrow 0$, $\tau^0 \leftarrow \tau_{in}$
2. Iteration de l'algorithme de Gausse-Nezton
3. Validation de contraintes : $\forall i = 1, \dots, s$, si $(t_i - t_{i-1}) < 0$, $t_i \leftarrow t_{i-1}$
4. $\forall i = 1, \dots, s$, si $(t_i - t_{i-1}) > T_p^\infty$, $t_i \leftarrow T_p^\infty$
5. Test d'arrêt : Si $\|\tau^{h+1} - \tau^h\| < \nu$, aller au pas 7.
6. $h \leftarrow h + 1$ et retourner au pas 2.
7. Fin

Remarque 4 Les étapes 3 et 4 de l'algorithme sont des cas particuliers. L'étape 3 évite les durées de modes négatives. Si $t_{i-1} > t_i$, alors le mode σ_{i-1} fait partie d'une séquence de modes avec une durée nulle. L'étape 4 traite le cas où une séquence de temps a une durée supérieure à la période du cycle. En effet, le cas $t_i > T_p^\infty + t_{i-1}$ pour $t_{i-1} > 0$ n'est pas admissible (La période de cycle est T_p^∞).

$\frac{\partial J(\tau)}{\partial \tau}$ est la fonction de sensibilité de J vis à vis les instants de commutations.

D'après les règles de dérivation de fonction composées :

$$\frac{\partial J(\tau)}{\partial \tau} = \mathcal{J}^T(\tau) \frac{\partial J(\tau)}{\partial \Delta X(\tau)} = \mathcal{J}^T(\tau) \mathbf{Q} \Delta X(\tau) \quad (2.27)$$

où $\mathcal{J}(\tau) : \mathbb{R}^s \rightarrow \mathbb{R}^{(ns) \times (s-1)}$ est la matrice jacobienne des écarts par rapport à chaque instant de commutation,

$$\mathcal{J}(\tau) = \begin{bmatrix} \frac{\partial \Delta x(t_1)}{\partial t_1} & \frac{\partial \Delta x(t_1)}{\partial t_2} & \dots & \frac{\partial \Delta x(t_1)}{\partial t_{s-1}} \\ \frac{\partial \Delta x(t_2)}{\partial t_1} & \frac{\partial \Delta x(t_2)}{\partial t_2} & \dots & \frac{\partial \Delta x(t_2)}{\partial t_{s-1}} \\ \vdots & \vdots & \dots & \vdots \\ \frac{\partial \Delta x(t_s)}{\partial t_1} & \frac{\partial \Delta x(t_s)}{\partial t_2} & \dots & \frac{\partial \Delta x(t_s)}{\partial t_{s-1}} \end{bmatrix} \quad (2.28)$$

et $\mathcal{H}(\tau)$ est une approximation de la matrice Hessienne inversible définie par :

$$\mathcal{H}(\tau) = \mathcal{J}^T(\tau) \mathbf{Q} \mathcal{J}(\tau) \quad (2.29)$$

Nous devons alors calculer les fonctions de sensibilité de l'écart de l'état par rapport à chaque temps de commutation. La proposition suivante aide pour réaliser ce calcul.

Proposition 22 Les fonctions de sensibilité de $\Delta x(\tau)$ vis à vis des temps de commutations sont données par : $\forall j = 1, \dots, s-1$, et $\forall i = 1, \dots, s$

$$\frac{\partial \Delta x(t_i)}{\partial t_j} = \begin{cases} 0 & \text{si } j > i \\ A_{\sigma_i} \Pi(t_i, 0) x(0) & \text{si } j = i \\ \Pi(t_i, t_j) \Delta P_j x(0) & \text{si } j < i \end{cases} \quad (2.30)$$

où $\Pi(t_i, t_j)$ est le produit des matrices de transition de $t = t_j$ à $t = t_i$:

$$\Pi(t_i, t_j) = \Phi_i(t_i, t_{i-1}) \Phi_{i-1}(t_{i-1}, t_{i-2}) \dots \Phi_{j+1}(t_{j+1}, t_j) \quad (2.31)$$

ΔP_j est une fonction sous la forme :

$$\Delta P_j = (A_{\sigma_j} - A_{\sigma_{j+1}}) \Pi(t_j, 0) \quad (2.32)$$

Preuve. Même si $x^\infty(t)$ est une référence variable dans le temps, pour un t_i donné, $x^\infty(t_i)$ est constante. D'après (2.21) :

$$\frac{\partial \Delta x(t_i)}{\partial t_j} = \frac{\partial x(t_i)}{\partial t_j}$$

Nous procédons avec le calcul de $\frac{\partial x(t_i)}{\partial t_j}$. Nous devons alors distinguer :

1. $j > i$: les termes t_j n'apparaissent pas dans les fonctions $x(t_i)$, alors $\frac{\partial x(t_i)}{\partial t_j} = 0$.
2. $j = i$: Considérons la dérivée à gauche de l'état $x(t)$ par rapport à l'instant de commutation t_j :

$$\frac{\partial x(t_j^-)}{\partial t_j} = \lim_{t \rightarrow t_j^-} \frac{\partial x(t)}{\partial t_j}$$

Nous calculons l'évolution du système entre $t_j - \epsilon$ et t_j comme $x(t_j) = x(t_j - \epsilon) + \int_{t_j - \epsilon}^{t_j} A_{\sigma_j} x(p) dp$. Sa dérivée donne :

$$\frac{\partial x(t_j^-)}{\partial t_j} = A_{\sigma_j} x(t_j) \quad (2.33)$$

Comme $j = i$, nous ne devons pas considérer les commutations suivantes, et :

$$\frac{\partial x(t_j)}{\partial t_j} = \frac{\partial x(t_j^-)}{\partial t_j} = A_{\sigma_j} x(t_j) \quad (2.34)$$

3. $j < i$: Contrairement au cas $j = i$, les commutations suivantes doivent être prises en compte. Considérons la dérivée à droite de l'état $x(t)$ par rapport à l'instant de commutation t_j :

$$\frac{\partial x(t_j^+)}{\partial t_j} = \lim_{t \rightarrow t_j^+} \frac{\partial x(t)}{\partial t_j}$$

Calculons comme précédemment l'évolution de l'état $x(t)$ comme $x(t) = x(t_j^+) + \int_{t_j}^t A_{\sigma_{j+1}} x(p) dp$. Nous pouvons obtenir facilement la dérivée de $x(t)$. Comme $x(t_j^+) = x(t_j^-) = x(t_j)$ et d'après (2.34), la limite $t \rightarrow t_j^+$ prend la forme :

$$\frac{\partial x(t_j^+)}{\partial t_j} = (A_{\sigma_j} - A_{\sigma_{j+1}})x(t_j) \quad (2.35)$$

D'après (2.16), nous exprimons $x(t_j)$ comme fonction de l'état initial comme $x(t_j) = \Pi(t_j, 0)x(0)$ et en utilisant la règle de dérivation :

$$\frac{\partial x(t_i)}{\partial t_j} = \frac{\partial x(t_i)}{\partial x(t_j)} \frac{\partial x(t_j)}{\partial t_j} = \frac{\partial x(t_i)}{\partial x(t_{i-1})} \frac{\partial x(t_{i-1})}{\partial x(t_{i-2})} \dots \frac{\partial x(t_{j-1})}{\partial x(t_j)} \frac{\partial x(t_j^+)}{\partial t_j} \quad (2.36)$$

Calculons donc la fonction de sensibilité de $x(t_i)$ par rapport à $x(t_{i-1})$. De (2.16), elle est facilement déduite comme $\frac{\partial x(t_i)}{\partial x(t_{i-1})} = \Phi_i(t_i, t_{i-1})$. Donc,

$$\frac{\partial x(t_i)}{\partial t_j} = \Pi(t_i, t_j)(A_{\sigma_j} - A_{\sigma_{j+1}})x(t_j) \quad (2.37)$$

■

2.4.2.1 Exemple numérique (Reprise du convertisseur buck-boost)

Reprenons l'exemple du convertisseur buck-boost de la section 4.2. Nous écrivons le système de façon autonome en ajoutant une variable d'état x_{n+1} avec $x_{n+1}(0) = 1$. Les matrices deviennent donc :

$$A_{\sigma_i} = \begin{bmatrix} \mathcal{A}_{\sigma_i} & \mathcal{B}_{\sigma_i} \\ 0 & 0 \end{bmatrix} \quad (2.38)$$

Le système est mis sous la forme autonome $\dot{x}(t) = A_{\sigma_i}x(t)$. La séquence de référence est celle obtenue dans la section 4.2 :

$$\mathcal{T}^\infty = \{0, 0.2509, 0.5\}, \quad \Omega^\infty = \{1, 2\}$$

Les valeurs du cycle de référence aux instants de commutation sont facilement obtenues comme :

$$\begin{aligned} x^\infty(t_2) &\in \text{Ker} \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - e^{A_2(t_2^\infty - t_1^\infty)} e^{A_1 t_1^\infty} \right) \quad \text{car } x^\infty(0) = x^\infty(t_2) \\ x^\infty(t_1) &= e^{A_1(t_2^\infty - t_1^\infty)} x^\infty(t_2) \end{aligned} \quad (2.39)$$

D'après (2.23), la fonction à minimiser est :

$$J(\tau) = \Delta X(\tau)^T \mathbf{Q} \Delta X(\tau) \quad (2.40)$$

où $\Delta X(\tau) = [\Delta x_1(\tau)^T, \Delta x_2(\tau)^T]^T$ avec

$$\begin{aligned}\Delta x(t_1) &= e^{A_1(t_1)}x(0) - x^\infty(t_1) \\ \Delta x(t_2) &= e^{A_2(T_p-t_1)}x(t_1) - x^\infty(t_2)\end{aligned}\tag{2.41}$$

$\mathbf{Q} = \text{diag}[\mathcal{Q}, \mathcal{Q}]$, et la matrice de poids $\mathcal{Q}$ est fixée à :

$$\mathcal{Q} = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\tag{2.42}$$

Calculons $\frac{\partial J(\tau)}{\partial \tau}$ comme $\mathcal{J}(\tau)^T \mathbf{Q} \Delta X(\tau)$ ou $\mathcal{J}(T) = \left[\frac{\partial \Delta x(t_1)^T}{\partial t_1}, \frac{\partial \Delta x(t_2)^T}{\partial t_1} \right]^T$ et d'après la proposition 22, les fonctions de sensibilité sont sous la forme :

$$\begin{aligned}\frac{\partial \Delta x(t_1)}{\partial t_1} &= A_1 \Pi(t_1, 0) x(0) \\ \frac{\partial \Delta x(t_2)}{\partial t_1} &= \Pi(t_2, t_1) \Delta P_1 x(0)\end{aligned}\tag{2.43}$$

où $\Pi(t_1, 0) = e^{A_1 t_1}$, $\Pi(t_2, t_1) = e^{A_2(T_p-t_1)}$ et $\Delta P_1 = (A_1 - A_2)\Pi(t_1, 0)$.

L'évolution de x_1 et de x_2 est montrée dans la figure 2.5. Sur la figure 2.6,

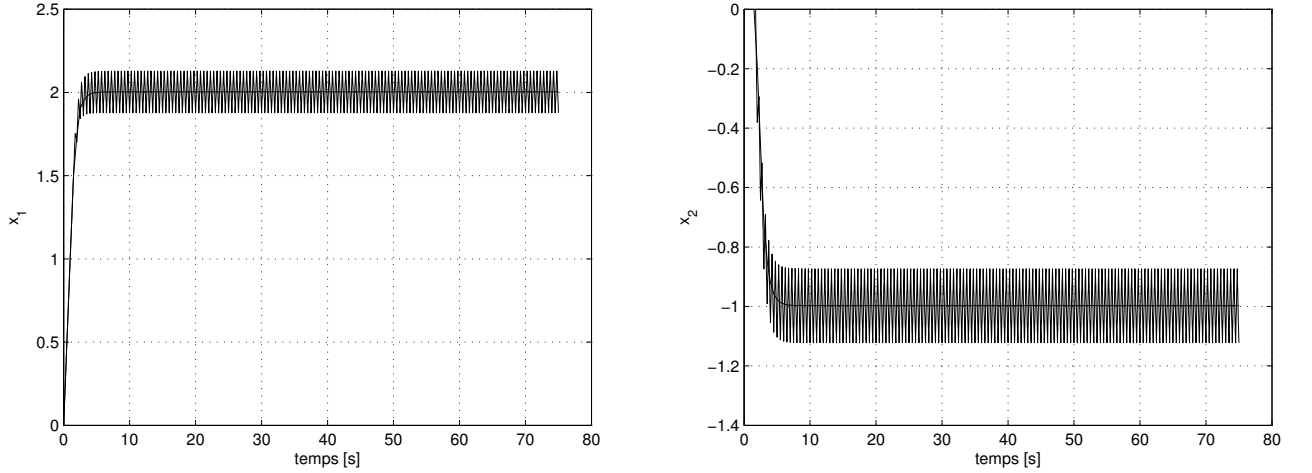


FIG. 2.5 – $x_1(t)$ et $x_2(t)$: Le but est d'avoir comme régime permanent la séquence $\mathcal{T} = \{0, 0.2509, 0.5\}$, $\Omega = \{1, 2\}$

nous observons le cycle de référence ainsi qu'un zoom de la trajectoire obtenue. Les durées $\tau_1 = t_1$ et $\tau_2 = T_p - t_1$ sont présentées dans la figure de droite.

Nous constatons sur cet exemple que le système arrive à joindre le cycle. Puisque le système du convertisseur buck-boost ne possède que deux modes, la séquence de modes Ω est connue aussi pour le régime transitoire et l'algorithme 21 converge vers le régime permanent souhaité.

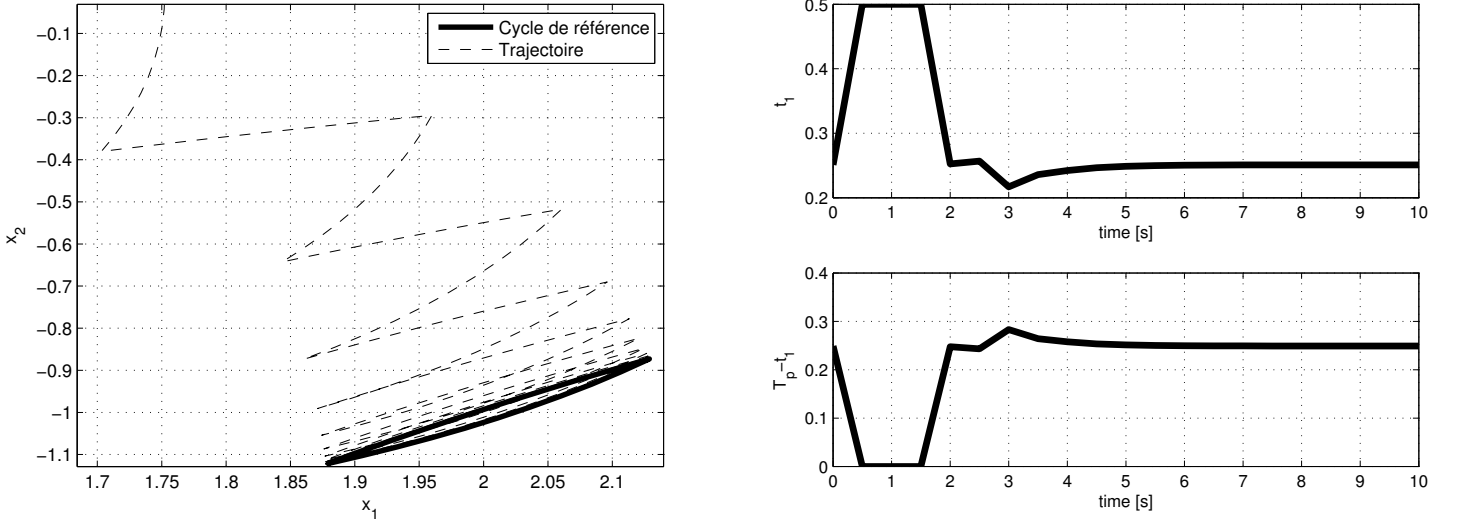


FIG. 2.6 – Le graphique de gauche montre le cycle de référence et la trajectoire x_1, x_2 proche à ce cycle. La graphique de droite montre les durées de chaque mode

2.5 Construction de la boucle fermée

L'algorithme 21 a comme paramètres d'entrées x_0 (Mesure de l'état), la séquence de modes Ω et la longueur de la séquence s . Si ces deux dernières sont connues à l'avance, la commande en boucle fermée est construite directement avec cet algorithme. Cependant, ce cas est bien particulier. En général, Ω et s ne sont pas connus. Deux solutions possibles sont :

1. Tester l'algorithme 21 avec toutes les séquences de modes de longueur s , où s varie entre 1 et s_{max} ($1 \leq s \leq s_{max}$), et prendre la séquence qui donne la valeur $J(\mathcal{T})$ minimum en $(\mathcal{P}_{\mathcal{T}})$.

Cette solution devient donc un procédé très coûteux numériquement parlant pour pouvoir l'appliquer en ligne et par conséquent, la commande ne sera pas disponible en temps réel. Ce procédé peut cependant être envisageable hors-ligne.

2. Faire varier x_0 et pour chaque x_0 , calculer $\mathcal{T}$ et Ω et s comme le point 1 mais hors-ligne. De cette manière, nous créons une table qui a comme entrées des valeurs de x_0 et comme sorties le mode employé σ_0 et sa durée $t_1 - t_0$. En suite nous utilisons une méthode d'interpolation pour établir une relation entrée - sortie. Cette relation est alors appliquée en ligne.

Une des méthodes qui a montré des bons résultats de robustesse et la rapidité de calcul est le réseau de neurones. Une fois que toutes les solutions

pour un ensemble de conditions initiales sont obtenues, le réseau de neurones les interpole de sorte qu'il peut imiter le comportement du régulateur (Voir Figure 2.7)

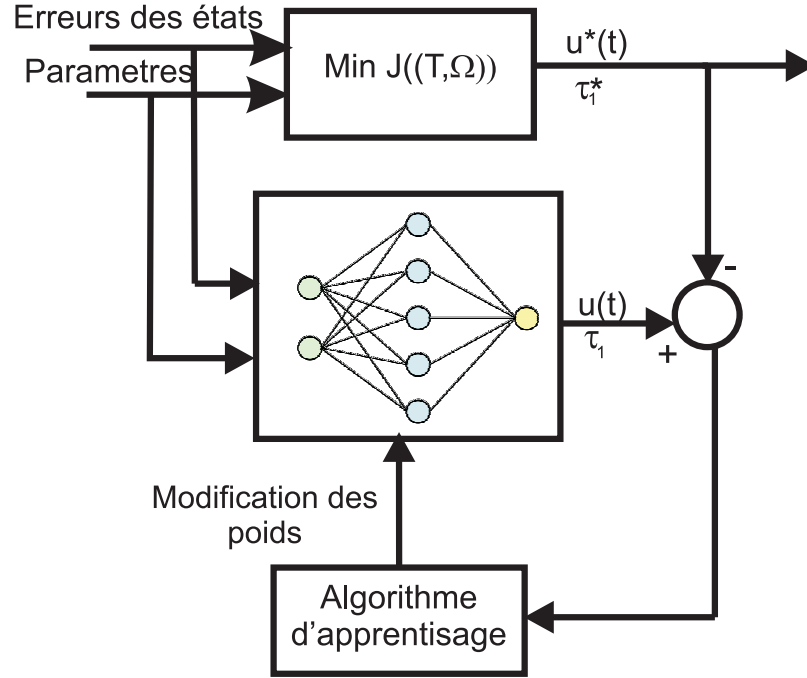


FIG. 2.7 – Structure d'un réseau de neurones

Puisque les réseaux de neurones sont des approximateurs universels, ils peuvent *apprendre* le comportement d'une fonction non linéaire et par conséquent calculer une commande en ligne sans la nécessité de faire beaucoup de calculs. Dans [RFNM96], un réseau de neurones est utilisé pour l'implémentation d'une commande prédictive. Les réseaux de neurones ont été appliqués aussi dans l'industrie comme par exemple dans le moteur à combustion [CCC⁺07], [AAaSI07].

2.6 Conclusions et perspectives

Dans ce chapitre nous avons proposé une solution pour commander un système hybride avec des entrées binaires.

La méthode proposée se décompose en trois parties : i) Le calcul d'un cycle optimal en boucle ouverte par programmation non-linéaire avec un critère de moindres oscillations et en respectant des contraintes de durée sur les commutations. ii) La poursuite de la trajectoire par une commande prédictive avec une stratégie de commande fonction de l'éloignement à la courbe de ralliement. iii)

Si c'est nécessaire, l'implémentation de la commande au moyen d'un réseau de neurones afin de satisfaire des contraintes de temps réel. La commande est alors obtenue par la sortie du réseau de neurones.

Un exemple sur un convertisseur buck-boost montre l'efficacité de la méthode. Dans le chapitre 4, nous verrons des applications pratique sur des convertisseurs de puissance.

Nous pouvons donc conclure que même si la méthode nécessite la résolution de beaucoup d'algorithmes d'optimisation, ils sont utilisés hors-ligne (si c'est nécessaire) et des techniques d'interpolation permettent d'établir une loi de commande par retour d'état.

La méthode de commande permet de suivre un cycle défini au préalable et par conséquent, de contrôler la forme d'onde. De ce fait elle doit permettre d'améliorer la qualité spectrale du signal de sortie. Bien qu'il n'existe pas de preuve concernant la stabilité, la méthode est flexible et facile à implémenter. Les simulations et les résultats expérimentaux obtenus montrent une bonne robustesse aux perturbations. La proposition présentée dans ce chapitre est donc viable. Cependant, il existe plusieurs points à étudier dans l'avenir :

- La prise en compte de saut autonomes, avec des changements de mode fonction de l'état. Il correspondent au mode de conduction discontinue dans les convertisseurs de puissance avec l'annulation des courants dans les diodes de roue libre.
- La variation de l'horizon de prédiction. Une étude du choix de l'horizon de prédiction peut améliorer les résultats pour les cas pratiques.
- La plus part des applications aujourd'hui sont des applications dans le domaine discrets. Pour ceci, une correction de la méthode doit prendre en compte la période d'échantillonnage de sorte que la période du cycle soit un multiple de cette fréquence.
- L'adaptation à une commande prédictive-adaptative. Pour ce cas, il faut étudier les techniques de filtrage pour des systèmes commutés.

Chapitre 3

Commande optimale par retour d'état des systèmes commutés

Résumé. Ce chapitre considère le problème de la commande optimale pour la classe des systèmes commutés affines à fonctionnement cyclique. Il est montré que le bon cadre d'étude est fourni par les solutions au sens de Fillipov qui étendent le domaine des commandes à son enveloppe convexe. La synthèse des trajectoires optimales passe alors par la détermination d'arcs singuliers. Afin d'obtenir ces arcs, des conditions algébriques sont énoncées pour des systèmes d'ordre faible. Nous proposons un algorithme qui génère les trajectoires optimales dans un secteur donné de l'espace d'état. Le but de cet algorithme est d'éviter le traditionnel problème aux deux bouts issu de la résolution des conditions nécessaires d'optimalité. L'interpolation de l'ensemble des solutions grâce à un réseau de neurones permet alors d'obtenir une loi de commande par retour d'état facile à implémenter en temps réel.

Mots clés. Commande optimale, commande singulière, retour d'état, systèmes commutés, réseau de neurones.

3.1 Introduction

Les techniques de commande actuelles cherchent de plus en plus à traiter des systèmes complexes modélisés par un ensemble d'équations constituées d'un grand nombre de paramètres. L'objectif de l'ingénieur automaticien consiste alors à asservir ce processus, autrement dit à amener le procédé d'un état à un autre en respectant certaines contraintes et en recherchant des performances évoluées, voire optimales (notion de productivité, de coût, de qualité des produits, ...). Ce problème d'optimalité n'est pas uniquement lié aux exigences économiques modernes. Une des grandes applications de la commande dite optimale a en effet été le programme Appolo dans les années 60 pour lequel l'une des difficultés

majeures était d'être capable d'aller sur la lune en consommant le minimum de carburant possible.

Historiquement, la théorie de la commande optimale est liée à la mécanique classique et plus particulièrement aux principes variationnels de la mécanique (principe de Fermat, de Huygens, équations d'Euler Lagrange). Bien que les principaux efforts et outils de calcul et d'analyse aient été proposés dans les années 50, la théorie mathématique sous-jacente date de 1638 lorsque Galilée cherchait à déterminer la courbe sur laquelle une bille roule d'un point A à un point B , sous l'unique action de la pesanteur, sans vitesse initiale, en un temps minimal. Ce problème, baptisé problème de la courbe brachystochrone, a été résolu dès 1696 par J. Bernoulli et a motivé le développement de la théorie du calcul variationnel devenue plus tard la théorie de la commande optimale. En effet, les solutions apportées vont être à l'origine des fondements du calcul de variations développé par la suite par Euler, Lagrange et Legendre et complété plus tard par Weierstrass.

Beaucoup plus tard (milieu de vingtième siècle), il commence à apparaître des outils très puissants tels que le principe de programmation dynamique de R. Bellman [Bel03] ou le principe minimum de Pontriaguine [PBG64]. Ces résultats jouent un rôle essentiel en théorie de commande optimale des systèmes linéaires et ont donné naissance à deux approches différentes des problèmes de commande optimale.

La première approche (programmation dynamique) repose sur le principe de Bellman et la caractérisation de la fonction de coût en termes de la solution d'une équation d'Hamilton-Jacobi-Bellman non linéaire [Ber84]. Les méthodes numériques de résolution de l'équation d'Hamilton-Jacobi-Bellman appliquent des schémas de discrétisation en temps et/ou en espace et sont généralement simples à mettre en œuvre. Malheureusement, ces algorithmes demandent un coût très élevé en mémoire.

La seconde approche est basée sur l'utilisation du principe du minimum de Pontriaguine [PBG64], [BYC75], [His04] qui fournit une formulation Hamiltonienne des problèmes de commande ainsi que de conditions nécessaires. Une famille importante de méthodes numériques, qui utilisent ces conditions, est constituée par les méthodes de tir. Ces méthodes très efficaces en toute dimension peuvent être difficiles à mettre en œuvre d'un point de vue théorique et demandent une connaissance a priori de la trajectoire.

Par rapport aux systèmes hybrides, l'écriture d'un principe du minimum hybride est un problème délicat qui a suscité de nombreuses recherches. La principale difficulté est d'arriver à prendre en compte les discontinuités du système

provoquées par les événements discrets et ensuite de les intégrer dans la formulation d'un problème d'optimisation. La théorie de la commande optimale des systèmes hybrides manquant d'un cadre unifié, il existe à l'heure actuelle diverses formulations de ce principe du minimum hybride liées à la définition ainsi qu'à la régularité des systèmes considérés [Sus04], [DK08], [SC03]. Il y a des autres approches comme [RIK03] qui se sert du principe de programmation dynamique et les équations de Hamilton-Jacobi-Bellman pour déduire un principe du minimum valable pour une classe très générale de systèmes hybrides. Une autre approche très intéressante au niveau algorithmique est celle présentée par [AA04] et [AAdW05]. Ces algorithmes appelés « algorithmes de variations fortes » permettent une formulation qui met à jour la commande continue (si elle existe) et la stratégie de commutation. Cette méthode est basée sur l'intégration numérique du système Hamiltonien

Nous nous sommes intéressés à l'utilisation de toute cette théorie de la commande optimale basée sur le principe du minimum et la formulation Hamiltonienne pour les systèmes commutés cycliques. A notre connaissance, la commande optimale de cette classe de systèmes n'a pas été abordée directement. Ce chapitre montre qu'il est possible d'obtenir une loi de commande optimale (sous hypothèse de faible dimension du système) pour des systèmes cycliques par retour d'état et non sous la forme d'une boucle ouverte, garantissant robustesse et haute performance.

Après avoir succinctement présenté des résultats classiques de la commande optimale pour les systèmes dynamiques et pour les systèmes hybrides, nous proposons une approche qui s'appuie sur le principe du minimum. Comme la classe des systèmes qui nous intéresse particulièrement est la classe des systèmes affines en la commande, ces travaux nous amèneront à considérer des solutions particulières appelées arcs singuliers. Des conditions algébriques seront alors énoncées pour déterminer ces arcs. Nous distinguerons dans cette étude le cas d'une ou plusieurs commandes. Nous terminons ce chapitre par un algorithme qui donne la commande par retour d'état. Comme dans le chapitre de la commande prédictive, on applique ces résultats à titre d'illustration sur le convertisseur buck-boost.

3.2 Rappels et Préliminaires

Considérons un système différentiel non-linéaire, variant dans le temps dont l'équation d'état est de la forme :

$$\dot{x}(t) = F(x(t), u(t), t) \quad x(0) = x_0 \quad (3.1)$$

où $x(t) \in \mathbb{R}^n$ ($n \in \mathbb{N}$) désigne l'état, et où les commandes admissibles u sont des fonctions mesurables à valeurs dans un ensemble borné U de $\mathbb{R}^l$. La classe

des commandes admissibles est notée $\mathcal{U}$ par la suite. x_0 est la condition initiale, laquelle est choisie librement dans un ensemble C_0 , $x_0 \in C_0$. Par commodité d'écriture, nous supposons que le temps initial $t_0 = 0$.

On introduit ensuite un critère sous la forme :

$$J(x(t), u(t), t) = \Phi(x(T), T) + \int_0^T L(x(t), u(t), t) dt \quad (3.2)$$

où T ($T \in \mathbb{R}^+ \cup \{\infty\}$) est l'horizon temporel sur lequel la commande est optimisée. La fonction Φ représente un critère terminal à l'arrivée. Le critère (3.2) peut représenter un coût de nature économique, ou une énergie, il peut aussi être, notamment le laps de temps T . Dans ce dernier cas, nous parlons de problème de temps minimal et il est formellement obtenu en posant dans (3.2) $L(x(t), u(t), t) = 1$ et $\Phi(x(T), T) = 0$. Ce critère est un des critères souvent utilisée dans le cadre de la commande optimale.

Le critère J sous la forme générale (3.2) est dit mixte ou de Bolza. Si seul existe le terme intégral ($\Phi(x(T), T) = 0$), le problème s'appelle intégral ou de Lagrange et s'il s'agit d'un critère terminal ($L(x(t), u(t), t) = 0$), le problème est dit terminal ou de Mayer. Également, nous pouvons envisager des contraintes d'égalité ou d'inégalité qui peuvent s'écrire sous la forme :

$$G(x(t), u(t), t) \geq 0 \quad (3.3)$$

Le problème de commande optimale est donc : *Déterminer, si elle existe, une commande $u^* \in \mathcal{U}$ minimisant, un critère J sous la forme donnée par (3.2) qui relie $x(0) = x_0 \in C_0$ à un état final $x(T) = x_f \in C_T$ où C_T est un ensemble donné. Le temps final T peut être fixé ou non. Si $C_T = \mathbb{R}^n$ (ou $C_0 = \mathbb{R}^n$), l'état final $x(T)$ (ou l'état initial $x(0)$) est dit libre.*

Tout au long de ce chapitre, nous ferons les hypothèses suivantes de régularité sur les fonctions F , Φ , L et G .

Hypothèse 2 Les fonctions F , Φ , L et G vérifient :

1. Pour tout couple $(u, t) \in U \times [0, T]$ fixé, les fonctions $F(., u, t)$ et $L(., u, t)$ sont $C^1(\mathbb{R}^n)$.
2. Pour tout $x \in \mathbb{R}^n$, les fonctions $F(x, ., .)$ et $L(x, ., .)$ sont intégrables.
3. $\Phi : \mathbb{R}^n \times [0, T] \rightarrow \mathbb{R}$ est continûment différentiable.
4. La fonction $G : \mathbb{R}^n \times U \times [0, T] \rightarrow \mathbb{R}^q$ ($q \in \mathbb{N}$) est continue sur $\mathbb{R}^n$.
5. C_0 et C_T sont des variétés différentiables de $\mathbb{R}^n$.

Avant de s'intéresser aux méthodes de résolution de ce type de problème, la première question que nous nous posons est celle de l'existence de trajectoires optimales du système (3.1) par rapport au critère considéré. Une fois vérifiée l'existence de solutions optimales, nous nous intéressons à la recherche des commandes optimales. En effet, nous pouvons employer des conditions nécessaires données par le principe du minimum pour obtenir un ensemble de solutions candidates. Ensuite et à partir du problème particulier traité, on recherche des conditions suffisantes comme par exemple l'unicité, donnant la solution optimale parmi l'ensemble des solutions candidates.

3.2.1 Conditions d'existence

Il y a deux questions qui se posent : Premièrement, étant donné le système (3.1), où peut-on aller en temps T en faisant varier la commande u ? Deuxièmement, parmi toutes les trajectoires du système (3.1) qui partent de x_0 et arrivent à x_f (Rappelons que x_f peut être libre), existe-t'il au moins une trajectoire qui minimise le critère (3.2) ?

Rappelons tout d'abord la notion d'ensemble accessible et commandabilité.

Définition 23 (Ensemble accessible [Tre05]) *L'ensemble accessible en temps T est l'ensemble des extrémités au temps T des solutions du système (3.1) partant de x_0 au temps $t = 0$. Notons cet ensemble comme $Acc(x_0, T)$*

Définition 24 (Commandabilité [Tre05]) *Le système (3.1) est dit commandable (en temps quelconque) depuis x_0 si :*

$$\mathbb{R}^n = \bigcup_{T \geq 0} Acc(x_0, T)$$

Il est dit commandable en temps T si $\mathbb{R}^n = Acc(x_0, T)$.

La deuxième question fait référence à l'existence des trajectoires minimisantes. Remarquons que si l'ensemble C_T n'est pas accessible depuis C_0 , alors il n'existe pas de commande u admissible et en conséquence, il n'y a pas de trajectoire du système minimisant (3.2). Le théorème énoncé ci-dessous fournit un résultat d'existence de solutions optimales qui dépend de la régularité du système et du critère.

Théorème 25 (Théorème d'existence [PBGM64], [Cla90], [Tre05], [Ber74]) *Considérons le système (3.1), le critère (3.2) et éventuellement les contraintes (3.3) sous les hypothèses 2. Supposons aussi que la commande u est à valeurs dans un compact $U \subset \mathbb{R}^l$. Soient C_0 et C_T deux ensembles compacts de $\mathbb{R}^n$ tels que C_T est accessible depuis C_0 . Soit $\mathcal{U}$ l'ensemble des commandes à valeurs dans U joignant C_0 à C_T . On suppose que :*

- il existe un réel positif b tel que toute trajectoire associée à une commande $u \in \mathcal{U}$ est uniformément bornée par b sur $[0, T]$, i.e.

$$\exists b > 0, \forall u \in \mathcal{U}, \forall t \in [0, T], \|x(t)\| \leq b \quad (3.4)$$

- pour tout $x \in \mathbb{R}^n$ et $t \in \mathbb{R}$, l'ensemble des vecteurs augmentés :

$$\tilde{V}(x, t) = \{(L(x, u, t), F(x, u, t)); u \in U\} \quad (3.5)$$

est convexe.

Alors il existe une commande optimale u sur $[0, T]$ telle que la trajectoire associée joint C_0 à C_T , en temps T et en coût minimal.

La condition la plus exigeante est bien sûr la propriété de convexité.

3.2.2 Principe du minimum de Pontriaguine

Notons par $(\mathcal{P}_B)$ le problème de Bolza sans contrainte obtenu en considérant le système (3.1), le critère (3.2) et les hypothèses 2.

Nous nous intéressons maintenant à la résolution approchée du problème de commande $(\mathcal{P}_B)$ grâce au principe du minimum de Pontriaguine. En effet, ce théorème permet pour un système classique d'obtenir des conditions nécessaires d'optimalité d'ordre un. Les références concernant ce principe sont nombreuses. Voir [PBGM64], [His04], [VG97], [BYC75], [Ber74], [You69] pour une démonstration.

Soit la fonction Hamiltonienne H des variables λ_0 , λ , x et u définie par :

$$H(t) = H(\lambda_0, \lambda, x, u, t) = \lambda^T F(x, u, t) + \lambda_0 L(x, u, t) \quad (3.6)$$

où $\lambda_0 \geq 0$ et où les dynamiques de λ et de x sont données par un système Hamiltonien :

$$\dot{\lambda} = -\frac{\partial H}{\partial x} \quad \dot{x} = \frac{\partial H}{\partial \lambda} \quad (3.7)$$

λ est souvent appelé variable adjointe ou co-état ou multiplicateur de Lagrange.

Théorème 26 (*Principe du minimum de Pontriaguine*, [PBGM64], [BYC75], [Tre05], [Ber74], [Son98]) Soit le problème $(\mathcal{P}_B)$, si la commande $u^* \in \mathcal{U}$ associée à une trajectoire x^* est optimale sur $[0, T]$, alors il existe une application non triviale $\lambda^* : [0, T] \rightarrow \mathbb{R}^n$ absolument continue appelée vecteur adjoint et un réel $\lambda_0^* \geq 0$ tels que les conditions suivantes sont vérifiées :

1. x^* et λ^* suivent la dynamique du système Hamiltonien presque partout (3.7).
2. La commande u^* minimise la fonction Hamiltonienne (3.6)

$$H^*(t) = H(\lambda_0^*, \lambda^*, x^*, u^*, t) = \min_{u \in U} H(\lambda_0^*, \lambda^*, x^*, u, t) \quad (3.8)$$

3. Pour presque tout $t \in [0, T]$:

$$\frac{d}{dt}H(x^*, \lambda_0^*, \lambda^*, u^*, t) = \frac{\partial}{\partial t}H(x^*, \lambda_0^*, \lambda^*, u^*, t) \quad (3.9)$$

4. λ doit vérifier aux instants initial et final les conditions de transversalité :

$$\lambda^*(0) \perp \Gamma_{x_0}C_0 \quad \lambda^*(T) - \lambda_0^* \frac{\partial}{\partial x}\Phi(x_f, T) \perp \Gamma_{x_f}C_T \quad (3.10)$$

où $\Gamma_{x_0}C_0$, $\Gamma_{x_f}C_T$ représentent les plans tangents aux variétés C_0 et C_T au point x_0 et x_f respectivement.

Remarque 5 Si le temps final T pour joindre C_T n'est pas fixé, nous avons alors la condition au temps final T :

$$\min_{u \in U} H(\lambda_0^*, \lambda^*, x^*, u, T) = -\lambda_0^* \frac{\partial}{\partial t}\Phi(x_f, T) \quad (3.11)$$

Remarque 6 Si la variété C_T s'écrit comme :

$$C_T = \{x \in \mathbb{R}^n, M_1(x) = M_2(x) = \dots = M_\mu(x) = 0\} \quad (3.12)$$

où M_i , $\forall i = 1, \dots, \mu$ sont des fonctions de classe C^1 sur $\mathbb{R}^n$, alors il existe un vecteur π_T de dimension μ et la condition (3.10) peut s'écrire sous la forme :

$$\lambda(T) = \lambda_0 \frac{\partial}{\partial x}\Phi(x_f, T) + \frac{\partial C_T^T}{\partial x} \pi \quad (3.13)$$

Définition 27 (Extrémale normale et anormale) Nous appelons extrémale du problème de commande tout quadruplet $(x^*, u^*, \lambda^*, \lambda_0^*)$ vérifiant les conditions du théorème 26. Si $\lambda_0 = 0$, on dit que l'extrémale est anormale (cas rare en pratique). En revanche, si $\lambda_0 \neq 0$, l'extrémale est dite normale.

Dans la suite de ce manuscrit, nous nous intéressons uniquement aux extrémales pour lesquelles λ_0 est non nul.

Nous faisons les observations suivantes :

- Si le champ de vecteur F , la fonction critère L et le terme sur l'état final Φ ne sont pas fonctions du temps t , alors H ne dépend pas du temps t non plus, et la condition (3.9) devient :

$$H^*(t) = cte \quad (3.14)$$

où cte est une constante. En outre, si T n'est pas fixé, $H^*(T) = 0$ et nous avons donc :

$$H^*(t) = 0 \quad \text{pour } t \in [0, T] \quad (3.15)$$

Notons que la version que nous présentons du principe du minimum, les contraintes d'égalité et/ou d'inégalité de l'équation (3.3) ne sont pas prises en compte. Ce problème est en effet réputé comme difficile. Il existe plusieurs versions de ce principe avec contraintes sur l'état. Pour les inclure, généralement, un vecteur discontinu s'ajoute à l'ensemble de conditions nécessaires à travers des conditions de Kuhn-Tucker, ce qui cause une possible discontinuité dans le vecteur adjoint λ . La théorie est donc beaucoup plus compliquée et nous ne l'abordons pas dans ce mémoire. Pour plus d'information voir [BYC75], [VG97], [Ber84] .

3.2.3 Conditions de transversalité pour les systèmes commutés

Dans la section précédente, nous avons énoncé le principe du minimum de Pontriaguine sous sa forme générale pour des systèmes variant dans le temps.

Lorsque l'on considère des systèmes commutés, des conditions de transversalité intermédiaires aux instants de commutation sont à prendre en compte. Le système modélisé par l'équation (3.1) est remplacé par le modèle du système commuté :

$$\dot{x} = F_{\sigma_i}(x) \quad t \in [t_i, t_{i+1}) \quad i = 0, 1, \dots \quad (3.16)$$

avec $x(0) = x_0 \in C_0$.

Le problème de commande optimale est alors : *Déterminer, si elle existe, la loi de commutation σ^* pour tout $t \in [0, T]$ minimisant le critère :*

$$J(x, t) = \Phi(x_f, T) + \int_0^T L(x, t) dt \quad (3.17)$$

qui relie $x_0 \in C_0$ à un état final $x(T) = x_f \in C_T$.

Supposons que les F_{σ_i} , $\sigma_i \in \mathcal{I} = \{1, 2, \dots, m\}$ vérifient les hypothèses 2.

Soit la fonction Hamiltonienne H_{σ_i} des variables λ_0 , λ , x et t définie par :

$$H_{\sigma_i}(\lambda_0, \lambda, x, t) = \lambda_0 L(x, t) + \lambda^T F_{\sigma_i}(x) \quad (3.18)$$

où $\lambda_0 \geq 0$. Les dynamiques de λ et x suivent alors suivant la valeur de σ_i , les sous systèmes :

$$\dot{x} = \frac{\partial H_{\sigma_i}}{\partial \lambda} \quad \dot{\lambda} = -\frac{\partial H_{\sigma_i}}{\partial x} \quad (3.19)$$

En utilisant le Principe d'optimalité de Bellman (Programmation dynamique), on parvient à la version suivante du Principe du minimum :

Théorème 28 (*[RIK03], [Rie00]*) *Si σ^* est une loi de commutation optimale associée à une trajectoire x^* sur $[0, T]$, alors il existe une application non triviale $\lambda^* : [0, T] \rightarrow \mathbb{R}^n$ absolument continue par morceaux et un réel $\lambda_0^* \geq 0$ tels que :*

1. x^* et λ^* suivent la dynamique du système Hamiltonien (3.19).
2. Aux instants de commutation t_i , (i.e. quand le mode actif change de σ_i^* à σ_{i+1}^*), la condition de transversalité suivante est satisfaite :
 - (a) Si t_i est un instant de saut contraint sur une variété définie par h contraintes ($h < n$) égalités $\mathcal{N}_{\sigma_i}(x(t_i^-), t_i) = 0$, depuis le mode σ_i^* vers le mode σ_{i+1}^* alors il existe un vecteur π_i tel que :

$$\lambda^*(t_i^+) = \lambda^*(t_i^-) - \frac{\partial \mathcal{N}_{\sigma_i}^T(x(t_i^-), t_i)}{\partial x} \pi_i \quad (3.20)$$

$$H_{\sigma_{i+1}^*}(\lambda_0^*, \lambda^*, x^*, t_i^+) = H_{\sigma_i^*}(\lambda_0^*, \lambda^*, x^*, t_i^-) + \frac{\partial \mathcal{N}_{\sigma_i}^T(x(t_i^-), t_i)}{\partial t} \pi_i \quad (3.21)$$

- (b) Si t_i est un instant de saut sans contrainte depuis le mode σ_i vers le mode σ_{i+1} alors :

$$\lambda^*(t_i^+) = \lambda^*(t_i^-) \quad (3.22)$$

$$H_{\sigma_{i+1}^*}(\lambda_0^*, \lambda^*, x^*, t_i^+) = H_{\sigma_i^*}(\lambda_0^*, \lambda^*, x^*, t_i^-) \quad (3.23)$$

3. λ^* doit vérifier aux instants initial et final les conditions de transversalité :

$$\lambda^*(0) \perp \Gamma_{x_0} C_0 \quad \lambda^*(T) - \lambda_0^* \frac{\partial}{\partial x} \Phi_x(x_f, T) \perp \Gamma_{x_f} C_T \quad (3.24)$$

où $\Gamma_{x_0} C_0$, $\Gamma_{x_f} C_T$ représentent les plans tangents aux variétés C_0 et C_T au point $x(0)$ et $x(T)$ respectivement.

Dans le cas où les ensembles C_0 et C_T seraient des variétés de $\mathbb{R}^n$, les conditions de transversalité données dans la remarque 6 (équation (3.13)) s'appliquent.

Remarque 7 *Le cas a) correspond au cas où la trajectoire x franchit la frontière du domaine du mode σ_i^* et la condition de garde est vérifiée pour l'arc $(\sigma_i^*, \sigma_{i+1}^*)$: Commutation autonome. Le cas b) correspond au cas où le mode actif σ est une variable de commande $x \in D_{\sigma_i}$ et $x \in G(\sigma_i, \sigma_{i+1})$: Commutation commandée.*

Remarque 8 Ce principe du minimum peut aussi prendre en compte une commande externe u et la formulation ne change pas. Voir [DK08] ou [RIK03]. Le plus grand inconvénient du théorème 28 est de déterminer les valeurs des constantes π . En effet, π ne suit pas une dynamique particulière et il n'existe pas une méthode systématique pour déterminer sa valeur.

Observons qu'il n'existe pas une condition équivalente à la condition (3.8). En effet, lorsque le domaine de commande discret U dépend de l'état x (dépendance continue ou discrète), une condition du minimum similaire à (3.8) par rapport à la commande discrète doit être réduite à l'ensemble de ses composantes connexes. Deux valeurs de commandes sont connexes s'il n'existe pas de contraintes concernant les commutations entre ces valeurs. L'utilisation de la programmation dynamique à travers le graphe connexe est alors nécessaire pour déterminer l'ensemble des solutions extrémales. Ceci est une des difficultés rencontrées dans la résolution des problèmes de commande optimale sur les systèmes commutés.

3.3 Formulation du problème pour des systèmes cycliques et applications des conditions nécessaires

Dans les sections précédentes, nous avons énoncé deux versions du principe du minimum suivant le type de systèmes à commander. Les deux principes nous permettent d'obtenir des conditions nécessaires d'optimalité. Nous voulons maintenant montrer que lorsqu'il n'existe pas de contraintes sur les sauts, la commande optimale pour un système commuté avec un régime permanent cyclique passe par le principe du minimum classique et l'obtention d'arcs singuliers.

Considérons un système affine en la commande u dont le domaine de commande U est discret et considérons le problème de commande optimale suivant :

$$\begin{aligned} \min_{u(\cdot)} \int_0^T L(z - z_{ref}) dt \\ \text{t.q. } \dot{z} &= R(z) + S(z)u \\ &= R(z) + \sum_{i=1}^r u_i S^i(z) \\ z(0) &= z_0 \quad u \in U = \{0, 1\}^r \end{aligned} \tag{3.25}$$

où $z \in \mathbb{R}^\nu$ ($\nu \in \mathbb{N}$) est l'état du système, les fonctions $R, S^i: \mathbb{R}^\nu \rightarrow \mathbb{R}^\nu, i = 1, \dots, r, (r \in \mathbb{N})$, et $L: \mathbb{R}^\nu \rightarrow \mathbb{R}$. T est le temps mis par la trajectoire issue de z ($T \in \mathbb{R}^+ \cup \{\infty\}$) selon la commande u pour atteindre z_{ref} . u est un vecteur

booléen de dimension r . Il existe donc $m = 2^r$ modes ou 2^r configurations. Nous supposons que $C_0 = z_0$ et que $C_T = \mathbb{R}^\nu$.

Nous faisons l'hypothèse suivante :

Hypothèse 3 *Il n'y a aucune dépendance liée à l'ensemble de commande U (ni temporelle, ni d'état). La commande $u(\cdot) : [0, T] \rightarrow U$ est une fonction mesurable au sens de Lebesgue avec U est un ensemble constant, fini et discret.*

Cette hypothèse implique que la commutation d'un mode à l'autre dépend que de la commande u et non pas de contraintes du type $\mathcal{N}_{\sigma_i}(x(t_i^-), t_i) = 0$. Les commutations autonomes sont donc exclues ; voir le théorème 28.

Si les hypothèses de régularité (2) sont validées, le principe du minimum classique, donné sous la forme du théorème 26, s'applique et est suffisant pour obtenir des conditions nécessaires pour le problème (3.25) car le système est mis sous la forme $\dot{x} = F(x, u)$ (un seul champ de vecteur).

Pour simplifier les écritures d'équations, nous commençons donc par transformer le problème de Lagrange initial sous une forme de Mayer en introduisant une autre variable d'état égale au critère à minimiser :

$$x_n = \int_0^t L(z(\tau)) d\tau \quad (3.26)$$

Nous définissons :

$$\begin{aligned} x &= [z^T, x_n]^T \\ x_0 &= [z_0^T, 0]^T \\ f(x) &= [R^T(z), L(z)]^T \\ g(x) &= [S^T(z), 0]^T \end{aligned}$$

Le problème de commande optimale (3.25) devient :

$$\begin{aligned} &\min_{u(\cdot)} x_n(T) \\ (\mathcal{P}_{MA}) \quad &t.q. \quad \dot{x}(t) = f(x(t)) + g(x(t))u(t) \\ &x(0) = x_0 \quad u(t) \in U = \{0, 1\}^r \end{aligned}$$

où $x \in \mathbb{R}^n$, ($n = \nu + 1$). Ce problème nous le notons comme $(\mathcal{P}_{MA})$ (Problème de Mayer affine en la commande).

Le but est donc : *Déterminer si elle existe, une loi de commande $u^*(t) \in U$ pour $t \in [0, T]$ minimisant $x_n(T)$ pour un état initial x_0 arbitraire.*

3.3.1 Application du principe de minimum

La fonction Hamiltonnienne de $(\mathcal{P}_{MA})$ est donc :

$$H(x, \lambda, u) = \lambda^T (f(x) + g(x)u) \quad (3.27)$$

Les dynamiques de λ et x sont données par l'équation (3.7) ($\dot{x} = \frac{\partial H}{\partial \lambda}$, $\dot{\lambda} = -\frac{\partial H}{\partial x}$). L'application du théorème 26 dans le cadre du problème $(\mathcal{P}_{MA})$ donne :

Corollary 29 (*Principe du minimum pour un système affine commuté*)

Si u^ est une commande optimale pour le problème $(\mathcal{P}_{MA})$ et si x^* désigne la trajectoire associée qui transfère le système de la position x_0 au temps $t = 0$ à une position $x(T)$ au temps final T , alors il existe une application non triviale $\lambda : [0, T] \rightarrow \mathbb{R}^n$ absolument continue, telle que :*

- pour tout t , H atteint son minimum par rapport à la commande u^* , i.e. :

$$H^* = H(\lambda^*, x^*, u^*) = \inf_{u \in U} H(\lambda^*, x^*, u) \quad (3.28)$$

- Pour tout $t \in [0, T]$:

$$H^*(t) = cst \text{ le long de la trajectoire optimale}$$

où cst est une constante et $cst = 0$ si T n'est pas fixé.

- λ vérifie les conditions initiale et finale suivantes :

$$\lambda^*(0) \text{ libre} \quad \text{et} \quad \lambda^*(T) = [0, \dots, 0, 1]^T \quad (3.29)$$

Comme nous l'avons déjà mentionné, les conditions nécessaires du corollaire 29 pré-supposent l'existence d'une solution. Ce n'est pas toujours le cas et notamment lorsque l'on considère des systèmes commutés. Pour se placer dans des conditions où l'on peut garantir l'existence d'une solution optimale, on introduit le problème relaxé [RIK03] qui consiste à étendre le domaine de la commande U à son enveloppe convexe :

$$co(U) = [0 \ 1]^r \quad (3.30)$$

Supposons que ce problème relaxé admette une solution optimale. Par exemple, si F est linéaire et L quadratique, sur le domaine de commande (3.30), la condition du théorème 25, est facilement satisfaite. Autrement, il faut chercher des techniques plus complexes de relaxations des fonctions. Voir [You69], [Rou97], [MPP07] pour des cas où F et L s'expriment comme des polynômes en u ou comme une fonction trigonométrique de u .

Nous pouvons alors faire les observations suivantes : Si une solution de type bang-bang (i.e. $\forall t, u(t) \in \{0, 1\}^r$) est obtenue sur le domaine de commande (3.30),

cette solution résout aussi le problème original $(\mathcal{P}_{MA})$. Sinon, la commande $u(\cdot)$ prend des valeurs dans l'enveloppe convexe, $u(t) \in co(U) \setminus U$, $t \in \mathcal{T}$, où, $\mathcal{T}$ est un ensemble de temps de mesure non null au sens donné par Lesbegue. Bien que cette solution u ne puisse pas s'appliquer au problème initial $(\mathcal{P}_{MA})$, nous pouvons néanmoins l'approcher d'aussi près que nous le voulons par valeur moyenne en commutant entre les différents modes suffisamment vite. Ceci produit un mouvement glissant sur la trajectoire optimale [Utk92]. Une preuve de ce résultat peut être fourni en utilisant un théorème de densité en temps infini [ISW03] (Voir théorème de relaxation 19 du chapitre I).

Nous avons donc intérêt à étudier le problème original $(\mathcal{P}_{MA})$ en utilisant l'enveloppe convexe du domaine de commande (3.30). Ceci revient à étudier les solutions au sens de Filpov de $(\mathcal{P}_{MA})$ [Cor08].

3.3.2 Type de solutions

Considérons maintenant les conditions nécessaires données par le théorème 29. Puisque la fonction Hamiltonienne est affine en la commande u , la fonction :

$$\phi(t) = \lambda^T(t)g(x)$$

définit une fonction de commutation. D'après la condition du minimum (3.28), nous cherchons à minimiser H par rapport à la commande u sachant que u appartient à $\{0, 1\}^r$. En conséquence, les commandes sont :

$$\text{Pour } i = 1, \dots, r \quad u_i(t) = \begin{cases} 0 & \text{si } \phi_i(t) > 0 \\ 1 & \text{si } \phi_i(t) < 0 \\ ? & \text{si } \phi_i(t) = 0 \end{cases} \quad (3.31)$$

Alors, deux types de commande apparaissent :

- i) Considérons t_c comme un temps de commutation d'un mode à l'autre. Si dans le voisinage de t_c , la commande est constante par morceaux avec $u_i = 1$ ou $u_i = 0$, cette commande est dite du type bang-bang ou régulière (les deux premiers cas de l'équation (3.31)). La figure 3.1 montre ce cas. La commande est alors complètement déterminée par le signe de $\phi(t)$. En effet, sa valeur est :

$$u_i(t) = \frac{1 - \text{sign}(\phi(t))}{2}$$

- ii) Il existe un intervalle de temps $[a, b]$, $b > a$ non réduit à $\{t_c\}$ tel que la fonction de commutation est identiquement nulle sur cet intervalle, i.e. pour tout $t \in [a, b]$, $\phi(t) = 0$. Le principe du minimum est insuffisant pour déterminer la valeur de la commande u . Cette situation fait référence à ce qui est connu sous le vocable d'arc singulier [Rob67] (Voir figure 3.2) et elle correspond précisément au cas où $u(t) \in co(U) \setminus U$, $t \in [a, b]$.

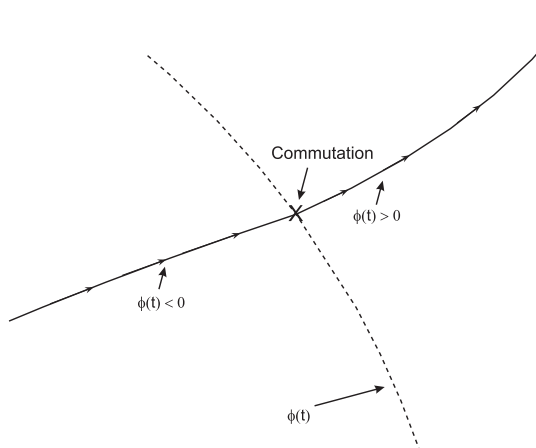


FIG. 3.1 – Trajectoire composée par arcs réguliers. X est le point où un instant de commutation entre $u = 1$ et $u = 0$ se produit.

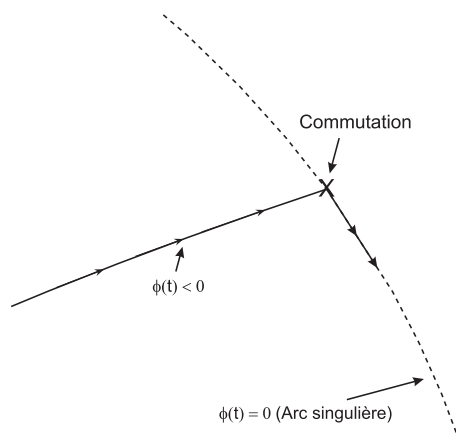


FIG. 3.2 – Trajectoire composée par un arc régulier et un arc singulier. X est le point où un instant de commutation entre $u = 1$ et l'arc singulier se produit.

3.4 Arcs singuliers

Cette section décrit un cas particulier obtenu dans la résolution du problème $(\mathcal{P}_{MA})$. Une littérature importante existe à ce sujet [Rob67], [Kre77], [Pow80], [Vol96], [BBC03], [CJT06], [Ron06].

Nous motiverons notre recherche en nous intéressant aux points de fonctionnement x_{ref} qui sont singuliers par nature. Ces points de fonctionnement expliquent pourquoi il y a de bonnes raisons pour que la commande optimale soit également singulière.

Notre but dans un premier temps est de faire une synthèse des principaux résultats concernant ces arcs singuliers. Puis, nous montrerons comment il est possible pour des systèmes d'ordre faible de faire une détermination explicite de ces arcs. Nous établirons un ensemble de conditions algébriques plus faciles à résoudre et nous proposerons des algorithmes de synthèse conduisant à une loi de commande u fonction de l'état x .

3.4.1 Points de fonctionnement et arcs singuliers

Pour les systèmes commutés avec un régime permanent cyclique, il y a une bonne raison qui plaide pour la recherche d'arcs singuliers : la nature des points de fonctionnement x_{ref} du système. Comme nous l'avons mentionné à la section

1.4.2 du Chapitre I, ces points de fonctionnement sont généralement la valeur moyenne de x sur le cycle et sont définis comme les points d'équilibre du modèle d'état moyen :

$$X_{ref} = \{x_{ref} \in \mathbb{R}^n : f(x_{ref}) + g(x_{ref})u_{ref} = 0, u_{ref} \in co(U)\}$$

Comme $u_{ref} \in co(U)$, ce modèle est exactement le modèle utilisé dans le problème relaxé ($\dot{x} = f(x) + g(x)u$, $u(t) \in co(U)$). De plus, comme la commande u_{ref} permettant le maintien sur le point d'équilibre x_{ref} est généralement telle que $u_{ref} \in co(U) \setminus U$, cette commande définit clairement une commande singulière pour le problème de commande optimale. La solution optimale a donc de grande chance de devenir singulière dans un voisinage du point de fonctionnement.

Par clarté, nous allons traiter uniquement du cas mono-dimensionnel ($r = 1$) dans un premier temps avant d'étudier le cas où $r > 1$.

3.4.2 Une seule commande singulière ($r = 1$)

Posons d'abord quelques définitions :

3.4.2.1 Résultats de base et condition nécessaire d'ordre deux

Définition 30 (*Arc singulier [Kre77], [Pow80], [Vol96]*) Si (x, λ, u) est une extrémale (solution vérifiant les conditions nécessaires du principe du minimum) telle que

$$\phi(t) = H_u = \frac{\partial H}{\partial u} \equiv 0 \quad (3.32)$$

sur un intervalle $[a, b] \subset T$, $a < b$ alors (x, λ, u) est un arc singulier sur cet intervalle.

Dans cette situation, la fonction Hamiltonienne est indépendante de la commande u et la condition de minimum (3.28) ne permet pas d'établir directement la commande u . Afin de déterminer sous quelles conditions l'égalité (3.32) peut avoir lieu, on peut procéder par dérivation de $\phi(t) \equiv 0$ sur $[a, b]$, $b < a$. Les dérivées successives sont aussi identiquement nulles sur l'intervalle i.e.

$$\frac{d}{dt}\phi(t) = 0, \quad \frac{d^2}{dt^2}\phi(t) = 0, \dots$$

Nous définissons alors les grandeurs suivantes :

Définition 31 (*Ordre du problème [Rob67], [Vol96]*) L'ordre du problème est le plus petit entier q tel que u apparait explicitement dans la dérivée $(2q)$ -ième. Le calcul s'effectue en substituant $\dot{x}$ et $\dot{\lambda}$ par leur expression données par (3.7). $(\phi^{(k)} = \frac{d^k}{dt^k}\phi(t))$ est la dérivée d'ordre k par rapport au temps t .

Dans notre cas, comme le problème est affine en u , la commande entre linéairement dans l'expression et nous avons donc

$$\phi^{(2q)} = A(x, \lambda) + B(x, \lambda)u \quad (3.33)$$

Définition 32 (*Ordre de l'arc* [Rob67], [Vol96]) Si (x^*, λ^*, u^*) est une extrémale singulière définie sur $[a, b] \subset T$ pour $(\mathcal{P}_{MA})$. L'ordre de l'arc (x^*, λ^*, u^*) est le plus petit entier p tel que

$$(\phi^{(2p)})_u = ((H_u)^{(2p)})_u(x^*, \lambda^*, u^*) \neq 0 \quad (3.34)$$

pour tout $t \in [a, b]$. Lorsque $k = \infty$, l'ordre de l'arc n'existe pas.

Notons que si $B(x, \lambda) = 0$, l'ordre de l'arc q est différent de l'ordre de problème p . Il peut donc se présenter des arcs d'ordre différents pour un même ordre de problème. Remarquons également que l'ordre de l'arc ne dépend pas uniquement de l'extrémale (x^*, λ^*, u^*) , mais aussi de l'intervalle $[a, b]$. Un cas particulier exceptionnel peut se produire lorsque l'ordre de l'arc change en tout point de (x, λ) . Nous supposons donc par la suite que :

Hypothèse 4 Pour tout point d'un arc singulier, il existe un intervalle de temps ouvert sur lequel l'ordre de l'arc est constant.

Dans [Rob67], l'auteur montre un cas isolé pour lequel cette hypothèse n'est pas vérifiée.

Les définitions 30, 31 et 32 sont consistantes avec la condition nécessaire suivante :

Proposition 33 ([Rob67], [Vol96]) L'ordre de l'arc et l'ordre du problème sont obtenus à partir des dérivées d'ordre pair de ϕ .

Lorsque l'ordre de l'arc existe, nous pouvons établir les équations suivantes le long d'une extrémale singulière (x, λ, u) :

$$\phi^{(i)} = 0, i = 0, \dots, 2p - 1 \quad (3.35)$$

$$\frac{\partial}{\partial u} \phi^{(i)} = 0, i = 0, \dots, 2p - 1 \quad (3.36)$$

$$\phi^{(2p)} = A(x, \lambda) + B(x, \lambda)u = 0 \quad (3.37)$$

$$\text{avec } B(x, \lambda) \neq 0 \text{ pour tout } t \in [a, b] \quad (3.38)$$

La commande est donc déterminée par l'équation (3.37) et elle est définie de manière unique par :

$$u = -A(x, \lambda)/B(x, \lambda) \quad (3.39)$$

Les dérivées d'ordre supérieur de ϕ permettent d'établir les équations différentielles non linéaires suivies par u et ses dérivées sur l'intervalle $[a, b]$. Mais aucune information supplémentaire pour la détermination des x et λ candidats ne peut être obtenue.

En revanche, l'ensemble des solutions de (3.35), (3.36) et (3.37) peut être réduit en validant une condition nécessaire d'optimalité d'ordre deux :

Théorème 34 (*Condition de Legendre Clebsh Généralisée [Rob67], [Kre77]*)

Soit le problème $(\mathcal{P}_{MA})$. Si (x^*, λ^*, u^*) est un arc singulier optimal, alors

1. Si l'ordre du problème est q sur $[a, b]$, alors l'arc vérifie les propriétés suivantes :

$$(-1)^q ((H_u)^{(2q)})_u \geq 0 \quad (3.40)$$

pour tout $t \in [a, b]$

2. Si l'ordre de l'arc $p < +\infty$ sur $[a, b]$, alors

$$(-1)^p ((H_u)^{(2p)})_u > 0 \quad (3.41)$$

pour tout t sur $[a, b]$.

3.4.2.2 Une méthode pratique pour l'obtention des arcs singuliers pour $r = 1$

Il n'est généralement pas simple de résoudre les équations non-linéaires fournies par (3.35), (3.36) et (3.37). Nous allons montrer à la fin de cette section que pour des systèmes d'ordre faible, il est possible de déterminer explicitement les valeurs de (x, λ, u) telles que (3.35), (3.36) et (3.37) soient vérifiés. La méthode est basée sur des conditions algébriques directement déduites des résultats de base montrés précédemment et aussi sur des hypothèses concernant l'ordre du problème et la dimension de l'espace d'état.

Commençons par déterminer formellement l'ordre du problème pour $(\mathcal{P}_{MA})$. Il suffit de dériver successivement la fonction de commutation $\phi(\cdot)$ par rapport au temps, le nombre nécessaire de fois, pour que la commande u apparaisse explicitement. Pour un système affine $(\mathcal{P}_{MA})$ et un ordre de problème égal à q , les dérivées s'écrivent donc sous la forme :

$$\begin{aligned} \lambda^T \text{ad}_f^k g(x) &= 0, k = 0, \dots, 2q - 1 \\ \lambda^T \text{ad}_f^{2q} g(x) + \lambda^T [g, \text{ad}_f^{2q-1} g](x) u &= 0 \end{aligned} \quad (3.42)$$

où $[f, g](x) := g_x(x)f(x) - f_x(x)g(x)$ est le crochet de Lie et $\text{ad}_f^k g(x) = [f(x), \text{ad}_f^{k-1} g(x)]$, $\text{ad}_f^0 g(x) = g(x)$ le crochet itéré de Lie d'ordre k .

Nous sommes maintenant en mesure d'énoncer le lemme suivant :

Lemme 35 ([Vol96], [FA89a]) Soit $(\mathcal{P}_{MA})$. Si $p = q$, les expressions $ad_f^k g(x)$, $k = 0, \dots, 2q - 1$ sont linéairement indépendantes le long de la solution extrême (x, λ, u) et $2q \leq n$,

Preuve. Nous voulons vérifier les deux affirmations :

1. *Indépendance linéaire de $ad_f^k g(x)$, $k = 0, \dots, 2q - 1$* : Supposons qu'il existe $\{\beta_0, \beta_1, \dots, \beta_{2q-1}\}$ tels qu'il existe une combinaison linéaire :

$$\beta_0 ad_f^0 g(x) + \beta_1 ad_f^1 g(x) + \dots + \beta_{2q-1} ad_f^{2q-1} g(x) = 0 \quad (3.43)$$

Par la linéarité du crochet de Lie :

$$\beta_0 [g, ad_f^0 g](x) + \beta_1 [g, ad_f^1 g](x) + \dots + \beta_{2q-1} [g, ad_f^{2q-1} g](x) = 0 \quad (3.44)$$

L'ordre du problème implique $[g, ad_f^k g](x) = 0$ pour $k = 0, \dots, 2q - 2$ et $[g, ad_f^{2q-1} g](x) \neq 0$. Par conséquent, $\beta_{2q-1} [g, ad_f^{2q-1} g](x) = 0$ et forcément β_{2q-1} a une valeur nulle.

Appliquons maintenant le crochet de Lie à (3.43) par rapport à f , nous avons :

$$\beta_0 ad_f^1 g(x) + \beta_1 ad_f^2 g(x) + \dots + \beta_{2q-2} ad_f^{2q-1} g(x) = 0 \quad (3.45)$$

Et le crochet de Lie par rapport à g :

$$\beta_0 [g, ad_f^1 g](x) + \beta_1 [g, ad_f^2 g](x) + \dots + \beta_{2q-2} [g, ad_f^{2q-1} g](x) = 0$$

Suivant le même raisonnement $\beta_{2q-2} = 0$.

Récursivement, il s'en suit que toutes les constantes $\beta_0 = \beta_1 = \dots, \beta_{2q-1} = 0$.

2. $2q \leq n$: Un ensemble de $n + 1$ vecteurs, $(\{ad_f^0 g(x), ad_f^1 g(x), \dots, ad_f^n g(x)\})$ est nécessairement linéairement dépendant. Il existe donc des fonctions non identiquement nulles $\beta_i(x)$ telles que :

$$\beta_0(x) ad_f^0 g(x) + \beta_1(x) ad_f^1 g(x) + \dots + \beta_n(x) ad_f^n g(x) = 0 \quad (3.46)$$

Si $2q > n$, alors d'après 1, $\beta_0(x) = \beta_1(x) = \dots = \beta_n(x) = 0$. Ce qui permet de conclure que $2q \leq n$.

■

Bien que cette proposition soit très utile lorsque $p = q$, il n'existe pas de borne supérieure pour la valeur de p et l'indépendance linéaire ne peut pas être prouvée au-delà des dérivées d'ordre $2q - 1$ car $[g, ad_f^k g](x) \neq 0$ pour $k = 2q, \dots, 2p - 1$. Par conséquent le nombre de dérivées nécessaires pour déterminer la commande n'est pas *a priori* connu.

A partir de l'ordre $2p$, nous observons que les dérivées par rapport au temps de ϕ , i.e. $\phi^{(k)}$, $k > 2p$, font intervenir les dérivées successives de la commande u . Ces équations additionnelles $\phi^{(k)}$, $k \geq 2p$ n'apportent pas directement de conditions supplémentaires pour déterminer (x, λ) . Notons que les dérivées de u par rapport au temps sont parfaitement justifiées par l'expression (3.39). L'équation différentielle $\phi^{(2p+1)} = 0$ donne alors la dynamique de la commande singulière u . Seul les équations $\phi^{(k)} = 0$, $k = 0, \dots, 2p$ sont nécessaires pour déterminer (x, λ, u) et les dérivées additionnelles donnent $u^{(i)}$, $i = 1, 2, \dots$

La présence de λ dans les équations (3.42), ne facilite pas le calcul des solutions. Cependant, il est possible pour des systèmes d'ordre faible d'éliminer cette variable des équations et ainsi de proposer une méthode pratique pour trouver le lieu des arcs singuliers.

En complétant l'ensemble des équations (3.42) avec ses dérivées $\phi^{(k)}$ jusqu'à l'ordre $n - 1$, le système (3.42) s'écrit de manière compacte comme :

$$\lambda^T [g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x), N_0, \dots, N_i, \dots, N_{n-2q-1}] = 0 \quad (3.47)$$

avec N_i , pour $i = 0, \dots, n - 2q - 1$ défini par

$$N_i(x, u) = ad_f^{2q+i} g(x) + \sum_{k=0}^i \Upsilon^{(i-k)}([g, ad_f^{2q+k-1} g](x)u(t)) \quad (3.48)$$

et l'opérateur Υ défini par

$$\Upsilon(*) = (-f_x - u(t)g_x)(*) + \frac{d(*)}{dt} \quad (3.49)$$

Comme λ n'est pas identiquement nul (voir le principe du minimum), nous avons la proposition suivante :

Proposition 36 (*Conditions nécessaires algébriques pour les arcs singuliers*) Soit (x, u) une solution extrémale singulière au problème $(\mathcal{P}_{MA})$ avec un ordre du problème q tel que $2q < n$. Le lieu de (x, u) satisfait l'équation algébrique :

$$S(x, y_0, \dots, y_{(n-2q-1)}) = 0 \quad (3.50)$$

où le vecteur

$$(y_0, \dots, y_{(n-2q-1)}) = (u, \dot{u}, u^{(2)}, \dots, u^{(n-2q-1)})$$

et

$$S = \det([g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x), N_0, N_1, \dots, N_{n-2q-1}]) \quad (3.51)$$

avec $N_i(x, u)$ défini par (3.48).

La proposition 36 est plus aisée à résoudre que (3.35), (3.36), (3.37), non seulement parce que les produits avec λ disparaissent, mais aussi parce que le nombre d'inconnues est réduit à $2(n - q)$.

Les racines du déterminant 3.51 fournissent la commande u en fonction de x et des dérivées de u . Il n'y a donc pas unicité des candidats à la singularité en fonction de x .

Examinons la proposition 36 de plus près et supposons que $p = q$ le long de l'arc.

Lorsque $n = 2q + 1$, nous obtenons l'expression suivante :

$$\det([g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x), ad_f^{2q} g(x) + u[g, ad_f^{2q-1} g](x)]) = 0.$$

Comme $ad_f^k g(x)$, $k = 0, \dots, 2q-1$ sont linéairement indépendants (Lemma 35) et par la propriété de multi-linéarité du déterminant, la commande est uniquement déterminée par :

$$u = -\frac{\det([g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x), ad_f^{2q} g(x)])}{\det([g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x), [g, ad_f^{2q-1} g](x)])}.$$

Malheureusement, la plupart des problèmes de commande ont un ordre de problème $q = 1$ ce qui limite la détermination des commandes candidates u en fonction de x au cas où $n = 3$. Néanmoins, des conditions additionnelles peuvent être prises en considération si nous traitons des problèmes dans lesquels il est nécessaire que la fonction Hamiltonienne s'annule. C'est effectivement le cas pour des critères très utilisés en pratique (Commande en temps optimale ou critères quadratiques en temps infini).

Dans ce cas, nous disposons de la condition suivante :

$$\lambda^T f(x) = 0 \tag{3.52}$$

Ce qui nous amène aux propositions suivantes :

Proposition 37 (Conditions nécessaires algébriques pour les arcs singuliers) *Supposons que $p = q$, $n = 2(q + 1)$ et que la condition $H = 0$ est imposée par le principe du minimum. Si $f(x)$ est linéairement indépendant de $ad_f^k g(x)$, $k = 0, \dots, 2q - 1$ le long de l'extrémale (x, λ, u) , alors la commande singulière u est uniquement déterminée par le retour d'état :*

$$u = -\frac{\det([f(x), g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x), ad_f^{2q} g(x)])}{\det([f(x), g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x), [g, ad_f^{2q-1} g](x)])} \tag{3.53}$$

et λ vérifie :

$$\lambda^T [f(x), g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x)] = 0. \tag{3.54}$$

Proposition 38 (*Conditions nécessaires algébriques pour les arcs singuliers*) Supposons que $p = q$ et $n = 2q + 1$. La commande singulière u est uniquement déterminée par le retour d'état :

$$u = -\frac{\det([g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x), ad_f^{2q} g(x)])}{\det([g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x), [g, ad_f^{2q-1} g](x)])} \quad (3.55)$$

et λ vérifie la relation suivante :

$$\lambda^T [g(x), ad_f^1 g(x), \dots, ad_f^{2q-1} g(x)] = 0. \quad (3.56)$$

Par la propriété d'homogénéité λ est choisie tel que $\|\lambda\| = 1$

Les cas pour lesquels $n < 2q$ n'existent pas à cause du lemma 35. Comme nous l'avons déjà dit, $q = 1$ est le cas plus fréquent. Nous pouvons donc utiliser les propositions 37 et 38 pour $n = 3$ ou 4. Le cas $n = 2$ est trivialement résolu d'après (3.42).

Remarque 9 Dans la pratique lorsque $n = 2(q + 1)$ ou $n = 2q + 1$, pour assurer que $p = q$, il suffit de vérifier que $p \geq q + 1$ est un cas impossible. Une manière de procéder est de montrer que l'espace $E = \text{span}\{g(x), ad_f^1 g(x), \dots, ad_f^{2q+1} g(x)\}$ est de rang complet n . En effet, si $p = q + 1$, d'après la relation (3.35),

$$\lambda^T [g(x), ad_f^1 g(x), \dots, ad_f^{2q+1} g(x)] = 0$$

si la dimension de l'espace E est égal à n alors une contradiction apparaît, car le vecteur λ devient nul et par conséquent $p > q$ est impossible.

Nous proposons donc de déterminer les arcs singuliers à l'aide de l'algorithme suivant :

Proposition 39 (*Algorithme pour l'obtention des arcs singuliers*). Si les hypothèses des propositions 37 ou 38 sont vérifiées, on obtient les arcs singuliers candidats suivant la démarche :

1. Déterminer l'ensemble des commandes possibles $u(x)$ et $\lambda(x)$ d'après les propositions 37 ou 38
2. Réduire cet ensemble aux solutions $u(x) \in \text{co}(U)$.
3. Stocker uniquement les solutions qui satisfont la condition de deuxième ordre du théorème 34.

3.4.3 Cas multi-commande $r > 1$

Dans cette partie, nous analyserons le cas concernant les systèmes qui ont plus d'une entrée de commande $r > 1$. Comme des nouvelles conditions apparaissent, la situation pour $r > 1$ est plus complexe. Citons les méthodes numériques proposées dans la littérature :

- Une technique de pénalisation dont un terme $\int_0^T (\epsilon/2)u^2 dt$ est ajouté à la fonction à minimiser en $(\mathcal{P}_{MA})$ est développée dans [MM71], [JGL70], [Pop05]. Bien que cette technique s'applique pour $r = 1$ ou $r > 1$ indifféremment, elle n'est pas bien conditionnée lorsque ϵ tend vers zéro.
- Des algorithmes de tir pour déterminer la solution du $(\mathcal{P}_{MA})$ pour une valeur initiale donnée x_0 sont proposées dans [FA89b], [Mau76]. Cette méthode pré-suppose la connaissance du nombre d'arcs singuliers présents dans la solution et nécessite une estimation initiale de la valeur de λ .

Dans cette section nous montrons qu'il est possible d'établir ces arcs d'une manière exacte en utilisant des conditions algébriques similaires à celle de la proposition 36 pour $r > 1$. Comme pour le cas concernant $r = 1$, nous commençons cette section par présenter quelques définitions et résultats généraux existants dans la littérature. Ensuite, nous montrons la démarche pour établir une loi de retour d'état $u(x)$.

3.4.3.1 Résultats de base et condition nécessaire d'ordre deux

Une fonction de commutation ϕ est un vecteur de dimension r tel que :

$$\begin{aligned}\phi(x, \lambda) &= [\phi_1(x, \lambda), \dots, \phi_r(x, \lambda)]^T \\ &= H_u(x, \lambda, u) \\ &= [g^1(x), \dots, g^r(x)]^T \lambda\end{aligned}$$

Un arc singulier apparaît quand, au moins une composante de ϕ est identiquement nulle sur un intervalle de temps de mesure non nulle (Lebesgue mesurable).

Supposons qu'il existe un sous ensemble non vide $\mathcal{R}$ de $\{1, 2, \dots, r\}$ tel que

$$\begin{aligned}\phi_i(t) &\equiv 0, \forall i \in \mathcal{R} \text{ sur } [a, b] \\ \phi_i(t) &\neq 0, \forall i \notin \mathcal{R} \text{ sur } [a, b]\end{aligned}\tag{3.57}$$

La dernière équation définit les commandes régulières dont les valeurs sont déterminées par la condition d'optimalité du principe du minimum (3.8). La condition de minimum détermine les commandes ($u_i(t) = 0$ ou 1) en fonction du signe de la fonction de commutation $\phi_i, \forall i \notin \mathcal{R}$ en $[a, b]$.

Comme les commandes u_i , $i \notin \mathcal{R}$ n'interviennent pas directement dans la détermination des commandes singulières (elles sont déjà déterminées), les termes $g^i u_i$, $i \notin \mathcal{R}$ s'ajoutent au terme f et nous noterons par la suite f_{drift} , cet ensemble. Sans perdre en généralité et pour simplifier la discussion, nous identifierons les éléments de $\mathcal{R}$ à l'ensemble $\{1, 2, \dots, r\}$ tout entier.

Définition 40 (Matrice de l'ordre du problème [Vol96]) Soit $(\mathcal{P}_{MA})$. La matrice de l'ordre du problème Q est une matrice composée par des éléments (q_{ij}) , $i, j = 1, \dots, r$. (q_{ij}) est le nombre pour lequel la commande u_j apparaît explicitement et pour la première fois dans la $2q_{ij}$ -ème dérivée de la fonction de commutation ϕ_i par rapport au temps t .

Définition 41 (Matrice de l'ordre de l'arc [Vol96]) Si (u^*, x^*, λ^*) est une extrémale singulière de $(\mathcal{P}_{MA})$, alors la matrice de l'ordre de l'arc P est une matrice dont les éléments $p_{i,j}$, $i, j = 1, \dots, r$ ont la forme :

$$p_{ij} := \min_k \{k : k \geq 0, (\phi_i^{(2k)})_{u_j}(u^*, x^*, \lambda^*) \neq 0\} \quad (3.58)$$

Définition 42 (Ordre du problème et ordre de l'arc [Rob67], [Vol96]) L'ordre du problème q est défini comme :

$$q = \min\{q_{ij}\}, \quad i, j = 1, \dots, r \quad (3.59)$$

et l'ordre de l'arc :

$$p = \min\{p_{ij}\}, \quad i, j = 1, \dots, r \quad (3.60)$$

Comme en dimension 1, et pour tout point d'un arc singulier, nous supposons qu'il est toujours possible de trouver un intervalle de temps sur $[a, b]$ sur lequel la matrice de l'ordre de l'arcs est constante.

De la définition 42, [Rob67], [Kre77], [Pow80], [Mel94] proposent aussi des conditions nécessaires d'optimalité d'ordre supérieur. Nous nous en servons pour distinguer les commandes optimales de celles qui ne le sont pas, mais qui satisfont les conditions de premier ordre (3.57).

Comme pour $r = 1$, il existe aussi des conditions du second ordre :

Théorème 43 (Condition généralisée Legendre Clebsh GLC [Rob67]) Si (x^*, λ^*, u^*) est un arc optimal singulier avec un ordre de l'arc $p < +\infty$ sur $[a, b]$. alors la matrice $(-1)^p ((H_u)^{(2p)})_u$ est semi-définie positive pour chaque t sur $[a, b]$.

[Kre77], [Vol96], [JGL70] montrent des conditions nécessaires de troisième degré pour vérifier l'optimalité de problèmes singuliers.

Lorsque $r = 1$, l'ordre du problème et l'ordre de l'arc sont forcément des nombres entiers tandis que ce n'est pas le cas pour $r > 1$. En contrepartie, la condition de second ordre du théorème 43 donne lieu à la proposition :

Proposition 44 [Rob67] *L'ordre de l'arc d'un arc optimal singulier est un nombre entier.*

[Vap67] a également prouvé cette proposition pour le cas particulier de $2q = 1$.

3.4.3.2 Une méthode pratique pour l'obtention des arcs singuliers lorsque $r > 1$

Comme pour le cas $r = 1$, dans cette partie, nous proposerons une formulation algébrique afin de déterminer les arcs singuliers.

Rappelons les conditions propres à l'obtention d'arcs singuliers :

- Il existe un sous-ensemble de composantes, $\mathcal{R}$, de la fonction de commutation qui s'annulent identiquement sur un intervalle $[a, b] \subset T$, $a < b$.
- Les commandes régulières u_i (pour les indices $i \notin \mathcal{R}$) sont fixées a priori à 0 ou 1 et on pose $f_{drift} = f + \sum_{i \notin \mathcal{R}} g^i u_i$.
- Considérant la fonction $\phi(x, \lambda)$ de commutation restreinte aux composantes $i \in \mathcal{R}$, le problème de commande singulière ($\mathcal{P}_{MA}$) revient à résoudre $\phi(x, \lambda) \equiv 0$ sur un intervalle de temps $[a, b] \subset T$ avec l'aide des conditions nécessaires de deuxième ordre.

Comme $g(x) = \sum_{i \in \mathcal{R}} g^i(x) u_i$, nous calculons les dérivées des composantes de la fonction de commutation analytiquement jusqu'à l'ordre $2q$:

$$\begin{aligned} \phi_i^{(k)}(x, \lambda) &= \lambda^T ad_{f_{drift}}^k g^i(x) \\ &= 0, k = 0, \dots, 2q - 1 \text{ et } i \in \mathcal{R} \end{aligned} \quad (3.61)$$

$$\begin{aligned} \phi_i^{(2q)}(x, \lambda) &= \lambda^T ad_{f_{drift}}^{2q} g^i(x) + \lambda^T \sum_{j \in \mathcal{R}} [g^j, ad_{f_{drift}}^{2q-1} g^i](x) u_j \\ &= 0, i \in \mathcal{R} \end{aligned} \quad (3.62)$$

Supposons maintenant qu'il existe une solution extrême (x, λ, u) pour laquelle l'ordre de l'arc est égal à p , et $2p > 2q$, nous avons donc des conditions additionnelles sous la forme :

$$\frac{\partial}{\partial u_j} \phi_i^{(2q+k)}(x, \lambda) = \lambda^T [g^j, ad_{f_{drift}}^{2q-1+k} g^i](x) \quad (3.63)$$

$$= 0, i \in \mathcal{R}, j \in \mathcal{R}, k = 0, \dots, 2p - 2q - 1$$

$$\phi_i^{(k)}(x, \lambda) = \lambda^T ad_{f_{drift}}^k g^i(x) \quad (3.64)$$

$$= 0, i \in \mathcal{R}, k = 2q + 1, \dots, 2p - 1$$

$$\phi_i^{(2p)}(x, \lambda) = \lambda^T (ad_{f_{drift}}^{2p} g^i(x) + \sum_{j \in \mathcal{R}} [g^j, ad_{f_{drift}}^{2p-1} g^i](x) u_j) \quad (3.65)$$

$$= 0, i \in \mathcal{R}$$

Finalement, les dérivées d'ordre supérieur pour $k \geq 1$ sont formées par l'expression :

$$\begin{aligned} \phi_i^{(2p+k)}(x, \lambda) &= \lambda^T (ad_f^{2p+k} g^i(x) + \sum_{l \in \mathcal{R}} \sum_{j=0}^k \Upsilon^{(k-j)}([g^l, ad_f^{2p+j-1} g^i](x) u_l(t)) \\ &= 0 \end{aligned} \quad (3.66)$$

où l'opérateur Υ est défini comme

$$\Upsilon(*) = -(f_x + \sum_{j \in \mathcal{R}} g_x^j u_j)(*) + \frac{d(*)}{dt} \quad (3.67)$$

A présent, nous pouvons faire les observations suivantes :

1. Toutes les dérivées prennent la forme $\lambda^T h_k(x, u)$ jusqu'à ce que l'ordre $2p$ soit atteint. Pour $k > 2p$, les dérivées successives par rapport au temps de u entrent dans les expressions de h_k .
2. La variable adjointe λ est orthogonale au sous espace vectoriel généré par la famille de champs de vecteurs $\{h_k(x, u), k = 0, 1, \dots\}$
3. Comme le vecteur adjoint λ n'est pas trivialement nul, les solutions singulières sont obtenues lorsque la dimension de l'espace est inférieure à n .
4. Le nombre des dérivées $2p$ à prendre en considération est a-priori inconnu et est directement lié au nombre de champs de vecteurs linéairement indépendant h_k pour un (x, u) donné.
5. Si pour un ordre de dérivation donné, toutes les composantes de u sont déterminées (en fonction de x), alors les dérivées supplémentaires déterminent un ensemble complet d'équations différentielles nonlinéaires impliquant u et ces dérivées. Il n'est donc pas nécessaire de dériver au delà pour déterminer les valeurs de x candidates.

Supposons maintenant que $p = q$, d'après les équations (3.61) et (3.62) et pour un x donné, on compte $(n + |\mathcal{R}|)$ inconnues pour $(2q + 1)|\mathcal{R}|$ équations. Comme λ n'est pas trivialement nul si $2q + 1$ est égal à n , nous obtenons pour $i \in \mathcal{R}$:

$$\det([g^i(x), ad_{f_{drift}}^1 g^i(x), \dots, ad_{f_{drift}}^{2q-1} g^i(x), ad_{f_{drift}}^{2q} g^i(x) + \sum_{j \in \mathcal{R}} [g^j, ad_{f_{drift}}^{2q-1} g^i](x) u_j]) = 0, .$$

Par la propriété de multi-linéarité du déterminant :

$$\begin{aligned} & \det([g^i(x), ad_{f_{drift}}^1 g^i(x), \dots, ad_{f_{drift}}^{2q-1} g^i(x), ad_{f_{drift}}^{2q} g^i(x)]) \\ & + \sum_{j \in \mathcal{R}} \det([g^i(x), ad_{f_{drift}}^1 g^i(x), \dots, ad_{f_{drift}}^{2q-1} g^i(x), [g^j, ad_{f_{drift}}^{2q-1} g^i](x)]) u_j = 0, \quad i \in \mathcal{R}. \end{aligned} \quad (3.68)$$

Définissons $[\det_{(i,j)}(x)]$ la matrice $|\mathcal{R}| \times |\mathcal{R}|$ dont les entrées $(i, j) \in M^2$ sont :

$$\det_{(i,j)}(x) = \det([g^i(x), ad_{f_{drift}}^1 g^i(x), \dots, ad_{f_{drift}}^{2q-1} g^i(x), [g^j, ad_{f_{drift}}^{2q-1} g^i](x)])$$

Si $[\det_{(i,j)}(x)]$ est inversible, il suffit de déterminer la commande par

$$u = [\det_{(i,j)}(x)]^{-1} W(x)$$

où l'entrée i -th du vecteur $W(x)$ est donnée par l'expression :

$$W_i(x) = \det([g^i(x), ad_{f_{drift}}^1 g^i(x), \dots, ad_{f_{drift}}^{2q-1} g^i(x), ad_{f_{drift}}^{2q} g^i(x)]).$$

De manière similaire au cas où $m = 1$, nous pouvons déduire la proposition suivante :

Proposition 45 (Conditions nécessaires algébriques pour les arcs singuliers) *Supposons $p = q$, $n = 2(q + 1)$ et que la condition $H = 0$ est imposée par le principe du minimum. Si $f_{drift}(x)$ est linéairement indépendant de $ad_{f_{drift}}^k g(x)$, $k = 0, \dots, 2q - 1$ le long de l'arc singulier (x, λ, u) et $[\det_{(i,j)}(x)]$ est inversible, alors le vecteur de commande singulière u est uniquement déterminé par le retour d'état :*

$$u = [\det_{(i,j)}(x)]^{-1} W(x) \quad (3.69)$$

où $[\det_{(i,j)}(x)]$ est la matrice $|\mathcal{R}| \times |\mathcal{R}|$ dont les entrées $(i, j) \in \mathcal{R}^2$ sont

$$\det_{(i,j)}(x) = \det([f_{drift}, g^i(x), ad_{f_{drift}}^1 g^i(x), \dots, ad_{f_{drift}}^{2q-1} g^i(x), [g^j, ad_{f_{drift}}^{2q-1} g^i](x)])$$

et où l'entrée i -th du vecteur $W(x)$ est donnée par :

$$W_i(x) = \det([f_{drift}, g^i(x), ad_{f_{drift}}^1 g^i(x), \dots, ad_{f_{drift}}^{2q-1} g^i(x), ad_{f_{drift}}^{2q} g^i(x)]). \quad (3.70)$$

De plus, λ vérifie

$$\lambda^T [f_{drift}(x), g^i(x), ad_{f_{drift}}^1 g^i(x), \dots, ad_{f_{drift}}^{2q-1} g^i(x)] = 0. \quad (3.71)$$

Contrairement au cas avec une seule commande $r = 1$, l'hypothèse $p = q$ est rarement constatée. Le cas le plus fréquent est $q = 1/2$ et $p = 1$ car les termes croisés $[g^i, g^j](x)$, pour $i \neq j$ ne disparaissent pas. La proposition suivante peut être utilisée pour essayer de déterminer une solution.

Proposition 46 (Conditions nécessaires algébriques pour les arcs singuliers) Soit (x, u) une solution optimale singulière à $(\mathcal{P}_{MA})$ avec un ordre de l'arc p tel que $2p \geq 2q$. Le lieu décrit par (x, u) vérifie les conditions algébriques :

$$\bigcap_l \{(x, u) : S_l(x, u) = 0\} \quad (3.72)$$

où S_l sont tous les mineurs de rang n i.e. une séquence indexée de déterminants $S_l(x, u) = \det([h_{i_1} \dots h_{i_n}])$ dont les colonnes $h_{i_k}(x, u)$, $1 \leq i_1 < \dots < i_k < \dots < i_n \leq i_{max} = 2(p - q)|\mathcal{R}|^2 + (2p + 1)|\mathcal{R}|$ sont sélectionnées à partir des ensembles $\alpha(x)$, $\beta(x)$ et $\gamma(x, u)$ avec

$$\begin{aligned} \alpha(x) &= \{ad_{f_{drift}}^k g^i(x), i \in \mathcal{R}, k = 0, \dots, 2p - 1\} \\ \beta(x) &= \{[g^j, ad_{f_{drift}}^{(2q+k-1)} g^i](x), i \in \mathcal{R}, j \in \mathcal{R}, k = 0, \dots, 2p - 2q - 1\} \\ \gamma(x, u) &= \{ad_{f_{drift}}^{2p+k} g^i(x) + \sum_{l \in \mathcal{R}} [g^l, ad_{f_{drift}}^{2p-1} g^i](x) u_l, i \in \mathcal{R}\}. \end{aligned}$$

Lorsque le principe du minimum exige $H = 0$, on ajoute $f(x)$ dans la liste précédente des vecteurs h_k .

Remarque 10 Cette proposition peut ne pas aboutir si toutes les composantes de u n'apparaissent pas explicitement dans $\gamma(x, u)$. Cette situation correspond au cas où il existe au moins un indice l tel que $[g^l, ad_{f_{drift}}^{2p-1} g^i](x) \equiv 0, i \in \mathcal{R}$.

Nous proposons donc de déterminer les arcs singuliers à l'aide de l'algorithme suivant :

Proposition 47 (Algorithme pour l'obtention des arcs singuliers) Les arcs singuliers candidats peuvent être déterminés à partir des cinq étapes qui suivent :

1. Pour tout sous-ensemble $\mathcal{R}$ d'indice de commande de $\{1, 2, \dots, r\}$, fixant à 0 ou 1 les valeurs des commandes n'appartenant pas à $\mathcal{R}$, utiliser la proposition 46 (ou 45) pour déterminer $u(x)$, pour toutes les valeurs entières admissibles de p , $2p \geq 2q$.
2. Stocker uniquement les solutions (x, u) telles que $u(x) \in co(U)$.
3. Déterminer a-posteriori le vecteur λ admissible via

$$\lambda \perp \{h_i(x, u), i = 1, \dots, i_{max}\}$$

4. Vérifier a-posteriori si la condition du minimum du principe du minimum est satisfaite (en raison des valeurs de commande fixées a-priori à 0 ou 1).
5. Stocker les solutions qui satisfont la condition du deuxième ordre du théorème 43.

3.5 Synthèse des trajectoires optimales

Cette section propose, pour une classe de critères, une méthode de synthèse des commandes optimales en temps rétrograde depuis le point de fonctionnement x_{ref} . Cette méthode évite le traditionnel problème aux deux bouts nécessitant l'utilisation d'algorithmes de tir.

Il est bien connu que les conditions nécessaires du principe de minimum mènent à un problème aux deux bouts pour lequel les conditions "initiales" du système Hamiltonien se répartissent aux deux extrémités de l'intervalle d'intégration. La résolution par des algorithmes de tir est tout de même possible mais la présence de trajectoires singulières complique la tâche. Nous supposons que l'hypothèse suivante est satisfaite :

Hypothèse 5 *Toutes les solutions optimales rallient le point de fonctionnement en temps fini ou éventuellement asymptotiquement pour un critère en temps infini.*

Remarque 11 *C'est évidemment le cas pour un critère en temps optimal ou un critère quadratique en temps infini. Mais l'hypothèse n'est pas vérifiée pour un critère quadratique en temps fini.*

Avec cette hypothèse et en exploitant le caractère singulier du point de fonctionnement x_{ref} , nous allons montrer qu'il est possible de déterminer les conditions finales du système grâce aux propositions 39 et 47. Le problème sera alors transformé en un simple problème de valeur initiale permettant de générer un ensemble dense de trajectoires finissant sur un point d'équilibre donné.

Comme nous avons mentionné dans la section 4.1., le point d'équilibre x_{ref} fait partie d'un arc singulier car la commande u_{ref} qui maintient l'état x sur l'équilibre x_{ref} ($f(x_{ref}) + g(x_{ref})u = 0$) n'est pas sur un sommet du domaine de commande.

Si une trajectoire rallie donc en temps fini ce point d'équilibre avec une commande régulière ($u_i = 0$ ou $u_i = 1$, $i = 1, \dots, r$), u_i doit commuter vers u_{ref} lorsque $x(t_r) = x_{ref}$ à l'instant t_r (temps de commutation). Par conséquence, à partir des propositions 39 et 47, les valeurs admissibles λ_{ref} pour $\lambda(t_r)$ sont alors

déduites . Un algorithme de tir n'est donc pas nécessaire puisque les conditions finales sont connues.

Si l'hypothèse 5 est valide et afin de générer les trajectoires, nous pouvons alors faire une intégration en temps rétrograde du système hamiltonien (3.7) en prenant pour condition initiale (x_{ref}, λ_{ref}) et les valeurs de commande données par $u_i = 0$, $u_i = 1$ ou $u_i = u_{ref}$, $i = 1, \dots, r$. En effet, comme tout point singulier est potentiellement un point de bifurcation, il faut envisager comme valeur de commande $u_i = 0$, 1 ou u_{ref} .

Cependant, ce n'est pas la bonne façon de procéder. Il y a au moins deux raisons pour lesquelles cette procédure n'est pas utilisable :

- La détermination des valeurs de la commande singulière passe par l'intégration d'un système différentiel hautement non linéaire (l'équation de u') et mal conditionné.
- Les arcs singuliers peuvent converger asymptotiquement vers le point d'équilibre (Cas d'un critère quadratique en temps infini). Le calcul des arcs en temps rétrograde est donc numériquement infaisable.

En revanche, si l'on dispose d'une solution algébrique des arcs singuliers conduisant au point d'équilibre x_{ref} , on peut commencer l'intégration en temps rétrograde à partir de points x_i sur ces arcs menant à x_{ref} . Le problème d'intégration lié à la partie de segment $(x_i \rightarrow x_{ref})$ et correspondant à une durée de temps infinie est de ce fait, rejeté.

L'intégration en temps rétrograde commence avec les conditions *ad hoc* pour x et λ i.e. les valeurs singulières obtenues d'après les propositions 39 et 47, et, avec des valeurs de commande choisies de sorte que si la composante i -ième de u est singulière, deux bifurcations sont considérées en commutant la commande u_i à 0 ou 1. Le reste de la trajectoire est généré en intégrant en temps rétrograde le système Hamiltonien (3.27) et en respectant la condition de minimum sur l'Hamiltonien.

Avec un choix adéquat de points initiaux sur ces arcs on parvient alors à générer un ensemble dense des trajectoires optimales (Voir Fig. 3.3).

Il convient néanmoins et avant de fixer cet ensemble de trajectoires de lever les éventuels conflits. En effet, il peut advenir que deux trajectoires se croisent dans l'espace d'état avec des commandes distinctes créant un conflit. On lève le conflit en considérant la valeur de la fonction coût. La branche de coût le plus élevé est alors coupé à partir du point de conflit puisqu'il existe une solution moins coûteuse (Voir Fig. 3.4).

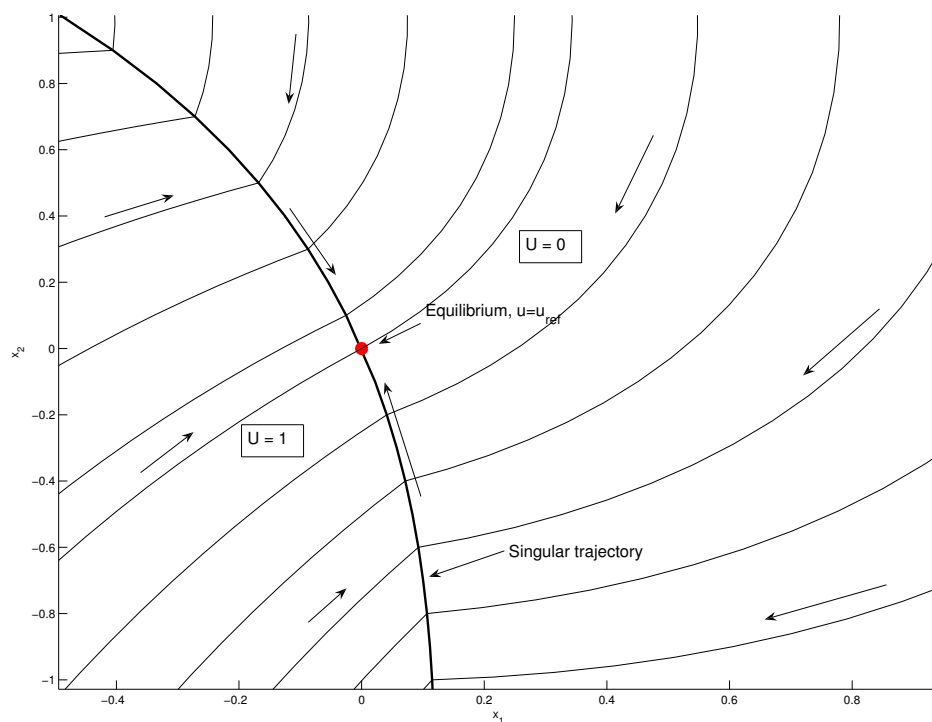
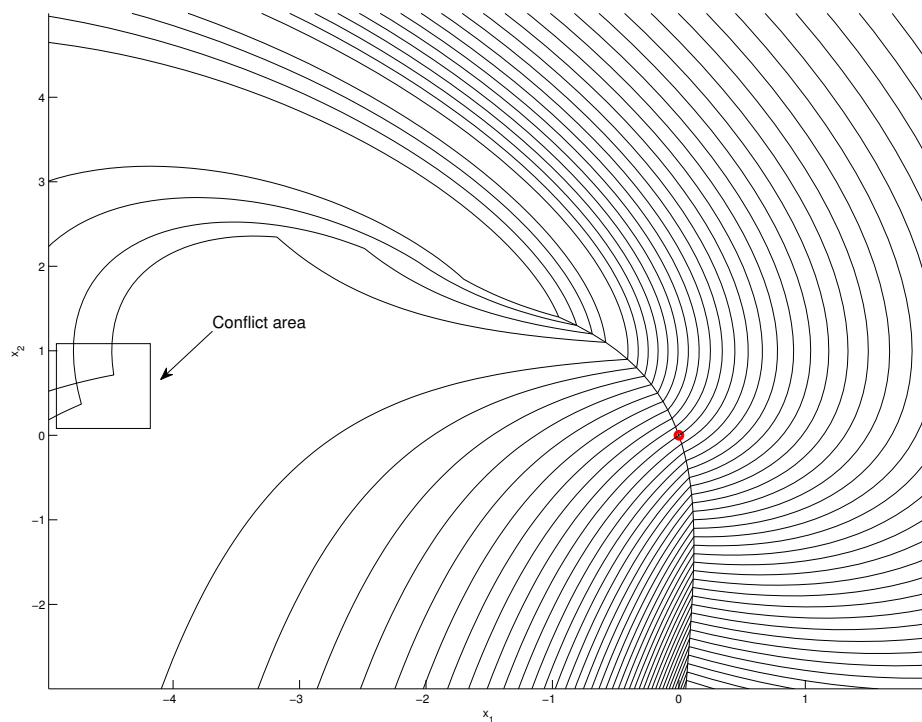


FIG. 3.3 – Génération de toutes les trajectoires : Intégration en temps rétrograde depuis les points sur l'arc singulier



Remarque 12 *Pour la classe des systèmes auxquels nous nous intéressons, les arcs singuliers sont connexes au point d'équilibre et mènent en ce point. En revanche, on ne peut exclure l'existence de trajectoires optimales composées alternativement d'arcs singuliers et réguliers. Ce cas de figure est un problème difficile et les conditions d'union entre les arcs singulières et les arcs réguliers ne sont pas complètement connus [RB95], [MP71]. Nous ne l'avons pas rencontré dans les exemples traités.*

Pour un ensemble suffisamment dense de trajectoires optimales calculées sur un pavé de $\mathbb{R}^n$, un réseau de neurones interpole les solutions optimales. La relation entrée sortie obtenue détermine un retour d'état $u(x)$ dont l'évaluation est simple car formée d'un nombre fini de sommes et produits. Par conséquent, la loi de commande est facilement implantable en temps réel. Comme les valeurs de la commande sont discrètes, l'interpolation obtenue est de très bonne qualité. Dans le chapitre 4, nous verrons un cas multivariable avec un exemple de commande optimale pour un convertisseur multiniveaux.

La proposition suivante résume la méthode :

Proposition 48 (Algorithme pour calculer la loi de commande par retour d'état). *Il est possible d'établir une commande par retour d'état $u(x)$ suivant le schéma suivant :*

1. Déterminer l'ensemble des arcs singuliers admissibles suivant les propositions 39 ou 47.
2. Intégrer en temps rétrograde le système Hamiltonien (3.7) en vérifiant les conditions nécessaires du principe du minimum. Les intégrations commencent avec une valeur de $x(t_f)$ et de $\lambda(t_f)$ sur les arcs singuliers et avec une commande u_i à $u_i = 0$ ou $u_i = 1$, si u_i est une commande singulière. Pour un pas de discrétisation fixé et suffisamment petit, un ensemble "dense" de trajectoires issues de points x_i , appartenant à la surface singulière, est généré par une intégration en temps rétrograde. La durée de cette intégration est choisie suffisamment grande pour couvrir la région de l'espace où le système est susceptible d'opérer.
3. Stocker toutes les trajectoires.
4. Lever les conflits éventuels.
5. Interpoler toutes les trajectoires optimales avec l'aide d'un réseau de neurones. Le réseau a comme entrée les valeurs de x et comme sortie la commande optimale u .

Remarque 13 *Le succès de la méthode nécessite :*

- un système d'ordre faible.
- l'unicité des commandes singulières candidates u et de la variable adjointe λ comme fonction de l'état x .
- un arc singulier au plus par trajectoire.

3.6 Exemple (Convertisseur buck-boost)

Revenons à l'exemple du convertisseur buck-boost montré dans le chapitre 2 et consacré à la commande prédictive avec un critère quadratique en temps infini. Nous avons

$$f(x) = \begin{bmatrix} \mathcal{A}_2 z + \mathcal{B}_2 E \\ \frac{1}{2}(z - z_{ref})^T \mathcal{Q}(z - z_{ref}) \end{bmatrix} \quad g(x) = \begin{bmatrix} (\mathcal{A}_1 - \mathcal{A}_2)z + (\mathcal{B}_1 - \mathcal{B}_2)E \\ 0 \end{bmatrix} \quad (3.73)$$

où

$$\mathcal{A}_1 = \begin{bmatrix} 0 & 0 \\ 0 & -\frac{1}{RC} \end{bmatrix} \quad \mathcal{A}_2 = \begin{bmatrix} 0 & \frac{1}{L} \\ -\frac{1}{C} & -\frac{1}{RC} \end{bmatrix}$$

$$\mathcal{B}_1 = [\frac{E}{L}, 0]^T \quad \mathcal{B}_2 = [0, 0]^T$$

$z = [i_L, v_C]^T$, $z_{ref} = [2, -1]^T$, $\mathcal{Q} = \text{diag}[1, 1]$ et $R = L = C = E = 1$, $x = [z, x_n]^T$. Comme l'ordre de problème est 1, et $n = 3$, alors nous sommes dans le cadre de la proposition 36. Puisque le critère est en temps infini, le Hamiltonien $H = \lambda^T f + \lambda^T g u = 0$. Alors, nous ajoutons l'équation $\lambda^T f = 0$ et nous avons donc :

$$[f(x), g(x), [f, g](x)] = \begin{bmatrix} x_2 - 1 & -x_2 + 2 & x_2 - 1 \\ -x_1 - x_2 - 1 & x_1 + 2 & 3 + x_1 \\ \frac{1}{2}x_1^2 + \frac{1}{2}x_2^2 & 0 & 2x_1 - 2x_2 \end{bmatrix}$$

$$S = \det(f(x), g(x), [f, g](x)) = 0$$

$$= -\frac{5}{2}x_1^2 x_2 + \frac{7}{2}x_1 x_2^2 - 8x_1 x_2 - 2x_2^2 + 2x_1^2 - \frac{1}{2}x_2^3 - x_1^3 x_2 + \frac{3}{2}x_1^3 - x_2^3 x_1 = 0. \quad (3.74)$$

La figure 3.5 montre les points sur l'espace d'état qui vérifient $S = 0$. Observons qu'il existe trois arcs singuliers candidats.

A partir de ces arcs, la variable adjointe doit vérifier :

$$\lambda \perp \{f(x), g(x), [f, g](x)\}$$

et u est sous la forme :

$$u = -\frac{\lambda^T \text{ad}_f^2 g(x)}{\lambda^T [g, \text{ad}_f^1 g](x)}.$$

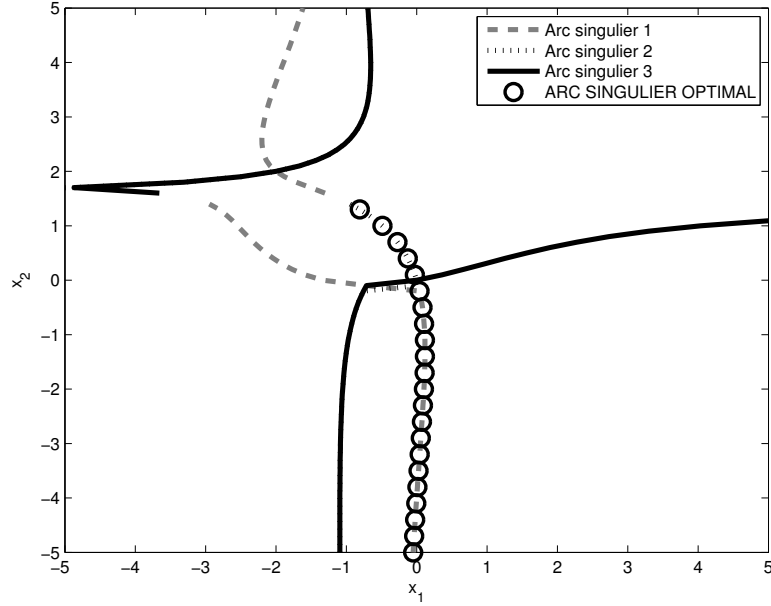


FIG. 3.5 – Arc singuliers candidats et arc singulier optimal. L’arc singulier qui vérifie la condition d’ordre deux est marqué avec O

Parmi les candidats admissibles, la condition d’ordre deux (3.40) détermine les arcs optimaux. La figure 3.5 montre l’arc optimal qui vérifie cette condition. La figure 3.6 montre un ensemble des trajectoires finissant au point d’équilibre ainsi que l’interpolation réalisée par le réseau de neurones. Nous pouvons remarquer que le réseau de neurones apprend la commande avec une approximation négligeable sur la frontière.

Le réseau de neurones utilisé pour apprendre la valeur de la commande comme fonction de l’état est un réseau feed-forward avec un algorithme d’apprentissage de type back-propagation. Le nombre de neurones de la couche cachée est 10 et nous avons besoin de 500 itérations de l’algorithme (implémenté avec la toolbox « neural network » de Matlab) afin d’avoir une erreur d’apprentissage acceptable de 0.00915. L’ensemble des données de validation correspondent à des trajectoires générées à partir de points initiaux sur l’arc singulier différent de ceux choisis pour l’apprentissage. L’erreur de validation obtenu est de 0.001354

La figure 3.7 montre la carte utilisée pour commander le convertisseur buck-boost en temps réel. Dans la figure 3.8, nous pouvons voir une trajectoire qui converge au point d’équilibre à partir des conditions initiales nulles $z(0) = [0, 0]^T$. Finalement, la figure 3.9 montre l’évolution des états par rapport au temps. La fréquence d’échantillonnage est 10 Hz.

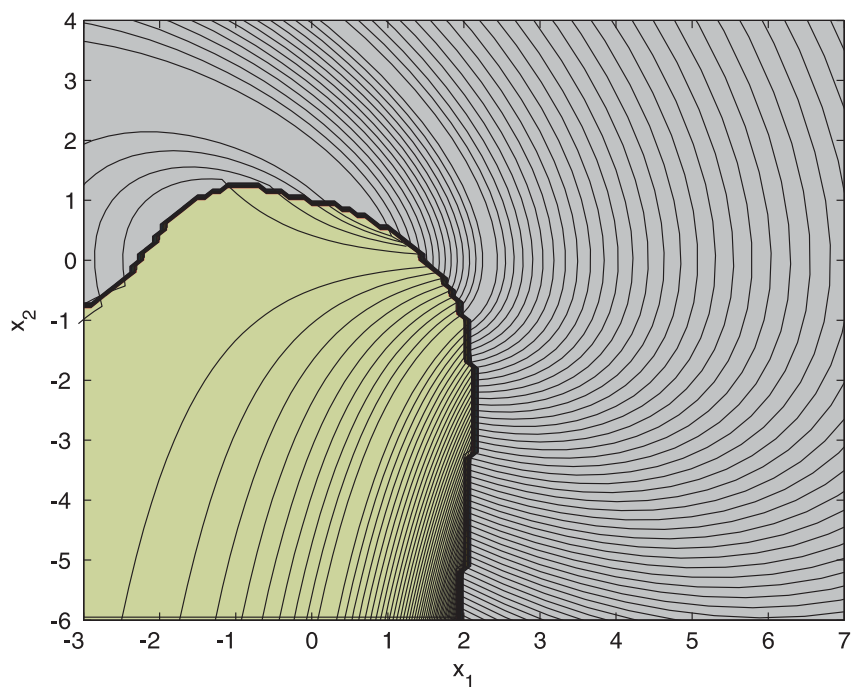


FIG. 3.6 – Trajectoires à partir de l'arc singulier pour un convertisseur buck-boost et interpolation faite par le réseau de neurones

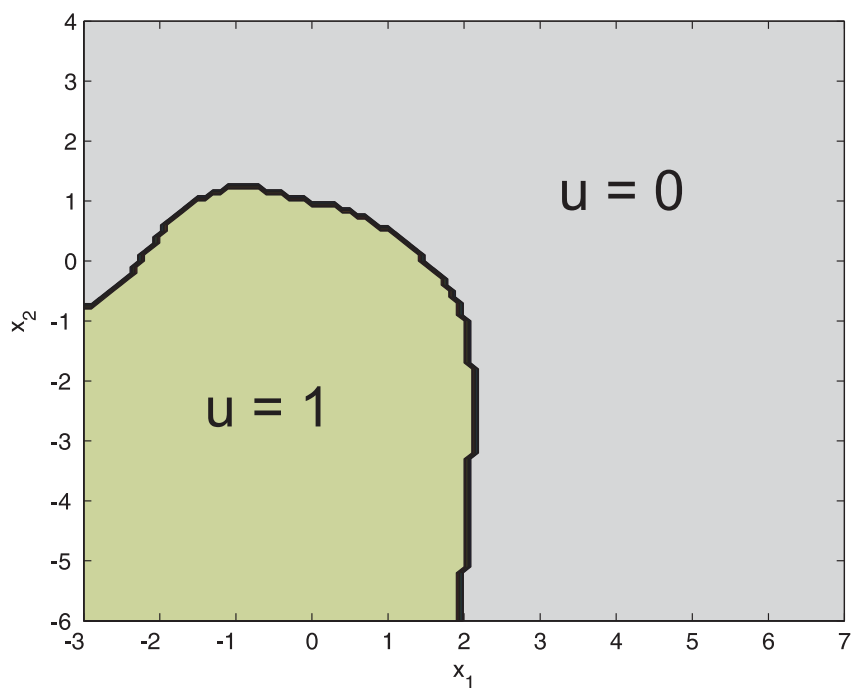


FIG. 3.7 – Carte pour la commande du convertisseur buck-boost

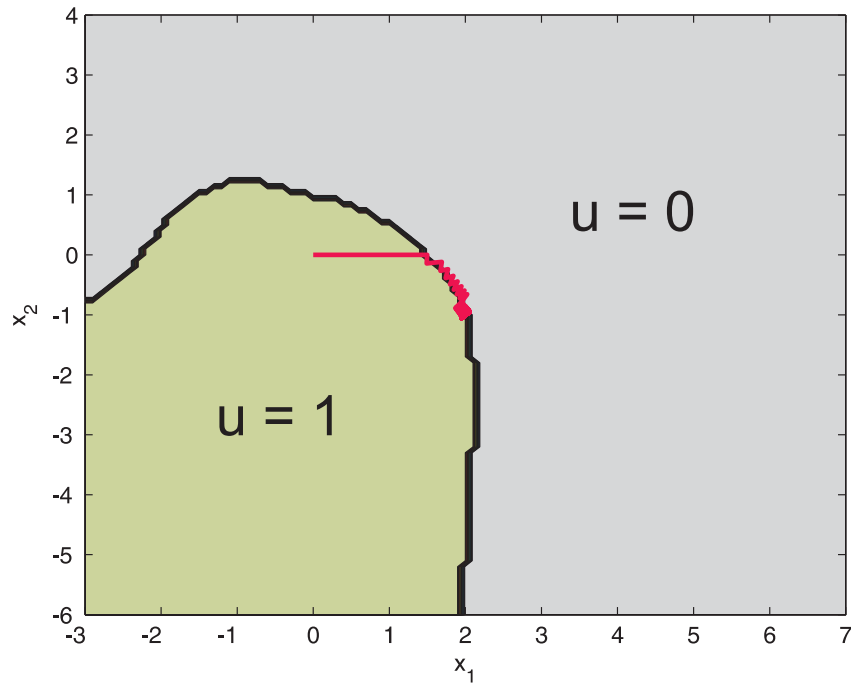


FIG. 3.8 – Trajectoire du système commande depuis conditions initiales nulles $z(0) = [0, 0]^T$.

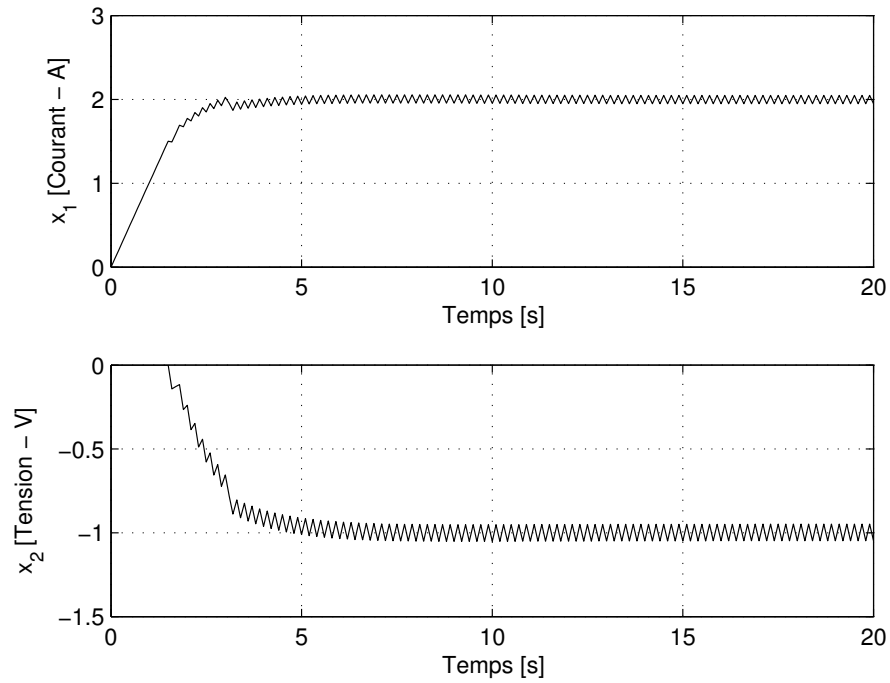


FIG. 3.9 – Evolution du système depuis conditions initiales nulles $z(0) = [0, 0]^T$, $(i_L(t), v_C(t))$.

3.7 Conclusions et Perspectives

Dans ce chapitre, nous avons proposé une méthodologie pour calculer une commande optimale par retour d'état pour des systèmes commutés de dimension faible se caractérisant par un régime permanent cyclique.

La méthode consiste à relaxer le domaine de commande à son enveloppe convexe et à considérer les solutions de ce problème étendu. Si les solutions obtenues sont bang-bang, elles sont également solutions du problème original. Mais comme il l'a été montré, ce n'est généralement pas le cas en raison de la nature du point de fonctionnement de ces systèmes. Les solutions incorporent alors une partie singulière et leur détermination nécessite l'utilisation de conditions nécessaires d'ordre supérieur. Elles correspondent à des solutions aux sens de Fillipov.

Bien que ces solutions ne soient pas admissibles pour le problème original, on peut les approcher, par mode glissant, d'autant près que l'on veut comme le prouvent les théorèmes de densité. Dans la pratique, on fixera une vitesse maximum de commutation ou une bande d'hystérésis.

La contribution de ce chapitre montre qu'il est possible de déterminer algébriquement les surfaces singulières pour des systèmes de dimension faible. Les solutions s'expriment comme les racines d'un ensemble de déterminants et satisfaisant une condition nécessaire d'ordre deux.

L'hypothèse 5 suppose que les solutions rallient le point de fonctionnement en temps fini (cas d'un critère en temps optimal) ou asymptotiquement (cas d'un critère quadratique). Sous cette hypothèse, la synthèse de trajectoires optimales se fait par l'intégration en temps rétrograde du système Hamiltonien depuis les surfaces singulières menant au point de fonctionnement. La connaissance des conditions aux limites sur ces surfaces transforme le problème traditionnel aux deux bouts en un simple problème de valeur initiale.

Nous proposons d'utiliser un réseau de neurones pour interpoler l'ensemble dense sur un pavé de l'espace d'état des solutions générées.

La relation entrée-sortie qui en résulte définit une commande par retour d'état (donc en boucle fermée), facile à implémenter en temps réel. La commande étant à valeurs discrètes, cette relation définit une partition de l'espace d'état suivant les valeurs de la commande optimale à appliquer. Les surfaces singulières correspondent à des parties de frontières définissant des surfaces de glissement optimales.

L'erreur d'approximation est faible et se situe sur le tracé des zones frontières de la partition de l'espace en raison du caractère discret de la commande et de l'utilisation d'un seuil. L'exemple du convertisseur buck-boost illustre la méthode.

La méthode développée dans ce chapitre est très utile pour assurer une valeur moyenne en régime permanent et pour régler le régime transitoire vers le point d'équilibre. En revanche, elle ne permet pas de contrôler les oscillations autour du point d'équilibre. Une réponse possible peut être de basculer au voisinage du point d'équilibre vers un correcteur qui réalise une poursuite de trajectoires comme dans le chapitre consacré à la commande prédictive.

L'inconvénient de la méthode est certainement la limite imposée sur la dimension du système. Cette limite est liée à la possibilité de définir de manière unique les commandes singulières candidates en fonction de x via les conditions algébriques présentées. La méthode n'est donc pas applicable si pour un x donné, les conditions algébriques laissent la commande indéterminée.

Les sauts autonomes n'ont pas été pris en compte dans la formulation du problème (Voir théorème 28). Pour les convertisseurs statiques, ce type de saut représente le mode de conduction discontinue. Nous y reviendrons plus tard dans le chapitre 4.

Chapitre 4

Application aux convertisseurs statiques

Résumé. Ce chapitre met en oeuvre sur la classe des convertisseurs de puissance les méthodes de commandes par retour d'état proposées dans les deux précédents chapitres. Les résultats présentés sont obtenus par simulation et/ou sur maquette suivant le type du convertisseur considéré. Deux convertisseurs sont privilégiés : Le convertisseur buck et le convertisseur multi-niveaux. La robustesse des méthodes est illustrée sur des variations de charge et de source.

Mots clés. commande prédictive, commande optimale, cycle limite.

4.1 Introduction

Comme nous l'avons déjà mentionné dans les chapitres précédents, une classe de systèmes commutés particulièrement importante en termes d'applications est la classe des convertisseurs de puissance. Ces dispositifs sont couramment embarqués sur les ordinateurs, les moteurs électriques, les téléphones portables et plus généralement tous les appareils comportant une alimentation électrique.

Dans les chapitres précédents nous avons utilisé à titre d'exemple, un convertisseur buck-boost. Étudions son fonctionnement de plus près [Tse00]. Nous présentons sa structure de nouveau dans la figure 4.1.

A partir d'une source de tension continue E , ce convertisseur donne un courant adapté au besoin d'une charge, ici représenté par la résistance R_0 . Observons que la capacité et l'inductance stockent de l'énergie lorsque le commutateur u est fermé. Au contraire, cette énergie est libérée lorsque u est ouvert. Ce système n'ayant que deux points d'équilibre suivant la position de u , le pilotage de ce

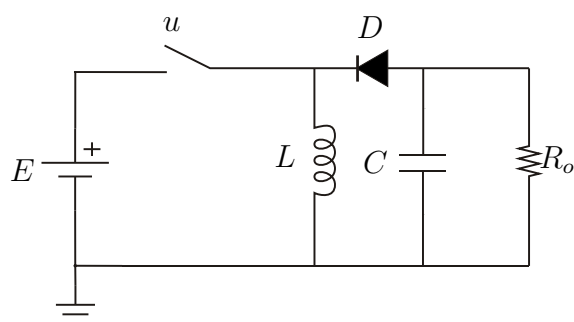


FIG. 4.1 – Convertisseur buck-boost

dispositif s'effectue par l'ouverture et la fermeture à fréquence élevée de cet interrupteur. L'objectif de commande est donc de délivrer en valeur moyenne le courant nécessaire au fonctionnement de la charge tout en assurant des transitoires rapides avec un dépassement limité.

En régime permanent, l'équilibrage en valeur moyenne autour de la référence de courant demandée conduit à un comportement cyclique.

Aujourd'hui, l'augmentation des fonctionnalités sur les objets nomades à alimenter et leur perfectionnement (intégration de DSP par exemple) conduit à des cahiers des charges assez stricts en termes de performance dynamique et de stabilité. L'utilisation de techniques classiques (PID) ne donne pas une réponse satisfaisante.

L'analyse classique et la recherche de lois de commande reposent typiquement sur des modèles moyens ou sur des approches de type "petits signaux" [MW73], [SNL⁺91], [ILG78]. La technique, à base de modèle moyen, est adaptée lorsque la moyenne est calculée sur une période de temps plus petite que les constantes de temps des différents sous systèmes. Cependant, cette approche est une approche basse fréquence qui ignore complètement l'aspect discontinu des commutations produites par les commutateurs. La forme d'onde résultante peut générer des sous-harmoniques ou des inter-harmoniques non désirés de la fréquence de coupure. [MB00].

Il existe aussi des approches basées sur des modèles linéaires échantillonnés. A partir du modèle non linéaire échantillonné, on utilise le modèle linéarisé autour du point de fonctionnement désiré [MVTH90], [HBY91]. Avec ces techniques, la récurrence non linéaire ne s'obtient pas facilement. Un inconvénient est la possibilité de voir apparaître des ondulations élevées entre les instants de commutation.

Une voie qui reçoit un écho très favorable dans la communauté des électrotechniciens, est celle qui consiste à commander directement les différents commu-

tateurs, sans passer par un modèle moyen, avec à la clé, la possibilité d'améliorer considérablement les performances dynamiques.

Il existe déjà des résultats de lois de commande testées par des compagnies qui montrent une amélioration de 50% des temps de réponse comparés aux techniques couramment utilisées [BG08].

Dans ce chapitre, nous sommes particulièrement intéressés par deux convertisseurs mentionnés dans le chapitre I. Nous commencerons par voir les résultats de la commande prédictive et la commande optimales sur un cas "simple", le convertisseur buck. Ensuite, nous couvrons un cas multi-entrées avec un convertisseur de quatre niveaux et trois cellules.

4.2 Le convertisseur buck

4.2.1 Topologie

La figure 4.2 montre un convertisseur buck synchronisé.

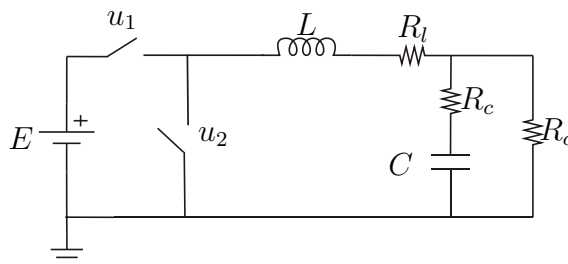


FIG. 4.2 – Convertisseur buck

Un convertisseur Buck, ou hacheur série, est une alimentation à découpage qui convertit une tension continue en une autre tension continue de plus faible valeur. Le convertisseur Buck a un fort rendement et offre la possibilité de réguler la tension de sortie. Il est donc plus adapté pour les applications d'électronique de puissance comme la réduction 12-24V fournie par une batterie d'ordinateur portable vers les quelques Volts nécessaires au fonctionnement du processeur. Ces alimentations doivent fournir un fort courant avec une faible ondulation de tension tout en restant dans un volume réduit [Tse00], [Ség99].

Son fonctionnement est assez simple, les deux commutateurs u_1 et u_2 travaillent de manière complémentaire : Si u_1 est fermé, u_2 est ouvert et si u_2 est fermé, u_1 est ouvert.

Ce convertisseur alterne donc entre deux modes de fonctionnement : lorsque u_1 est fermé, une partie de l'énergie fournie par la source E est stockée dans L

et C , le reste étant consommé par la charge. Si u_1 est ouvert, l'énergie stockée est déchargé ; l'énergie stockée diminue. Dans ce circuit, le rôle prépondérant est essentiellement joué par la self pour stocker l'énergie ; la capacité permet de lisser la tension délivrée à la charge.

4.2.1.1 Modèle mathématique

Comme u_1 et u_2 sont complémentaires, nous utiliserons le schéma équivalent de la figure 4.3 où u remplace u_1 et u_2 .

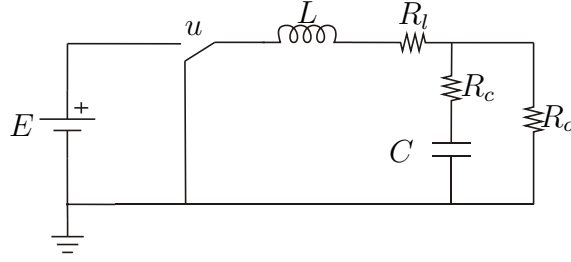


FIG. 4.3 – Circuit simplifié du convertisseur buck

Le nombre de variables d'état est liée au nombre de composants réactifs du système (capacité et inductance).

Définissons l'état comme $x(t) = [i_L(t), v_c(t)]^T$ où i_L est le courant de l'inductance et $v_c(t)$ est la tension aux bornes de la capacité. Le système est décrit alors par le modèle commuté :

$$\dot{x}(t) = \begin{cases} \mathcal{A}x(t) + \mathcal{B} & \text{Mode } \sigma = 1 : \text{Commutateur } u \text{ est fermé } (u = 1). \\ \mathcal{A}x(t) & \text{Mode } \sigma = 2 : \text{Commutateur } u \text{ est ouvert } (u = 0) \end{cases} \quad (4.1)$$

où les matrices $\mathcal{A}$ et $\mathcal{B}$ sont données par :

$$\mathcal{A} = \begin{bmatrix} -\frac{1}{L} \left(R_l + \frac{R_0 R_c}{R_0 + R_c} \right) & -\frac{1}{L} \frac{R_0}{R_0 + R_c} \\ \frac{1}{C} \frac{R_0}{R_0 + R_c} & -\frac{1}{C} \frac{1}{R_0 + R_c} \end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix} \frac{E}{L} \\ 0 \end{bmatrix} \quad (4.2)$$

La tension de sortie $v_0(t)$ à travers la charge R_0 est directement déduite comme :

$$v_0(t) = \mathcal{C}x(t) \quad \mathcal{C} = \frac{R_0}{R_0 + R_c} [R_c \quad 1] \quad (4.3)$$

Observons que le passage à un modèle avec un seul champ de vecteurs affine en la commande u est évident. Nous avons alors :

$$\begin{aligned} \dot{x}(t) &= \mathcal{A}x(t) + \mathcal{B}u(t), & u(t) &\in \{0, 1\} \\ y(t) &= \mathcal{C}x(t) \end{aligned} \quad (4.4)$$

4.2.1.2 Cahier des charges

Le cahier des charges présenté dans la suite a servi comme Benchmark pour comparer des méthodes développées récemment pour des systèmes commutés [AFJ⁺07].

Pour ce convertisseur, les paramètres choisis pour tester les méthodes de commande sont : $C = 100\mu F$, $L = 2mF$, $R_o = 50\Omega$, $R_c = 0.1\Omega$ et $R_L = 0.5\Omega$. La source de tension est fixée à $E = 50$ V.

Nous allons chercher une commande telle que :

1. La tension de sortie soit régulée à la valeur moyenne de $v_{0,ref} = 25$ V.
2. La loi de commande soit robuste vis-à-vis de variations de charge et de variation de tension d'entrée. Pour les résultats présentés dans ce paragraphe, un échelon de charge est appliqué de 50Ω à 100Ω et la source de tension varie de $50V$ à 35 V.

Quelques contraintes sont aussi imposées :

1. Le courant de l'inductance est limité à la valeur de $i_L(t) \leq i_{L,max}$ où $i_{L,max} = 2.5$ A.
2. La période du cycle limite est fixée à $T_p = 0.05$ ms.
3. Une seule commutation est autorisée par période du cycle.

Le courant de l'inductance et la tension de la capacité sont des variables complètement mesurées ainsi que la tension d'entrée.

4.2.2 Commande prédictive

La commande prédictive du chapitre II n'emploie pas de critères de robustesse dans sa formulation. Néanmoins, dû à la simplicité du modèle (4.4), nous pouvons employer des techniques d'optimisation classique pour l'estimation de la charge.

La stratégie du chapitre II impose :

1. Calcul du cycle limite de référence.
2. Poursuite du cycle limite de référence en régime permanent (Détermination des temps de commutation).
3. Interpolation de la solution (si nécessaire) afin de créer un retour d'état qui puisse calculer la commande en-ligne.

4.2.2.1 Détermination du régime permanent

Commençons donc par déterminer les points d'opération du convertisseur. D'après la section 4.3 du premier chapitre, le modèle moyen est :

$$\begin{aligned} \dot{x}(t) &= \mathcal{A}x(t) + \mathcal{B}u(t), & u(t) &\in [0, 1] \\ y(t) &= \mathcal{C}x(t) \end{aligned} \quad (4.5)$$

La première condition imposée par le cahier de charge est d'avoir $v_{0,ref}$ comme valeur moyenne en régime permanent. Soit $v_{0,ref} = \mathcal{C}x_{ref}$. De l'équation d'équilibre $\dot{x}_{ref}(t) = 0$, on déduit la commande u_{ref} en régime permanent par :

$$u_{ref}(t) = -\frac{v_{0,ref}}{\mathcal{C}\mathcal{A}^{-1}\mathcal{B}} \quad (4.6)$$

Il nous faut donc déterminer l'exécution $(\mathcal{T}^\infty, \Omega^\infty, \mathcal{X}^\infty)$ qui décrit un cycle limite ayant comme valeur de sortie moyenne $v_{0,ref}$. Ω^∞ est évident, car le cahier des charges impose une seule commutation par période de cycle, alors la seule possibilité est d'avoir $s^\infty = 2$ et

$$\Omega^\infty = \{1, 2\}$$

Afin d'obtenir $\mathcal{T}^\infty$ et $\mathcal{X}^\infty$, nous pouvons appliquer un algorithme d'optimisation non-linéaire hors-ligne comme nous avons fait pour le convertisseur buck-boost dans le Chapitre II. Cependant, grâce à la connaissance de la période du cycle, de la séquence de modes et l'absence de contraintes de temps minimum des commutateurs, la commande u_{ref} peut être considérée comme le rapport entre la durée du mode σ_0 et la durée complète de la période pour ce convertisseur. Cette valeur est connue comme le rapport cyclique [BPM06], [BPFM07]. Les temps de commutations du régime permanent seront alors :

$$\mathcal{T}^\infty = \{0, u_{ref}T_p, T_p\}$$

Finalement, les valeurs de l'ensemble $\mathcal{X}^\infty$ sont :

$$\mathcal{X}^\infty = \{x_0^\infty, x_1^\infty, x_2^\infty\}$$

où

$$\begin{aligned} x_0^\infty = x_2^\infty &\in Ker \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - e^{A_2(t_2^\infty - t_1^\infty)} e^{A_1(t_1^\infty - t_0^\infty)} \right) \\ x_1^\infty &= e^{A_1(t_1^\infty - t_0^\infty)} x_0^\infty \end{aligned} \quad (4.7)$$

4.2.2.2 Construction de la boucle fermée

1. **Optimisation des temps de commutation :** Posons le système sous une forme autonome $\dot{x}(t) = A_{\sigma_i}x(t)$, avec $x_{n+1}(0) = 1$, $\sigma_i \in \mathcal{I}$, $\mathcal{I} = \{1, 2\}$, nous avons alors :

$$A_1 = \begin{bmatrix} \mathcal{A} & \mathcal{B} \\ 0_{1 \times 2} & 0 \end{bmatrix} \quad A_2 = \begin{bmatrix} \mathcal{A} & 0_{2 \times 1} \\ 0_{1 \times 2} & 0 \end{bmatrix} \quad (4.8)$$

Sur un horizon d'une période, nous allons chercher à minimiser le temps de commutation t_1 à l'aide du critère :

$$J(t_1) = \Delta X(t_1)^T \mathbf{Q} \Delta X(t_1) \quad (4.9)$$

où ΔX représente l'erreur de poursuite aux instants de commutation. ΔX ne dépend que du choix de t_1 et s'écrit $\Delta X(t_1) = [\Delta x(t_1)^T, \Delta x(T_p)^T]^T$ avec

$$\begin{aligned} \Delta x(T_p) &= e^{A_2(T_p-t_1)}x(t_1) - x_2^\infty \\ \Delta x(t_1) &= x(t_1) - x_1^\infty = e^{A_1(t_1)}x(0) - x_1^\infty \end{aligned} \quad (4.10)$$

$\mathbf{Q} = \text{diag}[\mathcal{Q}, \mathcal{Q}]$, $x(0)$ est la valeur initiale, elle correspond à la valeur mesurée de l'état en début de cycle.

Calculons $\frac{\partial J(t_1)}{\partial t_1}$. D'après la proposition 22 du chapitre II, les fonctions de sensibilité sont de la forme :

$$\begin{aligned} \frac{\partial \Delta x(t_1)}{\partial t_1} &= A_1 \Pi(t_1, 0)x(0) \\ \frac{\partial \Delta x(T_p)}{\partial t_1} &= \Pi(T_p, t_1) \Delta P_1 x(0) \end{aligned} \quad (4.11)$$

où $\Pi(t_1, 0) = e^{A_1 t_1}$, $\Pi(T_p, t_1) = e^{A_2(T_p-t_1)}$ et $\Delta P_1 = (A_1 - A_2)\Pi(t_1, 0)$.

Nous sommes alors en mesure d'appliquer l'algorithme Gauss-Newton 21 proposé au chapitre 2, pour déterminer l'instant de commutation optimale t_1 . Les résultats de simulation confirmés sur maquette réelle montrent qu'il suffit d'une itération de cet algorithme pour obtenir la convergence (même pour une condition initiale éloignée du point de fonctionnement).

2. **Prise en compte de la contrainte de courant :** Un autre point important du cahier de charge est la prise en compte de la contrainte $i_L(t) \leq i_{L,max}$. Cette contrainte est constatée a-posteriori. C'est-à-dire, à partir de la nouvelle valeur de t_1 calculé dans le point 1, nous réalisons une prédiction de la valeur du courant i_L^{pred} comme :

$$i_L^{pred}(t_1) = [1 \quad 0 \quad 0] \Pi(T_p, 0)x(0) \quad \Pi(T_p, 0) = e^{A_2(T_p-t_1)}e^{A_1 t_1} \quad (4.12)$$

Si la valeur de cette prédiction $i_L^{pred} > i_{L,max}$, l'algorithme donc doit calculer de nouveau t_1 tel que $i_L^{pred} = i_{L,max}$. Nous cherchons les racines de $i_L^{pred}(t_1) - i_{L,max} = 0$. Il suffit de trouver t_1 tel que

$$t_1 = \operatorname{argmin}(i_L^{pred} - i_{L,max}) \quad (4.13)$$

Un algorithme de Newton d'une seule itération entre encore en jeu pour ce calcul. Lorsque la valeur de t_1 est obtenue, nous appliquons au système le mode 1 pour t_1 secondes et le mode 2 pour $T_p - t_1$ secondes.

3. **Estimation de charge :** Afin d'assurer la robustesse de la méthode et de limiter sa sensibilité aux variations de charge, une estimation paramétrique est effectuée.

A partir de la mesure de l'état en fin de période et de sa prédiction (à l'aide de la valeur t_1 calculée), $x^p(t_1) = \Pi(T_p, 0)x(0)$, une estimation de la charge résistive R_0 par un algorithme du gradient sur l'erreur de prédiction $e = (x(T_p) - x^p)^T Q_R (x(T_p) - x^p)$ est utilisée.

4.2.2.3 Résultat pratique et de simulation

Pour cet exemple, l'algorithme se résume à :

Algorithme 49 (*Algorithme de commande prédictive pour un convertisseur buck*)

Données d'entrées :

- Tension de référence moyenne $v_{0,ref}$.
- Période du cycle T_p .
- Valeur de courant maximal $i_{L,max}$
- Matrices de poids Q , Q_R .
- Values de R_c , R_l , R_o , L , C , E .

L'algorithme se décompose suivant les étapes (h est l'indice de boucle) :

1. Mesurer $x(0)$, $t_1^0 \leftarrow 0$, $h \leftarrow 0$.
2. Calculer les matrices $\mathcal{A}$, $\mathcal{B}$ et $\mathcal{C}$.
3. Calculer u_{ref} de (4.6), x_0^∞ et x_1^∞ de (4.7)
4. Calculer t_1^{h+1} de l'algorithme 21
5. $t_1 \leftarrow t_1^{h+1}$.
6. Calculer i_L^{pred} de (4.12).
7. Si $i_L^{pred} > i_{L,max}$, calculer la nouvelle valeur t_1 à l'aide d'un algorithme de Newton. Sinon, continuer à 8

8. Prédire l'état prochain $x_p \leftarrow \Pi(T_p, 0)x(0)$
9. Appliquer la séquence de temps $\{t_1, T_p\}$ et la séquence de modes $\{1, 2\}$ au convertisseur. Mesurer l'état à la fin pour obtenir $x(T_p)$.
10. Ajuster la valeur de la charge

$$\min_{R_0} \varepsilon(x(T_p), x_p) = \min_{R_0} (x(T_p) - x^p)^T Q_R (x(T_p) - x^p) \quad (4.14)$$

11. $x(0) \leftarrow x(T_p)$, $h \leftarrow h + 1$ et aller à 2.

Comme le modèle de convertisseur est très simple et comme les différents algorithmes utilisés (Gauss-Newton (Etape 4), Newton (Etape 7) et gradient (Etape 10)) convergent en une seule itération, il n'est pas nécessaire de réaliser des interpolations de la commande par un réseau de neurones. Par contre, un développement en série à l'ordre 2 est utilisé pour approcher les exponentielles de matrices.

Cette méthode prédictive a été testée sur un convertisseur buck dans le cadre du projet HYCON à l'aide de l'ETH (Swiss Federal Institut of Technology) à Zurich. Nous avons implanté la commande sur un DSP de Texas Instrument TMS320C6713 en utilisant le langage C.

Les matrices $\mathcal{Q}$, $\mathcal{Q}_R$, sont fixées par :

$$\mathcal{Q} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2000 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathcal{Q}_R = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Nous évaluons d'abord la performance du régulateur sur un modèle simulé. Ensuite, nous l'appliquons au système réel. La figure 4.4 montre la réponse du convertisseur lorsqu'une variation de charge est réalisée pour le modèle simulé et le modèle réel. La figure 4.5 montre l'effet de changer la capacité du circuit. Finalement, la figure 4.6 montre la réponse lorsqu'une variation de tension est appliquée.

Les résultats sont satisfaisants puisque l'objectif de poursuite est atteint, y compris lorsque le convertisseur est soumis à des changements de charge ou de paramètre internes comme la capacité ou de tension d'entrée.

Observons que la réponse réelle est plus lente que la réponse simulée. La raison de ce comportement est la présence d'un retard entre le moment où la commande est calculée et le moment où elle est appliquée.

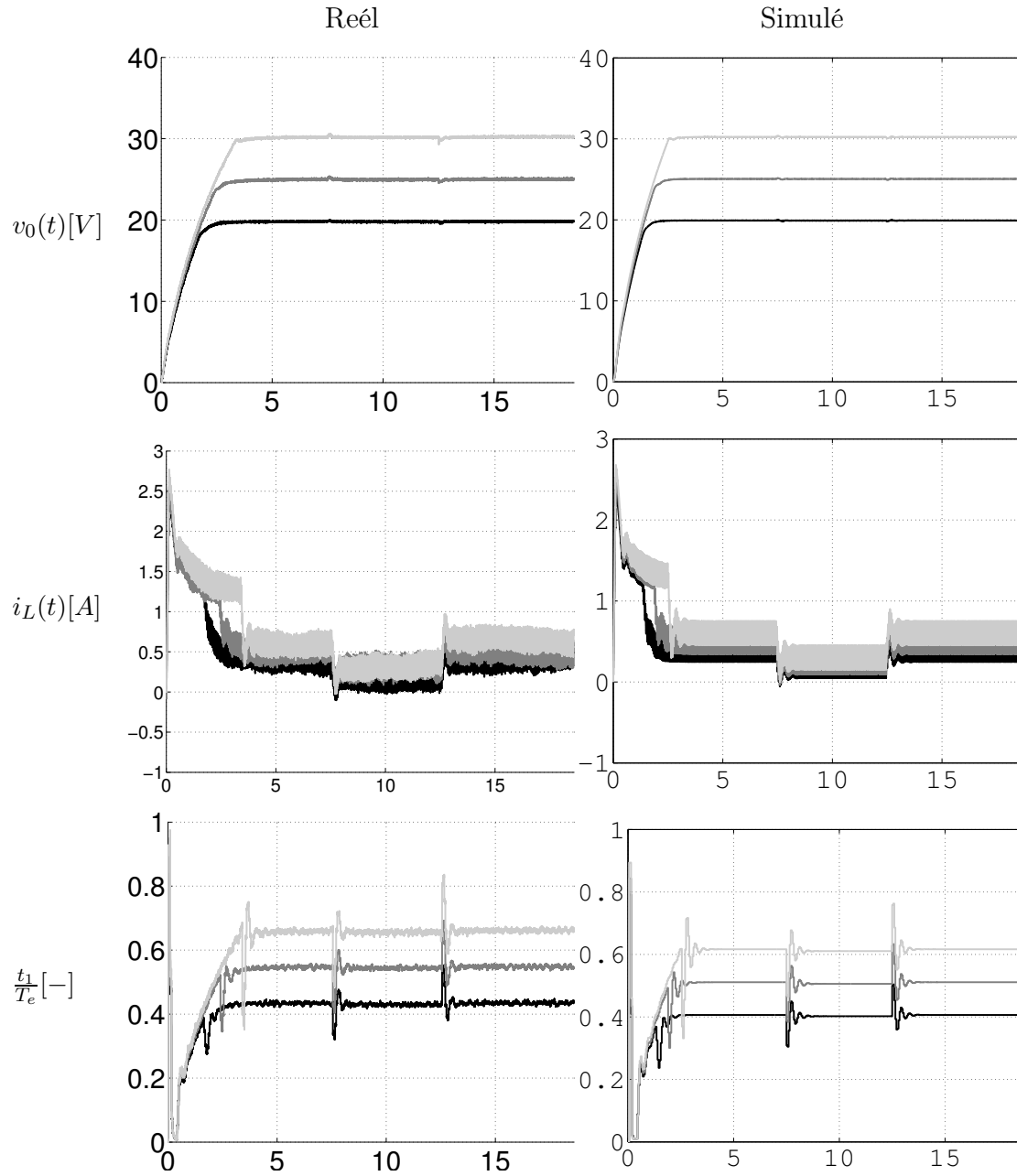


FIG. 4.4 – Test de démarrage. Comparaison entre les résultats simulés et le résultats pratiques. La charge change de $r_0 = 50\Omega$ à $r_0 = 100\Omega$ et de $r_0 = 100\Omega$ à $r_0 = 50\Omega$. Les valeurs de référence sont 20 V (noir), 25 V (gris) et 30 V (gris clair). L'échelle de temps est en ms

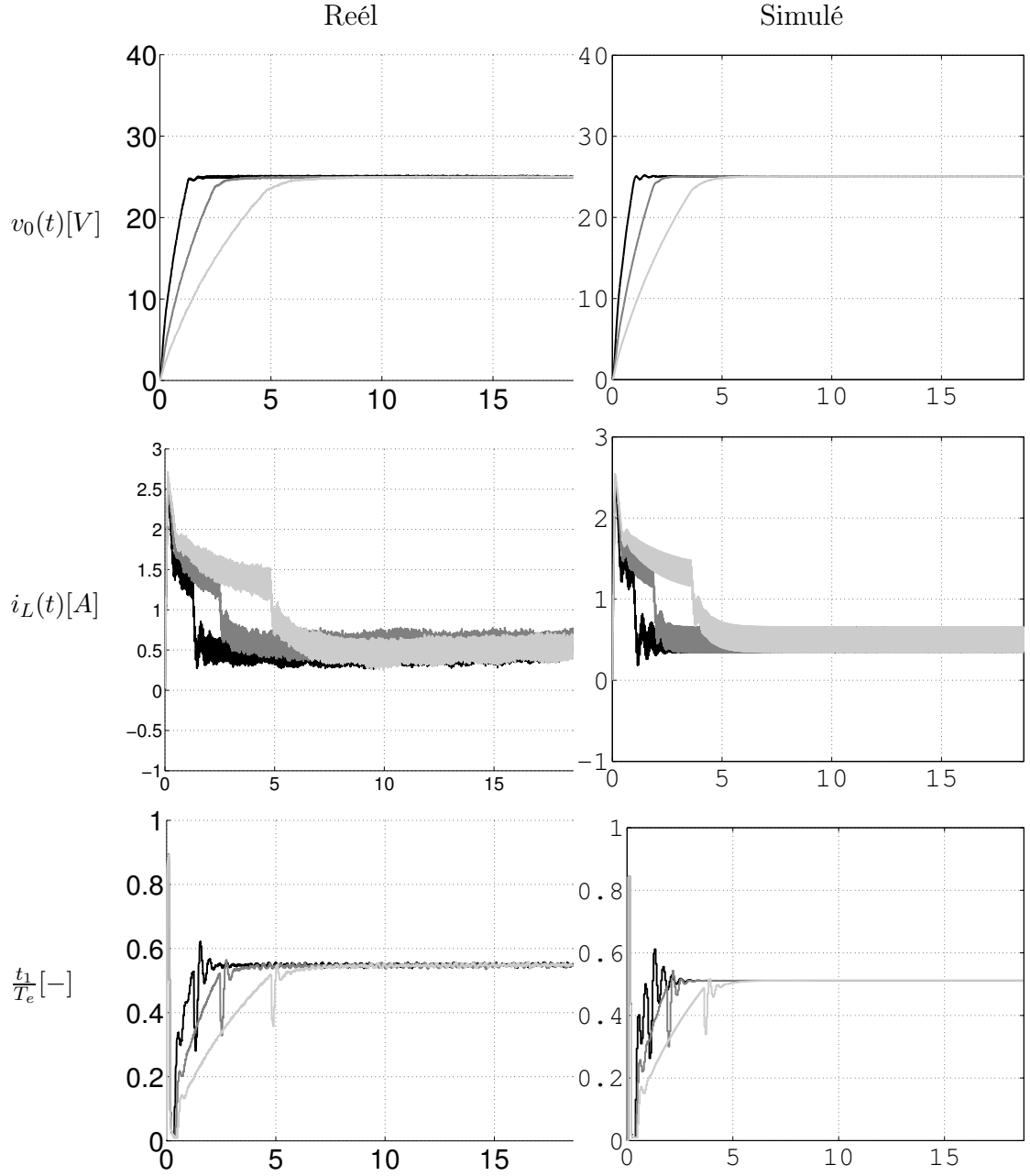


FIG. 4.5 – Test de robustesse. Comparaison entre les résultats simulés et pratiques. La capacité a une valeur différente à chaque fois. $x_c = 0.5x_{c,nom}$ (noir), $x_c = x_{c,nom}$ (gris) et $x_c = 2x_{c,nom}$ (gris clair). La valeur nominale est $x_{c,nom} = 100\mu F$. L'échelle de temps est en ms

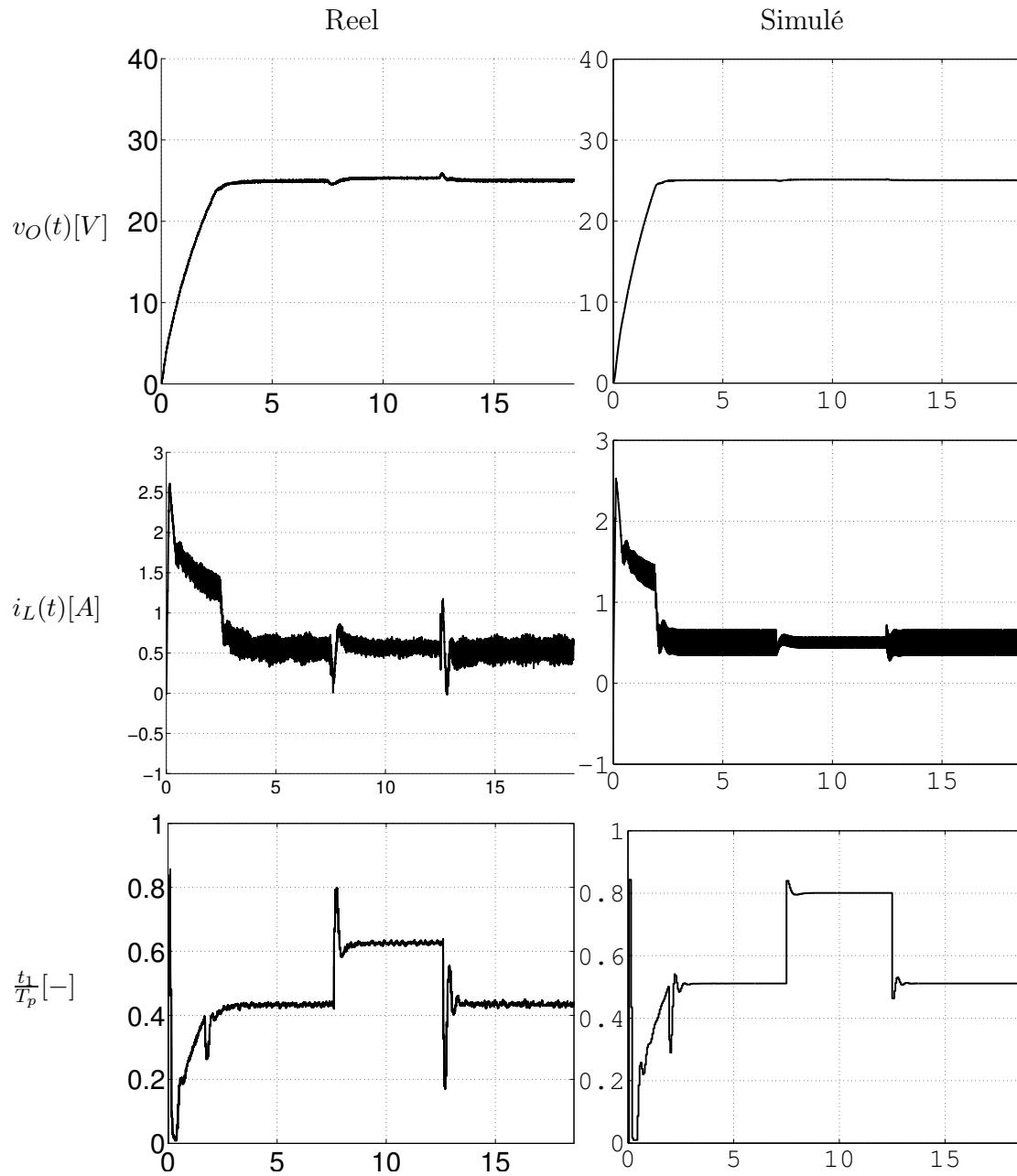


FIG. 4.6 – Test de variation de la source de tension. E change de 50V à 35V et de 35V à 50V. L'échelle de temps est en ms

4.2.3 Commande optimale

Reconsidérons le convertisseur de la figure 4.3. Nous allons appliquer la méthode de commande optimale du Chapitre III. Rappelons que la recherche des arcs singuliers est un point important de la méthode. A partir de ces arcs, nous déterminons une loi par retour d'état $u(x)$.

4.2.3.1 Recherche des arcs singuliers

La variable à réguler est la variable de sortie v_0 . Nous prendrons donc comme variable d'état $z(t) = [i_L(t), v_0(t)]^T$ obtenue grâce au changement de variable : $z = \mathfrak{M}x$ où $\mathfrak{M} = \begin{bmatrix} 1 & 0 \\ -\mathcal{C} \end{bmatrix}$. Le critère est un critère quadratique de la forme

$$J = \int_0^{+\infty} (z - z_{ref})^T Q (z - z_{ref}) dt$$

$$z_{ref} = [i_{L,ref}, v_{0,ref}]^T.$$

Le point z_{ref} correspond à un point d'équilibre du modèle moyen et dépend de la valeur de la tension d'entrée E et de la valeur de la résistance de sortie R_0 . En effet, comme nous l'avons montré dans le paragraphe 2.2.1. la référence moyenne dépende des matrices $\mathcal{A}$, $\mathcal{B}$ et $\mathcal{C}$.

f et g sont de la forme :

$$f(x) = [\mathfrak{M}\mathcal{A}z, (z - z_{ref})^T Q (z - z_{ref})]^T \quad (4.15)$$

$$g(x) = [\mathfrak{M}\mathcal{B}, 0]^T \quad (4.16)$$

En reprenant les notations du chapitre 3, on pose $x = [z^T, x_n^T]^T$ où x_n représente l'intégrale du critère sur l'intervalle $[0, t]$. Puisque le problème de la commande optimale est posé en temps infini, la fonction Hamiltonienne est identiquement nulle, alors pour tout t ,

$$H(\lambda, x, u) = \lambda^T f(x) + \lambda^T g(x) u = 0$$

On peut alors ajouter à l'équation de commutation $\lambda^T g(x) = 0$, l'équation $\lambda^T f(x) = 0$ ce qui conduit à la condition sur x (l'ordre du problème est $q = 1$) :

$$S(x) = \det(f(x), g(x), [f, g](x)) = 0$$

La variable adjointe est déterminée par les solutions de :

$$\lambda \perp \{f(x), g(x), [f, g](x)\}$$

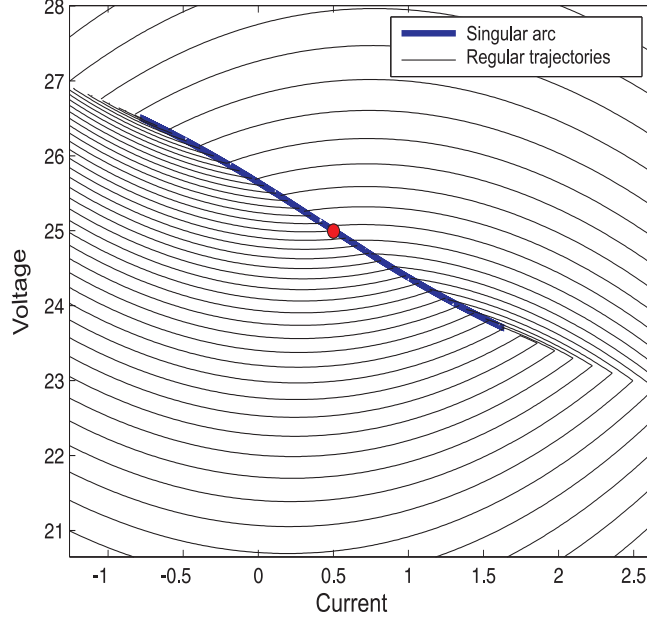


FIG. 4.7 – Arc singulier et réguliers

et la commande u est déduite de la dernière équation de (3.42)

$$u = -\lambda^T \text{ad}_f^2 g(x) / \lambda^T [g, \text{ad}_f^1 g](x)$$

D'après l'ensemble des candidates potentielles, la condition d'ordre 2 permet d'éliminer les portions d'arcs non optimaux. Par intégration en temps rétrograde, on obtient l'ensemble des trajectoires finissant au point d'équilibre (Figure 4.7). Notons que les portions singulières (en bleu sur la figure 4.7) correspondent à des valeurs de commande à l'intérieur de l'intervalle $[0, 1]$ et dont l'évolution temporelle est déterminée par l'équation différentielle non linéaire

$$\dot{u} = \frac{d}{dt} (-\lambda^T \text{ad}_f^2 g(x) / \lambda^T [g, \text{ad}_f^1 g](x))$$

Ces arcs singuliers définissent donc des surfaces de glissements optimales.

4.2.3.2 Retour d'état

Pour établir le retour d'état, nous allons appliquer l'algorithme de la proposition 48. Nous avons le pas suivants :

1. Déterminations des arcs singuliers : Dans la section 4.2.3.1, nous avons montré comment faire ce calcul. L'arc singulier optimal est celui montré dans la figure 4.7.

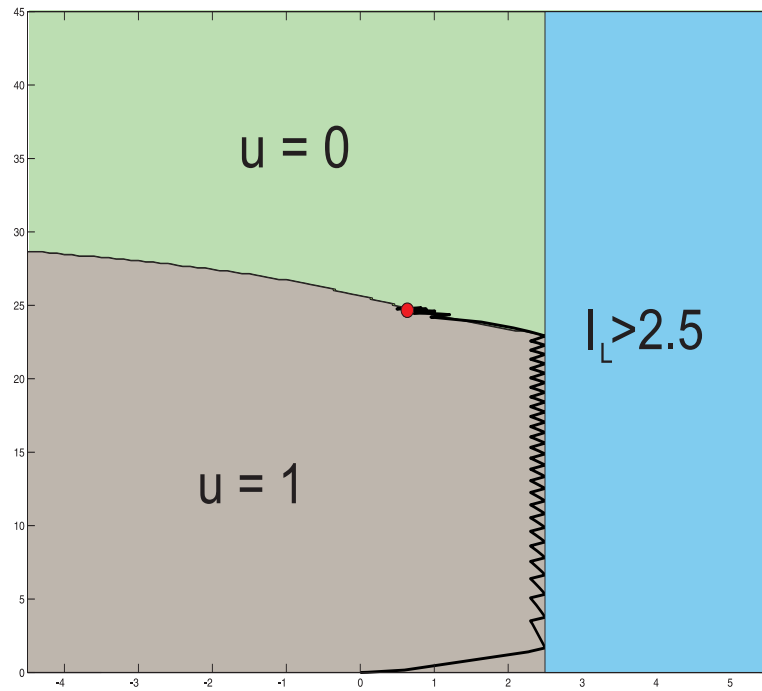


FIG. 4.8 – Retour d'état optimal et exemple de trajectoire dans l'espace d'état

2. Intégration en temps rétrograde : A partir des points sur l'arc singulier, nous faisons une intégration en temps rétrograde avec $u = 0$ et $u = 1$ pour un temps final $T = 10$ s.
3. Elimination de points de conflits : Pour ce convertisseur, il n'y a pas de points de conflit sur l'espace d'état. L'étape 4 de l'algorithme n'est pas nécessaire.
4. Interpolation : De ces courbes que l'on utilise comme points d'interpolation, l'apprentissage par réseau de neurones conduit à la partition de l'espace d'état donnée par la figure 4.8.

La zone de droite correspond à la prise en compte "après coup" d'une limitation du courant à $2.5A$. Autrement dit, la contrainte est ajoutée après la synthèse de la loi de commande. Une commutation est alors imposée si la contrainte de courant n'est pas vérifiée. Cette stratégie donne une zone de glissement à la frontière (Voir l'exemple d'une trajectoire dans l'espace d'état (figure 4.8).

Le mode de glissement sur les arcs singuliers est obtenu par l'utilisation d'une hystérésis comme bande de glissement ce qui conduit à un type de commande "asynchrone". La bande de glissement est ajustée de sorte que la période en régime permanent soit égal à $T_p = 0.05ms$. Cependant, comme l'approche est de nature "asynchrone", nous ne pouvons pas garantir que la période du cycle va être toujours 0.05 ms, pour tous les cas. Une approche synchrone est aussi

possible pour réaliser l'implémentation de la commande. Ce type d'approches a généralement besoin d'une synchronisation de la commande $u(kT_e)$ sur la période d'échantillonnage.

Remarque 14 *Le réseau de neurones utilisé a deux couches avec 10 neurones pour la couche cachée. L'algorithme d'apprentissage a convergé en 257 itérations.*

4.2.3.3 Résultats pratiques et de simulation

Les simulations de la réponse du système partant de conditions initiales nulles sont données sur la figure 4.9. Dans cette figure, nous montrons aussi un test de robustesse vis-à-vis de variations de la charge. Nous effectuons un changement de charge de $50\ \Omega$ à $100\ \Omega$ en $t = 7.5\ ms$ et de $100\ \Omega$ à $50\ \Omega$ à $t = 12.5\ ms$.

Un autre test de robustesse de la loi de commande est celui de changement de la valeur de capacité C . La figure 4.10 montre la réponse du système pour différentes valeurs de capacité. On observe un maintien parfait pour la tension de sortie à $25\ V$.

Finalement, le dernier test qui reste à faire est la variation de la source de tension en entrée. La figure 4.11 montre une la réponse lorsque la tension d'entrée change de $50V$ à $35V$ et de $35V$ à $50V$.

Toutes les trajectoires simulées suivent la valeur de référence parfaitement, même en présence de perturbation de charge ou perturbation de tension d'entrée. Pour cette dernière, nous constatons que le cycle limite change d'une période de $T_p = 0.05ms$ à $T_p = 0.1ms$ pour $E = 35V$. Le cahier de charge est toujours respecté. Nous pouvons constater sur la figure 4.12 un zoom sur le courant que le cycle limite change d'une période de $T_p = 0.05ms$ à $T_p = 0.1ms$ pour $E = 35V$.

4.2.4 Quelques commentaires des résultats

Les deux méthodes fournissent des résultats satisfaisants, car la tension de sortie arrive à la référence et respecte les contraintes imposées par le cahier de charge. Malheureusement, puisqu'un prototype n'était pas disponible au moment du développement de la méthode optimale pour ce convertisseur, nous n'avons pas pu tester cette méthode sur expérimentation.

Regardons les variables simulées dans la figure 4.4 (Commande prédictive) et la figure 4.9 (Commande optimale). Nous observons que la réponse de la commande prédictive est plus lente ($\approx 3.686ms$) que la réponse de la commande optimale $\approx 1.2ms$. La raison est la matrice de poids avec des valeurs "faibles" utilisée dans la commande prédictive. Ces valeurs dans la commande prédictive

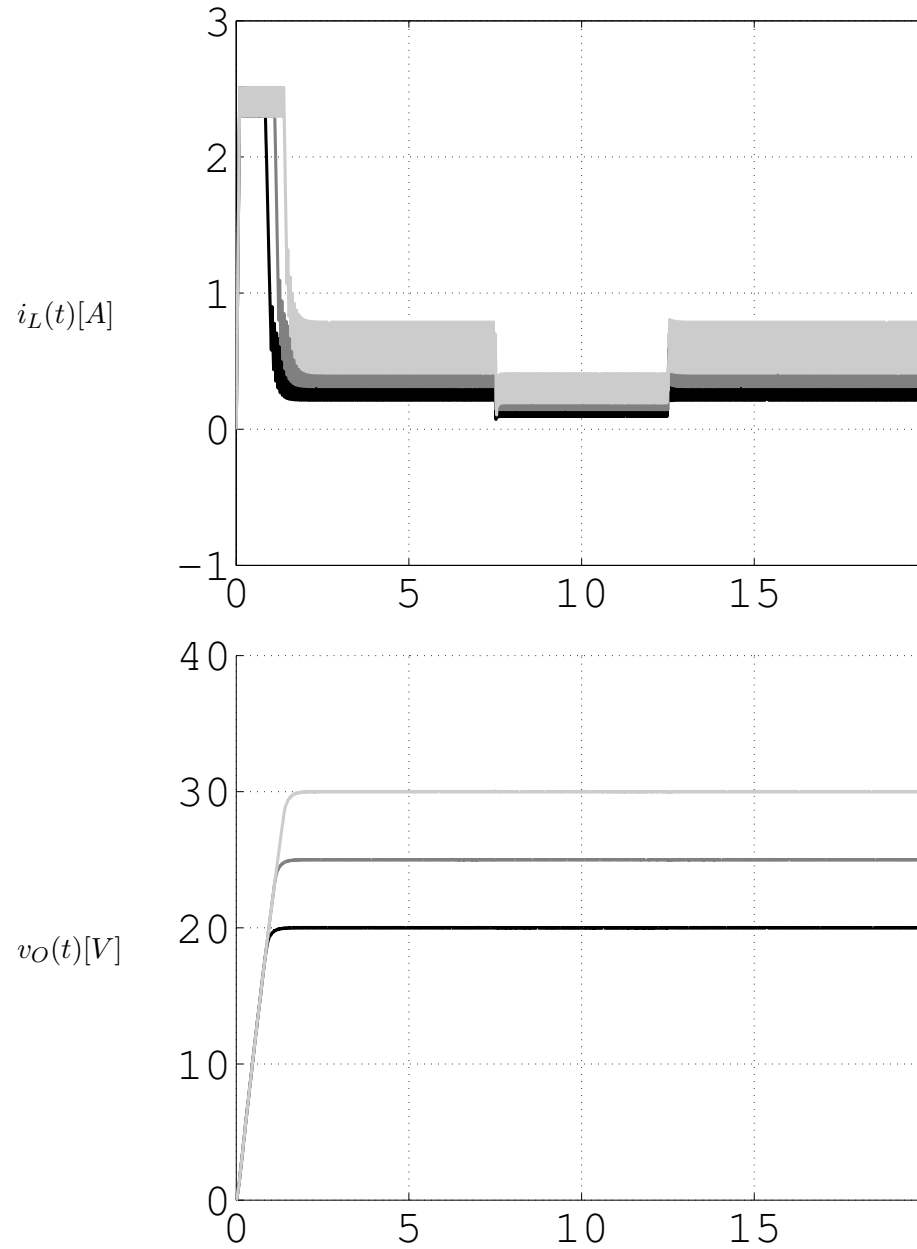


FIG. 4.9 – Test de démarrage. La charge change de $r_0 = 50\Omega$ à $r_0 = 100\Omega$ et de $r_0 = 100\Omega$ à $r_0 = 50\Omega$. La figure en dessus est le courant de l'inductance et la figure en dessous, la tension de sortie. Les valeurs de référence sont 20 V (noir), 25 V (gris) et 30 V (gris clair). L'échelle de temps est en ms .

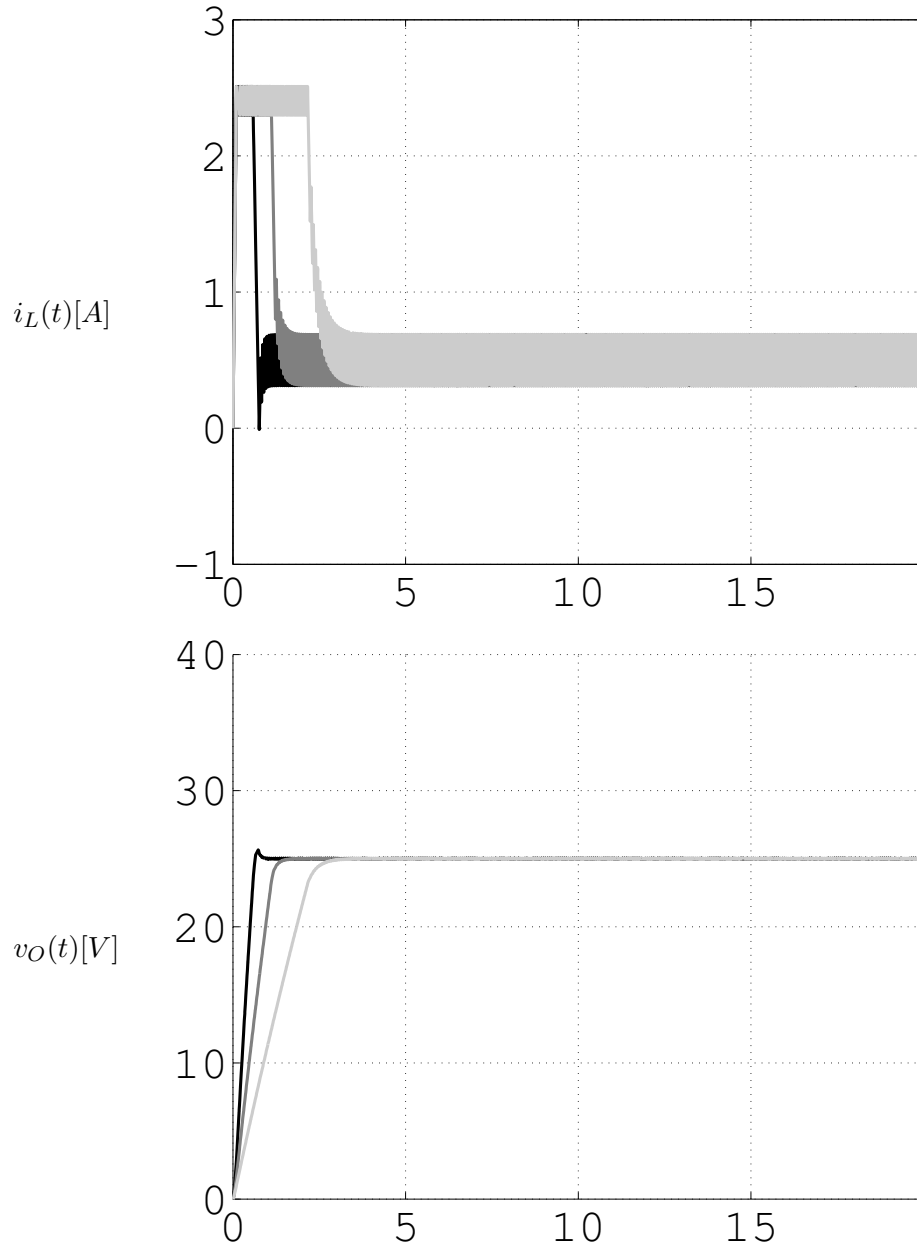


FIG. 4.10 – Test de robustesse. La capacité a une valeur différente à chaque fois. $x_c = 0.5x_{c,nom}$ (noir), $x_c = x_{c,nom}$ (gris) et $x_c = 2x_{c,nom}$ (gris clair). La valeur nominale est $x_{c,nom} = 100\mu F$. La figure en dessus est le courant de l'inductance et la figure en dessous est la tension de sortie. L'échelle de temps est en ms .

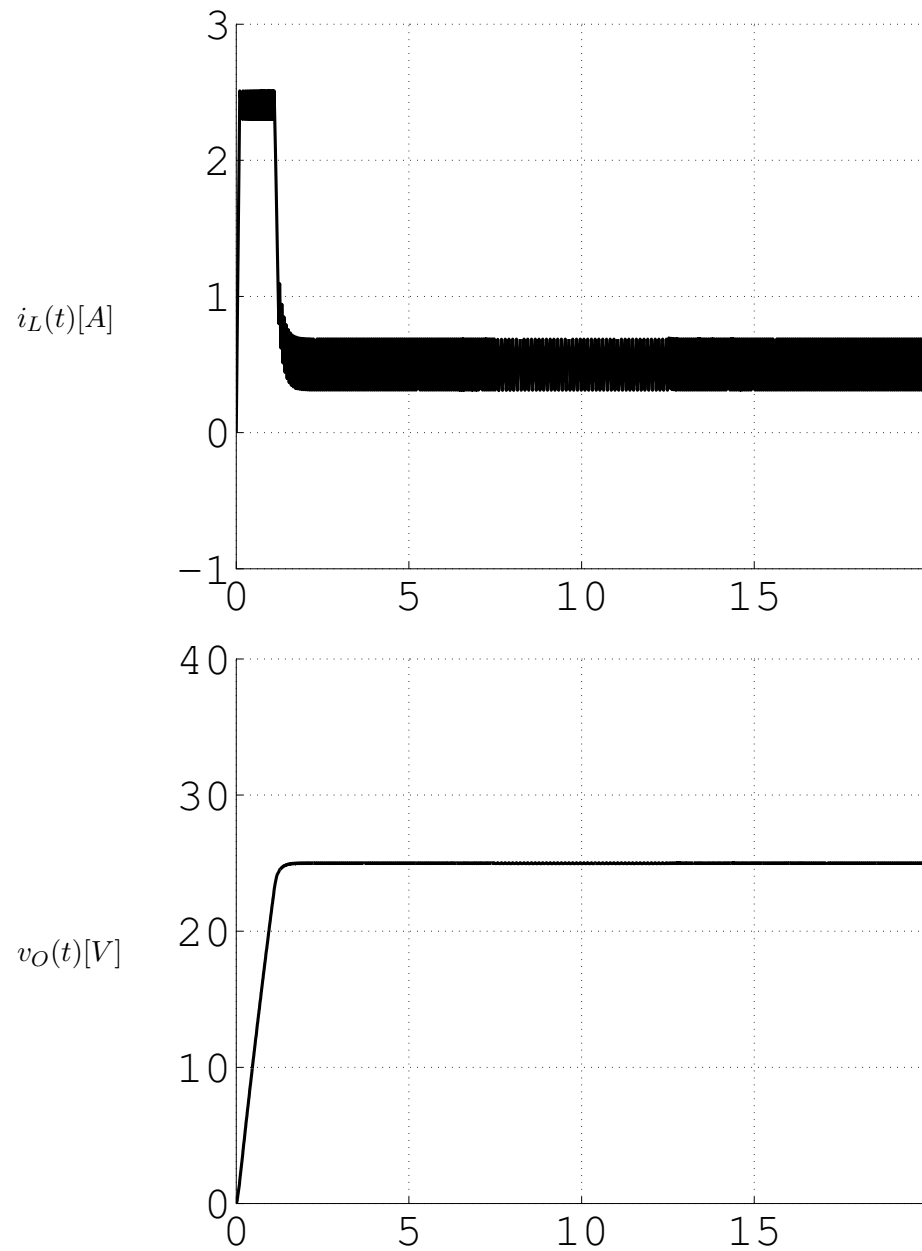


FIG. 4.11 – Test de variation de la source de tension. E change de $50V$ à $35V$ en $t = 7.5ms$. et de $35V$ à $50V$ en $t = 12.5ms$. L'échelle de temps est en ms .

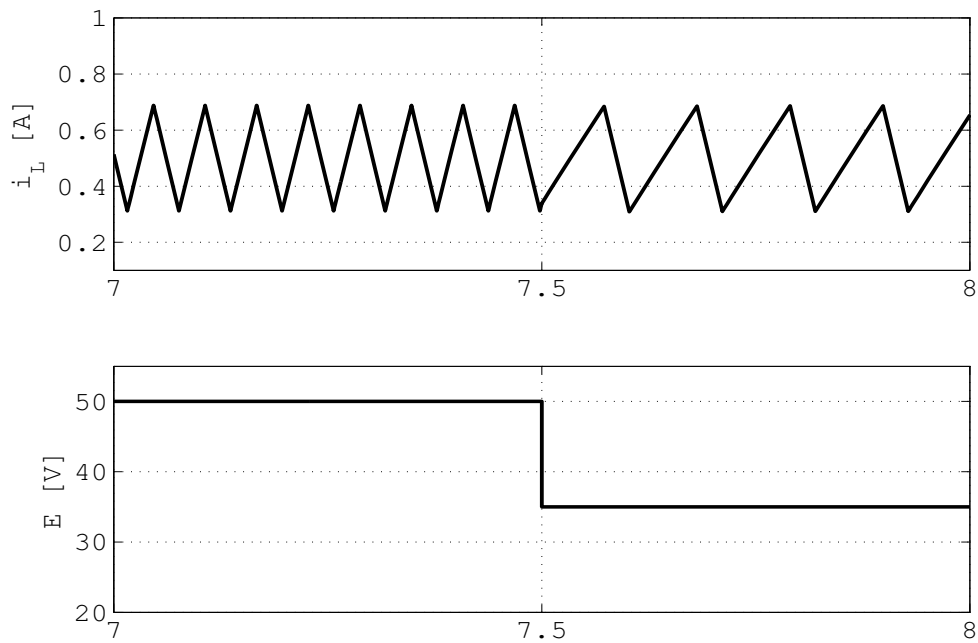


FIG. 4.12 – Commande en temps optimal

ont été choisies pour pas avoir des changements de temps t_1 très fortes entre deux périodes successives (chattering) et satisfaire la contrainte de courant.

Dans les mêmes figures (Figures 4.4 et 4.9), nous pouvons constater que le changement de la charge est bien compensé. Observons néanmoins que la commande optimale agit plus vite que la commande prédictive.

Concernant la variation des capacités, les résultats sont similaires (Voir Figure 4.5 et la Figure 4.10). Les commandes réagissent bien à la variation de paramètres. Puisqu'un changement de capacité implique un changement de la constante de temps, le temps de réponse change pour les deux approches. Par contre pour une valeur de $50\mu F$, nous observons un dépassement pas très important dans la commande optimale.

La grande différence entre les deux approches est observée dans les figures 4.6 et 4.11 lorsqu'un changement de la tension d'entrée est appliqué. La commande prédictive réduit la valeur du courant pour compenser cette variation, par contre la commande optimale, comme nous l'avons déjà mentionné, modifie la période du cycle. Nous expliquons ce phénomène avec la bande de glissement. En effet, cette bande de glissement a une amplitude constante est en conséquence, elle implique une amplitude constante des oscillations, mais une fréquence variable.

L'implémentation de l'algorithme 49 implique plusieurs procédures de minimisation. De plus, cette implémentation prend en compte la contrainte de courant et la contrainte sur les commutations. Une autre possibilité pour réduire le temps calcul est d'interpoler les solutions à l'aide d'un réseau de neurones. Cette interpolation n'est pas nécessaire ici grâce à la structure du système, nous évitons donc des approximations et nous travaillons directement avec l'algorithme.

4.3 Convertisseur multiniveaux

4.3.1 Topologie et état de l'art

Les applications industrielles avec des puissances de quelques mégawatts emploient généralement des tensions de plusieurs kilovolts. Les études et les développements effectués sur les convertisseurs multiniveaux, aussi appelés convertisseurs multicellulaires, au cours des dix dernières années ont montré d'excellentes caractéristiques [BB06], [DB00], [Chi03], [Gat97], [GFM⁺02]. En effet, ces systèmes sont particulièrement intéressants pour deux raisons : i) Ils permettent de réduire la tension dans les commutateurs en la divisant et la distribuant en niveaux intermédiaires, ii) Les tensions intermédiaires prennent des valeurs discrètes entre 0 et la tension d'entrée E . Ces niveaux de tension intermédiaires appliqués à la charge produisent une réduction des harmoniques, permettent une fréquence de commutation plus élevées et par conséquent, l'amélioration de la qualité spectrale. Un des inconvénients est que la commande de ce convertisseur devient plus complexe [MFT⁺02], [LP96].

La figure 4.13 montre le schéma d'un convertisseur à trois cellules. Sa fonction est d'alimenter une charge passive (R-L). En général, la structure d'un convertisseur multiniveaux résulte de la connexion de N sources de tension flottante placée en série de sorte que $N + 1$ niveaux discrets de tension de sortie soient obtenus. Les sources de tension sont donc : La tension de la source d'alimentation E et $N - 1$ capacités qui sont utilisées comme sources flottantes.

Nous devons donc réguler les tensions des capacités premièrement pour assurer une forme d'onde propre dans la charge [PRPK07] et deuxièmement pour limiter les tensions intermédiaires entre les commutateurs. Pour un convertisseur de N -niveaux, ceci peut être fait en réglant ces tensions des capacités aux valeurs de référence $\frac{E}{N}, \dots, \frac{(N-1)E}{N}$ [MH07a].

Plusieurs techniques ont été employées afin de trouver une loi de commande. Les premières approches sont des stratégies de boucle ouverte et elles consistent à réaliser un équilibrage par MLI (Modulation en Largeur d'Impulsion), pour un rapport cyclique fixé et décalé de $2\pi/N$ sur chaque cellule [MFT⁺02]. Malgré la

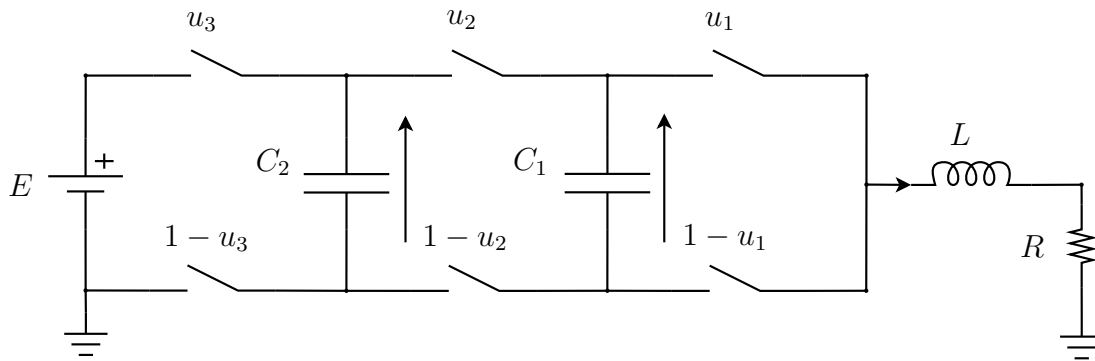


FIG. 4.13 – Convertisseur de quatre niveaux et trois cellules

qualité spectrale obtenue en régime permanent, la simplicité de ces approches mène à des performances très basses au niveau du transitoire (réponse très oscillante).

Les premières solutions en boucle fermée utilisent des régulateurs PI indépendants pour chaque tension intermédiaire afin de commander chaque cellule de commutation [SLCYDW01]. L'inconvénient est l'absence d'une manière systématique pour synthétiser les paramètres du PI.

Des autres méthodes comme les modes glissants ont été aussi employés pour ce convertisseur. En résumé, ces méthodes synthétisent une fonction de commutation par des fonctions de Lyapunov [PFM02]. Lorsque la charge est un moteur à induction, [EVA02] utilise un arbre de décision pour la commande des convertisseurs multiniveaux. Cette commande est basée sur une stratégie appelée Commande de Torque Direct. Enfin, [SMQG07] applique une commande prédictive classique et [DB00], [Don00] appliquent une commande spécifique aux convertisseurs multiniveaux.

Dans ce paragraphe, nous voulons appliquer les méthodes développées dans ce mémoire. Comme nous l'avons déjà mentionné, la méthode prédictive du chapitre II a comme avantage d'imposer un comportement cyclique autour du point de fonctionnement et la méthode du chapitre III donne une solution optimale pour un critère donné et par construction, elle garantit la stabilisation du système. Ces méthodes ont été testées dans un premier temps à SUPELEC - Rennes, [BPR⁺] montre les résultats de cette expérimentation. Les résultats présentés dans cette section ont été obtenus sur une maquette construite au CRAN.

4.3.1.1 Modèle mathématique

Le vecteur d'état du convertisseur de la figure 4.13 est composé par les tensions et le courant du circuit. Nous avons donc $x(t) = [v_{c_1}(t), v_{c_2}(t), i_L(t)]^T$. L'équation différentielle qui décrit ce système est :

$$\begin{aligned} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \end{bmatrix} &= \begin{bmatrix} -\frac{x_3(t)}{C_1} & \frac{x_3(t)}{C_1} & 0 \\ 0 & -\frac{x_3(t)}{C_2} & \frac{x_3(t)}{C_2} \\ \frac{x_1(t)}{L} & \frac{x_2(t)-x_1(t)}{L} & \frac{E-x_2(t)}{L} \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \\ u_3(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -\frac{R}{L}x_3(t) \end{bmatrix} \\ &= [S^1(x(t)), S^2(x(t)), S^3(x(t))]u(t) + R(x(t)) \end{aligned} \quad (4.17)$$

où $x_1(t)$, $x_2(t)$ sont les tensions de chaque condensateur et $x_3(t)$ le courant de la charge. Observons que ce modèle est affine en la commande u . $u_i(t)$, $i \in \{1, 2, 3\}$ représente la position des commutateurs qui fonctionnent par paire ou de façon complémentaire. Autrement dit, si $u_i(t)$ (voir la figure 4.13) est égal à 1, le commutateur supérieur est fermé et le commutateur inférieur est ouvert. Si $u_i(t) = 0$ le commutateur inférieur est fermé et le commutateur supérieur est ouvert.

4.3.1.2 Cahier de charges

La maquette du convertisseur, utilisée pour réaliser les tests, est montrée dans les figures 4.14 et 4.15.



FIG. 4.14 – Convertisseur multiniveaux du CRAN. Le système complet.

La figure 4.16 montre un diagramme par blocs de ses composants. En bref, elle est composée principalement par :



FIG. 4.15 – Convertisseur multiniveaux du CRAN. Chaque plaquette est un niveau du convertisseur.

- Carte DSPACE : La relation entrées-sorties obtenue et identifiée via le réseau de neurones est implantée et compilée sur ce dispositif. Autrement dit, l'algorithme de commande est chargé ici.
- Carte FPGA : La fonction de cette carte est d'assurer que les commutateurs de chaque cellule travaillent d'une façon complémentaire. De plus, cette carte a un algorithme de vérification du système (puissance, communication, etc.) avant de lancer la commande.
- Module de puissances : Chaque module de puissance représente chaque cellule du convertisseur. Ils ont été construits de manière modulaire. Nous pouvons augmenter et diminuer le nombre de cellules de 1 jusqu'à 5 cellules.
- Autres dispositifs : Arrêt d'urgence, Capteurs de courant et de tension, source de tension programmable.

L'objectif de ce convertisseur est d'assurer une valeur de courant moyenne constante. Il est possible de calculer les points de fonctionnement $i_{L,ref}$ en faisant $\dot{x}(t) = 0$ (Voir chapitre 1, paragraphe 1.4.2) et d'après le modèle (4.17), nous obtenons

$$i_{L,ref} = (E/R)u_3,$$

pour une valeur de commande moyenne fixée à $u_i \in [0 \ 1]$, $i = 1, 2, 3$ et

$$u_1 = u_2 = u_3.$$

Les tensions des capacités sont librement choisies. Comme nous l'avons déjà mentionné dans la topologie du convertisseur, afin d'améliorer la qualité du spectre et d'équilibrer les tensions aux bornes des commutateurs, les tensions de référence des capacités sont données par

$$v_{c2,ref} = \frac{2}{3}E \quad v_{c1,ref} = \frac{1}{3}E \quad (4.18)$$

Alors le vecteur de référence $x_{ref} = [v_{c1,ref}, v_{c2,ref}, i_{L,ref}]^T$.

Ce convertisseur a servi également de benchmark pour évaluer des méthodes de commande de systèmes commutés [AFJ⁺07], [PRI07], [BPC⁺07], [BPR⁺], [PRI08].

Les paramètres pour réaliser les simulations et obtenir les résultats pratiques sont $C_1 = C_2 = 40\mu F$, $L = 10mH$. et $R = 10\Omega$.

Le cahier de charge proposé est le suivant :

1. Pour une tension d'entrée $E = 30$, maintenir le courant de la charge à une valeur de référence donné $i_{L,ref} = 1$ A. et les tensions de capacité à $v_{c1,ref}$ et $v_{c2,ref}$ en minimisant les oscillations de courant. Obtenir un transitoire rapide.
2. Avoir une certaine robustesse vis-à-vis des variations de la tension d'entrée. La valeur de la tension d'entrée change de $E = 30V$ à $E = 15V$.
3. Les lois de commande $u(x)$ seront mises à jour à une fréquence d'échantillonnage de $1/T_e = 45$ kHz. En outre, pour des raisons physiques, $t_{min} = T_e$.

Remarque 15 Les résultats présentés ici insèrent un retard d'une demi-période ($t_{min}/2$) entre le moment où la commande est calculée et le moment où la commande est appliquée. Ce retard a été pris en compte pour les réponses simulées.

4.3.2 Commande prédictive

4.3.2.1 Détermination du régime permanent

La table 4.1 montre tous les modes de fonctionnement possibles.

Par programmation non-linéaire et avec $T_{p,max} = 0.4ms$, $s_{max} = 12$, $t_{min} = 1/45kHz = 0.022ms$, $\mathcal{Q} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 20000 \end{bmatrix}$ et prenant le point de fonctionnement

Mode	u_1	u_2	u_3
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

TAB. 4.1 – Equivalence entre modes et la valeur de chaque u_i

$x_0 = [10, 20, 1]^T$, nous vérifions l'équation (2.10) du chapitre II. La séquence de temps optimale et la séquence de modes optimale est alors :

$$\begin{aligned}\Omega^\infty &= \{0, 1, 3, 7, 2, 0, 4, 7, 4\} \\ \mathcal{T}^\infty &= \{0, 0.066ms, 0.088ms, 0.11ms, 0.132ms, 0.154ms, 0.22ms, 0.242ms, 0.264ms, \\ &\quad 0.286ms\}\end{aligned}\tag{4.19}$$

avec une condition initiale de $x(0) = [9.9247, 19.2928, 0.9823]^T$.

Observons d'après (4.19) que la période optimale qui minimise les oscillations est $T_p \approx 0.3$ ms. En outre, la séquence respecte le temps minimum de chaque commutateur t_{min} .

Remarque 16 Observons que la durée de chaque mode $t_{i+1} - t_i$, $i = 0, \dots, s^\infty - 1$ est multiple de T_e . Cela indique que le cycle est synchrone par rapport à $0.022ms$.

4.3.2.2 Construction de la boucle fermée

Pour cet exemple, nous avons employé une grille de 1000 points pour chaque variable d'état et calculé pour chacune la commande prédictive optimale solution du problème de poursuite.

Il existe deux manières d'implémenter la boucle fermée :

1. Une période d'échantillonnage $T_e \ll t_{min}$. Dans ce cas, la commande est donnée par le mode à utiliser et sa durée. Cette durée est donc facilement approchée avec des valeurs multiple de t_{min} .
2. Une période d'échantillonnage $T_e = t_{min}$. Dans ce cas, la commande est donnée par le mode. La durée à utiliser n'est pas très utile car seulement pour des cas très particuliers, elle est multiple de t_{min} . Nous devons donc

imposer comme une contrainte du problème de minimisation original des séquences de temps multiples de t_{min} .

Nous avons implanté la deuxième stratégie. Dans ce cas le réseau de neurones interpole le mode à utiliser par rapport à l'erreur vis à vis la référence. Ce réseau est composé par 50 neurones dans la couche cachée et utilise un algorithme d'apprentissage du type propagation rétrograde (back-propagation). Après l'apprentissage, le réseau de neurones est testé avec un ensemble de solutions connues qui n'a pas été utilisé pour l'apprentissage (Ensemble de validation). Si l'erreur due à cet ensemble n'est pas acceptable, nous augmentons le nombre de neurones dans la couche cachée.

Pour le courant de la charge, le domaine d'apprentissage va de 0 jusqu'à $2i_{L,ref}$, et pour les tensions des capacités de 0 à $2v_{c1,ref}$ pour C_1 et de 0 à $2v_{c2,ref}$.

4.3.2.3 Résultats pratiques et de simulation

Les indices de performances utilisés sont :

1. Test de démarrage. La figure 4.17 montre la réponse du système avec les paramètres nominaux. Nous constatons que la loi de commande rattrape la référence en 3 ms.
2. Test de variation de la ligne. La tension d'entrée change instantanément de valeurs durant l'opération. La figure 4.18 et la Figure 4.19 montrent le régime transitoire lorsqu'une variation de tension est appliquée : De $E = 30$ V à $E = 15$ V et de $E = 15$ V à $E = 30$ V.

Les figures montrent que la loi de commande peut alors compenser les différents changements de la source de tension.

Dans cette commande, les retards ont un effet important dans la réponse pratique et la réponse simulée. Cependant, le système arrive à un cycle limite au régime permanent même si on applique une variation de la source de tension.

4.3.3 Commande optimale

Comme pour le cas du convertisseur buck et afin d'être cohérent avec la notation du chapitre III, les variables d'état sont renommées à $z(t) = x(t) = [v_{c1}(t), v_{c2}(t), i_L(t)]^T$, $z_{ref} = [v_{c1,ref}, v_{c2,ref}, i_{L,ref}]^T$. Le critère est encore un critère quadratique de la forme :

$$J = \frac{1}{2} \int_0^\infty (z - z_{ref})^T \mathcal{Q} (z - z_{ref}) dt$$

où la matrice de poids est donnée par : $\mathcal{Q} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1000 \end{bmatrix}$.

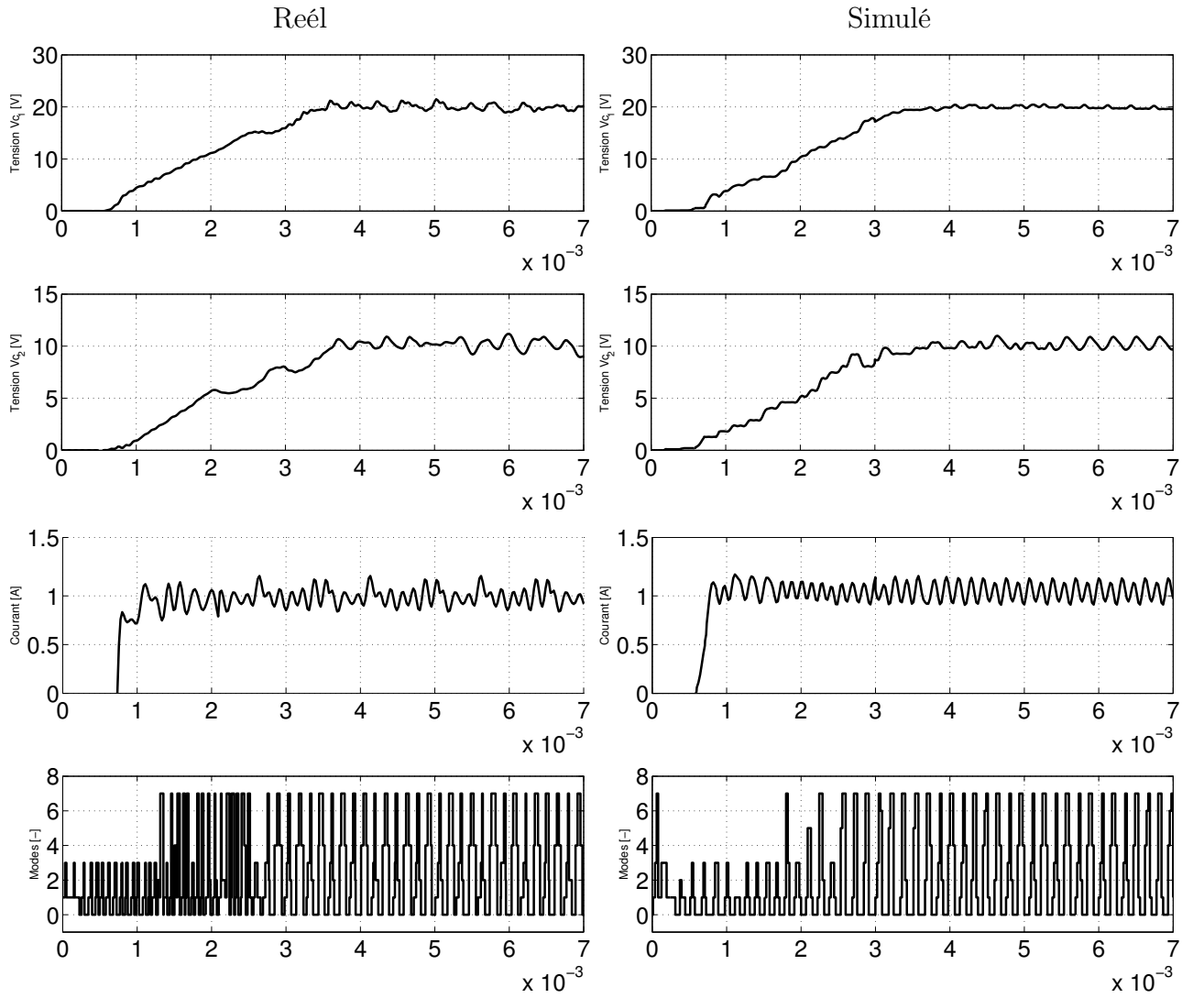


FIG. 4.17 – Test de démarrage. Commande prédictive. Comparaison entre les résultats simulés et les résultats pratiques.

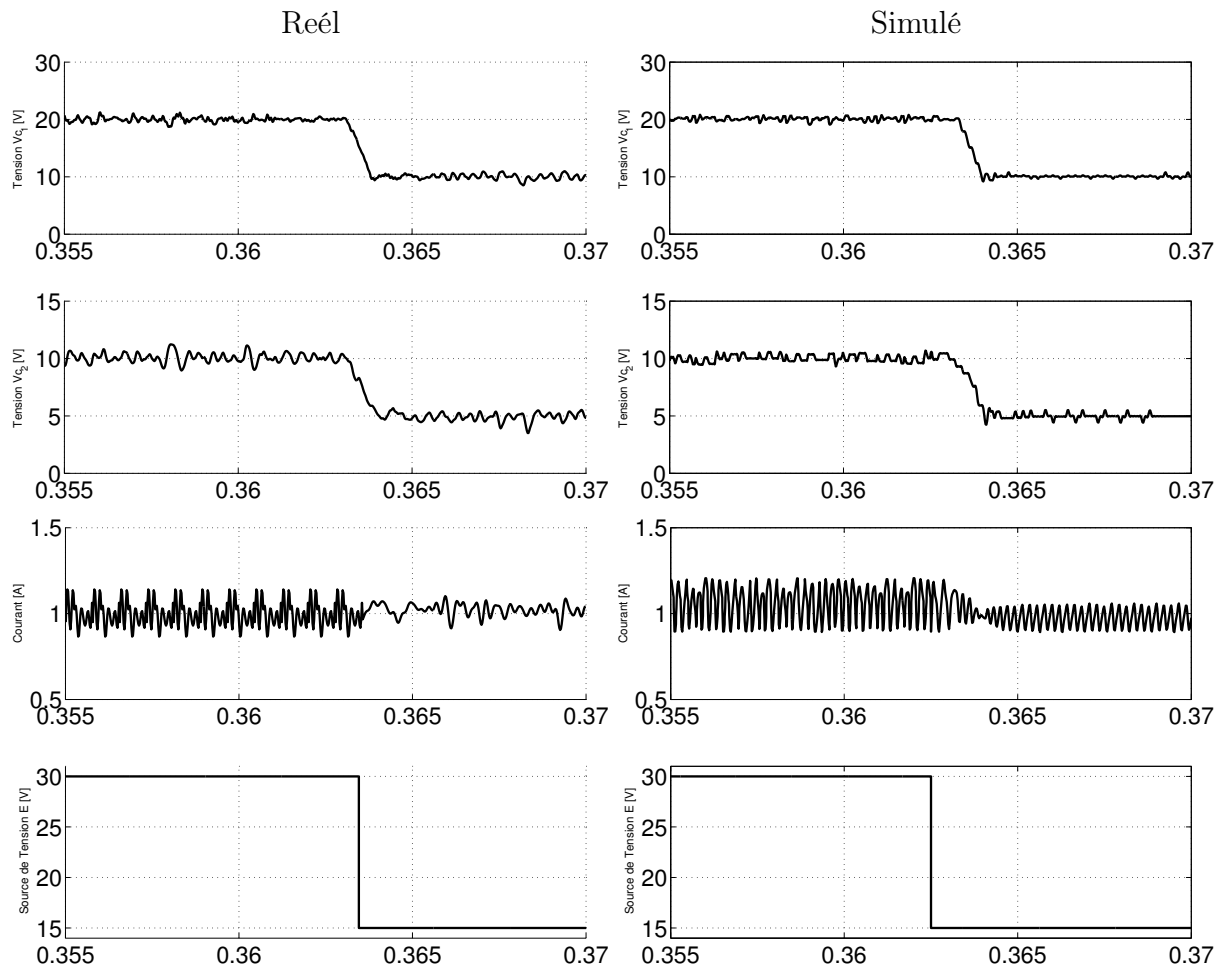


FIG. 4.18 – Test de variation de la source d’entrée. Commande prédictive. Comparaison entre les résultats simulés et les résultats pratiques. Un changement de la source de $E = 30$ à $E = 15$ V est appliqué

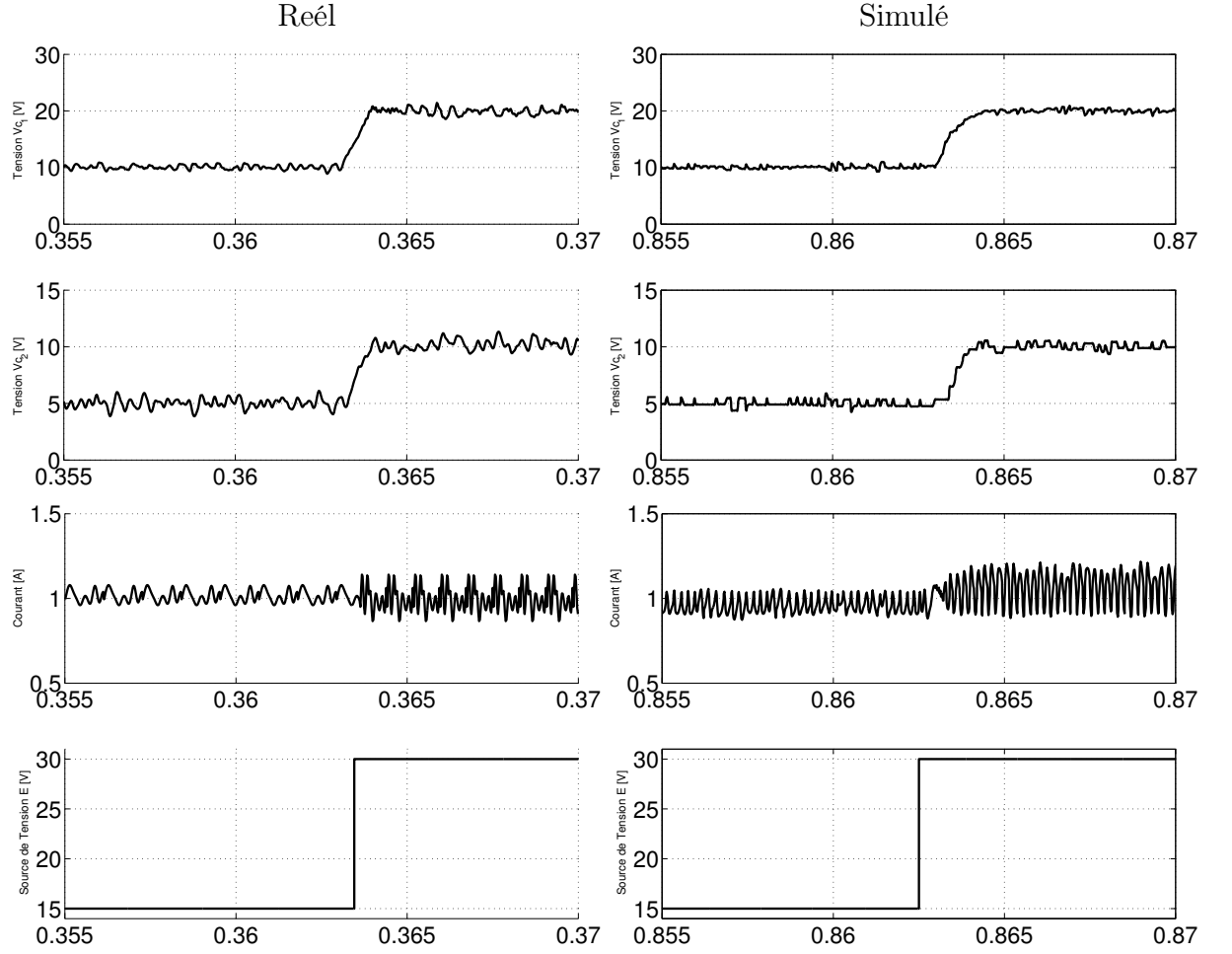


FIG. 4.19 – Test de variation de la source d'entrée. Commande prédictive. Comparaison entre les résultats simulés et les résultats pratiques. Un changement de la source de $E = 15$ à $E = 30$ V est appliqué

4.3.3.1 Détermination des arcs singuliers

Pour le convertisseur multiniveaux et en suivant les équations du système (4.17), nous avons

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \frac{1}{2} \int_0^\infty (z - z_{ref})^T \mathcal{Q}(z - z_{ref}) dt \end{bmatrix}$$

$$f(x) = \begin{bmatrix} R(z) \\ \frac{1}{2}(z - z_{ref})^T \mathcal{Q}(z - z_{ref}) \end{bmatrix} \quad g(x) = [g^1(x), g^2(x), g^3(x)] = \begin{bmatrix} S^1(z) & S^2(z) & S^3(z) \\ 0 & 0 & 0 \end{bmatrix}$$

Comme dans le cas du convertisseur buck, la fonction Hamiltonienne est identiquement nulle, $H(t) = 0$ grâce aux conditions de transversalité et le fait que $J < \infty$.

La fonction de commutation est :

$$\phi(x, \lambda) = \begin{bmatrix} \phi_1(x, \lambda) \\ \phi_2(x, \lambda) \\ \phi_3(x, \lambda) \end{bmatrix} = \begin{bmatrix} \frac{-x_3}{C_1} \lambda_1 + \frac{x_1}{L} \lambda_3 \\ \frac{x_3}{C_1} \lambda_1 - \frac{x_3}{C_2} \lambda_2 + \frac{x_2 - x_1}{L} \lambda_3 \\ \frac{x_3}{C_2} \lambda_2 + \frac{E - x_2}{L} \lambda_3 \end{bmatrix}$$

Premièrement, nous considérons les solutions singulières. Nous utilisons alors la proposition 46 du chapitre III pour obtenir l'ensemble d'équations suivant :

$$\begin{aligned} S_1(x, u) &= \det([f(x), g^1(x), ad_f g^1(x), ad_{g^2} g^1(x)]) \\ S_2(x, u) &= \det([f(x), g^2(x), ad_f g^2(x), ad_{g^1} g^2(x)]) \\ S_3(x, u) &= \det([f(x), g^1(x), ad_f g^1(x), ad_{g^3} g^1(x)]) \\ S_4(x, u) &= \det([f(x), g^3(x), ad_f g^3(x), ad_{g^1} g^3(x)]) \\ S_5(x, u) &= \det([f(x), g^2(x), ad_f g^2(x), ad_{g^3} g^2(x)]) \\ S_6(x, u) &= \det([f(x), g^3(x), ad_f g^3(x), ad_{g^2} g^3(x)]) \end{aligned}$$

$$S_7(x, u) = \det([g^1(x), ad_f g^1(x), ad_{g^3} g^1(x), ad_f^2 g^1(x) + \sum_{j \in \mathcal{R}} [g^j, ad_f g^1](x) u_j])$$

$$S_8(x, u) = \det([g^2(x), ad_f g^2(x), ad_{g^3} g^2(x), ad_f^2 g^2(x) + \sum_{j \in \mathcal{R}} [g^j, ad_f g^2](x) u_j])$$

$$S_9(x, u) = \det([g^1(x), ad_f g^1(x), ad_{g^2} g^1(x), ad_f^2 g^1(x) + \sum_{j \in \mathcal{R}} [g^j, ad_f g^1](x) u_j])$$

$$S_{10}(x, u) = \det([g^3(x), ad_f g^3(x), ad_{g^2} g^3(x), ad_f^2 g^3(x) + \sum_{j \in \mathcal{R}} [g^j, ad_f g^3](x) u_j])$$

$$S_{11}(x, u) = \det([g^2(x), ad_f g^2(x), ad_{g^1} g^2(x), ad_f^2 g^2(x) + \sum_{j \in \mathcal{R}} [g^j, ad_f g^2](x) u_j])$$

$$S_{12}(x, u) = \det([g^1(x), ad_f g^1(x), ad_{g^3} g^1(x), ad_f^2 g^1(x) + \sum_{j \in \mathcal{R}} [g^j, ad_f g^1](x) u_j])$$

$$\begin{aligned}
 S_{13}(x, u) &= \det([g^1(x), ad_f g^1(x), ad_f^2 g^1(x) + [g^1, ad_f g^1](x)u_1]) \\
 S_{14}(x, u) &= \det([g^2(x), ad_f g^2(x), ad_f^2 g^2(x) + [g^2, ad_f g^2](x)u_2]) \\
 S_{15}(x) &= \det([g^3(x), ad_f g^3(x), ad_f^2 g^3(x) + [g^3, ad_f g^3](x)u_3])
 \end{aligned}$$

La Table 4.3.3.1 montre comme à partir des équations précédentes, nous calculons les arcs singulières. Cette table est composée par :

- Dans la première colonne, les valeurs de l'ensemble $\mathcal{R}$.
- Dans la deuxième colonne, les termes f et les termes $g^i(x)u^i$, si $i \notin \mathcal{R}$. Nous devons donc fixer u^i aux valeurs 0 ou 1 pour pouvoir calculer la solution.
- Dans la dernière colonne, l'ensemble d'équation qui donne la solution (x, u) .

$\mathcal{R}$	Terme indépendant f_{drift}	Arc singulier
$\{1, 2, 3\}$	$f(x)$	$\bigcap_l \{(x, u), S_l(x, u) = 0\} \quad l = 1, \dots, 12$
$\{1, 2\}$	$f(x) + g^3(x)u_3$	$\{S_1 = 0\} \cap \{S_2 = 0\} \cap \{S_7 = 0\} \cap \{S_8 = 0\}$
$\{1, 3\}$	$f(x) + g^2(x)u_2$	$\{S_3 = 0\} \cap \{S_4 = 0\} \cap \{S_9 = 0\} \cap \{S_{10} = 0\}$
$\{2, 3\}$	$f(x) + g^1(x)u_1$	$\{S_5 = 0\} \cap \{S_6 = 0\} \cap \{S_{11} = 0\} \cap \{S_{12} = 0\}$
$\{1\}$	$f(x) + g^2(x)u_2 + g^3(x)u_3$	$\{S_{13} = 0\}$
$\{2\}$	$f(x) + g^1(x)u_1 + g^3(x)u_3$	$\{S_{14} = 0\}$
$\{3\}$	$f(x) + g^1(x)u_1 + g^2(x)u_2$	$\{S_{15} = 0\}$

TAB. 4.2 – Surface singulière. Cette table montre toutes les commandes singulières possibles et les équations associées pour les calculer

Toutes les solutions de cet exemple sont obtenues pour un ordre de l'arc $p = 1$. Dans le cas $p \geq 2$, les conditions nécessaires mènent à un ensemble vide. Par conséquent, il n'existe pas d'arcs singulières pour $p \geq 2$.

En résumé, afin d'obtenir les arcs singuliers, nous utilisons l'algorithme de la proposition 47 du Chapitre III :

Algorithme 50 (*Algorithme pour l'obtention des arcs singuliers pour un convertisseur multiniveaux*)

Données d'entrées

- Valeurs de R, L .
- Matrices de poids $\mathcal{Q}$.
- Région de l'espace d'état $([z_{i,min}, z_{i,max}], i = 1, 2, 3)$ et nombre de points pour chaque variable.

L'algorithme se décompose comme :

1. Déterminer les arcs singuliers candidats à l'optimalité suivant la table 4.2. Comme nous l'avons déjà remarqué $p = 1$.

2. Déterminer λ de sorte que λ soit orthogonal aux composantes de

$$\{h_k(x, u), k = 0, 1, \dots\}$$

$h_k(x, u)$ représente chaque colonne des matrices associées à chaque S_l de la table 4.2.

3. Evaluer les arcs singuliers afin d'obtenir une valeur numérique de u_i pour $i \in \mathcal{R}$ et pour le vecteur λ et vérifier que $u_i(x) \in \text{co}(U)$ pour chaque point.
4. Pour les commandes non singulières ($u_j(x)$, $j \notin \mathcal{R}$), il faut vérifier que :

$$u_j(x, \lambda) = \frac{1 - \text{sign}(\phi_i(x, \lambda))}{2} \quad (4.20)$$

Les commande qui ne vérifient pas cette condition et les points (λ, x) associé ne sont pas prises en compte.

5. Finalement, nous prenons les points qui vérifient la condition de Legendre-Clebsch : $-(H_u^{(2)})_u \geq 0$. Tout l'ensemble de points est alors stocké. Ces points composent les arcs singuliers.

4.3.3.2 Retour d'état

L'algorithme montré dans la section précédente permet d'obtenir les points sur les arcs singuliers. En effet, il donne les points (x, λ) sur les arcs singuliers. Il est possible donc de calculer la valeur du critère correspondant à chaque point sur ces arcs. Une fois que nous avons calculé les arcs singuliers admissibles, nous devons déterminer les arcs réguliers. Ceux-ci sont facilement générés par une intégration en temps rétrograde à partir des points singuliers du système Hamiltonien :

$$\dot{x} = f(x) + g(x)u \quad \dot{\lambda} = -\frac{\partial f(x)}{\partial x} - \frac{\partial g(x)}{\partial x}u$$

La condition initiale d'intégration est le point sur les arcs singuliers. A chaque pas d'intégration, nous obtenons la commande associé d'après l'équation (4.20) .

L'intégration en temps rétrograde de ces arcs provoque des "conflits" (Des commandes différents pour un même point sur l'espace d'état). Dans ce cas, la branche avec un coût plus élevé est coupée à partir du point de conflit (Voir section 3.5 du Chapitre III). Les trajectoires restantes sont interpolées avec un réseau de type feed-forward et composé par 30 neurones dans la couche cachée. La fonction d'activation des neurones est la fonction sigmoïde *logsig* définie par :

$$\text{logsig}(n) = \frac{1}{1 + e^{-n}}$$

Ce réseau a comme entrée l'écart entre la référence et l'état $[z - z_{ref}] = [z_1 - z_1^{ref}, z_2 - z_2^{ref}, i_l - i_l^{ref}]^T$ et comme sorties les trois commandes u_1 , u_2 et u_3 . La région d'intérêt dans l'espace d'état va de 0 à $2i_{L,ref}$ pour le courant, de 0 à $2v_{c1,ref}$ pour la tension de la première capacité et de 0 à $2v_{c2,ref}$ pour la tension de la deuxième capacité. De toutes les données obtenues de l'intégration en temps rétrograde, 70% des données font parties de l'ensemble d'apprentissage et le 30% restant servent de données de validation. L'algorithme d'apprentissage a pris 2586 itérations. Et l'erreur de validation est de l'ordre de $0.156e(-08)$.

Observons que lorsqu'une commande par retour d'état est déterminée, la commande définit aussi une partition de l'espace. L'espace d'état est donc divisé en régions dont certaines frontières sont composées par les arcs singuliers.

4.3.3.3 Résultats pratiques et de simulation

Les indices de performances pour ce test sont logiquement les mêmes que pour la commande prédictive.

1. Test de démarrage. La figure 4.20 montre la réponse du système avec les paramètres nominaux. Nous constatons que la loi de commande rattrape la référence en 3 ms.
2. Test de variation de la ligne. La tension d'entrée change instantanément de valeurs durant l'opération. La figure 4.21 et la Figure 4.22 montrent le régime transitoire lorsqu'une variation de tension est appliquée : De $E = 30$ V à $E = 15$ V et de $E = 15$ V à $E = 30$ V.

4.3.4 Quelques commentaires des résultats

Comme pour le convertisseur buck, les deux méthodes fournissent des résultats satisfaisants. Les tensions de capacités et le courant de l'inductance arrivent à la valeur désirée en moyenne. Cependant, les oscillations observées en régime permanent ne sont pas négligeables. La raison de ce comportement est la limitation physique de la fréquence de commutation à $45kHz$ des dispositifs de commutation. Un test intéressant serait de changer les commutateurs par des commutateurs à grande vitesse.

Concernant la réponse du démarrage, la commande prédictive (Voir figure 4.17) et la commande optimale (Voir figure 4.20) arrivent à un comportement cyclique. Observons que comme les méthodes n'ont pas été conçues pour des systèmes avec des bruits et avec des retards, il existe quelques différences entre la réponse simulée et la réponse réelle. Nous pouvons aussi constater que la réponse du système commandé avec la commande prédictive donne des amplitudes des oscillations en régime permanent inférieures à la réponse du système commandé avec

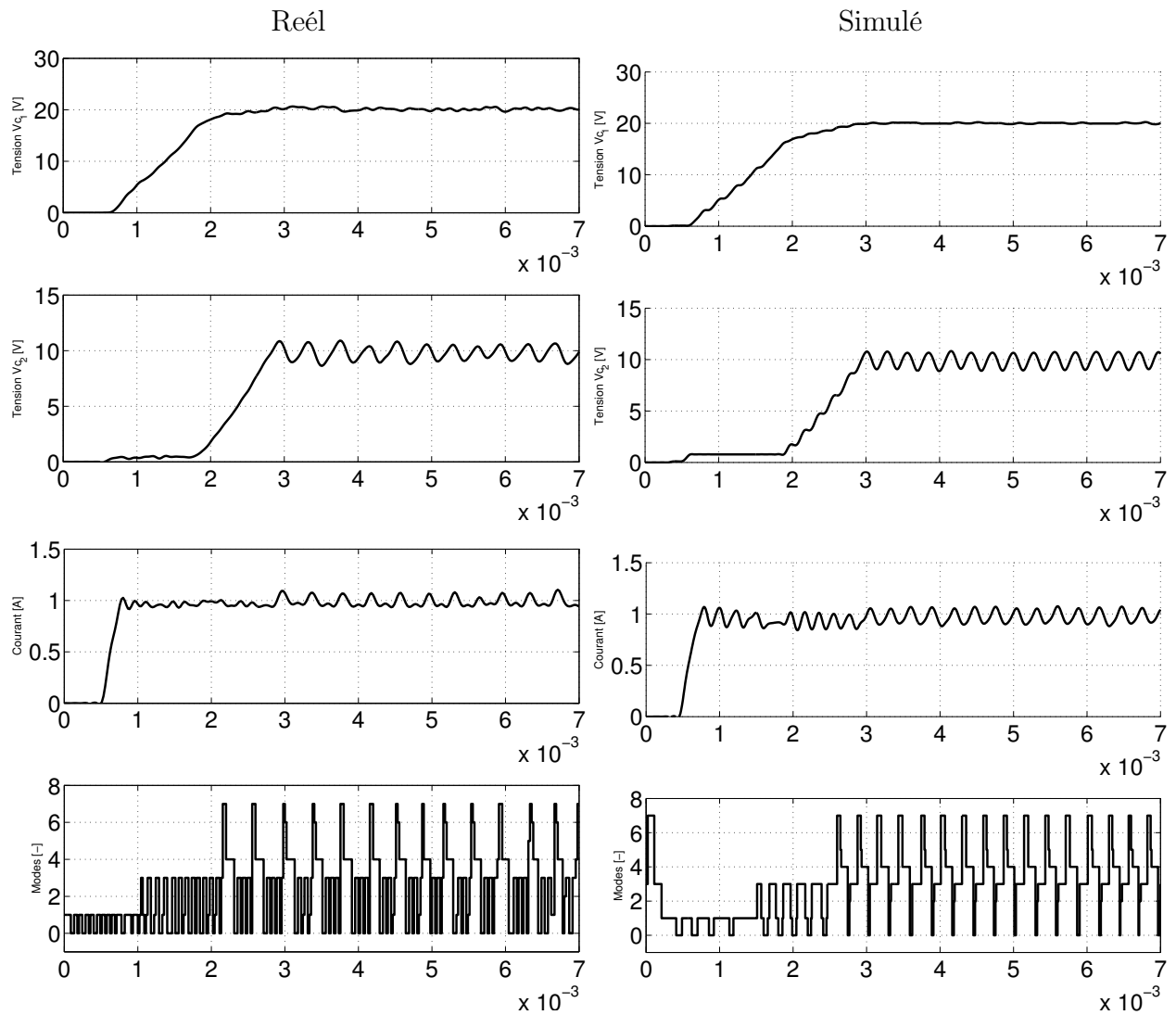


FIG. 4.20 – Test de d  marrage. Commande optimale. Comparaison entre les r  sultats simul  s et les r  sultats pratiques.

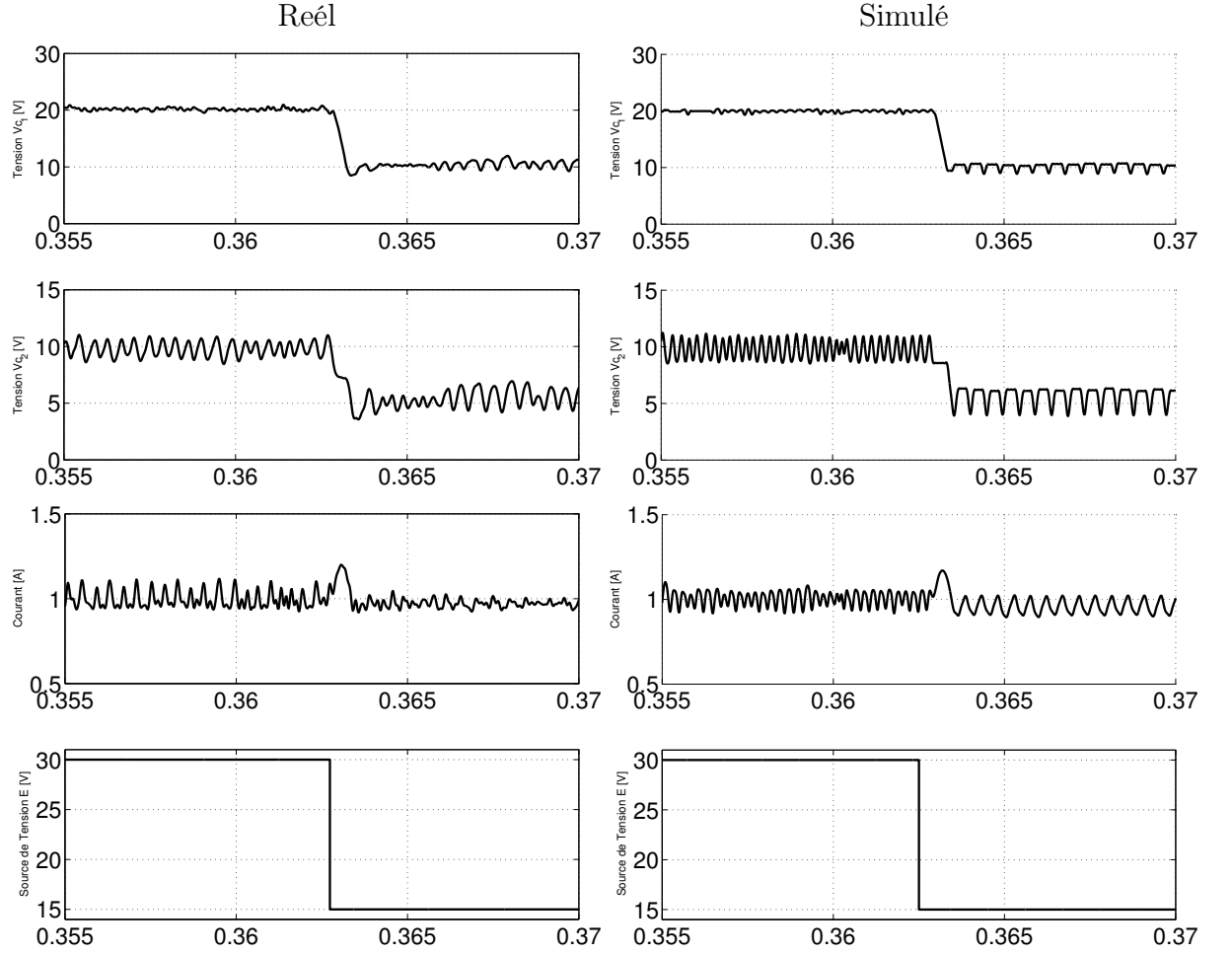


FIG. 4.21 – Test de variation de la source d’entr  e. Commande optimale. Comparaison entre les r  sultats simul  s et les r  sultats pratiques. Un changement de la source de $E = 30$    $E = 15$ V est appliqu  

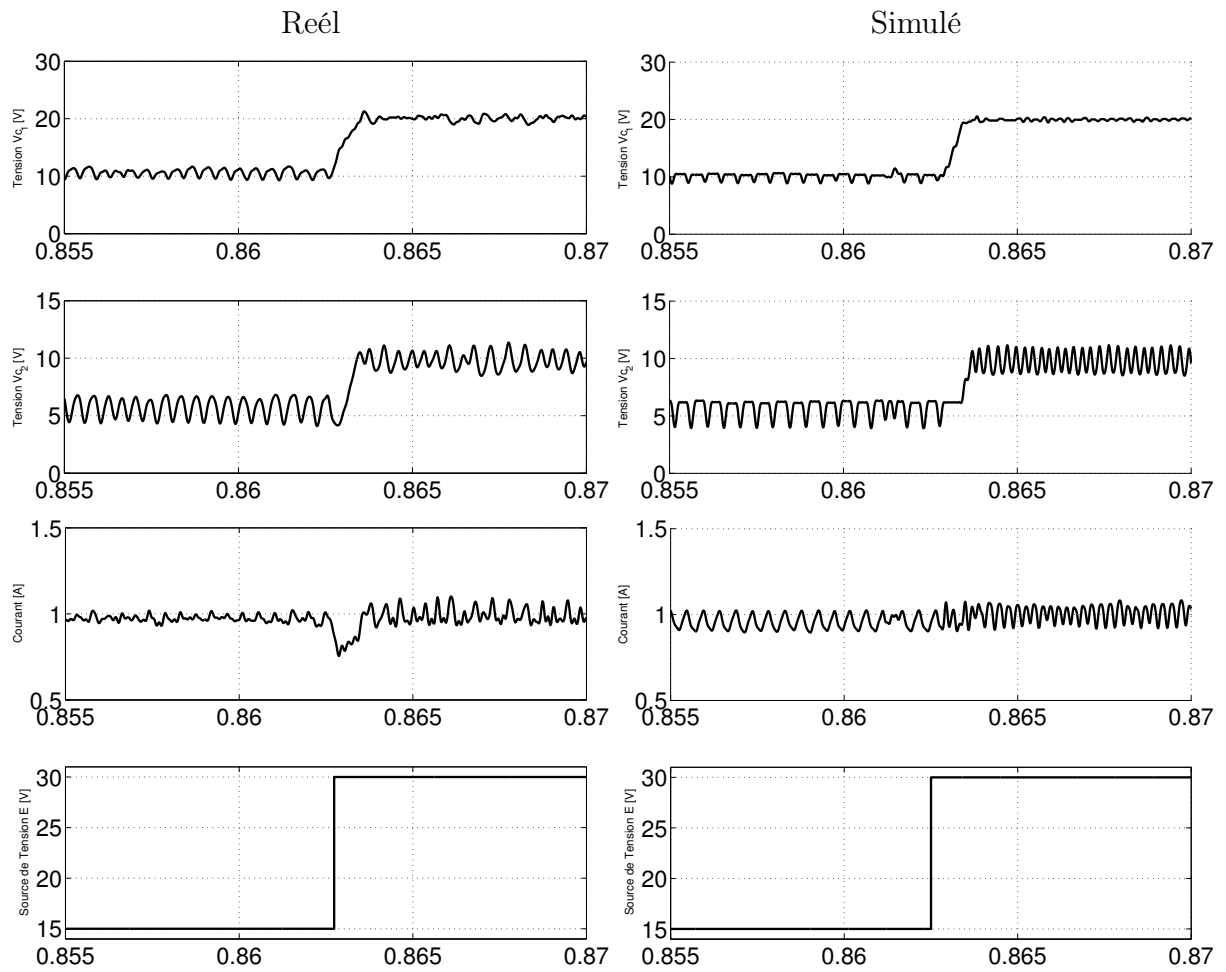


FIG. 4.22 – Test de variation de la source d’entrée. Commande optimale. Comparaison entre les résultats simulé et les résultats pratiques. Un changement de la source de $E = 15$ à $E = 30$ V est appliqué

la commande optimale. Les régulateurs arrivent donc à compenser les variations de la tension d'entrée.

4.4 Conclusion

Dans ce chapitre, nous avons illustré sur deux convertisseurs (Buck et multi-niveaux) les méthodes de commande prédictive et optimale proposées. Le premier est un convertisseur simple avec une seule variable de commande. Le deuxième convertisseur est plus complexe et possède trois variables de commande. La valeur de la commande à chaque instant représente une configuration des commutateurs du circuit.

Après avoir présenté d'une manière succincte le fonctionnement de chaque convertisseur, le modèle mathématique et le cahier des charges, ce chapitre a permis de valider sur des exemples réels, le bon fonctionnement des différentes algorithmes de commandes développés dans les chapitres précédents. L'approche prédictive permet de rattraper un cycle limite en régime permanent. Cette approche n'a pas une garantie de stabilité, mais présente les avantages d'être aisée à mettre en oeuvre. Dans le cas de la stratégie de la commande optimale, puisque la commande vérifie un critère quadratique, la stabilité est garantie.

Concernant la comparaison avec des approches classiques existantes [MFT⁺02], les performances des méthodes sont notamment supérieures particulièrement dans le transitoire. Les deux approches sont des approches par natures multi-variables. De plus, bien que les méthodes n'aient pas une garantie de robustesse, nous avons mis en évidence par des tests en simulation et sur systèmes réels la robustesse vis-à-vis d'échelons sur les paramètres. Par contre, un défaut que nous avons constaté dans la méthode prédictive est la sensibilité aux retards.

D'autres méthodes pour faire l'interpolation de la commande peuvent être envisagées. Cependant, les réseaux de neurones apportent une solution flexible malgré le grand nombre des paramètres (états et paramètres du système sur des intervalles déterminées) à prendre en compte. Rappelons que ces calculs sont effectués hors ligne.

Comme perspective de ce chapitre, nous pouvons mentionner :

- L'amélioration de chaque méthode (voir conclusion de Chapitre II et III) : Prise en compte de retard, des sauts autonomes, de robustesse dans la formulation de chaque méthode, etc.

- Il serait aussi très intéressant d'adapter les méthodes pour des cas où la référence n'est plus une valeur moyenne, mais un signal variant dans le temps. Ce cas est très important puisqu'il couvre les convertisseurs DC/AC ou AC/DC.
- L'inclusion d'un estimateur d'état dans la formulation de chaque méthode.
- Comme les méthodes de commande présupposent une valeur connue de la charge, un algorithme efficace en ligne pour l'estimation de charge peut améliorer les résultats obtenus.
- Les méthodes) de commande ont été conçues pour des systèmes continus. Une autre paramètre qu'on doit prendre en compte à l'avenir est la période d'échantillonnage.

Conclusion générale et perspectives

Les travaux présentés dans cette thèse, ont montré qu'il était possible d'effectuer la synthèse de commandes par retour d'état pour la classe des systèmes commutés cycliques en temps continu. Les lois de commande proposées conduisent à des algorithmes de commande facilement implémentables en temps réel. Deux méthodes adaptées à cette problématique ont été privilégiées : une méthode basée sur les concepts de la commande prédictive et une autre basée sur la théorie de la commande optimale.

L'originalité de la première est d'offrir la possibilité de spécifier un cycle optimal de ralliement au sens défini par un critère. Un algorithme rapide de Gauss-Newton, qui utilise les fonctions de sensibilité, est développé pour déterminer les commandes optimales. Les solutions du problème de poursuite sont alors interpolées via un réseau de neurones qui permet d'assurer une implémentation en temps réel.

La seconde approche basée sur le principe de minimum, permet de faire la synthèse de loi de commande optimale. Pour la classe des systèmes envisagés, les travaux exposés montrent que la synthèse des lois nécessite de faire la recherche d'arcs singuliers. La structure de ces systèmes et la nature des points de fonctionnement expliquent la présence de ces trajectoires particulières. Des conditions qui permettent de déterminer algébriquement les arcs singuliers pour des systèmes de dimension faible sont proposées. La synthèse sur une région de l'espace est alors obtenue par intégration en temps rétrograde depuis les points singuliers et interpolées par un réseau de neurones.

Les deux méthodes ont montré leur efficacité sur des applications de convertisseurs de puissance comme en atteste les résultats obtenus sur maquettes réelles lors de tests de robustesse.

Ces premiers travaux qui montrent la faisabilité de ces approches, sont encourageants. Cependant, nous pensons que ces résultats sont perfectibles sur un certain nombre de points.

Sur la synthèse de commande optimale, l'existence de trajectoires singulières est intrinsèque au problème et limite la portée de cette approche à des systèmes d'ordre faible. La discrétisation du problème est alors une voie de contournement possible via des algorithmes numériques de type programmation non linéaire. Cette approche aurait comme second avantage de supprimer le caractère asynchrone de la méthode au profit d'une commande synchronisée sur la période d'échantillonnage. Une autre issue peut être envisagée en utilisant la théorie des moments [MPP07].

La synthèse d'une commande optimale sur critère quadratique en temps infini est de fait une commande stabilisante. Cette garantie n'existe pas pour la commande prédictive. En revanche, cette dernière est un moyen flexible d'assurer un comportement particulier en régime permanent. Nous pensons que l'utilisation conjointe de ces deux méthodes : l'une pour la phase de ralliement et l'autre pour la poursuite d'un cycle particulier est à étudier.

La prise en compte des commutations autonomes est toujours un sujet difficile et largement inexploré. Pour le cas de la commande prédictive, l'inclusion des commutations autonomes dans l'étape de synthèse de la loi de commande est relativement simple et doit permettre de prendre en compte ce mode de fonctionnement particulier.

La robustesse des méthodes est également un point important non directement pris en compte.

Dans ce travail, seule une évaluation expérimentale de la robustesse a été faite, ce point mériterait sans aucun doute des développements théoriques. En ce qui concerne les convertisseurs statiques plusieurs points peuvent être améliorés.

Nous n'avons pas pu spécifier, faute de temps, de critères autres que la minimisation de l'erreur quadratique. La recherche d'une forme d'onde particulière n'a pas été suffisamment investie et l'optimisation du cycle sur un critère fréquentiel devrait permettre de réduire les perturbations induites sur la charge.

L'estimation des paramètres de charge mais également l'observation des variables d'état sont des problèmes ouverts sur la classe des systèmes considérés. Une amélioration simple et immédiate de la robustesse peut être obtenue en ajoutant un intégrateur sur l'erreur de poursuite. Nous pensons néanmoins qu'un schéma adaptatif pour l'estimation des paramètres de charge serait utile compte tenu de la sensibilité du modèle à ces paramètres.

Les commandes proposées nécessitent la mesure complète de l'état ; la réduction du nombre de capteurs, grâce à l'observation de ces grandeurs, augmenterait la fiabilité et réduirait les coûts.

L'hypothèse d'un point de fonctionnement fixe a été privilégiée dans cette étude, il est naturel d'envisager la prise en compte d'une référence variable, par exemple dans un fonctionnement en onduleur d'un convertisseur. La figure 1 présente, sur l'exemple du convertisseur multiniveaux et lorsque la référence est une sinusoïde, une utilisation directe de la loi de commande optimale obtenue au chapitre 3 et donc sans prise en compte de la variation de la référence.

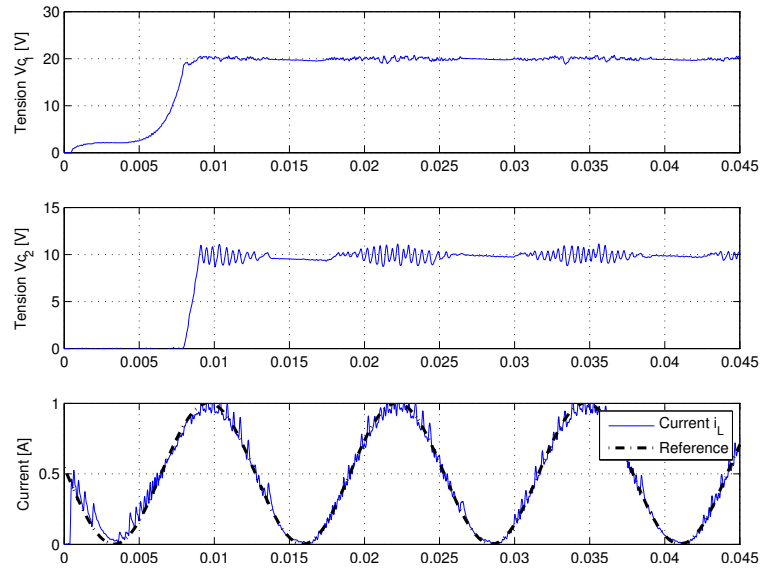


FIG. 1 – Commande optimale d'un convertisseur des quatre cellules fonctionnant comme onduleur

Annexe A

Détermination des lois de commande avec Matlab

Dans cette annexe, nous allons décrire l'implémentation des lois de commande depuis la conception. Le but est de détailler les algorithmes précédemment présentés dans les Chapitres 2 et 3.

Tout au long de l'annexe, nous utiliserons le convertisseur buck-boost afin de faciliter les explications des différents algorithmes. Dans un premier temps, nous montrerons la commande prédictive d'un point de vue algorithmique. Ensuite, nous procéderons avec l'algorithme de la commande optimale. Egalement, dans cette section, nous allons décrire comment utiliser, de manière générale, la toolbox de Matlab pour l'interpolation utilisant le réseau de neurones.

Rappelons les équations liées au convertisseur buck-boost. La dynamique de ce système, dont les variables d'état sont le courant dans l'inductance i_L et la tension aux bornes de la capacité v_c , a une dynamique :

$$\dot{x} = \begin{cases} \mathcal{A}_1 x + \mathcal{B}_1 & \text{pour } u = 1 \text{ (commutateur fermé - Mode 1)} \\ \mathcal{A}_2 x + \mathcal{B}_2 & \text{pour } u = 0 \text{ (commutateur ouvert - Mode 2)} \end{cases}$$

où

$$\begin{aligned} \mathcal{A}_1 &= \begin{bmatrix} 0 & 0 \\ 0 & -\frac{1}{RC} \end{bmatrix} & \mathcal{A}_2 &= \begin{bmatrix} 0 & \frac{1}{L} \\ -\frac{1}{C} & -\frac{1}{RC} \end{bmatrix} \\ \mathcal{B}_1 &= [\frac{E}{L}, 0]^T & \mathcal{B}_2 &= [0, 0]^T \end{aligned}$$

Supposons $R = L = C = E = 1$.

Remarque 17 *La méthodologie de programmation appliquée au convertisseur step-down ainsi qu'au convertisseur multiniveaux est très similaire à celle employée pour le convertisseur buck-boost.*

A.1 Commande prédictive

D'après la section 2.4.1, nous devons d'abord calculer le régime permanent grâce aux méthodes de programmation non-linéaire.

A.1.1 Calcul du régime permanent

Le critère est mis comme une variable d'état supplémentaire au système et le problème d'optimisation est donc sous la forme de Mayer. Le code de Matlab qui décrit le système est alors :

```
function [Xp] = buckboost(X,u,Xref)
%Buck-boost converter with a
quadratic cost function
%Inputs:
% X: State
% u: Control signal
%Outputs
% Xp: Vector field

Q = [100 0 0;0 1 0;0 0 0]; %Weight matrix

%System matrices
A1=[ 0 0;0 -1];
B1=[1; 0];
A2=[ 0 1;-1 -1];
B2=zeros(2,1);
Xe=X(1:2);
Xp = u*(A1*Xe+B1-A2*Xe-B2)+A2*Xe+B2; %System
Xp(3) = (X-Xref).'*Q*(X-Xref); %Cost function
```

Une méthode d'intégration du système d'équations différentielles sera nécessaire pour réaliser l'évaluation du critère et des contraintes. Nous choisissons l'algorithme de Runge-Kutta d'ordre 4 pour ses bonnes propriétés de convergence et à sa facilité de codification. Nous noterons cet algorithme comme RK4 par la suite.

Le premier pas de l'algorithme 20 du Chapitre 2 fixe la séquence de mode Ω et sa longueur s . Nous optimisons par rapport à $\mathcal{T} = \{0, t_1, T_p\}$ et par rapport à la valeur de l'état initial x_0 . Ces variables sont mises sur un seul vecteur var :

$$var = [t_1, x_0, T_p]^T \quad (\text{A.1})$$

Pour le cas du convertisseur buck-boost, nous avons que *var* a quatre composants. Le critère est codifié à l'aide de Matlab comme :

```
function [J,X] = fun(var,xinf,U, NN,tmin,Tpmax)
%Defining the cost function using a Ruge-Kutte 4 algorithm
%Inputs:
%  var:      Optimization variables. Vector containg [t_1;X0;T_p]
%  xinf:      Operation point
%  U:         Control sequence
%  NN:        Number of points used for the integration
%  tmin:      Minimal switching time
%  Tpmax:     Bound on the cucle period
%Output:
%  J:         Value of the cost function
%  X:         State value x(T_p)
tr(1) = var(1);
X = [var(2:3);1];
Tp = var(4);
tr(2) = Tp - tr(1);
for h=1:length(U)
    for k=1:NN
        [X]=RK4(@buckboost,U(h),X,tr(h)/NN,[xinf;0]);
    end
end
J = X(3);
end
```

et les constraints (2.7) et (2.9) sont implémentées sur une autre fonction Matlab :

```
function [c,ceq] = constraintt(var,xinf,U,NN,tmin,Tpmax)
;%Defining non-linear constraints
%Inputs:
%  var:      Optimization variables. Vector containg [t_1;X0;T_p]
%  xinf:      Operation point
%  U:         Control sequence
%  NN:        Number of points used for the integration
%  tmin:      Minimal switching time
%  Tpmax:     Bound on the cucle period
%Output:
%  c:         Innequality constrains in the form c<0
```

```
% ceq:      Equality constrains in the form ceq = 0
tr(1) = var(1)
X = [var(2:3);1];
Tp = var(4);
tr(2) = Tp - tr(1);
for h=1:length(U)
    for k=1:NN
        [X]=RK4(@buckboost,U(h),X,tr(h)/NN,[xinf;0]);
    end
end
%t_1 > t_min Elapsed time for the switch
c(1) = -var(1)+tmin;
%T_p < t_{max}
c(2) = Tp - Tpmax;
    %T_p > t_1      This is only to avoid non-logical answers
c(3) = -Tp+var(1);
    %Assuring that x(0) = x(T)
ceq(1) = X(1) - var(2);
ceq(2) = X(2) - var(3);
```

La dernière contrainte d'inégalité marquée comme $c(3)$ assure que la période soit plus grande que le temps de commutation t_1 ($T_p > t_1$). De manière générale : $T_p > t_{s-1}$.

Pour lancer la procédure d'optimisation, Matlab dispose de plusieurs fonctions. La fonction la plus adaptée au cas de programmation non-linéaire est la fonction `fmincon` (Voir l'aide de Matlab). Alors, pour chaque séquence de modes, nous devons lancer

```
[dt0,fval,exitflag]=fmincon(@fun,varin, [], [], [],
    [], [], [], @constraintt, options, xinf, U, NN, tmin, Tpmax);
```

où *varin* est un vecteur avec une solution initiale donnée par l'utilisateur. *fun* est le critère et *constraintt* est les contraintes définies précédemment. *dt0* est le vecteur des variables optimisées, *fval* est la valeur du critère et une valeur de *exitflag* = 1 indique si l'algorithme a bien fini l'optimisation.

Les paramètres que nous avons utilisés pour l'initialisation de l'algorithme sont : $t_{min} = 0.25$, $xinf = [2, -1]^T$, $NN = 10$, U varie entre toutes les séquences possibles de commande. $Tmax = 1$. Le vecteur *varin* est donc :

$$\begin{bmatrix} t_{min} \\ xinf \\ 0.3 \end{bmatrix}$$

A.1.2 Optimisation des temps de commutation

Une fois le cycle de référence est établi, nous nous intéressons aux trajectoires de ralliement. La méthode présentée dans l'algorithme 21 du Chapitre 2 propose l'itération de Gauss-Newton. Les équations de base pour le code Matlab ci-dessous sont déduites dans l'exemple 2.4.2.1 du Chapitre 2.

```
function t = buckboost_Newton(t,t0,Tp,x,xinf,Q,NN)
%Algorithm for computing the switching time for
%a buck boost converter
%Inputs:
%  t:      Initial guess of the solution. It is a vector
%          with the initial switching times
%  t0:     Current time
%  Tp:     Period of a cycle
%  x:      Current state. Also a vector.
%  xinf:   Cycle Steady state values. It is A matrix.
%          Each column is the
%          value of the cycle at each switching time
%  Q:      Weight matrix
%  NN:     Number of algorithm iterations
%Outputs:
%  t:      Switching times
Q = [Q zeros(3,3);zeros(3,3) Q];    %Augmenting the matrix
%Buck-Boost matrices
F1=[ 0 0;0 -1]; B1=[1; 0]; F2=[ 0 1;-1 -1]; B2=zeros(2,1);
%Augmenting the converter matrices (Affine -> linear)
A1 = [F1 B1;zeros(1,2) 0]; A2 = [F2 B2;zeros(1,2) 0];
%Gauss-Newton algorithm
for k=1:NN
%Prediction of the state
    xt1 = expm(A1*(t(1)-t0))*(x);
%Prediction error at t_1
    Delta_t1 =expm(A1*(t(1)-t0))*(x)-xinf(:,1);
%Prediction error at t_2
    Delta_t2 = expm(A2*(t(2)-t(1)))*xt1 -xinf(:,2);
%Put together in a vector
    Delta_X = [Delta_t1;Delta_t2];
    Gama_t1 = expm(A1*(t(1)-t0));          %Gama(t1,t0)
    Gama_t2 = expm(A2*(t(2)-t(1)));        %Gama(t2,t1)
    Delta_P1= (A1-A2)*Gama_t1;             %DeltaP_1
%Computing the Jacobian
```

```

    J = [(A1*Gama_t1*(x)).' (Gama_t2*Delta_P1*x).']';
    grad = J.'*Q*Delta_X;          %Gradient of the cost function
    H = J.'*Q*J;                  %Hessian matrix of the cost function
    t(1) = t(1) - inv(H)*grad; %Update
end
%Verifying feasibility of the solution
%Step 3 and 4 of the algorithm 22, Chapter 2
t(2) = t0+Tp;
% if the duration is > T_p
if (t(1)-t0)>Tp
    t(1) = t0+Tp;
    t(2) = t(1);
end
% if the duration is < 0
if (t(1)-t0)<0
    t(1) = t0;
end

```

Pour obtenir les résultats montrés dans l'exemple [2.4.2.1](#), nous avons employé les conditions initiales suivantes :

```

Tp = 0.5;
x  = [0;0];
t(1) = 0.25;
t0 = 0;
t(2) = Tp;
NN = 2;

```

A.2 Commande optimale

L'algorithme de commande optimale [48](#) du Chapitre 3 a comme première étape la détermination des arcs singuliers de manière algébrique. Nous avons besoin donc d'utiliser des outils de programmation symbolique pour faire ce calcul. Sur Matlab, il existe une toolbox dédiée au calcul symbolique. Ce toolbox nous permet de créer une fonction pour calculer le crochet de Lie comme :

```

function ad_fng = liebracket(f,g,x,n)
% LIEBRACKET (Nth order iterated Lie bracket of f and g)
%
% Usage:
% ad_fng = liebracket(f,g,x,n)
%

```

```

% Input:
% f = symbolic vector field of length l1 (first operand)
% g = symbolic vector field of length l1 (second operand)
% x = symbolic vector of variables
% n = order of the Lie Bracket
%
% Output:
% ad_fng = [ g      [f,g]  [f,[f,g]] [f,[f,[f,g]]] .... ]
%          n=  0      1      2      3
%          g      adf_g  adf2_g  adf3_g
%
%
% Note: define the symbolic variables as
%       x1 =sym(x1,'real')
%       in order to get rid of the conjugates
%       (only if they are reals).

ad_fng = sym(zeros(length(f),n+1)); ad_fng(:,1) = g;

if n>0
    for t = 2:n+1
        ad_fng(:,t) = jacobian(ad_fng(:,t-1),x)*f
            - jacobian(f,x)*ad_fng(:,t-1);
    end
end

ad_fng = expand(ad_fng);

et le crochet itéré de Lie :

function br = ad(f,g,x,n)
%Function to compute the iterated Lie bracket.
%      (n)
%      ad   g (x)
%      f
%Inputs:
%  f: Vector field f
%  g: Vector field g
%  x: Symbolic vector with the state vector
%  n: Order of the iterated Lie bracket
%Output
%  br: The iterated Lie bracket

```

```
ad_fng=liebracket(f,g,x,n); br = ad_fng(:,end);
```

Nous sommes donc en mesure de calculer les arcs singuliers après avoir défini les vecteurs f et g explicités par l'équation (3.73) du Chapitre 3.

```
function [s_a, contr] = singular_arcs_buckboost(f, g, x)
%Function to compute the singular arcs for a 2nd order system
%      Classic example: Buck-boost
%Inputs:
%  f:      Vector field f(x)
%  g:      Matrix field g(x)
%  x:      Symbolic vector. This one is used to compute the iterated Lie
%          bracket
%Outputs:
%  s_a:     The candidate singular arcs
%  contr:   Matrix that has [f g ad_f g];

syms x1 syms x2
contr = [f g ad(f,g,x,1)];      %Matrix to compute the singular arc
x3 = 1; contr = eval(contr);
S = det(contr);                 %Computing the determinant of the matrix
s_a = solve(S,'x1');             %Solving with respect to x1
```

Afin d'obtenir l'arc singulier optimal, la variable λ doit être orthogonale à la matrice $\{f(x), g(x), [f, g](x)\}$ comme l'exemple 3.6 du Chapitre 2 l'indique. De plus, la commande correspondant à chaque point singulier reste toujours à calculer. Egalement, l'algorithme doit vérifier la condition de deuxième ordre.

```
function [X_singul, Lambda_singul, u_singul, ns] =
    singular_compute_buckboost(s_a, f, g, x, x2_v)
%Function to compute the optimal singular arc for
%a buckboost converter
%Inputs:
%  s_a:     Candidates optimal singular arcs
%  f:      Vector field f(x)
%  g:      Matrix field g(x)
%  x:      Symbolic vector. This one is used to compute the iterated Lie
%          bracket
%  x2_v:    Values of the vector x_2
%Output:
%  X_singul: The state value of the optimal singular arc
%  Lambda_singul: The costate value of the optimal singular arc
```

```

% u_singul:    The control value of the optimal singular arc
% ns:         Vector that contains the number of points in
% each matrix that are singulars
%
%
syms x1
syms x2
contr = [f g ad(f,g,x,1)];           %Matrix to compute the singular arc
x3 = 1;
%contr = eval(contr);
contr = contr([1 2 4],1:3);
contr = eval(contr);
%Analysing all the possible candidates
for k=1:length(s_a)
    con = 1;
%For each point of the x2 variable
    for j=1:length(x2_v)
        x2 = x2_v(j);
%Evaluating each singular candidate solution
        x1_v(k) = (eval(s_a(k)));
        x1 = x1_v(k);
        if ~isnan(x1)
            x1 = real(x1);
            contr_v = (eval(contr.'));
%Lambda value as the null space of the matrix contr
            lambda = null((contr_v),'r');
        end
        if ~isempty(lambda)
            lambda(4) = 1;
            lambda(3) = 0;
            XX(:,con,k) = [x1;x2];
            Lambda(:,con,k) = [lambda(1);lambda(2)];
%Second order condition
            c_secord(con,k) = -(lambda.'*eval(ad(g,ad(f,g,x,1),x,1)));
%Control value
            u(con,k) = (lambda.'*(eval(ad(f,g,x,2))))/c_secord(con,k);
            con = con + 1;
        end
    end
end
tray = con-1;

```



```

u_r = real(u(1:tray,k));
pos = find(u_r>=0.1 & u_r<=1 & c_secord(1:tray,k)>=0);
%Saving the solutions which have a control between 0 and 1, 0<u<1 and
%verify the second order condition
X_singul(:,1:length(pos),k) = XX(:,pos,k);
Lambda_singul(:,1:length(pos),k) = Lambda(:,pos,k);
u_singul(1:length(pos),k) = u(pos,k);
ns(k) = length(pos);
end

```

Une fois obtenu l'ensemble de points x , λ et u singuliers, il reste à calculer la valeur de critère associée à chaque'un des points de cet ensemble. Ces valeurs seront utiles pour les cas d'un conflit. Nous implémentons la routine suivante sur Matlab :

```

function [cost] = initial_cost(X_singul, u_singul, ns, Q, NN, h, Xref)
%Funtion to obtain the initial cost value and to delete the singular arcs
%that do not converge to the equilibrium point
%Inputs:
% X_singul: Points in the state space corresponding to the sigular arcs
% u_singul: The control value of the optimal singular arc
% ns: Vector that contains the number of points from X_singul,
% Lambda_singul and u_singul that are really singular arcs.
% Q: Weight matrix
% NN: Number of points for the integration
% h: Integration step
% Xref: Equilibrium points
%Outputs:
% cost: The cost associated to each singular point
% If cost==NaN the point(i,j) associated to this cost
% does not converge to the equilibrium point.
%Initilalizing cost in zeros
cost = zeros(size(u_singul,1),size(u_singul,2));
elim = [];
for k=1:length(ns) %For all the candidate singular arcs
    for i = 1:ns(k) %For all the points in each singular arc
        %Initializing the system in the singular point
        X0 = [X_singul(:,i,k)+Xref;0];
        %Initializing the control in the singular point
        u = u_singul(i,k);
        j = 1;
    end
end

```

```

while norm(X0(1:2)-Xref)>0.1 & j<NN;
    %Intagrating with the control values corresponding to the
    %singular arc
    [X0]=RK4(@buckboost, u, X0, h, [Xref;0]);
    temp = (X0(1:2)-Xref)*ones(1,ns(k)) -
            X_singul(:,1:ns(k),k);
    dist = sqrt(temp(1,:).^2+temp(2,:).^2);
    [val,pos] = min(dist);
    u = u_singul(pos,k);
    j = j + 1;
end
cost(i,k) = X0(3);           %Stocking the cost
%If the integration did not converge mark the cost with NaN
if j==NN
    cost(i,k) = NaN;
end
end
end
end

```

Observons que la détermination des valeurs du critère passe par une intégration en prenant comme condition initiale chaque point sur l'arc singulier. Cette méthode montre si le point sur l'arc singulier converge vers le point de fonctionnement. Elle est donc utile pour éliminer des points qui n'amènent pas le système à la valeur désirée.

L'étape 2 de l'algorithme 48 du Chapitre 3 indique que nous devons réaliser une intégration en temps rétrograde à partir des valeurs de x et de λ situées sur l'arc singulier. Il suffit de fixer u à 0 et à 1. Cette intégration donnera lieu aux trajectoires régulières. La fonction suivante de Matlab permet réaliser le calcul des arcs régulières ainsi que le stockage de tous les arcs.

```

function [ZZ,UU,numtray,costt] =
backward_time_buckboost(A1, A2, B1, B2, xp, Q, X_singul, Lambda_singul,
                        ns, xmin, cost0)
%Function to time backward integrate the buck-boost equation from the
%singular arc
%Inputs:
%   A1:      A Matrix for mode one
%   A2:      A Matrix for mode two
%   B1:      B Matrix for mode one
%   B2:      B Matrix for mode two
%   xp:      Equilibrium point

```

```
% Q:      Weight matrix
% X_singul: Points in the state space corresponding
% to the singular arcs
% Lambda_singul: Points in the costate space being singular
% ns      : Number of points which are reglly trajectories
% xmin:    Vector containg the area for each state to
%           stablsh the feedback
% cost0:    The value of the cost function associated to each point in
%           the singular arc.
%Outputs:
% ZZ:      All the trajectories set [x;lambda]
% UU:      The control value for each trajectory point
% numtray: Number of the trajectory associated to each point
% costt:    The value of the cost function associated to each
%           point in the trajectory

%Writing the equations as an only autonomous system for each subsystem
%
% x'  = A1x+B1 ->[A1   B1] x
%           [0   0 ] x_{n+1}
%
QQ=[Q zeros(2,1);zeros(1,3)];
A11=[A1 A1*xp+B1;zeros(1,3)];
A22=[A2 A2*xp+B2;zeros(1,3)];
%Adding the dynamics of lambda for each systems
%lambda' = -Qx-A lambda
M(:, :, 1)=[A11 zeros(3);-QQ -A11.'];
M(:, :, 2)=[A22 zeros(3);-QQ -A22.'];
D=[ zeros(3,6);
    A11-A22 zeros(3)];
%-----Integration parameters-----
r = 200;
h = 0.05;
con = 1;
%Saving trajectories
ZZ = [];
UU = [];
numtray = [];
costt = [];
for k=1:length(ns)-1
```

```

    for i=1:ns(k)
        X0=[X_singul(1,i,k);X_singul(2,i,k);1;
            Lambda_singul(1,i,k);Lambda_singul(2,i,k);0];
        %changing the control value from the singular arc to u=0 and u=1
        for j=1:2
            %If the point converges to the equilibrium point
            if ~isnan(cost0(i,k))
                Ma = M(:, :, j) ;
                U = 2 - j; %Assigning control
                %Initializing
                xx=X0;
                Z = xx;
                cost = [h*xx(1:2).'*Q*xx(1:2)]; %Initial cost value
                numtray = [numtray con];
                for n=1:r
                    xx=expm(-Ma*h)*xx;          %Time Backward integration
                %Cost function value
                    cost = [cost h*(cost(end) + [xx(1:2).'*Q*xx(1:2)])];
                    %Is the trajectory inside the rang given by the vector
                    %xmin?
                    if (abs(xx(1))<=xmin(1) && abs(xx(2))<=xmin(2))
                        Z=[Z xx];
                %Control that follows the minimum principle
                    u=0.5*(1-sign(xx.'*D*xx));
                    U = [U u];
                    numtray = [numtray con];
                    Ma=u*(M(:, :, 1)-M(:, :, 2))+M(:, :, 2);
                else %If not, stop the backward integration
                    break
                end
            end
            end
        ZZ = [ZZ Z];
        costt = [costt cost];
        UU = [UU U];
        con = con + 1;
    end
end
end
end
end
end

```

Avant d'établir la loi de commande par retour d'état, l'algorithme 48 nécessite d'analyser les points de conflits éventuels. Une fois ces points identifiés, nous considérons la valeur du critère. La portion de trajectoire, dont la valeur de critère est la plus élevée, est alors coupée à partir du point de conflit. Le code suivant sur Matlab donne un ensemble de trajectoires sans confi. Les lignes de code sont commentées pour faciliter la compréhension de la méthode.

```
function [ZZ,UU,numtray] = conflicts(ZZ, UU, numtray, costt, tol)
%Function to eliminate the conflicts for the buck-boost converter
%Inputs:
% ZZ:      The whole set of trajectories
% UU:      The corresponding control for each point in ZZ
% numtray:  The corresponding number of trajectory
% costt:    The value of the cost function fdor each point
% tol:      A tolerance value which indicates the possible
%           conflict.
%           This value is ussually small.
%           For the buck boost = 0.05
%Outputs:
% ZZ:      The set of trajectories without conflicts
% UU:      The controls without conflicts
% numtray:  The number of the trajectory without conflicts
col = [];
%For each point in the trajectory set
for i=10:size(ZZ,2)
    %It does not start from i=1 because at the beginning all the
    %trajectories are near to each other. Therefore, all the points
    %could be consider as conflicts.
    %Distance from one point of the trajectory to the others.
    temp = (ZZ(1:2,i)*ones(1,size(ZZ,2))-ZZ(1:2,:)).^2;
    dist = sqrt(temp(1,:)+temp(2,:));
    %Obtaing the smallest distance.
    [val,pos] = sort(dist);
    %We take the second because the first one is the same point
    s_dis = val(2);
    s_pos = pos(2);
    %Detecting a conflict:
    %If the distance is less than the tolerance
    % the trajectory is different
    % the control is different
    % the trajectory is not just starting
```

```

        if (s_dis<tol & (numtray(s_pos)~=numtray(i)) &
UU(i)~=UU(s_pos) & numtray(i-9)==numtray(i))
            %Compare the cost value between both trajectories
            if costt(i)>costt(s_pos)
                tray = numtray(i);
                pn = i;
            else
                tray = numtray(s_pos);
                pn = s_pos;
            end
            %Save the position of the trajectory corresponding
            %to the one
            %with the higher cost value
            col = [col find(numtray==tray & 1:size(ZZ,2) > pn)];
        end
    end
end
%Eliminate the conflicts points and the trajectory
%portions with the highest cost value.
ZZ(:,col) = [];
UU(:,col) = [];
numtray(col) = [];

```

L'étape 5 de l'algorithme permet d'établir une loi par retour d'état grâce à l'interpolation par un réseau de neurones. Il existe une toolbox sur Matlab appelée "neural control toolbox" avec des fonctions spécifiques pour l'apprentissage. Les fonctions importantes que nous utiliserons sont :

- newff : Cette fonction permet de créer un réseau de neurones. Sa syntaxe est :

$$NET = NEWFF(P_R, [S_1, S_2, \dots, S_N], TF_1, TF_2, \dots, TF_N)$$

où

- P_R : Matrice $R \times 2$ avec le minimum et le maximum des éléments d'entrée.
- S_i : Taille de la couche i-ème, pour N couches.
- TF_i : Fonction d'activation liée à la couche i-ème.
- train : Cette fonction réalise l'apprentissage. Elle est utilisée comme :

$$NET = TRAIN(NET, P, T) \quad (A.2)$$

où

- NET : Réseau précédemment initialisé avec $NEWFF$.
- P : Entrées du réseau.

- T : Objectifs de sortie du réseau.

La sortie du réseau donne une structure avec un ensemble de paramètres. Ce qui nous intéresse sont les poids de la couche cachée et ceux de la couche de sortie. Afin de les récupérer, nous employons les instructions :

- $NET.IW\{1\}$: Poids de la couche cachée.
- $NET.LW\{2,1\}$: Poids de la couche de sortie.
- $NET.b\{1\}$: Vecteur de polarisation de la couche cachée.
- $NET.b\{2\}$: Vecteur de polarisation de la couche de sortie.

Pour le convertisseur buck-boost, nous avons implémenté la fonction suivante sur Matlab :

```
function net = feedback_neural_buckboost(ZZ, UU, range, nn, epochs)
%Function for obtaining a function input-output using a neural network from
%the trajectories of the buck-boost converter
%Inputs:
%  ZZ: Matrix that contains the trajectory values that will be used to
%      interpolate
%  UU: A vector that contains the control associated to each point in the
%      state space.
%  range: Matrix that describes the range of each state variable.
%  nn   : Number of neurons for the hidden layer
%  epochs: Number of iterations for the training
%Outputs:
%  net:   Structure with the neural network that must be used for the
%         control
%
%
%Example: net = feedback_neural_buckboost(ZZ, UU, [-5 5;-5 5], 10, 500)
net = newff(range,[nn 1],{'logsig' 'satlin'});
net.trainParam.epochs = epochs;
net = train(net,ZZ(1:2,:),UU);
```

Les paramètres correspondant aux résultats de l'exemple 2.4.2.1 du Chapitre 2 sont $net = feedback_neural_buckboost(ZZ, UU, [-55; -55], 10, 500)$.

Le réseau donne une fonction d'entrée-sortie. Observons que pour le convertisseur buck-boost, la fonction d'activation de la couche d'entrée est une fonction *logsig* définie comme :

$$logsig(n) = \frac{1}{1 - e^{-n}}$$

et pour la couche cachée est une fonction *satlin* :

$$satlin(n) = \begin{cases} 1 & n > 0 \\ 0 & n < 0 \end{cases}$$

A partir des poids obtenus pour chaque couche et les fonctions d'activation, la sortie pour un vecteur d'état arbitraire $x = [x_1, x_2]^T$ et nn neurones dans la couche cachée est alors obtenue comme :

$$u = \sum_{i=1}^{nn} satlin \left(wl_i \left[\sum_{j=1}^2 logsig(wi_{i,j}x_j + bi_j) \right] + bl_i \right) \quad (A.3)$$

où $wi_{i,j}$ sont les poids de la couche cachée ($NET.IW\{1\}$), wl_i sont les poids de la couche de sortie ($NET.LW\{2, 1\}$), bi_j est le vecteur de polarisation de la couche cachée ($NET.b\{1\}$) et bl_i est le vecteur de polarisation de la couche de sortie ($NET.b\{2\}$).

Bibliographie

- [AA04] M. Alamir and S.A. Attia. On solving optimal control problems for switched nonlinear systems by strong variations algorithms. In *Proceedings of NOLCOS 2004 Symposium on Nonlinear Control Systems*. Stuttgart, Germany, 2004.
- [AAaSI07] A.A. Al-Alawi and S.M. Al-Alawi and S.M. Islam. Predictive control of an integrated pv-diesel water and power supply system using an artificial neural network. *Journal of Renewable Energy*, 32 :1426–1439, 2007.
- [AAdW05] S.A. Attia, M. Alamir, and C. Canudas de Wit. Sub-optimal control of switched nonlinear systems under location and switching constraints. In *Proceedings of the 16th IFAC World Congress*, volume 16. Czech republic, 2005.
- [ACH⁺95] R. Alur, C. Coucoubetis, N. Halbwachs, T.A. Henzinger, P.H. Ho, X. Nicolin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138 :3 – 34, 1995.
- [AFJ⁺07] S. Almer, H. Fujioka, U. Jonsson, C.-Y. Kao, D. Patino, P. Riedinger, T. Geyer, G. Beccuti, G. Papafotiou, M. Morari, A. Wernrud, and A. Rantzer. Hybrid control techniques for switched mode dc-dc converters part i : The step-down topology. In *Proceedings of IEEE 26th American Control Conference*, volume 1, pages 5450–5457. New York City, USA, 2007.
- [Ala06] M. Alamir. A benchmark for optimal control problem solvers for hybrid nonlinear systems. *Automatica*, 42 :1593 – 1598, 2006.
- [ASY97] K. Alligood, T. Sauer, and J. A. Yorke. *Chaos : An Introduction to Dynamical Systems*. Springer-Verlag, 1997.
- [BB06] O. Béthoux and J.-P. Barbot. Commande permettant le contrôle du convertisseur multicellulaire série à nombre non premier de cellules. In *Conférence Internationale Francophone en Automatique*. Bordeaux, France, 2006.

- [BBC03] B. Bonnard, C. Bernard, and M. Chyba. *Singular Trajectories and their Role in Control Theory*. Series : Mathématiques et Applications, 2003.
- [BBM03] F. Borelli, M. Baotic, and M. Morari. An efficient algorithm for computing the state feedback optimal control problem. In *Proceedings of 1st IFAC Conference on Analysis and Design of Hybrid Systems*, volume 6, pages 4717 –4722. Saint Malo, France, 2003.
- [BCR02] J. Buisson, H. Cormerais, and P.Y. Richard. Analysis of the bond graph model of hybrid physical systems with ideal switches. *Journal of Systems and Control Engineering*, 216(11) :47–72, 2002.
- [BCZ⁺04] J. Buisson, H. Cormerais, M. Zainea, H. Gueguen, and E. Godoy. Formal approach to compute hybrid automata model for linear physical systems with switches. In *Proceedings of the 2004 IEEE Conference on Computer Aided Control Systems Design*, volume 1, pages 47–52, Taipei, Taiwan, 2004.
- [BD96] P. Boucher and D. Dumur. *La commande prédictive*. Editions technip, 1996.
- [Bel03] R.E. Bellman. *Dynamic Programming*. Princeton University Press, 2003.
- [Bem98] A. Bemporad. A predictive controller with artificial lyapunov function for linear systems with input/state constraints. *Automatica*, 34 :1255–1260, 1998.
- [Ber74] L.D. Berkowitz. Optimal control theory. *Applied Mathematical Sciences*, 12, 1974.
- [Ber84] D.P. Bertsekas. *Dynamic Programming and Optimal Control. Vol. 1 & 2*. Athena Scientific, 1984.
- [BFM00] A. Bemporad, G. Ferrari, and M. Morari. Observability and controllability of piecewise affine and hybrid systems. *IEEE Transaction on Automatic Control*, 4 :3966 – 3971, 2000.
- [BG08] L. Brush and D. Group. New techniques challenge the power supply industry. *Bodo's Power Systems*, (1) :20–21, 2008.
- [BM99a] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics and constraints. *Automatica*, 35(3) :407 – 427, 1999.
- [BM99b] A. Bemporad and M. Morari. Robust model predictive control : A survey. *Lecture Notes in Control and Information Sciences*, 1 :3–34, 1999.
- [BNT04] V. Blondel, Y. Nesterov, and J. Theys. Approximations of the rate of growth of switched linear systems. In *Proceedings of Hybrid*

-
- Systems : computation and control*, volume 2993, pages 173 – 186, 2004.
- [BPC⁺07] M. Baja, D. Patino, H. Cormerais, P. Riedinger, and J. Buisson. Hybrid control of a three-level three-cell converter. In *Proceedings of IEEE 26th American Control Conference*, volume 1, pages 5450–5457. New York City, USA, 2007.
- [BPFM07] A.G. Beccuti, G. Papafotiou, R. Frasca, and M. Morari. Explicit hybrid model predictive control of the dc-dc boost converter. In *Proceedings of 37th IEEE Power Electronics Specialist Conference*, volume 1, pages 2503 – 2509, Orlando, Florida, USA, 2007.
- [BPM06] A. Beccuti, G. Papafotiou, and M. Morari. Optimal control of the buck dc-dc converter operating in both the continuous and discontinuous conduction regimes. In *Proceedings of 45th IEEE Conference on Decision and Control*, pages 6205 – 6210. San Diego, USA, 2006.
- [BPR⁺] M. Baja, D. Patino, P. Riedinger, H. Cormorais, J. Buisson, and C. Iung. Alternative control methods for a four-level three-cell dc-dc converter. *In review. IEEE Transactions on Control Systems Technology*.
- [Bra98] M.S. Branicky. Multiple lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE Transaction on Automatic Control*, 43 :475–482, 1998.
- [BYC75] A.E. Bryson and H. Yu-Chi. *Applied optimal control*. Hemisphere Pub. Corp., 1975.
- [CA05] A. Chemori and M. Alamir. Limit cycle generation for a class of non-linear systems with jumps using a low dimensional predictive control. *International Journal of Control*, 78 :1206 – 1217, October 2005.
- [CB05] E.F. Camacho and C. Bordons. *Model predictive Control*. Springer, 2005.
- [CCC⁺07] G. Colin, Y. Chamaillard, A. Charlet, G. Bloch, and G. Corde. Exact and linearized neural predictive control : A turbocharged si engine example. *Journal of Dynamic Systems, Measurement, and Control*, 129 :527–533, 2007.
- [Chi03] J. N. Chiasson. Control of a multilevel converter using resultant theory. *IEEE Transactions on Control Systems Technology*, 11(3) :345 – 354, 2003.
- [CJT06] Y. Chitour, F. Jean, and E. Trélat. Genericity results for singular trajectories. *Journal of differential Geometry*, 73(1) :45 – 73, 2006.

- [Cla88] D.W. Clarke. Application of generalized predictive control to industrial processes. *IEEE Control Systems Magazine*, 8(2) :49–55, 1988.
- [Cla90] F.H. Clarke. *Optimization and Nonsmooth Analysis*. SIAM, 1990.
- [CMT87] D.W. Clarke, C. Mohtadi, and P.S. Tuffs. Generalized predictive control. *Automatica*, 23(2) :137–160, 1987.
- [Cor08] J. Cortes. Discontinuous dynamical systems. *IEEE Control Systems Magazine*, 4 :36 – 73, June 2008.
- [CR80] C.R. Cutler and B.C. Ramaker. Dynamic matrix-control - a computer control algorithm. In *Proceedings of the Automatic Control Conference*. San Francisco, USA, 1980.
- [DB00] A. Donzel and G. Bornard. New control law for capacitor voltage balance in multilevel inverter with switching rate control (cvc). In *Proceedings of the IEEE Annual Meeting of Industry Applications Society*, volume 3, pages 2037–2044. Rome, Italy, 2000.
- [Dem91] H. Demircioglu. Continuous time generalised predictive control. *Automatica*, 27(1) :55–74, 1991.
- [DG92] H. Demircioglu and P. J. Gawthrop. Multivariable continuous-time generalized predictive control. *Automatica*, 28(4) :697–713, 1992.
- [DITH04] M. Deng, A. Inoue, K. Takeda, and Y. Hirashima. Continuous-time anti-windup generalized predictive control of uncertain processes with input constraints and time delays. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, volume 5, pages 5053–5058. Piscataway, USA, 2004.
- [DK08] A.V. Dmitruk and A.M. Kaganovich. The hybrid maximum principle is a consequence of pontryagin maximum principle. *Systems & Control Letters*, 57 :964–970, November 2008.
- [Don00] A. Donzel. *Analyse géométrique et commande active sous observateur d’un onduleur triphasé à structure multicellulaire série*. Thèse de Doctorat. Institut National Polytechnique de Grenoble, 2000.
- [DRI02] J. Daafouz, P. Riedinger, and C. Iung. Stability analysis and control synthesis for switched systems : A switched lyapunov function approach. *IEEE Transaction on Automatic Control*, 74 :1883 – 1887, 2002.
- [EADMS07] A. El-Aroudi, M. Debbat, and L. Martinez-Salamero. Poincaré maps modeling and local orbital stability analysis of discontinuous piecewise affine periodically driven systems. *Nonlinear dynamics*, 50 :431 – 445, 2007.

-
- [EVA02] M. F. Escalande, J. C. Vannier, and A. Arzandé. Flying capacitor multilevel inverters and dtc motor drive applications. *IEEE Transactions on Industrial Electronics*, 49(4) :809 – 815, august 2002.
- [FA89a] G. Fraser-Andrew. Finding candidate singular optimal controls : A state of the art survey. *Journal of Optimization Theory and Applications*, 60(2) :173 – 190, 1989.
- [FA89b] G. Fraser-Andrew. Numerical methods for singular optimal control. *Journal of Optimization Theory and Applications*, 61(3) :377–401, 1989.
- [FA02] C.C. Fang and E. H. Abed. Robust feedback stabilization of limit cycles in pwm dc-dc converters. *Nonlinear dynamics*, 27 :295 – 309, 2002.
- [FJSR08] M. Fliess, C. Join, and H. Sira-Ramirez. Non-linear estimation is easy. *International Journal of Modelling, Identification and Control*, 4(1) :12 – 27, 2008.
- [FRL06] D. Flieller, P. Riedinger, and J. P. Louis. Computation and stability of limit cycles in hybrid systems. *Nonlinear Analysis*, 64(2) :352–367, 2006.
- [Gal05] Z. Galias. On rigorous study of poincaré maps defined by piecewise linear systems (electronic circuit example). In *Volume on IEEE International Symposium*, volume 4, pages 3407–3410, 2005.
- [Gat97] G. Gateau. *Contribution à la commande des convertisseurs statiques multicellulaires série*. Thèse de Doctorat. Institut National Polytechnique de Toulouse, 1997.
- [GC06] J.C. Geromel and P. Colaneri. Stability and stabilization of discrete time switched systems. *International Journal of Control*, 79 :719 – 728, 2006.
- [GDSA98] P. J. Gawthrop, H. Demircioglu, and I. Siller-Alcala. Multivariable continuous-time generalised predictive control : A state-space approach to linear and nonlinear systems. In *IEEE proceedings on control theory and applications*, 1998.
- [GFM⁺02] G. Gateau, G. Fades, P. Maussion, R. Bensaid, and T. A. Meynard. Multicell converters : Active control and observation of flying capacitor voltages. *IEEE Transactions on Industrial Electronics*, 49(5) :998 – 1008, 2002.
- [GMD03] J.M. Golcalves, A. Megretski, and M.A. Dahleh. Global analysis of piecewise linear systems using impact maps and surface lyapunov

- p>functions.
- IEEE Transactions on Automatic Control*
- , 48 :2089 – 2106, 2003.
- [Gon03] J.M. Goncalves. Regions of stability for limit cycles of piecewise linear systems. In *Proceedings of 42nd IEEE Conference on Decision and Control*, volume 1, pages 651–656, 2003.
- [GPM89] C.E. Garcia, D.M. Prett, and M. Morari. Model predictive control : Theory and practice - a survey. *Automatica*, 25(3) :335–348, 1989.
- [GPM05] T. Geyer, G. Papafotiou, and M. Morari. Model predictive control in power electronics : A hybrid systems approach. In *Proceedings of the 44th IEEE Conference on Decision and Control*, volume 12, pages 5606 – 5611. Seville, Spain, 2005.
- [HB04] M. Hovd and R.R. Bitmead. Interaction between control and state estimation in nonlinear mpc. In *Proceedings of 7th International Symposium on Dynamics and Control of Process Systems*. Boston, USA, 2004.
- [HBY91] F. Huliehel and S. Ben-Yaakov. Low frequency sampled data models of switched mode dc-dc converters. *IEEE Transactions on Power Electronics*, 6(1) :55–61, 1991.
- [His01] I.A. Hiskens. Stability of hybrid system limit cycles : Application to the compass gait biped robot. In *Proceedings of the 40th IEEE Conference on Decision and Control*, volume 1, pages 774–779, 2001.
- [His04] A.K. Hishan. *La commande optimale des systèmes dynamiques*. Lavoisier, 2004.
- [HP00] I.A. Hiskens and M.A. Pai. Trajectory sensitivity analysis of hybrid systems. *IEEE Transaction on Circuits and Systems*, 49(2) :204–220, 2000.
- [HR07] I.A. Hiskens and P.B. Reddy. Switching-induced stable limit cycles. *Nonlinear Dynamics*, 50 :575 – 585, 2007.
- [HSB01] W.P.M.H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7) :1085–1091, 2001.
- [HSSR03] V.M. Hernandez, R. Silva, and H. Sira-Ramirez. On the stability of limit cycles in resonant dc-to-dc power converters. In *Proceedings of the 42th Conference on Decision and Control*, volume 2, pages 1141–1146, 2003.
- [ILG78] C. Iung, J.P. Louis, and V. Gelezevicius. Analysis of the validity of linear approximations for the regulation of dc motor fed by ac/dc

-
- converters. In *Proceeding of International Conference on Electrical Machines*, 1978.
- [ISW03] B. Ingalls, E. D. Sontag, and Y. Wang. An infinite-time relaxation theorem for differential inclusions. In *Proceedings of the 2002 American Mathematical Society*, volume 131, pages 487–499. Providence, USA, 2003.
- [JGL70] D.H. Jacobson, S.B. Gershwin, and M.M. Lele. Computation of optimal singular controls. *IEEE Transaction on Automatic Control*, 15(1) :67 – 73, 1970.
- [JKN00] R. Johnson, M. Kamenskii, and P. Nistri. Bifurcation and multiplicity results for periodic solutions of a damped wave equation in a thin domain. *Journal of Computational and Applied Mathematics archive*, 113 :123 – 139, 2000.
- [Joh02] M. Johansson. Piecewise linear control systems - a computational approach. In *Lecture Notes in control and Information letters*. Germany, 2002.
- [JR98] M. Johanssons and A. Rantzer. Computation of piecewise quadratic lyapunov function for hybrid systems. *IEEE Transactions on Automatic control*, 43 :555–559, 1998.
- [KFA69] R.E. Kalman, P.L. Falb, and M.A. Arbib. *Topics in Mathematical System Theory*. Mc Graw-Hill, 1969.
- [Kha01] H.K. Khalil. *Non-linear systems*. Prentice Hall, 2001.
- [Kre77] A. J. Krener. The higher order maximal principle and its application to singular extremals. *SIAM Journal of Control and Optimization*, 15 :256 – 293, 1977.
- [Law73] M. Lawrence. Optimal control of limit cycles or what control theory can do to cure a heart attack or to cause one. In *Proceedings of Symposium on Ordinary Differential Equations*, volume 312, pages 108–134, 1973.
- [LJS⁺03] J. Lygeros, K. H. Johansson, S. N. Simic, J. Zhang, and S. S. Sastry. Dynamical properties of hybrid automata. *IEEE Transaction on Automatic Control*, 48(1) :2–16, 2003.
- [LK04] L. Lazar and R. De Keyser. Non-linear predictive control of a dc-to-dc converter. In *Proceedings of the Symposium on Power Electronics, Electrical Drives, Automation and Motion*. Capri, Italy, 2004.
- [LMG94] J.H. Lee, M. Morari, and C.E. Garcia. State-space interpretation of model predictive control. *Automatica*, 30(4) :707–717, 1994.

- [LP96] J. S. Lai and F. Z. Peng. Multilevel converters-a new breed of power converters. *IEEE Transactions on Industrial Applications*, 32(3) :2348 – 2356, 1996.
- [LR07] H. Logemann and E. P. Ryan. *Stability and Asymptotic Behaviour of Nonlinear Systems : An Introduction*. Lecture Notes in Control and Information Sciences. Mathematical Methods for Robust and Nonlinear Control. Springer Berlin., 2007.
- [LW04] B. Lu and F. Wu. Switching lpv control designs using multiple parameter-dependent lyapunov functions. *Automatica*, 40(11) :1973–1980, 2004.
- [Mau76] H. Maurer. Numerical solution of singular control problems using multiple shooting techniques. *SIAM Journal of Optimization Theory and Applications*, 18 :235–257, 1976.
- [MB00] E. Möllerstedt and B. Bernhardsson. Out of control because of harmonics - an analysis of the harmonic response of an inverter locomotive. *IEEE Control Systems Magazine*, 20(4) :70–81, 2000.
- [MBSL05] K.D. Mombaur, H.G. Bock, J.P. Schlöder, and R.W. Longman. Open-loop stable solutions of periodic optimal control problems in robotics. *ZAMM - Journal of Applied Mathematics and Mechanics*, 85 :499 – 515, 2005.
- [Mel94] A.A. Melikyan. Necessary optimality conditions for a singular surface in the form of synthesis. *Journal of Optimization Theory and Applications*, 82 :203 – 217, 1994.
- [MFT⁺02] T. A. Meynard, H. Foch, P. Thomas, J. Courault, R. Jakob, and M. Nahrstaedt. Multicell converters : basic concepts and industry applications. *IEEE Transactions on Industrial Electronics*, 49(5) :955–964, october 2002.
- [MH07a] B. P. McGrath and D. G. Holmes. Analytical modelling of voltage balance dynamics for a flying capacitor multilevel converter. In *Proceedings of 2007 IEEE Power Electronics Specialists Conference*, pages 1810–1816, Orlando, 2007.
- [MH07b] L. Mei and S. Huihe. Robust predictive control of polytopic uncertain systems with both state and input delays. *Journal of Systems Engineering and Electronics*, 18 :616–621, 2007.
- [MM71] P.J. Moylan and J.B. Moore. Generalizations of singular optimal control theory. *Automatica*, 7(5) :591 – 598, 1971.
- [MP71] J.P. McDanell and W.F. Powers. Necessary conditions for joining optimal singular and nonsingular subarcs. *SIAM Journal of Control and Optimization*, 9 :161–173, 1971.

-
- [MPP07] R. Meziat, D. Patino, and P. Pedregal. An alternative approach for non-linear optimal control problems based on the method of moments. *Comput. Optim. Appl.*, 38(1) :147–171, 2007.
- [MRFA06] D.Q. Mayne, S.V. Rakovic, R. Findeisen, and F. Allgower. Robust output feedback model predictive control of constrained linear systems. *Automatica*, 42(7) :1217–1222, 2006.
- [MVTH90] K. Mahabir, G. Verghese, J. Thottuvelil, and A. Heyman. Linear averaged and sampled data models for large signal control of high power factor converter. In *Proceedings of IEEE Power Electronics Specialists Conference*, pages 372 – 381. San Antonio, Texas, 1990.
- [MW73] R. Middlebrook and G.W. Wester. Low-frequency characterization of switched dc dc converters. *IEEE Transactions on Aerospace and Electronic Systems*, AES9 :376 – 385, 1973.
- [NOaSY93] X. Nicollin, A. Oliveri, and J. Sifakis and S. Yovine. An approach to the description and analysis of hybrid systems. *Hybrid Systems Lecture Notes in Computer Science*, 736 :209 – 229, 1993.
- [PBGM64] L.S. Pontryagin, V.G. Boltyanskii, R.V. Gamkrelidze, and E.F. Mishchenko. *The Mathematical Theory of Optimal Processes*. Pergamon, 1964.
- [PC89] T.S. Parker and L.O. Chua. *Practical Numerical Algorithms for chaotic systems*. Springer, Verlag, 1989.
- [Per01] L. Perko. *Differential Equations and Dynamical Systems (Third Edition)*. Springer, 2001.
- [Pet84] V. Peterka. Predictor-based self-tuning control. *Automatica*, 20(1) :39–50, 1984.
- [PFM02] D. Pinon, M. Fadel, and T. Meynard. Commande par mode de glissement d’un hacheur a deux cellules. *Modélisation et commande des entraînements électriques EPE’99*, 5(1) :147 – 162, 2002.
- [PJS92] H. Peitgen, H. Jügens, and D. Saupe. *Chaos and Fractals : New Frontiers of Science*. Springer-Verlag, 1992.
- [PM98] B.G. Pierrette and C. Maurice. *Mathématiques pour la physique*. Eyrolles, 1998.
- [Pop05] M. Popescu. Control of affine nonlinear systems with nilpotent structure in singular problems. *Journal of Optimization Theory and applications*, 124 :455–466, 2005.
- [Pow80] W.F. Powers. On the order of singular optimal control problems. *Journal of Optimization Theory and Applications*, 32(4) :479–489, 1980.

- [PRI07] D. Patino, P. Riedinger, and C. Iung. Une approche prédictive pour la commande d'un convertisseur multicellulaire. In *2 èmes Journées Nationales du GDR MACS'07*, volume 3, pages 50–55. Lyon, Septembre, 2007.
- [PRI08] D. Patino, P. Riedinger, and C. Iung. Predictive control approach for multicellular converters. In *Proceedings of 34th IEEE Industrial Electronics Society*. Orlando, USA, 2008.
- [PRPK07] M. Pérez, J. Rodriguez, J. Pontt, and S. Kouro. Power distribution in hybrid multi-cell converter with nearest level modulation. In *Proceedings of IEEE International Symposium on Industrial Electronics*, pages 736–741, 2007.
- [RAG99] E. Ronco, T. Arsan, and P.J. Gawthrop. Open-loop intermittent feedback control : practical continuous-time gpc. *IEEE Proceedings on control theory and applications*, 146(5) :426–434, 1999.
- [RB95] D.J.W. Ruxton and D. Bell. Junction times in singular optimal control. *Journal of Applied Mathematics and Computation*, 70 :143–154, 1995.
- [Rei91] E. Reithmeier. Periodic solutions of nonlinear dynamical systems. *Lecture notes in mathematics*, 1483 :171, 1991.
- [RFNM96] A. Rusnak, M. Fikar, K. Najim, and A. Meszaros. Generalized predictive control based on neural networks. *Neural Processing Letters*, 4 :107–112, 1996.
- [RH05] P.B. Reddy and I.A. Hiskens. Limit-induced stable limit cycles in power systems. In *Proceedings of the IEEE Saint Petesburg PowerTech*, volume 27-30, pages 1–5, 2005.
- [Rie00] P. Riedinger. *Thèse de Doctorat : Contribution à la commande optimale des systèmes dynamiques hybrides*. Institut National Polytechnique de Lorraine, Centre de Recherche en Automatique de Nancy, Nancy, France, 2000.
- [RIK03] P. Riedinger, C. Iung, and F. Kratz. An optimal control approach for hybrid systems. *European Journal of Control*, 9(5) :449–458, 2003.
- [RL00] M. Rubensson and B. Lennartson. Stability of limit cycles in hybrid systems using discrete-time lyapunov techniques. In *Proceedings of 39th IEEE Conference on Decision and Control*, volume 2, pages 1397 – 1402, 2000.
- [RMB06] L. Rosier, G. Millerioux, and G. Bloch. Chaos synchronization for a class of discrete dynamical systems on the n-dimensional torus. *Systems and Control Letters*, 3 :223 – 231, 2006.

-
- [RN94] B.J.P. Roset and H. Nijmeijer. Observer-based model predictive control. *International Journal of Control*, 77 :1452–1462, 1994.
- [Rob67] H.M. Robbins. A generalized legendre-clebsch-condition for the singular cases of optimal control. *IBM Journal of Research and Development*, 11(4) :361–372, 1967.
- [Ron06] A. Rondepierre. *Thèse de Doctorat : Algorithmes hybrides pour le contrôle optimal des systèmes non linéaires*. Institut National Polytechnique de Grenoble, 2006.
- [Rou97] T. Roubicek. *Relaxation in Optimization Theory and Variational Calculus*. De Gruyter, 1997.
- [SC64] Sansone and Conti. *Non-linear differential equations*. Pergamon Press, 1964.
- [SC03] M.S. Shaikh and P.E. Caines. On the optimal control of hybrid systems : Analysis and algorithms for trajectory and schedule optimization. In *Proceedings of the IEEE Conference on Decision and Control*, volume 3, pages 2144– 2149, Acapulco, Mexico, 2003.
- [Sey87] R. Seydel. New methods for calculating the stability of periodic solutions. *Computers & mathematics with applications*, 14 :505–510, 1987.
- [Sey88] R. Seydel. *From equilibrium to chaos. Practical bifurcation and stability analysis*. Elsevier, 1988.
- [SJM02] S.N. Simic, K.H. Johansson, and J. Lygeros. Hybrid limit cycles and hybrid poicaré bendixon. In *Proceedings of the 15th IFAC Congress*, volume 15, pages 22–26, 2002.
- [SJM00] S. Simic, K.H. Johansson, S. Sastry, and J. Lygeros. Towards a geometric theory of hybrid systems. *Hybrid systems : Computation and Control*, 1790 :421–436, 2000.
- [SLCYDW01] B. M. Song, J.S. Lai, J. Chang-Yong, and Y. Dong-Wook. A soft-switching high-voltage active power filter with flying capacitors for urban maglev system applications. In *Proceedings of 2007 IEEE Industry Applications Conference*, pages 1461–1468, 2001.
- [SM01] A.V. Savkin and A.S. Matveev. A switched server system of order n with all its trajectories converging to $(n-1)!$ limit cycles. *Automatica*, 2 :303 – 306, 2001.
- [SMQG07] E.I. Silva, B. McGrath, D. Quevedo, and G. Goodwin. Predictive control of a flying capacitor converter. In *Proceeding of IEEE American Control Conference*, volume 1, pages 3763–3768, New York, USA, 2007.

- [SNL⁺91] S. Sanders, J. M. Noworolski, X. Z. Liu, G. C. Verghese Sender, and J. M. Noworolski. Generalized averaging method for power conversion circuits. *IEEE Transactions on Power Electronics*, 6(2) :251–159, April 1991.
- [Son98] E. D. Sontag. *Mathematical control theory*. Mathematical Control Theory. Deterministic Finite Dimensional Systems, 1998.
- [Spa82] C. Sparrow. *The Lorentz Equations*. Springer-Verlag, 1982.
- [Str94] S. H. Strogatz. *Nonlinear dynamics and chaos*. Addison-Wesley, 1994.
- [Sus04] H.J. Sussman. A maximum principle for hybrid optimal control problems. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, volume 1, pages 425 – 430, Maui, Hawaii, 2004.
- [SV85] J. A. Sanders and F. Verhulst. *Averaging Methods in Nonlinear Dynamical Systems*. Springer-Verlag, 1985.
- [Ség99] G. Séguier. *Electronique de puissance*. Dunod, 1999.
- [Tre05] E. Trelat. *Contrôle optimal : théorie et applications*. Collection Mathématiques Concrètes. Vuibert, 2005.
- [Tse00] C. K. Tse. *Complex Behavior of switching power converters*. CRC Press, 200.
- [Utk92] V.I. Utkin. *Sliding modes in control and optimization*. Springer-Verlag, 1992.
- [Vap67] I.B. Vapnyarskii. An existence theorem for optimal control in the boltz problem, some of its applications and necessary conditions for teh optimality of moving and singular systems. *U.S.S.R. Computational Mathematics & Mathematical Physic*, 7 :22–54, 1967.
- [VG97] T.L. Vincent and W.J. Grantham. *Nonlinear and Optimal Control Systems*. Wiley, 1997.
- [Vol96] M. Volker. Singular optimal control - the state of the art. CAS-29, 1996.
- [You69] L. C. Young. *Lectures on Calculus of Variations and Optimal Control Theory*. Saunders, 1969.
- [Zhe07] Z. Zhengxin. On the poincare mapping and periodic solutions of nonautonomous differential systems. *Communications on Pure and Applied Analysis*, 6 :541 – 547, 2007.

Résumé :

Cette thèse s'intéresse au pilotage des cycles limites pour une classe particulière de systèmes hybrides (SDH) : les systèmes commutés cycliques. La thématique des SDH est née du constat d'insuffisance des modèles dynamiques classiques pour décrire les comportements lorsque des aspects événementiels interviennent. Une classe particulièrement importante de SDH est formée par celle qui présente un régime permanent cyclique. Ces systèmes ont des points de fonctionnement non auto-maintenables : il n'existe pas de commande qui maintienne le système sur ce point. Le maintien n'est assuré qu'en valeur moyenne, en effectuant un cycle dans un voisinage du point par commutation des sous systèmes. L'établissement d'une loi de commutation pour cette classe de systèmes doit répondre aux objectifs de stabilité et de performance dynamique, mais doit également garantir la satisfaction de critères liés à la forme d'onde. A l'heure actuelle, peu de méthodes de commande prennent en compte le caractère cyclique du système. Les travaux de cette thèse ont pour objectif de développer des méthodes génériques et robustes pour piloter cette classe de systèmes. Les algorithmes proposés doivent également pouvoir être implémenté en temps réels. On modélise le système comme un système non - linéaire affine en la commande dont la loi de commande apparaît dans le modèle. Ce type de modélisation permet d'envisager deux types de synthèse : l'une à base de commande prédictive et l'autre à base de commande optimale. Ce travail est validé par une partie applicative sur des manipulations dans le CRAN et dans des laboratoires du réseau d'excellence européenne HYCON dans le cadre duquel s'est déroulé cette étude.

Mots-clés : Cycle limite, Systèmes hybrides, fonctions de sensibilités, programmation non-linéaire, commande singulière, retour d'état, réseau de neurones, convertisseur buck, buckboost, multi-niveaux.

Abstract: This work deals with limit cycle control for one particular class of hybrid dynamical systems (HDS) : The cyclic switched systems. The HDS were born because the traditional dynamical models were not able to describe complex behaviors and most of all, behaviors with discontinuities. From an application point of view, one important class of HDS depicts a cyclic behavior in steady state. The main characteristic of these systems is that the operation point cannot be maintained : It does not exist a control that maintains the system on a desired operation point. However, this point can be obtained in average by turning into its neighborhood. Thus, a cycle is produced by switching among the system modes. A switched control law must satisfy stability and dynamic performance. Moreover, criteria related to the waveform must be verified. Nowadays, few methods take into account the cyclic behavior of the system. In this research, some generic methods are studied. They show good performance for controlling the cyclic switched systems. The proposed algorithms can be implemented in real-time. The approaches are based on an affine non-linear model of the system whose control explicitly appears. Two control methods are considered : i) A predictive control, ii) An optimal control. Since the predictive control is a good choice for tracking, it will be able to maintain the system in a cycle. The optimal control yields solutions that can be applied to the transients. Some experiments with both control methods applied to the power converters are shown. These tests were carried out not only in our laboratory (CRAN), but also in other laboratories as part of the HYCON excellence network.

Keywords: Limit cycle, hybrid system, sensibility functions, non-linear programming, singular control, state feedback, neural networks, buck converter, buckboost converter, multilevel converter.

AUTORISATION DE SOUTENANCE DE THESE
DU DOCTORAT DE L'INSTITUT NATIONAL
POLYTECHNIQUE DE LORRAINE

o0o

VU LES RAPPORTS ETABLIS PAR :

Monsieur Alamir MAZEN, Directeur de Recherche, LAG-ENSIEG, Saint Martin d'Hères

Monsieur Jean BUISSON, Professeur, SUPELEC, Cesson-Sevigne

Le Président de l'Institut National Polytechnique de Lorraine, autorise :

Monsieur Diego PATINO

à soutenir devant un jury de l'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE,
une thèse intitulée :

**"Pilotage des cycles limites dans les systèmes dynamiques hybrides : application aux
alimentations électriques statiques"**

NANCY BRABOIS
2, AVENUE DE LA
FORET-DE-HAYE
BOITE POSTALE 3
F - 54 501
VANDOEUVRE CEDEX

en vue de l'obtention du titre de :

DOCTEUR DE L'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE

Spécialité : « **Automatique et Traitement du Signal** »

Fait à Vandoeuvre, le 27 janvier 2009

Le Président de l'I.N.P.L.,

F. LAURENT

