Analyse et contrôle des systèmes dynamiques polynomiaux

Mohamed Amin Ben Sassi

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques Appliquées**

Arrété ministérial :

Présentée par

Mohamed Amin Ben Sassi

Thèse dirigée par **Guillaume James**
et codirigée par **Antoine Girard**

préparée au sein **du Laboratoire Jean Kuntzmann**
et de **l'Ecole Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique**

Analyse et contrôle des systèmes dynamiques polynomiaux

Thèse soutenue publiquement le 15 Avril 2013 ,
devant le jury composé de :

M, Jean-Luc Gouze
DR INRIA, Sophia Antipolis, Rapporteur
M, Didier Henrion
DR CNRS, Université de Toulouse, Rapporteur
M, Anatoli Iouditski
PR, Université de Grenoble, Examinateur
M, Nacim Ramdani
PR, Université d'Orléans, Examinateur
M, Jean-Pierre Raymond
PR, Université de Toulouse, Examinateur
M, Guillaume James
PR, Université de Grenoble, Directeur de thèse
M, Antoine Girard
MdC, Université de Grenoble, Co-Directeur de thèse

Table des matières

Résumé :

Cette thèse présente une étude des systèmes dynamiques polynomiaux motivée à la fois par le grand spectre d'applications de cette classe (modèles de réactions chimiques, modèles de circuits électriques ainsi que les modèles biologiques) et par la difficulté (voire incapacité) de la résolution théorique de tels systèmes.

Dans une première partie préliminaire, nous présentons les polynômes multivariés et nous introduisons les notions de forme polaire d'un polynôme (floraison) et de polynômes de Bernstein qui seront d'un grand intérêt par la suite.

Dans une deuxième partie, nous considérons le problème d'optimisation polynomial dit POP. Nous décrivons dans un premier temps les principales méthodes existantes permettant de résoudre ou d'approcher la solution d'un tel problème. Puis, nous présentons deux relaxations linéaires se basant respectivement sur le principe de floraison ainsi que les polynômes de Bernstein permettant d'approcher la valeur optimale du POP.

La dernière partie de la thèse sera consacré aux applications de nos deux méthodes de relaxation dans le cadre des systèmes dynamiques polynomiaux. Une première application s'inscrit dans le cadre de l'analyse d'atteignabilité : en effet, on utilisera notre relaxation de Bernstein pour pouvoir construire un algorithme permettant d'approximer les ensembles atteignables d'un système dynamique polynomial discrétisé. Une deuxième application sera la vérification et le calcul d'invariants pour un système dynamique polynomial. Une troisième application consiste à calculer un contrôleur et un invariant pour un système dynamique polynomial soumis à des perturbations. Dans le contexte de l'invariance, on utilisera la relaxation se basant sur le principe de floraison. Enfin, une dernière application sera d'exploiter les principales propriétés de la forme polaire pour pouvoir étudier des systèmes dynamiques polynomiaux dans des rectangles.

Abstract :

This thesis presents a study of polynomial dynamical systems motivated by both the wide spectrum of applications of this class (chemical reaction models, electrical models and biological models) and the difficulty (or inability) of theoretical resolution of such systems.

In a first preliminary part, we present multivariate polynomials and we introduce the notion of polar form of a polynomial (blossoming) and Bernstein polynomials which will be of great interest thereafter.

In a second part, we consider the polynomial optimization problem said POP. We first describe existing methods allowing us to solve or approximate the solution

of such problems. Then, we present two linear relaxations based respectively on the blossoming principle and the Bernstein polynomials allowing us to approximate the optimal value of the POP.

The last part of the thesis is devoted to applications of the two relaxation methods in the context of polynomial dynamical systems. A first application is in the context of reachability analysis. In fact, we use our Bernstein relaxation in order to build an algorithm allowing us to approximate the reachable sets of a discretized polynomial dynamical system. A second application deals with the verification and the computation of invariants for polynomial dynamical systems. A third application consists in calculating a controller and an invariant for a polynomial dynamical system subject to disturbances. For the invariance problem, we use the relaxation based on the blossoming principle. Finally, the last application consists in exploiting the main properties of the polar form in order to study polynomial dynamical systems in rectangles.

Chapitre 1

Introduction

1.1 Motivations

Dans cette thèse, nous allons nous intéresser à l'étude de systèmes dynamiques polynomiaux, c'est à dire des équations différentielles dont le champ de vecteurs est donné par des polynômes. Les motivations de cette étude sont diverses.

D'abord, ce type de systèmes intervient dans de nombreuses applications comme une variété de modèles de réactions chimiques [76, 117], des modèles de circuits électriques non linéaires [67, 190], les modèles proies-prédateurs [112, 185] ainsi que plusieurs modèles provenant de la biologie [58].

Ensuite, contrairement aux systèmes linéaires, les systèmes polynomiaux peuvent produire une représentation significative de dynamiques compliquées comme les mouvements chaotiques.

Enfin, la plus grande motivation de cette étude est le fait que la résolution analytique des systèmes dynamiques polynomiaux est dans la plupart des cas trop compliquée voire impossible. En effet, pour comprendre les dynamiques de ces systèmes, on utilise souvent des simulations numériques qui nous permettent juste d'approcher les trajectoires. Comme chaque simulation correspond à une trajectoire (pour une condition initiale fixée) il nous faut à priori un grand nombre de simulations voire une infinité pour bien pouvoir étudier ces systèmes.

Cependant, on peut éviter de simuler toutes les trajectoires du système en l'étudiant d'une manière qualitative pour en dégager les principales propriétés (stabilité, périodicité,..). On peut aussi s'intéresser à l'étude des propriétés d'atteignabilité et d'invariance qui représentent des outils importants surtout dans le domaine de vérification des systèmes dynamiques. En effet, dans ce cadre on n'a pas besoin d'étudier le comportement asymptotique d'un modèle (un système dynamique) mais plutôt de vérifier que ce modèle est bien approprié pour certaines propriétés. Par exemple, une des propriétés intéressantes à vérifier est la propriété de sûreté. Cette propriété consiste à vérifier que toutes les trajectoires du système n'entreront jamais dans un ensemble donné dit ensemble d'états interdits. Pour vérifier une telle propriété, connaître l'ensemble atteignable du système (ensemble que les trajectoires du système peuvent atteindre à partir d'un ensemble d'états initiaux donné) ou une sur-approximation de cet ensemble sera utile : c'est le contexte de l'analyse d'atteignabilité. D'autre part, connaître une région invariante (région dans laquelle si une

trajectoire du système entre alors elle restera toujours) peut nous aider à vérifier la sûreté d'un système. En effet, si cette région n'intersecte pas l'ensemble d'états interdits, il suffit de prendre l'ensemble des états initiaux à l'intérieur de cette région pour que notre propriété de sûreté soit vérifiée sans avoir à réaliser un nombre infini de simulations.

Dans cette thèse, le problème d'atteignabilité ainsi que le problème d'invariance pour les systèmes dynamiques polynomiaux seront considérés. Le problème de synthèse de contrôleurs assurant l'invariance pour des systèmes dynamiques polynomiaux avec perturbations sera aussi traité. On verra que la résolution de ces problèmes repose essentiellement sur la résolution des problèmes d'optimisation polynomiaux. Par conséquent, une grande partie sera dédiée à l'optimisation polynomiale. Le cas particulier important des systèmes dynamiques dont le champ de vecteurs est multi-affine est traité à part. En effet, on proposera une approche permettant d'étudier des systèmes dynamiques polynomiaux dans des domaines rectangulaires en se ramenant au cas multi-affine, ce qui nous permettra de résoudre un problème de planification des trajectoires.

1.2 Etat de l'art

Pour en arriver à l'étude des systèmes dynamiques polynomiaux et plus généralement non linéaires, le cas linéaire a été intensivement étudié.

Pour une équation différentielle linéaire

$$\dot{x}(t) = Ax(t), \quad x \in \mathbb{R}^n \tag{1.1}$$

où $A \in \mathbb{R}^{n\times n}$, le comportement des solutions est contrôlé par le signe des parties réelles des valeurs propres de la matrice A donc la vérification de certaines propriétés (stabilité, périodicité, ...) se ramène à un problème spectral.

Considérons maintenant le système contrôlé [1] suivant :

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x \in \mathbb{R}^n \tag{1.2}$$

où $B \in \mathbb{R}^{n\times p}$ et $u(t) \in \mathbb{R}^p$ est le contrôleur que l'on veut calculer.
La plupart des temps, on cherche un contrôleur dépendant de la variable d'état ($u = u(x)$) dans une région bien précise U (souvent bornée). Dans ce cas il faut aussi vérifier que pour tout $x \in X$ ($X \subseteq \mathbb{R}^n$ désigne l'ensemble des états admissibles) $u = u(x)$ est bien inclus dans U : c'est la condition d'admissibilité.
Dans la littérature, les premiers travaux ont été dédiés à la synthèse d'un contrôleur linéaire ($u(x) = Kx$ où $K \in \mathbb{R}^{p\times n}$) et toute une théorie a été développé (voir [188, 189]). De nombreux résultats permettant de résoudre des problèmes de découplage [123, 148, 62], contrôlabilité et observabilité [95, 77], calcul d'ensembles admissibles et stabilité [80, 63, 82, 181] ont été obtenus. Ayant comme objectif de trouver une matrice K admissible permettant de forcer le système à avoir un certain comportement, ces approches donnent généralement un ensemble de conditions sur

1. Tout le long de cette section, les systèmes dynamiques seront donnés par convention en temps continu.

cette matrice (par exemples des conditions sur les valeurs propres) qui dépendent des données du système. Dans le cas où les ensembles sont polyédraux, des conditions aux sommets permettant le calcul d'une matrice admissible K ont aussi été obtenues.
Pour le problème de stabilisation d'un système dynamique (trouver un contrôleur qui le rend stable), l'approche classique est celle de Lyapunov [113]. On parle parfois de système Lyapunov-stable. En effet, démontrer la stabilité revient à trouver une fonction décroissante tout au long des trajectoires dite fonction de Lyapunov. Ainsi, la stabilisation d'un système dynamique consiste à trouver à la fois une fonction de Lyapunov ainsi qu'un contrôleur associé. On peut voir par exemple [128, 81] pour le calcul d'un contrôleur linéaire en utilisant une fonction de Lyapunov quadratique.

Une situation plus complexe est obtenue lorsque (1.2) est perturbé par un terme source :

$$\dot{x}(t) = Ax(t) + Bu(t) + Ed(t), \quad x \in \mathbb{R}^n \tag{1.3}$$

où $E \in \mathbb{R}^{n\times q}$ et $d(t) \in \mathbb{R}^q$.
Pour la stabilisation de (1.3), l'approche de Lyapunov est utilisée dans [167, 118] tandis que Blanchini [25, 27] propose une approche se basant sur la programmation linéaire dans le cas d'ensembles polyédraux.
Cependant, la connaissance imparfaite de la dynamique exacte rend plus approprié de l'étudier avec une certaine incertitude c'est à dire prendre $A = A(q(t))$ et $B = B(q(t))$ où $q(t)$ est la fonction représentant l'incertitude. Notons que le système reste toujours linéaire en x mais son étude devient plus compliquée. Dans [108, 40, 11, 9, 10, 193] des fonctions de Lyapunov quadratiques ont été utilisées pour la résolution du problème de contrôle. Dans [28, 29], Blanchini résout le problème de stabilisation sans fixer une forme particulière pour la fonction de Lyapunov (souvent quadratique) ni pour le contrôleur (souvent linéaire).
Notons enfin que pour le cas linéaire, la notion d'invariance [64, 63, 23, 24] joue un rôle crucial. Dans [181, 18, 17, 25, 26] on voit bien que cette notion est fondamentale pour la résolution du problème de stabilité. En effet, ce concept représente un outil permettant d'assurer à la fois les conditions d'admissibilité ainsi que la stabilité du système dynamique.

Lorsque $f : \mathbb{R}^n \to \mathbb{R}^n$ est une application non linéaire, l'étude du système

$$\dot{x}(t) = f(x(t)), \quad x \in \mathbb{R}^n \tag{1.4}$$

est généralement plus complexe.

En présence d'un contrôleur $u(t)$ et de perturbation $d(t)$, le système précédent aura la forme :

$$\dot{x}(t) = f(x(t), u(t), d(t)), \quad x \in \mathbb{R}^n. \tag{1.5}$$

Dans la plupart des cas, il n'est pas possible de résoudre le système (1.4) analytiquement et on cherche comme précédemment à faire une étude qualitative pour pouvoir comprendre le comportement des trajectoires du système.
Pour un champ de vecteurs multi-affines (c'est à dire affines pour chaque composante) plusieurs résultats ont été obtenus. Dans [97], le problème d'atteignabilité associé à un système multi-affine a été étudié tandis que le problème de synthèse

de contrôleur assurant l'invariance et la sortie en temps fini a été résolu dans [13] pour les domaines rectangulaires (des boîtes). Le problème de boîte invariante a été généralisé pour une classe de fonctions dites quasi-multi-affines [1].
Pour la classe de champs de vecteurs polynomiaux de nombreux travaux ayant comme objectif la vérification de la stabilité ont été réalisés. Une grande partie de ces travaux proposent des moyens pour le calcul d'un ensemble invariant pour le domaine d'attraction. Dans [55, 142, 179, 39, 88] des fonctions de Lyapunov dont les courbes de niveau donnent l'ensemble invariant sont calculées. Le problème de synthèse d'un contrôleur est aussi étudié et sa résolution efficace représente un défi important. Dans [91] un algorithme itératif se basant sur l'approche de Lyapunov a été proposé pour le calcul d'un contrôleur qui stabilise le système. Cependant, ce contrôleur n'est pas globalement optimal et parfois l'algorithme ne réussit pas à le trouver. D'autre part, étant donné la difficulté d'assurer les conditions de stabilité données par l'approche de Lyapunov pour le problème de synthèse d'un contrôleur, Rantzer [145] présente un nouveau critère de convergence se basant sur une fonction de densité ayant de meilleures propriétés de convexité que celle de Lyapunov. Prajna [140] a exploité ce critère pour résoudre le problème de synthèse d'un contrôleur et une application de ce travail dans le cas du contrôle d'un avion hypersonique a été réalisée dans [6].
D'autre part, des méthodes plus directes profitant du progrès réalisé ces dernières années dans la programmation polynomiale [101, 132] et notamment de l'introduction des relaxations SOS [131] (somme de carrés) sont maintenant disponibles [186, 68, 173, 170, 171, 176]. On peut citer notamment les travaux de Henrion [45, 44, 115, 116, 47, 114, 48, 46] utilisant essentiellement les relaxations LMI (inégalités matricielles linéaires) introduite par Lasserre [101].
Ainsi, pour conclure, on a toujours besoin d'un outil permettant la résolution des problèmes d'optimisation polynomiaux pour pouvoir étudier les systèmes dynamiques polynomiaux (vérification de certaines propriétés ou synthèse d'un contrôleur pour les assurer). Ceci fut l'intérêt d'une grande partie de cette thèse.

1.3 Plan et contributions

1.3.1 Motivations

Problème d'atteignabilité :

Un des problèmes abordés dans cette thèse est l'analyse d'atteignabilité dans le cadre d'un système dynamique discret :

$$x_{k+1} = f(x_k),\ k \in \mathbb{N},\ x_k \in \mathbb{R}^n,\ x_0 \in X_0$$

où f est un champ de vecteurs polynomial et X_0 désigne un polytope borné de $\mathbb{R}^n$.

L'analyse d'atteignabilité de ce système en temps borné consiste à calculer la suite $X_k \subseteq \mathbb{R}^n$ des ensembles atteignables à l'instant k du système jusqu'à un certain temps $K \in \mathbb{N}$. En effet, il est facile de vérifier que cette suite vérifie la relation de récurrence suivante $X_{k+1} = f(X_k)$. Cependant, la difficulté provient du fait que même si X_0 est un polytope, les autres éléments de la suite ne le sont pas forcément. En réalité, ils ne sont généralement même pas convexes.

Pour résoudre ce problème, on va sur-approximer ces ensembles en utilisant des polytopes bornés $\overline{X}_k = Poly(A_k, b_k)$ [2] qui peuvent clairement être calculées par induction en posant $\overline{X}_0 = X_0$ et en assurant la relation $f(\overline{X}_k) \subseteq \overline{X}_{k+1}$ pour tout $k = 0, \ldots, K-1$.

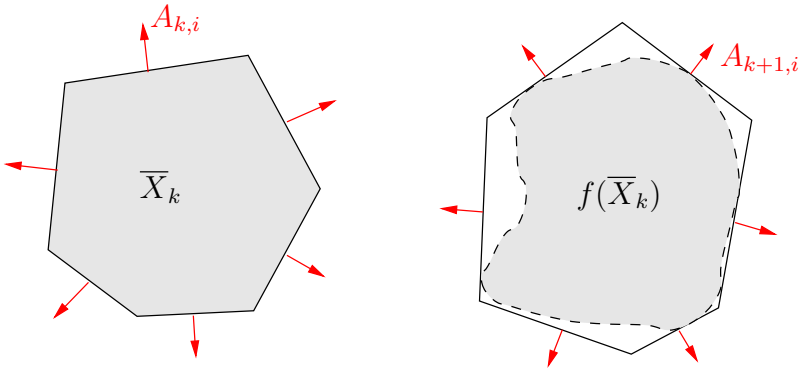

FIGURE 1.1 – Sur-approximation polyédrale

Ainsi, si le polytope $\overline{X}_k$ et la matrice de direction A_{k+1} sont donnés alors le nouveau vecteur de direction b_{k+1} associé à $\overline{X}_{k+1}$ doit vérifier l'inégalité suivante :

$$-b_{k+1,i} \leq \min_{x \in \overline{X}_k} -A_{k+1,i} \cdot f(x) \text{ pour tout } i = 1, \ldots, m,$$

où m désigne le nombre de faces du polytope $\overline{X}_{k+1}$ (voir Figure 1.1).
Par conséquent, on va devoir trouver une borne minorante associée au problème d'optimisation polynomial $\min_{x \in \overline{X}_k} -A_{k+1,i} \cdot f(x)$.

Problème d'invariance polyédral :

Un autre problème que l'on veut étudier dans cette thèse est le problème d'invariance. En effet, considérons le système dynamique suivant :

$$\dot{x}(t) = f(x(t))$$

où f est un champ de vecteurs polynomial.

Etant donné un polytope P, on se propose de vérifier si P est invariant sous l'action du champ de vecteurs f c'est à dire si toutes les trajectoires commençant à l'intérieur de P restent dans P. Pour vérifier cette propriété, un moyen sera d'étudier l'orientation du champ de vecteurs sur la frontière du polytope. Plus précisément, sur les faces du polytope il faut que le champ de vecteurs pointe vers l'intérieur du domaine.

Pour le voir, on considère comme exemple un polytope P ayant 5 faces (voir Figure 1.2) et pour toute face F_i on désigne par n_i sa normale sortante et par b_i sa position. Par conséquent, une condition nécessaire et suffisante pour l'invariance

2. $Poly(A_k, b_k)$ désigne le polytope dont les directions sont données par la matrice A_k et les positions des faces par le vecteur b_k.

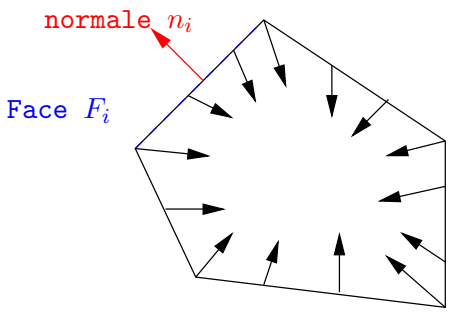

FIGURE 1.2 – Polytope invariant

de P est la positivité pour tout $i = 1, \ldots, 5$ des valeurs optimales p_i associées aux problèmes d'optimisation suivants :

$$\begin{array}{rl} \text{minimiser} & -n_i \cdot f(x) \\ \text{s.c} & n_i \cdot x = b_i, \\ & n_j \cdot x \leq b_j, \quad j \in \{1, \ldots, 5\} \setminus \{i\}. \end{array}$$

Ainsi, pour pouvoir réaliser notre étude sur les systèmes dynamiques polynomiaux on a besoin de traiter des problèmes d'optimisation polynomiale que l'on notera POP. Cette thèse présente donc deux types de contributions, la première concernant la résolution d'un POP et la deuxième concernant l'analyse et le contrôle des systèmes dynamiques polynomiaux.

1.3.2 Problème d'optimisation polynomiale

On s'intéresse au problème de minimisation d'une fonction objectif polynomiale sur un ensemble de contraintes données aussi par des polynômes.

Présentation du problème

La formulation générale du problème est la suivante :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in \mathbb{R}^n \\ & g_i(x) \leq 0 \ , \ i = 1, \ldots, m \\ & h_i(x) = 0 \ , \ j = 1, \ldots, s \end{array} \tag{1.6}$$

où $p, g_1, \ldots, g_m, h_1, \ldots, h_s$ sont tous des polynômes multi-variés.

C'est un problème important dont la résolution efficace demeure jusqu'à nos jours un grand défi. Notre objectif est de pouvoir trouver une borne inférieure "suffisamment" proche de la valeur optimale de (1.6).

Aperçu de l'état de l'art

Dans la littérature, plusieurs travaux ont été élaborés pour pouvoir étudier et résoudre le problème (1.6). Le Chapitre 4 sera dédié aux méthodes existantes classées en deux catégories.

La première, dite algébrique, permet généralement de calculer une solution exacte. Cependant, l'inconvénient de ces méthodes est le coût élevé des algorithmes qu'elles utilisent puisqu'elles calculent explicitement tous les minima locaux pour en déduire le minimum global du problème d'optimisation à résoudre.

La deuxième catégorie englobe les méthodes de relaxation. L'idée principale est de ramener la résolution d'un problème polynomial (difficile à résoudre) à la résolution d'une relaxation de plus grande taille (des variables et des contraintes supplémentaires) qui soit plus facile à résoudre. Ainsi, les algorithmes fournis à l'aide de ces relaxations sont beaucoup moins coûteux que ceux des méthodes exactes. Cependant, ils nous donnent généralement des bornes inférieures à la valeur optimale que l'on cherche. Plus précisément, on va distinguer deux types de relaxation.
La première, surnommée relaxation RLT, se base sur la programmation linéaire. Par conséquent, des algorithmes efficaces pour la résolution existent mais en contre partie les bornes fournies à l'aide de cette relaxation ne sont pas trop précises. La deuxième, surnommée relaxation LMI, est beaucoup plus précise. Cependant, comme elle se base sur la programmation semi-définie, les algorithmes utilisés coûtent plus chers.

Relaxations linéaires proposées

Dans le chapitre 5, deux nouvelles relaxations linéaires pour le problème d'optimisation polynomiale sont présentées. Elles se basent respectivement sur le principe de floraison et sur les polynômes de Bernstein.
L'intérêt principal de nos relaxations est mis en oeuvre grâce à l'exemple 10 du chapitre 5. En effet, à l'aide de cet exemple, on peut voir que la taille des relaxations obtenus en utilisant le principe de floraison où les polynômes de Bernstein sont nettement meilleures que les autres tout en ayant une précision raisonnable.
En d'autres termes, nous allons pouvoir profiter de l'efficacité de la programmation linéaire tout en ayant un nombre raisonnable de variables et de contraintes.

Dans ce qui suit, nous allons formuler ces relaxations dans le cas où les contraintes sont affines (l'ensemble admissible est polyédral).

1) Relaxation se basant sur le principe de floraison :

Dans le contexte de l'invariance (voir Chapitre 7), on se propose de trouver une borne inférieure à la valeur optimale p^* du problème :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in R, \\ & a_i \cdot x \leq b_i, \quad i = 1, \ldots, m, \\ & c_j \cdot x = d_j, \quad j = 1, \ldots, r. \end{array} \tag{1.7}$$

où R désigne un rectangle de $\mathbb{R}^n$.
Pour cela, nous allons utiliser le principe de floraison (introduit dans le Chapitre 2) d'un polynôme multi-varié. Un polynôme multi-varié de degré $\delta = (\delta_1, \ldots, \delta_n)$ est une fonction $p : \mathbb{R}^n \to \mathbb{R}$ s'écrivant sous la forme :

$$p(x) = p(x_1, \ldots, x_n) = \sum_{(k_1,\ldots,k_n)\in\Delta} p_{k_1,\ldots,k_n} x_1^{k_1} \ldots x_n^{k_n},$$

où $\Delta = \{0, \ldots, \delta_1\} \times \cdots \times \{0, \ldots, \delta_n\}$ et $\{p_{k_1,\ldots,k_n} \in \mathbb{R}\ ,\ (k_1, \ldots, k_n) \in \Delta\}$ désigne l'ensemble de ses coefficients.

A partir d'un tel polynôme, le principe de floraison nous permet de construire un autre polynôme $q_\delta : \mathbb{R}^{\delta_1+\cdots+\delta_n} \to \mathbb{R}$ (forme polaire de p) multi-affine (les nouveaux degrés en chacune des variables sont au plus égaux à 1), symétrique par rapport à ses arguments et tel que $q_\delta(x_1, \ldots, x_1, \ldots, x_n, \ldots, x_n) = p(x_1, \ldots, x_n)$ (voir Définition 4).

Maintenant, on se fixe un rectangle R de $\mathbb{R}^n$ dont les bornes sont $\underline{x} = (\underline{x_1}, \ldots, \underline{x_n})$, $\overline{x} = (\overline{x_1}, \ldots, \overline{x_n})$ et dont l'ensemble de sommets est V. On lui associe ensuite un nouveau rectangle R' de $\mathbb{R}^{\delta_1+\cdots+\delta_n}$, son ensemble de sommets V' ainsi qu'une relation d'équivalence $\cong$ entre les sommets du rectangle R' lié à la symétrie de q_δ (voir Section 3.3.1).

En notant $\overline{V}' = (V'/\cong)$, on obtient le Théorème suivant :

Théorème 6. La valeur optimale $p_\delta{}^*$ du programme linéaire :

$$\begin{array}{lll} \max & t & \\ \text{s.c} & t \in \mathbb{R},\ \lambda \in \mathbb{R}^m,\ \mu \in \mathbb{R}^r, & \\ & \lambda_i \geq 0, & i = 1, \ldots, m, \\ & t \leq q_\delta(\overline{v}) + \displaystyle\sum_{i=1}^{m} \lambda_i (a_i' \cdot \overline{v} - b_i) & \\ & \quad + \displaystyle\sum_{j=1}^{r} \mu_j (c_j' \cdot \overline{v} - d_j), & \overline{v} \in \overline{V}' \end{array} \tag{1.8}$$

est une borne inférieure à p^*.

2) Relaxation se basant sur les polynômes de Bernstein :

Dans le contexte de l'analyse d'atteignabilité (Chapitre 6), on va considérer une variante du problème (1.7) :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in R = [0,1]^n, \\ & Ax \leq b, \end{array} \tag{1.9}$$

où $A \in \mathbb{R}^{m \times n}$ et $b \in \mathbb{R}^m$.

Pour obtenir une borne inférieure à l'optimum de ce problème, nous allons utiliser les polynômes de Bernstein introduits au Chapitre 3. En effet, dans la base de Bernstein, le polynôme p de degré δ s'écrit sous la forme :

$$p(x) = \sum_{I \leq \delta} b_I B_{I,\delta}(x)$$

où les polynômes $(B_{I,\delta}(x))_{I \leq \delta}$ formant la base de Bernstein sont définis à l'aide de (3.5) et les coefficients $(b_I)_{I \leq \delta}$ (qu'on les notera aussi $(\hat{p}_{I,\delta})_{I \leq \delta}$) sont calculés en utilisant (3.6).

Une borne inférieure associée au problème (1.9) est obtenue en résolvant la relaxation linéaire donnée par la Proposition suivante :

Théorème 8. Soit $p_\delta{}^*$ la valeur optimale du programme linéaire suivant :

$$\begin{array}{lll} \text{minimiser} & \sum_{I\leq\delta} \hat{p}_{I,\delta} z_I & \\ \text{s.c} & z_I \in \mathbb{R}, & I \leq \delta, \\ & 0 \leq z_I \leq B_{\delta,I}(\frac{I}{\delta}), & I \leq \delta, \\ & \sum_{I\leq\delta} z_I = 1, & \\ & \sum_{I\leq\delta} (A\frac{I}{\delta}) z_I \leq b. & \end{array} \tag{1.10}$$

Alors, $p_\delta{}^* \leq p^*$ où p^* est la valeur optimale du problème (1.9).

Dans le Chapitre 5, les deux dernières relaxations sont généralisées dans le cadre du problème (1.6), c'est à dire le cadre d'un ensemble de contraintes polynomiales (ensemble semi-algébrique). Cependant, pour les applications considérés dans cette thèse, nous utiliserons uniquement les relaxations précédemment définies puisque les ensembles que l'on étudiera seront des polytopes.

Résultat de convergence :

Pour améliorer la précision de notre relaxation, une possibilité sera d'augmenter le degré $K = \delta$ de la relaxation (ce qui revient à considérer les polynômes $p, g_1, \ldots, g_m$ ayant un degré maximal δ comme appartenant à un ensemble de polynômes de degré plus grand $K \geq \delta$).
Dans le cas des relaxations précédentes le résultat de convergence est le suivant :

Théorème 9. Pour tout $K \geq \delta$,

$$0 \leq p_C^* - p_K^* \leq O(\tfrac{1}{k_1} + \cdots + \tfrac{1}{k_n}).$$

où p_C^* est la valeur optimale du problème convexe suivant :

$$\begin{array}{ll} \text{minimiser} & C(p)(x) \\ \text{s.c} & x \in R, \\ & Ax \leq b, \end{array}$$

et $C(p)(x)$ désigne l'enveloppe convexe de p sur le rectangle R.

1.3.3 Analyse et contrôle des systèmes dynamiques polynomiaux

Notons par $f = (f_1, \ldots, f_n) : \mathbb{R}^n \to \mathbb{R}^n$ un champ de vecteurs polynomial et par R un rectangle de $\mathbb{R}^n$. Pour la partie analyse des systèmes dynamiques polynomiaux, les principales contributions sont :

1) Analyse d'atteignabilité

Dans le Chapitre 6, on utilisera notre relaxation de Bernstein dans le cadre de l'analyse d'atteignabilité. En effet, on va proposer un algorithme permettant de sur-approximer les ensembles atteignables d'un système dynamique polynomial discrétisé.

Aperçu de l'état de l'art

Des méthodes d'analyse par intervalles [92] peuvent être utilisées pour calculer l'ensemble atteignable. Cependant, ces méthodes sont généralement utilisés quand il s'agit de domaines rectangulaires et pas dans le cas des domaines polyédraux.

L'approche la plus similaire à la notre à été présentée par Dang [49, 53]. En effet, dans ces travaux, l'idée de calculer des relaxations linéaires pour trouver des sur-approximations est utilisée. Cependant, les relaxations proposées consistent à trouver grâce à la forme de Bernstein une fonction affine permettant de minorer notre fonction polynomiale tandis que dans notre approche la fonction minorante est une fonction affine par morceaux beaucoup plus précise. De plus, cette approche ne traite aussi que des domaines rectangulaires.

Méthode :

Pour résoudre ce problème, on va sur-approximer ces ensembles en utilisant des polytopes bornés $\overline{X}_k$. En effet, ces sur-approximations peuvent clairement être calculées par induction en posant $\overline{X}_0 = X_0$ et en assurant la relation $f(\overline{X}_k) \subseteq \overline{X}_{k+1}$ pour tout $k = 0, \ldots, K-1$.

Le résultat principal décrivant notre méthode est donné à l'aide de l'algorithme d'atteignabilité 6.1.3. Cet algorithme permet essentiellement de ramener le calcul de la sur-approximation $\overline{X}_{k+1}$ à la résolution d'un problème de la forme (1.9). Par conséquent, on pourra s'en servir de la relaxation (1.10) pour la calculer.

Notons aussi que dans les problèmes d'optimisations que l'on se propose de résoudre, la matrice de direction (la matrice A_k) est fixée à l'avance et le calcul se restreint à trouver les positions des faces (le vecteur b_k). Par ailleurs, une autre contribution est la façon dont ces "modèles polyédraux" seront choisis. En effet, on propose deux sortes de modèles : modèles statiques ne prenant pas en considération l'évolution du système au cours du temps (algorithmes pas chers mais précision restreinte) et modèles dynamiques (algorithmes plus précis mais plus chers). Tout cela est mis en oeuvre dans la Section 6.2.1.

2) Vérification et calcul d'invariant

Le Chapitre 7 illustre une deuxième application concernant la vérification et le calcul d'invariants pour un système dynamique polynomial dans le cas continu. Dans ce contexte, on utilisera la relaxation se basant sur le principe de floraison.

Problème :

On va considérer le système dynamique (7.1) :

$$\dot{x}(t) = f(x(t)),\ x(t) \in \mathbb{R}^n.$$

Un premier problème que l'on se propose de résoudre est de vérifier si un polytope donné P est invariant pour ce système : c'est à dire si une trajectoire du système appartient à P à l'instant t_0 alors elle restera dans P pour tout $t \geq t_0$.
Un deuxième problème est le calcul à partir d'un polytope P (qui a priori n'est pas invariant), d'un polytope P' ayant les mêmes directions que P et qui soit invariant.

Aperçu de l'état de l'art

Le calcul des invariants pour des systèmes dynamiques polynomiaux est souvent abordée à l'aide de la programmation semi-définie (SDP) par l'intermédiaire des relaxations SOS (voir Section 4.3.2). Une grande partie de la littérature dans ce sujet consiste à calculer les fonctions de Lyapunov dont les courbes de niveau sont les invariants (voir [186, 175]).

Dans [13, 1], on traite le problème de la boîte invariante pour des dynamiques multi-affines. Cependant, notre méthode permet de trouver des polytopes invariants pour des systèmes dynamiques polynomiaux.

Méthode :

On va montrer que la vérification d'invariance d'un polytope P peut se formuler (voir Section 7.2.1) en un problème de vérification de positivité des valeurs optimales de problèmes d'optimisation polynomial de la forme (1.7). Ainsi, une condition suffisante assurant l'invariance de P est la positivité des valeurs optimales des relaxations associées ayant la forme (1.8).

Pour la résolution du second problème, notre principal ingrédient est un résultat dérivant de l'analyse de sensibilité (voir Section 7.1.3). Plus précisément, nous considérons le problème suivant (une variation du problème (1.7)) :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in R, \\ & a_i \cdot x \leq b_i + \alpha_i, \quad i \in I, \\ & c_j \cdot x = d_j + \beta_j, \quad j \in J, \end{array} \tag{1.11}$$

où $\alpha_i \in \mathbb{R}$, pour tout $i \in I$ et $\beta_j \in \mathbb{R}$ pour tout $j \in J$.
On note par $p^*(\alpha, \beta)$ sa valeur optimale et par ${p_\delta}^*(\alpha, \beta)$ la valeur optimale de de la relaxation (1.10) associée. On a le théorème suivant :

Théorème 11. Pour tous $\alpha \in \mathbb{R}^{m_I}$ et $\beta \in \mathbb{R}^{m_J}$, tels que (1.11) est admissible, on a :

$$p^*(\alpha, \beta) \geq {p_\delta}^*(\alpha, \beta) \geq {p_\delta}^*(0, 0) - \lambda^* \cdot \alpha - \mu^* \cdot \beta.$$

C'est en utilisant ce théorème qu'on va construire à partir de notre polytope initial P, un nouveau polytope P' ayant les mêmes directions de P (voir Figure 1.3) et qui soit invariant. En effet, le fait de modifier juste les positions des faces du polytope

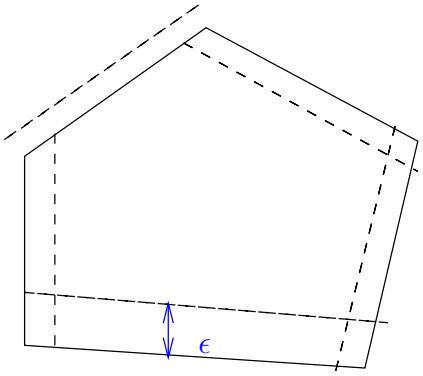

FIGURE 1.3 – Nouveau polytope (en pointillé) obtenu après perturbation du polytope initial (en continu)

P revient à perturber un problème de la forme (1.7) pour obtenir un problème de la forme (1.11). Ainsi, en utilisant l'analyse de sensibilité, on cherche à trouver une perturbation (la plus petite possible) qui nous permet de garantir la positivité de la valeur optimale de la relaxation associée au problème perturbé (voir Section 7.2.2).

Maintenant, nous allons établir les contributions dans le cadre du contrôle des systèmes dynamiques polynomiaux :

3) Synthèse d'un contrôleur assurant l'invariance

Une extension du Chapitre 7 est proposée par l'intermédiaire du Chapitre 8. En effet, dans ce chapitre on se propose de calculer un contrôleur et un invariant pour un système dynamique polynomial sous l'action de perturbations.

Problème :

On va considérer un système de contrôle soumis à des contraintes d'entrées et des perturbations bornées (8.1) :

$$\dot{x}(t) = f(x(t), d(t)) + g(x(t), d(t))u(t),\ d(t) \in D\ , u(t) \in U$$

où $d(t) \in D \subseteq \mathbb{R}^m$ est une perturbation extérieure et $u(t) \in U \subseteq \mathbb{R}^p$ désigne le contrôleur. La matrice de contrôle $g : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{(n\times p)}$, est défini à l'aide de polynômes multi-variés.

Soit $\underline{P} \subseteq \overline{P} \subseteq R$ deux polytopes compacts et convexes. On va considérer le problème de synthèse d'un contrôleur u pour le système (8.1) tel que toutes les trajectoires contrôlées initialisées à l'intérieur de $\underline{P}$ restent dans $\overline{P}$. Ceci peut être vu comme étant une propriété de sûreté où $\underline{P}$ est l'ensemble des états initiaux et $\overline{P}$ est l'ensemble des états sûres.

Aperçu de l'état de l'art

L'intérêt de résoudre le problème précédent est d'établir une propriété de stabilité du système en trouvant un contrôleur assurant l'invariance d'une certaine région. Dans la littérature, la plupart des approches pour la synthèse d'un tel contrôleur et d'une telle région sont basées sur la théorie de Lyapunov ainsi que les techniques de relaxations SOS. Le lecteur peut voir [30, 91] pour une revue de ces approches.

Méthode :

Le problème peut être résolu en synthétisant conjointement un contrôleur et un invariant polyédral $P \subseteq R$ contenant $\underline{P}$ et inclus dans $\overline{P}$. Pour cela, on propose une approche itérative décrite dans la Section 8.3.2 comportant deux principales étapes.

Dans la première on va utiliser le résultat principal du Chapitre 8 donné à l'aide de la Proposition 19. Ce résultat permet, étant donné un polytope P et une forme du contrôleur fixé au préalable, de trouver les coefficients d'un candidat à l'invariance (c'est à dire un contrôleur $u(x) \in U$ pour tout $x \in R$). Si ce contrôleur assure l'invariance du polytope P pour le système (8.1) on a résolu notre problème. Sinon on passe à la deuxième étape consistant à utiliser l'analyse de sensibilité pour modifier P (tout en restant compris entre $\underline{P}$ et $\overline{P}$). Dans le cas où notre contrôleur u assure l'invariance du nouveau polytope P' le problème est résolu sinon on reprend la première étape en remplaçant P par P'.

4) Synthèse de contrôleur dans des rectangles

Une dernière application décrite dans le Chapitre 9 traite le cas particulier de systèmes dynamiques polynomiaux dans des domaines rectangulaires. Les propriétés d'invariance, de sortie en temps fini et le problème de synthèse d'un contrôleur assurant ces propriétés seront traités. Tout cela va nous permettre à la fin du chapitre d'étudier le problème de planification des trajectoires.

Problème :

On commence par étudier le système dynamique non contrôlé suivant (9.1) :

$$\dot{x}(t) = f(x(t)), \quad x \in R.$$

Le problème avec contrôle considéré est le suivant (9.14) :

$$\dot{x}(t) = f(x(t)) + Bu(t), \quad x \in R$$

où $B \in \mathbb{R}^{n \times p}$ et $u(t) \in \mathbb{R}^p$ est le contrôleur à calculer.

On veut construire un contrôleur assurant l'invariance du rectangle R ainsi qu'un contrôleur assurant la sortie à travers une face donnée du rectangle R.

Aperçu de l'état de l'art

Le problème de synthèse de contrôleur rendant un rectangle invariant ou assurant la sortie par une face donnée du rectangle est étudié dans le cas d'un champ de vecteur multi-affine [13, 97]. Notre idée sera de généraliser l'approche décrite dans [13] dans

le cas d'un système dynamique polynomial.

Méthode :

Pour étudier l'invariance et la sortie à travers une face donnée du rectangle R des trajectoires du système (9.1), l'idée est de construire un système dynamique équivalent de plus grande dimension dont le champ de vecteur est multi-affine. C'est le cadre de notre méthode d'abstraction (voir Section 9.1) où le principal ingrédient est le principe de floraison.

Une fois que l'on a à disposition un système dynamique multi-affine, on utilise l'approche décrite dans [13] pour pouvoir établir des conditions permettant d'assurer l'invariance ou la sortie d'une face donnée du rectangle.
On se base ensuite sur les propriétés de symétrie d'une forme polaire en exploitant notre relation d'équivalence $\cong$ pour pouvoir simplifier les résultats obtenus.

Le résultat principal concernant l'invariance est donné par :

Théorème 17. Le rectangle R est invariant pour le système (9.1) si pour tout sommet $\overline{v} \in \overline{V}'$ et pour tout $i = 1, \ldots, n$, les contraintes suivantes sont vérifiées :

— Si $l_i(\overline{v}) = 0$, il faut que $f_{i,\delta}(\overline{v}) \geq 0$.

— Si $l_i(\overline{v}) = \delta_i$, il faut que $f_{i,\delta}(\overline{v}) \leq 0$

où la définition de $l_i(v)$ pour tout $i = 1, \ldots, n$ est donnée dans la Section 3.3.

En effet, les conditions de ce théorème nous permettent de s'assurer que toutes les faces de R sont bloquées (voir Définition 10) d'où son invariance.

Pour la sortie à travers une face donné du rectangle, on va devoir assurer à la fois que toutes les autres faces sont bloquées et que toutes les trajectoires quittent le rectangle en temps fini. Cela est établit par la Proposition 27. Enfin, on montre que tous les résultats de cette partie peuvent être formulés à l'aide des coefficients de Bernstein.

Dans une deuxième partie, on va devoir généraliser les programmes linéaires de la première partie dans le cas du système contrôlé (9.14). En effet, on obtient d'autres programmes linéaires fournissant de plus un vecteur de contrôle λ. On montre ainsi comment un contrôleur polynomial admissible peut être construit à l'aide de ce vecteur de contrôle. Pour plus de détail, le lecteur peut voir la Section 9.3.

1.3.4 Plan

Dans cette section, nous décrivons brièvement le plan de cette thèse se composant de trois parties.

Partie I : Préliminaires

La première partie de cette thèse nous servira comme préliminaire dans lequel les principales notations, définitions et résultats seront cités. On verra qu'on se référera tout le long de la thèse au contenu de cette partie. Dans cette partie nos deux principaux outils (principe de floraison et polynômes de Berstein) seront présentés.

Partie II : Optimisation polynomiale

La deuxième partie de la thèse, essentiellement théorique, est consacrée au problème d'optimisation polynomiale. Dans cette partie nos deux relaxations linéaires se basant sur le principe de floraison et les polynômes Bernstein seront introduites.

Partie III : Applications dans le cadre des systèmes dynamiques polynomiaux

La troisième et dernière partie de la thèse confirme l'intérêt pratique de la partie théorique sur l'optimisation polynomiale. Elle sera consacré aux applications de nos deux relaxations dans le cadre des systèmes dynamiques polynomiaux pour des domaines polyédraux.

1.3.5 Publications

Les travaux présentés dans cette thèse ont donnés lieu à plusieurs publications. En effet, tous les chapitres de la dernière partie de la thèse (la partie des applications) ont été l'objet d'une soumission dans des journaux où des conférences sélectives.

On citera dans cette section la liste complètes de ces contributions :

Journaux

Computation of Polytopic Invariants for Polynomial Dynamical Systems using Linear Programming. Mohamed Amin Ben Sassi et Antoine Girard. Publié dans *Automatica, 48(12) :3114-3121, 2012*.

Controller Synthesis for Robust Invariance of Polynomial Dynamical Systems using Linear Programming. Mohamed Amin Ben Sassi et Antoine Girard. Publié dans *Systems and Control Letters, 61(4) :506-512, 2012*.

Actes de conférences

Reachability Analysis of Polynomial Dynamical Systems using Linear Programming. Mohamed Amin Ben Sassi, Romain Testylier, Thao Dang et Antoine Girard. Presenté à *Automated Technology for Verification and Analysis*, Trivandrum, Kirala, India, Octobre 2012.

Motion Planning of Polynomial Dynamical Systems on Rectangles. Mohamed Amin Ben Sassi et Antoine Girard. Présenté à *European Control Conference*, Zurich, Suisse, July 2013.

Première partie

Préliminaires

Chapitre 2

Généralités sur les polynômes et principe de floraison

Dans ce chapitre, on va définir les polynômes dans leur cadre le plus général (polynômes multi-variés) ainsi que leurs principales propriétés. On traitera à part une classe bien particulière de ces polynômes dite fonctions multi-affines qui jouera un rôle important par la suite. A la fin de ce chapitre, on introduira le principe de floraison ou notion de forme polaire d'un polynôme qui nous permettra de passer de manière pratique du cas polynomial au cas multi-affine.

2.1 Polynômes multi-variés

Un polynôme d'une seule variable, qu'on appellera aussi un polynôme uni-varié, est toute fonction $p : \mathbb{R} \to \mathbb{R}$ s'écrivant sous la forme :

$$p(x) = \sum_{k=1}^{r} p_k x^k$$

où $r \in \mathbb{N}$ désigne le degré du polynôme p et $\{p_k \in \mathbb{R}\ ,\ k = 1, \ldots, r\}$ désigne l'ensemble de ses coefficients.
On utilisera la notation $\mathbb{R}[x]$ pour désigner l'algèbre des polynômes de variable x. On peut généraliser cette définition dans le cas de plusieurs variables et avoir ainsi un polynôme multi-varié.

Définition 1. *Un polynôme multi-varié est toute fonction $p : \mathbb{R}^n \to \mathbb{R}$ s'écrivant sous la forme :*

$$p(x) = p(x_1, \ldots, x_n) = \sum_{(k_1,\ldots,k_n)\in\Delta} p_{k_1,\ldots,k_n} x_1^{k_1} \ldots x_n^{k_n}.$$

avec $\Delta = \{0, \ldots, \delta_1\} \times \cdots \times \{0, \ldots, \delta_n\}$ où $\delta_1, \ldots, \delta_n$ sont les degrés de p dans les variables respectives $x_1, \ldots, x_n$ et $\{p_{k_1,\ldots,k_n} \in \mathbb{R}\ ,\ (k_1, \ldots, k_n) \in \Delta\}$ désigne l'ensemble de ses coefficients.

Une autre écriture peut être donnée en introduisant la relation d'ordre suivante : soit $I = (i_1, \ldots, i_n) \in \mathbb{N}^n$ et $\delta = (\delta_1, \ldots, \delta_n) \in \mathbb{N}^n$, on dit que $I \leq \delta$ (inégalité entre

multi-indices) si $i_j \leq \delta_j$ pour tout $j \in \{1, \ldots, n\}$.
On note aussi $|I| = i_1 + \cdots + i_n$ et pour $x \in \mathbb{R}^n$, $x^I = x_1^{i_1} \ldots x_n^{i_n}$.
Ainsi, le polynôme p peut s'écrire sous la forme :

$$p(x) = \sum_{I \leq \delta} p_I x^I \text{ avec } p_I \in \mathbb{R}, \ \forall I \leq \delta.$$

Ce polynôme peut être considéré comme polynôme de degré δ (un multi-indice) ou polynôme de degré $|\delta|$.
On s'intéressera au problème d'optimisation d'une fonction objectif polynomiale sur des ensembles compacts bien particuliers tels que les boîtes (produit d'intervalles), les polytopes (contraintes affines) où plus généralement les ensembles semi-algébriques (contraintes polynomiales). Pour étudier ces problèmes, plusieurs techniques sont conçues. L'intérêt de la plupart de ces techniques est de fournir des relaxations faciles à résoudre permettant d'approcher notre solution optimale qu'on notera p^* (c'est-à-dire trouver une borne inférieure ou supérieure à p^*).
Ces relaxations proviennent essentiellement de la théorie de représentation de polynômes positifs sur des ensembles donnés (qui seront dans notre cas les ensembles définis par les contraintes).

Pour cela on va citer les principaux résultats de représentation de polynômes positifs sur quelques exemples d'ensembles compacts.

2.1.1 Représentation des polynômes positifs

Dans la littérature on pourra distinguer deux sortes de représentations de polynômes positifs [138], on va commencer par citer les premières représentations qui ont été établies dans le cas unidimensionnel sur l'intervalle $[-1, 1]$.

Théorème 1. *(Bernstein [20] et Hausdorff [137])*
Soit $p \in \mathbb{R}[x]$. Si $p(x) > 0$ sur $[-1, 1]$ alors

$$p(x) = \sum_{i+j \leq d} c_{i,j}(1-x)^i(1+x)^j, \tag{2.1}$$

pour un entier suffisamment grand d et des constantes $c_{i,j} \geq 0$ pour tout entier i, j vérifiant $i + j \leq d$.

En effet la condition $x \in [-1, 1]$ sera traduite par les contraintes $1 - x \geq 0$ et $1 + x \geq 0$, on peut remarquer que cette représentation est la somme des produits de $1 - x$ et $1 + x$ (ayant des puissances bornés en somme par un certain degré maximal d) pondérés par des coefficients positifs.
La deuxième représentation utilise une classe spéciale de polynômes qu'on appelle somme de carrés de polynômes.

Définition 2. *$p \in \mathbb{R}[x]$ est dit somme de carrés de polynômes qu'on notera $p \in \Sigma[x]$ si et seulement si :*

$$p(x) = \sum_{i=1}^{k} h_i(x)^2$$

pour certains polynômes (h_i), $i = 1 \ldots, k$ avec k entier positif.

En utilisant ces polynômes on a la représentation suivante :

Théorème 2. *(Schmudgen's [160])*
Soit $p \in \mathbb{R}[x]$ de degré r. Alors $p(x) \geq 0$ sur $[-1,1]$ si et seulement si

$$p(x) = f_0 + (1-x)f_1 + (1+x)f_2 + (1-x^2)f_3, \tag{2.2}$$

avec $f_0, f_1, f_2, f_3 \in \Sigma[x]$ ayant tous un degré inférieur ou égal à r.

Ces résultats peuvent s'étendre en dimension supérieure avec des contraintes plus générales que des intervalles. Posons

$$\mathcal{K} = \{x \in \mathbb{R}^n \mid g_k(x) \geq 0, \quad k = 0, \ldots, m\}. \tag{2.3}$$

Dans le cas où $\mathcal{K}$ est un polytope convexe d'intérieur non vide (les g_i sont des fonctions affines) le Théorème 1 se généralise de la façon suivante :

Théorème 3. *(Handelmann [87])*
Soit $p \in \mathbb{R}[x]$. Si $p(x) > 0$ sur $\mathcal{K}$ alors

$$p(x) = \sum_{|\alpha| \leq d} b_\alpha g_1(x)^{\alpha_1} \ldots g_m(x)^{\alpha_m}, \tag{2.4}$$

pour un entier suffisamment grand d et des constantes $b_\alpha \geq 0$ pour tout $\alpha \in \mathbb{N}^n$ vérifiant $|\alpha| = \alpha_1 + \ldots \alpha_n \leq d$.

Par ailleurs, la représentation (2.4) est aussi valable si K est un ensemble semi-algébrique compact (les g_i sont des polynômes) vérifiant la condition suivante [180] :

Hypothèse 1. *L'ensemble $(1, g_j)_{j=1,\ldots,m}$ génère l'algèbre $\mathbb{R}[x] = \mathbb{R}[x_1, \ldots, x_n]$.*

Pour la version généralisé du Théorème 2, la condition sur la positivité du polynôme p sera réduite en positivité stricte et l'ensemble $\mathcal{K}$ défini par (2.3) peut être pris comme étant un ensemble semi-algébrique compact vérifiant maintenant la condition suivante [141] :

Hypothèse 2. *Il existe un polynôme multi-varié $u : \mathbb{R}^n \longrightarrow \mathbb{R}$ tel que l'ensemble $\{x \in \mathbb{R}^n \mid u(x) \geq 0\}$ est compact, et*

$$u(x) = u_0(x) + \sum_{k=1}^{m} g_i(x) u_i(x) \text{ pour tout } x \in \mathbb{R}^n \tag{2.5}$$

où les polynômes $u_i(x)$ sont tous des sommes de carrés de polynômes, $i = 0, \ldots, m$.

Et on aura la représentation suivante :

Théorème 4. *(Putinar [141])*
Soit $p \in \mathbb{R}[x]$. Si $p(x) > 0$ sur $\mathcal{K}$ alors

$$p(x) = f_0 + \sum_{j=1}^{m} f_j g_j, \tag{2.6}$$

avec $f_j \in \Sigma[x], \quad \forall j \in 0, \ldots, m.$

Remarque 1. *Il est évident que si $p \in \Sigma[x]$ alors $p(x) \geq 0$. La réciproque n'est pas toujours vraie.*
En effet, en 1888, Hilbert montre que la réciproque est vrai pour un polynôme à n variables de degré $2d$ (avec $2d = |\delta|$) seulement dans les cas suivantes :

- *Cas d'un polynôme d'une seule variable ($n = 1$).*
- *Cas d'un polynôme quadratique ($d = 1$).*
- *Cas d'un polynôme bi-varié quartique ($n = 2$ et $d = 2$).*

Dans tous les autres cas il existe des contre-exemples de polynômes positifs qui ne sont pas somme de carrés de polynômes (voir Reznick [149] pour plus de détails).

Remarque 2. *(Lasserre [101]) Il est important de remarquer que l'Hypothèse 2 n'est pas très restrictive. En effet, elle est satisfaite par exemple, s'il existe un polynôme g_k tel que l'ensemble $\{x \in \mathbb{R}^n \mid g_k(x) \geq 0\}$ est compact, car dans ce cas il suffira de prendre tous les $u_i \equiv 0$ sauf $u_k \equiv 1$ dans (2.5).*
Plus généralement, une façon de l'assurer sera d'ajouter dans la définition de $\mathcal{K}$ la contrainte supplémentaire $g_{m+1}(x) = a^2 - ||x||^2$ pour un réel a suffisamment grand, ce qui est toujours possible étant donné que l'ensemble $\mathcal{K}$ est compact, où $||x||$ désigne la norme Euclidienne de x.

Les relaxations qu'on étudiera dans la seconde partie, et plus précisément dans le premier chapitre de cette partie, reposent essentiellement sur ces représentations. En effet, l'idée est de représenter le polynôme positif $p(x)-p^*$ en fonction des contraintes données par l'ensemble $\mathcal{K}$. Ainsi, toute la difficulté sera d'en trouver les coefficients constants ou polynomiaux (selon le type de la représentation choisie) associés. C'est à l'aide de la relaxation, et plus précisément de son dual, que ces coefficients pourront être obtenus. Pour cela, on va faire face à deux problèmes : le premier est le fait que le degré de la représentation et donc de la relaxation n'est pas connu à l'avance et le deuxième est le fait que le polynôme $p(x) - p^*$ n'est pas forcément strictement positif. Enfin, on finit par remarquer que trouver des coefficients polynomiaux f_i sera certainement plus difficile et plus coûteux que trouver des coefficients constants b_α.

2.1.2 Cas particulier important : les fonctions multi-affines

Une fonction multi-affine est un polynôme multi-varié qui s'écrit comme fonction affine en chacune de ses variables quand les autres sont considérées comme des constantes.

Définition 3. *Une fonction multi-affine $p : \mathbb{R}^n \to \mathbb{R}$ est un polynôme multi-varié dont le degré en chaque variable $x_1, \ldots, x_n$ est au plus égal à 1 :*

$$p(x) = p(x_1, \ldots, x_n) = \sum_{(k_1,\ldots,k_n)\in\{0,1\}^n} p_{k_1,\ldots,k_n} x_1^{k_1} \ldots x_n^{k_n}$$

avec $p_{k_1,\ldots,k_n} \in \mathbb{R}$ pour tout $(k_1, \ldots, k_n) \in \{0, 1\}^n$.

Par exemple, une fonction multi-affine bi-variée est de la forme :

$$p(x_1, x_2) = p_{00} + p_{10}x_1 + p_{01}x_2 + p_{11}x_1x_2.$$

Exemple 1. *Les équations de Lotka-Volterra que l'on désigne aussi sous le terme de modèle proie-prédateur, ont été proposées indépendamment par Lotka en 1925 et Volterra en 1926 pour décrire la dynamique de systèmes biologiques dans lesquels un prédateur et sa proie interagissent.*

Ces équations ont la forme suivante :

$$\begin{cases} \dot{x} &= f_1(x, y) = \alpha x - \beta xy, \\ \dot{y} &= f_2(x, y) = \gamma xy - \delta y, \end{cases}$$

où $\alpha, \beta, \gamma, \delta$ sont des constantes.

Un intérêt des fonctions multi-affines réside dans le fait qu'elles présentent une certaine particularité sur les domaines rectangulaires (les boîtes). Avant de l'énoncer on a besoin d'introduire les notations $R_n = \{x = (x_1, .., x_n) \in \mathbb{R}^n | \forall i \in \{1, .., n\} : a_i \leqslant x_i \leqslant b_i\}$ pour désigner un rectangle de $\mathbb{R}^n$, $V_n = \prod_{i=1}^{n} \{a_i, b_i\}$ l'ensemble de sommets associés à R_n et la fonction $\xi_k : \{a_k, b_k\} \mapsto \{0, 1\}$ avec $\xi_k(a_k) = 0$ et $\xi_k(b_k) = 1 \quad \forall k \in \{1, \ldots, n\}$. Soit $f : R_n \to \mathbb{R}^p$ une fonction multi-affine c'est à dire que les fonctions f_i sont multi-affines pour tout $i \in \{1, .., n\}$.
Le résultat est donné par le lemme suivant dit lemme de convexité [13] :

Lemme 1. *$f(x_1, \ldots, x_n)$ est une combinaison convexe des valeurs de f au sommets de R_n pour tout élément $x = (x_1, \ldots, x_n) \in R_n$.*

Grâce à ce lemme on déduit simplement le corollaire suivant :

Corollaire 1. *Soit $p : \mathbb{R}^n \to \mathbb{R}$ une fonction multi-affine et soit R_n un rectangle de $\mathbb{R}^n$ avec un ensemble de sommets noté V_n, alors $\min_{x \in R_n} p(x) = \min_{v \in V_n} p(v)$.*

2.2 Le principe de floraison ou forme polaire d'un polynôme

Le principe de floraison (en Anglais "blossom") est un outil mathématique mis en oeuvre par Lille Ramshaw [143] en 1987. La motivation était de simplifier la construction des courbes polynomiales et mettre en oeuvre de nouvelles représentations surfaciques et de nouveaux algorithmes. Ainsi cet outil s'est avéré utile dans le domaine de la géométrie assistée par ordinateur [161]. Par ailleurs, le principe de floraison débouche sur de nombreuses simplifications dans le contexte des B-splines comme c'est décrit dans [144, 161] et plus particulièrement dans les travaux de De Casteljau (voir [56, 57]).

La floraison ou forme polaire d'un polynôme donné est une transformation qui permet de réduire la complexité de ce polynôme en ajoutant de nouvelles variables tout en ayant une certaine propriété de symétrie. Cette transformation est unique si le degré du polynôme est fixé à l'avance. On donnera par la suite la définition dans le cadre général de la forme polaire d'un polynôme multi-varié et on citera ses principales propriétés. Celles-ci seront très utiles dans la deuxième partie de cette thèse.

Définition 4. *Soit $\delta \in \mathbb{N}^n$. La floraison relative à δ d'un polynôme multi-varié $p : \mathbb{R}^n \to \mathbb{R}$ est la fonction $q_\delta : \mathbb{R}^{\delta_1+\cdots+\delta_n} \to \mathbb{R}$ donnée pour tout élément $z = (z_{1,1}, \ldots, z_{1,\delta_1}, \ldots, z_{n,1}, \ldots, z_{n,\delta_n}) \in \mathbb{R}^{\delta_1+\cdots+\delta_n}$ par :*

$$q_\delta(z) = \sum_{(k_1,\ldots,k_n)\in\Delta} p_{k_1,\ldots,k_n} \mathcal{B}_{k_1,\delta_1}(z_{1,1}, \ldots, z_{1,\delta_1}) \ldots \mathcal{B}_{k_n,\delta_n}(z_{n,1}, \ldots, z_{n,\delta_n})$$

avec

$$\mathcal{B}_{k,d}(z_1, \ldots, z_d) = \frac{1}{\binom{d}{k}} \sum_{\sigma\in C(k,d)} z_{\sigma_1} \ldots z_{\sigma_k}$$

et $C(k, d)$ désigne l'ensemble des combinaisons de k éléments dans $\{1, \ldots, d\}$.

Remarque 3. *On notera tout simplement q la forme polaire de p où $q = q_\delta$, δ est le degré du polynôme p. Notons que p peut être vu comme étant un polynôme de degré supérieur δ' en ajoutant des coefficients non nuls. Ainsi, on pourra obtenir une nouvelle forme polaire $q_{\delta'}$ de p relative à δ'.*

Un exemple aidera à bien comprendre la définition précédente :

Exemple 2. *La forme polaire du polynôme bi-variés $p(x_1, x_2) = 3x_1 + x_1^2x_2^2 + 2x_2^3$ est la suivante :*

$$q(z_{1,1}, z_{1,2}, z_{2,1}, z_{2,2}, z_{2,3}) = \frac{3}{2}(z_{1,1}+z_{1,2})+\frac{1}{3}z_{1,1}z_{1,2}(z_{2,1}z_{2,2}+z_{2,1}z_{2,3}+z_{2,2}z_{2,3})+2z_{2,1}z_{2,2}z_{2,3}.$$

Proposition 1. *Il découle de la Définition 4 que la forme polaire q_δ du polynôme p satisfait les propriétés suivantes :*

1. *C'est une fonction multi-affine .*
2. *C'est une fonction symétrique par rapport à ses arguments :*

   $$q_\delta(z_{1,1}, \ldots, z_{1,\delta_1}, \ldots, z_{n,1}, \ldots, z_{n,\delta_n}) = q_\delta(\pi_1(z_{1,1}, \ldots, z_{1,\delta_1}), \ldots, \pi_n(z_{n,1}, \ldots, z_{n,\delta_n})),$$

   avec π_i est une permutation quelconque de ses δ_i arguments .
3. *Elle satisfait la propriété diagonale :*

   $$q_\delta(z_1, \ldots, z_1, \ldots, z_n, \ldots, z_n) = p(z_1, \ldots, z_n).$$

Démonstration. Il est évident par construction que le polynôme q_δ est bien une fonction multi-affine.
La propriété de symétrie provient du fait que tous les $\mathcal{B}_{k_i,\delta_i}$ sont symétriques par rapport à leurs arguments $z_{i,1}, \ldots, z_{i,\delta_i}$ avec $i \in \{1, \ldots, n\}$ et $k_i \in \{1, \ldots, \delta_i\}$.
Enfin la propriété diagonale peut être vérifiée facilement comme suit :

$$\begin{aligned} q_\delta(z_1, .., z_1, .., z_n, \ldots, z_n) &= \sum_{(k_1,\ldots,k_n)\in\Delta} p_{k_1,\ldots,k_n} \mathcal{B}_{k_1,\delta_1}(z_1, .., z_1) \ldots \mathcal{B}_{k_n,\delta_n}(z_n, .., z_n) \\ &= \sum_{(k_1,\ldots,k_n)\in\Delta} p_{k_1,\ldots,k_n} {z_1}^{k_1} \ldots {z_n}^{k_n} \\ &= p(z_1, \ldots, z_n). \end{aligned}$$

□

La floraison permet donc de passer d'un polynôme de n variables de degrés $\delta_1, \ldots, \delta_n$ à une fonction multi-affine de $\delta_1 + \cdots + \delta_n$ variables. Ainsi on a un gain en complexité mais en contre partie une croissance en dimension. Par ailleurs, on aura une symétrie importante à exploiter entre les nouvelles variables.

Chapitre 3

Polynômes de Bernstein

Les polynômes de Bernstein (dûs au mathématicien Ukrainien Sergeï Natanovich Bernstein) ont permis de donner une preuve constructive du théorème de Stone-Weierstrass [1]. Ils sont aussi utilisés dans la formulation générale des courbes de Bézier. Ces polynômes représentent également un outil important pour approcher l'image d'un polynôme multi-varié dans un domaine rectangulaire (boîte). En effet, en 1965, Cargo [61] a d'abord utiliser la forme de Bernstein pour borner l'image par un polynôme de l'intervalle $[0, 1]$ c'est à dire l'ensemble $\{p(x) \ : 0 \leq x \leq 1\}$ où p est un polynôme d'une seule variable. Le cas bi-varié a été traité par Garloff [69] en 1985. L'idée a été ensuite généralisée pour les polynômes multi-variés [150, 152, 99] . Par conséquent de nombreuses applications se basant sur la forme de Bernstein ont vu le jour notamment dans le domaine d'analyse par intervalle avec Rokne [153, 155, 154, 111], Fisher [65] et Stahl dans sa thèse [166].

Dans le cadre de cette thèse, ces polynômes vont servir à donner des relaxations linéaires et convexes à des problèmes d'optimisation polynomiaux généralement non linéaires et non convexes. En effet, à l'aide de ces polynômes on peut approcher des fonctions polynomiales par d'autres fonctions affines ou affines par morceaux [74] et donc un ensemble non convexe par un ensemble convexe (son enveloppe convexe par exemple). Enfin, ces polynômes peuvent aussi aider à résoudre certains problèmes de contrôle comme c'est le cas dans [71, 72].

3.1 Polynômes de Bernstein : définition et propriétés

On commence par donner la définition des polynômes de Bernstein dans le cas uni-dimensionnel puis on donnera la définition dans le cas général. Notons que les expressions de ces polynômes seront données dans des boîtes unités. On peut se ramener au cas des boîtes quelconques en appliquant un changement de variable affine.

1. Il s'agit d'un théorème prouvant qu'on peut approcher une fonction continue sur un intervalle par une suite de polynômes

Définition 5. *Dans* $[0,1]$*, les polynômes de Bernstein de degré* m *sont définis comme suit :*

$$B_{i,m}(x) = \begin{pmatrix} m \\ i \end{pmatrix} x^i (1-x)^{m-i}, \quad \forall i \in \{0,\dots,m\}. \tag{3.1}$$

Proposition 2. *Les polynômes de Bernstein forment une base pour les polynômes de degré* m *et plus précisément pour tout* $0 \leq i \leq m$ *on a :*

$$x^i = \sum_{j=i}^{m} \frac{\begin{pmatrix} j \\ i \end{pmatrix}}{\begin{pmatrix} m \\ i \end{pmatrix}} B_{j,m}(x). \tag{3.2}$$

Démonstration. Pour tout $i = 0,\dots,m$, on a :

$$\begin{aligned} x^i &= x^i(x+(1-x))^{m-i} = \sum_{k=0}^{m-i} \begin{pmatrix} m-i \\ k \end{pmatrix} x^{i+k}(1-x)^{m-i-k} \\ &= \sum_{j=i}^{m} \begin{pmatrix} m-i \\ j-i \end{pmatrix} x^j (1-x)^{m-j} = \sum_{j=i}^{m} \frac{\begin{pmatrix} j \\ i \end{pmatrix}}{\begin{pmatrix} m \\ i \end{pmatrix}} \begin{pmatrix} m \\ j \end{pmatrix} x^j (1-x)^{m-j} \\ &= \sum_{j=i}^{m} \frac{\begin{pmatrix} j \\ i \end{pmatrix}}{\begin{pmatrix} m \\ i \end{pmatrix}} B_{j,m}(x). \end{aligned}$$

□

Ainsi, grâce à (3.2) on peut voir que les éléments x^i sont des combinaisons linéaires des polynômes de Bernstein pour tout $i = 0,\dots,m$.
Maintenant, soit p un polynôme de degré m alors p s'écrit sous la forme suivante :

$$p(x) = \sum_{i=0}^{m} p_i x^k$$

où l'ensemble $\{p_i \in \mathbb{R}\ ,\ i = 0,\dots,m\}$ désigne l'ensemble des coefficients du polynôme p dans la base canonique.

A l'aide de (3.2), on pourra écrire le polynôme p dans la base de Bernstein et calculer explicitement son nouvel ensemble de coefficients.

Proposition 3. *Dans la base de Bernstein* p *s'écrit sous la forme :*

$$p(x) = \sum_{i=0}^{m} b_i B_{i,m}(x)$$

avec pour tout $i = 0,\dots,m$ *:*

$$b_i = \sum_{j=0}^{i} \frac{\begin{pmatrix} i \\ j \end{pmatrix}}{\begin{pmatrix} m \\ j \end{pmatrix}} p_j. \tag{3.3}$$

Démonstration. Pour la preuve il suffit d'utiliser (3.2) :

$$\begin{aligned} p(x) &= \sum_{j=0}^{m} p_j x^j = \sum_{j=0}^{m} p_j \sum_{i=j}^{m} \frac{\binom{i}{j}}{\binom{m}{j}} B_{i,m}(x) \\ &= \sum_{i=0}^{m} \sum_{j=0}^{i} p_j \frac{\binom{i}{j}}{\binom{m}{j}} B_{i,m}(x) = \sum_{i=0}^{m} b_i B_{i,m}(x). \end{aligned}$$

Ainsi pour tout $i = 0, \ldots, m$ on aura :

$$b_i = \sum_{j=0}^{i} \frac{\binom{i}{j}}{\binom{m}{j}} p_j.$$

□

Citons maintenant les principales propriétés qui caractérisent ces polynômes.

Proposition 4. *Les propriétés suivantes sont valables pour tout $x \in [0,1]$:*

1. *Partition de l'unité :* $\sum_{i=0}^{m} B_{i,m}(x) = 1.$
2. $0 \leq B_{i,m}(x) \leq B_{i,m}(\frac{i}{m})$ *avec* $i = 0, \ldots, m.$
3. *Symétrie :* $B_{i,m}(x) = B_{m-i,m}(1-x)$ *avec* $i = 0, \ldots, m.$
4. $x = \sum_{i=0}^{m} B_{i,m}(x) \frac{i}{m}.$
5. $B_{i,m-1}(x) = \frac{n-i}{n} B_{i,m}(x) + \frac{i+1}{n} B_{i+1,m}(x).$

Démonstration. Pour la partition de l'unité il suffit d'utiliser la formule du binôme de Newton. En effet,

$$\sum_{i=0}^{m} B_{i,m}(x) = \sum_{i=0}^{m} \binom{m}{i} x^i (1-x)^{m-i} = (x + 1 - x)^m = 1.$$

D'autre part, une simple dérivation montre que sur $[0,1]$, la fonction $B_{i,m}(x)$ est positive et qu'elle atteint son maximum au point $\frac{i}{m}$ pour tout $i \in \{0, \ldots, m\}$.
Pour la propriété de symétrie, il suffit de voir que x et $1-x$ jouent des rôles symétriques dans la Définition 5 en permutant les degrés i et $m-i$.
Pour démontrer la propriété 4, il suffit de remarquer que :

$$\sum_{i=0}^{m} B_{i,m}(x) \frac{i}{m} = x \left(\sum_{i=1}^{m} B_{i-1,m-1}(x) \right) = x.$$

Enfin, la dernière propriété concernant l'écriture d'un polynôme de Bernstein comme étant une combinaison linéaire d'autres polynômes de Bernstein de dimension supérieure, se démontre comme suit :

$$\begin{aligned} A &= \frac{n-i}{n}B_{i,m}(x) + \frac{i+1}{n}B_{i+1,m}(x) \\ &= x^i(1-x)^{m-1-i}\left[\frac{n-i}{n}\begin{pmatrix} m \\ i \end{pmatrix}(1-x) + \frac{i+1}{n}\begin{pmatrix} m \\ i+1 \end{pmatrix}x\right] \\ &= x^i(1-x)^{m-1-i}\left[\begin{pmatrix} m-1 \\ i \end{pmatrix}(1-x) + \begin{pmatrix} m-1 \\ i \end{pmatrix}x\right] \\ &= B_{i,m-1}(x). \end{aligned}$$

□

On verra plus loin dans cette thèse qu'en utilisant la décomposition de polynômes dans la base de Bernstein ainsi que les propriétés précédentes, on arrive à obtenir des relaxations linéaires à des problèmes d'optimisation polynomiale généralement non linéaires et non convexes.

Maintenant, on va généraliser ces notions au cas des polynômes multi-variés c'est à dire $p(x) = p(x_1, \ldots, x_n)$ où $x = (x_1, \ldots, x_n) \in U = [0,1]^n$.
Pour cela, on aura besoin des notations suivantes :
Soient $I = (i_1, \ldots, i_n)$, $J = (j_1, \ldots, j_n)$ deux multi indices :

- Pour $x \in \mathbb{R}^n$, $\quad 1_n - x = (1 - x_1, \ldots, 1 - x_n)$.
- $\frac{I}{J} = \left(\frac{i_1}{j_1}, \ldots, \frac{i_n}{j_n}\right)$ pour tout couple de multi indice (I, J).
- $\begin{pmatrix} I \\ J \end{pmatrix} = \begin{pmatrix} i_1 \\ j_1 \end{pmatrix} \ldots \begin{pmatrix} i_n \\ j_n \end{pmatrix}$.
- $I_{r,k} = (i_1, \ldots, i_{r-1}, i_r + k, i_{r+1}, \ldots, i_n)$ pour tout multi indice I et tout entier $r \in \{1, \ldots, n\}$, $k \in \mathbb{Z}$.

Fixons $\delta = (\delta_1, \ldots, \delta_n) \in \mathbb{N}^n$ comme étant le degré maximal du polynôme p. Ainsi p s'écrit sous la forme suivante :

$$p(x) = \sum_{I \leq \delta} p_I x^I \text{ avec } p_I \in \mathbb{R},\ \forall I \leq \delta.$$

Les polynômes de Bernstein seront ainsi donnés par :

$$B_{I,\delta}(x) = \beta_{i_1,\delta_1}(x_1) \ldots \beta_{i_n,\delta_n}(x_n) \text{ avec } \beta_{i_j,\delta_j}(x_j) = \begin{pmatrix} \delta_j \\ i_j \end{pmatrix} x_j^{i_j}(1-x_j)^{\delta_j - i_j}. \quad (3.4)$$

Par conséquent on pourra écrire :

$$B_{I,\delta}(x) = \begin{pmatrix} \delta \\ I \end{pmatrix} x^I (1_n - x)^{\delta - I}. \quad (3.5)$$

Le résultat suivant se démontre de manière analogue au cas unidimensionnel.

Proposition 5. *Dans la base de Bernstein p s'écrit sous la forme :*

$$p(x) = \sum_{I \leq \delta} b_I B_{I,\delta}(x)$$

avec une séquence de coefficients $(b_I)_{I \leq \delta}$ donnée par

$$b_I = \sum_{J \leq I} \frac{\begin{pmatrix} i_1 \\ j_1 \end{pmatrix} \dots \begin{pmatrix} i_n \\ j_n \end{pmatrix}}{\begin{pmatrix} \delta_1 \\ j_1 \end{pmatrix} \dots \begin{pmatrix} \delta_n \\ j_n \end{pmatrix}} p_J = \sum_{J \leq I} \frac{\begin{pmatrix} I \\ J \end{pmatrix}}{\begin{pmatrix} \delta \\ J \end{pmatrix}} p_J. \tag{3.6}$$

Par ailleurs, on peut généraliser les propriétés de la Proposition 4.

Proposition 6. *Pour tout $x = (x_1, \dots, x_n) \in U$ on a les propriétés suivantes :*

1. *Partition de l'unité :* $\sum_{I \leq \delta} B_{I,\delta}(x) = 1$.
2. $0 \leq B_{I,\delta}(x) \leq B_{I,\delta}(\frac{I}{\delta})$.
3. *Symétrie :* $B_{I,\delta}(x) = B_{\delta - I,\delta}(1_n - x)$ *où* 1_n *est le n-uplet* $(1, \dots, 1)$.
4. $x = \sum_{I \leq \delta} B_{I,\delta}(x) \frac{I}{\delta}$.
5. $B_{I,\delta_{r,-1}} = \frac{\delta_r - i_r}{\delta_r} B_{I,\delta} + \frac{i_r + 1}{\delta_r} B_{I_{r,1},\delta}$.

Démonstration. La preuve de chaque propriété se fait par la généralisation de celle faite dans le cas $n = 1$.
En effet, pour la première propriété il suffit d'écrire :

$$\begin{aligned} \sum_{I \leq \delta} B_{I,\delta}(x) &= \sum_{I \leq \delta} \begin{pmatrix} \delta \\ I \end{pmatrix} x^I (1_n - x)^{\delta - I} \\ &= \sum_{i_1 \leq \delta_1} \begin{pmatrix} \delta_1 \\ i_1 \end{pmatrix} {x_1}^{i_1} (1 - x_1)^{\delta_1 - i_1} \times \dots \times \sum_{i_n \leq \delta_n} \begin{pmatrix} \delta_n \\ i_n \end{pmatrix} {x_n}^{i_n} (1 - x_n)^{\delta_n - i_n} \\ &= \sum_{i_1 \leq \delta_1} \beta_{i_1,\delta_1}(x_1) \times \dots \times \sum_{i_n \leq \delta_n} \beta_{i_n,\delta_n}(x_n) = 1. \end{aligned}$$

Pour la deuxième, on sait déjà que pour tout $j = 1, \dots, n$, on a :

$$0 \leq B_{i_j,\delta_j}(x_j) \leq B_{i_j,\delta_j}(\frac{i_j}{\delta_j}).$$

Ainsi, en faisant les produits de ces inégalités on obtient :

$$0 \leq B_{I,\delta}(x) \leq B_{I,\delta}(\frac{I}{\delta}).$$

La troisième propriété se démontre comme suit :

$$\begin{aligned} B_{I,\delta}(x) &= \beta_{i_1,\delta_1}(x_1) \dots \beta_{i_n,\delta_n}(x_n) \\ &= \beta_{\delta_1 - i_1,\delta_1}(1 - x_1) \dots \beta_{\delta_n - i_n,\delta_n}(1 - x_n) \\ &= B_{\delta - I,\delta}(1_n - x). \end{aligned}$$

Pour la propriété 4, il suffit de remarquer que :

$$\begin{aligned}\sum_{I\leq\delta} B_{I,\delta}(x)\frac{I}{\delta} &= \left(\sum_{i_1=0}^{\delta_1} B_{i_1,\delta_1}(x_1)\frac{i_1}{\delta_1},\ldots,\sum_{i_n=0}^{\delta_n} B_{i_n,\delta_n}(x_n)\frac{i_n}{\delta_n}\right)\\ &= (x_1,\ldots,x_n) = x.\end{aligned}$$

Enfin, d'après la définition de $\delta_{r,-1}$ on aura :

$$B_{I,\delta_{r,-1}} = \beta_{i_1,\delta_1}\times\cdots\times\beta_{i_{r-1},\delta_{r-1}}\times B_{i_r,\delta_r-1}\times B_{i_{r+1},\delta_{r+1}}\times\cdots\times B_{i_n,\delta_n}.$$

Et d'après la propriété 5 de la Proposition 4 on aura :

$$\beta_{i_{r-1},\delta_{r-1}} = \frac{\delta_r - i_r}{\delta_r}\beta_{i_r,\delta_r} + \frac{i_r+1}{\delta_r}\beta_{i_r+1,\delta_r}.$$

Il suffit alors d'injecter ce résultat dans l'expression de $B_{I,\delta_{r,-1}}$ pour que le résultat de la propriété soit obtenu. □

Dans le cas d'une boîte quelconque $R = [\underline{x_1},\overline{x_1}]\times\cdots\times[\underline{x_n},\overline{x_n}]$ il suffit de faire le changement de variables $x_j = \underline{x_j} + z_j(\overline{x_j} - \underline{x_j})$ pour tout $j = 1,\ldots,n$ pour avoir ainsi $z = (z_1,\ldots,z_n)\in U$.
Plus précisément, soit $T:\mathbb{R}^n\longrightarrow\mathbb{R}^n$ la transformation affine donnée par :

$$z = T(x) = \left(\frac{x_1-a_1}{b_1-a_1},\ldots,\frac{x_n-a_n}{b_n-a_n}\right). \tag{3.7}$$

Il suffit alors de calculer son inverse T^{-1} qui renvoie la boîte unité U vers R_n. Ainsi, si on pose $\tilde{p} = p\circ T^{-1}$, le polynôme p peut être écrit dans la base de Bernstein sous la forme :

$$p(x) = \tilde{p}(z) = \sum_{I\leq\delta} b_{I,\delta}(\tilde{p})B_{I,\delta}(z).$$

Dans la section suivante nous donnons quelques applications des polynômes de Bernstein qui nous seront utiles par la suite.

3.2 Quelques applications des polynômes de Bernstein

Comme on l'a annoncé au début de ce chapitre, une application importante de la forme de Bernstein est de pouvoir borner l'ensemble $p(U)$ (l'image de l'ensemble U par le polynôme p).

3.2.1 Image d'une boîte par un polynôme

Un aspect important des coefficients de Bernstein est le fait qu'ils nous permettent de borner l'image de la boîte U par un polynôme p.
En effet, on a le résultat suivant :

Proposition 7. *L'ensemble image d'un polynôme à valeurs $x \in U$ vérifie l'inégalité suivante [70] :*

$$\min\{b_I, I \leq \delta\} \leq p(x) \leq \max\{b_I, I \leq \delta\}. \tag{3.8}$$

Ce résultat a aussi la formulation convexe suivante [72] :

$$\left\{ \begin{pmatrix} x \\ p(x) \end{pmatrix} : x \in U \right\} \subseteq conv \left\{ \begin{pmatrix} \frac{I}{\delta} \\ b_I \end{pmatrix} : I \leq \delta \right\} \tag{3.9}$$

où les points $\begin{pmatrix} \frac{I}{\delta} \\ b_I \end{pmatrix}$ *avec* $I \leq \delta$ *sont appelés les points de contrôle.*

Démonstration. La preuve découle des propriétés des polynômes de Bernstein qu'on a cité dans la Proposition 6. Le premier résultat est une conséquence de la Propriété 1 des polynômes de Bernstein. En effet pour tout multi-indice $I \leq \delta$ et tout $x \in U$ on a :

$$(\min_{I \leq \delta} b_I) B_{I,\delta}(x) \leq b_I B_{I,\delta}(x) \leq (\max_{I \leq \delta} b_I) B_{I,\delta}(x).$$

Ainsi en passant à la somme et en utilisant le fait que $\sum_{I \leq \delta} B_{I,\delta}(x) = 1$ on aura :

$$\min\{b_I, I \leq \delta\} \leq p(x) \leq \max\{b_I, I \leq \delta\}.$$

Pour le second résultat concernant la formulation convexe, fixons $x \in U$. Il suffit d'utiliser la Propriété 4 pour voir que :

$$\begin{pmatrix} x \\ p(x) \end{pmatrix} = \sum_{I \leq \delta} B_{I,\delta}(x) \begin{pmatrix} \frac{I}{\delta} \\ b_I \end{pmatrix}$$

et comme $0 \leq B_{I,\delta}(x) \leq 1$ pour tout $x \in U$ (d'après la Propriété 2) alors $\begin{pmatrix} x \\ p(x) \end{pmatrix}$ est une combinaison convexe des points de contrôle $\begin{pmatrix} \frac{I}{\delta} \\ b_I \end{pmatrix}$, d'où le résultat. □

Ainsi les coefficients de Bernstein nous permettent de sur-approximer l'ensemble $p(U)$. Une question naturelle sera donc de savoir si l'on peut avoir l'image exacte, et si oui sous quelles conditions.
Le lemme suivant montre que c'est uniquement le cas quand une *condition au sommets* est assurée [70, 166].

Lemme 2. *Notons* $S = \{0, \ldots, \delta_1\} \times \cdots \times \{0, \ldots, \delta_n\}$ *et* $S_0 = \{0, \delta_1\} \times \cdots \times \{0, \delta_n\}$. *Notons par* $\overline{p_U}$ *(respectivement* $\underline{p_U}$*) la valeur maximale (respectivement minimale) du polynôme* p *dans la boîte* U. *On aura :*

$$\overline{p_U} = \max\{b_I, I \in S\} \text{ si seulement si } \max\{b_I, I \in S\} = \max\{b_I, I \in S_0\}. \tag{3.10}$$

$$\underline{p_U} = \min\{b_I, I \in S\} \text{ si seulement si } \min\{b_I, I \in S\} = \min\{b_I, I \in S_0\}. \tag{3.11}$$

Démonstration. On commencera par donner la preuve en dimension 1 c'est à dire pour $U = [0,1]$ et $\delta = \delta_1$. On verra que le cas n-dimensionnel est simplement une généralisation de cette preuve.
On va démontrer (3.10), la preuve de (3.11) se faisant de façon analogue.
" $\Leftarrow$ " On suppose que $\max\{b_i, i \in 0, \dots, \delta_1\} = \max\{b_0, b_{\delta_1}\}$.
Or $b_0 = p_0 = p(0)$ et $b_{\delta_1} = p_0 + p_1 + \cdots + p_{\delta_1} = p(1)$, donc on a :

$$\max\{b_i, i \in \{0, \dots, \delta_1\}\} \leq \overline{p_U}.$$

Ainsi d'après (3.8) on déduit que :

$$\max\{b_i, i \in \{0, \dots, \delta_1\}\} = \overline{p_U}.$$

Pour le cas général on note I^* le multi-indice associé à la solution optimale : $b_{I^*} = \max\{b_I, I \in S_0\}$, et comme $I^* \in S_0$, on montre que $b_{I^*} = p(v)$ avec $v \in U$ vérifiant $v_j = 0$ si ${I^*}_j = 0$ et $v_j = 1$ si ${I^*}_j = \delta_j$.
" $\Rightarrow$ " On suppose que $\overline{p_U} = \max\{b_i, i \in 0, \dots, \delta_1\}$.
-Si $b_0 = b_1 = \cdots = b_{\delta_1}$ alors il est évident que $\max\{b_i, i \in \{0, \dots, \delta_1\}\} = \max\{b_0, b_{\delta_1}\}$.
-Sinon, si $0 < x < 1$ alors $0 < B_{i,\delta_1} < 1$ pour tout $i = 0, \dots, \delta_1$. Ainsi :

$$p(x) = \sum_{i=0}^{m} b_i B_{i,m}(x) < \max\{b_i, i \in \{0, \dots, \delta_1\}\} \sum_{i=0}^{m} B_{i,m}(x) = \max\{b_i, i \in \{0, \dots, \delta_1\}\}.$$

Par conséquent, p atteint son maximum sur $[0,1]$ en $x = 0$ ou en $x = 1$, et donc :

$$\max\{b_i, i \in \{0, \dots, \delta_1\}\} = \overline{p_U} = \max\{p(0), p(1)\} = \max\{b_0, b_{\delta_1}\}.$$

□

Exemple 3. *On considère le polynôme* $p(x) = x^3 - x^2 + x$.
Les coefficients de Bernstein associés sont $b_0 = 0$, $b_1 = \frac{1}{3}$, $b_2 = \frac{1}{3}$ *et* $b_3 = 1$. *Par conséquent pour tout* $x \in [0,1]$ *on a :* $0 \leq p(x) \leq 1$ *comme le montre la Figure 3.1.*

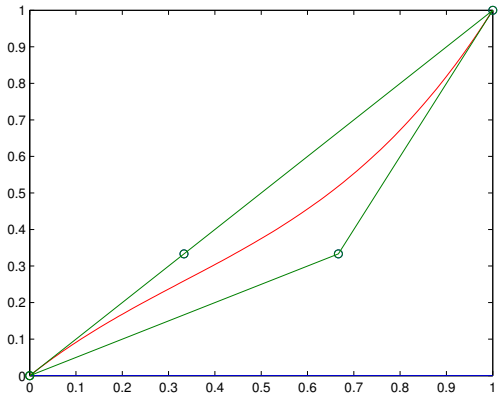

FIGURE 3.1 – Le graphe de la fonction p sur $[0,1]$ ainsi que ses points de contrôle.

3.2.2 Convergence en degré supérieur

Un polynôme multi-varié $p(x) = p(x_1, \ldots, x_n)$ de degré $\delta = (\delta_1, \ldots, \delta_n)$ peut toujours être considéré comme étant un polynôme de degré supérieur $\delta' = (\delta'_1, \ldots, \delta'_n)$ pour certain δ' avec $\delta \leq \delta'$. Il suffit pour cela d'ajouter des coefficients nuls au polynôme initial pour le rendre de degré supérieur.

Nous allons voir que si on fait croître δ alors les coefficients de Bernstein $b_I = b_{I,\delta}$ vont tendre vers les valeurs du polynôme p au points $\frac{I}{\delta}$. Pour cela on va utiliser le théorème suivant [111] :

Théorème 5. *Soit p un polynôme multi-varié de degré $\delta = (\delta_1, \ldots, \delta_n)$ où $b_{I,\delta} = b_I$ sont les coefficients de Bernstein associés à p. Alors on a :*

$$\left| b_{I,\delta} - p\left(\frac{I}{\delta}\right) \right| = O\left(\frac{1}{\delta_1} + \cdots + \frac{1}{\delta_n}\right) \text{ pour tout } I \leq \delta. \tag{3.12}$$

Exemple 4. *On reprend l'Exemple 3, on considère en un premier temps p comme étant un polynôme de degré $\delta = 5$ puis de degré $\delta = 10$. La Figure 3.2 montre que les points de contrôle se rapprochent de plus en plus du polynôme p.*

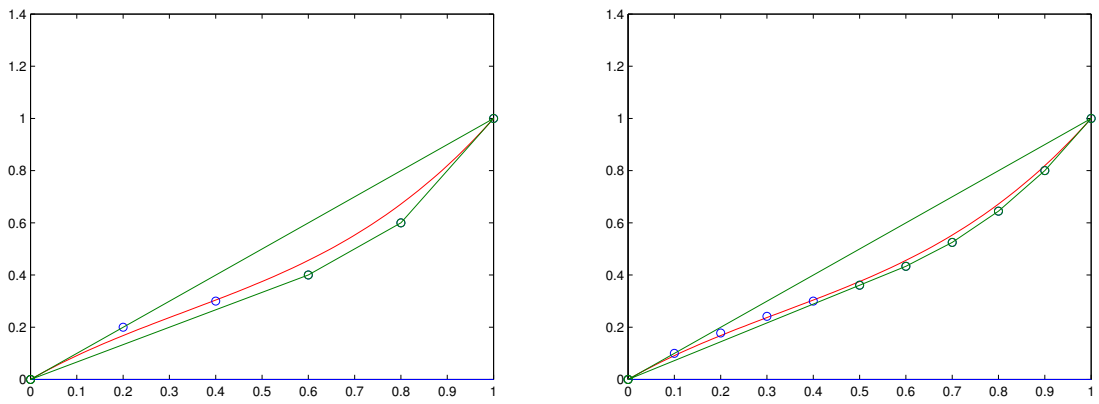

FIGURE 3.2 – Le graphe de la fonction p sur $[0, 1]$ ainsi que ses points de contrôle en la considèrant comme étant un polynôme de degré $\delta = 5$ (à gauche) et de degré $\delta = 10$ (à droite)

Ce résultat montre qu'on peut toujours augmenter la précision des coefficients de Bernstein b_I en considérant notre polynôme p de degré δ comme étant un polynôme de degré supérieur δ'. Ainsi, si on fait tendre le degré δ vers l'infini l'ensemble des coefficients de Bernstein $\{b_I, I \leq \delta\}$ tend vers l'ensemble des valeurs de p sur U c'est à dire l'ensemble $p(U)$. Cette propriété est illustrée par la Figure 3.2.

3.3 Relation entre les polynômes de Bernstein et le principe de floraison

Nous allons mettre en évidence une relation entre les polynômes de Bernstein et forme polaire (voir par exemple [143] dans le cas uni-varié). Nous décrivons ce

lien dans le cas multi-varié en introduisant une relation d'équivalence permettant de simplifier l'analyse et qui sera d'un grand intérêt dans la suite de la thèse.

3.3.1 Relation d'équivalence associée à une forme polaire

Commençons tout d'abord par définir la notion de relation d'équivalence associée à une forme polaire.
Soit $p : \mathbb{R}^n \to \mathbb{R}$ un polynôme multi-varié de degré $\delta = (\delta_1, \ldots, \delta_n)$, et notons par $q_\delta : \mathbb{R}^{\delta_1+\cdots+\delta_n} \to \mathbb{R}$ sa forme polaire relative à δ. Pour $z, z' \in \mathbb{R}^{\delta_1+\cdots+\delta_n}$, avec $z = (z_{1,1}, \ldots, z_{1,\delta_1}, \ldots, z_{n,1}, \ldots, z_{n,\delta_n})$ et $z' = (z'_{1,1}, \ldots, z'_{1,\delta_1}, \ldots, z'_{n,1}, \ldots, z'_{n,\delta_n})$, on note $z \cong z'$ si, pour tout $k = 1, \ldots, n$, il existe une permutation π_k telle que $(z_{k,1}, \ldots, z_{k,\delta_k}) = \pi_k(z'_{k,1}, \ldots, z'_{k,\delta_k})$.
Il est bien évident que $\cong$ est une relation d'équivalence.

Maintenant, soit $R = \prod_{k=1}^{k=n}[\underline{x}_k, \overline{x}_k]$ un rectangle de $\mathbb{R}^n$ et soit $R' = \prod_{k=1}^{k=n}[\underline{x}_k, \overline{x}_k]^{\delta_k}$ le rectangle associé à $\mathbb{R}^{\delta_1+\cdots+\delta_n}$ dont l'ensemble de sommets est $V' = \prod_{k=1}^{k=n}\{\underline{x}_k, \overline{x}_k\}^{\delta_k}$. Pour $v = (v_{1,1}, \ldots, v_{1,\delta_1}, \ldots, v_{n,1}, \ldots, v_{n,\delta_n}) \in V'$ et $k \in \{1, \ldots, n\}$, $l_k(v)$ désigne le nombre d'éléments $v_{k,1}, \ldots, v_{k,\delta_k}$ qui sont tous égaux à $\overline{x}_k$. Il est alors facile de vérifier que pour $v, v' \in V'$, $v \cong v'$ si et seulement $l_k(v) = l_k(v')$ pour tout $k \in \{1, \ldots, n\}$.
On note $\overline{V}' = (V'/\cong)$ l'ensemble des classes d'équivalence de la relation $\cong$ dans l'ensemble V', par conséquent $\overline{V}'$ a $(\delta_1 + 1) \times \cdots \times (\delta_n + 1)$ éléments.

D'après la discussion précédente, pour $\overline{v} \in \overline{V}'$ et $k \in \{1, \ldots, n\}$ on peut définir $l_k(\overline{v})$ (car $l_k(v) = l_k(v')$ pour tout $v, v' \in \overline{v}$). De même, d'après la propriété de symétrie de la forme polaire, on peut bien définir $q(\overline{v})$ (puisque $q(v) = q(v')$ pour tout $v, v' \in \overline{v}$).

Exemple 5. *Pour bien comprendre ces notations nous allons les détailler dans un cas particulier simple.*
On prend $n = 2$, $\delta_1 = 3$, $\delta_2 = 1$ *et* $R = [0, 1] \times [-1, 1]$.
Le nouveau rectangle R' *sera* $[0, 1]^3 \times [-1, 1]$ *et l'ensemble* $\overline{V}'$ *va donc avoir* $(3+1) \times (1 + 1) = 8$ *éléments. Plus précisément,*

$\overline{V}' = \{\overline{(0, 0, 0, -1)}, \overline{(0, 0, 0, 1)}, \overline{(1, 0, 0, -1)}, \overline{(1, 0, 0, 1)}, \overline{(1, 1, 0, -1)}, \overline{(1, 1, 0, 1)},$

$\overline{(1, 1, 1, -1)}, \overline{(1, 1, 1, 1)}\}$. *Par exemple :*

$$\overline{(0, 0, 0, -1)} = \{(0, 0, 0, -1)\}.$$
$$\overline{(1, 0, 0, -1)} = \{(1, 0, 0, -1), (0, 1, 0, -1), (0, 0, 1, -1)\}.$$

3.3.2 Relation entre forme de Bernstein et forme polaire

Soit $p : \mathbb{R}^n \to \mathbb{R}$ un polynôme multi-varié de degré $\delta = (\delta_1, \ldots, \delta_n)$ et soit $q_\delta : \mathbb{R}^{\delta_1+\cdots+\delta_n} \to \mathbb{R}$ sa forme polaire associée relative à δ.
On commence par se placer dans la boîte unité U de $\mathbb{R}^n$ (c'est à dire notre rectangle est $R = U$).

Soit U' la boite unité associé à $\mathbb{R}^{\delta_1+\cdots+\delta_n}$ et V' son ensemble de sommets.
Notons par $\{p_I \in \mathbb{R}\ ,\ I \leq \delta\}$ l'ensemble des coefficients du polynôme p dans la base canonique et rappelons que dans la base de Bernstein p s'écrit sous la forme :

$$p(x) = \sum_{I \leq \delta} b_{I,\delta} B_{I,\delta}(x) \tag{3.13}$$

où les coefficients de Bernstein $b_I = b_{I,\delta}$ sont donnés par (3.6) et les polynômes de Bernstein $B_{I,\delta}(x)$ sont donnés par (3.4) pour tout $I \leq \delta$.
La relation entre les coefficients de Bernstein associés à p et sa forme polaire q_δ est donnée par la Proposition suivante :

Proposition 8. *Les coefficients de Bernstein associés au polynôme p sont donnés par les valeurs au sommets $\overline{V'}$ de sa forme polaire. En effet, on a l'égalité suivante :*

$$q_\delta(\overline{v_{I,\delta}}) = b_{I,\delta} \text{ pour tout } I \leq \delta \text{ , avec :}$$

$$\overline{v_{I,\delta}} = \{v = (v_{1,1}, \ldots, v_{1,\delta_1}, \ldots, v_{n,1}, \ldots, v_{1,\delta_n}) \in V' \quad | \quad l_k(v) = i_k \text{ avec } k = 1 \ldots n\}.$$

Plus précisément, on peut écrire (3.13) sous la forme suivante :

$$p(x) = \sum_{I \leq \delta} q_\delta(\overline{v_{I,\delta}}) B_{I,\delta}(x). \tag{3.14}$$

Démonstration. Soient $z = (z_1, \ldots, z_n) \in \mathbb{R}^{\delta_1+\cdots+\delta_n}$, $\overline{v_{I,\delta}} = (v_1(i_1, \delta_1), \ldots, v_n(i_n, \delta_n))$ avec pour tout $k \in \{1, \ldots, n\}$, $z_k, v_k(i_k, \delta_k) \in \mathbb{R}^{\delta_k}$.
On sait que :

$$q_\delta(z) = \sum_{I \leq \delta} p_I \mathcal{B}_{I,\delta}(z) \text{ avec } \mathcal{B}_{\delta,I}(z) = \mathcal{B}_{\delta_1,i_1}(z_1) \ldots \mathcal{B}_{\delta_n,i_n}(z_n).$$

Ainsi pour $z = \overline{v_{I,\delta}}$ on aura :

$$q_\delta(\overline{v_{I,\delta}}) = \sum_{I \leq \delta} p_I \mathcal{B}_{I,\delta}(\overline{v_{I,\delta}}).$$

Montrons dans un premier temps que : $q_\delta(\overline{v_{I,\delta}}) = \sum_{J \leq I} p_J \mathcal{B}_{J,\delta}(\overline{v_{I,\delta}})$.
Pour cela on commence par écrire $q_\delta(\overline{v_{I,\delta}}) = A_1 + A_2$ avec :

$$A_1 = \sum_{J \leq I} p_J \mathcal{B}_{J,\delta}(\overline{v_{I,\delta}}) \text{ et } A_2 = \sum_{J \not\leq I,\ J \leq \delta} p_J \mathcal{B}_{J,\delta}(\overline{v_{I,\delta}}).$$

Il suffit de prouver que $A_2 = 0$.
En effet, si $J \not\leq I$ et $J \leq \delta$ on aura $\mathcal{B}_{J,\delta}(\overline{v_{I,\delta}}) = 0$ car :
$J \not\leq I$ et $J \leq \delta \iff$ il existe $k_0 \in \{1, \ldots, n\}$ tel q ue $i_{k_0} < j_{k_0} \leq \delta_{k_0}$.
Par conséquent : $\mathcal{B}_{j_{k_0},\delta_{k_0}}(v_{k_0}) = 0$ par définition de $\mathcal{B}_{j_{k_0},\delta_{k_0}}$ puisque $l_{k_0}(v) = i_{k_0} < j_{k_0}$.
Ainsi, on en déduit que $A_2 = 0$.

Maintenant pour $J \leq I$, montrons que : $\mathcal{B}_{J,\delta}(\overline{v_{I,\delta}}) = \dfrac{\binom{I}{J}}{\binom{\delta}{J}}$.

En effet, pour tout $k = 1, \ldots, n$, on a :

$$\mathcal{B}_{j_k,\delta_k}(v_k) = \frac{1}{\binom{\delta_k}{j_k}} \sum_{\sigma \in C(j_k,\delta_k)} v_{k,\sigma_1} \ldots v_{k,\sigma_{j_k}} = \frac{\binom{i_k}{j_k}}{\binom{\delta_k}{j_k}}.$$

Ainsi

$$\begin{aligned} \mathcal{B}_{J,\delta}(\overline{v_{I,\delta}}) &= \mathcal{B}_{j_1,\delta_1}(v_1) \ldots \mathcal{B}_{j_n,\delta_1}(v_n) \\ &= \frac{\binom{i_1}{j_1} \cdots \binom{i_n}{j_n}}{\binom{\delta_1}{j_1} \cdots \binom{\delta_n}{j_n}} = \frac{\binom{I}{J}}{\binom{\delta}{J}}. \end{aligned}$$

On conclut alors que :

$$q_\delta(\overline{v_{I,\delta}}) = \sum_{J \leq I} p_J \frac{\binom{I}{J}}{\binom{\delta}{J}} = b_{I,\delta}.$$

□

Remarque 4. *On peut remarquer que le degré δ apparaît comme indice dans la forme polaire ainsi que dans les coefficients de Bernstein associés à p. La raison vient du fait qu'un polynôme de degré δ peut être considéré, comme un polynôme de degré supérieur δ'. Dans ce cas, on aura une forme polaire différente et des coefficients de Bernstein différents mais le résultat de la Proposition 8 reste toujours vrai à condition que le même degré δ' soit fixé pour la forme polaire et les polynômes de Bernstein.*

La généralisation de ce résultat dans le cas d'un rectangle quelconque $R = \prod_{k=1}^{k=n} [\underline{x}_k, \overline{x}_k]$ est donnée au moyen de la proposition suivante :

Proposition 9. *Les valeurs $q_\delta(\overline{v})$ pour $\overline{v} \in \overline{V}'$ sont les coordonnées du polynôme p dans la base de Bernstein :*

$$p(x) = \sum_{\overline{v} \in \overline{V}'} q_\delta(\overline{v}) B_{l(v),\delta} \left(\frac{x - \underline{x}}{\overline{x} - \underline{x}} \right)$$

où $l(v) = (l_1(v), l_2(v), \ldots, l_n(v))$, $\underline{x} = (\underline{x_1}, \ldots, \underline{x_n})$ et $\overline{x} = (\overline{x_1}, \ldots, \overline{x_n})$.

Démonstration. La preuve découle immédiatement du fait que le changement de variable $y = h(x)$, où la fonction affine $h : \mathbb{R}^n \to \mathbb{R}^n$ est donnée par

$$h_k(x) = h_k(x_k) = \frac{x_k - \underline{x_k}}{\overline{x_k} - \underline{x_k}} \text{ pour tout } k = 1, \ldots, n,$$

envoie la boîte unité U sur le rectangle $R = \prod_{k=1}^{k=n} [\underline{x}_k, \overline{x}_k]$. □

Deuxième partie

Optimisation polynomiale

Chapitre 4

Problème d'optimisation polynomial (POP) : cadre général

4.1 Introduction

L'optimisation est la recherche des valeurs minimales ou maximales d'une fonction dans une région définie par des contraintes. Historiquement, les premières techniques d'optimisation été appliquées à des problèmes de logistique de personnel ainsi que des problèmes de gestion des systèmes de transports. Typiquement, ces problèmes ont été modélisés sous forme d'une minimisation d'une fonction coût linéaire avec des contraintes linéaires : c'est un problème d'optimisation linéaire qu'on note LP. En 1948, le mathématicien américain George Dantzig a inventé la programmation linéaire qui a permis de résoudre ce genre de problèmes grâce au fameux algorithme du simplexe [54]. Ce fut la naissance de ce qu'on appelle la programmation mathématique. La complexité de cet algorithme croît dans le pire des cas exponentiellement avec le nombre des variables du problème. Par ailleurs, l'algorithme du simplexe est extrêmement efficace dans la plupart des cas pratiques. Quelques années après, une autre technique pour la résolution des problèmes d'optimisation linéaires apparaît. Elle est donnée par ce qu'on appelle l'algorithme de l'ellipsoïde. Théoriquement, la complexité de la méthode est juste polynomiale. Mais en pratique, elle n'est pas aussi efficace que la méthode du simplexe, et elle est donc rarement utilisée.

Cependant, dans la plupart des applications, on est souvent face à des problèmes d'optimisation non linéaire. En réalité, mesurer la difficulté d'un problème d'optimisation n'est pas une question de linéarité/non linéarité mais plutôt une question de convexité/non convexité du problème. Un problème est dit convexe s'il s'agit d'une minimisation (respectivement maximisation) d'une fonction convexe (respectivement concave) dans une région convexe (c'est à dire l'ensemble des points admissibles est convexe). Ceci est dû au résultat fondamental d'analyse convexe qui affirme qu'une solution locale d'un problème convexe est aussi une solution globale. Autrement dit, si notre problème d'optimisation est convexe il suffit de trouver une solution locale pour qu'il soit résolu. D'un point de vue algorithmique, l'importance de ce résultat tient au fait que la plupart des algorithmes testent à chaque itération si la solution courante est localement optimale (c'est à dire solution pour un certain voisinage

prédéfini), donc si le problème est convexe, on pourra effectivement garantir l'optimalité globale de la solution. Par contre, les problèmes non convexes peuvent avoir plusieurs minima locaux et on ne dispose généralement pas d'un moyen permettant de dire si un minimum calculé est bien le minimum global qu'on cherche. Une classe importante de problèmes d'optimisation globale consiste à étudier des cas où la fonction objectif ou les contraintes (ou les deux) sont données par des polynômes. C'est le problème d'optimisation polynomiale qu'on notera POP.

La formulation générale du problème POP est la suivante :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in \mathbb{R}^n \\ & g_i(x) \leq 0 \ , \ i = 1, \ldots, m \\ & h_i(x) = 0 \ , \ j = 1, \ldots, s, \end{array} \tag{4.1}$$

où $p, g_1, \ldots, g_m, h_1, \ldots, h_s$ sont tous des polynômes multi-variés.
Sans perte de généralité, on peut se restreindre au problème suivant :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in \mathbb{R}^n \\ & g_i(x) \leq 0 \ , \ i = 1, \ldots, m, \end{array} \tag{4.2}$$

où $p, g_1, \ldots, g_m$ sont tous des polynômes multi-variés.

En effet une contrainte égalité $h_j(x) = 0$ n'est autre que deux contraintes inégalités $h_j(x) \leq 0$ et $-h_j(x) \leq 0$.
L'importance de cette classe provient de son large spectre d'applications. En effet, de nombreux problèmes de robotique, de biologie ou d'ingénierie peuvent être formulés en POP (voir par exemple [184, 122]). De plus, plusieurs problèmes fondamentaux d'analyse et de synthèse en théorie de contrôle robuste et non linéaire peuvent être résolus via des POP [59]. Toute une partie de cette thèse sera dédiée à l'étude de quelques unes de ces applications (problème d'invariance, d'atteignabilité, synthèse d'un contrôleur).
D'autre part, la classe des POP couvre comme cas particuliers plusieurs types de problèmes d'optimisation bien connus. On peut citer par exemple les problèmes entiers 0-1. En effet, une contrainte binaire $x_i \in \{0, 1\}$ peut se traduire par deux contraintes polynomiales $x_i^2 - x_i \leq 0$ et $x_i - x_i^2 \leq 0$. Une procédure de programmation linéaire dite "lift-and-project" a été proposée par Balas [8] pour résoudre ce genre de problème. Aussi, les problèmes d'optimisation quadratiques (convexes et non convexes) représentent un cas particulier important des POP où beaucoup d'approches théoriques de résolution sont établies se basant sur les méthodes du gradient et du simplexe ainsi que sur les méthodes de points intérieurs. Finalement, un autre cas particulier des POP qui sera étudié par la suite, est le cas d'un problème avec des inégalités matricielles linéaires (LMI) ou un problème avec des inégalités matricielles bilinéaires (BMI).

Résoudre un POP représente souvent une grande difficulté vu qu'il s'agit d'un problème non linéaire et généralement non convexe. Les premières techniques conçues pour le résoudre sont données par les méthodes dites algébriques dont les algorithmes diffèrent radicalement de ceux donnés dans le cas LP. Théoriquement, ces méthodes

fournissent la solution exacte de notre POP mais vu le coût des algorithmes utilisés, elles ne sont pas d'un grand intérêt pratique. Une alternative sera de se ramener à la résolution d'une relaxation du problème initiale dont la résolution est beaucoup plus simple. On distinguera deux types de relaxations : une relaxation linéaire consistant à résoudre un programme linéaire qu'on note LP et une relaxation semi-définie consistant à résoudre un programme semi-défini qu'on note SDP. A l'aide de ces relaxations on obtient souvent des bornes permettant d'approcher notre solution optimale.

Nous nous intéresserons par la suite aux différentes méthodes existantes pour la résolution d'un POP.

4.2 Méthodes de résolution algébriques du POP

Dans cette section, nous allons décrire quelques méthodes algébriques utilisées pour la résolution d'un POP [42].
A la base, ces méthodes permettent de résoudre un système d'équations polynomiales et de calculer explicitement ses racines. Par conséquent on va tout d'abord expliquer comment la résolution d'un POP, et plus précisément le calcul des minima locaux, peut se réduire à une résolution d'un système d'équations polynomiales puis on décrira brièvement les principales méthodes algébriques permettant la résolution de ce système. L'inconvénient de ces méthodes est leur coût élevé puisqu'on calcule tous les minima locaux pour pouvoir déterminer le minimum global.

4.2.1 Réduction en une résolution d'un système d'équations polynomiales

Pour résoudre un problème d'optimisation, une approche classique est de trouver des conditions nécessaires et/ou suffisantes permettant de traiter un problème plus simple. Notons L le lagrangien du problème (4.2) défini comme suit :

$$L(x,\lambda) = p(x) + \sum_{i=1}^{m} \lambda_i g_i(x) \ , \ \lambda \in \mathbb{R}^m \text{ tel que } \lambda_i \geq 0 \quad \forall i = 1,\dots,m.$$

Les conditions de K-K-T (Karush-Kuhn-Tucker) fournissent des conditions nécessaires pour trouver des minima locaux. Elles s'écrivent :

$$(K-K-T) \begin{cases} \nabla_x p(x) + \sum_{i=1}^{m} \lambda_i \nabla_x g_i(x) &= 0, \\ \lambda_i g_i(x) &= 0 \ , \ i = 1,\dots,m \\ g_i(x) &\leq 0 \ , \ i = 1,\dots,m \\ \lambda_i &\geq 0 \ , \ i = 1,\dots,m. \end{cases} \tag{4.3}$$

La première ligne de (4.3) désigne les conditions d'optimalité du premier ordre. La seconde correspond à des conditions de complémentarité. Les contraintes d'inégalité sont données par les conditions d'admissibilité et de positivité des multiplicateurs

de Lagrange.
Dans le cas d'un problème convexe on peut montrer que ces conditions sont nécessaires et suffisantes pour l'existence de minima locaux. Dans le cas général (non convexe comme le nôtre), on doit en plus ajouter d'autres conditions dites de régularité ou qualification des contraintes (voir par exemple [12]). Ainsi, on peut toujours ramener la résolution de notre POP à la résolution d'un système d'équations polynomiales avec contraintes d'inégalités. Donc si on dispose d'un moyen permettant la résolution d'un système d'équations polynomiales (qui est un problème difficile), on pourra trouver un ensemble de points dont on ne gardera que ceux vérifiant les contraintes d'inégalité, et trouver ainsi notre ensemble de minima locaux. Enfin, par l'évaluation de la fonction objectif en ces minima locaux, on trouve l'optimum ainsi que la valeur optimale (parfois des conditions du second ordre doivent être vérifiées en ce point pour affirmer que c'est bien un minimum).

La résolution d'un système d'équations polynomiales apparaît donc comme étape clef dans le contexte de l'optimisation polynomiale.

4.2.2 Méthodes de résolution

On se fixe un vecteur polynomial $P : \mathbb{R}^n \to \mathbb{R}^m$ avec $P = (p_1, p_2, \ldots, p_m)$, où $p_1, p_2, \ldots, p_m$ sont des polynômes multi-variés et on se propose de résoudre $P(x) = 0$ c'est à dire le système d'équations suivant :

$$(S_m) \begin{cases} p_1(x_1, \ldots, x_n) = 0, \\ p_2(x_1, \ldots, x_n) = 0, \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ p_m(x_1, \ldots, x_n) = 0. \end{cases}$$

Pour résoudre un système d'équations polynomiales, plusieurs méthodes algébriques ont été développées. On va décrire brièvement quelques unes et citer des références pour une description plus détaillée.

Résolution à l'aide des bases de Gröbner

La notion de bases de Gröbner a été introduite dans les années 1960 par Bruno Buchberger, qui lui a donné le nom de son directeur de thèse Wolfgang Gröbner.
En effet, Buchberger a développé dans sa thèse un algorithme (voir Théorème principal de la théorie algorithmique des bases de Gröbner [33]) permettant de trouver à partir d'un ensemble de polynômes $P = (p_1, \ldots, p_m)$ une base $GB(P)$ dite de Gröbner. L'intérêt de cette base provient des deux propriétés suivantes :

1. Les systèmes d'équations $V(P)$ [1] et $V(GB(P))$ admettent exactement le même ensemble de solutions.

1. $V(P) = V(p_1, \ldots, p_m) = \{x \in \mathbb{R}^n | p_1(x) = p_2(x) = \cdots = p_m(x) = 0\}$

2. La propriété d'élimination (avec respect de l'ordre lexicographique $\prec$) : c'est la généralisation du procédé d'élimination de Gauss pour les systèmes linéaires aux systèmes polynomiaux [34].

Ainsi, pour résoudre un système polynomial $V(P)$, on suit les étapes suivantes : D'après la propriété (1), il suffit de résoudre $GB(P)$ donc on commence par calculer $GB(P)$. Ensuite, on résout $GB(P)$ grâce à la propriété (2) d'élimination puisque le nouveau système sera triangulaire. En effet, on aura un ou plusieurs polynômes à une seule variable (disons x_1) à considérer puis un ou plusieurs polynômes à deux variables (disons x_1 et x_2), donc en substituant par les valeurs possibles de x_1 on se ramène au cas d'un polynôme d'une seule variable, et ainsi de suite. Pour plus de détails, on peut aussi consulter les références [38, 84] décrivant la résolution d'un POP à l'aide des bases de Gröbner.

Méthodes des valeurs propres

La résolution d'un système d'équations polynomiales peut se faire aussi en le ramenant à une forme matricielle et réduire la recherche des solutions à une recherche de valeurs propres. Dans le cas uni-varié, les solutions correspondent aux valeurs propres de la matrice adjointe de Frobenius correspondante (c'est exactement ce qu'utilise la commande "roots" en Matlab pour calculer les racines). La généralisation dans le cas multi-varié a été étudié par Stetter et Möller [124, 168].
Par ailleurs, la formulation d'un système d'équations polynomiales (S_m) en un problème aux valeurs propres n'est pas du tout triviale. Plusieurs approches pour le faire ont été décrites [106, 119, 94].

Une méthode consiste à construire une base de monômes bien particulière vérifiant une propriété de fermeture pour la multiplication par tous les x_i [124] (on peut utiliser les bases de Gröbner pour la construire). Ensuite, on construit la matrice de multiplication associée à cette base [168, 169] et on aura ainsi une forme matricielle du problème S_m. Enfin, il suffit de résoudre le problème aux valeurs propres équivalent pour pouvoir obtenir les coordonnées ${x_i}^*$ des racines x^* comme valeurs propres associées.

Pour une illustration, un exemple simple avec $n = m = 2$ est proposé dans [60].

Méthode d'homotopie numérique

Une autre méthode permettant de résoudre le système d'équations S_m est la méthode d'homotopie numérique [110, 183]. L'idée consiste à définir un système facile à résoudre $Q(x) = 0$ où $Q = (q_1, q_2, \ldots, q_m)$ puis suivre les courbes dans la variable réelle t permettant d'avoir l'ensemble de solutions suivant :

$$H(x,t) = (1-t)Q(x) + tP(x) = 0.$$

Ainsi, si l'ensemble $Q(x) = 0$ est bien choisi on aura les propriétés suivantes :

1. Les solutions de $Q(x) = 0$ sont explicitement connues.
2. L'ensemble solution de $H(x,t) = 0$ pour $0 \leq t < 1$ consiste en un nombre fini de courbes lisses, chacune paramétrée par $t \in [0,1)$.

3. Toute solution isolée de l'ensemble $H(x,1) = P(x)$ peut être atteinte par une certaine courbe qui commence en une solution de $H(x,0) = Q(x)$.

Par conséquent, les courbes solutions de $H(x,t) = (1-t)Q(x) + tP(x) = 0$ peuvent être suivies à partir des points initiaux ($t = 0$) connus grâce à la propriété (1) jusqu'à l'instant $t = 1$ permettant de trouver les solutions de $H(x,1) = P(x)$ en utilisant des méthodes numériques standard [4, 3].

4.3 Méthodes de relaxation du POP

Les méthodes algébriques sont des méthodes exactes calculant avec une précision arbitraire les solutions d'un POP .Leur inconvénient est le coût des algorithmes utilisés pour pouvoir résoudre un système d'équations polynomiales puis calculer la valeur optimale. Cependant, il existe des méthodes dites méthodes de relaxation permettant d'approcher la valeur optimale d'un POP d'une manière raisonnable. L'idée consiste à se ramener d'un problème non convexe (et donc difficile à résoudre) à une relaxation convexe de dimension supérieure (variables et contraintes supplémentaires) qu'on pourra résoudre efficacement grâce à des algorithmes bien connus et dont le coût est beaucoup moins élevé que ceux utilisés par les méthodes algébriques. Généralement, il y a toujours des moyens nous permettant d'augmenter la précision de ces relaxations (monter en dimension ou découper l'ensemble admissible) et des résultats de convergences vers la valeur optimale peuvent être établis.

Dans cette partie on décrira deux méthodes de relaxation : la première est une relaxation LP où on se ramène à un programme linéaire, la deuxième est basée sur la programmation semi-définie (dite relaxation SDP).

Dans ce qui suit, on supposera connu les notions classiques d'optimisation linéaire (admissibilité,dualité,...). Comme revue le lecteur peut consulter [32].
Pour des raisons d'admissibilité, on suppose dans toute la suite que l'ensemble $\mathcal{K} = \{x \in \mathbb{R}^n \mid g_k(x) \leq 0, \quad k = 1, \ldots, m\}$ est un ensemble compact.

4.3.1 Relaxation LP : technique de reformulation et linéarisation

Une relaxation est dite LP si elle permet de transformer le problème initial en un problème linéaire qui n'est pas forcément équivalent mais qui nous permet d'obtenir une borne (inférieure dans le cas d'une minimisation) à la valeur optimale de notre problème initial.

La programmation linéaire LP :

La programmation linéaire [177] est la résolution d'un problème d'optimisation ayant la forme suivante :

$$\begin{array}{ll} \text{minimiser} & p \cdot x \\ \text{s.c} & x \in \mathbb{R}^n \\ & Ax \leq b, \end{array} \tag{4.4}$$

où $p \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ et $b \in \mathbb{R}^m$.

Le problème (9.3) est dit linéaire et sa résolution peut se faire, comme on l'a déjà souligné dans l'introduction de ce chapitre, à l'aide de l'algorithme du simplexe ou celui de l'ellipsoïde .
Ainsi, plusieurs solveurs existent permettant la résolution d'une manière efficace de ce type de problèmes comme par exemple Linpro (en scilab), Linprog (en matlab) ainsi que LP-SOLVE (en C++).

La technique de reformulation et linéarisation (RLT)

La technique de reformulation et linéarisation (RLT) a été introduite par Sherali dans [162, 163]. Elle propose un algorithme basé sur la programmation linéaire permettant de donner une borne inférieure à la valeur optimale p^* du POP qui est souvent écrit sous la forme suivante :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in R \\ & g_i(x) \leq 0 \ , \ i = 1, \ldots, m, \end{array} \tag{4.5}$$

où $R = \prod_{k=1}^{k=n} [\underline{x}_k, \overline{x}_k]$ est un rectangle de $\mathbb{R}^n$.

Notons que notre POP (4.2) peut s'écrire sous cette forme étant donné que l'ensemble $\mathcal{K}$ est borné. En effet, les problèmes (4.2) et (4.5) sont équivalents pour un rectangle R suffisamment grand. Par conséquent, la contrainte $x \in R$ peut être incorporée dans les g_i tout en ayant $2n$ contraintes supplémentaires.
Comme son nom l'indique, l'idée principale derrière cette méthode est de reformuler le problème (4.5) en un problème linéaire qu'on pourra résoudre efficacement.

En effet, on fixe tout d'abord un degré δ (δ est souvent pris comme étant le degré maximal des polynômes $p, g_1, \ldots, g_m$). Pour chaque monôme x^α de degré inférieur ou égal à δ, on associe une nouvelle variable y_α. En incorporant ces variables dans notre polynôme p (notre fonction objective), on va pouvoir éliminer toutes ses non linéarités et obtenir ainsi une version linéarisée.

Pour les contraintes, on construit un nouvel ensemble de contraintes (linéaires) en linéarisant comme précédemment tous les produits possibles des anciennes contraintes par respect du degré δ (c'est à dire que les nouvelles contraintes qu'on va linéariser doivent toutes avoir un degré inférieur ou égal à δ). En exploitant ces produits, des relations entre les nouvelles variables s'établissent et permettent de récupérer des informations qu'on perd à cause de la linéarisation. Par exemple, dans le cas où les contraintes sont affines, la relaxation d'ordre δ est donnée comme suit :

$$\begin{array}{rll} {p_\delta}^* = & \text{minimiser} & [p(x)]_L \\ & \text{s.c} & 0 \leq [g_1(x)^{\alpha_1} \ldots, g_m(x)^{\alpha_m}]_L, \quad |\alpha| \leq \delta. \end{array} \tag{4.6}$$

où $[f(x)]_L$ désigne la linéarisation décrite précédemment d'un polynôme f.
Intuitivement, plus le degré δ est grand plus on a de contraintes produits, plus on récupère d'informations perdues et donc plus le résultat est précis. En effet, si on note par ${p_\delta}^*$ la valeur optimale du programme linéaire donné par la méthode (RLT), on montre que la suite ${p_\delta}^*$ est croissante en δ et que ${p_K}^* \leq p^*$ pour tout degré $K \geq \delta$.

Remarque 5. *Si on fixe un degré δ , on peut montrer (voir [162]) que les contraintes générées par la méthode (RLT) pour un degré δ' avec $1 \leq \delta' < \delta$ sont toutes impliquées par les contraintes générées par la méthode pour le degré δ. Ainsi, pour un degré δ fixé, il suffit de considérer les contraintes de degré δ. Par conséquent, le programme (4.6) devient :*

$$\begin{array}{lll} p_\delta{}^* = & \textit{minimiser} & [p(x)]_L \\ & s.c & 0 \leq [g_1(x)^{\alpha_1} \ldots, g_m(x)^{\alpha_m}]_L, \quad |\alpha| = \delta. \end{array} \tag{4.7}$$

On donnera un exemple illustratif permettant de comprendre cette approche.

Exemple 6. *On se propose de calculer une borne inférieure à la valeur optimale du POP suivant :*

$$\begin{array}{ll} \textit{minimiser} & x_1 + x_1x_2 + x_1^2 \\ s.c & (x_1, x_2) \in [0,1]^2 \\ & x_1^2 + x_2^2 - 1 \leq 0. \end{array}$$

En prenant $\delta = 2$ et en utilisant la méthode (RLT) on obtient le programme linéaire suivant :

$$\begin{array}{ll} \textit{minimiser} & y_{10} + y_{11} + y_{20} \\ s.c & -y_{10} \leq 0, \quad y_{10} - 1 \leq 0 \\ & -y_{01} \leq 0, \quad y_{01} - 1 \leq 0 \\ & -y_{20} \leq 0, \quad -1 - y_{20} + 2y_{10} \leq 0 \\ & -y_{02} \leq 0, \quad -1 - y_{02} + 2y_{01} \leq 0 \\ & -y_{11} \leq 0, \quad -y_{10} + y_{20} \leq 0 \\ & -y_{10} + y_{11} \leq 0, \quad -y_{01} + y_{11} \leq 0 \\ & -y_{10} + y_{02} \leq 0, \quad y_{20} + y_{02} - 1 \leq 0 \\ & -y_{10} - y_{01} + y_{11} + 1 \leq 0 \end{array}$$

où y_{ij} est la linéarisé de ${x_1}^i {x_2}^j$ avec $i, j \in \mathbb{N}$ vérifiant $i + j \leq 2$.

En effet, ici on a $m = 5$ contraintes qui sont :
$g_1(x) = -x_1$, $g_2(x) = x_1 - 1$, $g_3(x) = -x_2$, $g_4(x) = x_2 - 1$ et $g_5(x) = x_1^2 + x_2^2 - 1$. Comme la fonction g_5 n'est pas affine (polynomiale de degré $d = 2$) alors si on prend $\alpha_5 = 1$ il faut prendre $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 0$ et on obtient ainsi la contrainte $y_{20} + y_{02} - 1 \leq 0$. Si $\alpha_1 = \alpha_3 = \alpha_5 = 0$ et $\alpha_2 = \alpha_4 = 1$, on obtient la contrainte $y_{11} - y_{10} - y_{01} + 1 \leq 0$.

Maintenant grâce à la Remarque 5, il suffira de résoudre le programme linéaire suivant :

$$\begin{array}{ll} \textit{minimiser} & y_{10} + y_{11} + y_{20} \\ s.c & -y_{20} \leq 0, \quad -1 - y_{20} + 2y_{10} \leq 0 \\ & -y_{02} \leq 0, \quad -1 - y_{02} + 2y_{01} \leq 0 \\ & -y_{11} \leq 0, \quad -y_{10} + y_{20} \leq 0 \\ & -y_{10} + y_{11} \leq 0, \quad -y_{01} + y_{11} \leq 0 \\ & -y_{10} + y_{02} \leq 0, \quad y_{20} + y_{02} - 1 \leq 0 \\ & -y_{10} - y_{01} + y_{11} + 1 \leq 0. \end{array}$$

Résultats de convergence pour la relaxation RLT

On va étudier les résultats de convergence de la relaxation RLT. Ceci revient à étudier le comportement asymptotique de la suite $(p_\delta^*)_\delta$. En effet, ces résultats

permettent de juger la précision d'une relaxation (c'est à dire la différence $p^* - p_\delta^*$ entre la valeur optimale du POP initial et celle de la relaxation). Si cette différence tend vers zéro, la relaxation peut être aussi précise que l'on veut à condition de passer à une dimension supérieure, ce qui augmente le coût de l'algorithme vu qu'il impliquera la génération de plusieurs variables et contraintes supplémentaires.
On verra que pour les relaxations RLT, les résultats de convergence se démontrent [102] essentiellement par l'utilisation de la formulation duale de la relaxation et des résultats de positivité des polynômes décrits dans le premier chapitre de cette thèse.
On reprend le POP (4.5) avec $R = [0,1]^n$, c'est à dire le problème suivant :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in [0,1]^n \\ & g_i(x) \leq 0 \ , \ i = 1, \ldots, m. \end{array} \tag{4.8}$$

Pour simplifier, on va traiter le cas $n = 1$ sans contraintes (les fonctions g_i sont nulles), voir le lien avec les résultats du premier chapitre et décrire brièvement les résultats possibles pour le cas général (4.8) selon la nature des fonctions g_i et donc de l'ensemble $\mathcal{K}$. Ainsi, notons par p^* la valeur optimale du POP :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in [0,1]. \end{array} \tag{4.9}$$

Notons par p_δ^* la valeur optimale de la relaxation linéaire que propose la méthode RLT pour un degré $\delta \geq 1$ (δ peut être pris en un premier temps comme étant le degré du polynôme p). La relaxation RLT est donnée par :

$$\begin{array}{lll} {p_\delta}^* = & \text{minimiser} & [p(x)]_L \\ & \text{s.c} & 0 \leq [x^k(1-x)^m]_L, \quad \forall 0 \leq k+m \leq \delta. \end{array} \tag{4.10}$$

Proposition 10. *On a le résultat de convergence suivant :*

$$p_\delta^* \uparrow p^* \ \textit{quand} \ \delta \to \infty.$$

La notation $x_n \uparrow l$ signifie que la suite (x_n) est croissante et tend vers l quand n tend vers l'infini.

Démonstration. La preuve détaillée est dans [102] et repose essentiellement sur la formulation duale. Sans perte de généralité on suppose que $p(0) = 0$. Alors en passant au dual du problème (4.10) on trouve le programme linéaire équivalent :

$$\begin{array}{ll} \text{maximiser} & -\displaystyle\sum_{m=1}^{\delta} c_{0m} \\ \text{s.c} & c_{km} \geq 0, \\ & \displaystyle\sum_{k \leq i,\ k+m \geq i} (-1)^{i-k} \begin{pmatrix} m \\ i-k \end{pmatrix} c_{km} = p_i, \quad i = 1, \ldots, \delta. \end{array} \tag{4.11}$$

Maintenant, on va utiliser le résultat de représentation des polynômes positifs (2.1) où le polynôme $p(x)$ sera dans notre cas le polynôme $p(x) - p^*$ dont on connaît la

positivité sur $[0,1]$ (le résultat est donné dans le premier chapitre pour l'intervalle $[-1,1]$; pour $[0,1]$ il suffit de remplacer la contrainte $1+x \geq 0$ par $x \geq 0$).
Le problème est que le résultat nécessite que le polynôme soit strictement positif ; pour cela notre polynôme sera pris comme étant $p(x)-p^*+\epsilon$ pour un $\epsilon > 0$ arbitraire. Ainsi le résultat (2.1) implique l'existence d'un degré $\delta(\epsilon)$ tel que :

$$p(x) - p^* + \epsilon = \sum_{0 \leq k+m \leq \delta(\epsilon)} d_{k,m} x^k (1-x)^m, \quad d_{k,m} \geq 0. \tag{4.12}$$

En identifiant les termes ayant la même puissance de chaque côté de l'égalité (4.12) on obtient les relations suivantes :

$$\sum_{k \geq i; k+m \geq i} (-1)^{i-k} \left(\begin{array}{c} m \\ i-k \end{array} \right) d_{km} = p_i, \ i = 1, \ldots, \delta(\epsilon).$$

Et pour le terme constant on aura :

$$\sum_{m \leq \delta(\epsilon)} d_{0m} = -p^* + \epsilon.$$

Par conséquent, pour tout $\delta \geq \delta(\epsilon)$, la suite $(d_{k,m})$ est admissible pour (4.11) ayant comme valeur optimale $p^* - \epsilon$ et ceci pour tout $\epsilon > 0$. Le résultat est donc obtenu en faisant tendre ϵ vers zéro. □

Remarque 6. *En effet, les variables de décision c_{km} calculées par le problème dual, nous permettent d'obtenir une représentation du polynôme positif $p(x)-p^*+\epsilon$ sous la forme (4.12). Maintenant, si on note x^* la solution optimale associée ($p^* = p(x^*)$), il est évident que la relaxation ne peut pas être exacte (c'est à dire $\epsilon = 0$) si $x^* \notin \{0,1\}$ car dans ce cas $p(x) - p^*$ ne pourra pas avoir la représentation (4.12) puisque pour $x = x^*$ toutes les coefficients c_{km} seront nuls.*

Pour le cas avec contraintes, en utilisant la formulation duale ainsi que le résultat (2.4) de Handelmann [87], on montre que la convergence aura lieu pour le cas où $\mathcal{K}$ est un polytope (voir [102] pour plus de détails).
Si ces contraintes sont polynomiales, Lasserre [103] se basant sur un résultat de représentation de polynômes positifs sur un ensemble algébrique (Vasilescu [180]), montre que l'on pourra aussi converger vers p^* en utilisant la formulation duale ainsi que la représentation (2.4) où les g_i sont des polynômes vérifiant l'**Hypothèse 1** du premier chapitre.

Algorithme de "branch-and-bound" pour la relaxation RLT

Théoriquement, le fait d'augmenter le degré d'une relaxation est un moyen important permettant d'améliorer sa précision. Cependant, l'intérêt pratique de cette approche est réduit puisqu'une augmentation du degré entraîne l'ajout de plusieurs variables et contraintes supplémentaires, rendant ainsi le problème beaucoup plus complexe. Par ailleurs, un autre moyen, plus efficace, est donné par ce qu'on appelle les algorithmes de "branch-and-bound". L'idée d'un algorithme de "branch-and-bound" consiste à découper "intelligemment" le rectangle R pour pouvoir améliorer

la valeur p_δ^* de la relaxation c'est à dire le découper dans le but d'avoir une (des) sous boîte(s) dans laquelle la relaxation sera exacte ou avoir une (des) sous boîte(s) dont on peut affirmer que la solution optimale n'y appartient pas. En effet, la motivation principale est le fait que, en plus du degré, la précision de la relaxation dépend aussi de la taille de la boîte (voir [166] page 95) c'est à dire que plus la boîte est petite plus l'erreur $|p^* - p_\delta^*|$ diminue.

Considérons la relaxation RLT de degré δ associée au POP (4.5) (où δ est un entier positif) et notons respectivement par $(y_\alpha^*)_\delta$ et ${p_\delta}^*$ la solution ainsi que la valeur optimale de cette relaxation ($\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$ vérifiant $|\alpha| = \alpha_1 + \cdots + \alpha_n \leq \delta$). On utilisera aussi les notations :

$$
\begin{aligned}
{y_1}^* &= {y_\alpha}^* \text{ pour } \alpha = (1, 0, \ldots, 0) \\
{y_2}^* &= {y_\alpha}^* \text{ pour } \alpha = (0, 1, \ldots, 0) \\
&\cdots\cdots \\
{y_n}^* &= {y_\alpha}^* \text{ pour } \alpha = (0, 0, \ldots, 1).
\end{aligned}
$$

L'algorithme (présenté dans [162]) se base sur les deux résultats suivants :

Lemme 3. *Relaxation exacte :*
Supposons que ${y_{\alpha,\delta}}^* = ({y_1}^*)^{\alpha_1} \times \cdots \times ({y_n}^*)^{\alpha_n}$ *pour tout* $\alpha \in \mathbb{N}^n$ *tel que* $|\alpha| \leq \delta$.
Alors ${p_\delta}^* = p^*$, *c'est à dire que la relaxation RLT de degré* δ *est exacte.*

Lemme 4. *Stratégie de décomposition :*
On suppose qu'il existe un entier $k \in \{1, \ldots, n\}$ *tel que* ${y_k}^* = \underline{x}_k$ *alors :*

$$
{y_{\alpha(k,1),\delta}}^* = \underline{x}_k {y_{\alpha,\delta-1}}^* \text{ pour tout } \alpha \in \mathbb{N}^n \text{ vérifiant } |\alpha| \leq \delta - 1,
$$

avec $\alpha(k,1) = (\alpha_1, \ldots, \alpha_{k-1}, \alpha_k + 1, \alpha_{k+1}, \ldots, \alpha_n)$.
De façon similaire si ${y_k}^* = \overline{x}_k$ *alors :*

$$
{y_{\alpha(k,1),\delta}}^* = \overline{x}_k {y_{\alpha,\delta-1}}^* \text{ pour tout } \alpha \in \mathbb{N}^n \text{ vérifiant } |\alpha| \leq \delta - 1.
$$

La preuve de ces deux lemmes est donnée dans [162].
A chaque itération, l'algorithme teste si la relaxation est exacte dans chaque sous-boîte à l'aide du Lemme 3. Si c'est le cas, on sauvegarde le résultat, sinon on découpe la sous-boîte en utilisant le Lemme 4. En effet, la direction de décomposition r est donnée par :

$$
r \in \arg \max_{k \in \{1,\ldots,n\}} \theta_k \tag{4.13}
$$

avec

$$
\theta_k = \max_{t=1,\ldots,\delta-1} \max_{|\alpha|=t} |{y_{\alpha(k,1)}}^* - {y_k}^* {y_\alpha}^*| \text{ pour tout } k \in \{1, \ldots, n\}.
$$

Remarque 7. *Sherali montre dans [162] qu'avec cet algorithme ou bien on converge après un nombre fini d'itérations en trouvant la solution optimale ainsi que la valeur de (4.5), ou bien une suite infinie d'itérations est générée, telle qu'à travers toute branche infinie de l'arborescence, tout point d'accumulation résout (4.5).*

4.3.2 Relaxation SDP : méthode de Lasserre

Une relaxation est dite SDP ou programme semi-défini si le problème initial est ramené à un problème d'optimisation avec des inégalités matricielles linéaires qu'on note LMI.

La programmation semi définie SDP

La programmation semi-définie [178] peut être vue comme étant une extension de la programmation linéaire dans laquelle les inégalités vectorielles sont remplacées par des inégalités matricielles. Plus précisément, la programmation semi définie est la résolution d'un problème d'optimisation ayant la forme suivante :

$$\begin{array}{ll} \text{minimiser} & p \cdot x \\ \text{s.c} & x \in \mathbb{R}^n \\ & F_0 + \sum_{i=1}^{n} x_i F_i \succeq 0, \end{array} \tag{4.14}$$

où $p \in \mathbb{R}^n$ et les $F_i \in \mathbb{R}^{n \times m}$ sont des matrices symétriques pour tout $i = 1, \ldots, n$.

L'inégalité matricielle $F \succeq 0$ signifie que la matrice F est semi définie positive ; c'est à dire $z^\top F z \geq 0$ pour tout $z \in \mathbb{R}^m$.
L'importance de cette classe provient du fait qu'elle englobe une grande partie des problèmes d'optimisation convexe tel que les programmes linéaires (précédemment introduits) ainsi que les programmes quadratiques convexes.
Pour la résolution d'un programme semi-défini, on peut souligner le fait que la méthode de simplexe n'est pas applicable. Par ailleurs, des extensions de cette méthode pour le cas des programmes semi-définis ont été établies par Pataki [133] et Lasserre [100]. Par ailleurs, la méthode d'ellipsoïde de Nemirovsky [192] ou de Shor [164] permet de résoudre le programme en un temps polynomial. En matlab, les solveurs SEDUMI et YALMIP permettent la résolution de ce genre de programmes.

La programmation somme de carrés SOS

La programmation somme de carrés a été introduite par Parillo dans sa thèse [131]. Elle permet de manipuler des polynômes appartenant à $\Sigma[x]$ (voir Définition 2 du Chapitre 2). En effet, dans ce genre de programmation la condition de positivité des polynômes est relaxée en somme de carrés de polynômes qui est une condition suffisante mais pas nécessaire (voir Remarque 1 du Chapitre 2).
La forme générale d'un programme SOS est la suivante :

$$\begin{array}{ll} \text{minimiser} & c \cdot \alpha \\ \text{s.c} & \alpha \in \mathbb{R}^n \\ & f_{1,0}(x) + f_{1,1}(x)\alpha_1 + \cdots + f_{1,1}(x)\alpha_1 \in \Sigma[x] \\ & \cdots \\ & \cdots \\ & f_{N,0}(x) + f_{N,1}(x)\alpha_1 + \cdots + f_{N,1}(x)\alpha_1 \in \Sigma[x] \end{array} \tag{4.15}$$

où $c \in \mathbb{R}^n$ et $\{f_{j,k}\}$ sont des polynômes pour tout $j = 1, \ldots, N$ et tout $k = 1, \ldots, m$.

Pour la résolution on peut utiliser SOSTOOLS, YALMIP ou SOSOPT.

Relation entre SDP et SOS

La programmation SOS peut être aussi considérée comme une programmation SDP. En effet, tous les solveurs qu'on a cités pour la résolution d'un programme SOS procèdent d'abord à sa conversion en un programme SDP. L'idée est que si un polynôme est dans $\Sigma[x]$, sa représentation peut être calculée en utilisant la programmation semi-définie. Plus précisément on a le résultat suivant [31] :

Proposition 11. *Un polynôme p de degré $2d$ est un SOS si et seulement si il existe une matrice symétrique semi-définie positive vérifiant :*

$$p(x) = z_d(x)'Qz_d(x) \text{ pour tout } x \in \mathbb{R}^n$$

où $z_d(x) = (x^\alpha)_{|\alpha|\leq d} = (1, x_1, \ldots, x_n, {x_1}^2, x_1x_2, \ldots, x_{n-1}x_n, {x_n}^2, \ldots, {x_1}^d, \ldots, {x_n}^d)'$.

Ainsi pour déterminer la représentation d'un polynôme p dans $\Sigma[x]$, il suffit de trouver la matrice symétrique semi-définie positive Q et on se trouve ainsi dans le cadre de la programmation semi-définie.

La méthode de Lasserre (LMI)

En 2001, Lasserre a introduit une méthode se basant sur les inégalités matricielles linéaires (LMI) ainsi que la théorie des moments permettant de donner une borne inférieure à la valeur optimale p^* du POP (4.1) (voir [101]). Nous allons décrire brièvement cette approche.

Le POP qu'on veut résoudre admet une formulation équivalente basée sur la notion de mesures. Concrètement on peut montrer que :

$$p^* = \min_{\mu \in P(\mathcal{K})} \int p(x)\mu(dx)$$

où $P(\mathcal{K})$ est l'espace des mesures finis de Borel à support dans $\mathcal{K}$.

Comme dans l'approche RLT, on commence tout d'abord par fixer un degré maximal δ. A l'aide des notations introduites au premier chapitre sur les polynômes, on peut écrire :

$$p(x) = \sum_{|\alpha|\leq\delta} p_\alpha x^\alpha.$$

Ainsi, en permutant somme et intégrale on aura :

$$\int p(x)\mu(dx) = \sum_{|\alpha|\leq\delta} p_\alpha y_\alpha$$

où la suite $y := \{y_\alpha\}_{|\alpha|\leq\delta}$ des moments d'ordre δ est donnée par : $y_\alpha := \int x^\alpha d\mu_x$ avec $d\mu_x = \mu(dx)$ pour tout $|\alpha| \leq \delta$.

L'idée sera de remplacer le problème de minimisation sur les mesures $\mu \in P(\mathcal{K})$ par un problème de minimisation sur les moments y_α. On sait déjà que si une mesure μ est donnée, on peut construire la suite des moments associés $y := \{y_\alpha\}$ pour un ordre δ fixé. La réciproque n'est pas vrai.

En effet, soit $r = E(\frac{\delta}{2})^+$ où $E(x)^+$ désigne la partie entière supérieure de x (c'est à dire le plus petit entier supérieur ou égal à x). Par la théorie des moments une condition nécessaire pour que la suite y admette une mesure représentative μ_y est donnée par une condition de positivité de la matrice des moments d'ordre r qu'on note $M_r(y)$. Plus précisément si $M_r(y) \succeq 0$ alors il existe une mesure μ_y telle que $y_\alpha := \int x^\alpha d\mu_x$ pour tout α vérifiant $|\alpha| \leq \delta$.
En utilisant ce résultat et des résultat analogues permettant d'assurer que le support de la mesure μ_y est inclus dans $\mathcal{K}$, on aura la relaxation suivante :

$$\begin{array}{ll} \text{minimiser} & \displaystyle\sum_{|\alpha|\leq\delta} p_\alpha y_\alpha \\ \text{s.c} & M_r(y) \succeq 0 \\ & M_{r-v_k}(-g_k y) \succeq 0 \quad k = 1, \ldots, m, \end{array} \tag{4.16}$$

avec $v_k = E(\frac{\delta_k}{2})^+$ où δ_k désigne le degré maximal de g_k , pour tout $k = 1, \ldots, m$. La construction des matrices des moments $M_r(y), M_{r-v_1}(-g_1 y), \ldots, M_{r-v_m}(-g_m y))$ est bien expliquée dans [101].

Le problème (4.18) est bien une relaxation du POP (4.1), c'est à dire que si on note par ${p_\delta}^*$ sa solution optimale, on a ${p_\delta}^* \leq p^*$. Il est évident aussi que plus le degré δ est grand, plus la relaxation est précise.

Exemple 7. *On reconsidère l'exemple 6. La relaxation SDP pour le degré $\delta = 2$ est la suivante :*

$$\begin{array}{ll} \textit{minimiser} & y_{10} + y_{11} + y_{20} \\ s.c & \begin{pmatrix} 1 & y_{10} & y_{01} \\ y_{10} & y_{20} & y_{11} \\ y_{01} & y_{11} & y_{02} \end{pmatrix} \succeq 0 \\ & y_{10} - 1 \leq 0 \\ & -y_{10} \leq 0 \\ & y_{01} - 1 \leq 0 \\ & y_{20} + y_{02} - 1 \leq 0. \end{array}$$

Résultats de convergence pour la relaxation LMI

D'une façon analogue, on décrira brièvement les résultats de convergence des relaxations SDP données par la méthode LMI de Lasserre pour le problème (4.8) qui sont aussi décrites dans [102]. Les preuves sont basées sur les mêmes ingrédients (formulation duale et résultats de représentations de polynômes positifs). La seule différence sera la manière dont on représente un polynôme positif. En effet, comme on considère un programme semi-défini, le dual nous fournit une représentation des polynômes positifs comme somme de carrés de polynômes. Ainsi, pour pouvoir identifier les coefficients comme précédemment, on aura besoin d'utiliser les résultats de représentation donnés par des sommes de carrés de polynômes.
Par ailleurs, on a aussi des résultats d'exactitude pour certaines relaxations à partir d'un certain rang ; c'est à dire que l'on montre parfois l'existence d'un entier δ_0 tel que $p^* = p^*_\delta = p^*_{\delta_0}$ pour tout $\delta \geq \delta_0$.

On commence, comme dans RLT, par le cas unidimensionnel sans contraintes (4.9). On se fixe un degré δ (supérieur ou égal au degré de p) et on prend $r = E(\frac{\delta}{2})^+$.

Proposition 12. *Notons par $p_\delta{}^*$ la solution optimale de la relaxation :*

$$\begin{array}{ll} \textit{minimiser} & \sum_{i\leq\delta} p_\alpha y_i \\ s.c & M_r(y) \succeq 0 \\ & M_{r-1}(-g_k y) \succeq 0 \quad k = 1, 2, \end{array} \tag{4.17}$$

avec $g_1(x) = x$ et $g_2(x) = 1 - x$.
Alors $p^* = p_\delta{}^*$ pour tout degré δ supérieur ou égal au degré de p (c'est à dire la relaxation est exacte pour le cas unidimensionnel sans contraintes).

Démonstration. La preuve est établie dans [101, 102]. En effet, une première façon de démontrer ce résultat est donnée par l'intermédiaire de la théorie des moments [43], où l'on dispose d'une condition nécessaire et suffisante assurant l'existence d'une mesure de probabilité à support dans $[0, 1]$. La deuxième façon consiste en l'utilisation de la représentation (2.2) ainsi que la formulation duale. La relaxation est exacte car la représentation (2.2) est valable pour tout polynôme positif (et pas uniquement pour les polynômes strictement positifs). □

Le cas avec contraintes donné par le problème (4.2) s'en déduit automatiquement en utilisant la généralisation de la représentation (2.2).

Proposition 13. *Notons maintenant $p_\delta{}^*$ la solution optimale de la relaxation :*

$$\begin{array}{ll} \textit{minimiser} & \sum_{|\alpha|\leq\delta} p_\alpha y_\alpha \\ s.c & y \in \mathbb{R}^{s(\delta)} \\ & M_r(y) \succeq 0 \\ & M_{r-v_k}(-g_k y) \succeq 0 \quad k = 1, \ldots, m. \end{array} \tag{4.18}$$

*Si l'ensemble $\mathcal{K}$ est semi-algébrique compact vérifiant l'***Hypothèse 2** *donnée dans le premier chapitre alors on a :*

$$p_\delta^* \uparrow p^* \text{ quand } \delta \to \infty.$$

Démonstration. La preuve est donnée dans [101]. Elle se base sur le résultat de Putinar [141] concernant la représentation de polynômes positifs sur des ensembles semi-algébriques. □

Remarque 8. *Plus précisément, la relaxation est exacte pour le cas unidimensionnel vu que tout polynôme positif est une somme de carrés de polynômes. Ce résultat n'est pas valable en dimension supérieure (voir Remarque 1 du Chapitre 2), où seulement un résultat de convergence asymptotique est établi.*

Chapitre 5

Relaxations linéaires des problèmes d'optimisation polynomiaux

Dans ce chapitre, nous allons construire deux nouvelles relaxations LP pour obtenir une borne inférieure au POP (4.2). Les deux principaux ingrédients pour la construction de ces relaxations seront les polynômes de Bernstein ainsi que le principe de floraison décrits dans la première partie de cette thèse. Notons que pour ces deux relaxations, on a besoin de fixer à l'avance un rectangle R, c'est à dire que notre ensemble admissible $\mathcal{K}$ sera donné par :

$$\mathcal{K} = \{x \in R \mid g_i(x) \leq 0, \quad i = 1, \ldots, m\} \text{ où } R = \prod_{k=1}^{k=n} [\underline{x}_k, \overline{x}_k] \text{ est un rectangle de } \mathbb{R}^n.$$

Ainsi comme dans la méthode RLT, notre POP sera donné par le problème

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in R \\ & g_i(x) \leq 0 \ , \ i = 1, \ldots, m, \end{array}$$

où $p, g_1, \ldots, g_m$ sont tous des polynômes multi-variés .

5.1 Relaxation d'un POP en utilisant la forme polaire

L'approche que nous allons proposer pour la construction de la relaxation est basée essentiellement sur le principe de floraison ou la forme polaire d'un polynôme. Dans cette section, l'ensemble $\mathcal{K}$ contiendra aussi des contraintes d'égalité ; c'est à dire il sera donné par :

$$\mathcal{K} = \{x \in R \mid g_i(x) \leq 0, \quad i = 1, \ldots, m \text{ et } h_j(x) = 0, \quad j = 1, \ldots, m\}.$$

L'intérêt de considérer ce cas plus général sera clarifié par la suite. On commencera par traiter le cas où l'ensemble $\mathcal{K}$ est un polytope puis on étudiera le cas général.

Enfin, notons $V = \prod_{k=1}^{n} \{\underline{x}_k, \overline{x}_k\}$ l'ensemble des sommets associés au rectangle $R = \prod_{k=1}^{k=n} [\underline{x}_k, \overline{x}_k]$.

5.1.1 Cas où l'ensemble des contraintes $\mathcal{K}$ est un polytope

Dans cette partie le POP que nous allons considérer est donné par :

$$\begin{array}{ll} \min & p(x) \\ \text{s.c} & x \in R, \\ & a_i \cdot x \leq b_i, \quad i = 1, \ldots, m, \\ & c_j \cdot x = d_j, \quad j = 1, \ldots, r, \end{array} \tag{5.1}$$

où $a_i \in \mathbb{R}^n$, $b_i \in \mathbb{R}$, pour tout $i = 1, \ldots, m$ et $c_j \in \mathbb{R}^n$, $d_j \in \mathbb{R}$, pour tout $j = 1, \ldots, r$.

Cas particulier : p est une fonction multi-affine

On va supposer dans un premier temps que le polynôme p est multi-affine, c'est à dire que $\delta_i \leq 1$ pour tout $i = 1, \ldots, n$.
Dans ce cas, la relaxation linéaire sera donnée par la formulation duale de (5.1). En effet, en écrivant le Lagrangien du problème (5.1) on aura :

$$L(x, \lambda, \nu) = p(x) + \sum_{i=1}^{m} \lambda_i (a_i \cdot x - b_i) + \sum_{j=1}^{r} \nu_i (c_j \cdot x - d_j)$$

où $x \in R$, $\lambda_i \geq 0$ pour tout $i \in 1, \ldots, m$.
Par conséquent, la formulation duale du problème (5.1) est

$$\begin{array}{ll} \max & \min\limits_{x \in R} L(x, \lambda, \nu) \\ \text{s.c} & \lambda \in \mathbb{R}^m, \nu \in \mathbb{R}^r, \\ & \lambda_i \geq 0, i = 1, \ldots, m. \end{array} \tag{5.2}$$

Notons $d^* \in \mathbb{R}$ la valeur optimale de (5.2). Il est bien connu d'après la théorie de dualité (voir [32]) que $d^* \leq p^*$ où p^* est la valeur optimale du problème (5.1). C'est ce qu'on appelle la dualité faible.

Remarque 9. *On sait aussi d'après la théorie de dualité, que la dualité forte équivalente au fait que $d^* = p^*$ aura lieu en général dans le cas d'un problème convexe. Or, une fonction multi-affine est généralement non convexe ; par conséquent, de façon générale, on ne peut pas s'attendre à avoir l'égalité $d^* = p^*$.*

On va montrer maintenant que le problème (5.2) peut s'écrire comme étant un programme linéaire :

Proposition 14. *Le problème dual de (5.1) est équivalent au programme linéaire suivant :*

$$\begin{array}{lll} max & t & \\ s.c & t \in \mathbb{R},\ \lambda \in \mathbb{R}^m,\ \mu \in \mathbb{R}^r, & \\ & \lambda_i \geq 0, & i = 1, \ldots, m, \\ & t \leq p(v) + \sum\limits_{i=1}^{m} \lambda_i (a_i \cdot v - b_i) & \\ & \quad + \sum\limits_{j=1}^{r} \mu_j (c_j \cdot v - d_j), & v \in V. \end{array} \tag{5.3}$$

Démonstration. On commence déjà par remarquer que le Lagrangien $L(x, \lambda, \mu)$ est une fonction multi-affine en x. Par suite, en appliquant le Lemme 1 (le lemme de convexité énoncé au Chapitre 1), on sait que le minimum de $L(x, \lambda, \mu)$ où $x \in R$ est atteint en un sommet de R :

$$\min_{x \in R} L(x, \lambda, \mu) = \min_{v \in V} L(v, \lambda, \mu).$$

Ainsi, le problème (5.2) consiste à minimiser une fonction linéaire par morceaux sous des contraintes linéaires ce qui peut se formuler simplement (voir [32], pages 150-151) en programme linéaire (5.3). □

Par conséquent, dans le cas où p est multi-affine, le programme (5.3) est bien une relaxation linéaire de notre problème (5.1) donnée grâce à la formulation duale. Dans le cas général où p est un polynôme de degré δ, cette approche va être généralisée au moyen du principe de floraison.

Cas général : p polynôme multi varié

On considère maintenant le problème (5.1) où p est un polynôme multi-varié de degré δ. En utilisant le principe de floraison défini dans le premier chapitre, on construit la forme polaire associée à p relative à δ qu'on note $p_\delta : \mathbb{R}^{\delta_1 + \cdots + \delta_n} \to \mathbb{R}$ donnée pour $z = (z_{1,1}, \ldots, z_{1,\delta_1}, \ldots, z_{n,1}, \ldots, z_{n,\delta_n})$ par

$$p_\delta(z) = \sum_{I \leq \delta} p_I \prod_{j=1}^{j=n} \mathcal{B}_{i_j,\delta_j}(z_{j,1}, \ldots, z_{j,\delta_j})$$

où l'expression des $\mathcal{B}_{i_j,\delta_j}$ est donnée par la Définition 4 du Chapitre 1.
On rappelle aussi la définition du rectangle associé $R' = \prod_{k=1}^{k=n} [\underline{x}_k, \overline{x}_k]^{\delta_k}$ dont l'ensemble de sommets est $V' = \prod_{k=1}^{k=n} \{\underline{x}_k, \overline{x}_k\}^{\delta_k}$.
Pour construire notre relaxation linéaire, on va utiliser les propriétés de la forme polaire données par la Proposition 1 du Chapitre 1.
En effet, pour se ramener au cas multi-affine, on va remplacer le polynôme p par sa forme polaire p_δ (qui est une fonction multi-affine d'après la propriété 1 de la forme polaire), tout en ajoutant des contraintes d'égalité pour que les deux problèmes restent équivalents. Ceci est assuré par la propriété 3 (propriété diagonale d'une forme polaire). Puis, on applique la Proposition 14 pour obtenir une relaxation linéaire au problème (5.1). On souligne à ce stade le fait qu'on se retrouve avec un problème de minimisation d'une fonction multi-affine sous contraintes d'inégalité et d'égalité, ce qui explique le fait qu'on ait tenu compte des contraintes d'égalité dans la définition de l'ensemble $\mathcal{K}$. Enfin, on utilisera la propriété 2 (propriété de symétrie) pour simplifier le programme linéaire obtenu et réduire sa complexité.

Concrètement en appliquant la propriété 3, on montre que le problème (5.1) est

équivalent à :

$$\begin{array}{ll} \text{minimise} & p_\delta(z) \\ \text{s.c} & z \in R', \\ & a_i{}' \cdot z \leq b_i, \quad i = 1, \ldots, m, \\ & c_j{}' \cdot z = d_j, \quad j = 1, \ldots, r, \\ & e_{k,l} \cdot z = 0, \quad k \in \{1, \ldots, n\}, l \in \{1, \ldots, \delta_k - 1\} \end{array} \tag{5.4}$$

avec

$$a_i'^{\top} = (\frac{a_{i,1}}{\delta_1}, \ldots, \frac{a_{i,1}}{\delta_1}, \ldots, \frac{a_{i,n}}{\delta_n}, \ldots, \frac{a_{i,n}}{\delta_n}), \text{ pour tout } i = 1, \ldots, m$$

$$c_j'^{\top} = (\frac{c_{j,1}}{\delta_1}, \ldots, \frac{c_{j,1}}{\delta_1}, \ldots, \frac{c_{j,n}}{\delta_n}, \ldots, \frac{c_{j,n}}{\delta_n}), \text{ pour tout } j = 1, \ldots, r$$

et les $e_{k,l} \in \mathbb{R}^{\delta_1 + \cdots + \delta_n}$ sont les vecteurs vérifiant l'égalité $e_{k,l} \cdot z = z_{k,l} - z_{k,l+1}$, pour tout $z \in \mathbb{R}^{\delta_1 + \cdots + \delta_n}$, $k \in \{1, \ldots, n\}$, $l \in \{1, \ldots, \delta_k - 1\}$.

Comme p_δ est une fonction multi-affine, le problème (5.4) est similaire à celui considéré dans la section précédente. Ainsi, on peut utiliser la Proposition 14 pour obtenir son dual, qui est donné par le programme linéaire suivant :

$$\begin{array}{ll} \max & t \\ \text{s.c} & t \in \mathbb{R},\ \lambda \in \mathbb{R}^m,\ \mu \in \mathbb{R}^r, \\ & \alpha \in \mathbb{R}^{(\delta_1 - 1) + \cdots + (\delta_n - 1)}, \\ & \lambda_i \geq 0, \qquad\qquad\qquad\qquad\qquad\qquad i = 1, \ldots, m, \\ & t \leq p_\delta(v) + \displaystyle\sum_{i=1,\ldots,m} \lambda_i (a_i' \cdot v - b_i) \\ & \qquad + \displaystyle\sum_{j=1,\ldots,l} \mu_j (c_j' \cdot v - d_j) \\ & \qquad + \displaystyle\sum_{k \in \{1,\ldots,n\}} \sum_{l \in \{1,\ldots,\delta_k - 1\}} \alpha_{k,l} (e_{k,l} \cdot v), \quad v \in V'. \end{array} \tag{5.5}$$

D'après la Proposition 14, la valeur optimale de ce programme linéaire donne une borne inférieure à la valeur optimale du problème d'optimisation polynomial (5.1). Cependant, on ne doit pas résoudre directement le programme linéaire (5.5) puisque la relation d'équivalence $\cong$ définie dans la Section 3.3.1 du Chapitre 2 peut être utilisée pour exploiter les symétries et réduire la complexité du programme linéaire. En effet, pour tout $v \cong v'$, $i = 1, \ldots, m$ et $j = 1, \ldots, r$ on sait déjà que $p_\delta(v) = p_\delta(v')$, $a_i \cdot v = a_i \cdot v'$ et $c_j' \cdot v = c_j' \cdot v'$; ainsi on peut bien écrire ces termes comme étant $a_i \cdot \overline{v}$ et $c_j \cdot \overline{v}$ pour $\overline{v} \in \overline{V}'$. Plus précisément, en exploitant cette relation d'équivalence, on obtient une relaxation linéaire équivalente au problème (5.5) avec beaucoup moins de variables et de contraintes.

Théorème 6. *La valeur optimale du programme linéaire (5.5) est égale à la valeur optimale $p_\delta{}^*$ de :*

$$
\begin{array}{lll}
max & t & \\
s.c & t \in \mathbb{R},\ \lambda \in \mathbb{R}^m,\ \mu \in \mathbb{R}^r, & \\
 & \lambda_i \geq 0, & i = 1, \ldots, m, \\
 & t \leq p_\delta(\overline{v}) + \displaystyle\sum_{i=1}^{m} \lambda_i (a_i' \cdot \overline{v} - b_i) & \\
 & \quad + \displaystyle\sum_{j=1}^{r} \mu_j (c_j' \cdot \overline{v} - d_j), & \overline{v} \in \overline{V}'.
\end{array}
\tag{5.6}
$$

Par suite, $p_\delta{}^* \leq p^*$ avec p^* est la valeur optimale de (5.1).

La preuve de ce théorème repose sur la formulation duale des problèmes (5.6) et (5.5). Elle est détaillée dans l'Appendice de ce chapitre.

Exemple 8.

$$
\begin{array}{ll}
minimiser & p(x_1, x_2) = x_1^2 + x_2 \\
s.c & (x_1, x_2) \in [0,1] \times [0,1], \\
 & -x_1 - x_2 \leq -1.
\end{array}
$$

La valeur optimale de ce problème est $p^* = 3/4$ obtenue pour $(x_1, x_2) = (0.5, 0.5)$. La forme polaire de p relative à $\delta = (2,1)$ est la fonction multi-affine

$$q_\delta(z_{1,1}, z_{1,2}, z_{2,1}) = z_{1,1} z_{1,2} + z_{2,1}.$$

Le programme linéaire (7.5) est le suivant :

$$
\begin{array}{lll}
maximiser & t & \\
s.c & t \in \mathbb{R},\ \lambda \in \mathbb{R}, & \\
 & \lambda \geq 0, & i \in I, \\
 & t \leq q_\delta(\overline{v}) + \lambda(a' \cdot \overline{v} + 1), & \overline{v} \in \overline{V'}.
\end{array}
$$

où $a' = (\frac{-1}{2}, \frac{-1}{2}, -1)$ et $\overline{V'} = \{(0,0,0), (0,0,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1)\}$ a 6 éléments. La valeur optimale de ce problème est $p_\delta{}^* = 2/3$ obtenue pour $\lambda = 4/3$. On peut bien vérifier que $p_\delta{}^* \leq p^*$.

On va maintenant discuter l'intérêt de résoudre (5.6) à la place de (5.5).

Remarque 10. *Résoudre (5.6) à la place de (5.5) nous permet un gain considérable en complexité. D'abord, les variables de décision $\alpha_{k,l}$ du problème (5.5) n'apparaissent plus dans (5.6). De plus, le nombre de contraintes indexées par $v' \in V'$ dans (5.5) est égal à $2^{\delta_1 + \cdots + \delta_n}$ tandis que le nombre de contraintes indexées par $\overline{v} \in \overline{V}'$ est uniquement $(\delta_1 + 1) \times \cdots \times (\delta_n + 1)$. En effet, le programme linéaire (5.5) a $m + r + (\delta_1 - 1) + \cdots + (\delta_n - 1) + 1$ variables et $2^{\delta_1 + \cdots + \delta_n} + m$ contraintes d'inégalité alors que le programme linéaire (5.6) a uniquement $m + r + 1$ variables et $(\delta_1 + 1) \times \cdots \times (\delta_n + 1) + m$ contraintes d'inégalité.*

5.1.2 Cas où l'ensemble $\mathcal{K}$ est semi-algébrique

Maintenant, nous allons généraliser l'approche décrite précédemment dans le cas où les fonctions g_i et les fonctions h_i sont données par des polynômes.
Soit $\delta = (\delta_1, \ldots, \delta_n)$ où les δ_i sont les degrés maximaux des polynômes $p, g_1, \ldots, g_m$, $h_1, \ldots, h_r$ en les variables x_i et notons par $p_\delta, g_{1\delta}, \ldots, g_{m\delta}, h_{1\delta}, \ldots, h_{r\delta}$ les formes polaires associées relatives à δ.
Pour obtenir une relaxation linéaire du problème (5.1) pour le cas semi-algébrique, on va suivre les mêmes étapes que dans le cas polytope. D'abord, en utilisant la propriété diagonale des formes polaires $p_\delta, g_{1\delta}, \ldots, g_{m\delta}, h_{1\delta}, \ldots, h_{r\delta}$ on obtient le problème multi-affine équivalent suivant :

$$\begin{array}{lll} \text{minimiser} & p_\delta(z) & \\ \text{s.c} & z \in R', & \\ & g_{i\delta}(z) \leq 0, \quad i = 1, \ldots, m, & \\ & h_{j_\delta}(z) = 0, \quad j = 1, \ldots, r, & \\ & e_{k,l} \cdot z = 0, \quad k \in \{1, \ldots, n\}, l \in \{1, \ldots, \delta_k - 1\}, & \end{array} \tag{5.7}$$

où les $e_{k,l}$ sont définis comme dans le cas polytope.
En écrivant le Lagrangien du problème (5.7) on aura :

$$L(z, \lambda, \nu) = p_\delta(z) + \sum_{i=1}^{m} \lambda_i g_{i\delta}(z) + \sum_{j=1}^{r} \nu_j h_{j_\delta}(z) + \sum_{k \in \{1,\ldots,n\}} \sum_{l \in \{1,\ldots,\delta_k - 1\}} \alpha_{k,l}(e_{k,l} \cdot z)$$

où $z \in R'$, $\lambda_i \geq 0$ pour tout $i \in \{1, \ldots, m\}$, $\nu_j \in \mathbb{R}$ pour tout $j \in \{1, \ldots, r\}$ et $\alpha_{k,l} \in \mathbb{R}$ pour tout $k \in \{1, \ldots, n\}$ et tout $l \in \{1, \ldots, \delta_k - 1\}$.
En utilisant le fait que $L(z, \lambda, \nu)$ est multi-affine en z ainsi que le lemme de convexité des fonctions multi-affines, on obtient le programme linéaire suivant :

$$\begin{array}{lll} \max & t & \\ \text{s.c} & t \in \mathbb{R},\ \lambda \in \mathbb{R}^m,\ \mu \in \mathbb{R}^r, & \\ & \alpha \in \mathbb{R}^{(\delta_1 - 1) + \cdots + (\delta_n - 1)}, & \\ & \lambda_i \geq 0, & i = 1, \ldots, m, \\ & t \leq p_\delta(v) + \displaystyle\sum_{i=1,\ldots,m} \lambda_i g_{i\delta}(z) & \\ & \quad + \displaystyle\sum_{j=1,\ldots,l} \mu_j h_{j_\delta}(z) & \\ & \quad + \displaystyle\sum_{k \in \{1,\ldots,n\}} \sum_{l \in \{1,\ldots,\delta_k - 1\}} \alpha_{k,l}(e_{k,l} \cdot v), & v \in V'. \end{array} \tag{5.8}$$

Enfin, en utilisant la relation d'équivalence, on obtient la relaxation linéaire suivante :

Théorème 7. *La valeur optimale du programme linéaire (5.8) est égale à la valeur optimale $p_\delta{}^*$ de :*

$$
\begin{array}{lll}
max & t & \\
s.c & t \in \mathbb{R},\ \lambda \in \mathbb{R}^m,\ \mu \in \mathbb{R}^r, & \\
 & \lambda_i \geq 0, & i = 1, \ldots, m, \\
 & t \leq p_\delta(\overline{v}) + \sum_{i=1}^{m} \lambda_i g_{i\delta}(\overline{v}) & \\
 & \quad + \sum_{j=1}^{r} \mu_j h_{j\,\delta}(\overline{v}), & \overline{v} \in \overline{V}'.
\end{array} \tag{5.9}
$$

Par suite, $d^* \leq p^*$ avec p^* est la valeur optimale de (5.1).

La preuve est similaire à celle du Théorème 6 et se base essentiellement sur la propriété de symétrie d'une forme polaire. Elle est brièvement décrite dans l'Appendice de ce chapitre.

Dans la section qui suit, on va exploiter le lien décrit dans la première partie de la thèse entre les coefficients de Bernstein et la forme polaire pour pouvoir reformuler les résultats de cette section à l'aide des coefficients de Bernstein. Par dualité, on verra que les programmes linéaires qu'on obtient proviennent immédiatement des propriétés des polynômes de Bernstein décrite dans le Chapitre 3. En effet, on donnera une relaxation encore plus précise que celle obtenue à l'aide de la forme polaire en exploitant uniquement ces différents propriétés.

5.2 Relaxation d'un POP en utilisant la forme de Bernstein

5.2.1 Cas général : $\mathcal{K}$ est semi-algébrique

Analogie avec la relaxation obtenue à l'aide de la forme polaire

Dans la section précédente, on a présenté une relaxation linéaire pour notre problème d'optimisation polynomial se basant sur la forme polaire d'un polynôme (le principe de floraison). Etant donné la relation établie dans la Section 3.3 entre la forme polaire d'un polynôme et les polynômes de Bernstein, il est tout à fait naturel de vouloir reformuler cette relaxation à l'aide des coefficients de Bernstein. L'intérêt pratique de cette reformulation vient de la remarque suivante.

Remarque 11. *Pour la relaxation linéaire (5.9), on a besoin de calculer les valeurs de $p_\delta(\overline{v}), g_{1\delta}(\overline{v}), \ldots, g_{m\delta}(\overline{v}), h_{1\delta}(\overline{v}), \ldots, h_{r\delta}(\overline{v})$ pour tout $\overline{v} \in \overline{V}'$. En effet, ces opérations risquent d'être trop coûteuse puisque les formes polaires associées peuvent compter $2^{\delta_1+\cdots+\delta_n}$ termes. Afin de garder le coût de calcul le plus petit possible, on devra éviter de calculer explicitement les formes polaires ; il est préférable d'utiliser la Proposition 9 qui montre qu'il suffit de calculer les $(\delta_1+1)\times\cdots\times(\delta_n+1)$ coefficients des polynômes $p, g_1, \ldots, g_m, h_1, \ldots, h_r$ dans la base de Bernstein.*

Tout d'abord, comme les polynômes de Bernstein sont définis dans la boîte unité $U = [0,1]^n$, on va se restreindre au rectangle $R = [0,1]^n$. Le cas général se déduit à

l'aide d'une transformation linéaire.
On commence tout d'abord par calculer explicitement à l'aide de la formule (3.6) du Chapitre 3 les ensembles $\{\hat{p}_{I,\delta} \in \mathbb{R}\ ,\ I \leq \delta\}$, $\{\hat{g}_{iI,\delta} \in \mathbb{R}\ ,\ I \leq \delta\}$ pour tout $i = 1, \ldots, m$ et $\{\hat{h_{j}}_{I,\delta} \in \mathbb{R}\ ,\ I \leq \delta\}$ pour tout $j = 1, \ldots, r$, désignant les ensembles des coefficients de Bernstein d'ordre δ associés respectivement aux polynômes $p, g_1, \ldots, g_m, h_1, \ldots, h_r$.

D'après la Proposition 9, la relaxation (5.9) est équivalente à :

$$\begin{array}{lll} \max & t & \\ \text{s.c} & t \in \mathbb{R},\ \lambda \in \mathbb{R}^m,\ \mu \in \mathbb{R}^r, & \\ & \lambda_i \geq 0, & i = 1, \ldots, m, \\ & t \leq \hat{p}_{I,\delta} + \sum_{i=1}^{m} \lambda_i \hat{g}_{iI,\delta} & \\ & \quad + \sum_{j=1}^{r} \mu_j \hat{h_{j}}_{I,\delta}, & I \leq \delta. \end{array} \tag{5.10}$$

Dualité et propriétés des polynômes de Bernstein

Nous allons voir maintenant que, le programme linéaire (5.10) (et plus précisément son dual qui lui est équivalent) pourra être établi simplement en exploitant les propriétés des polynômes de Bernstein qu'on a énumérées dans le Chapitre 3.

Proposition 15. *La formulation duale du problème (5.10) est donnée par le programme linéaire suivant :*

$$\begin{array}{lll} \textit{minimiser} & \sum_{I \leq \delta} \hat{p}_{I,\delta} z_I & \\ \textit{s.c} & z_I \in \mathbb{R}, & I \leq \delta, \\ & z_I \geq 0, & I \leq \delta, \\ & \sum_{I \leq \delta} z_I = 1, & \\ & \sum_{I \leq \delta} \hat{g}_{iI,\delta} z_I \leq 0, & i = 1, \ldots, m. \\ & \sum_{I \leq \delta} \hat{h_{j}}_{I,\delta} z_I = 0, & j = 1, \ldots, r. \end{array} \tag{5.11}$$

De plus les problèmes (5.10) et (5.11) ont la même valeur optimale.

On détaillera la preuve dans l'Appendice de ce chapitre. Cependant, on va voir comment on peut obtenir ce programme linéaire sans passer par la forme polaire, c'est à dire en n'utilisant que les résultats liés aux polynômes de Bernstein.
On rappelle que notre POP initial est le suivant :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in [0,1]^n \\ & g_i(x) \leq 0\ ,\ i = 1, \ldots, m. \\ & h_j(x) = 0\ ,\ j = 1, \ldots, r. \end{array} \tag{5.12}$$

On commence par écrire tous les polynômes $p, g_1, \ldots, g_m$ dans la base de Bernstein d'ordre δ. Ainsi on aura :

$$p(x) = \sum_{I \leq \delta} \hat{p}_{I,\delta} B_{\delta,I}(x) \text{ avec } \hat{p}_{I,\delta} \in \mathbb{R},\ \forall I \leq \delta,$$

$$
\begin{aligned}
g_1(x) &= \sum_{I\leq\delta} \hat{g_1}_{I,\delta} B_{\delta,I}(x) \text{ avec } \hat{g_1}_{I,\delta} \in \mathbb{R},\ \forall I \leq \delta,\\
&\cdots\\
&\cdots\\
g_m(x) &= \sum_{I\leq\delta} \hat{g_m}_{I,\delta} B_{\delta,I}(x) \text{ avec } \hat{g_m}_{I,\delta} \in \mathbb{R},\ \forall I \leq \delta,\\
h_1(x) &= \sum_{I\leq\delta} \hat{h_1}_{I,\delta} B_{\delta,I}(x) \text{ avec } \hat{h_1}_{I,\delta} \in \mathbb{R},\ \forall I \leq \delta,\\
&\cdots\\
&\cdots\\
h_r(x) &= \sum_{I\leq\delta} \hat{h_r}_{I,\delta} B_{\delta,I}(x) \text{ avec } \hat{h_r}_{I,\delta} \in \mathbb{R},\ \forall I \leq \delta.
\end{aligned}
$$

En injectant ces expressions dans (5.12), on obtient le problème équivalent suivant :

$$
\begin{array}{ll}
\text{minimiser} & \sum_{I\leq\delta} \hat{p}_{I,\delta} B_{\delta,I}(x)\\
\text{s.c} & x \in [0,1]^n\\
& \sum_{I\leq\delta} \hat{g}_{iI,\delta} B_{\delta,I}(x) \leq 0\ ,\ i = 1,\ldots,m.\\
& \sum_{I\leq\delta} \hat{h_j}_{I,\delta} B_{\delta,I}(x) = 0\ ,\ j = 1,\ldots,r.
\end{array}
\tag{5.13}
$$

Ainsi, il suffit de remplacer dans (5.13) chaque polynôme de Bernstein $B_{\delta,I}(x)$ par une variable de décision z_I et utiliser le fait que $B_{\delta,I}(x) \geq 0$ pour tout $I \leq \delta$ et que $\sum_{I\leq\delta} B_{\delta,I}(x) = 1$ pour pouvoir obtenir la relaxation linéaire au problème (5.12) donnée par le programme linéaire (5.11).

Une meilleure relaxation

On a vu précédemment qu'une relaxation linéaire équivalente à (5.9) peut être obtenue grâce à l'utilisation de certaines propriétés sur les polynômes de Bernstein (celle sur la partition de l'unité et celle sur la positivité des polynômes de Bernstein) pour pouvoir récupérer le programme linéaire (5.11) dont on a montré l'équivalence avec (5.9). Cependant, une question qu'on peut se poser est : Peut-on faire mieux ? La réponse est affirmative et l'idée consiste à ajouter des contraintes supplémentaires sur nos variables de décision z_I.

Proposition 16. *Notons par $\tilde{p}^*_\delta$ la valeur optimale du programme linéaire :*

$$
\begin{array}{lll}
\textit{minimiser} & \sum_{I\leq\delta} \hat{p}_{I,\delta} z_I & \\
\textit{s.c} & z_I \in \mathbb{R}, & I \leq \delta,\\
& 0 \leq z_I \leq B_{\delta,I}(\frac{I}{\delta}), & I \leq \delta,\\
& \sum_{I\leq\delta} z_I = 1, & \\
& \sum_{I\leq\delta} \hat{g}_{iI,\delta} z_I \leq 0, & i = 1,\ldots,m,\\
& \sum_{I\leq\delta} \hat{h_j}_{I,\delta} z_I = 0, & j = 1,\ldots,r.
\end{array}
\tag{5.14}
$$

Alors, ${p_\delta}^* \leq \tilde{p}^*_\delta \leq p^*$ où p^* est la valeur optimale du problème (5.13) et ${p_\delta}^*$ est la valeur optimale du problème (5.11).

Démonstration. Il est évident que $p_\delta{}^* \leq \tilde{p}_\delta^*$ vu que le programme (5.14) n'est autre que le programme (5.11) avec des contraintes supplémentaires et il s'agit d'une minimisation.
Maintenant pour montrer que $\tilde{p}_\delta^* \leq p^*$ il suffit de remarquer qu'il s'agit bien d'une relaxation du problème initial (5.13) puisque les contraintes supplémentaires $z_I \leq B_{\delta,I}(\frac{I}{\delta})$ proviennent du fait que $B_{\delta,I}(x) \leq B_{\delta,I}(\frac{I}{\delta})$ pour tout $I \leq \delta$ (ceci provient de la deuxième propriétés des polynômes de Bernstein). □

On verra plus loin sur des exemples qu'en résolvant (5.14) au lieu de (5.11) un gain important en précision sera obtenu.

5.2.2 Cas particulier : $\mathcal{K}$ est un polytope

Le cas d'un polytope est d'un grand intérêt pour les applications (voir la troisième partie de cette thèse). Nous allons donc le traiter en détail. On supposera dans un premier temps (comme précédemment) que le rectangle R est la boîte unité. Le cas d'un rectangle quelconque sera traité par la suite.
Comme $\mathcal{K}$ est un polytope, on aura $g_i(x) = a_i \cdot x - b_i$ avec $a_i \in \mathbb{R}^n$ et $b_i \in \mathbb{R}$ pour tout $i = 1, \ldots, m$. Ainsi, notre POP s'écrit sous la forme suivante :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in R = [0,1]^n, \\ & Ax \leq b \end{array} \tag{5.15}$$

où $A \in \mathbb{R}^{m\times n}$ est la matrice dont les colonnes sont les vecteurs $a_i, i = 1 \ldots, m$ et $b \in \mathbb{R}^m$ est le vecteur dont les coordonnés sont les $b_i, i = 1, \ldots, m$.
Soient $\delta_1, \ldots, \delta_n$ les degrés respectifs de p en $x_1, \ldots, x_n$ et notons $\delta = (\delta_1, \ldots, \delta_n)$.
Dans la base canonique le polynôme p s'écrit sous la forme :

$$p(x) = \sum_{I\leq\delta} p_{I,\delta} x^I \text{ avec } p_{I,\delta} \in \mathbb{R},\ \forall I \leq \delta.$$

Dans la base de Bernstein d'ordre δ le polynôme p s'écrit sous la forme :

$$p(x) = \sum_{I\leq\delta} \hat{p}_{I,\delta} B_{\delta,I}(x) \text{ avec } \hat{p}_{I,\delta} \in \mathbb{R},\ \forall I \leq \delta \tag{5.16}$$

où l'ensemble $\{\hat{p}_{I,\delta} \in \mathbb{R}\ ,\ I \leq \delta\}$ désigne l'ensemble des coefficients de Bernstein qu'on pourra calculer explicitement à l'aide de la formule (3.6) du Chapitre 3.

Pour établir une relaxation LP à notre problème (5.15), on aura besoin des propriétés suivantes (voir Proposition 6 du Chapitre 3) :

1. Pour $x \in \mathbb{R}^n$,
$$\sum_{I\leq\delta} B_{\delta,I}(x) = 1 \text{ et } \sum_{I\leq\delta} \tfrac{I}{\delta} B_{\delta,I}(x) = x.$$
2. Pour tout $x \in [0,1]^n$,
$$0 \leq B_{\delta,I}(x) \leq B_{\delta,I}(\tfrac{I}{\delta})$$
avec $B_{\delta,I}(\frac{I}{\delta}) = \prod_{j=1}^{j=n} \begin{pmatrix} \delta_j \\ i_j \end{pmatrix} \frac{i_j^{i_j} (\delta_j - i_j)^{\delta_j - i_j}}{\delta_j^{\delta_j}}$.

On peut utiliser ces propriétés pour construire une relaxation LP à notre problème (5.15) :

Théorème 8. *Soit $p_\delta{}^*$ la valeur optimale du programme linéaire suivant :*

$$\begin{array}{lll} \textit{minimiser} & \sum_{I\leq\delta}\hat{p}_{I,\delta}z_I & \\ \textit{s.c} & z_I\in\mathbb{R}, & I\leq\delta, \\ & 0\leq z_I\leq B_{\delta,I}(\frac{I}{\delta}), & I\leq\delta, \\ & \sum_{I\leq\delta}z_I=1, & \\ & \sum_{I\leq\delta}(A\frac{I}{\delta})z_I\leq b. & \end{array} \tag{5.17}$$

Alors, $p_\delta{}^*\leq p^*$ où p^* est la valeur optimale du problème (5.15).

Démonstration. En utilisant l'expression de p dans la base de Bernstein (5.16) ainsi que la première propriété, on peut récrire le problème (5.15) sous la forme suivante :

$$\begin{array}{ll} \text{minimiser} & \sum_{I\leq\delta}\hat{p}_{I,\delta}B_{\delta,I}(x) \\ \text{s.c} & x\in[0,1]^n, \\ & \sum_{I\leq\delta}(A\frac{I}{\delta})B_{\delta,I}(x)\leq b. \end{array}$$

Soit x^* l'optimum du problème (5.15) et soit $z_I=B_{\delta,I}(x^*)$ pour tout $I\leq\delta$. Il est évident d'après les propriétés 1 et 2 que les z_I satisfont les contraintes du problème (5.17). Par conséquent, la valeur optimale $p_\delta{}^*$ de (5.17) est nécessairement plus petite que celle de (5.15) d'où $p_\delta{}^*\leq p^*$. □

Maintenant, on revient au cas où $R=\prod_{k=1}^{k=n}[\underline{x}_k,\overline{x}_k]$ est un rectangle quelconque de $\mathbb{R}^n$. En effet, dans ce cas il suffit de se ramener au cas d'une boîte unité.
On commence par écrire le rectangle R sous une forme matricielle c'est à dire qu'il est vu comme étant un polytope de $2n$ faces. Ainsi on peut écrire :

$$R=\{x\in\mathbb{R}^n|\quad \tilde{I}x\leq\tilde{d}\}\text{ avec }\tilde{I}=\begin{bmatrix} I \\ -I\end{bmatrix},\ \tilde{d}=\begin{bmatrix}\overline{x}\\ -\underline{x}\end{bmatrix},$$

où $I\in\mathbb{R}^{n\times n}$ est la matrice identité et les vecteurs $\underline{x}\in\mathbb{R}^n$, $\overline{x}\in\mathbb{R}^n$ sont donnés par :

$$\underline{x}=(\underline{x_1},\ldots,\underline{x_n})^\top\text{ et }\overline{x}=(\overline{x_1},\ldots,\overline{x_n})^\top.$$

Puis, on procède au changement de variable $x=q(y)$ où la fonction affine $q:\mathbb{R}^n\to\mathbb{R}^n$ est donnée par :

$$q(y)=Dy+\underline{x}$$

où D est une matrice diagonale d'entrées $\underline{x_k}-\overline{x_k}$.
Le changement de variable q renvoie essentiellement la boîte $[0,1]^n$ au rectangle R. Pour finir, il faut aussi transformer la matrice A et le vecteur b. Cette transformation est donnée par :

$$A'=AD,\ b'=b-A\underline{x}.$$

Ainsi le problème (5.15) est équivalent au problème :

$$\begin{array}{ll} \text{minimiser} & p\circ q(y) \\ \text{s.c} & y\in[0,1]^n, \\ & A'y\leq b'. \end{array} \tag{5.18}$$

Par conséquent, une borne inférieure $p_\delta{}^*$ au problème (5.15) peut être calculée en utilisant le Théorème 8.

5.3 Exemples et comparaison

On va comparer les résultats obtenus par nos relaxations ainsi que ceux obtenus par la relaxation RLT et la relaxation LMI sur quelques exemples de problèmes d'optimisation polynomiaux. Ces exemples seront implémentés sur Matlab.
On va étudier les deux exemples suivants :

Exemple 9. *Considérons le problème unidimensionnel suivant [184] (voir aussi [163]) :*

$$\begin{array}{ll} min & y^4 - 3y^3 - 1.5y^2 + 10y \\ s.c & y \in [-5, 5]. \end{array} \tag{5.19}$$

La valeur ainsi que la solution optimale sont données par $p^* = -7.5$ *et* $y^* = -1$. *Les caractéristiques des quatre méthodes de relaxations sont données par le tableau de la Figure 5.1. En comparant ces relaxations, il est évident que la relaxation LMI*

	RLT	LMI	Bernstein	Floraison
Contraintes	5	3×3+2(2×2)	2×5+1	5
Variables	4	4	5	1
Valeur optimale	-908.3	-7.5	-486.32	-837.5
Temps de calcul (sec)	-	0.246	0.0324	0.1370

FIGURE 5.1 – Caractéristiques des quatre méthodes pour le problème (5.19).

donne la meilleure borne (la valeur optimale exacte), cependant elle nécessite plus de ressources de calcul[1].
Pour les autres relaxations, on remarque que nos deux relaxations sont plus précises que la relaxation RLT et que la relaxation de Bernstein est celle qui donne la meilleure borne. En effet, comme cela a été discuté auparavant, cette relaxation a davantage de contraintes que celle utilisant le principe de floraison, ce qui explique le fait qu'elle soit plus précise. Cependant, toutes les bornes inférieures fournies par les relaxations basées sur la programmation linéaire sont loin de la vraie valeur optimale. Ceci est dû au fait que les sommets de la boîte fixée à l'avance (dans ce cas c'est l'intervalle $[-5, 5]$*) sont trop loin de la solution optimale (*$y^* = -1$*), en réduisant la taille de la boîte les bornes deviendront nettement meilleures comme le montre le tableau 5.2. On remarque que la valeur optimale est bien obtenue pour*

Boîte	[-5,5]	[-4,4]	[-3,3]	[-2,2]	[-1,1]
Valeur optimale : Floraison	-837.5	-372	-136.5	-38	-7.5
Valeur optimale : Bernstein	-486.32	-203.5	-65.76	-12.56	-7.5

FIGURE 5.2 – Amélioration des valeurs optimales données par notre méthode pour le problème (5.19) en réduisant la taille de la boîte.

la boîte $[-1, 1]$ *pour nos deux relaxations, ce qui est évident car pour cette boîte la*

1. Pour l'approche RLT, les linéarisations sont calculées manuellement. Pour cela, le temps de calcul n'est pas indiqué

condition aux sommets est établie puisque $y^* = -1$ est bien un sommet de cette boîte.

Exemple 10. *Maintenant on considère le problème en 3 dimensions avec contraintes suivant [162] :*

$$\begin{array}{ll} min & x_1x_2x_3 + {x_1}^2 - 2x_1x_2 - 3x_1x_3 \\ & +5x_2x_3 - {x_3}^2 + 5x_2 + x_3 \\ s.c & x = (x_1, x_2, x_3) \in [2,5] \times [0,10] \times [4,8], \\ & 4x_1 + 3x_2 + x_3 \leq 20, \\ & x_1 + 2x_2 + x_3 \geq 1. \end{array} \tag{5.20}$$

La valeur optimale ainsi que la solution associée sont données par $p^* = -119$ et $x^* = ({x_1}^*, {x_2}^*, {x_3}^*) = (3,0,8)$. Les caractéristiques des quatre relaxations sont collectées dans le tableau de la Figure 5.3.

	RLT	LMI	Bernstein	Floraison
Contraintes	56+2	10×10+8(4×4)	2×18+1+2	18+2
Variables	19	34	18	3
Valeur optimale	-120	-119	-119.25	-120
Temps de calcul (sec)	-	0.548	0.1808	0.2082

FIGURE 5.3 – Caractéristiques des quatre méthodes pour le problème (5.20).

On peut remarquer que nos deux approches sont celles impliquant les relaxations les plus simples (problèmes avec moins de variables et de contraintes). Le temps nécessaire pour calculer une borne inférieure à la valeur optimale est inférieur à celui de l'approche LMI. Pour la relaxation se basant sur le principe de floraison, la borne inférieure est égale à celle calculée par la relaxation RLT mais elle n'est pas si précise que celle calculée par la relaxation se basant sur les polynômes de Bernstein (qui est très proche de la valeur optimale) et par rapport à la relaxation LMI qui trouve la valeur optimale exacte.
Par contre, si on découpe notre boîte R en deux sous boîtes $R_1 = [2,3] \times [0,10] \times [4,8]$ et $R_2 = [3,5] \times [0,10] \times [4,8]$, en utilisant nos deux relaxations, la valeur optimale peut être obtenue (voir le tableau de la Figure 5.4). En effet, la valeur optimale est obtenue dans les deux sous boîtes. L'explication est dûe au fait que dans chacune de ces sous boîtes la condition au sommets est établie en $v = (3,0,8)$ et donc on en déduit que $x^* = (3,0,8)$.

Boîte	R_1	R_2
Valeur optimale	-119	-119

FIGURE 5.4 – Les valeurs optimales données par nos méthodes au problème (5.20) pour les deux sous boîtes.

Pour conclure, on a vu grâce aux deux exemples précédents que grâce à nos deux relaxations on pouvait obtenir des bornes pour des problèmes d'optimisation polynomiaux non triviaux. Théoriquement, on a montré que la relaxation de Bernstein est

plus efficace que celle utilisant le principe de floraison. Ce résultat est bien confirmé par nos applications numériques. Cependant, on verra dans la troisième partie de la thèse l'utilité de la relaxation fournie à l'aide de la forme polaire.

5.4 Amélioration de la précision de nos relaxations

Dans cette section, on va citer deux moyens permettant d'augmenter la précision de nos relaxations. Une première alternative consiste à décomposer le domaine en petits morceaux, et donc raffiner la recherche de l'optimum ainsi que de la solution optimale. C'est le cadre des algorithmes de "branch-and-bound". La deuxième alternative consiste à augmenter le degré du polynôme (donné par le multi indice δ) en le considérant comme étant un polynôme de degré supérieur δ' en lui ajoutant des coefficients nuls .

5.4.1 Algorithmes de "branch-and-bound" utilisant Bernstein

Comme on a pu le remarquer dans la section précédente, la valeur optimale peut être obtenue si une bonne décomposition de notre domaine rectangulaire est faite. C'est l'idée des algorithmes de "branch-and-bound".
Récemment, plusieurs travaux ont été réalisés pour la construction des algorithmes de "branch-and-bound" se basant sur les polynômes de Bernstein, notamment par Ray et Nataraj [125, 146, 147], permettant essentiellement de calculer le minimum (respectivement le maximum) d'un polynôme multi-varié sur un domaine rectangulaire. Il s'agit de la résolution du problème (4.5) dans le cas sans contraintes (c'est à dire on se restreint au rectangle R). L'idée principale est d'utiliser les résultats de la Section 3.2 du Chapitre 3 et plus précisément la Proposition 7 permettant de borner l'image d'une boîte par un polynôme à l'aide des coefficients de Bernstein et le Lemme 2 permettant à l'aide d'une condition aux sommets de savoir si cette image est exacte (relaxation exacte).
On va décrire brièvement les étapes principales de ces algorithmes :

Etape de résolution :
La relaxation sera donnée en utilisant la Proposition 7. Il s'agit de calculer le minimum des coefficients de Bernstein dans la boîte R. Rappelons à ce stade, que la Proposition 7 n'est valable que pour $R = [0, 1]^n$ et qu'on se ramène au cas d'une boîte unité au moyen d'un changement de variable. Cependant, pour éviter de faire un changement de variables pour chaque boîte, on peut utiliser la "sweep-procedure", permettant de calculer à partir des coefficients de Bernstein associés à une boîte les coefficients de Bernstein associés aux sous boîtes (voir [73]).

Etape de tests :
Pour tester si la relaxation est exacte il suffit d'appliquer le Lemme 2.
Cependant, plusieurs autres tests, dit tests d'accélération sont fournis tels que :

— Test de monotonie : il s'agit de tester la monotonie du polynôme p dans la sous-boîte (à l'aide des coefficients de Bernstein de la dérivée p'). L'idée principale est que si p est strictement monotone dans une sous-boîte alors forcément x^* ne s'y trouve pas.
— Test de concavité : il s'agit de vérifier la concavité à l'aide des coefficients de Bernstein de p''. Si p est concave dans la sous-boîte alors x^* n'y appartient pas.

Etape de décomposition :
La plupart des contributions se font dans le cadre de cet étape. En effet, on cherche toujours un bon moyen de découper la boîte de telle sorte que le nombre final de décompositions soit minimal.
On commence par décrire quelques lois triviales pour le choix de la direction de décomposition r :

1. Loi cyclique : elle consiste à prendre $r = 1$ puis $r = 2$ et ainsi de suite (quand $r = n$ la direction suivante sera de nouveau $r = 1$).
2. Largeur maximale : elle consiste à choisir la direction correspondant au plus grand côté du rectangle R.
3. Dérivée maximale : Elle consiste à choisir la direction suivant laquelle la dérivée partielle (calculée à l'aide des coefficients de Bernstein) est maximale.

Dans [125], une amélioration des lois précédentes en les multipliant par un certain facteur de décomposition est proposée. D'autre part, étant donné un ensemble de boîtes, une autre approche consiste à calculer une direction de décomposition seulement pour la boîte contenant le coefficient de Bernstein minimal (voir [146]). Cette direction sera ensuite utilisée pour toutes les autres boîtes.
Maintenant, pour une direction r fixée, on veut calculer un point de décomposition x_r. Un choix trivial est de prendre le point du milieu c'est à dire de prendre $x_r = \frac{\underline{x}_r + \overline{x}_r}{2}$. D'autres choix moins triviaux existent (voir par exemple [146, 147]). En effet, étant donné une direction de décomposition r, un bon point de décomposition doit être un point en lequel la dérivée partielle par rapport à r qu'on note p'_r est proche de zéro. Une approche permettant d'assurer cela est celle décrite dans [147].

Remarque 12. *On peut construire un algorithme de "branch-and-bound" se basant sur les relaxations introduites dans ce chapitre pour la résolution du problème (4.5). Notons que si on s'intéresse au cas sans contraintes, le résultat de notre relaxation se basant sur le principe de floraison donné par le programme linéaire (5.9) n'est autre que :*

$$\begin{array}{ll} \textit{minimiser} & p_\delta(\overline{v}) \\ s.c & \overline{v} \in \overline{V'} \end{array} \tag{5.21}$$

et coïncide avec le minimum des coefficients de Bernstein sur le rectangle R. Cependant, comme on l'a remarqué précédemment, le résultat obtenu à l'aide de notre relaxation de Bernstein (5.17), après avoir effectué un changement de variable pour se ramener à la boîte unité, sera plus précis grâce à ses contraintes supplémentaires.

5.4.2 Elévation du degré et résultat de convergence

Sans perte de généralité, on peut supposer dans cette section que $R = [0,1]^n$. On va considérer le problème (5.15) et établir un résultat de convergence pour la relaxation donnée par le programme linéaire (5.17) fourni à l'aide des coefficients de Bernstein. On verra que les contraintes supplémentaires sur les z_I introduites pour améliorer le programme linéaire (5.9) construit à l'aide du principe de floraison ne vont pas intervenir dans la preuve. Par conséquent les résultats établis resteront valables dans le cas de la relaxation (5.6) donnée à l'aide de la forme polaire.
On commence par remarquer que le polynôme p (de degré δ) peut être considéré comme étant un polynôme de degré supérieur en ajoutant des monômes de degré supérieur avec des coefficients nuls.

Dans ce qui suit, p est considéré comme étant un polynôme de degré K où $K = (k_1, \ldots, k_n)$ avec $K \geq \delta$. Ainsi, dans la base de Bernstein d'ordre K, p a la forme suivante :

$$p(x) = \sum_{I \leq K} \hat{p}_{I,K} B_{K,I}(x).$$

On va utiliser le Théorème 5 énoncé dans le Chapitre 2 concernant la convergence en degré supérieur des coefficients de Bernstein vers les valeurs du polynôme sur $[0,1]^n$. En appliquant ce théorème on a : pour tout $I \leq K$,

$$\left|\hat{p}_{I,K} - p(\tfrac{I}{K})\right| = O\left(\frac{1}{k_1} + \cdots + \frac{1}{k_n}\right). \tag{5.22}$$

Maintenant, notons par ${p_K}^*$ la valeur optimale du programme linéaire (5.17) avec $\delta = K$:

$$\begin{array}{lll} \text{minimiser} & \sum_{I \leq \delta} \hat{p}_{I,\delta} z_I & \\ \text{s.c} & z_I \in \mathbb{R}, & I \leq \delta, \\ & 0 \leq z_I \leq B_{\delta,I}(\frac{I}{\delta}), & I \leq \delta, \\ & \sum_{I \leq \delta} z_I = 1, & \\ & \sum_{I \leq \delta} (A\frac{I}{\delta}) z_I \leq b. & \end{array}$$

On veut déterminer la limite de p_K^* quand tous les k_i tendent vers l'infini.

Notons $c_K^*(x)$ la valeur optimale du programme linéaire :

$$\begin{array}{lll} \text{minimiser} & \sum_{I \leq K} \hat{p}_{I,K} z_I & \\ \text{s.c} & z_I \in \mathbb{R}, & I \leq K, \\ & 0 \leq z_I \leq B_{K,I}(\frac{I}{K}), & I \leq K, \\ & \sum_{I \leq K} z_I = 1, & \\ & x = \sum_{I \leq K} \frac{I}{K} z_I & \end{array} \tag{5.23}$$

Alors,

$${p_K}^* = \min_{\substack{x \in [0,1]^n \\ Ax \leq b}} c_K^*(x) \leq p^*.$$

Pour énoncer notre résultat de convergence, on a besoin de la définition suivante (voir [89]) :

Définition 6. *On appelle enveloppe convexe d'une fonction f sur un ensemble convexe non vide U la fonction notée $C(f) = C_U(f)$ vérifiant :*

— *$C(f)$ est une fonction convexe définie sur U.*

— *$C(f)(x) \leq f(x)$ pour tout $x \in C$.*

— *Si $h : U \to D$ est une fonction convexe telle que $h(x) \leq f(x)\ \forall x \in U$, alors $h(x) \leq C(f)(x)\ \forall x \in U$.*

Autrement dit, l'enveloppe convexe d'une fonction f sur un ensemble convexe U est donnée par le maximum en chaque point de toutes les fonctions convexes minorant la fonction f sur U, c'est à dire c'est la plus grande fonction convexe plus petite que f sur U.

Nous allons montrer que la fonction minorante $c_K^*(x)$ qu'on vient de construire à l'aide de notre programme (5.23) converge vers la fonction $C(p)(x)$ sur $[0,1]^n$ quand K tend vers l'infini [2]. Plus précisément, on a le résultat suivant :

Proposition 17. *Pour tout $K \geq \delta$:*

$$0 \leq C(p)(x) - c_K^*(x) \leq O(\tfrac{1}{k_1} + \cdots + \tfrac{1}{k_n}).$$

Démonstration. On commence tout d'abord par montrer que la fonction c_K^* est convexe sur $[0,1]^n$.
Soit x_1 et x_2 deux éléments de $[0,1]^n$ et $\lambda \geq 0$. Montrons que :

$$c_K^*(\lambda x_1 + (1-\lambda)x_2) \leq \lambda c_K^*(x_1) + (1-\lambda)c_K^*(x_2).$$

En effet, notons par ${z_{I,1}}^*$, $I \leq K$, la solution optimale de (5.23) pour $x = x_1$ et ${z_{I,2}}^*$, $I \leq K$, la solution optimale de (5.23) pour $x = x_2$. Ainsi on a :

$$c_K^*(x_1) = \sum_{I \leq K} \hat{p}_{I,K} {z_{I,1}}^* \text{ où } x_1 = \sum_{I \leq K} {z_{I,1}}^* \frac{I}{K},$$

$$c_K^*(x_2) = \sum_{I \leq K} \hat{p}_{I,K} {z_{I,2}}^* \text{ où } x_2 = \sum_{I \leq K} {z_{I,2}}^* \frac{I}{K}.$$

Par conséquent on obtient :

$$\lambda c_K^*(x_1) + (1-\lambda)c_K^*(x_2) = \sum_{I \leq K} \hat{p}_{I,K}(\lambda {z_{I,1}}^* + (1-\lambda){z_{I,2}}^*).$$

Posons $X = \lambda x_1 + (1-\lambda)x_2$. En utilisant les expressions de x_1 et x_2 on aura :

$$X = \sum_{I \leq K} (\lambda {z_{I,1}}^* + (1-\lambda){z_{I,2}}^*) \frac{I}{K}.$$

Par suite, par définition de $c_K^*(X)$ on aura :

$$c_K^*(X) \leq \sum_{I \leq K} \hat{p}_{I,K}(\lambda {z_{I,1}}^* + (1-\lambda){z_{I,2}}^*).$$

2. Tendre $K = (k_1, \ldots, k_n)$ vers l'infini veut dire tendre chaque k_i vers l'infini.

Cela achève la preuve de la convexité de la fonction c_K^* sur $[0,1]^n$.
D'autre part, grâce aux propriétés des polynômes de Bernstein, on peut montrer que pour tout $x \in [0,1]^n$ fixé, l'ensemble $(B_{K,I}(x))_{I \leq K}$ est admissible pour (5.23). Par conséquent, on déduit que $c_K^*(x) \leq p(x)$ et donc par définition de l'enveloppe convexe d'une fonction on aura :

$$c_K{}^*(x) \leq C(p)(x) \text{ pour tout } x \in [0,1]^n.$$

Maintenant, soit $x \in [0,1]^n$, et soit $z_I{}^*$, $I \leq K$, la solution optimale de (5.23) :

$$c_K^*(x) = \sum_{I \leq K} \hat{p}_{I,K} z_I{}^* \text{ et } x = \sum_{I \leq K} z_I{}^* \frac{I}{K}.$$

En utilisant les propriétés de l'enveloppe convexe d'une fonction on a :

$$\begin{aligned} C(p)(x) &= C(p)\left(\textstyle\sum_{I \leq K} \frac{I}{K} z_I{}^*\right) \\ &\leq \sum_{I \leq K} C(p)\left(\tfrac{I}{K}\right) z_I{}^* \leq \sum_{I \leq K} p\left(\tfrac{I}{K}\right) z_I{}^*. \end{aligned}$$

Il en découle que :

$$0 \leq C(p)(x) - c_K^*(x) \leq \sum_{I \leq K} (p(\tfrac{I}{K}) - \hat{p}_{I,K}) z_I{}^*.$$

Enfin, en utilisant (5.22), on a pour tout $x \in [0,1]^n$:

$$0 \leq C(p)(x) - c_K^*(x) \leq O(\tfrac{1}{k_1} + \cdots + \tfrac{1}{k_l}),$$

ce qui complète la preuve. □

Maintenant notons p_C^* la valeur optimale du problème d'optimisation suivant :

$$\begin{array}{ll} \text{minimiser} & C(p)(x) \\ \text{s.c} & x \in [0,1]^n, \\ & Ax \leq b. \end{array} \tag{5.24}$$

Grâce à la Proposition 17, on déduit facilement le résultat de convergence suivant :

Théorème 9. *Pour tout $K \geq \delta$,*

$$0 \leq p_C^* - p_K^* \leq O(\tfrac{1}{k_1} + \cdots + \tfrac{1}{k_n}).$$

Démonstration. Le résultat est une conséquence immédiate de la Proposition 17. □

Grâce à ce théorème, on peut montrer que dans le cas sans contraintes, on converge vers la valeur optimale p^* quand K tend vers l'infini. Ceci est dû au fait que le minimum d'une fonction sur un ensemble convexe U coïncide avec le minimum de son enveloppe convexe (par respect de U) sur U. Plus précisément, on a le théorème suivant (voir [89], Théorème 4 page 149).

Théorème 10. *Soit f une fonction continue sur un ensemble convexe U et soit $g = C_U(f)$ son enveloppe convexe sur U alors on a :*

$$\min_{x\in U} f(x) = \min_{x\in U} g(x)$$

Démonstration. La démonstration repose essentiellement sur le fait que l'enveloppe convexe d'une fonction est la plus grande fonction convexe qui minore cette fonction. En effet, soit $f^* = f(x^*) = \min_{x\in U} f(x)$. On sait déjà que $g(x^*) \leq f(x^*)$ puisque g est l'enveloppe convexe de f. Montrons de plus qu'on peut pas avoir $g(x^*) < f(x^*)$. Supposons que $g(x^*) < f(x^*)$ et considérons la fonction constante $h(x) = f(x^*)$. h est bien une fonction convexe qui minore f ainsi on aura $h(x) \leq g(x^*)$ d'où l'absurdité. Pour achever la démonstration, il suffit de montrer de la même manière que $g(x) \geq \min_{x\in U} f(x)$ pour tout $x \in U$. □

Ainsi, grâce à ce théorème, on déduit que dans le cas sans contraintes p_K^* converge vers p^* puisqu'on aura $p_C^* = p^*$.
Autrement dit, quand on augmente le degré K du polynôme p, on augmente la précision de la borne inférieure p_K^*. Cependant, cette borne ne peut pas être meilleure que p_C^*, qui est la valeur optimale d'une version convexifiée du problème (5.15). De plus, si p est convexe où si on n'a pas de contraintes, alors on peut approcher arbitrairement la valeur optimale p^*.

Remarque 13. *Comme on l'a signalé au début de cette section, la preuve reste valable dans le cas où ${p_K}^*$ est la valeur optimale de la relaxation :*

$$\begin{array}{lll} \textit{minimiser} & \sum_{I\leq K} \hat{p}_{I,K} z_I & \\ s.c & z_I \in \mathbb{R}, & I \leq K, \\ & z_I \geq 0, & I \leq K, \\ & \sum_{I\leq K} z_I = 1, & \\ & \sum_{I\leq \delta} (A\tfrac{I}{\delta}) z_I \leq b.. & \end{array}$$

Par conséquent, en utilisant l'équivalence établie dans le chapitre précédent, on déduit que le résultat de convergence reste valable dans le cas de la relaxation (5.6) donnée à l'aide de la forme polaire.

5.5 Appendice

5.5.1 Preuve de la Proposition 15

Démonstration. D'abord, on commence par écrire le problème (5.10) comme étant un problème de minimisation :

$$\begin{array}{lll} \text{minimiser} & -t & \\ \text{s.c} & t \in \mathbb{R},\ \lambda \in \mathbb{R}^m,\ \mu \in \mathbb{R}^r, & \\ & t - \hat{p}_{I,\delta} - \sum_{i=1}^{m} \lambda_i \hat{g}_{i\,I,\delta} - \sum_{j=1}^{r} \mu_j \hat{h_j}_{I,\delta} \leq 0, & I \leq \delta, \\ & -\lambda_i \leq 0, & i = 1, \ldots, m. \end{array} \tag{5.25}$$

Ainsi,le Lagrangien de ce problème est :

$$L'(t,\lambda,\mu,z,\theta) = -t + \sum_{I\leq\delta} z_I \left(t - \hat{p}_{I,\delta} - \sum_{i=1}^{m} \lambda_i \hat{g}_{iI,\delta} - \sum_{j=1}^{r} \mu_j \hat{h}_{jI,\delta} \right) - \sum_{i=1}^{m} \theta_i \lambda_i$$

avec $z_I \geq 0$, pour tout $I \leq \delta$, et $\theta_i \geq 0$, pour tout $i = 1,\ldots,m$. Ceci peut être reformulé comme suit :

$$\begin{aligned} L'(t,\lambda,\mu,z,\theta) &= t\left(\sum_{I\leq\delta} z_I - 1\right) - \sum_{i=1}^{m} \lambda_i \left(\theta_i + \sum_{I\leq\delta} z_I \hat{g}_{iI,\delta}\right) \\ &\quad - \sum_{j=1}^{r} \mu_j \left(\sum_{I\leq\delta} z_I \hat{h}_{jI,\delta}\right) - \sum_{I\leq\delta} z_I \hat{p}_{I,\delta}. \end{aligned}$$

Ainsi, en minimisant $L'(t,\lambda,\mu,z,\theta)$ pour $t \in \mathbb{R}$, $\lambda \in \mathbb{R}^m$ et $\mu \in \mathbb{R}^r$, on obtient

$$\min_{t\in\mathbb{R},\lambda\in\mathbb{R}^m,\mu\in\mathbb{R}^r} L'(t,\lambda,\mu,z,\theta) = \begin{cases} -\sum_{I\leq\delta} z_I \hat{p}_{I,\delta}, & \text{si } \begin{cases} \sum_{I\in\delta} z_I = 1, \\ \theta_i + \sum_{I\leq\delta} z_I \hat{g}_{iI,\delta} = 0, i \in I_m, \\ \sum_{I\leq\delta} z_I \hat{h}_{jI,\delta} = 0, j \in I_r. \end{cases} \\ -\infty \text{ sinon} \end{cases}$$

où pour un entier $k \in \mathbb{N}$, $I_k = \{1,\ldots,k\}$.
Maintenant, en maximisant avec $z_I \geq 0$, pour tout $I \leq \delta$, et $\theta_i \geq 0$, pour tout $i = 1,\ldots,m$ on obtient le dual du problème (5.25)

$$\begin{aligned} \text{maximiser} \quad & -\sum_{I\leq\delta} z_I \hat{p}_{I,\delta} \\ \text{s.c} \quad & z_I \geq 0, && I \leq \delta, \\ & \theta_i \geq 0, && i = 1,\ldots,m, \\ & \sum_{I\leq\delta} z_I = 1, \\ & \theta_i + \sum_{I\leq\delta} z_I \hat{g}_{iI,\delta} = 0, && i = 1,\ldots,m, \\ & \sum_{I\leq\delta} z_I \hat{h}_{jI,\delta} = 0, && j = 1,\ldots,r. \end{aligned}$$

La variable θ peut être supprimée et le problème devient

$$\begin{aligned} \text{maximiser} \quad & -\sum_{I\leq\delta} z_I \hat{p}_{I,\delta} \\ \text{s.c} \quad & z_I \geq 0, && I \leq \delta, \\ & \sum_{I\leq\delta} z_I = 1, \\ & \sum_{I\leq\delta} z_I \hat{g}_{iI,\delta} \leq 0, && i = 1,\ldots,m, \\ & \sum_{I\leq\delta} z_I \hat{h}_{jI,\delta} = 0, && j = 1,\ldots,r. \end{aligned} \tag{5.26}$$

Ainsi le problème est équivalent à (5.11). De plus, comme (5.25) est un programme linéaire, on a bien la dualié forte (voir [32]) donc les solutions de (5.25) et (5.26) sont égaux. Par conséquent (5.11) et (5.3) sont égaux. □

5.5.2 Preuve du Théorème 6

Démonstration. Notons par d^* la valeur optimale de (5.5) et $\overline{d}^*$ la valeur optimale de (5.6), on veut montrer que $d^* = \overline{d}^*$. Comme précédemment, vu que (5.5) et (5.6) sont des programmes linéaires, d^* et $\overline{d}^*$ peuvent être calculés à partir des problèmes duaux. En gardant les notations des formes polaires et en utilisant la Proposition 15, le duale de (5.5) est

$$\begin{array}{lll}
\text{minimiser} & \displaystyle\sum_{v\in V'} y_v p_\delta(v) & \\
\text{s.c} & y \in \mathbb{R}^{2^{\delta_1+\cdots+\delta_n}}, & \\
& y_v \geq 0, & v \in V', \\
& \displaystyle\sum_{v\in V'} y_v = 1, & \\
& a_i' \cdot \displaystyle\sum_{v\in V'} y_v v \leq b_i, & i = 1,\ldots,m, \\
& c_j' \cdot \displaystyle\sum_{v\in V'} y_v v = d_j, & j = 1,\ldots,r, \\
& e_{k,l} \cdot \displaystyle\sum_{v\in V'} y_v v = 0, \quad k \in \{1,\ldots,n\},\ l \in \{1,\ldots,\delta_k - 1\}. &
\end{array} \tag{5.27}$$

D'une façon similaire, on montre que le duale de (5.6) est

$$\begin{array}{lll}
\text{minimiser} & \displaystyle\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} p_\delta(\overline{v}) & \\
\text{s.c} & z \in \mathbb{R}^{(\delta_1+1)\times\cdots\times(\delta_n+1)}, & \\
& z_{\overline{v}} \geq 0, & \overline{v} \in \overline{V'}, \\
& \displaystyle\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} = 1, & \\
& a_i' \cdot \displaystyle\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}}\overline{v} \leq b_i, & i = 1,\ldots,m, \\
& c_j' \cdot \displaystyle\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}}\overline{v} = d_j, & j = 1,\ldots,r.
\end{array} \tag{5.28}$$

D'abord, on va montrer que $\overline{d}^* \leq d^*$. Soit $y \in \mathbb{R}^{2^{\delta_1+\cdots+\delta_n}}$ un point admissible du problème (5.27) tel que $d^* = \sum_{v\in V'} y_v p_\delta(v)$. Pour $\overline{v} \in \overline{V'}$, soit $z_{\overline{v}} = \sum_{v\in\overline{v}} y_v$, il est évident que $z_{\overline{v}} \geq 0$. De plus,

$$\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} = \sum_{\overline{v}\in\overline{V'}} \sum_{v\in\overline{v}} y_v = \sum_{v\in V'} y_v = 1.$$

Puisque pour tout $v \in \overline{v}$, $i = 1,\ldots,m$ et tout $j = 1,\ldots,r$, on a

$$a_i' \cdot v = a_i' \cdot \overline{v} \text{ et } c_j' \cdot v = c_j' \cdot \overline{v}$$

par suite,

$$a_i' \cdot \sum_{\overline{v}\in\overline{V'}} z_{\overline{v}}\overline{v} = \sum_{\overline{v}\in\overline{V'}} \sum_{v\in\overline{v}} y_v (a_i' \cdot \overline{v}) = \sum_{v\in V'} y_v (a_i' \cdot v) \leq b_i,$$

et

$$c'_j \cdot \sum_{\overline{v}\in\overline{V'}} z_{\overline{v}}\overline{v} = \sum_{\overline{v}\in\overline{V'}}\sum_{v\in\overline{v}} y_v(c'_j \cdot \overline{v}) = \sum_{v\in V'} y_v(c'_j \cdot v) = d_j.$$

Ainsi, z est admissible pour le problème (5.28). Finalement, comme pour tout $v \in \overline{v}$, $p_\delta(v) = p_\delta(\overline{v})$, on en déduit que

$$\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} p_\delta(\overline{v}) = \sum_{\overline{v}\in\overline{V'}}\sum_{v\in\overline{v}} y_v p_\delta(\overline{v}) = \sum_{v\in V'} y_v p_\delta(v) = d^*.$$

Ainsi, $\overline{d}^* \leq d^*$. Maintenant on va montrer que $d^* \leq \overline{d}^*$. Soit $z \in \mathbb{R}^{(\delta_1+1)\times\cdots\times(\delta_n+1)}$ un point admissible du problème (5.28) tel que $\overline{d}^* = \sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} p_\delta(\overline{v})$. Soit $n(\overline{v})$ le nombre de sommets $v \in V'$ tels que $v \in \overline{v}$. Pour tout $v \in \overline{v}$, posons $y_v = z_{\overline{v}}/n(\overline{v})$. Il est évident que $y_v \geq 0$ et

$$\sum_{v\in V'} y_v = \sum_{\overline{v}\in\overline{V'}}\sum_{v\in\overline{v}} y_v = \sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} = 1.$$

On a aussi

$$a'_i \cdot \sum_{v\in V'} y_v v = \sum_{\overline{v}\in\overline{V'}}\sum_{v\in\overline{v}} y_v(a'_i \cdot v) = \sum_{\overline{v}\in\overline{V'}}\sum_{v\in\overline{v}} \frac{z_{\overline{v}}}{n(\overline{v})}(a'_i \cdot \overline{v}) = \sum_{\overline{v}\in\overline{V'}} z_{\overline{v}}(a'_i \cdot \overline{v}) \leq b_i,$$

et

$$c'_j \cdot \sum_{v\in V'} y_v v = \sum_{\overline{v}\in\overline{V'}}\sum_{v\in\overline{v}} y_v(c'_j \cdot v) = \sum_{\overline{v}\in\overline{V'}}\sum_{v\in\overline{v}} \frac{z_{\overline{v}}}{n(\overline{v})}(c'_j \cdot \overline{v}) = \sum_{\overline{v}\in\overline{V'}} z_{\overline{v}}(c'_j \cdot \overline{v}) = d_j.$$

De plus,

$$e_{k,l} \cdot \sum_{v\in V'} y_v v = \sum_{\overline{v}\in\overline{V'}} e_{k,l} \cdot \sum_{v\in\overline{v}} \frac{z_{\overline{v}}}{n(\overline{v})} v = \sum_{\overline{v}\in\overline{V'}} \frac{z_{\overline{v}}}{n(\overline{v})} \left(e_{k,l} \cdot \sum_{v\in\overline{v}} v \right).$$

En remarquant que pour tout $\overline{v} \in \overline{V'}$, $e_{k,l} \cdot \sum_{v\in\overline{v}} v = 0$, on montre que $e_{k,l} \cdot \sum_{v\in V'} y_v v = 0$. Ainsi, y est admissible pour le problème (5.27). Enfin,

$$\sum_{v\in V'} y_v p_\delta(v) = \sum_{\overline{v}\in\overline{V'}}\sum_{v\in\overline{v}} \frac{z_{\overline{v}}}{n(\overline{v})} p_\delta(\overline{v}) = \sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} p_\delta(\overline{v}) = \overline{d}^*.$$

Cela prouve que $d^* \leq \overline{d}^*$, d'où l'égalité. □

5.5.3 Preuve du Théorème 7

Démonstration. La preuve précédente se généralise facilement dans le cas général (semi-algébrique). En effet, on montre de la même façon que les valeurs optimales

d^* et $\overline{d}^*$ des programmes linéaires respectifs :

$$\begin{array}{lll}
\text{minimiser} & \displaystyle\sum_{v\in V'} y_v p_\delta(v) & \\
\text{s.c} & y \in \mathbb{R}^{2^{\delta_1+\cdots+\delta_n}}, & \\
& y_v \geq 0, & v \in V', \\
& \displaystyle\sum_{v\in V'} y_v = 1, & \\
& \displaystyle\sum_{v\in V'} y_v g_{i\delta}(v) \leq 0, & i = 1,\ldots,m, \\
& \displaystyle\sum_{v\in V'} y_v h_{j\delta}(v) = 0, & j = 1,\ldots,r, \\
& e_{k,l}\cdot \displaystyle\sum_{v\in V'} y_v v = 0, \quad k \in \{1,\ldots,n\},\ l \in \{1,\ldots,\delta_k - 1\}, &
\end{array} \tag{5.29}$$

et

$$\begin{array}{lll}
\text{minimiser} & \displaystyle\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} p_\delta(\overline{v}) & \\
\text{s.c} & z \in \mathbb{R}^{(\Delta_1+1)\times\cdots\times(\delta_n+1)}, & \\
& z_{\overline{v}} \geq 0, & \overline{v} \in \overline{V'}, \\
& \displaystyle\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} = 1, & \\
& \displaystyle\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} g_{i\delta}(\overline{v}) \leq 0, & i = 1,\ldots,m, \\
& \displaystyle\sum_{\overline{v}\in\overline{V'}} z_{\overline{v}} h_{j\delta}(\overline{v}) = 0, & j = 1,\ldots,r,
\end{array} \tag{5.30}$$

sont égaux.
L'approche est exactement celle présentée dans la preuve précédente, il suffira d'utiliser en plus le fait que pour tout $v \in \overline{v}$, $i = 1,\ldots,m$ et tout $j = 1,\ldots,r$ on peut écrire :

$$g_{i\delta}(v) = g_{i\delta}(\overline{v}) \text{ et } h_{j\delta}(v) = h_{j\delta}(\overline{v}).$$

□

Troisième partie

Applications dans le cadre des systèmes dynamiques polynomiaux

Chapitre 6

Analyse d'atteignabilité des systèmes dynamiques polynomiaux

L'analyse d'atteignabilité a été un sujet de recherche majeur dans le domaine des systèmes hybrides pendant plus d'une décennie. Des progrès spectaculaire ont été accomplis au cours des dernières années pour la classe des systèmes hybrides où les dynamiques continues sont décrites par des équations différentielles affines [78, 86, 107]. Cependant, gérer efficacement les systèmes dynamiques non-linéaires présente un problème beaucoup plus difficile qui doit être résolu afin de permettre l'utilisation de la vérification des systèmes hybrides dans de nombreuses applications du monde réel. Outre son application à la vérification des systèmes hybrides, l'analyse d'atteignabilité des systèmes continus non linéaires est aussi motivée par ses nombreuses applications potentielles dans le domaine des systèmes biologiques [85, 50, 191] ainsi que la vérification des circuits analogiques [67, 190].

Dans ce chapitre, nous proposons une nouvelle approche pour l'analyse d'atteignabilité des systèmes dynamiques polynomiaux en temps discret. En représentant les ensembles atteignables par des modèles polyédraux, nous montrons que l'analyse d'atteignabilité peut être réduite à un ensemble de problèmes d'optimisation de polynômes sur des polytopes bornés. Pour résoudre ces problèmes, on va utiliser la relaxation de Bernstein introduite dans la partie précédente. En effet comme on l'a vu précédemment, les valeurs optimales données par les relaxations linéaires offrent des bornes inférieures (dans le cas d'une minimisation) aux valeurs optimales des problèmes originaux. Ainsi, à l'aide de cette approche on va pouvoir calculer des sur-approximations garanties des ensembles atteignables associés à nos systèmes dynamiques polynomiaux en temps discret. On présentera ensuite des techniques assurant un choix approprié de ces modèles polyédraux et on discutera du compromis entre la complexité de la procédure et la qualité de la sur-approximation. Enfin, nous montrerons le mérite de notre approche sur plusieurs exemples.

Notons que ce travail a été effectué en collaboration avec des chercheurs en informatique du laboratoire Verimag [1].

1. Romain Testylier et Thao Dang

6.1 Algorithme d'atteignabilité pour des systèmes dynamiques polynomiaux

On va considérer le système dynamique à temps discret suivant :

$$x_{k+1} = f(x_k),\ k \in \mathbb{N},\ x_k \in \mathbb{R}^n,\ x_0 \in X_0 \tag{6.1}$$

où $f : \mathbb{R}^n \to \mathbb{R}^n$ est un polynôme de degrés $\delta = (\delta_1, \ldots, \delta_n)$ et X_0 un polytope borné de $\mathbb{R}^n$. Dans ce qui suit, on va s'intéresser à l'analyse d'atteignabilité du système (6.1) en temps borné. Ceci consiste à calculer la suite $X_k \subseteq \mathbb{R}^n$ des ensembles atteignables à l'instant k du système jusqu'à un certain temps $K \in \mathbb{N}$. Il est facile de vérifier que cette suite vérifie la relation de récurrence suivante $X_{k+1} = f(X_k)$. La difficulté provient du fait que même si X_0 est un polytope, les autres éléments de la suite ne sont pas forcément des polytopes. De plus, ils ne sont généralement même pas convexes.

Pour résoudre ce problème, on va sur-approximer ces ensembles en utilisant des polytopes bornés $\overline{X}_k$. En effet, ces sur-approximations peuvent clairement être calculées par induction en posant $\overline{X}_0 = X_0$ et en assurant la relation $f(\overline{X}_k) \subseteq \overline{X}_{k+1}$ pour tout $k = 0, \ldots, K-1$. Par conséquent, on va se concentrer tout d'abord sur le calcul d'une sur-approximation polyédrale $\overline{X}_{k+1}$ de l'image du polytope borné $\overline{X}_k$ par le champ de vecteurs polynomial f.

Maintenant, on va formuler le problème d'atteignabilité que l'on veut résoudre :

Problème 1. *Etant donné un polytope borné $\overline{X}_k$, on veut calculer un ensemble polyédral $\overline{X}_{k+1}$ vérifiant $f(\overline{X}_k) \subseteq \overline{X}_{k+1}$.*

Dans ce qui suit on propose une solution à ce problème basée sur l'utilisation des modèles polyédraux et l'optimisation polynomiale.

6.1.1 Modèles polyédraux

Pour représenter $\overline{X}_{k+1}$, on va utiliser les modèles polyédraux. On va commencer par quelques rappels concernant ces modèles. En effet, les modèles polyédraux sont couramment utilisés dans l'analyse statique des programmes pour calculer des invariants [41]. Ils sont aussi utilisés comme étant des domaines abstraits pour la représentation de l'ensemble d'états (voir par exemple [158] et les références qui y sont citées).

Dans notre contexte, un modèle est un ensemble de fonctions linéaires dans $\mathbb{R}^n$; pour cela, on le désigne par une matrice $A \in \mathbb{R}^{m \times n}$. Etant donné un modèle A ainsi qu'un vecteur de coefficients $b \in \mathbb{R}^m$, on définit le polyèdre

$$Poly(A, b) = \{x \in \mathbb{R}^n |\ Ax \leq b\}$$

où les inégalités doivent être vues composante par composante.
En faisant varier la valeur de b, on obtient une famille de polyèdres correspondant au modèle A. En effet, le modèle A définit les directions des faces et le vecteur b définit les positions. L'avantage de l'utilisation des modèles polyédraux à la place d'autres ensembles convexes réside surtout dans le fait que les opérations booléennes (union,

intersection) ainsi que d'autres opérations géométriques peuvent être effectuées plus efficacement.

Dans ce qui suit, on suppose que $\overline{X}_{k+1}$ est un polytope borné avec un modèle donné $A_{k+1} \in \mathbb{R}^{m\times n}$. On ajoute l'indice $k+1$ pour souligner que le modèle peut ne pas être le même pour tous les polytopes $\overline{X}_{k+1}$, $k = 0, \ldots, K-1$. Ainsi,

$$\overline{X}_{k+1} = Poly(A_{k+1}, b_{k+1})$$

où le vecteur $b_{k+1} \in \mathbb{R}^m$ doit être déterminé à chaque itération. Le choix du modèle A_{k+1} sera discuté ultérieurement.

6.1.2 Formulation basée sur l'optimisation polynomiale

D'après la discussion précédente, le calcul de l'ensemble $\overline{X}_{k+1}$ se réduit à la détermination du vecteur position b_{k+1} si la matrice de direction A_{k+1} (le modèle) est fixée à l'avance. Notons par $b_{k+1,i}$ la i-ème composante du vecteur b_{k+1} et $A_{k+1,i}$ la i-ème ligne de la matrice A_{k+1}.

Lemme 5. *Si pour tout* $i = 1, \ldots, m$

$$-b_{k+1,i} \leq \min_{x\in\overline{X}_k} -A_{k+1,i} f(x) \tag{6.2}$$

alors $f(\overline{X}_k) \subseteq \overline{X}_{k+1}$ *où* $\overline{X}_{k+1} = Poly(A_{k+1}, b_{k+1})$.

Démonstration. Soit $y \in f(\overline{X}_k)$. Pour $i = 1, \ldots, m$, il est clair qu'on a

$$A_{k+1,i} y \leq \max_{x\in\overline{X}_k} A_{k+1,i} f(x).$$

Ainsi, en remarquant que $\max_{x\in\overline{X}_k} A_{k+1,i} f(x) = -\min_{x\in\overline{X}_k} -A_{k+1,i} f(x)$ et d'après (6.2) on obtient $A_{k+1,i} y \leq b_{k+1,i}$. □

Par conséquent, pour calculer les sur-approximations polyédrales $\overline{X}_{k+1}$, il nous suffit de fournir pour tout k une borne inférieure au POP : $\min_{x\in\overline{X}_k} -A_{k+1,i} f(x)$ où la fonction objectif $-A_{k+1,i} f(x)$ est un polynôme et l'ensemble des contraintes est un polytope borné. Par suite, on peut utiliser par exemple la relaxation linéaire donnée à l'aide des coefficients de Bernstein introduite dans la Section 5.2. Notons à ce stade que la relaxation de Bernstein nécessite qu'on se ramène à la boîte unité $[0,1]^n$ et que l'on doit donc fixer à l'avance une boîte ou plus généralement un parallélotope englobant notre polytope. Tout cela sera détaillé dans notre algorithme d'atteignabilité.

6.1.3 Algorithme d'atteignabilité

Selon la discussion précédente, pour chaque pas de temps $k \in \mathbb{N}$ on doit calculer une borne inférieure à la valeur

$$p_{k+1,i}{}^* = \min_{x\in\overline{X}_k} -A_{k+1,i} f(x) \text{ pour tout } i = 1, \ldots, m.$$

Pour faire cela, on propose un algorithme avec essentiellement trois étapes. Dans la première étape nous calculons un parallélotope englobant notre polytope $\overline{X}_k$ dit parallélotope englobant. La seconde consiste à faire un changement de variable pour se ramener au cas d'un problème d'optimisation d'un polynôme sur un polytope inclus dans la boîte unité $[0,1]^n$. Dans la dernière étape, on utilisera la relaxation linéaire donnée par le Théorème 8 pour pouvoir calculer une borne inférieure et par conséquent trouver une sur-approximation de l'ensemble atteignable.

Etape 1 : Calcul d'un parallélotope englobant

Comme $\overline{X}_k$ est un polytope borné de $\mathbb{R}^n$ on peut l'écrire sous la forme $\overline{X}_k \cap Q_k$ où Q_k est un parallélotope englobant $\overline{X}_k$ donné par $Q_k = Poly(\tilde{C}_k, \tilde{d}_k)$ avec

$$\tilde{C}_k = \begin{bmatrix} C_k \\ -C_k \end{bmatrix},\ \tilde{d}_k = \begin{bmatrix} \overline{d}_k \\ -\underline{d}_k \end{bmatrix}.$$

$C_k \in \mathbb{R}^{n\times n}$ est une matrice inversible, $\underline{d}_k \in \mathbb{R}^n$ et $\overline{d}_k \in \mathbb{R}^n$. On suppose que la matrice des directions C_k est donnée comme entrée de l'algorithme (une méthode décrivant comment calculer cette matrice sera donnée ultérieurement). On calcule les composantes $\overline{d}_{k,i}$ et $\underline{d}_{k,i}, i = 1,\dots n$ du vecteur position $\tilde{d}_k$ comme étant les valeurs optimales des programmes linéaires suivants :

$$\overline{d}_{k,i} = \max_{x\in\overline{X}_k} C_{k,i}\cdot x \text{ et } \underline{d}_{k,i} = \max_{x\in\overline{X}_k} -C_{k,i}\cdot x \quad \forall i = 1,\dots n.$$

Etape 2 : Changement de variable

Maintenant, on va procéder au changement de variable suivant $x = q_k(y)$ où l'application affine $q_k : \mathbb{R}^n \to \mathbb{R}^n$ est donnée par

$$q_k(y) = {C_k}^{-1} D_k y + {C_k}^{-1}\underline{d_k}$$

où D_k est la matrice diagonale d'entrées $\overline{d}_{k,i} - \underline{d}_{k,i}$. Le changement de variable q_k renvoie essentiellement la boîte unité $[0,1]^n$ vers le parallélotope Q_k. On définit ensuite le polynôme g_k comme étant $g_k(y) = f(q_k(y))$. Finalement, la nouvelle matrice de direction $A'_k \in \mathbb{R}^{m\times n}$ et le nouveau vecteur position $b'_k \in \mathbb{R}^m$ sont donnés par

$$A'_k = A_k {C_k}^{-1} D_k,\ b'_k = b_k - A{C_k}^{-1}\underline{d}.$$

Remarque 14. *Il est évident que g_k est un polynôme. Concernant les degrés $\delta' = (\delta'_1,\dots,\delta'_n)$ de g_k des variables $y_1,\dots,y_n$, on doit considérer deux cas particuliers différents dépendant de la nature du parallélotope Q_k.*
Si Q_k est une boîte alignée avec les axes (c'est à dire si C_k est une matrice diagonale), alors les degrés de g_k sont les mêmes que ceux de f d'où $\delta' = \delta$. Ceci n'est pas le cas quand Q_k n'est pas aligné avec les axes (boîte orientée). Dans ce cas, le changement de variable augmente généralement les degrés du polynôme et g_k peut être vu comme étant un polynôme de degrés $\delta' = (\delta_{max},\dots,\delta_{max})$ où $\delta_{max} = \sum_{i=1}^{n}\delta_i$.

Etape 3 : Résolution du problème d'optimisation

Après le changement de variable, on trouve facilement le POP équivalent suivant :

$$\begin{array}{ll} \text{minimiser} & -A_{k+1,i} \cdot g_k(y) \\ \text{s.c} & y \in [0,1]^n, \\ & A'_k y \leq b'_k. \end{array}$$

Par suite, il suffit d'appliquer le Théorème 8 pour que la borne inférieure $-b_{k+1,i}$ soit trouvée.
L'ensemble atteignable à l'instant $k+1$ sera ainsi $\overline{X}_{k+1} = Poly(A_{k+1}, b_{k+1})$.

6.2 Quelques compléments utiles

Dans la section précédente, un algorithme concernant l'analyse d'atteignabilité des systèmes dynamiques polynomiaux en temps discret a été fourni. Cependant, l'algorithme nécessite que les modèles A_k des polytopes $\overline{X}_k$ ainsi que les modèles C_k pour les parallélotopes Q_k soient donnés comme entrées à chaque itération. Un choix trivial sera de prendre les directions du polytope initial X_0 pour tous les modèles A_k et de prendre C_k comme étant la matrice identité. L'inconvénient de ce choix est le fait qu'il ne prend pas en considération l'évolution de la dynamique du système. Pour cela, on va proposer une approche pour le choix des modèles A_k pour les polytopes $\overline{X}_k$ et C_k pour les parallélotopes Q_k. Ensuite, on proposera une méthode (autre que la méthode explicite) permettant de calculer efficacement les coefficients de Bernstein. Enfin, on achèvera cette section par une brève étude de complexité de notre approche ainsi qu'une comparaison avec les approches existantes.

6.2.1 Choix des modèles

Modèles dynamiques pour le polytope $\overline{X}_k$

On va proposer une approche itérative permettant de calculer des modèles dynamiques A_k pour les polytopes $\overline{X}_k$ c'est à dire des modèles prenant en compte la dynamique du système.
Pour cela, on suppose qu'on dispose d'une matrice de direction A_k (le modèle) pour le polytope $\overline{X}_k = \{x \in \mathbb{R}^n |\ A_k \cdot x \leq b_k\}$ et on propose une méthode permettant le calcul du modèle A_{k+1} associé au polytope $\overline{X}_{k+1}$.

En effet, on veut calculer après une itération la nouvelle matrice de direction A_{k+1} qui reflète le changement de la forme de $\overline{X}_k$ sous l'action du polynôme f. Pour ce faire, on utilise une approximation linéaire locale de la dynamique du système polynomial (6.1) donnée par le développement de Taylor du premier ordre autour du centroïde x_k^* du dernier polytope calculé $\overline{X}_k$:

$$f(x) \ \approx L_k(x) = f(x_k^*) + J(x_k^*)(x - x_k^*)$$

où J désigne la matrice Jacobienne de la fonction f. Ainsi, si on note $F_k = J(x_k^*)$ et $h_k = f(x_k^*) - J(x_k^*)x_k^*$, alors dans un voisinage de x_k^* la dynamique non linéaire peut

être grossièrement approché par $x_{k+1} = F_k x_k + h_k$. Supposons que F_k est inversible, cela donne $x_k = F_k^{-1} x_{k+1} - F_k^{-1} h_k$. En exprimant les contraintes sur x_k données par le polytope $\overline{X}_k$ en fonction de x_{k+1}, on obtient

$$A_k F_k^{-1} x_{k+1} \leq b_k + A_k F_k^{-1} h_k.$$

Ainsi, un choix raisonnable du modèle pour $\overline{X}_{k+1}$ peut être $A_{k+1} = A_k F_k^{-1}$. Ce nouveau modèle sera ensuite utilisé dans l'itération suivante pour le calcul du polytope $\overline{X}_{k+1}$ en utilisant la méthode décrite dans la section précédente. On peut aussi remarquer que ce choix implique que notre algorithme d'atteignabilité sera exact si f est une fonction affine. De plus, on peut voir à l'aide de La Figure 6.1 l'avantage de l'utilisation des modèles dynamiques calculés en utilisant l'approximation linéaire ci-dessus. En effet, l'erreur de la sur-approximation (en gris) dans le cas du modèle statique est beaucoup plus grande que celle du cas dynamique.

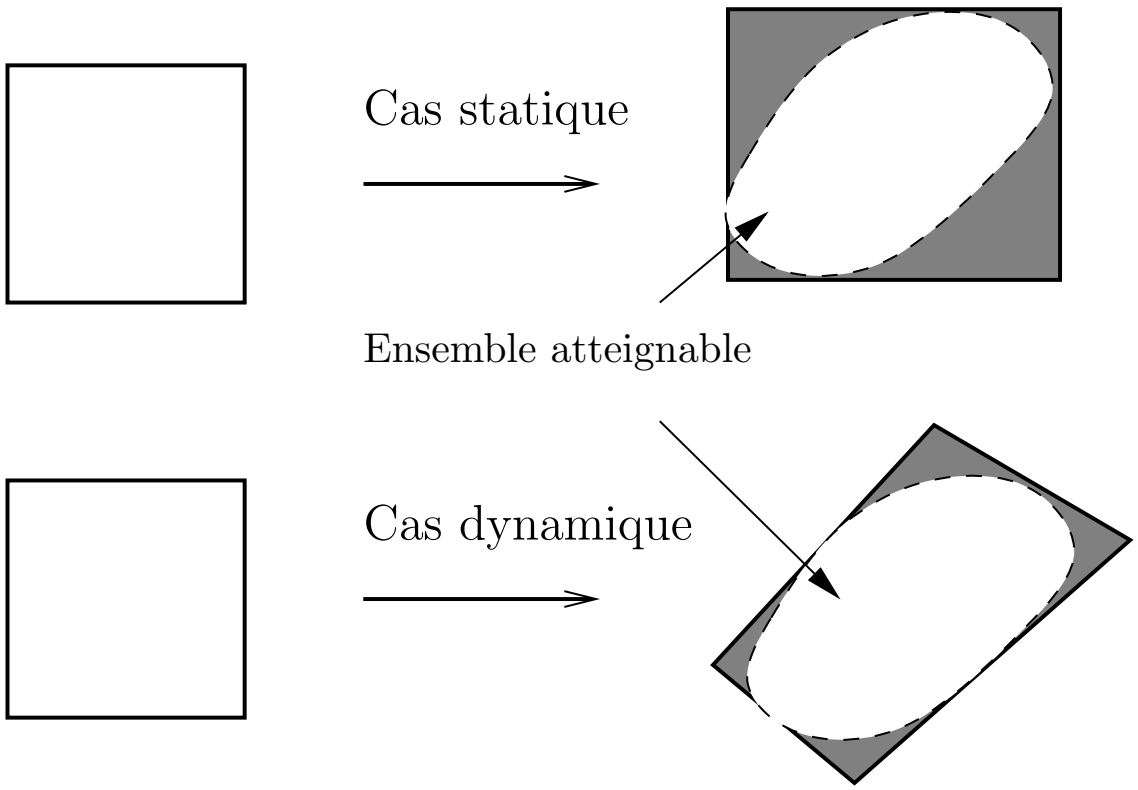

FIGURE 6.1 – Avantage de l'utilisation des modèles dynamiques à la place des modèles statiques.

Modèles dynamiques pour le parallélotope Q_k

Dans certains cas, il est utile de prendre des boîtes statiques alignées avec les axes pour Q_k (c'est à dire C_k est choisie comme étant la matrice identité pour tout entier $k = 0, \ldots, K$). Ce choix nous permet de préserver les degrés des polynômes

quand on fera notre changement de base (voir Remarque 14). Cependant, l'utilisation des boîtes statiques peut produire des erreurs d'approximation de plus en plus grandes. De façon similaire au polytopes $\overline{X}_k$, la précision sera meilleure si on utilise des modèles dynamiques qui prennent en considération la dynamique du système (essentiellement l'effet de rotation).

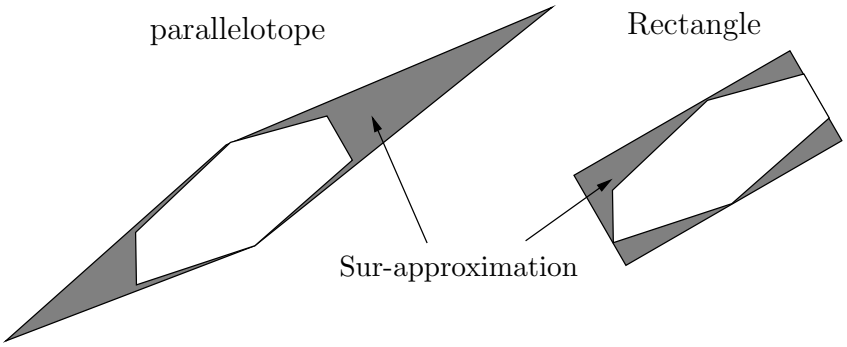

FIGURE 6.2 – Avantage de l'utilisation des boîtes orientées.

En fait, nous allons choisir Q_k comme étant une boîte rectangulaire orientée. En faisant ce choix, on évite quelques sur-approximations qui peuvent avoir lieu quand le parallélotope devient allongé, (voir Figure 6.2 pour une illustration de ce problème). On doit mentionner que l'image d'une boîte rectangulaire orientée Q_k par une application linéaire F_k n'est pas nécessairement une boîte rectangulaire orientée donc on ne peut pas utiliser directement la matrice F_k^{-1} calculée précédemment. Pour résoudre ce problème on va utiliser une technique populaire dans l'analyse par intervalle [127] basée sur la Décomposition QR des matrices. Essentiellement, F_k va être écrite comme étant un produit de deux matrices $F_k = \mathsf{Q_k R_k}$ où $\mathsf{Q_k}$ est une matrice orthogonale et $\mathsf{R_k}$ est une matrice triangulaire supérieure. Ainsi, pour choisir le modèle C_{k+1} associé à la boîte orientée Q_{k+1}, on applique notre matrice de rotation $\mathsf{Q_k}$; ce qui est équivalent à choisir le modèle $C_{k+1} = C_k \mathsf{Q_k^\top}$. Bien sûr, dans ce cas, on manipule des boîtes qui ne sont pas alignés avec les axes (boîtes orientées), ce qui entraîne des degrés supérieurs pour notre polynôme mais l'approximation sera moins conservatrice que celle utilisant des modèles statiques.

6.2.2 Calcul des coefficients de Bernstein

L'ingrédient principal théorique de notre approche est l'expansion des polynômes de Bernstein [21, 22]. En effet, étant donné le polynôme $g_k = f \circ q_k$ de degré δ' (obtenu après le changement de variables), on a vu que l'ensemble des coefficients $\{\hat{g}_{k\,I,\delta'} \in \mathbb{R}\ ,\ I \leq \delta'\}$ dans la base Bernstein peut être calculé explicitement à l'aide de la formule (3.6) du Chapitre 2.
On propose aussi une approche alternative pour le calcul des coefficients $\hat{g}_{k\,I} = \hat{g}_{k\,I,\delta'}$ en utilisant une interpolation aux points $\frac{J}{\delta'} = (\frac{j_1}{\delta'_1}, \ldots, \frac{j_n}{\delta'_n})$ pour $J \leq \delta'$:

$$\sum_{I \leq \delta'} \hat{g}_{k\,I} B_{\delta',I}(\tfrac{J}{\delta'}) = g_k(\tfrac{J}{\delta'}) = f(q_k(\tfrac{J}{\delta'})).$$

On dénote par $\mathsf{B}_{\delta'}$ la matrice dont les lignes sont indexées par $J \leq \delta'$ et les colonnes par $I \leq \delta'$ et dont les coefficients sont $B_{\delta',I}(\frac{J}{\delta'})$. De plus, soit $\hat{\mathsf{g}}_k$ le vecteur colonne indexé par $I \leq \delta'$ dont les composantes sont $\hat{g_k}_I$ et g_k le vecteur colonne indexé par $J \leq \delta'$ dont les composantes sont les $g_k(\frac{J}{\delta}) = f(q_k(\frac{J}{\delta}))$. Ainsi, en écrivant l'équation précédente en forme matricielle on aura :

$$\mathsf{B}_{\delta'}\hat{\mathsf{g}}_k = \mathsf{g}_k.$$

D'après la théorie standard d'interpolation polynomiale, la matrice $\mathsf{B}_{\delta'}$ est inversible et les coefficients de Bernstein sont donnés par

$$\hat{\mathsf{g}}_k = \mathsf{B}_{\delta'}^{-1}\mathsf{g}_k. \tag{6.3}$$

6.2.3 Complexité et comparaison avec d'autres méthodes

Complexité de l'approche

On a présenté deux approches permettant chacune de calculer la forme de Bernstein après un changement de variable. Les deux méthodes ont une complexité polynomiale en temps et en espace en $N_{\delta'} = (\delta'_1 + 1) \times \cdots \times (\delta'_n + 1)$. La taille de la matrice $\mathsf{B}_{\delta'}$ est $(\delta'_1 + 1) \times \cdots \times (\delta'_n + 1)$. Cependant, dans le contexte de l'analyse d'atteignabilité, la matrice inverse $\mathsf{B}_{\delta'}^{-1}$ doit être calculée une seule fois et peut être utilisée à chaque itération pour calculer les coefficients de Bernstein à l'aide d'un simple produit d'une matrice et d'un vecteur.

En fait, on va voir grâce à des expériences numériques que les deux méthodes de calcul ont leurs avantages en fonction de la nature du parallélotope Q. Si Q est une boîte alignée avec les axes, on a déjà vu que le changement de variable n'accroît pas le degré du polynôme. Par suite, la matrice $\mathsf{B}_{\delta'}$ aura une taille raisonnable et le calcul des coefficients $\hat{g_k}_I$ en utilisant l'équation (6.3) est généralement plus efficace. Par contre dans le cadre d'un parallélotope général, la matrice $\mathsf{B}_{\delta'}$ pourrait être beaucoup plus grande. Dans ce cas, puisque plusieurs coefficients f_i du polynôme f pourraient effectivement être égaux à zéro, il sera plus efficace de calculer explicitement le changement de variable puis d'utiliser (3.6) pour calculer les coefficients de Bernstein.

Le programme linéaire a aussi une complexité polynomiale en son nombre de variables de décision qui est aussi $N_{\delta'}$. Par conséquent, la complexité de toute la procédure est polynomiale en $N_{\delta'}$. Enfin, on remarque que $N_{\delta'}$ peut être beaucoup plus petit quand le Q est une boîte alignée avec les axes. En effet, dans ce cas $N_{\delta'} = (\delta_1 + 1) \times \cdots \times (\delta_n + 1)$ tandis que dans le cas d'une boîte orientée on a $N_{\delta'} = (\delta_{max} + 1)^n$. Ce point devra être pris en considération dans l'algorithme d'atteignabilité quand on fera le choix du modèle pour le parallélotope Q_{k+1}.

Comparaison avec d'autres méthodes

Dans le cadre de l'analyse d'atteignabilité, on peut aussi utiliser des méthodes d'analyse par intervalles [92] pour calculer l'ensemble atteignable. Cependant, ces méthodes sont généralement utilisées quand il s'agit de domaines rectangulaires. D'autre part, grâce à notre approche on arrive à obtenir des bornes qui sont plus fines que celles calculées en utilisant l'arithmétique des intervalles [120]. De plus,

les techniques de pré-conditionnement utilisées par les approches d'intervalles (voir [127]) peuvent aussi être utilisées dans notre approche comme on l'a déjà évoqué.

Par ailleurs, une approche populaire pour les systèmes non linéaires sera de caractériser l'ensemble atteignable en utilisant la formulation de Hamilton-Jacobi [121], puis résoudre les équations aux dérivées partielles résultantes qui nécessitent des calculs coûteux. Des résultats récents basés sur cette approche sont donnés dans [96].

Une autre approche existante est basée sur la discrétisation de l'espace d'état pour l'abstraction (voir [174, 97]) et l'approximation en utilisant spécialement des techniques de linéarisation (voir [5, 52]).

Pour la classe particulière des systèmes non linéaires que l'on étudie (les systèmes polynomiaux), une approche similaire utilisant des méthodes directes pour l'analyse d'atteignabilité (sans la discrétisation de l'espace d'état) à été présentée par Dang [49, 53]. En effet, dans ces travaux, l'idée de calculer des relaxations linéaire pour trouver des sur-approximations est utilisée. Cependant, les relaxations proposées consistent à trouver grâce à la forme de Bernstein une fonction affine permettant de minorer notre fonction polynomiale tandis que dans notre approche la fonction minorante est une fonction affine par morceaux beaucoup plus précise. De plus, notre approche nous permet de travailler avec des domaines polyédraux arbitraires au lieu des domaines rectangulaires.

Concernant la représentation des ensembles, le travail présenté dans ce chapitre s'inspire de l'approche utilisant les modèles polyédraux [158]. Dans le cadre de la vérification des systèmes hybrides, l'optimisation polynomiale peut être aussi utilisée pour trouver les fonctions barrières [139, 158], et les propriétés algébriques des polynômes seront utilisées pour le calcul des invariants polynomiaux [174] et pour étudier les problèmes liés au calcul de l'ensemble image [134].

6.3 Experimentation numérique

Notre approche d'atteignabilité a été implémenté [2] en C++ et testée sur des exemples variés. Pour résoudre les programmes linéaires, nous utilisons la bibliothèque lp_solve.

6.3.1 Modèle neuronal de FitzHugh-Nagumo

Le premier exemple que l'on va étudier est une version discrétisée du modèle de FitzHugh-Nagumo [66] qui consiste en un système dynamique polynomial modélisant l'activité électrique des neurones :

$$\begin{cases} x_1(k+1) &= x_1(k) + h\left((x_1(k) - \frac{x_1(k)^3}{3} - x_2(k) + I\right) \\ x_2(k+1) &= x_2(k) + h\left(0.08(x_1(k) + 0.7 - 0.8x_2(k))\right) \end{cases}$$

où le paramètre modèle I est égal à $\frac{7}{8}$ et le pas de temps $h = 0.05$. Il est déjà connu que ce système admet un cycle limite. En testant notre approche sur ce système et en utilisant des boîtes alignés avec les axes pour les polytopes P_k et les parallélotopes

2. L'implémentation est faite par Romain Testylier

Q_k (plus précisément on prend $P_k = Q_k$), on peut voir comme le montre la Figure 6.3 un phénomène dit "effet enveloppant" ou "Wrapping effect". En effet, ce phénomène se produit quand les erreurs de sur-approximations s'accumulent et par conséquent l'ensemble atteignable explose au bout d'un certain temps $t = K$. Ainsi, on ne parvient pas dans le cas de cette configuration à voir le cycle limite. Ceci est dû au fait que la dynamique du système n'est pas prise en considération puisqu'on utilise que des boîtes statiques.

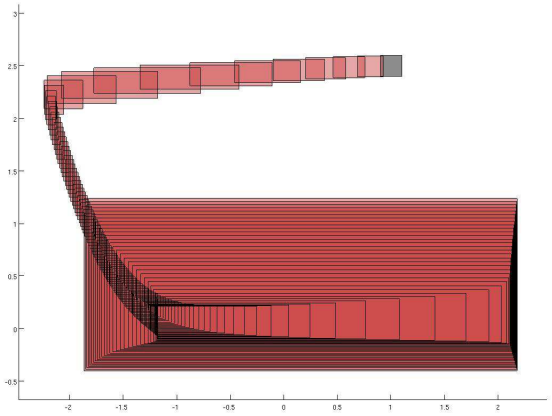

FIGURE 6.3 – L'effet enveloppant pour le modèle neuronal de FitzHugh-Nagumo en utilisant des boîtes alignées avec les axes : $P_k = Q_k$.

Pour résoudre ce problème, il faudra tenir compte de la dynamique du système. Pour cela, on va opter pour le choix des modèles dynamiques pour les polytopes en un premier temps puis pour les boîtes.
La Figure 6.4 montre deux évolutions de l'ensemble atteignable où l'ensemble initial est donné par un octogone régulier et une boîte enveloppante $[0.9, 1.1] \times [2.4, 2.6]$.

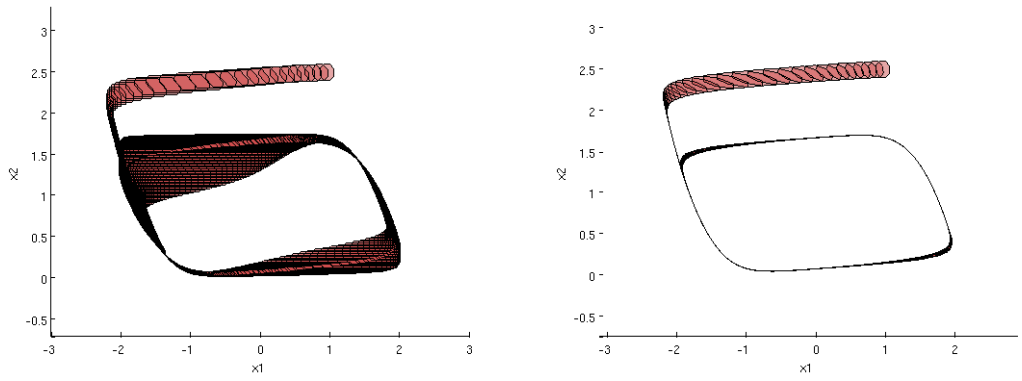

FIGURE 6.4 – Calcul de l'ensemble atteignable pour le modèle neuronal en utilisant un modèle statique (gauche) et dynamique (droite) pour les polytopes $\overline{X}_k$ et des boîtes statiques.

La figure à gauche est calculée en utilisant un modèle statique pour les polytopes

$\overline{X}_k$ tandis qu'un modèle dynamique est utilisé pour la figure à droite. Dans les deux cas on a utilisé des boîtes alignées avec les axes pour Q_k.

Pour une meilleure lisibilité, les ensembles atteignables sont tracés une fois tous les 5 pas de temps. Nous avons observé un cycle limite après 1000 itérations. Le temps de calcul est de 1.16 secondes en utilisant un modèle statique et 1.22 secondes en utilisant un modèle dynamique. Nous pouvons voir sur la figure une amélioration significative de précision obtenue en utilisant le modèles dynamique, à peu de frais supplémentaire (légère augmentation du temps de calcul).

En outre, la Figure 6.5 montre deux évolutions supplémentaires de l'ensemble atteignable en utilisant maintenant des modèles dynamiques pour les polytopes $\overline{X}_k$ avec des boîtes statiques Q_k (en clair) et des boîtes dynamiques (en foncée). Le temps de calcul lors de l'utilisation des boîtes dynamiques passe à $8,24$ secondes. Nous pouvons voir aussi que nous obtenons plus de précision en utilisant les modèles dynamiques pour les boîtes, bien que l'amélioration est moins spectaculaire que celle observée sur la Figure 6.4.

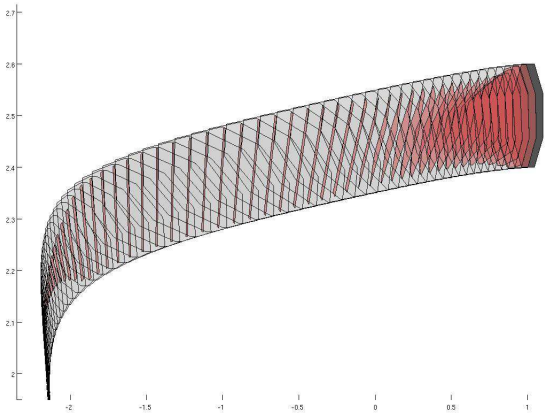

FIGURE 6.5 – Calcul de l'ensemble atteignable pour le modèle neuronal en utilisant à la fois un modèle statique et dynamique pour les boîtes Q_k.

6.3.2 Modèle de croissance du Phytoplancton

Le deuxième exemple est un modèle discrétisé en temps modélisant la croissance du Phytoplankton [19] décrit par le système suivant :

$$\begin{cases} x_1(k+1) &= x_1(k) + h\left(1 - x_1(k) - \frac{x_1(k)x_2(k)}{4}\right) \\ x_2(k+1) &= x_2(k) + h\left((2x_3(k) - 1)x_2(k)\right) \\ x_3(k+1) &= x_3(k) + h\left(\frac{x_1(k)}{4} - 2x_3(k)^2\right) \end{cases}$$

où h désigne le pas de temps (ici on fixe $h = 0.01$). Il est connu que ce système converge vers un point d'équilibre pour des conditions initiales proches de l'origine.

L'ensemble initial que l'on va considérer est le rectangle $[-0.3, -0.2] \times [-0.3, -0.2] \times [-0.05, 0.05]$. La Figure 6.6 montre deux ensemble atteignables obtenus après 500 itérations : l'ensemble clair est calculé en utilisant un modèle statique et l'ensemble foncé est calculé en utilisant les modèles dynamiques pour les polytopes P_k. Dans les deux cas, les Q_k sont des boîtes alignées avec les axes. Le temps de calcul est de 5.36 secondes en utilisant les modèles statiques et 6.12 secondes en utilisant les modèles dynamiques. Nous observons comme dans l'exemple précédent que l'utilisation des modèles dynamiques permet d'obtenir un résultat plus précis encore, à peu de frais.

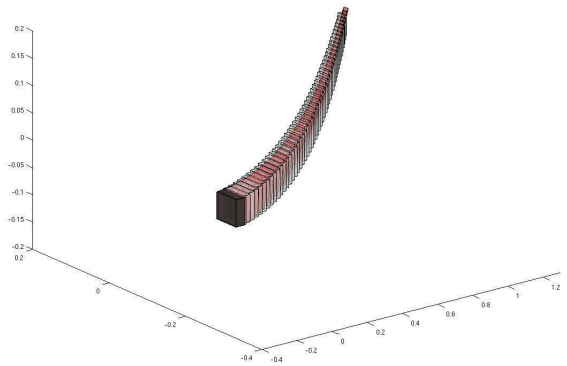

FIGURE 6.6 – Calcul de l'ensemble atteignable pour le modèle de croissance du Phytoplancton avec des modèles statiques et dynamiques.

6.3.3 Modèle proie-prédateur de Lotka-Volterra

Maintenant nous considérons les équations généralisées de Lotka-Volterra modélisant la dynamique de la population de n espèces biologiques connues sous le nom de modèle proie-prédateur. Ses équations sont données par :

$$\dot{x_i} = x_i(r_i + A_i x)$$

où $i \in \{1, 2, \ldots, n\}$, r_i désigne le i-ème élément d'un vecteur $r \in \mathbb{R}^n$ et A_i la i-ème ligne d'une matrice $A \in \mathbb{R}^{n \times n}$.
Nous avons effectué le calcul de l'ensemble atteignable pour le système discrétisé de Lotka-Volterra en deux dimension suivant :

$$\begin{cases} x_1(k+1) &= x_1(k) + h(0.1x_1 - 0.01x_1x_2), \\ x_2(k+1) &= x_2(k) + h(-0.05x_2 + 0.001x_1x_2). \end{cases}$$

La Figure 6.7 montre le comportement cyclique du système à travers l'ensemble atteignable calculé en utilisant un pas de temps $h = 0.3$ pour la discrétisation avec une boîte initiale de dimensions $[49, 51] \times [14, 16]$ durant 700 itérations. La figure de gauche est calculée en 1.87 secondes en utilisant un modèle dynamique pour les polytopes et des boîtes alignées avec les axes. L'autre est calculée en 3.46 secondes

en utilisant un modèle dynamique pour les polytopes et des boîtes orientées. Un gain significatif de précision en utilisant les boîtes orientées peut être observé, toutefois le temps de calcul est presque doublé. Nous avons également évalué la performance

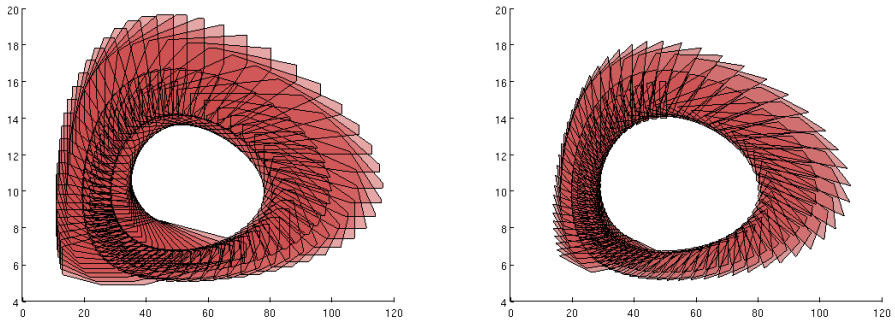

FIGURE 6.7 – Calcul de l'ensemble atteignable pour le modèle proie-prédateur en 2 dimensions en utilisant un modèle dynamique pour le polytope $\overline{X}_k$ avec boîtes alignées avec les axes (statiques) à gauche et des boîtes orientées (dynamiques) à droite.

de notre méthode en utilisant les deux méthodes introduites précédemment pour calculer les coefficients de Bernstein (explicitement et par interpolation) avec des équations de Lotka-Volterra en dimension n récursivement générées données par :

$$\begin{cases} x_1(k+1) &= x_1(k) + h\left(x_1(k)(1 - x_2(k) + x_n(k))\right), \\ x_2(k+1) &= x_i(k) + h\left(x_i(k)(-1 - x_{i+1}(k) + x_{i-1}(k))\right), \\ \quad \ldots & \ldots \quad \ldots \\ x_n(k+1) &= x_n(k) + h\left(x_n(k)(-1 - x_0(k) + x_{n-1}(k))\right). \end{cases}$$

Pour le changement de variable on utilise les boîtes alignées avec les axes (voir Tables 6.1).

dim	explicit	interpol	$\mathsf{B}_{\Delta'}^{-1}$	dim	explicit	interpol	$\mathsf{B}_{\Delta'}^{-1}$
2	0.0235	0.0221	0.0001	7	1.905	1.274	0.099
3	0.0536	0.0484	0.0004	8	5.682	3.674	0.494
4	0.1112	0.1008	0.0008	9	19.35	12.65	2.66
5	0.2612	0.2124	0.0052	10	63.92	44.41	16.23
6	0.68	0.499	0.016				

TABLE 6.1 – Temps de calcul d'un ensemble atteignable associé à quelques équations générées de Lotka-Volterra

Nous observons que la méthode d'interpolation fournit des résultats plus efficaces que ceux fournis par le calcul explicite des coefficient de Bernstein, mais nécessite de calculer la matrice B_{δ}^{-1} avant de commencer l'analyse d'atteignabilité. Une évaluation similaire a été effectuée à l'aide des boîtes orientées, mais les résultats montrent que cette méthode n'est pas utilisable pour une dimension supérieure à 4. La raison est bien évidemment l'élévation du degré des polynômes due au changement de variable lorsqu'on utilise des boîtes orientées.

Dans ce chapitre, on a proposé une approche pour le calcul des ensembles atteignables des systèmes dynamiques polynomiaux. Cette approche combine l'optimisation polynomiale et la représentation des ensembles en utilisant les modèles polyédraux. La relaxation linéaire fournie à l'aide des coefficients de Bernstein est appliquée pour obtenir des bornes fines à nos POP et ainsi des sur-approximations des ensembles atteignables sont établies.
D'autre part, en exploitant l'évolution du système, nous avons proposé un moyen de déterminer dynamiquement les modèles de sorte que les ensembles atteignables puissent être estimés avec plus de précision. L'approche a été testée sur plusieurs exemples et les résultats numériques semblent prometteurs.

Pour les extensions de ce travail on peut citer un certain nombre de directions. Une première direction sera la généralisation de l'approche en temps continu. Une autre direction sera l'étude des systèmes dynamiques polynomiaux incluant des paramètres et des perturbations. En outre, l'évolution des modèles peut être estimée localement autour de chaque face plutôt que globalement au centre de gravité du polytope.

Chapitre 7

Vérification et calcul d'invariants polyédraux pour des systèmes dynamiques polynomiaux

Les ensembles invariants jouent un rôle important dans la théorie du contrôle, notamment pour l'analyse de performance, la robustesse ou la stabilité des systèmes (voir [30] pour plus de détails). Ils sont également d'un grand intérêt dans le domaine de l'analyse formelle des systèmes dynamiques, en particulier pour la vérification des propriétés de sûreté. Dans de tels problèmes, le but est de prouver qu'à partir d'un ensemble donné d'états initiaux, les trajectoires d'un système dynamique ne parviendront jamais à atteindre un ensemble d'états déterminé dit non sécurisé. Des approches dites directes calculent des sur-approximations de l'ensemble des états atteignables : c'est le cadre de l'analyse d'atteignabilité (voir [50, 51, 129] pour de récents progrès dans le cadre des systèmes dynamiques non linéaires ainsi que le chapitre précédent [16]). Cependant, malgré les progrès réalisés du point de vue de la complexité des calculs, ces approches peuvent seulement certifier que l'ensemble des états dangereux (non sécurisé) ne sera pas atteint sur un intervalle de temps borné. Les approches indirectes consistent en la recherche d'un ensemble invariant, contenant l'ensemble des états initiaux, et dont l'intersection avec l'ensemble des états dangereux est vide. Pour les systèmes dynamiques polynomiaux, ces invariants sont généralement donnés par des ensembles semi-algébriques [139, 157]. Pour des sous-classes spécifiques tels que les systèmes dynamiques multi-affines ou quasi multi-affines, ces ensembles sont donnés par des rectangles [13, 1].
Dans ce chapitre on va traiter le problème de vérification et de calcul d'ensembles invariants polyédraux pour les systèmes dynamiques polynomiaux. Remarquons à ce stade que les rectangles forment une sous classe des invariants polyédraux, ainsi on peut considérer ce travail comme étant une extension de [13, 1]. D'autre part, les polytopes forment une sous-classe des ensembles algébriques et il y a donc une possibilité de concevoir des procédures algorithmiques spécifiques qui peuvent être plus efficaces que celles présentées dans [139, 157].
On verra que les invariants polyédraux peuvent être vérifiés par la résolution d'un ensemble de problèmes d'optimisation de polynômes multivariés sur des polytopes bornés (ensembles de POP). On utilisera la relaxation qu'on a établie à l'aide du

principe de floraison pour calculer des bornes inférieures aux valeurs optimales de ces POP. Cela nous permet de proposer une méthode basée sur la programmation linéaire nous permettant de vérifier si un polytope est invariant pour un système dynamique polynomial donné. En outre, on va montrer, en utilisant l'analyse de sensibilité des programmes linéaires, qu'un ensemble invariant polyédral peut aussi être calculé par une méthode itérative. Enfin, on montre à l'aide d'un ensemble d'exemples empruntés des applications en ingénierie et en biologie que notre approche est utile dans la pratique.

7.1 Invariance et analyse de sensibilité

7.1.1 Problèmes d'invariance et formulation en POP

On va considérer le système dynamique suivant :

$$\dot{x}(t) = f(x(t)),\ x(t) \in \mathbb{R}^n \tag{7.1}$$

où $f : \mathbb{R}^n \to \mathbb{R}^n$ est un champ de vecteur polynomial.
On commence tout d'abord par rappeler la définition d'ensemble invariant associé au système (7.1).

Définition 7. *Un ensemble I est dit invariant pour le système dynamique (7.1) s'il vérifie la propriété suivante :*
Si l'état du système appartient à l'ensemble I à un instant donné, il restera toujours dans I. Autrement dit, s'il existe un temps t_0 tel que $x(t_0) \in I$ alors $x(t) \in I$ pour tout $t \geq t_0$.

Dans ce chapitre on se propose d'étudier deux problème liés à l'invariance : le premier est celui de la vérification d'invariants et le deuxième est celui du calcul d'invariants.

Problème 1. Etant donné un polytope P, on veut vérifier si P est un invariant pour le système (7.1). C'est le problème de vérification.

Problème 2. Etant donné un polytope P (qui à priori n'est pas un invariant), on veut calculer à l'aide de P un polytope P', dont les directions des faces sont les mêmes que P, qui sera invariant pour le système (7.1).

On verra que la résolution de ces problèmes se base essentiellement sur la résolution des POP. En effet, étant donné un polytope P avec un ensemble de faces $\{F_k |\ k \in K\}$, il en découle d'après la caractérisation standard des ensembles invariant (voir par exemple [7]) que P est invariant pour le système (7.1) si et seulement si

$$\forall k \in K,\ \forall x \in F_k,\ a_k \cdot f(x) \leq 0 \tag{7.2}$$

où a_k est le vecteur normal à F_k qui pointe à l'extérieur de P. Comme souligné dans [1], et par l'application du Théorème de Tarski [172], c'est un problème décidable. Cependant, la complexité de la procédure de décision donne peu d'espoir

pour l'application pratique. Notons que (7.2) peut être reformulé comme suit :

$$\forall k \in K,\ \min_{x \in F_k} -a_k \cdot f(x) \geq 0. \tag{7.3}$$

Ceci consiste à montrer que les valeurs minimales des polynômes multi variés $-a_k \cdot f$ dans les polytopes bornées F_k sont positifs. Donc, si nous sommes en mesure de calculer des bornes inférieures non négatives à ces valeurs minimales, on pourra prouver que le polytope P est invariant pour le système dynamique (7.1). Par conséquent, il nous suffit d'appliquer une des relaxations linéaires et les bornes inférieures seront fournis par l'intermédiaire des programmes linéaires de ces relaxations. C'est l'approche que l'on va suivre dans le présent chapitre en optant pour la deuxième relaxation : celle se basant sur le principe de floraison.

7.1.2 Relaxation linéaire

Comme on l'a déjà indiqué, la vérification des invariants polyédraux pour des systèmes dynamiques polynomiaux peut être traitée par la résolution d'un ensemble de problèmes d'optimisation de polynômes multi-variés sur des polytopes bornés. Par conséquent, dans cette section, nous considérons le POP suivant :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in R, \\ & a_i \cdot x \leq b_i, \quad i \in I, \\ & c_j \cdot x = d_j, \quad j \in J, \end{array} \tag{7.4}$$

où $p : \mathbb{R}^n \to \mathbb{R}$ est un polynôme multi-varié de degré δ, $R = [\underline{x_1}, \overline{x_1}] \times \cdots \times [\underline{x_n}, \overline{x_n}]$ est un rectangle de $\mathbb{R}^n$ (le rectangle R peut être pris comme étant la boîte enveloppante du polytope P) ; $I = \{1, \ldots, m_I\}$ et $J = \{1, \ldots, m_J\}$ des ensemble d'indices ; $a_i \in \mathbb{R}^n$, $b_i \in \mathbb{R}$, pour tout $i \in I$ et $c_j \in \mathbb{R}^n$, $d_j \in \mathbb{R}$, pour tout $j \in J$.
On rappelle que notre relaxation sera construite comme suit :
Soit $q_\delta : \mathbb{R}^{\delta_1 + \cdots + \delta_n} \to \mathbb{R}$ la forme polaire associée à p relative à δ. Ainsi, en utilisant les notations de la Section 5.1, la relaxation linéaire associée au problème (7.4) est donnée par :

$$\begin{array}{lll} \text{maximiser} & t & \\ \text{s.c} & t \in \mathbb{R},\ \lambda \in \mathbb{R}^{m_I},\ \mu \in \mathbb{R}^{m_J}, & \\ & \lambda_i \geq 0, & i \in I, \\ & t \leq q_\delta(\overline{v}) + \sum_{i \in I} \lambda_i (a_i' \cdot \overline{v} - b_i) + \sum_{j \in J} \mu_j (c_j' \cdot \overline{v} - d_j), & \overline{v} \in \overline{V'}. \end{array} \tag{7.5}$$

La raison principale expliquant le choix de la relaxation donnée par le programme linéaire (7.5) est l'analyse de sensibilité.

7.1.3 Analyse de sensibilité

Une caractéristique intéressante de la dualité lagrangienne est qu'elle permet une analyse de sensibilité (voir par exemple [32]). Dans cette section, nous nous intéressons à l'analyse des variations de la valeur optimale de (7.4) et spécialement de

sa borne inférieure donnée par la valeur optimale de (7.5) en vertu des modifications du polytope P. Le résultat obtenu sera utilisé dans la section suivante pour le calcul des invariants polyédraux associés à des systèmes dynamiques polynomiaux. Plus précisément, nous considérons la variation suivante du problème (7.4) :

$$\begin{array}{ll} \text{minimiser} & p(x) \\ \text{s.c} & x \in R, \\ & a_i \cdot x \leq b_i + \alpha_i, \quad i \in I, \\ & c_j \cdot x = d_j + \beta_j, \quad j \in J, \end{array} \tag{7.6}$$

où $\alpha_i \in \mathbb{R}$, pour tout $i \in I$ et $\beta_j \in \mathbb{R}$ pour tout $j \in J$.
Ce problème coïncide avec le problème original (7.4) quand $\alpha = 0$ et $\beta = 0$. Soient p^* et $p^*(\alpha, \beta)$ les valeurs optimales des problèmes (7.4) et (7.6), respectivement. Soient ${p_\delta}^*$ et ${p_\delta}^*(\alpha, \beta)$ les bornes inférieures de p^* et $p^*(\alpha, \beta)$ obtenues par application de la relaxation (7.5).

Théorème 11. *Soient ${p_\delta}^*$ et (t^*, λ^*, μ^*) la valeur ainsi que la solution optimale du programme linéaire (7.5). Alors, pour tous $\alpha \in \mathbb{R}^{m_I}$ et $\beta \in \mathbb{R}^{m_J}$, tels que (7.6) est admissible, on a :*

$$p^*(\alpha, \beta) \geq {p_\delta}^*(\alpha, \beta) \geq {p_\delta}^* - \lambda^* \cdot \alpha - \mu^* \cdot \beta.$$

Démonstration. En appliquant la relaxation (7.5) au problème (7.6), il est évident que ${p_\delta}^*(\alpha, \beta)$ est la valeur optimale de :

$$\begin{array}{lll} \text{maximiser} & t & \\ \text{s.c} & t \in \mathbb{R},\ \lambda \in \mathbb{R}^{m_I},\ \mu \in \mathbb{R}^{m_J}, & \\ & \lambda_i \geq 0, & i \in I, \\ & t \leq q_\delta(\overline{v}) + \sum_{i \in I} \lambda_i (a_i' \cdot \overline{v} - b_i - \alpha_i) + \sum_{j \in J} \mu_j (c_j' \cdot \overline{v} - d_j - \beta_j), & \overline{v} \in \overline{V'}. \end{array} \tag{7.7}$$

Le fait que $p^*(\alpha, \beta) \geq {p_\delta}^*(\alpha, \beta)$ est trivial.
Maintenant, montrons que $(t^* - \lambda^* \cdot \alpha - \mu^* \cdot \beta, \lambda^*, \mu^*)$ est admissible pour (7.7).
Il est évident que $\lambda_i^* \geq 0$, pour tout $i \in I$. De plus, pour tout $\overline{v} \in \overline{V'}$,

$$\begin{aligned} & q_\delta(\overline{v}) + \sum_{i \in I} \lambda_i^* (a_i' \cdot \overline{v} - b_i - \alpha_i) + \sum_{j \in J} \mu_j^* (c_j' \cdot \overline{v} - d_j - \beta_j) \\ = \ & q_\delta(\overline{v}) + \sum_{i \in I} \lambda_i^* (a_i' \cdot \overline{v} - b_i) + \sum_{j \in J} \mu_j^* (c_j' \cdot \overline{v} - d_j) - \lambda^* \cdot \alpha - \mu^* \cdot \beta \\ \geq \ & t^* - \lambda^* \cdot \alpha - \mu^* \cdot \beta. \end{aligned}$$

Ainsi, $(t^* - \lambda^* \cdot \alpha - \mu^* \cdot \beta, \lambda^*, \mu^*)$ est admissible pour (7.7). Par conséquent, on aura ${p_\delta}^*(\alpha, \beta) \geq t^* - \lambda^* \cdot \alpha - \mu^* \cdot \beta$ ce qui conduit à l'inégalité prévu puisque $d^* = t^*$. □

7.2 Vérification et calcul d'invariant pour les systèmes dynamiques polynomiaux

Dans ce qui suit, nous montrons comment les résultats développés dans la section précédente peuvent être utilisés pour la vérification et le calcul des invariants

polyédraux pour les systèmes dynamiques polynomiaux. On commence tout d'abord par rappeler notre système dynamique :

$$\dot{x}(t) = f(x(t)),\ x(t) \in \mathbb{R}^n$$

où $f : \mathbb{R}^n \to \mathbb{R}^n$ est un champ de vecteurs polynomial.
Soit $R = [\underline{x_1}, \overline{x_1}] \times \cdots \times [\underline{x_n}, \overline{x_n}]$ un rectangle of $\mathbb{R}^n$ délimitant une région d'intérêt pour l'étude de la dynamique (7.1). Dans cette section, on va détailler une approche permettant de résoudre le **Problème 1** et le **Problème 2**. En effet, on montre tout d'abord comment vérifier qu'un polytope donné $P \subseteq R$ est un invariant pour le système (7.1). Ensuite, on montre que si notre approche échoue, c'est à dire qu'elle ne permet pas d'affirmer que P est invariant, l'analyse de sensibilité peut nous aider à modifier P de façon à trouver un polytope invariant pour (7.1) ayant les mêmes directions que P.

7.2.1 Vérification d'invariants polyédraux

On considère un polytope borné $P \subseteq \mathbb{R}^n$ défini comme suit :

$$P = \{x \in \mathbb{R}^n |\ a_k \cdot x \leq b_k,\ \forall k \in K\}$$

où $K = \{1, \ldots, m_K\}$ est un ensemble d'indices, $a_k \in \mathbb{R}^n$, $b_k \in \mathbb{R}$, pour tout $k \in K$. Les faces du polytope sont $\{F_k |\ k \in K\}$ où

$$F_k = \{x \in \mathbb{R}^n |\ a_k \cdot x = b_k,\ \text{et } a_i \cdot x \leq b_i,\ \forall i \in K \setminus \{k\}\}.$$

On suppose que $P \subseteq R$ et que toutes les faces F_k ne sont pas vides. Comme indiqué au début du chapitre, P est un invariant pour le système dynamique (7.1) si et seulement si

$$\forall k \in K,\ \min_{x \in F_k} -a_k \cdot f(x) \geq 0.$$

Puisque pour tout $k \in K$, $F_k \subseteq P \subseteq R$, alors ce problème est équivalent à montrer que les valeurs optimales p_k^* des POP suivantes sont positives pour tout $k \in K$:

$$\begin{array}{lll} \text{minimiser} & -a_k \cdot f(x) & \\ \text{s.c} & x \in R, & \\ & a_i \cdot x \leq b_i, & i \in K \setminus \{k\}, \\ & a_k \cdot x = b_k. & \end{array} \tag{7.8}$$

Comme $-a_k \cdot f$ est un polynôme multi-varié, ce problème est similaire à (7.4). Par conséquent, en appliquant la relaxation (7.5) on obtient le résultat suivant :

Proposition 18. *Pour $k \in K$, soit $p_{k,\delta}^*$ la valeur optimale du programme linéaire :*

$$\begin{array}{lll} \textit{maximiser} & t & \\ \textit{s.c} & t \in \mathbb{R},\ \lambda \in \mathbb{R}^{m_K}, & \\ & \lambda_i \geq 0, & i \in K \setminus \{k\}, \\ & t \leq q_{k,\delta}(\overline{v}) + \sum_{i \in K} \lambda_i (a_i' \cdot \overline{v} - b_i) & \overline{v} \in \overline{V'}, \end{array} \tag{7.9}$$

où $q_{k,\delta}$ est la forme polaire du polynôme multi-varié $-a_k \cdot f$ par respect d'un degré $\delta = \delta(k)$ (souvent pris comme étant le degré du polynôme $-a_k \cdot f$).
Si pour tout $k \in K$, $p^*_{k,\delta} \geq 0$, alors P est un invariant polyédral pour le système dynamique (7.1).

Démonstration. Il est évident que $p^*_k \geq p^*_{k,\delta}$, pour tout $k \in K$. Ainsi, $p^*_{k,\delta} \geq 0$ implique que $p^*_k \geq 0$, pour tout $k \in K$ et par conséquent le polytope P est invariant. □

Ainsi, on a montré que l'invariance d'un polytope P peut être vérifiée par la résolution d'un ensemble de programmes linéaires (un par face). Dans le cas où on n'arrive pas à vérifier que le polytope P est invariant, l'analyse de sensibilité peut nous aider à le modifier dans le but de trouver un polytope invariant pour (7.1).

Remarque 15. *Notons que les degrés des polynômes multi-variés $-a_k \cdot f$ peuvent ne pas être pas tous les mêmes. Ceci affectera bien évidemment les vecteurs a'_i, et les ensembles $\overline{V'}$ qui vont dépendre de l'indice $k \in K$. Il est possible d'éviter ce problème en définissant $q_{k,\delta} = q_{k,\delta_{max}}$ comme étant $-a_k \cdot g_{\delta_{max}}$ où $g_{\delta_{max}}$ est la forme polaire du champ de vecteurs polynomial f relative au degré δ_{max} dont les composantes sont définie comme suit : pour $i, j \in \{1, \ldots, n\}$, notons δ_{ij} le degré de x_i dans le polynôme à plusieurs variables $f_j(x)$ et soit $\delta_{i,max} = \max_{j \in \{1,\ldots,n\}} \delta_{ij}$. Pour tout $j \in \{1, \ldots, n\}$, il est possible de voir f_j comme étant un polynôme à plusieurs variables de degrés $\delta_{1,max}, \ldots, \delta_{n,max}$ avec éventuellement des coefficients nuls et de définir sa forme polaire associée $g_{j,\delta_{max}}$. Ensuite, on prendra $g_{j,\delta_{max}}$ comme étant les composantes de la forme polaire $g_{\delta_{max}} : \mathbb{R}^{\delta_{1,max}+\cdots+\delta_{n,max}} \to \mathbb{R}^n$ du champ de vecteurs polynomial f pour tout $j \in \{1, \ldots, n\}$. Alors, pour tout $k \in K$, $q_k = -a_k \cdot g_{\delta_{max}}$ sont des fonctions multi-affines définies sur $\mathbb{R}^{\delta_{1,max}+\cdots+\delta_{n,max}}$ avec des propriétés semblables à la forme polaire de $-a_k \cdot f$ qui peut être utilisée dans le problème (7.9).*

7.2.2 Calcul d'invariants polyédraux

Maintenant si notre approche échoue, c'est à dire qu'elle ne permet pas d'affirmer que P est un invariant pour le système dynamique (7.1) : c'est le cas si $p^*_{k,\delta} < 0$, pour un certain $k \in K$.
Dans ce cas, nous pouvons être intéressés par la modification de P dans le but de trouver un polytope invariant. Pour cela, on va considérer les polytopes dont les face sont parallèles à celles de P, c'est à dire ayant le même modèle (même matrice de direction). Pour $\alpha \in R^{m_K}$, soit P_α le polytope donné par

$$P_\alpha = \{x \in \mathbb{R}^n |\ a_k \cdot x \leq b_k + \alpha_k,\ \forall k \in K\}.$$

Pour $\alpha = 0$, on retrouve le polytope P. Notre but est de trouver un α assurant que le polytope P_α sera un invariant pour (7.1). On impose des contraintes supplémentaires pour le polytope P_α :

- $P_\alpha \subseteq R$: ceci peut être assuré par des contraintes de la forme $b_k + \alpha_k \leq \overline{b_k}$.
- P_α est non vide : ceci peut être assuré par des contraintes de la forme $\underline{b_k} \leq b_k + \alpha_k$.
- P_α est relativement proche de P : ceci est assuré en imposant $-\varepsilon \leq \alpha_k \leq \varepsilon$ où ε est un paramètre qui peut être réglé.

On note pour $k \in K$, $p^*_{k,\delta}(\alpha)$ la valeur optimale des problèmes (7.9) pour le polytope P_α ; d'après le Théorème 11 se basant sur l'analyse de sensibilité on a

$$\forall k \in K,\ p^*_{k,\delta}(\alpha) \geq p^*_{k,\delta} + \lambda^*_k \cdot \alpha$$

où $p^*_{k,\delta}$ et (t^*_k, λ^*_k) sont les valeurs et les solutions optimales des problèmes (7.9) pour le polytope P et $k \in K$. Ainsi, d'après la Proposition 18, pour que P_α soit un invariant polyédral pour le système dynamique (7.1), il suffit que pour tout $k \in K$, $p^*_{k,\delta} + \lambda^*_k \cdot \alpha \geq 0$. Afin de trouver un bon candidat pour α, on doit résoudre le problème suivant :

$$\begin{array}{ll} \text{maximiser} & \min\limits_{k\in K} \left(p^*_{k,\delta} + \lambda^*_k \cdot \alpha\right) \\ \text{s.c} & \alpha \in \mathbb{R}^{m_K}, \\ & \underline{\alpha_k} \leq \alpha_k \leq \overline{\alpha_k}, \qquad k \in K \end{array}$$

où $\underline{\alpha_k} = \max(-\varepsilon, \underline{b_k} - b_k)$ et $\overline{\alpha_k} = \min(\varepsilon, \overline{b_k} - b_k)$. Ce problème peut être reformulé à l'aide du programme linéaire suivant :

$$\begin{array}{ll} \text{maximiser} & t \\ \text{s.c} & t \in \mathbb{R},\ \alpha \in \mathbb{R}^{m_K}, \\ & t \leq p^*_{k,\delta} - {\lambda_k}^* \cdot \alpha, \quad k \in K, \\ & \underline{\alpha_k} \leq \alpha_k \leq \overline{\alpha_k}, \qquad k \in K. \end{array} \tag{7.10}$$

Notons (t^*, α^*) la solution du programme linéaire. Si la valeur optimale t^* est positive alors on a montré que P_{α^*} est un invariant pour le système dynamique (7.1). Si la valeur optimale est strictement négative, alors on calcule $p^*_{k,\delta}(\alpha^*)$ en résolvant les problèmes (7.9) pour le polytope P_{α^*} pour tout $k \in K$. Si tous les $p^*_{k,\delta}(\alpha^*)$ sont positifs, alors d'après la Proposition 18, P_{α^*} est un invariant. Si la vérification échoue, alors on utilise l'analyse de sensibilité pour modifier P_{α^*} et trouver un invariant. Ceci donne une approche itérative pour le calcul des invariants polyédraux pour les systèmes dynamiques polynomiaux.

Remarque 16. *Remarquons que le polytope P_{α^*} calculé par la résolution de (7.10) peut avoir des faces vides. Ceci résulte, pour une face vide F_k, d'une valeur non bornée de $p^*_{k,\delta}(\alpha^*) = +\infty$. Pour éviter ce genre de situations, il peut être utile de remplacer α^* par $\tilde{\alpha}^*$ tel que $P_{\tilde{\alpha}^*}$ n'a pas de faces vides et $P_{\alpha^*} = P_{\tilde{\alpha}^*}$ (voir Figure 7.1). Encore une fois, cela peut être fait en résolvant un ensemble de programmes linéaires.*

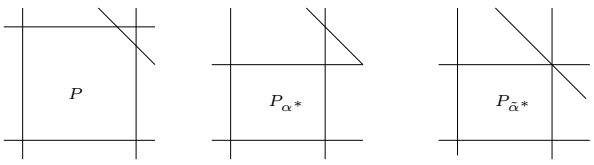

FIGURE 7.1 – Le polytope P_{α^*} peut avoir des faces vides (polytope du milieu), on remplace α^* par $\tilde{\alpha}^*$ tel que $P_{\tilde{\alpha}^*}$ n'a pas de faces vides et $P_{\alpha^*} = P_{\tilde{\alpha}^*}$ (polytope à droite).

7.2.3 Autres approches

Le calcul des invariants pour des systèmes dynamiques polynomiaux est souvent abordée à l'aide de la programmation semi-définie (SDP) par l'intermédiaire des relaxations SOS. Une grande partie de la littérature dans ce sujet consiste à calculer les fonctions de Lyapunov dont les courbes de niveau sont les invariants (voir par exemple [186, 175]). Une approche similaire pour le calcul des ensembles invariants ne contenant pas de point d'équilibre stable peut être trouvée dans [139]. Dans ces travaux, les ensembles invariants sont des ensembles semi-algébriques décrites par des inégalités polynomiales. Toutefois, les polytopes sont plus faciles à manipuler et, comme expliqué dans [2], il est parfois préférable d'avoir des invariants décrits par des polytopes plutôt que par des ensembles semi-algébriques. En particulier, les polytopes dont les directions des faces sont fixées par un modèle se sont avéré très utiles pour le calcul des ensembles invariants pour les systèmes hybrides avec une dynamique affine dans [159] et pour les systèmes multi-affine dans [13, 1]. Dans un certain sens, notre travail s'appuie sur ces approches et les étend à la classe des systèmes dynamiques polynomiaux.

Il convient aussi de mentionner les travaux sur les fonctions de Lyapunov polyédrales pour la classe des systèmes dynamiques décrites par des inclusions différentielles linéaires. Pour cette classe de systèmes, il peut être démontré que l'existence d'un ensemble invariant contenant l'origine est équivalente à la stabilité de Lyapunov [30]. Des méthodes reposant sur la programmation linéaire pour le calcul des invariants polyédraux pour des inclusions différentielles linéaires ont été développées (voir par exemple [136]). Nous tenons à souligner que notre approche peut être facilement adaptée pour le calcul des ensembles invariants pour les inclusions différentielles polynomiales (et donc linéaires). Par conséquent, notre approche peut être considérée comme une généralisation du travail mentionné ci-dessus même si les algorithmes utilisés pour le calcul des invariants polyédraux sont différents.

7.3 Exemples

Nous avons implémenté notre approche dans Matlab. Dans ce qui suit, nous montrons pour une série d'exemples provenant des applications d'ingénierie ou de la biologie que notre approche est efficace dans la pratique. Tous les calculs rapportés prennent quelques secondes.

7.3.1 Modèle de Moore-Greitzer d'un moteur à réaction

Nous avons testé notre approche sur le système dynamique polynomial suivant correspondant à un modèle de Moore-Greitzer d'un moteur à réaction avec un paramètre de stabilisation [98] :

$$\begin{cases} \dot{x}_1 &= -x_2 - \frac{3}{2}x_1^2 - \frac{1}{2}x_1^3, \\ \dot{x}_2 &= 3x_1 - x_2. \end{cases}$$

Ce système admet l'origine comme équilibre stable. Grâce à notre approche, nous avons essayé de trouver un polytope invariant pour ce système autour de l'origine. En

se fixant le rectangle $[-0.2, 0.2]^2$, nous avons pu trouver un polytope invariant avec $m_K = 24$ faces dont les orientations sont uniformément réparties (voir la Figure 7.2). A partir de l'ensemble représenté en pointillé, notre approche a besoin de 5 itérations pour trouver le polytope invariant représenté par un trait continu. On peut vérifier sur la figure que c'est effectivement un invariant. A chaque itération, nous avons

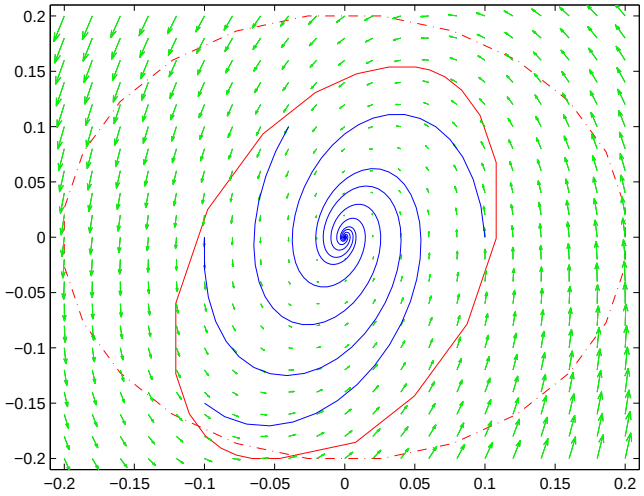

FIGURE 7.2 – Invariant polyédral pour le modèle de Moore-Greitzer (représenté par un trait continu) obtenu après 5 itérations commençant par le polytope en pointillé.

besoin de résoudre pour l'étape de vérification $m_K = 24$ programmes linéaires de la forme (7.9) avec $m_K+1 = 25$ variables et $m_K-1+(\delta_1+1)\times(\delta_2+1) = 31$ contraintes inégalités. Pour le calcul d'un invariant, en utilisant une analyse de sensibilité, nous avons besoin à chaque itération de résoudre un programme linéaire de $m_K + 1 = 25$ variables et $2m_K = 48$ contraintes inégalités.

7.3.2 Modèle neuronal de FitzHugh-Nagumo

Nous avons également appliqué notre approche sur le modèle de FitzHugh-Nagumo [66] (dont une version discrétisée a été étudié dans le chapitre précédent). On rappelle qu'il s'agit d'un système dynamique polynomial modélisant l'activité électrique d'un neurone :

$$\begin{cases} \dot{x}_1 &= x_1 - x_1^3/3 - x_2 + I, \\ \dot{x}_2 &= 0.08(x_1 + 0.7 - 0.8x_2), \end{cases}$$

où le paramètre I est pris égal à $\frac{7}{8}$. Il est connu que ce système admet un cycle limite. En effet, en utilisant notre approche, nous avons synthétisé un polytope invariant contenant le cycle limite. En se fixant le rectangle $[-2.5, 2.5] \times [-1.5, 3.5]$, on trouve un invariant polyédral à 8 faces dont les orientations sont uniformément réparties (voir Figure 7.3). Commençant par l'ensemble représenté par un trait discontinu,

notre approche a besoin de 15 itérations pour trouver l'invariant polyédral tracé à l'aide d'un trait continu. On peut vérifier à l'aide de la figure que c'est effectivement un invariant. Remarquons que cet invariant polyédral P ainsi que l'existence d'un équilibre instable à l'intérieur de P fournit par l'application du théorème de Poincaré Bendixon (voir appendice de la première partie) une preuve formelle de l'existence d'un cycle limite à l'intérieur du polytope P. Pour chaque itération, on a besoin de

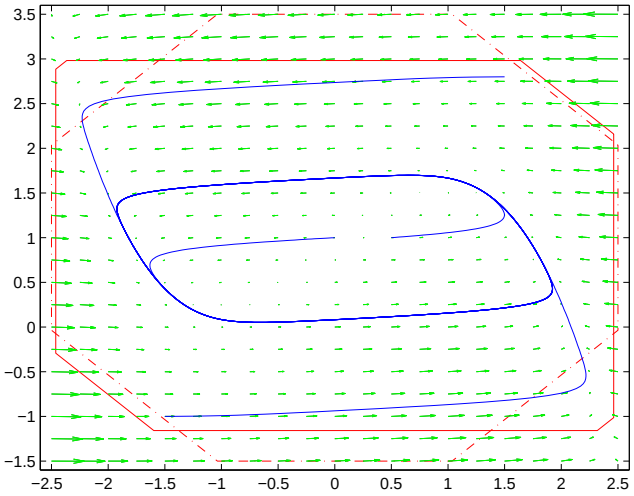

FIGURE 7.3 – Invariant polyédral pour le modèle de FitzHugh-Nagumo (représenté par un trait continu) obtenu après 15 itérations commençant par le polytope en pointillé. L'invariant contient le cycle limite.

résoudre pour la vérification de l'invariant $m_K = 8$ programmes linéaires de la forme (7.9) avec $m_K + 1 = 9$ variables et $m_K - 1 + (\delta_1 + 1) \times (\delta_2 + 1) = 15$ contraintes inégalité. Pour le calcul de l'invariant en utilisant l'analyse de sensibilité, on a besoin de résoudre pour chaque itération un programme linéaire de $m_K + 1 = 9$ variables et $2m_K = 16$ contraintes inégalités.

7.3.3 Modèle de croissance du phytoplancton

Le dernier exemple a aussi été traité dans le chapitre précédent (après une discrétisation d'Euler). C'est le modèle de croissance du phytoplancton [19] :

$$\begin{cases} \dot{x}_1 &= 1 - x_1 - \frac{x_1 x_2}{4}, \\ \dot{x}_2 &= (2x_3 - 1)x_2, \\ \dot{x}_3 &= \frac{x_1}{4} - 2x_3^2. \end{cases}$$

Ce système admet un équilibre stable. En utilisant notre approche, on calcule un invariant polyédral contenant l'équilibre. En fixant le rectangle $[0, 3] \times [-0.1, 2] \times [0, 0.6]$, on sera capable de trouver un invariant polyédral avec $m_K = 18$ faces (un octogone régulier : voir Figure 7.4). Commençant par le polytope représenté dans

la partie gauche de la figure, notre approche a besoin de 11 itérations pour trouver l'invariant polyédral tracé dans la partie droite de la figure. On peut vérifier à l'aide de la figure que c'est bien un invariant.

A chaque itération, on doit résoudre pour le problème de vérification $m_K = 18$ programmes linéaires de la forme (7.9) avec $m_K + 1 = 19$ variables et $m_K - 1 + (\delta_1 + 1) \times (\delta_2 + 1) \times (\delta_3 + 1) = 29$ contraintes d'inégalité. Pour le calcul d'invariant en utilisant l'analyse de sensibilité, on aura besoin de résoudre pour chaque itération un programme linéaire de $m_K + 1 = 19$ variables et $2m_K = 36$ contraintes d'inégalité.

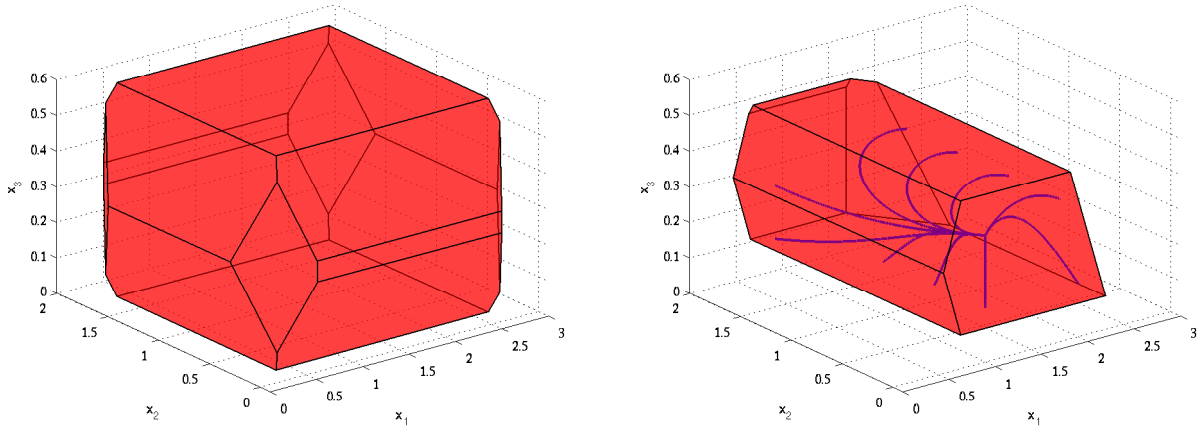

FIGURE 7.4 – Invariant polyédral pour le modèle de croissance phytoplancton (à droite) obtenu après 11 itérations commençant par le polytope de gauche.

Dans ce chapitre, on a présenté une approche basée sur la programmation linéaire pour la vérification et le calcul des invariants polyédraux pour des systèmes dynamiques polynomiaux. La relaxation choisie est celle se basant sur le principe de la floraison ainsi que la dualité Lagrangienne. Ce choix est dû essentiellement à l'application de l'analyse de sensibilité pour le calcul d'un invariant polyédral ayant les mêmes directions que le polytope initial. Bien que notre approche soit conservative (parfois on ne peut pas à vérifier l'invariance d'un polytope), nous avons montré sur plusieurs exemples qu'elle peut être utile dans des applications pratiques.

Dans le chapitre suivant, on va généraliser l'approche présentée dans ce chapitre pour la synthèse d'un contrôleur pour les systèmes dynamiques polynomiaux avec perturbations.

Chapitre 8

Synthèse de contrôleur pour l'invariance robuste d'un système dynamique polynomial

Dans les chapitres précédents, on s'est restreint à l'analyse des systèmes dynamique polynomiaux en étudiant les propriétés telles que l'atteignabilité et l'invariance. Dans ce chapitre la partie contrôle sera étudiée. En effet, la conception des systèmes non linéaires reste un problème difficile dans la théorie de contrôle. Au cours de la dernière décennie, en s'appuyant sur les percées spectaculaires dans l'optimisation des fonctions polynomiales [101, 132], plusieurs méthodes de calcul ont été développées pour la synthèse de contrôleurs dans le cas des systèmes dynamiques polynomiaux [140, 105]. Ces approches ont montré un succès dans plusieurs problèmes de synthèse tels que la stabilisation ou le contrôle optimal dans lequel les fonctions de Lyapunov et les fonctions coût peuvent être représentées ou approchées par des polynômes. Cependant, ces approches ne sont pas convenables pour d'autres problèmes tels que ceux impliquant des systèmes dynamiques polynomiaux avec des contraintes sur les variables d'états ainsi que les entrées et soumis à des perturbations bornées.

Dans ce chapitre, nous considérons le problème de synthèse de contrôleur pour cette classe de systèmes. Plus précisément, étant donné un système dynamique polynomial avec des contraintes sur les entrées ainsi que des perturbations bornées, étant donné un ensemble d'états initiaux $\underline{P}$ et un ensemble d'états sûrs $\overline{P}$, nous visons à synthétiser un contrôleur satisfaisant les contraintes des entrées et garantissant que les trajectoires initialisées dans $\underline{P}$ demeurent dans $\overline{P}$ pour toutes les perturbations possibles. Ce problème peut être résolu en calculant conjointement le contrôleur et un ensemble invariant pour le système contrôlé contenant $\underline{P}$ et inclus dans $\overline{P}$ (voir par exemple [30, 91]). Ici, il convient de mentionner que, même dans le cas linéaire, le problème de la conception conjointe d'un contrôleur et d'un invariant n'est pas du tout trivial [181, 26] et peut conduire à un problème non linéaire.

Dans ce qui suit, nous proposons une méthode de calcul basée sur l'utilisation des expressions paramétrées des modèles pour le contrôleur et l'invariant. Etant donné un candidat polyédral, nous montrons que la synthèse de contrôleurs assurant son invariance peut être formulée comme un problème d'optimisation faisant intervenir

des fonctions objectives polynomiales sur des polytopes bornés. L'amélioration par rapport au travail du chapitre précédent [14] réside dans le fait que dans le présent chapitre, des entrées avec des contraintes ainsi que des perturbations bornées sont considérés ; aussi une approche itérative est donnée pour calculer conjointement un contrôleur et un invariant polyédral. Chaque itération de l'approche consiste principalement à résoudre deux programmes linéaires (un pour le contrôleur et l'autre pour l'invariant). Enfin, nous montrons l'utilité de notre approche dans la pratique en l'appliquant à plusieurs exemples.

8.1 Formulation du problème

Dans ce chapitre, on va considérer un système de contrôle non linéaire affine soumis à des contraintes d'entrées et des perturbations bornées :

$$\dot{x}(t) = f(x(t), d(t)) + g(x(t), d(t))u(t),\ d(t) \in D\ , u(t) \in U \tag{8.1}$$

où $x(t) \in R_X \subseteq \mathbb{R}^n$ est l'état du système, $d(t) \in D \subseteq \mathbb{R}^m$ est une perturbation extérieure et $u(t) \in U \subseteq \mathbb{R}^p$ désigne le contrôleur. On suppose que le champ de vecteurs $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ et la matrice de contrôle $g : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{(n\times p)}$, définissant la dynamique du système, sont des polynôme multi-variés.

On suppose également que l'ensemble des états est un domaine borné rectangulaire : $R_X = [\underline{x_1}, \overline{x_1}] \times \cdots \times [\underline{x_n}, \overline{x_n}]$ et que l'ensemble des perturbations D et l'ensemble des entrées U sont des polytopes convexes et compacts :

$$D = \{d \in \mathbb{R}^m |\ \alpha_{D,k} \cdot d \leq \beta_{D,k},\ \forall k \in \mathcal{K}_D\}$$

$$U = \{u \in \mathbb{R}^p |\ \alpha_{U,k} \cdot u \leq \beta_{U,k},\ \forall k \in \mathcal{K}_U\}$$

où $\alpha_{D,k} \in \mathbb{R}^m$, $\beta_{D,k} \in \mathbb{R}$, $\alpha_{U,k} \in \mathbb{R}^p$, $\beta_{U,k} \in \mathbb{R}$, $\mathcal{K}_D$ et $\mathcal{K}_U$ sont deux ensemble finis d'indices.

On dénote $R_D = [\underline{d_1}, \overline{d_1}] \times \cdots \times [\underline{d_m}, \overline{d_m}]$ la boîte enveloppante du polytope D, qui est le plus petit domaine rectangulaire contenant D ; et par $V_X = \{\underline{x_1}, \overline{x_1}\} \times \cdots \times \{\underline{x_n}, \overline{x_n}\}$ $V_D = \{\underline{d_1}, \overline{d_1}\} \times \cdots \times \{\underline{d_m}, \overline{d_m}\}$ les ensembles des sommets de R_X et R_D. Le présent travail traite la synthèse de contrôleurs pour une notion d'invariance robuste définie comme suit :

Définition 8. *Considérons un ensemble d'états $P \subseteq R_X$ et un contrôleur $h : R_X \to U$, le système contrôlé*

$$\dot{x}(t) = f(x(t), d(t)) + g(x(t), d(t))h(x(t)),\ d(t) \in D, \tag{8.2}$$

est dit P-invariant si toutes les trajectoires vérifiant $x(0) \in P$ satisfont $x(t) \in P$ pour tout $t \geq 0$.

Remarquons que c'est une notion d'invariance robuste puisque elle doit avoir lieu pour toutes perturbations possibles. Soit $\underline{P} \subseteq \overline{P} \subseteq R_X$ deux polytopes compacts et convexes. On va considérer le problème de synthèse d'un contrôleur h pour le système (8.1) tel que toutes les trajectoires contrôlées initialisées à l'intérieur de $\underline{P}$

restent dans $\overline{P}$. Ceci peut être vu comme étant une propriété de sûreté où $\underline{P}$ est l'ensemble des états initiaux et $\overline{P}$ est l'ensemble des états sûres. Le problème peut être résolu en synthétisant conjointement un contrôleur et un invariant polyédral $P \subseteq R_X$ contenant $\underline{P}$ et inclus dans $\overline{P}$:

Problème 2. *Trouver un contrôleur $h : R_X \to U$ et un polytope convexe et compact P vérifiant $\underline{P} \subseteq P \subseteq \overline{P}$ tel que le système contrôlé (8.2) soit P-invariant.*

Dans ce qui suit, on va décrire une approche permettant de résoudre ce problème. Pour limiter l'espace de recherche, on va utiliser les expressions paramétrés des modèles pour le contrôleur h et l'invariant P. D'abord, on va imposer les orientations des faces du polytope P en choisissant les vecteurs normaux dans l'ensemble $\{\gamma_k \in \mathbb{R}^n |\ k \in \mathcal{K}_X\}$ où $\mathcal{K}_X$ est un ensemble fini d'indice. Ainsi, le polytope P peut être écrit sous la forme :

$$P = \{x \in \mathbb{R}^n |\ \gamma_k \cdot x \leq \eta_k,\ \forall k \in \mathcal{K}_X\}$$

où le vecteur $\eta \in \mathbb{R}^{|\mathcal{K}_X|}$ que l'on veut déterminer, spécifie la position des faces. Les faces de P sont notées par F_k pour $k \in \mathcal{K}_X$, où

$$F_k = \{x \in \mathbb{R}^n |\ \gamma_k \cdot x = \eta_k \text{ et } \gamma_i \cdot x \leq \eta_i,\ \forall i \in \mathcal{K}_X \setminus \{k\}\}.$$

Pour simplifier, on va supposer que les polytopes $\underline{P}$ et $\overline{P}$ sont de la forme :

$$\underline{P} = \{x \in \mathbb{R}^n |\ \gamma_k \cdot x \leq \underline{\eta}_k,\ \forall k \in \mathcal{K}_X\}$$

$$\overline{P} = \{x \in \mathbb{R}^n |\ \gamma_k \cdot x \leq \overline{\eta}_k,\ \forall k \in \mathcal{K}_X\}.$$

Ainsi, la condition $\underline{P} \subseteq P \subseteq \overline{P}$ se traduit en $\underline{\eta}_k \leq \eta_k \leq \overline{\eta}_k$, $\forall k \in \mathcal{K}_X$.

Ensuite, on va chercher un contrôleur h dans un sous ensemble engendré par la matrice polynomiale :

$$h(x) = H(x)\theta$$

où $\theta \in \mathbb{R}^q$ est un paramètre que l'on déterminera et la matrice $H : \mathbb{R}^n \to \mathbb{R}^{(p \times q)}$ est donné à l'aide d'un ensemble de polynômes multi-variés. L'utilisation d'un modèle est naturelle lors de la recherche d'un contrôleur avec une structure particulière. La contrainte associée à ce contrôleur (c'est à dire pour tout $x \in R_X$, $h(x) \in U$) est ainsi équivalente à

$$\forall k \in \mathcal{K}_U,\ \forall x \in R_X,\ \alpha_{U,k} \cdot H(x)\theta \leq \beta_{U,k}. \tag{8.3}$$

Sous ces hypothèses, la dynamique du système contrôlé (8.2) peut être écrite sous la forme

$$\dot{x}(t) = f(x(t), d(t)) + G(x(t), d(t))\theta,\ d(t) \in D,$$

où $G(x, d) = g(x, d)H(x)$.
D'après la caractérisation standard des invariants (voir [7]), il en découle que le système contrôlé (8.2) est P-invariant si et seulement si

$$\forall k \in \mathcal{K}_X,\ \forall x \in F_k, \forall d \in D,\ \gamma_k \cdot (f(x, d) + G(x, d)\theta) \leq 0. \tag{8.4}$$

Ainsi, le Problème 2 peut être résolu en calculant les vecteurs $\theta \in \mathbb{R}^q$ et $\eta \in \mathbb{R}^{|\mathcal{K}_X|}$ avec $\underline{\eta}_k \leq \eta_k \leq \overline{\eta}_k$ pour tout $k \in \mathcal{K}_X$, et tel que les inégalités (8.3) et (8.4) auront lieu.

Dans ce qui suit, on montre d'abord comment, étant donné un vecteur $\eta \in \mathbb{R}^{|\mathcal{K}_X|}$ (et donc un polytope P), on pourra calculer, en utilisant la programmation linéaire, le paramètre θ (et ainsi le contrôleur h) tel que le système contrôlé (8.2) soit P-invariant. Puis, on montre comment on calcule conjointement le contrôleur h et le polytope P en utilisant une approche itérative dont les ingrédients sont principalement ceux utilisés dans le chapitre précédent : la relaxation basée sur la forme polaire et l'analyse de sensibilité des programmes linéaires.

8.2 Synthèse d'un contrôleur

Etant donné un polytope $P = \{x \in \mathbb{R}^n | \ \gamma_k \cdot x \leq \eta_k, \ \forall k \in \mathcal{K}_X\}$, on va montrer dans cette section comment construire un contrôleur $h : R_X \to U$ tel que le système (8.2) soit P-invariant. Comme on l'a expliqué auparavant, on cherche un contrôleur dans un sous ensemble engendré par une matrice polynomiale : $h(x) = H(x)\theta$ où $\theta \in \mathbb{R}^q$. Soit $F(x,d) = f(x,d) + G(x,d)\theta$, c'est un polynôme de degré $\delta_1, \ldots, \delta_n, \rho_1, \ldots, \rho_m$ pour les variables $x_1, \ldots, x_n, d_1, \ldots, d_m$. Sa forme polaire est $F_b = f_b + G_b\theta$ où f_b et G_b désignent les formes polaires des polynômes f et G vus comme étant des polynômes de degrés $\delta_1, \ldots, \delta_n, \rho_1, \ldots, \rho_m$. Soit H_b la forme polaire associée à la matrice H vue comme étant un polynôme de degré $\delta_1, \ldots, \delta_n$ pour les variables $x_1, \ldots, x_n$. Soit $R'_X = [\underline{x_1}, \overline{x_1}]^{\delta_1} \times \cdots \times [\underline{x_n}, \overline{x_n}]^{\delta_n}$, $V'_X = \{\underline{x_1}, \overline{x_1}\}^{\delta_1} \times \cdots \times \{\underline{x_n}, \overline{x_n}\}^{\delta_n}$, $V'_D = \{\underline{d_1}, \overline{d_1}\}^{\rho_1} \times \cdots \times \{\underline{d_m}, \overline{d_m}\}^{\rho_m}$, $\overline{V'_X} = V'_X/\cong$ et $\overline{V'_D} = V'_D/\cong$.

On établit tout d'abord des conditions suffisantes vérifiant que $h(x) \in U$ pour tout $x \in R_X$:

Lemme 6. *Si $\forall l \in \mathcal{K}_\mathcal{U}$,$\forall \overline{v_X} \in \overline{V'_X}$, on a $\alpha_{U,l} \cdot H_b(\overline{v_X})\theta \leq \beta_{U,l}$, alors $h(x) \in U$ pour tout $x \in R_X$.*

Démonstration. Si pour tout $l \in \mathcal{K}_\mathcal{U}$, pour tout $\overline{v_X} \in \overline{V'_X}$, $\alpha_{U,l} \cdot H_b(\overline{v_X})\theta \leq \beta_{U,l}$, alors en utilisant la troisième propriété de la forme polaire on aura pour tout $l \in \mathcal{K}_\mathcal{U}$ et tout $v_X \in V'_X$, $\alpha_{U,l} \cdot H_b(v_X)\theta \leq \beta_{U,l}$. Comme H_b est une fonction multi-affine, alors d'après le Lemme 1 (lemme de convexité) on en déduit que pour tout $l \in \mathcal{K}_\mathcal{U}$ et tout $z \in R'_X$, $\alpha_{U,l} \cdot H_b(z)\theta \leq \beta_{U,l}$. Ainsi, en utilisant la propriété diagonale de la forme polaire, on montre que pour tout $l \in \mathcal{K}_\mathcal{U}$ et tout $x \in R_X$, $\alpha_{U,l} \cdot H(x)\theta \leq \beta_{U,l}$. Ce qui est équivalent à dire que pour tout $x \in R_X$, $h(x) \in U$. □

Le résultat précédent donne un ensemble fini de contraintes linéaires qui doit être satisfait par le paramètre θ. Maintenant, on va établir des conditions assurant que le polytope P sera invariant pour le système (8.2).
Soit $k \in \mathcal{K}_X$, on dit qu'une face F_k du polytope P est bloquée si pour tout $x \in F_k$, pour tout $d \in D$, $\gamma_k \cdot (f(x,d) + G(x,d)\theta) \leq 0$. Il est clair que le système (8.2) est P-invariant si et seulement si toutes les faces sont bloquées.

Lemme 7. *Soit $\theta \in \mathbb{R}^q$ et $k \in \mathcal{K}_X$, alors la face F_k est bloquée si et seulement si la valeur optimale $p_k^*(\theta)$ du problème d'optimisation suivant est positive :*

$$\begin{array}{lll} \textit{minimiser} & -\gamma_k \cdot (f(x,d) + G(x,d)\theta) & \\ s.c & x \in R_X, d \in R_D & \\ & \alpha_{D,j} \cdot d \leq \beta_{D_j}, & j \in \mathcal{K}_D, \\ & \gamma_i \cdot x \leq \eta_i, & i \in \mathcal{K}_X \setminus \{k\}, \\ & \gamma_k \cdot x = \eta_k. & \end{array} \tag{8.5}$$

Une borne inférieure $d_k^*(\theta)$ de $p_k^*(\theta)$ est donnée par la valeur optimale du programme linéaire :

$$\begin{array}{ll} \textit{maximiser} & t \\ s.c & t \in \mathbb{R},\ \lambda^k \in \mathbb{R}^{|\mathcal{K}_X|}, \tilde{\lambda}^k \in \mathbb{R}^{|\mathcal{K}_D|}, \\ & \lambda_i^k \geq 0, i \in \mathcal{K}_X \setminus \{k\}, \\ & \tilde{\lambda}_j^k \geq 0, j \in \mathcal{K}_D, \\ & t \leq -\gamma_k \cdot (f_b(\overline{v_X}, \overline{v_D}) + G_b(\overline{v_X}, \overline{v_D})\theta) + \sum_{i=1}^{|\mathcal{K}_X|} \lambda_i^k (\gamma_i{}' \cdot \overline{v_X} - \eta_i) \\ & + \sum_{j=1}^{|\mathcal{K}_D|} \tilde{\lambda}_j^k (\alpha'_{D,j} \cdot \overline{v_D} - \beta_{D,j}), \quad \overline{v_X} \in \overline{V_X}', \overline{v_D} \in \overline{V_D}', \end{array} \tag{8.6}$$

où pour tout $i \in \mathcal{K}_X$ et tout $j \in \mathcal{K}_D$ les vecteurs $\gamma_i{}'$ et $\alpha'_{D,j}$ sont donnés par :

$$\gamma_i{}' = (\frac{\gamma_{i,1}}{\delta_1}, \ldots, \frac{\gamma_{i,1}}{\delta_1}, \ldots, \frac{\gamma_{i,n}}{\delta_n}, \ldots, \frac{\gamma_{i,n}}{\delta_n})$$

$$\alpha'_{D,j} = (\frac{\alpha_{D,j,1}}{\rho_1}, \ldots, \frac{\alpha_{D,j,1}}{\rho_1}, \ldots, \frac{\alpha_{D,j,m}}{\rho_m}, \ldots, \frac{\alpha_{D,j,m}}{\rho_m}).$$

Démonstration. En remarquant que $F_k = R_X \cap F_k$ et $D = R_D \cap D$, la première partie de la Proposition est triviale. Pour la deuxième partie, on commence par remarquer que d'après la définition de la relation d'équivalence $\cong$ (voir Section 3.3.1), on a $(V'_X \times V'_D)/\cong$ est égale à $\overline{V_X}' \times \overline{V_D}'$. Ainsi, on a juste à appliquer notre relaxation linéaire décrite dans la Section 5.1 au couple (x, d) à la place de x et dont le polynôme p est égale à $f + G\theta$. □

Maintenant on va montrer comment choisir $\theta \in \mathbb{R}^q$ tel que le contrôleur associé vérifie pour tout $x \in R_X$, $h(x) \in U$ et que le système contrôlé (8.2) soit P-invariant.

Proposition 19. *Soit d^* et $\left(t^*, (\lambda^{k*})_{k\in\mathcal{K}_X}, (\tilde{\lambda}^{k*})_{k\in\mathcal{K}_D}, \theta^*\right)$ la valeur optimale ainsi*

que la solution optimale du programme linéaire suivant :

$$
\begin{array}{ll}
\textit{maximiser} & t \\
\textit{s.c} & t \in \mathbb{R},\ , \theta \in \mathbb{R}^q, \\
& \lambda^k \in \mathbb{R}^{|\mathcal{K}_X|}, \tilde{\lambda}^k \in \mathbb{R}^{|\mathcal{K}_D|},\ \ k \in \mathcal{K}_{\mathcal{X}}, \\
& \lambda_i^k \geq 0,\ \ k \in \mathcal{K}_{\mathcal{X}}, i \in \mathcal{K}_X \setminus \{k\}, \\
& \tilde{\lambda}_j^k \geq 0,\ \ k \in \mathcal{K}_{\mathcal{X}}, j \in \mathcal{K}_D, \\
& \alpha_{U,l} \cdot H_b(\overline{v_X})\theta \leq \beta_{U,l},\ \ l \in \mathcal{K}_{\mathcal{U}}, \overline{v_X} \in \overline{V_X'}, \\
& t \leq -\gamma_k \cdot (f_b(\overline{v_X}, \overline{v_D}) + G_b(\overline{v_X}, \overline{v_D})\theta) + \sum_{i=1}^{|\mathcal{K}_X|} \lambda_i^k(\gamma_i{}' \cdot \overline{v_X} - \eta_i) \\
& + \sum_{j=1}^{|\mathcal{K}_D|} \tilde{\lambda}_j^k(\alpha_{D,j}' \cdot \overline{v_D} - \beta_{D,j}),\ \ k \in \mathcal{K}_{\mathcal{X}}, \overline{v_X} \in \overline{V_X}', \overline{v_D} \in \overline{V_D}'.
\end{array}
\tag{8.7}
$$

Alors, si d^* est positive, le contrôleur $h(x) = H(x)\theta^*$ satisfait pour tout $x \in R_X$, $h(x) \in U$ et le système contrôlé (8.2) est P-invariant.

Démonstration. On commence tout d'abord par remarquer que le problème (8.7) est équivalent au problème d'optimisation suivant :

$$
\begin{array}{lll}
\text{maximiser} & t & \\
\text{s.c} & t \in \mathbb{R},\ \theta \in \mathbb{R}^q & \\
& \alpha_{U,l} \cdot H_b(\overline{v_X})\theta \leq \beta_{U,l}, & l \in \mathcal{K}_{\mathcal{U}}, \quad \overline{v_X} \in \overline{V_X'}, \\
& t \leq d_k^*(\theta), & k \in \mathcal{K}_{\mathcal{X}}
\end{array}
\tag{8.8}
$$

où $d_k^*(\theta)$ est la valeur optimale du programme linéaire (8.6). Ainsi, si $d^* \geq 0$, ceci signifie que pour la valeur optimale θ^*, on a pour tout $k \in \mathcal{K}_X$, $d_k^*(\theta^*) \geq 0$. De plus, d'après le Lemme 7, toutes les faces de P sont bloquées et par conséquent le système contrôlé (8.2) est P-invariant. Les contraintes sur θ garantissent d'après le Lemme 6 que pour tout $x \in R_X$, $h(x) \in U$. □

8.3 Synthèse conjointe du contrôleur et de l'invariant

Dans cette section, on va présenter une approche itérative pour la synthèse à la fois du contrôleur h ainsi que l'invariant polyédral P permettant de résoudre le Problème 2. L'approche est basée sur l'analyse de sensibilité des programmes linéaires. A chaque itération, on se donne un polytope P candidat pour la résolution de notre problème d'invariance. Puis, en suivant l'approche décrite dans la section précédente, on essaye de construire un contrôleur h qui rend P invariant. Si on ne parvient pas à rendre P invariant, on utilise l'analyse de sensibilité des programmes linéaires (8.7) pour modifier P et obtenir une nouvelle estimation de l'invariant polyédral. La procédure sera répétée jusqu'à ce que le Problème 2 soit résolu.

8.3.1 Analyse de sensibilité

Soit $\eta, \mu \in \mathbb{R}^{|\mathcal{K}_X|}$, fixons les polytopes :

$$P = \{x \in \mathbb{R}^n |\ \gamma_k \cdot x \leq \eta_k,\ \forall k \in \mathcal{K}_X\}$$

$$P_\mu = \{x \in \mathbb{R}^n | \ \gamma_k \cdot x \leq \eta_k + \mu_k, \ \forall k \in \mathcal{K}_X\}$$

P_μ peut être vu comme étant une perturbation du polytope P. On va adapter le résultat du Théorème 11 du chapitre précédent se basant sur l'analyse de sensibilité au cas de nos programmes linéaires (8.7). Le résultat principal est donné à l'aide de la proposition suivante :

Proposition 20. *Soit d^* et $\left(t^*, (\lambda^{k*})_{k\in\mathcal{K}_X}, (\tilde{\lambda}^{k*})_{k\in\mathcal{K}_D}, \theta^*\right)$ la valeur optimale ainsi que la solution optimale du programme linéaire (8.7). Soit d^*_μ la valeur optimale du programme linéaire (8.7) où η a été remplacé par $\eta + \mu$. On a alors :*

$$d^*_\mu \geq \min_{k\in\mathcal{K}_X}(d^* - \lambda^{k*} \cdot \mu).$$

Démonstration. La preuve est similaire à celle du Théorème 11, en effet :
Pour tout $k \in \mathcal{K}_X$, pour tout $\overline{v_X} \in \overline{V_X}'$ et $\overline{v_D} \in \overline{V_D}'$, on a :

$$\begin{aligned}
& -\gamma_k \cdot (f_b(\overline{v_X}, \overline{v_D}) + G_b(\overline{v_X}, \overline{v_D})\theta) + \sum_{i=1}^{|\mathcal{K}_X|} \lambda_i^{k*}({\gamma_i}' \cdot \overline{v_X} - \eta_i - \mu_i) \\
& + \sum_{j=1}^{|\mathcal{K}_D|} \tilde{\lambda}_j^{k*}(\alpha'_{D,j} \cdot \overline{v_D} - \beta_{D,j}) \\
= \ & -\gamma_k \cdot (f_b(\overline{v_X}, \overline{v_D}) + G_b(\overline{v_X}, \overline{v_D})\theta) + \sum_{i=1}^{|\mathcal{K}_X|} \lambda_i^{k*}({\gamma_i}' \cdot \overline{v_X} - \eta_i) \\
& + \sum_{j=1}^{|\mathcal{K}_D|} \tilde{\lambda}_j^{k*}(\alpha'_{D,j} \cdot \overline{v_D} - \beta_{D,j}) - \lambda^{k*} \cdot \mu \\
\geq \ & t^* - \lambda^{k*} \cdot \mu \geq \min_{k'\in\mathcal{K}_X}(t^* - \lambda^{k'*} \cdot \mu).
\end{aligned}$$

Ceci montre que $\left(\min_{k\in\mathcal{K}_X}(t^* - \lambda^{k*} \cdot \mu), (\lambda^{k*})_{k\in\mathcal{K}_X}, (\tilde{\lambda}^{k*})_{k\in\mathcal{K}_D}, \theta^*\right)$ est une solution admissible pour le programme linéaire (8.7) où η a été remplacé par $\eta+\mu$. Il en découle que $d^*_\mu \geq \min_{k\in\mathcal{K}_X}(t^* - \lambda^{k*} \cdot \mu)$ ce qui implique que $d^* = t^*$. □

Le résultat précédent a les conséquences suivantes. Supposons que l'on ne soit pas capable en résolvant le programme linéaire (8.7) de synthétiser un polytope P qui rende notre système (8.2) P-invariant, cela signifie que $d^* \leq 0$. Ainsi, le résultat précédent nous dit comment obtenir une modification P_μ du polytope P afin d'obtenir ${d^*}_\mu \geq 0$ (ou au moins obtenir une amélioration c'est à dire ${d^*}_\mu \geq d^*$). Cela donne à penser que nous pouvons résoudre le Problème 2 en utilisant une approche itérative décrite dans le paragraphe suivant.

8.3.2 Approche itérative

Initialement, on suppose que l'on a un candidat initial pour le polytope P ; on peut par exemple utiliser $\overline{P}$ mais d'autres choix sont possibles. On utilise une approche itérative pour résoudre le Problème 2 ; chaque itération consiste en deux étapes principales.

Première étape : synthèse d'un contrôleur

Etant donné un polytope $P = \{x \in \mathbb{R}^n | \ \gamma_k \cdot x \leq \eta_k, \ \forall k \in \mathcal{K}_X\}$, on utilise la Proposition 19 pour la synthèse d'un contrôleur h. Soit d^* et $\left(t^*, (\lambda^{k*})_{k\in\mathcal{K}_X}, (\tilde{\lambda}^{k*})_{k\in\mathcal{K}_D}, \theta^*\right)$ la valeur optimale ainsi que la solution optimale du programme linéaire (8.7). Si $d^* \geq 0$, alors on a trouvé un contrôleur rendant P invariant pour le système contrôlé (8.2) et le Problème 2 est résolu. Si $d^* < 0$, alors on passe à la deuxième étape.

Deuxième étape : modifier le polytope

Maintenant on essaye de trouver $\mu \in \mathbb{R}^{|\mathcal{K}_X|}$ assurant que le polytope

$$P_\mu = \{x \in \mathbb{R}^n | \ \gamma_k \cdot x \leq \eta_k + \mu_k, \ \forall k \in \mathcal{K}_X\}$$

soit invariant pour le système contrôlé (8.2). A ce propos, la Proposition 20 nous dit qu'il suffit d'avoir $d^* - \lambda^{k*} \cdot \mu \geq 0$, pour tout $k \in \mathcal{K}_X$. La condition $\underline{P} \subseteq P_\mu \subseteq \overline{P}$ peut être translatée à $\underline{\eta_k} - \eta_k \leq \mu_k \leq \overline{\eta}_k - \eta_k$ pour tout $k \in \mathcal{K}_X$. De plus, comme l'analyse de sensibilité est pertinente essentiellement pour des petites perturbations, on impose que pour tout $k \in \mathcal{K}_X$, $-\varepsilon \leq \mu_k \leq \varepsilon$ où ε est un paramètre à régler. Ainsi, trouver un μ convenable peut être fait en résolvant le programme linéaire suivant :

$$\begin{array}{ll} \text{maximiser} & t \\ \text{s.c} & t \in \mathbb{R}, \ \mu \in \mathbb{R}^{|\mathcal{K}_X|}, \\ & t \leq d^* - \lambda^{k*} \cdot \mu, \quad k \in \mathcal{K}_X, \\ & \min(-\varepsilon, \underline{\eta_k} - \eta_k) \leq \mu_k \leq \max(-\varepsilon, \underline{\eta_k} - \eta_k), k \in \mathcal{K}_X. \end{array} \tag{8.9}$$

Soit (t^*, μ^*) une solution de ce programme linéaire. Si la valeur optimale t^* de ce problème est positive alors on peut montrer que le contrôleur $h : R_X \to U$ calculé dans la première étape et le polytope P_{μ^*} résolvent le Problème 2. Sinon, si $t^* < 0$, alors on reprend la première étape et on commence une nouvelle itération avec $P = P_{\mu^*}$.

8.3.3 Complexité de l'approche

Discutons brièvement de la complexité de notre approche. Chaque itération consiste principalement à résoudre deux programmes linéaires. Le programme linéaire (8.7) a $1+q+|\mathcal{K}_X|(|\mathcal{K}_X|+|\mathcal{K}_D|)$ variables et $|\mathcal{K}_X|(|\mathcal{K}_X|+|\mathcal{K}_D|+|\overline{V'_X}|+|\overline{V'_D}|-1)+|\mathcal{K}_U||\overline{V'_X}|$ contraintes d'inégalité. Remarquons que $|\overline{V'_X}| = (\delta_1+1)\times\cdots\times(\delta_n+1)$ et $|\overline{V'_D}| = (\rho_1+1)\times\cdots\times(\rho_n+1)$. Comme la complexité de la programmation linéaire est polynomiale en moyenne dans le nombre de variables et des contraintes, il en découle que la première étape de l'itération a un coût polynomial dans le nombre de contraintes des polytopes P, D et U et dans les degrés des polynômes. Le programme linéaire (8.9) a $1 + |\mathcal{K}_X|$ variables et $2|\mathcal{K}_X|$ contraintes d'inégalité. Il en découle qu'une itération de la deuxième étape a un coût polynomial dans le nombre des contraintes du polytope P.

8.4 Exemples

Notre approche est implémentée sous Matlab et appliquée à plusieurs exemples.

8.4.1 Modèle de Moore-Greitzer d'un moteur à réaction

On réutilise ici le modèle du moteur de Moore-Greitzer [98] avec sa version non stabilisée et avec une certaine perturbation :

$$\begin{cases} \dot{x}_1 &= -x_2 - \frac{3}{2}x_1^2 - \frac{1}{2}x_1^3 + d, \\ \dot{x}_2 &= u. \end{cases} \tag{8.10}$$

On va tout d'abord travailler dans le rectangle $R_X = [-0.2, 0.2]^2$ avec une perturbation $d \in D = R_D = [-0.02, 0.02]$. On veut construire un contrôleur linéaire c'est à dire $h(x_1, x_2) = \theta_1 x_1 + \theta_1 x_2$ tel que $h(x) \in U = [-0.35, 0.35]$, pour tout $x \in R_X$. Fixons $\underline{P}$ et $\overline{P}$ comme étant des polytopes avec $m = 24$ faces dont les orientations sont uniformément distribuées et tangentes aux cercles de centre $(0, 0)$ et de rayons 0.01 et 0.2, respectivement. En utilisant notre approche, on trouve le contrôleur $h(x_1, x_2) = 0.8076x_1 - 0.9424x_2$ qui rend le polytope P (voir Figure 8.1) invariant.

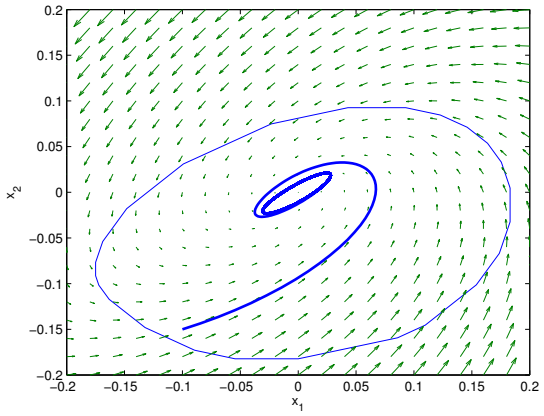

FIGURE 8.1 – Polytope invariant P avec 24 faces et une trajectoire de (8.10) illustrant l'invariance pour une perturbation $d(t) = 0.02\cos(0.5t)$.

On réalise une deuxième expérimentation en travaillant dans le rectangle $R_X = [-0.2, 0.2]^2$ avec une perturbation $d \in D = R_D = [-0.025, 0.025]$. Maintenant on veut construire un contrôleur polynomial dont les degrés sont 3 en x_1 et 1 en x_2 (les mêmes que le champ de vecteur). $\underline{P}$ et $\overline{P}$ sont des polytopes avec $m = 8$ faces dont les orientations sont uniformément distribuées et tangentes aux cercles de centre $(0, 0)$ et de rayon 0.01 et 0.2, respectivement. En utilisant notre approche, on trouve un contrôleur qui rend le polytope P (voir Figure 8.2) invariant. Cette dernière expérimentation montre qu'en cherchant un contrôleur de plus grand degré, on est capable de trouver des invariants plus simples pour des perturbations plus grandes.

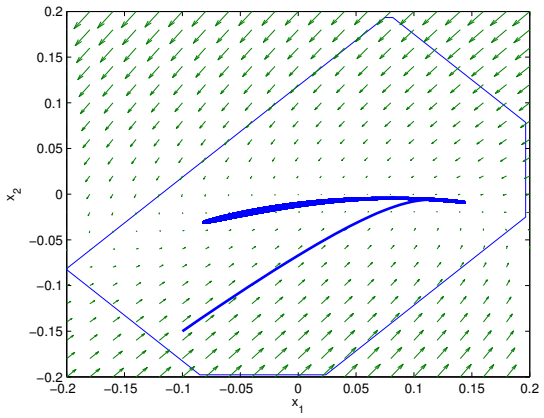

FIGURE 8.2 – Polytope invariant P avec 8 faces et une trajectoire (8.10) illustrant l'invariance pour une perturbation $d(t) = 0.025\cos(0.1t)$.

8.4.2 Le modèle uni-cycle

Maintenant on va considérer un simple modèle d'un uni-cycle :

$$\begin{cases} \dot{x} &= v\cos(\varphi), \\ \dot{y} &= v\sin(\varphi), \\ \dot{\varphi} &= \omega, \end{cases}$$

où v et ω sont les entrées du système représentant respectivement la vitesse et la vitesse angulaire. Dans la suite, on va considérer v comme étant une perturbation et ω comme étant le contrôleur. En utilisant le changement de coordonnés $z_1 = x\cos(\varphi) + y\sin(\varphi)$, et $z_2 = x\sin(\varphi) - y\cos(\varphi)$, on obtient le système suivant :

$$\begin{cases} \dot{z}_1 &= v - z_2\omega, \\ \dot{z}_2 &= z_1\omega. \end{cases} \tag{8.11}$$

On va travailler dans le rectangle $R_X = [-0.1, 0.1] \times [0.9, 1.1]$ avec une perturbation $v \in D = R_D = [0.96, 1.04]$. On veut synthétiser un contrôleur affine $h(z_1, z_2) = \theta_0 + \theta_1 z_1 + \theta_2 z_2$. Dans cet exemple, on n'impose pas de contraintes sur les valeurs des entrées. $\underline{P}$ et $\overline{P}$ sont définis comme étant des polytopes avec $m = 24$ faces dont les orientations sont uniformément distribuées et tangentes aux cercles de centre $(0, 1)$ et de rayon 0.01 et 0.1 respectivement. En utilisant notre approche, on trouve un contrôleur $h(z_1, z_2) = 1.0178 + 1.8721z_1 - 0.0253z_2$ qui rend le polytope de la Figure 8.3 invariant.

8.4.3 Mouvement de corps rigide

Le dernier exemple est un modèle décrivant le mouvement d'un corps rigide. Il est emprunté à [36] :

$$\begin{cases} \dot{x}_1 &= u_1, \\ \dot{x}_2 &= u_2, \\ \dot{x}_3 &= x_1 x_2. \end{cases} \tag{8.12}$$

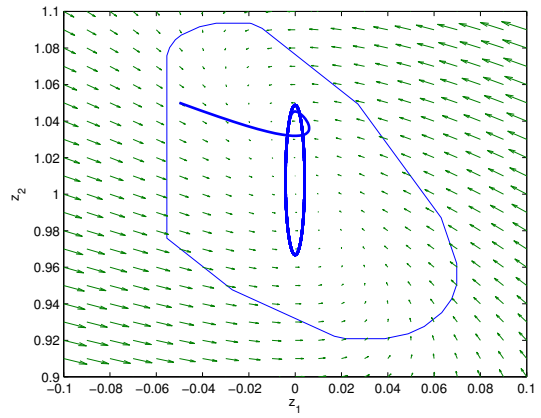

FIGURE 8.3 – Polytope invariant P avec 24 faces et une trajectoire de (8.11) illustrant l'invariance pour une perturbation $v(t) = 1 + 0.04\cos(0.1t)$.

On prend $R_X = [-0.2, 0.4] \times [-0.2, 0.2] \times [-0.2, 0.4]$. Dans cet exemple, on ne considère pas les perturbations. On veut construire un contrôleur multi-affine c'est à dire $h : \mathbb{R}^3 \to \mathbb{R}^2$ (défini par seize paramètres), tels que $h(x) \in U = [-1, 1]^2$, pour tout $x \in R_X$. En utilisant notre approche, on trouve un contrôleur qui rend le polytope de 18 faces que montre la Figure 8.4, invariant.

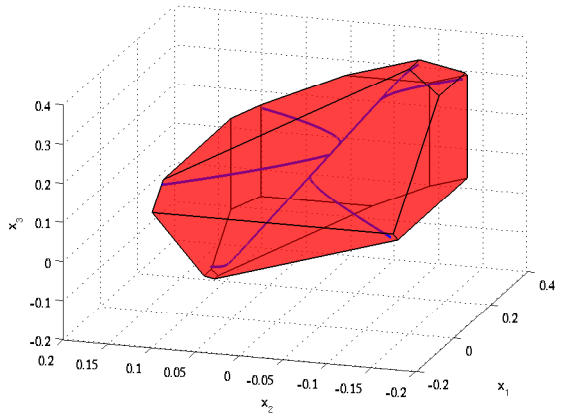

FIGURE 8.4 – Polytope invariant P avec 18 faces et des trajectoires de (8.12) illustrant l'invariance.

Dans ce chapitre, on a examiné le problème de synthèse de contrôleurs assurant l'invariance robuste des systèmes dynamiques polynomiaux. En utilisant notre relaxation de POP se basant sur le principe de floraison, nous avons développé une approche itérative pour résoudre ce problème. Elle est principalement basé sur la programmation linéaire et elle est donc efficace. Nous avons montré plusieurs exemples d'applications démontrant l'utilité de l'approche.

Chapitre 9

Analyse et contrôle des systèmes dynamiques polynomiaux dans des rectangles

Dans ce chapitre on décrira une approche permettant d'étudier les systèmes dynamiques polynomiaux dans des domaines rectangulaires (des boîtes). Il s'agit d'une généralisation du travail de Belta [13] traitant des systèmes dynamiques multi-affines dans des rectangles. En effet, l'idée principale sera l'utilisation du principe de floraison pour passer d'un système dynamique polynomial à un système dynamique multi-affine. Ce passage sera appelé abstraction et le nouveau système sera appelé système abstrait.

On peut aussi voir ce chapitre comme étant un cas particulier de l'étude faite dans le chapitre précédent [15] où les ensembles polyédraux sont pris comme des boîtes et les perturbations sont nulles. Pour ce cas particulier, on verra que la complexité des programmes linéaires sera réduite par rapport à ceux du chapitre précédent.

Dans un premier temps, on étudiera des problèmes d'analyse de systèmes dynamiques polynomiaux dans des domaines rectangulaires. Ensuite, on étendra les résultats dans le cadre de synthèse d'un contrôleur polynomial. Enfin, une application consistant à traiter un problème de planification des trajectoires sera fournie.

9.1 Abstractions multi-affines pour des systèmes polynomiaux

On va considérer le système dynamique S suivant :

$$\dot{x}(t) = f(x(t)) = (f_1(x(t)), \ldots, f_n(x(t))), \tag{9.1}$$

où $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ est un champ de vecteurs polynomial.

Dans cette section on verra comment passer du système dynamique polynomial (9.1) à un système dynamique multi-affine. Ce passage sera appelé abstraction et notre outil principal sera le principe de floraison.

On commence par noter $\delta_{1,j}, \ldots, \delta_{n,j}$ les degrés respectifs des variables $x_1, \ldots, x_n$ du polynôme f_j pour tout $j \in \{1, \ldots, n\}$ et soit $\delta_i = \max_{j \in \{1,\ldots,n\}} \delta_{i,j}$ pour tout $i \in$

$\{1,\dots,n\}$. Remarquons que tout polynôme f_i peut être considéré comme étant un polynôme de degré $\delta_1,\dots,\delta_n$ dans les variables respectives $x_1,\dots,x_n$ (en ajoutant des coefficients nuls si nécessaire).
Maintenant notons $f_{i,\delta}$ la forme polaire de f_i relative à δ pour tout $i=1,\dots,n$ et soit $f_\delta=(f_{1,\delta},\dots,f_{n,\delta})$.
Pour tout $\alpha\geq 0$, notre système dynamique abstrait ${S'}_\alpha$ sera donné par :

$$\dot{z}(t)=g_\alpha(z(t))=(g_{\alpha,1,1}(z(t)),..,g_{\alpha,1,\delta_1}(z(t)),..,g_{\alpha,n,1}(z(t)),..,g_{\alpha,n,\delta_n}(z(t))), \quad (9.2)$$

où le champ de vecteurs $g_\alpha:\mathbb{R}^{\delta_1+\cdots+\delta_n}\to\mathbb{R}^{\delta_1+\cdots+\delta_n}$ est donné par :

$$g_{\alpha,i,j}(z)=f_{i,\delta}(z)+\alpha\left(\sum_{k\in\{1,\dots,\delta_i\}\setminus\{j\}} z_{i,k}-(\delta_i-1)z_{i,j}\right),$$

pour tout $i=1\dots,n$ et tout $j=1,\dots,\delta_i$, avec $z=(z_{1,1},\dots,z_{1,\delta_1},\dots,z_{n,1},\dots,z_{n,\delta_1})$.

On a aussi besoin de définir l'espace vectoriel H donné par l'ensemble suivant :

$$H:=\{z\in\mathbb{R}^{\delta_1+\cdots+\delta_n}|\quad \forall i\in\{1,\dots,n\},\quad z_{i,j}\text{ sont égaux }\forall j\in\{1,\dots,\delta_i\}\}.$$

L'intérêt de la méthode d'abstraction provient du théorème suivant :

Théorème 12. *Pour tout $\alpha\geq 0$, les systèmes S et ${S'}_\alpha$ sont équivalents sur l'espace vectoriel H ; c'est à dire que si x,z sont des trajectoires respectives des systèmes S et ${S'}_\alpha$ telles que $x_i(0)=z_{i,j}(0)$ pour tout $i=1,\dots,n$, $j=1,\dots,\delta_i$ ($z(0)\in H$) alors $x_i(t)=z_{i,j}(t)$ pour tout $t\geq 0$.*
De plus, pour $\alpha>0$ l'espace H est un attracteur pour le système ${S'}_\alpha$.

Démonstration. En utilisant la propriété diagonale de la forme polaire on montre facilement que les systèmes (9.1) et (9.2) sont équivalents si on se restreint à l'ensemble H. Maintenant, soit $i\in\{1,\dots,n\}$ tel que $\delta_i>1$ et soient j_1 et j_2 deux éléments distincts de $\{1,\dots,\delta_i\}$. On a :

$$\dot{z}_{i,j_1}(t)-\dot{z}_{i,j_2}(t)=-\delta_i\alpha(z_{i,j_1}(t)-z_{i,j_2}(t))$$

Ce qui implique que pour tout $\alpha>0$:

$$z_{i,j_1}(t)-z_{i,j_2}(t)=C.e^{-\delta_i\alpha t}\longrightarrow_{t\to+\infty}0$$

□

Exemple 11. *Comme illustration, on va considérer comme système dynamique S l'oscillateur de Van Der Pol [135].*

$$(S)\quad\begin{cases}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= x_2(1-{x_1}^2)-x_1.\end{cases}$$

Il est bien connu que le système admet un cycle limite comme le montre la Figure 9.1.
Pour tout $\alpha\geq 0$, le système dynamique abstrait ${S'}_\alpha$ sera donné par :

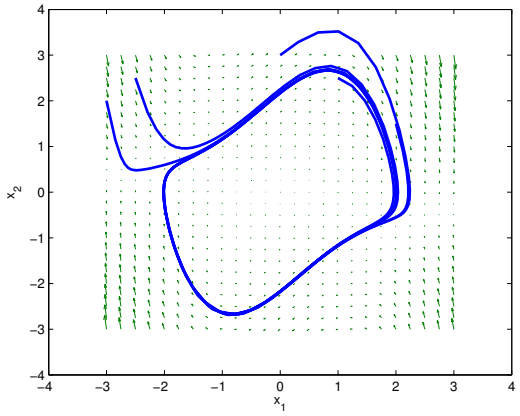

FIGURE 9.1 – Quelques trajectoires du système S montrant l'existence d'un cycle limite.

$$(S'_\alpha) \quad \begin{cases} \dot{z}_{1,1} &= \frac{1}{2}(z_{1,1}+z_{1,2}) + \alpha(z_{1,2}-z_{1,1}), \\ \dot{z}_{1,2} &= \frac{1}{2}(z_{1,1}+z_{1,2}) + \alpha(z_{1,1}-z_{1,2}), \\ \dot{z}_{2,1} &= z_{2,1}(1-z_{1,1}z_{1,2}) - \frac{1}{2}(z_{1,1}+z_{1,2}). \end{cases}$$

Pour un réel $\alpha = 50$, on a tracé (voir Figure 9.2) quelques trajectoires du système S'_{50} dont celles ayant les mêmes conditions initiales que les trajectoires de la Figure 9.1 après abstraction. Par exemple, la trajectoire correspondant à la condition initiale $x_0 = (-3, 2)$ sera remplacée par la trajectoire ayant comme condition initiale $z_0 = (-3, -3, 2)$. On peut remarquer ainsi l'équivalence entre le système initial S et le système S'_{50} sur l'attracteur $H := \{z \in \mathbb{R}^3 | \ \ z_{1,1} = z_{1,2}\}$. En effet, sur la Figure 9.2 on récupère bien le cycle limite donné par la Figure 9.1.

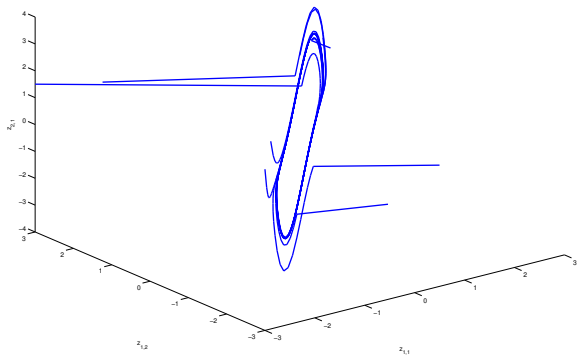

FIGURE 9.2 – Quelques trajectoires du système S'_{50} montrant l'équivalence avec le système S sur l'attracteur H.

9.2 Analyse des systèmes polynomiaux dans des rectangles

Dans cette section, on va appliquer notre méthode d'abstraction pour trouver des conditions permettant la résolution des problèmes suivants :

9.2.1 Formulation des problèmes et résultats préliminaires

Soit $R_n = \prod_{k=1}^{k=n}[a_k, b_k]$ un rectangle donné de $\mathbb{R}^n$.

Problème 1 (Sortie du rectangle) :
Il consiste à trouver des conditions suffisantes et (ou) nécessaires assurant la sortie de toutes les trajectoires du système (9.1) du rectangle R_n en temps fini.
Problème 2 (Invariance du rectangle) :
Il consiste à trouver des conditions suffisantes et (ou) nécessaires assurant que toutes les trajectoires du système (9.1) commençant dans R_n restent dans R_n, c'est à dire si $x(t)$ est une trajectoire du système (9.1) satisfaisant $x(t_0) \in R_n$, alors $x(t) \in R_n$ pour tout $t \geq t_0$.
Problème 3 (Sortie par une face donnée du rectangle) :
Il consiste à trouver, étant donné une face F_0 du rectangle R_n, des conditions suffisantes et (ou) nécessaires assurant la sortie de toutes les trajectoires du système (9.1) du rectangle R_n à travers la face F_0.

Soit F une face arbitraire du rectangle R_n et n_F sa normale sortante. Pour pouvoir résoudre nos différents problèmes, on aura besoin des définitions suivantes :

Définition 9. *(Ensemble de sortie pour une face) :*
L'ensemble de sortie d'une face F qu'on note F^* est l'ensemble défini par :
$F^* = \overline{\{x \in F \quad | \quad n_F.f(x) > 0\}}$ *avec $\overline{E}$ dénote la fermeture de l'ensemble E.*

Définition 10. *(Face bloquée) :*
La face F est dite bloquée si son ensemble de sortie est vide.
Il est évident que F est bloquée si et seulement si : $\forall x \in F, n_F.f(x) \leq 0$.
Si cette inégalité est stricte on dit que la face est strictement bloquée.

Le résultat [83] donné par le lemme suivant est également nécessaire :

Lemme 8. *Si une trajectoire $x(t)$ du système (9.1) commençant dans R_n quitte R_n à travers un point x^* d'une face F, alors $x^* \in F^*$.*

Dans [83], le résultat est donné pour le cas d'un champ de vecteurs affine. La preuve peut être facilement adaptée dans le cas d'un champ de vecteurs polynomial. Comme application directe de ce lemme on a le corollaire suivant :

Corollaire 2. *Si F est une face bloquée alors toutes les trajectoires du système (9.1) commençant dans R_n ne peuvent pas quitter R_n à travers cette face.*

Démonstration. F est une face bloquée alors F^* est vide, or s'il existe une trajectoire $x(t)$ commençant dans R_n qui quitte R_n à travers un point x^* de la face F alors par le Lemme 8 on déduit que $x^* \in F^*$ ce qui contredit le fait que F^* est vide. □

Ainsi, on dispose déjà du résultat préliminaire suivant :

Proposition 21. *La résolution des problèmes d'invariance et de sortie par une face donnée peut se faire grâce aux équivalences suivantes :*

1. *Le* **Problème 2** *est résolu* $\Longleftrightarrow$ *Pour toute face* F *de* R_n, F^* *est vide.*
2. *Le* **Problème 3** *est résolu pour une face* F_0 $\Longleftrightarrow$ *Le* **Problème 1** *est résolu et pour toute face* F *de* R_n *telle que* $F \neq F_0$, F^* *est vide.*

Démonstration. On commence par montrer la première équivalence :
$\Longleftarrow$: Si F^* est vide pour toute face F de R_n alors toutes les faces de R_n sont bloquées. Ainsi, d'après le Corollaire 2, le rectangle R_n est invariant.
$\Longrightarrow$: Supposons qu'il existe une face de R_n telle que F^* est non vide. Il suffit alors de prendre $x^* \in F^*$ pour prouver l'absurdité puisque toute trajectoire du système (9.1) initialisée en x^* va quitter R_n.
Maintenant démontrons la deuxième équivalence :
$\Longrightarrow$: Comme toutes les trajectoires du système (9.1) quittent R_n, le **Problème 1** est résolu. Supposons qu'il existe une face F_1 de R_n différente de F_0 telle que ${F_1}^*$ est non vide. Il suffit alors de prendre $x^* \in {F_1}^*$ pour prouver l'absurdité puisque toute trajectoire du système (9.1) initialisée en x^* va quitter R_n par la face F_1.
$\Longleftarrow$: On se fixe une trajectoire arbitraire $x(t)$ du système (9.1) commençant dans R_n. Comme le **Problème 1** est résolu on sait déjà que $x(t)$ va quitter le rectangle R_n. Et puisque toutes les faces de R_n différentes de F_0 sont bloquées, alors forcément cette trajectoire quittera R_n à travers la face F_0. □

9.2.2 Cas d'un champ de vecteurs multi-affine

On suppose dans un premier temps que le champ de vecteurs f du système (9.1) est multi-affine ($\delta_1 = \cdots = \delta_n = 1$).

Problème 1 : Sortie du rectangle

Une condition suffisante assurant la sortie du rectangle est la suivante :

Théorème 13. *Toutes les trajectoires du système (9.1) commençant dans* R_n *à l'instant* $t = t_0$ *quitteront* R_n *s'il existe une direction* $\vec{n}$ *telle que :*

$$\vec{n}.f(v) > 0 \text{ pour tout } v \in V_n.$$

Démonstration. On commence tout d'abord par remarquer qu'en utilisant le Lemme 1, l'existence d'une direction $\vec{n}$ telle que $\vec{n}.f(v) > 0$ pour tout $v \in V_n$ est équivalente à l'existence d'une direction $\vec{n}$ telle que $\vec{n}.f(x) > 0$ pour tout $x \in R_n$.
D'autre part, l'existence d'une direction $\vec{n}$ vérifiant $\vec{n}.f(x) > 0$ pour tout $x \in R_n$ est équivalente à dire que : $\vec{n}.\dot{x}(t) > 0$ pour toute trajectoire $x(t) \in R_n$. Ainsi, on en déduit que $\vec{n}.x(t)$ croit au cours du temps pour toute trajectoire $x(t) \in R_n$.
Comme R_n est borné, toutes les trajectoires du système (9.1) commençant à l'intérieur de R_n à l'instant $t = t_0$ quitteront R_n. □

Remarque 17. *Il est évident que si la condition "il existe une direction $\vec{n}$ telle que $\vec{n}.f(v) > 0$ pour tout $v \in V_n$" dans le Théorème 13 est remplacée par "il existe une direction $\vec{n}$ telle que $\vec{n}.f(x) > 0$ pour tout $x \in D$", le résultat reste valable pour tout champ de vecteurs f de classe C^1 et tout ensemble borné D.*

Numériquement, la vérification d'existence et le calcul d'une telle direction $\vec{n}$ peut se réduire à un problème d'admissibilité d'un programme linéaire. En effet, il suffit de résoudre le programme linéaire suivant :

$$\begin{array}{ll} \text{minimiser} & p \cdot y \\ \text{s.c} & y \in \mathbb{R}^n \\ & Ay > 0. \end{array} \tag{9.3}$$

où le vecteur $p \in \mathbb{R}^n$ est fixé arbitrairement et $A \in \mathbb{R}^{2^n \times n}$ est une matrice dont les lignes sont données par les valeurs au sommets $f(v)$ pour tout $v \in V_n$.
En effet, dans le programme linéaire (9.3), on ne s'intéresse pas au calcul de la valeur optimale (ce qui explique le choix arbitraire du vecteur p) mais plutôt à l'existence d'un vecteur $y^* \in \mathbb{R}^n$ admissible ; c'est à dire vérifiant $Ay^* > 0$. Dans ce cas, la direction $\vec{n} = y^*$ résout notre problème de sortie.
On va appliquer le résultat précédent sur un exemple dans le cas plan.

Exemple 12. *On prend $R = [0,1]^2$ et $f : \mathbb{R}^2 \mapsto \mathbb{R}^2$ le champ de vecteurs multi-affine définit comme suit :*

$$\begin{cases} \dot{x} &= f_1(x,y) = 1 + x + y + 2xy, \\ \dot{y} &= f_2(x,y) = 2 + x + 3y + 2xy. \end{cases}$$

On veut vérifier l'existence d'une direction $\vec{n}=\begin{pmatrix} a \\ b \end{pmatrix}$ vérifiant $\vec{n}.f(v) > 0$ pour tout sommet $v \in \{0,1\} \times \{0,1\}$.
On devra ainsi résoudre le problème d'admissibilité suivant :

$$\begin{cases} a + 2b > 0 \\ 2a + 5b > 0 \\ 2a + 3b > 0 \\ 5a + 10b > 0. \end{cases}$$

Dans ce cas, la matrice $A \in \mathbb{R}^{4\times 2}$ est donnée par : $\begin{pmatrix} 1 & 2 \\ 2 & 5 \\ 2 & 3 \\ 5 & 10 \end{pmatrix}$.

Il est évident que $\vec{n}=\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ est admissible pour ce problème. Par conséquent, on pourra affirmer la sortie des trajectoires commençant à l'intérieur du rectangle $R = [0,1]^2$ (voir Figure 9.3).
Maintenant si $R = [-1,1]^2$, notre problème d'admissibilité sera le suivant :

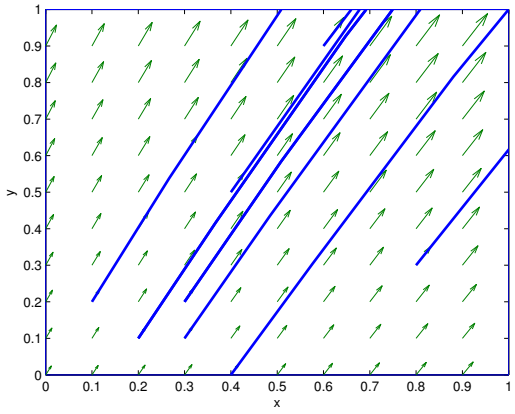

FIGURE 9.3 – Quelques trajectoires du système illustrant la sortie en temps fini

$$\begin{cases} a > 0 \\ -a + 2b > 0 \\ -a - 2b > 0 \\ 5a + 10b > 0. \end{cases}$$

Ce problème n'est pas admissible et donc on ne peut pas trouver une direction monotone $\vec{n}$ assurant la sortie.
Question : Que peut on dire alors concernant la sortie du rectangle $R = [-1, 1]^2$?

Dans le cas plan, une condition nécessaire et suffisante assurant la sortie en temps fini du rectangle est la suivante :

Théorème 14. *En dimension $n = 2$, toutes les trajectoires du système (9.1) commençant dans $R = R_2$ à l'instant $t = t_0$ le quitteront si et seulement si on n'a pas de point d'équilibre à l'intérieur du rectangle.*

Démonstration. Il est évident que si il existe un point d'équilibre à l'intérieur de R alors une trajectoire initialisée en ce point ne le quittera jamais.
Pour la réciproque, on optera pour une démonstration par l'absurde et on utilisera le théorème de Poincaré-Bendixson [93] ainsi que la théorie d'indice [187].
Si il existe pas de point d'équilibre, d'après le théorème de Poincaré-Bendixon (valable juste pour le cas plan), on aura une orbite périodique. Ainsi, d'après la théorie d'indice on aboutit à une contradiction puisque d'une part une orbite périodique a un indice égale à $+1$ et d'autre part l'indice d'une courbe ne contenant aucun point d'équilibre est zéro. □

Exemple 13. *On reprend notre Exemple 12 avec $R = [-1, 1]^2$.*
En utilisant le Théorème 14, on peut répondre à la question concernant la sortie du rectangle R. En effet, il est évident que le système ne contient pas de point d'équilibre ainsi toutes les trajectoires commençant à l'intérieur de R vont le quitter sans qu'il existe une direction monotone comme le montre la Figure 9.4.

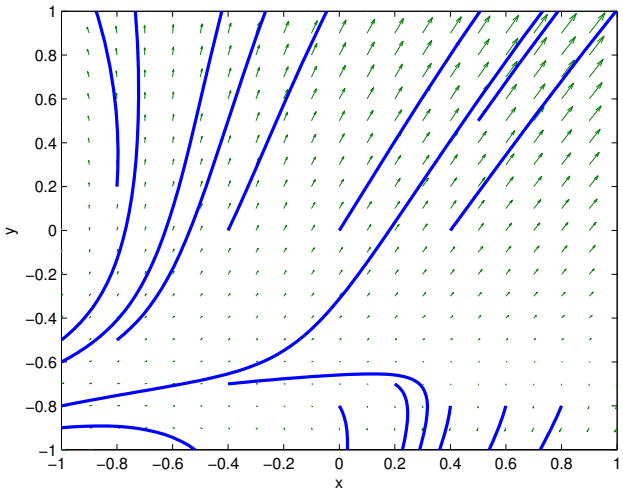

FIGURE 9.4 – Quelques trajectoires du système illustrant la sortie du rectangle.

Remarque 18. *Dans la preuve du Théorème 14, on n'a pas utilisé le fait que le champ de vecteurs est multi-affine. En effet, le résultat reste vrai pour n'importe quel champ de vecteurs de classe* C^1.
D'autre part dans le cas d'un champ de vecteurs f *affine, on aura les équivalences suivantes [156] :*

a) Sortie en temps fini de R_n .

b) $\exists \vec{n}$ *telle que* $\vec{n}.f(x) > 0$, $\forall x \in R_n$.

c) Pas de point d'équilibre à l'intérieur de R_n

Problème 2 : Invariance du rectangle

On va avoir besoin des notations suivantes [13] :
— $\xi_k : \{a_k, b_k\} \mapsto \{0, 1\}$ avec $\xi_k(a_k) = 0$ et $\xi_k(b_k) = 1$ pour tout $k \in \{1, \ldots, n\}$.
— $F_{j,\xi_j(w_j)} = R_n \cap \{x \in \mathbb{R}^n | x_j = w_j\}$ avec $w_j \in \{a_j, b_j\}$ et $j \in \{1, \ldots, n\}$: c'est l'ensemble des faces de R_n.
— $n_{j,\xi_j(w_j)} = (-1)^{(\xi_j(w_j)+1)} e_j$: c'est la normale sortante à la face $F_{j,\xi_j(w_j)}$ où e_j, $j \in \{1, \ldots, n\}$ dénote la base canonique de R^n.

On dispose du théorème suivant [13] :

Théorème 15. *Toutes les trajectoires du système (9.1) commençant dans* R_n *à l'instant* $t = t_0$ *restent dans* R_n *pour tout* $t \geq t_0$ *(*R_n *invariant pour le système (9.1)) si et seulement si* $n_{j,\xi_j(v_j)}.f(v) \leqslant 0$ *pour tout sommet* $v = (v_1, \ldots, v_n) \in V_n$ *et tout entier* $j \in \{1, \ldots, n\}$.

En d'autre termes, ce théorème nous dit qu'un rectangle est invariant sous l'action d'un champ de vecteurs f multi-affine si et seulement si tous les vecteurs champ aux sommets de ce rectangle sont dirigés vers l'intérieur.

Problème 3 : Sortie par une face donnée du rectangle

D'après la Proposition 21, la résolution du **Problème 3** peut se déduire à l'aide de la résolution des problèmes précédents. Plus précisément, on se fixe une face de sortie $F_{j,\xi_j(w_j)}$ de R_n avec $j \in \{1,\dots,n\}$ et $w_j \in \{a_j, b_j\}$.
On dispose du théorème suivant [13] :

Théorème 16. *Toutes les trajectoires du système (9.1) commençant dans R_n le quitteront à travers la face $F_{j,\xi_j(w_j)}$ si pour tout sommet $v = (v_1,\dots,v_n) \in V_n$ on a :*

$$n_{j,\xi_j(w_j)}.f(v) > 0 \text{ et } \forall i \in \{1,\dots,n\},\ i \neq j,\ n_{i,\xi_j(v_i)}.f(v) \leqslant 0.$$

Remarque 19. *Dans la condition du théorème on exige que $F^*_{j,\xi_j(w_j)} = F_{j,\xi_j(w_j)}$ et donc on impose la sortie dès qu'une trajectoire du système atteint un point de cette face. C'est la raison pour laquelle la réciproque est fausse. En effet, on peut bien quitter R_n par la face $F_{j,\xi_j(w_j)}$ sans que tous les points de cette face ne soient des points de sortie.*

9.2.3 Cas d'un champ de vecteurs polynomial

Maintenant, on va étudier le système (9.1) dans le cas général, c'est à dire dans le cas où le champ de vecteurs est polynomial. Pour cela, on va utiliser la méthode d'abstraction décrite dans la section précédente. Ainsi pour un réel $\alpha \geq 0$, on se ramène au système multi-affine S'_α suivant :

$$\dot{z}(t) = g_\alpha(z(t)), \quad z \in R'_n. \tag{9.4}$$

Problème 1 : Sortie du rectangle

En adaptant le résultat du Théorème 13, on obtient le résultat suivant :

Proposition 22. *Toutes les trajectoires du système (9.1) commençant dans R_n le quitteront s'il existe une direction $\vec{w} \in \mathbb{R}^n$ vérifiant $\vec{w}.f_\delta(v) > 0$ pour tout $v \in V'_n$.*

Démonstration. Comme $\vec{w}.f_\delta(v) > 0$ pour tout $v \in V'_n$ et f_δ est multi-affine, on en déduit d'après le Lemme 1 que $\vec{w}.f_\delta(z) > 0$ pour tout $z \in R'_n$.
Soit $x = (x_1,\dots,x_n) \in R_n$ et posons $z_x = (x_1,\dots,x_1,\dots,x_n,\dots,x_n)$.
Puisque $z_x \in R'_n$ alors $\vec{w}.f_\delta(z_x) > 0$. Or, d'après la propriété diagonale d'une fonction multi-affine on aura : $f(x) = f_\delta(z_x)$.
Par conséquent, on a montré que $\vec{w}.f(x) > 0$ pour tout $x \in R_n$. Ainsi, toutes les trajectoires du système (9.1) commençant dans R_n le quitteront. □

On peut remarquer que le choix de α pour la résolution du problème de sortie n'a pas d'importance et donc on peut prendre $\alpha = 0$.

Problème 2 : Invariance du rectangle

Pour pouvoir appliquer le Théorème 15 dans le cas d'un champ de vecteurs polynomial, on va utiliser la proposition suivante :

Proposition 23. *S'il existe $\alpha \geq 0$ tel que le rectangle $R'_n = [a_1, b_1]^{\delta_1} \times \cdots \times [a_n, b_n]^{\delta_n}$ est invariant pour le système S'_α alors le rectangle $R_n = [a_1, b_1] \times \cdots \times [a_n, b_n]$ est invariant pour le système S.*

Démonstration. Soit $x(t)$ une trajectoire du système S vérifiant $x(0) \in R_n$ et posons $z(t) = (x_1(t), \ldots, x_1(t), \ldots, x_n(t), \ldots, x_n(t)) \in \mathbb{R}^{\delta_1 + \cdots + \delta_n}$. Comme $z(0) \in R'_n$ et R'_n est invariant pour le système S'_α alors $z(t) \in R'_n$ pour tout $t \geq 0$. Par conséquent $x(t) \in R_n$ pour tout $t \geq 0$ et donc R_n est invariant pour le système S. □

Ce résultat nous donne une condition suffisante à notre problème d'invariance. En effet, la manière la plus simple sera de prendre $\alpha = 0$ et essayer de vérifier que R'_n est invariant pour le système S'_0 pour en déduire l'invariance de R_n pour le système dynamique S. Cependant, pour ce cas particulier la réciproque est fausse comme le montre l'exemple suivant.

Exemple 14. *On considère le système dynamique suivant :*

$$\dot{x} = f(x) = -3x^2 + x + 1, \quad x \in [0, 1]. \tag{9.5}$$

Comme $f(0) = 1 > 0$ et $f(1) = -1 < 0$ alors l'intervalle $[0, 1]$ est strictement invariant[1] *(voir Figure 9.5). Cependant, en utilisant notre abstraction pour $\alpha = 0$,*

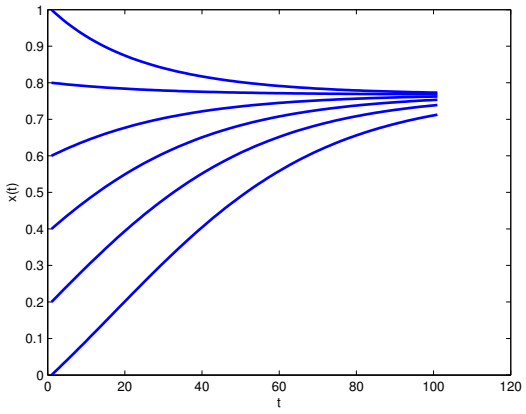

FIGURE 9.5 – Quelques trajectoires du système (9.5)

on aura le système suivant :

$$\dot{z} = g(z), \quad z = (z_1, z_2) \in [0, 1]^2. \tag{9.6}$$

où $g : \mathbb{R}^2 \mapsto \mathbb{R}^2$ est le champ de vecteurs donné par :

$$\begin{cases} g_1(z_1, z_2) & = \dot{z_1} & = -3z_1z_2 + \frac{1}{2}(z_1 + z_2) + 1. \\ g_2(z_1, z_2) & = \dot{z_2} & = -3z_1z_2 + \frac{1}{2}(z_1 + z_2) + 1. \end{cases}$$

Il est facile de voir que le rectangle $[0, 1]^2$ n'est pas invariant pour le système (9.5). En effet, pour qu'il le soit, il faut d'après le Théorème 15 que le champ de vecteurs

1. Les contraintes à vérifier sont strictes

g soit dirigé vers l'intérieur du rectangle pour chaque sommets.
On prend par exemple le sommet $v_1 = (1, 0)$.
Une face contenant ce sommet est $F_1 = \{z \in [0,1]^2 \mid z_2 = 1\}$.
Une normale sortante associée à cette face est $\vec{n_1} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ *et* $\vec{n_1} \cdot g(v_1) = \frac{3}{2} > 0$.
Ainsi, pour le sommet v_1*, le champ de vecteurs g n'est pas dirigé vers l'intérieur du rectangle* $[0,1]^2$ *et par conséquent* $[0,1]^2$ *n'est pas invariant pour le système (9.5). Maintenant si on prend* $\alpha = 2$*, notre champ de vecteurs g devient :*

$$\begin{cases} g_1(z_1, z_2) & = \dot{z_1} & = -3z_1z_2 + \frac{1}{2}(z_1 + z_2) + 1 + 2(z_2 - z_1). \\ g_2(z_1, z_2) & = \dot{z_2} & = -3z_1z_2 + \frac{1}{2}(z_1 + z_2) + 1 + 2(z_1 - z_2). \end{cases}$$

Dans ce cas, on peut vérifier que les 8 *contraintes du Théorème 15 sont vérifiées (notamment* $\vec{n_1} \cdot g(v_1) = -\frac{1}{2} < 0$*). Par conséquent, en prenant* $\alpha = 2$*, on pourra démontrer l'invariance de l'intervalle* $[0,1]$ *pour le système (9.5).*

Pour exploiter le résultat de la Proposition 23, on a besoin de généraliser les notations d'une face et de sa normale sortante pour le rectangle R'_n.
Plus précisément, pour tout $i \in \{1, \ldots, n\}$ et tout $k \in \{1, \ldots, \delta_i\}$, on notera :

- $F^k_{i,\xi_i(w_i)} = R'_n \cap \{z \in \mathbb{R}^{\delta_1+\cdots+\delta_n} | z_{i,k} = w_i\}$ pour tout $w_i \in \{a_i, b_i\}$: c'est l'ensemble des faces de R'_n.
- $n^k_{i,\xi_i(w_i)} = (-1)^{(\xi_i(w_i)+1)} e_{i,k}$ où les $e_{i,k}$ forment la base canonique de $\mathbb{R}^{\delta_1+\cdots+\delta_n}$: c'est la normale sortante associée à la face $F^k_{i,\xi_i(w_i)}$.

Corollaire 3. *Le rectangle* R_n *est invariant s'il existe un réel* $\alpha \geq 0$ *vérifiant pour tout sommet* $v \in {V_n}'$ *:*

$$n^k_{i,\xi_i(v_{i,k})} g_\alpha(v) \leq 0 \text{ pour tout } i \in \{1, \ldots, n\} \text{ , } k \in \{1, \ldots, \delta_i\}. \tag{9.7}$$

Démonstration. On suppose qu'il existe un réel $\alpha \geq 0$ tel que les conditions (9.7) seront vérifiées pour tout sommet $v \in {V_n}'$. Ainsi, le Théorème 15 implique que le rectangle R'_n est invariant pour le système ${S'}_\alpha$. Par conséquent, il suffit d'utiliser la Proposition 23 pour montrer que le rectangle R_n est invariant pour le système S. □

Problème 3 : Sortie par une face donnée

A l'aide des notations précédentes on va pouvoir généraliser le résultat du Théorème 16 :

Proposition 24. *Toutes les trajectoires du système (9.1) commençant dans* R_n *le quitteront à travers une face fixée* $F_{j,\xi_j(w_j)}$ *s'il existe un réel* $\alpha \geq 0$ *vérifiant pour tout sommet* $v = (v_1, \ldots, v_n) \in V'_n$ *:*

$$n^k_{j,\xi_j(w_j)} . g_\alpha(v) > 0, \quad \forall k \in \{1, \ldots, \delta_j\} \tag{9.8}$$

et pour tout $i \in \{1, \ldots, n\}$ *tel que* $i \neq j$ *:*

$$n^k_{i,\xi_i(v_{i,k})} . g_\alpha(v) \leqslant 0, \ \forall k \in \{1, \ldots, \delta_i\}. \tag{9.9}$$

Démonstration. La preuve est la généralisation de celle du Théorème 16 dans le cas de notre abstraction. En effet, les conditions (9.9) impliquent que toutes les faces de R_n différentes de $F_{j,\xi_j(w_j)}$ sont bloquées. Maintenant, il suffit d'utiliser les conditions (9.8) pour prouver que la direction $\vec{n} = n_{j,\xi_j(w_j)}$ est une direction monotone de sortie. Par conséquent, on déduit d'après la Proposition 21 que toutes les trajectoires du système (9.1) commençant dans R_n le quitteront à travers $F_{j,\xi_j(w_j)}$. □

Dans la section suivante, on va étudier les réductions qui pourront être établies dans le cadre du **Problème 1** (Proposition 22) ainsi que le **Problème 2** (Corollaire 3) puis déduire celles concernant le **Problème 2** (Proposition 24). On établira aussi le lien avec Bernstein.

9.2.4 Réduction de la complexité

Dans la section précédente, des résultats concernant la vérification d'invariance ainsi que la sortie en temps fini pour des domaines rectangulaires ont été établis se basant sur la méthode d'abstraction. Cependant, la grande dimension du nouveau système abstrait S'_α rend cette vérification trop coûteuse. Pour cela, on se propose d'établir un ensemble de réduction se basant essentiellement sur les principales propriétés de la forme polaire. On verra aussi que tous les résultats qu'on obtient peuvent être exprimés à l'aide des coefficients de Bernstein.
On considère le système (9.1) et on note δ le multi-indice désignant le degré maximal du champ de vecteur f. Notons pour tout $i = 1, \ldots, n$, $f_{i,\delta}$ la forme polaire de f_i par respect de δ et soit $f_\delta = (f_{1,\delta}, \ldots, f_{n,\delta})$.

Problème 1 : Sortie du rectangle

Pour la sortie du rectangle, on va réduire la complexité du résultat donné par la Proposition 22 grâce à la proposition suivante :

Proposition 25. *Toutes les trajectoires du système (9.1) commençant dans R_n à l'instant $t = t_0$ quitteront R_n en un temps fini s'il existe une direction $\vec{w} \in \mathbb{R}^n$ telle que : $\vec{w}.f_\delta(\overline{v}) > 0$ pour tout sommet $\overline{v} \in \overline{V_n}'$.*

Démonstration. Comme f_δ est multi-affine, il suffit d'utiliser la relation d'équivalence $\cong$ pour réduire l'ensemble de sommets V'_n en $\overline{V_n}'$. □

On a pu passer de la recherche d'une direction vérifiant $2^{\delta_1+\cdots+\delta_n}$ contraintes à la recherche d'une direction ne vérifiant que $(\delta_1 + 1) \times \cdots \times (\delta_n + 1)$ contraintes.

Problème 2 : Invariance du rectangle

Pour voir les réductions possibles concernant le résultat du Corollaire 3, on va commencer par étudier l'exemple suivant :

Exemple 15. *On considère le système dynamique suivant :*

$$\dot{x} = ax^2 + bx + c, \quad x \in R = [0,1]. \tag{9.10}$$

où a, b et c sont des paramètres réels.
Notons que pour les valeurs $a = -3$, $b = 1$ et $c = 1$ on retrouve bien le cas de l'exemple 9.5.
On se propose de déterminer les conditions dépendantes de nos paramètres permettant de vérifier que le rectangle $[0,1]$ est strictement invariant (les faces sont strictement bloquées) pour le système (9.10).
Par la méthode d'abstraction, on aura le système dynamique suivant :

$$\begin{cases} \dot{z_1} &= az_1z_2 + \frac{b}{2}(z_1+z_2) + c + \alpha(z_2 - z_1) \\ \dot{z_2} &= az_1z_2 + \frac{b}{2}(z_1+z_2) + c + \alpha(z_1 - z_2). \end{cases}$$

où $z \in R' = [0,1]^2$.
Ainsi, pour vérifier l'invariance, on doit vérifier 2×4 contraintes (deux par sommets) données comme suit :

— *$g_\alpha(0,0) = (c,c)$ donc les deux conditions se réduisent en une seule : $c \geq 0$.*

— *$g_\alpha(1,1) = (a+b+c, a+b+c)$ donc les deux conditions se réduisent en une seule : $a+b+c \leq 0$.*

— *$g_\alpha(0,1) = (\frac{b}{2}+c+\alpha, \frac{b}{2}+c-\alpha)$ donc on doit vérifier que : $\frac{b}{2}+c+\alpha \geq 0$ et $\frac{b}{2}+c-\alpha \leq 0$.*

— *$g_\alpha(1,0) = (\frac{b}{2}+c-\alpha, \frac{b}{2}+c+\alpha)$ donc on doit vérifier que : $\frac{b}{2}+c-\alpha \leq 0$ et $\frac{b}{2}+c+\alpha \geq 0$.*

Tout d'abord, il est évident que pour $\alpha = 0$ on ne pourra pas trouver des paramètres assurant l'invariance stricte du rectangle R' puisque l'on aura l'absurdité suivante :

$$\frac{b}{2} + c > 0 \text{ et } \frac{b}{2} + c < 0.$$

De plus, les sommets $(0,1)$ et $(1,0)$ (équivalents pour la relation d'équivalence $\cong$) donnent exactement les mêmes conditions donc on peut les réduire en un seul sommet et on se retrouve ainsi avec 4 contraintes à vérifier. A l'aide des coefficients de Bernstein associés au polynôme $p(x) = ax^2 + bx + c$, ces contraintes sont les suivantes :

$$b_{0,2}(p) \geq 0, b_{1,2}(p) + \alpha \geq 0, b_{1,2}(p) - \alpha \leq 0 \text{ et } b_{2,2}(p) \leq 0.$$

Enfin, les conditions $\frac{b}{2}+c+\alpha \geq 0$ et $\frac{b}{2}+c-\alpha \leq 0$ peuvent être vérifiées indépendamment des valeurs des paramètres b et c en prenant un α suffisamment grand. Par conséquent, on se retrouve avec uniquement deux contraintes : $c = f(0) \geq 0$ et $a+b+c = f(1) \leq 0$. Ainsi, on retrouve les conditions optimales pour que $R = [0,1]$ soit invariant pour le système (9.10).

Dans l'exemple précédent, on remarque que le nombre de contraintes à vérifier pour le problème d'invariance se réduit de 8 à 2 contraintes seulement et que ces contraintes peuvent être exprimées à l'aide des coefficients de Bernstein.

Le résultat dans le cas général est donné à l'aide du théorème suivant :

Théorème 17. *Le rectangle R_n est invariant pour le système (9.1) si pour tout sommet $\overline{v} \in \overline{V_n}'$ et pour tout $i = 1, \ldots, n$, les contraintes suivantes sont vérifiées :*
- *Si $l_i(\overline{v}) = 0$, il faut que $f_{i,\delta}(\overline{v}) \geq 0$.*
- *Si $l_i(\overline{v}) = \delta_i$, il faut que $f_{i,\delta}(\overline{v}) \leq 0$,*

où la définition de $l_i(v)$ pour tout $i = 1, \ldots, n$ est donnée dans la Section 3.3.

Démonstration. Comme g_α est multi-affine, d'après la propriété de symétrie d'une telle fonction il suffit de vérifier que les conditions (9.7) du Corollaire 3 sont vérifiées pour tout sommet $\overline{v} \in \overline{V_n}'$.
On fixe arbitrairement un réel $\alpha \geq 0$, des entiers $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, \delta_i\}$ et un sommet $\overline{v} \in \overline{V_n}'$. On sait que :

$$n^j_{i,\xi_i(\overline{v}_{i,j})} g_\alpha(\overline{v}) = (-1)^{(\xi_i(\overline{v}_{i,j})+1)} \left(g_{i,j}(\overline{v}) + \alpha C\right),$$

avec $C = \sum_{k \in \{1,\ldots,\delta_i\} \setminus \{j\}} \overline{v}_{i,k} - (\delta_i - 1)\overline{v}_{i,j}$.

Si $0 < l_i(\overline{v}) < \delta_i$:
Dans le cas où $\overline{v}_{i,j} = a_i$ on aura : $C > 0$ et $(-1)^{(\xi_i(a_i)+1)} = -1$. Ainsi la condition (9.7) du Corollaire 3 peut s'écrire sous la forme $-g_{i,j}(\overline{v}) - \alpha C \leq 0$. Comme le choix de $\alpha \geq 0$ est arbitraire, il suffit de prendre un réel α suffisamment grand pour que cette contrainte soit vérifiée.
Dans le cas où $\overline{v}_{i,j} = b_i$ on aura une contrainte de la forme $g_{i,j}(\overline{v}) + \alpha C \leq 0$, avec une constante $C < 0$. Cette contrainte pourra également être satisfaite en prenant un réel α suffisamment grand.

Si $l_i(\overline{v}) = 0$:
On aura $C = 0$ et $(-1)^{(\xi_i(\overline{v}_{i,j})+1)} = -1$, ainsi la condition (9.7) du Corollaire 3 peut s'écrire sous la forme : $g_{i,j}(\overline{v}) = f_{i,\delta}(\overline{v}) \geq 0$.

Si $l_i(\overline{v}) = \delta_i$:
On aura $C = 0$ et $(-1)^{(\xi_i(\overline{v}_{i,j})+1)} = 1$, ainsi la condition (9.7) du Corollaire 3 peut s'écrire sous la forme : $g_{i,j}(\overline{v}) = f_{i,\delta}(\overline{v}) \leq 0$. □

Une interprétation géométrique de ce résultat est la suivante :
Pour un sommet $\overline{v} \in \overline{V_n}'$, on peut trouver l'élément $x_{\overline{v}} \in R_n$ qui lui correspond, où pour tout $j = 1, \ldots, n$, la j-ième composante de $x_{\overline{v}}$ est égale à $x_{\overline{v},j} = \frac{v_{j,1}+\cdots+v_{j,\delta_j}}{\delta_j}$.
Ainsi, suivant la position de l'élément $x_{\overline{v}}$ dans R_n, les degrés de libertés pour les vecteurs champ assurant l'invariance diffèrent.
Une simple illustration est donnée par la Figure 9.6 à l'aide d'un rectangle de $\mathbb{R}^2$ et quelques éléments pris dans des positions différentes du rectangle. Pour chacun de ces points, on doit vérifier des conditions concernant le signe du champ de vecteurs $f = (f_1, f_2) = (\dot{x}, \dot{y})$. On peut voir par exemple que pour un point se situant à l'intérieur du rectangle, toutes les directions sont possibles et par conséquent le

champ de vecteurs n'a pas de condition à vérifier en un tel point.
Dorénavant, on notera $R = R_n$ et $V = V_n$.

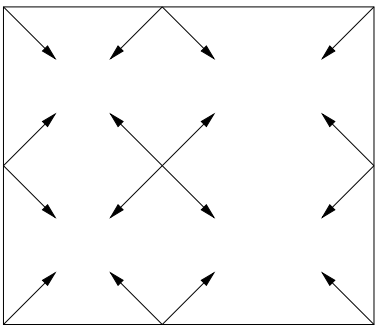

FIGURE 9.6 – Les champs de vecteurs assurant l'invariance pour quelques points du rectangle

Pour tout $j = 1, \ldots, n$ et tout $\overline{v} \in \overline{V'}$, on pose :

$$\overline{V'}_j = \{v \in \overline{V'} \text{ vérifiant } l_j(\overline{v}) = 0 \text{ ou } \delta_j\} \text{ et } \sigma_j(\overline{v}) = \begin{cases} 1 & \text{si } l_j(\overline{v}) = 0 \\ -1 & \text{si } l_j(\overline{v}) = \delta_j \\ 0 & \text{sinon} \end{cases}$$

En utilisant cette notation on aura le résultat suivant :

Proposition 26. *Si la valeur*

$$\tilde{p_\delta}^* = \min_{j=1,\ldots,n,\ \overline{v} \in \overline{V'}_j} \sigma_j(\overline{v}) f_{j,\delta}(\overline{v}) \tag{9.11}$$

est positive alors le rectangle R est invariant pour le système (9.1).

Démonstration. On suppose que $\tilde{p_\delta}^* \geq 0$ et on fixe $j \in \{1, \ldots, n\}$ et $\overline{v} \in \overline{V'}_j$:
Si $l_j(\overline{v}) = 0$ alors $0 \leq \sigma_j(\overline{v}) f_{j,\delta}(\overline{v}) = f_{j,\delta}(\overline{v})$ et donc $f_{j,\delta}(\overline{v}) \geq 0$.
Si $l_j(\overline{v}) = \delta_j$ alors $0 \leq \sigma_j(\overline{v}) f_{j,\delta}(\overline{v}) = -f_{j,\delta}(\overline{v})$ et donc $f_{j,\delta}(\overline{v}) \leq 0$.
Ainsi, d'après le Théorème 23, on déduit que le rectangle R est invariant pour le système (9.1). □

Remarque 20. *Dans le Chapitre 7, le problème d'invariance a été traité pour des polytopes. Si le polytope P est pris égal à un rectangle R, un résultat similaire à celui de la Proposition 26 peut être obtenu. Cependant, il vaut mieux utiliser le résultat de la Proposition 26 (voir si la valeur $\tilde{p_\delta}^*$ est positive) plutôt que celui de la Proposition 18 (voir si pour chaque face F_k du rectangle R, la valeur $p^*_{k,\delta}$ donnée par le programme (7.9) est positive) puisque plusieurs contraintes inutiles ont été supprimées. Plus précisément, pour une direction $j \in \{1, \ldots, n\}$, on ne considère pas les contraintes données par les sommets $\overline{v} \in \overline{V'}$ vérifiant $0 < l_j(\overline{v}) < \delta_j$.*

L'exemple suivant met en oeuvre cette remarque :

Exemple 16. *On reprend l'exemple neuronal (voir Section 7.3.2 du Chapitre 7). On rappelle que la dynamique du système est donnée par :*

$$\begin{cases} \dot{x}_1 &= x_1 - x_1^3/3 - x_2 + \frac{7}{8}, \\ \dot{x}_2 &= 0.08(x_1 + 0.7 - 0.8x_2), \end{cases}$$

On se propose de vérifier l'invariance du rectangle $R = [-2.5, 2.5] \times [-1.5, 3.5]$. *En utilisant la Proposition 18 du Chapitre 7, on aura* 4 *programmes linéaires à traiter (un par face) contenant chacun* $8 = |\overline{V'}|$ *contraintes. La valeur optimale minimale est égale à* -0.048. *Maintenant, en utilisant le résultat de la Proposition 26, on obtient aussi* $\tilde{p_\delta}^* = -0.048$ *par résolution d'un simple problème de minimisation de* $12 = |\overline{V'}_1| + |\overline{V'}_2|$ *éléments.*
Dans les deux cas, on ne peut pas conclure concernant l'invariance du rectangle R. *Cependant, le gain en complexité est considérable.*

Problème 3 : Sortie par une face donnée

Les résultats sur l'invariance et la sortie ayant établis, on va pouvoir en déduire un résultat concernant la sortie par une face donnée :

Proposition 27. *On se fixe une face de sortie* $F_{j,\xi_j(w_j)}$ *de* R_n *avec* $j \in \{1, \ldots, n\}$ *et* $w_j \in \{a_j, b_j\}$. *Si la valeur :*

$$\min_{i=1,\ldots,n, i\neq j} \min_{\overline{v} \in \overline{V'}_i} \sigma_i(\overline{v}) f_{i,\delta}(\overline{v}) \tag{9.12}$$

est positive et si la valeur :

$$\min_{\overline{v} \in \overline{V'}} (-1)^{(\xi_j(w_j)+1)} f_{j,\delta}(\overline{v}) \tag{9.13}$$

est strictement positive alors toutes les trajectoires du système (9.1) commençant dans R_n *le quitteront à travers la face* $F_{j,\xi_j(w_j)}$.

Démonstration. La preuve est similaire à celle du Théorème 16 en utilisant maintenant les nouveaux résultats de sortie et d'invariance démontrés dans cette section. En effet, si la valeur optimale du problème (9.12) est positive on va pouvoir en déduire que toutes les faces du rectangle R_n différentes de $F_{j,\xi_j(w_j)}$ sont bloquées. De plus, si la valeur optimale du problème (9.13) est strictement positive alors la direction $\vec{n} = n_{j,\xi_j(w_j)}$ est une direction monotone de sortie. □

Remarque 21. *Pour établir le lien avec Bernstein, il suffit de procéder à un changement de variable qu'on note* T *(voir (3.7) du Chapitre 3) permettant de se ramener au cas de la boîte unité puis de poser* $\tilde{f}_j = f_j \circ T^{-1}$ *pour tout* $j = 1, \ldots, n$, $\tilde{f} = (\tilde{f}_1, \ldots, \tilde{f}_n)$ *et* $b_{I,\delta}(\tilde{f}) = (b_{I,\delta}(\tilde{f}_1), \ldots, b_{I,\delta}(\tilde{f}_n))$ *pour tout multi-indice* $I \leq \delta$.
Par conséquent, grâce aux résultats d'équivalence entre forme polaire et coefficients de Bernstein donnés dans la Section 3.3, tous les résultats de cette section pourront être exprimés à l'aide des coefficients de Bernstein.

9.3 Synthèse d'un contrôleur

Dans cette section, comme dans le Chapitre 8, on va généraliser les résultats obtenus précédemment dans le cas d'un système dynamique polynomial contrôlé. Le problème avec contrôle considéré est le suivant :

$$\dot{x}(t) = f(x(t)) + Bu(t), \quad x \in R \tag{9.14}$$

où R est un rectangle de $\mathbb{R}^n$, $B \in \mathbb{R}^{n\times p}$ et $u(t) \in \mathbb{R}^p$ est le contrôleur à calculer.

On va décrire comment on peut construire un contrôleur assurant l'invariance d'un rectangle et un contrôleur assurant la sortie à travers une face donnée d'un rectangle. Rappelons tout d'abord que dans l'approche décrite dans le chapitre précédent, on se fixe la forme du contrôleur en l'écrivant sous la forme $H(x)\theta$ où $H(x)$ est une matrice servant à fixer cette forme (affine, multi-affine, polynomial,..) et θ désigne le vecteur des coefficients qu'on veut déterminer. Cependant, si on se restreint à la recherche d'un contrôleur polynomial ayant le même degré δ que le champ de vecteurs f, on peut construire un contrôleur sans passer par le calcul du vecteur des coefficients θ en calculant directement les valeurs aux sommets de sa forme polaire u_δ où les coefficients de Bernstein de degré δ de $\tilde{u}$.

9.3.1 Problème 2 : Invariance du rectangle

Pour assurer l'invariance d'un rectangle R, il est nécessaire de construire un contrôleur dont les valeurs aux sommets de sa forme polaire permettent de bloquer toutes ses faces. Ceci peut être fait à l'aide de la proposition suivante :

Proposition 28. *Soit $D = |\overline{V'}| = (\delta_1 + 1) \times \cdots \times (\delta_n + 1)$ et B_i dénote le i-éme vecteur de la matrice B.*
Si il existe $\{\lambda = (\lambda_{\overline{v}}) \in \mathbb{R}^{p\times D} \mid \overline{v} \in \overline{V}'\}$ tel que $\sigma_i(\overline{v})\left(f_{i,\delta}(\overline{v}) + B_i\lambda_{\overline{v}}\right) \geq 0$ pour tout $i = 1, \ldots, n$ et tout $\overline{v} \in \overline{V'}_i$, alors on peut construire, en utilisant le vecteur λ un contrôleur polynomial u ayant le même degré que f assurant l'invariance du rectangle R pour le système dynamique (9.14).

Démonstration. Pour la preuve, on commence tout d'abord par décrire la construction du contrôleur polynomial u à partir du vecteur $\lambda = (\lambda_1, \ldots, \lambda_D) \in \mathbb{R}^{p\times D}$:
Notons u_δ la forme polaire relative à δ du contrôleur u qu'on veut construire et fixons $u_\delta(\overline{v}^r) = \lambda_r$ pour tout $r = 1, \ldots, D$ où $\overline{v}^r$ désigne le r-ième élément de $\overline{V'}$. Ainsi, en utilisant les polynômes de Bernstein et la Proposition 9, on peut construire un contrôleur polynomial $u(x)$ pour tout $x \in R$.
Ensuite, on pose $F(x) = f(x) + Bu(x)$ et on applique la Proposition 26 à sa forme polaire $F_\delta = f_\delta + Bu_\delta$ relative à δ pour montrer que le rectangle R est invariant pour le nouveau champ de vecteurs F. □

Remarque 22. *Pour construire le contrôleur polynomial u, on cherche un vecteur λ vérifiant un ensemble d'inégalités linéaires. Ceci peut être fait par résolution d'un programme linéaire, par exemple :*

$$\begin{array}{ll} \textit{maximiser} & t \\ \textit{s.c} & t \in \mathbb{R}, \lambda \in \mathbb{R}^{p\times D} \\ & t \leq \sigma_i(\overline{v})\left(f_{i,\delta}(\overline{v}) + B_i\lambda_{\overline{v}}\right), \quad i = 1, \ldots, n,\ \overline{v} \in \overline{V'}_i. \end{array} \tag{9.15}$$

En effet, si on note (t^*, λ^*) une solution du programme linéaire (9.15), il suffit que t^* soit positive pour que le contrôleur u construit à l'aide de λ^* assure l'invariance du rectangle R.
De plus, si on veut restreindre notre contrôleur à une région bornée U de l'espace $\mathbb{R}^p$, il suffit d'imposer les contraintes données par l'ensemble U sur le vecteur λ dans notre programme linéaire.

Remarque 23. *Si on utilise les coefficients de Bernstein, la construction du contrôleur polynomial u sera comme suit :*
Soit $I_\delta = \{I \in \mathbb{N}^n \text{ tel que } I \leq \delta\}$, comme $|I_\delta| = D$ on peut écrire $I_\delta = \{I_1, \ldots, I_D\}$. Ensuite, pour tout $r = 1, \ldots, D$ soit $b_{I_r,\delta}(\tilde{u}) = \lambda_r$ où $\tilde{u} = u \circ T^{-1}$ et T est la transformation affine qui envoie la boîte unité sur le rectangle R. Les coefficients de Bernstein $b_{I,\delta}(\tilde{u})$ sont connus, on peut construire $\tilde{u}$ en utilisant les polynômes de Bernstein $B_{I,\delta}(\tilde{u})$. Ainsi, en utilisant le fait que $u = \tilde{u} \circ T$, on obtient le contrôleur polynomial u.

9.3.2 Problème 3 : Sortie à travers une face donnée du rectangle

On souhaite construire un contrôleur assurant la sortie de toutes les trajectoires du système (9.14) à travers une face F donnée de R_n. En effet, pour résoudre ce problème on doit bloquer toutes les faces de R différentes de F et imposer la sortie à travers la direction donné par la normale sortante à cette face.
Concrètement, si on fixe arbitrairement une face $F_{i_0,\xi_{i_0}(w_{i_0})}$ de R avec $i_0 \in \{1, \ldots, n\}$ et $w_{i_0} \in \{a_{i_0}, b_{i_0}\}$. Une solution à ce problème est donnée à l'aide de la proposition suivante :

Proposition 29. *Si il existe $\{\lambda = (\lambda_{\overline{v}}) \in \mathbb{R}^{p\times D} \mid \overline{v} \in \overline{V}'\}$ tel que :*

1. *$\sigma_i(\overline{v})\,(f_{i,\delta}(\overline{v}) + B_i\lambda_{\overline{v}}) \geq 0$ pour tout $i \in \{1, \ldots, n\}$ et tout $\overline{v} \in \overline{V'}_i$.*
2. *$(-1)^{(\xi_{i_0}(w_{i_0})+1)}(f_{i_0,\delta}(\overline{v}) + B_{i_0}\lambda_{\overline{v}}) > 0$ pour tout $\overline{v} \in \overline{V'}$.*

Alors on peut construire à l'aide du vecteur λ un contrôleur polynomial u ayant le même degré que f et assurant la sortie à travers la face $F_{i_0,\xi_{i_0}(w_{i_0})}$ de R .

Démonstration. La construction du contrôleur polynomial u est la même que celle de la Proposition 28. Il suffit de poser $F(x) = f(x) + Bu(x)$ et d'appliquer la Proposition 27 pour montrer que toutes les trajectoires correspondant à ce nouveau champ de vecteurs F quittent R par la face $F_{i_0,\xi_{i_0}(w_{i_0})}$. □

Remarque 24. *De la même façon, on peut trouver un vecteur admissible λ par résolution du programme linéaire suivant :*

$$\begin{array}{lll} \textit{maximiser} & t & \\ \textit{s.c} & t \in \mathbb{R}, \lambda \in \mathbb{R}^{p\times D} & \\ & t \leq \sigma_i(\overline{v})\,(f_{i,\delta}(\overline{v}) + B_i\lambda_{\overline{v}})\,, & i \in \{1, \ldots, n\}\backslash\{i_0\},\ \overline{v} \in \overline{V'}_i, \\ & t \leq (-1)^{(\xi_{i_0}(w_{i_0})+1)}(f_{i_0,\delta}(\overline{v}) + B_{i_0}\lambda_{\overline{v}}), & \overline{v} \in \overline{V'}. \end{array} \tag{9.16}$$

Exemple 17. *On considère le système dynamique introduit dans [130] :*

$$\begin{cases} \dot{x}_1 &= -x_2 - 1.5x_1 - 0.5x_1^3 - x_2 + u_1(x_1, x_2), \\ \dot{x}_2 &= x_1 + u_2(x_1, x_2), \end{cases} \tag{9.17}$$

Soit $R = [-2, 2] \times [-1.5, 3]$. *On se propose de trouver un contrôleur polynomial* $u(x) \in U = [-10, 10]$ *de degré* 3 *en* x_1 *et* 1 *en* x_2 *permettant pour le système (9.17) de :*

1. *Résoudre le* **Problème 2**.
2. *Résoudre le* **Problème 3** *pour une face de sortie* F.

En utilisant le programme linéaire (9.15), on obtient une valeur optimale $t^* = 8 > 0$. *Ainsi on peut construire à l'aide de* λ^* *un contrôleur polynomial* u *qui rend invariant le rectangle* $R = [-2, 2] \times [-1.5, 3.5]$ *pour le système (9.17) et donc résout bien le* **Problème 2** *comme le montre la Figure 9.7.*

Maintenant, on fixe $F = \{(x, y) \in R \,/\, y = 3\}$. *Grâce au programme linéaire*

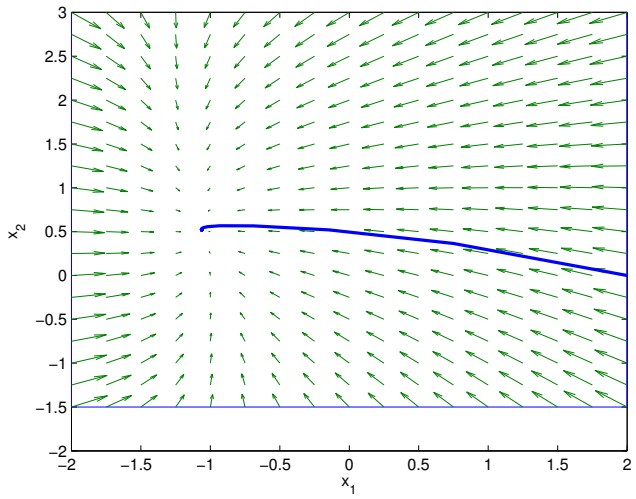

FIGURE 9.7 – Les vecteurs champ du système (9.17) pour le contrôleur polynomial u ainsi que quelques trajectoires illustrant l'invariance du rectangle R.

(9.16), on obtient $t^* = 8 > 0$. *Par conséquent le contrôleur polynomial* u *construit à l'aide de* λ^* *résout bien le* **Problème 3** *pour la face de sortie* F *comme le montre la Figure 9.8.*

9.4 Application : Planification des trajectoires

Dans cette section, on va utiliser les résultats de la section précédente pour planifier le parcours des trajectoires d'un système dynamique S.
Plus précisément, on se fixe un ensemble de rectangles R_i deux à deux adjacents avec $i \in \Gamma$ où Γ désigne un ensemble d'indices. La planification des trajectoires consiste à se fixer un rectangle initial $R_{i_1} = R_{initial}$ et un rectangle final $R_{i_2} = R_{final}$ avec

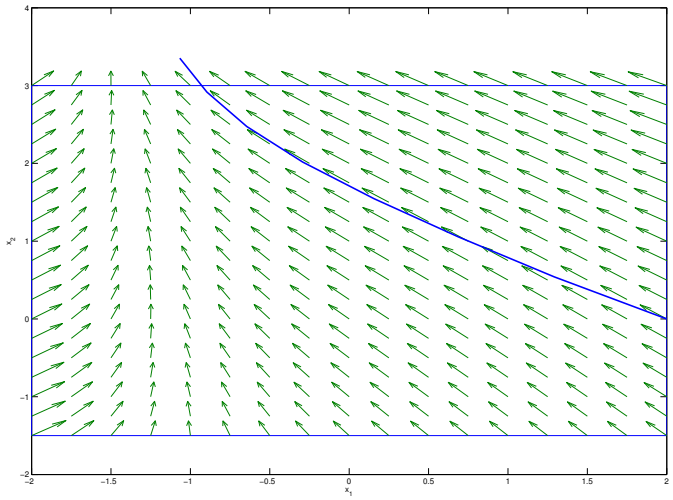

FIGURE 9.8 – Les vecteurs champ du système (9.17) ainsi qu'une trajectoires illustrant la sortie par la face F du rectangle R.

i_1 et i_2 deux indices différents de Γ et d'imposer que les trajectoires du système S commençant dans le rectangle $R_{initial}$ finissent dans le rectangle R_{final} tout en passant par les rectangles se situés entre $R_{initial}$ et R_{final}.
Soit m le nombre de rectangles que les trajectoires du système S doivent visiter et notons par R_1 le rectangle initial R_{i_1}, R_2 le rectangle adjacent à R_1 et ainsi de suite jusqu'à ce que l'on arrive à R_m désignant le rectangle final R_{i_2}. Par conséquent, pour tout $j = 1, \ldots, m-1$, il suffit de trouver un contrôleur assurant le passage de R_j à R_{j+1} : cela peut être assuré en cherchant à trouver un contrôleur qui bloque toutes les faces de R_j sauf celle adjacente à R_{j+1} et imposer la sortie par cette face. Pour $j = m$, il suffit de rester infiniment dans R_m : cela peut être assuré en cherchant à trouver un contrôleur qui bloque toutes les faces de R_m pour le rendre invariant.
Cet algorithme est illustré par l'exemple qui suit :

Exemple 18. *On reprend à nouveau l'Exemple 17 .*
Soient $R_1 = R_{initial} = [-2, -1] \times [-1.5, -0.5]$ *et* $R_8 = R_{final} = [1, 2] \times [2.5, 3]$.
On se propose de trouver des contrôleurs polynomiaux (un pour chaque rectangle) assurant que les trajectoires commençant dans R_1 *finissent dans* R_8 *en passant respectivement par les rectangles suivant :*
$R_2 = [-1, 0] \times [-1.5, -0.5]$, $R_3 = [-1, 0] \times [-0.5, 0.5]$, $R_4 = [0, 1] \times [-0.5, 0.5]$, $R_5 = [0, 1] \times [0.5, 1.5]$, $R_6 = [1, 2] \times [0.5, 1.5]$, *et* $R_7 = [1, 2] \times [1.5, 2.5]$.
En utilisant les résultats de la section précédente on arrive à trouver dans chaque rectangle R_i, $i = 1, \ldots, 7$, *un contrôleur polynomial* u_i *résolvant le* **Problème 3** *pour la face commune entre* R_i *et* R_{i+1}, *et un contrôleur polynomial* u_8 *résolvant le* **Problème 2** *pour le rectangle* R_8 *(Voir figure 9.9).*

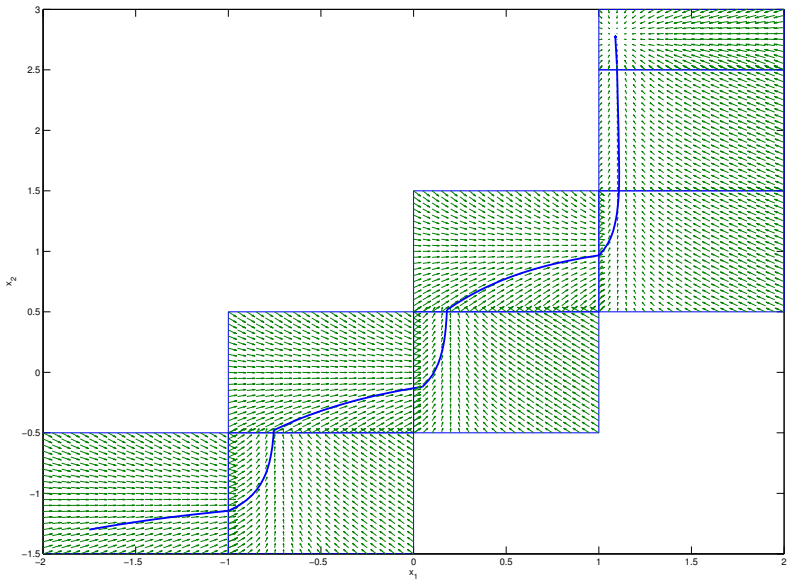

FIGURE 9.9 – Les champs de vecteurs du système (9.17) dans les rectangles $R_1, \ldots, R_8$ associés respectivement aux contrôleurs $u_1, \ldots, u_8$ ainsi qu'une trajectoire illustrant la planification.

Conclusion

Dans cette thèse on s'intéresse à une classe particulière de systèmes dynamiques non linéaires, ce sont les systèmes dynamiques polynomiaux. Une étude qualitative pour l'analyse et le contrôle de cette classe de systèmes est proposée. Pour ce faire, un autre problème aussi compliqué que celui d'origine est posé : c'est le problème de l'optimisation polynomiale qu'on note POP. En effet, notre étude qualitative des systèmes dynamiques polynomiaux peut se voir comme étant une application directe de l'optimisation polynomiale. Pour élaborer tout cela, la thèse comporte trois parties dont on rappelle pour chacune les grandes lignes.

La première partie est une partie préliminaire. Dans cette partie on a définit les notions et les résultats nécessaires. On a introduit également nos deux principaux outils pour la suite : le principe de floraison ainsi que les polynômes de Bernstein. A la fin de cette partie, on a établit le lien entre ces deux outils.

La deuxième partie est consacrée à la résolution du problème d'optimisation polynomiale. Dans un premier chapitre, concernant l'état de l'art, on a présenté les méthodes existantes dans la littérature pour pouvoir traiter ce genre de problèmes. Le deuxième chapitre constitue la contribution essentielle de cette thèse. Dans ce chapitre, on a exploité le principe de floraison pour construire une relaxation linéaire associée au problème d'optimisation polynomial. Puis, une relaxation équivalente a été formulée à l'aide des polynômes de Bernstein. Cette relaxation est ensuite améliorée grâce à l'utilisation des propriétés des polynômes de Bernstein. On a achevé cette partie par une comparaison numérique entre les différents relaxations présentées ainsi que des moyens permettant l'amélioration des résultats obtenus.

La dernière partie de la thèse est consacrée à notre problématique initiale :
Analyse et contrôle des systèmes dynamiques polynomiaux.
Pour pouvoir résoudre cette problématique on a eu besoin de traiter un ensemble de POP. Par conséquent, les chapitres de cette partie sont vus comme des applications des relaxations précédemment présentées. Des approches permettant l'étude de certaines propriétés telles que l'atteignabilité et l'invariance pour ces systèmes dynamiques ainsi que la synthèse d'un contrôleur les assurant ont été proposées. Remarquons que pour ce faire, la forme des ensembles ainsi que celle du contrôleur doit être fixée à l'avance. En effet, on s'est contenté de la forme polyédrale pour les ensembles (dont le cas particulier des rectangles a été traité à part) et de la forme polynomiale avec un degré fixé pour le contrôleur (dont aussi les cas particuliers affine et multi-affine ont été traités).

De nombreuses perspectives pourront avoir lieu comme suite logique aux travaux qui ont été traités dans cette thèse.

Dans la deuxième partie concernant l'optimisation polynomiale, on a déjà vu que grâce aux propriétés des polynômes de Bernstein (Proposition 6), un gain considérable en précision a été obtenu. Cependant, la dernière propriété (propriété 5), proposant une relation de récurrence entre les polynômes de Bernstein ayant des degrés successifs, n'a pas été utilisée. Cette propriété va permettre d'avoir des bornes additionnelles sur nos variables de décisions z_I dépendant des polynômes de Bernstein d'ordre supérieur. Par exemple, pour le cas unidimensionnel on pourra en plus des inégalités supplémentaires reliant des indices consécutifs :

$$\frac{n-i}{n} z_i + \frac{i+1}{n} z_{i+1} \leq B_{i,m-1}\left(\frac{i}{m-1}\right).$$

Ainsi, on pourra incorporer ces contraintes dans notre relaxation et analyser l'amélioration de la valeur optimale associée.

Aussi, un éventuel moyen permettant d'améliorer les bornes des relaxations introduites sera de développer un algorithme de "branch-and-bound" efficace utilisant l'une de nos deux relaxations. La difficulté sera essentiellement la stratégie de décomposition à suivre et une idée sera de bénéficier des informations que l'on pourra avoir en incrémentant l'ordre de la relaxation pour trouver une bonne direction et un bon point de décomposition.

D'autre part, un inconvénient de nos relaxations est le fait que l'on a toujours besoin de fixer un rectangle R au préalable. Dans la thèse [109], le problème de minimisation polynomiale sur le simplexe a été traité en fournissant une base de Bernstein associé au simplexe. On peut penser à généraliser cela dans le cas des polytopes et plus généralement dans le cas d'un ensemble $\mathcal{K}$ semi-algébrique. Ainsi, on pourra éviter de calculer (sur-approximer) une boîte enveloppante associée à notre ensemble $\mathcal{K}$. Dans ce cas, on peut espérer obtenir la convergence vers vrai la valeur optimale p^* (au lieu de ${p_C}^*$) si on aura une convergence vers la fonction $C_{\mathcal{K}}(p)$ [2] au lieu de converger vers la fonction $C(p) = C_R(p)$.

Dans la troisième partie, plusieurs applications ont été traitées. Des extensions à ces applications pourront avoir lieu.

Une extension du travail effectué dans le cadre de l'analyse d'atteignabilité sera de le généraliser dans le cas continu. Pour cela, il va falloir, en plus des sur-approximations des trajectoires pour des temps fixé t et $t+h$, sur-approximer le tube des trajectoires entre les instants t et $t+h$ pour tout temps t où h est un pas de discrétisation.

Dans le cadre du calcul d'invariants, le candidat possible pour l'invariance est choisi aléatoirement. Une extension sera la façon de le choisir. En effet, un bon candidat va certainement entraîner moins d'itérations dans notre algorithme (analyse de sensibilité) pour en trouver un invariant. Le choix de la forme du polytope initial

2. Enveloppe convexe de p sur l'ensemble $\mathcal{K}$

(nombre et positions des faces) doit être lié au comportement du système (stabilité, points d'équilibre,...).

Dans le dernier chapitre, une variante du problème de planification des trajectoires dans des rectangles a été traitée. Cependant, cette variante est un peu restrictive puisque l'on se fixe des rectangles adjacents deux à deux tout en imposant la sortie par la face commune jusqu'à ce que l'on atteint un rectangle final. Cependant, il sera préférable de pouvoir résoudre ce problème en ayant juste une décomposition en sous rectangles du domaine (le grand rectangle) ; c'est à dire que si l'on se fixe un rectangle initial et un rectangle final parmi ces sous rectangles, l'algorithme doit trouver des contrôleurs permettant de les joindre. Pour cela il va falloir trouver un critère assurant à chaque fois le choix de la face (où l'ensemble des faces) de sortie possible.

D'autre part, une extension intéressante du contenu de notre troisième partie sera d'étudier les applications proposées dans des domaines plus compliqués que les polytopes : les ensembles semi-algébriques. En effet, malgré que dans la partie théorique des relaxations linéaires associées à ces ensembles sont fournies, la généralisation des résultats n'est pas du tout évidente puisque la manipulation de ces ensembles est beaucoup plus complexe que les ensembles polyédraux.

Enfin, on pourra étendre nos travaux dans le cadre le plus général : les systèmes dynamiques non linéaires et se ramener au cas polynomial via une approximation. On peut aussi considérer les systèmes dynamiques hybrides tels que les systèmes à commutation.

Bibliographie

[1] A. Abate, A. Tiwari, and S. Sastry. Box invariance for biologically-inspired dynamical systems. *Automatica*, 45(7) :1601–1610, 2009.

[2] A. Alessio, M. Lazar, A. Bemporad, and W.P.M.H. Heemels. Squaring the circle : An algorithm for generating polyhedral invariant sets from ellipsoidal ones. *Automatica*, 43(12) :2096–2103, 2007.

[3] E.L. Allgower and K. George. Continuation and path following. *Acta Numerica*, 2 :1–64, 1993.

[4] E.L. Allgower and K. George. Numerical continuation methods, an introduction. *Springer series in computational mathemathic*, 13, 1993.

[5] E. Asarin, T. Dang, and A. Girard. Hybridization methods for the analysis of nonlinear systems. *Acta Informatica*, 43(7) :451–476, 2007.

[6] A. Ataei-Esfahani and Q. Wang. Nonlinear control design of a hypersonic aircraft using sum-of-squares method. In *Proceedings of the American Control Conference. New York*, pages 5278–5283, 2007.

[7] J.P. Aubin. *Viability Theory*. Birkhauser, 1991.

[8] E. Balas, S.Ceria, and G. Gornué. A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Mathemathical programming*, 58 :295–323, 1993.

[9] B.R Barmish. Stabilization of uncertain systems via linear control. *IEEE Transactions on Automatic and Control*, 28(8) :848–850, 1983.

[10] B.R Barmish. Necessary and sufficient conditions for quadratic stabilizability of an uncertain system. *Journal of Optimization theory and applications*, 46(4) :398–408, 1985.

[11] B.R Barmish, M. Coreless, and G. Leitmann. A new class of stabilizing controllers for uncertain dynamic system. *SIAM Journal on Control and Optimization*, 21(2) :246–255, 1983.

[12] M.S. Bazzra, H.D. Sherali, and C.M. Shetty. *Non linear programming : Theory and Algorithmes*. Wiley, 1993.

[13] C. Belta and L.C.G.J.M. Habets. Controlling a class of nonlinear systems on rectangles. volume 51, pages 1749–1759, 2006.

[14] M.A. Ben Sassi and A. Girard. Computation of polytopic invariants for polynomial dynamical systems using linear programming. *Automatica*, 48(12) :3114–3121, 2012.

[15] M.A. Ben Sassi and A. Girard. Controller synthesis for robust invariance of polynomial dynamical systems using linear programming. *Systems and Control Letters*, 61(4) :506–512, 2012.

[16] M.A. Ben Sassi, R. Testylier, T. Dang, and A. Girard. Reachability analysis of polynomial dynamical systems using linear programming. In *Automated Technology for Verification and Analysis, Thiruvananthapuram, India*, pages 137–151, 2012.

[17] A. Benzaouia and C. Burgat. The regulator problem for a class of linear systems with constrained control. *Systems and Control Letters*, 10(3) :357–363, 1988.

[18] A. Benzaouia and C. Burgat. Regulator problem for linear discrete-time systems with nonsymmetrical constrained control. *Systems and Control Letters*, 48(6) :2441–2451, 1988.

[19] O. Bernard and J.-L. Gouze. Global qualitative description of a class of nonlinear dynamical systems. *Artificial Intelligence*, 136 :29–59, 2002.

[20] S. Bernstein. Sur la représentation des polynômes positifs. *Soobshch Kharkov marem ob-va*, 2(14) :227–228, 1915.

[21] S. Bernstein. *Collected Works*, volume 1. USSR Academy of Sciences, 1952.

[22] S. Bernstein. *Collected Works*, volume 2. USSR Academy of Sciences, 1954.

[23] G. Bistoris. On the positive invariance of polyhedral sets for discrete-time systems. *Systems and Control Letters*, 11(3) :243–248, 1988.

[24] G. Bistoris. Positively invariant polyhedral sets of discrete-time linear systems. *International Journal of Control*, 47(6) :1713–1727, 1988.

[25] F. Blanchini. Control synthesis for discrete time systems with control and state bounds in the presence of disturbances. *JOURNAL of optimization theory and applications*, 65(1) :466–484, 1990.

[26] F. Blanchini. Constrained control for uncertain linear systems. *Journal of Optimization Theory and Applications*, 71 :466–484, 1991.

[27] F. Blanchini. Feedback control for linear time-invariant systems with state and control bounds in the presence of disturbances. *IEEE Transactions on automatic control*, 35(11) :466–484, 1991.

[28] F. Blanchini. Ultimate boundedness control for uncertain discrete-time systems via set-induced lyapunov functions. *IEEE Transactions on automatic control*, 38 :428–483, 1994.

[29] F. Blanchini. Nonquadratic lyapunov functions for robust control. *Automatica*, 31(3) :451–461, 1995.

[30] F. Blanchini. Set invariance in control. *Automatica*, 35 :1747–1777, 1999.

[31] N. K. Bose and C. C. Li. A quadratic form representation of polynomials of several variables and its applications. pages 447–448, 1968.

[32] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[33] B. Buchberger. *An algorithm for finding a basis for the residue class ring of a zero-dimensional polynomial ideal*. PhD thesis, Math.Inst, University of Innsbruck, Austria, 1965.

[34] B. Buchberger. *A note on the complexity of Gröbner bases*. 1983.

[35] B. Buchberger. *Gröbner bases and applications*, volume 251. London Mathemathical society, 1998.

[36] C.I. Byrnes and A. Isidori. New results and examples in nonlinear feedback stabilization. *Systems & Control Letters*, 12(4) :437–442, 1989.

[37] S. Cantat. Théorème de poincaré-bendixon. *Le journal des maths des élèves*, 1(3) :140–145, 1995.

[38] Y.-J. Chang and B. W. Wah. Polynomial programming using groebner bases. *Computer Software and Applications Conference, Taiwan*, pages 236–241, 1994.

[39] H.-D. Chiang and J. S. Thorp. Stability regions of nonlinear dynamical systems : A constructive methodology. *IEEE Transaction on Automatic Control*, 34(12) :1229–1241, 1989.

[40] M. Coreless and G. Leitmann. Continuous state feedback guaranteeing uniform ultimate boundedness for uncertain dynamic system. *IEEE Transactions on Automatic and Control*, 26(5) :1139–1144, 1981.

[41] P. Cousot and R. Cousot. Static determination of dynamic properties of programs. In *Proc. of the Second Int. Symp. on Programming*, pages 106–130, 1976.

[42] D.A. Cox, J.B. Little, and D. Oshea. *Using algebraic geometry.* Springer-Verlag, 1998.

[43] R.E. Curto and L.A. Fialkow. Recursiveness, positivity, and truncated moment problems. *Houston J.Math*, 17 :603–635, 1991.

[44] C. Louembet D. Henrion. Convex inner approximations of nonconvex semialgebraic sets applied to fixed-order controller design. In *Proceedings of the International Symposium on Mathematical Theory of Networks and Systems, Budapest, Hungary*, 2010.

[45] C. Savorgnan D. Henrion, J. B. Lasserre. Nonlinear optimal control synthesis via occupation measures. In *Proceedings of the IEEE Conference on Decision and Control, Cancun, Mexico*, 2008.

[46] J. B. Lasserre D. Henrion. Inner approximations for polynomial matrix inequalities and robust stability regions. *IEEE Transactions on Automatic Control*, 57(6) :1456–1467, 2012.

[47] S. Bennani D. Henrion, M. Ganet-Schoeller. Measures and lmi for space launcher robust control validation. In *Proceedings of the IFAC Symposium on Robust Control Design, Aalborg, Denmark*, 2011.

[48] T. Vyhlidal D. Henrion. Positive trigonometric polynomials for strong stability of difference equations. *Automatica*, 48(9) :2207–2212, 2012.

[49] T. Dang. Approximate reachability computation for polynomial systems. In *HSCC'06*, volume 3927 of *LNCS*, pages 138–152. Springer, 2006.

[50] T. Dang, C. Le Guernic, and O. Maler. Computing reachable states for nonlinear biological models. In *CMSB'09*, volume 5688 of *LNCS*, pages 126–141. Springer, 2009.

[51] T. Dang, C. Le Guernic, and O. Maler. Computing reachable states for nonlinear biological models. In *Computational Methods in Systems Biology*, volume 5688/2009 of *LNCS*, pages 126–141. Springer, 2009.

[52] T. Dang, O. Maler, and R. Testylier. Accurate hybridization of nonlinear systems. In *HSCC'10*, pages 11–20, 2010.

[53] T. Dang and D. Salinas. Image computation for polynomial dynamical systems using the Bernstein expansion. In *Computer Aided Verification*, volume 5643/2009 of *LNCS*, pages 219–232. Springer, 2009.

[54] G.B. Dantzig. *Linear programming and extensions*. Princeton University Press, 1963.

[55] E. J. Davison and E. M. Kurak. A computational method for determining quadratic lyapunov functions for nonlinear systems. *Automatica*, 7 :627–636, 1971.

[56] P. de Casteljau. Outillages méthodes calcul. *Technical report, Andre Citroen*, 1959.

[57] P. de Casteljau. Formes à pôles. *Hermes*, 1985.

[58] E.S. Dimitrova. *Polynomial Models for Systems Biology : Data Discretization and Term Order Effect on Dynamics*. PhD thesis, Polytechnic Institute and State University, The faculty of the Virginia, 2006.

[59] P. Dorato. Qunatified multivariate polynomial inequalities. *The mathemathics of pratical control d problems. IEEE control system magazine*, 20(5) :48–58, 2000.

[60] P. Dreesen and B.D. Moor. Polynomial optimization are eignvalue problems. *Chapter 4 of Model-Based Control :Bridging Rigorous Theory and Advanced Technology*, pages 49–68.

[61] G.T. Cargo et O. Shisha. The bernstein form of a polynomial. *Journal of Research of NBS*, 70(1) :79–81, 1965.

[62] P.L. Falb and W.A. Wolovich. Decoupling in the design and synthesis of multivariable control systems. *IEEE Transactions on Automatic Control*, 12(6) :651–659, 1967.

[63] A. Feuer and M. Heymann. Admissible sets in linear feedback systems with bounded controls. *Int.J.Control*, 23(3) :381–392, 1976.

[64] A. Feuer and M. Heymann. Omega-invariance in control systems with bounded controls. *Journal of Mathematical Analysis and Applications*, 53 :266–276, 1976.

[65] H.C. Fisher. Range computations and applications. *BIT*, 21 :112–117, 1981.

[66] R. FitzHugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical J.*, 1 :445–466, 1961.

[67] Goran Frehse, Bruce H. Krogh, and Rob A. Rutenbar. Verifying analog oscillator circuits using forward/backward abstraction refinement. In *DATE'06*, 2006.

[68] A. Tesi G. Chesi, A. Garulli and A. Vicino. Lmi-based computation of optimal quadratic lyapunov functions for odd polynomial systems. *Int. J. Robust Nonlinear Control*, 15 :35–49, 2005.

[69] J. Garloff. Convergent bounds for the range of multivariate polynomials. *Lecture notes in Computer Science*, 212 :37–56, 1986.

[70] J. Garloff. The bernstein algorithm. *Reliable Computing*, 2 :154–168, 1993.

[71] J. Garloff. Application of bernstein expansion to the solution of control problems. *Reliable computing*, 6(3) :303–320, 2000.

[72] J. Garloff. The bernstein expansion and its applications. *J.Am. Roumanian Acad*, 25(27) :80–85, 2003.

[73] J. Garloff and B. Graf. Solving polynomial inequalities by bernstein expansion. *Symbolic Methods in Control System Analysis and Design. IEEE Control Engineering*, 56 :339–352, 1999.

[74] J. Garloff, C. Jansson, and P. Smith. Lower bound functions for polynomials. *Journal of computational and applied Mathemathics*, 57(1) :207–225, 2003.

[75] J. Garloff and P. Smith. Solutions of systems of polynomial equations by using bernstein expansion. *Symbolic and algebraic methods and verification methods*, pages 87–97, 2001.

[76] K. Gatermann and B. Huber. A family of sparse polynomial systems arising in chemical reaction systems. *J. Symbolic Computation*, 33 :275–305, 2002.

[77] E.G. Gilbert. Controllability and observability in multivariable control systems. *J.S.I.A.M control*, 2(1) :128–151, 1963.

[78] A. Girard, C. Le Guernic, and O. Maler. Efficient computation of reachable sets of linear time-invariant systems with inputs. In *HSCC'06*, volume 3927 of *LNCS*, pages 257–271. Springer, 2006.

[79] M.R. Greenstreet and I. Mitchell. Reachability analysis using polygonal projections. In *HSCC'99*, volume 1569 of *LNCS*, pages 103–116. Springer, 1999.

[80] S. Gutman and G. Leitmann. Stabilizing control for linear systems with bounded parameter and input uncertainty. In *7th IFIP Conference on Optimization Techniques, Berlin, Germany*. Springer-Verlag, 1975.

[81] P.O Gutmann and M. Cwikel. Admissible sets and feedback control for discrete-time linear dynamical systems with bounded controls and states. *IEEE Transactions on Automatic and Control*, 31(4) :373–377, 1986.

[82] P.O Gutmann and P. Hagander. A new design of constrained controllers for linear systems. *IEEE TRANSACTIONS ON AUTOMATIC CONTROL*, 30(1) :22–33, 1985.

[83] L.C.G.J.M Habets, P.J.Collins, and J.H. Van Schuppen. Reachability and control synthesis for piecewise-affine hybrid system on simplices. volume 51, pages 938–948, 2006.

[84] K. Hagglöf, P.O. Lindberg, and L. Svensson. Computing global minima to polynomial optimization problems using gröbner bases. *Journal of global optimization*, 7 :115–125, 1995.

[85] A. Halasz, V. Kumar, M. Imielinski, C. Belta, O. Sokolsky, S. Pathak, and H. Rubin. Analysis of lactose metabolism in e.coli using reachability analysis of hybrid systems. *IET Systems Biology*, 1(2) :130–148, 2007.

[86] Zhi Han and Bruce H. Krogh. Reachability analysis of large-scale affine systems using low-dimensional polytopes. In *HSCC'06*, volume 3927 of *LNCS*, pages 287–301. Springer, 2006.

[87] D. Handelmann. Representing polynomials by positive linear function on comapct convex poyhedra. *Pacific J.Math. Ann*, 132 :35–62, 1988.

[88] J. Hauser and M. C. Lai. Estimating quadratic stability domains by nonsmooth optimization. In *Proc. American Control Conf., Chicago*, page 571–576, 1992.

[89] R. Horst and H. Tuy. Global optimization : Deterministic approaches. *Springer-Verlag 2nd edition*, 1993.

[90] H.Samelson. On the brouwer fixed point theorem. *Portugal.Math*, 22 :189–191, 1963.

[91] Z. W. Jarvis-Wloszek. *Lyapunov Based Analysis and Controller Synthesis for Polynomial Systems using Sum-of-Squares Optimization*. PhD thesis, UC Berkeley, 2003.

[92] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis*. Springer, 2001.

[93] J.H.Hubbard. *Differential equations : a dynamical systems approach*, volume 18. Springer-Verlag New York,Inc.

[94] G.F. Jonsson and S.A. Vavasis. Accurate solution of polynomial system macaulay resultant matrices. *Math. Comput*, 74(249) :221–262, 2004.

[95] R.E. Kalman. On the general theory of control systems. In *Proc. First International Congress of Automatic Control, Moscow*, 1960.

[96] S. Kaynama, J. Maidens, M. Oishi, I. Mitchell, and G.A. Dumont. Computing the viability kernel using maximal reachable set. In *HSCC'12*, 2012.

[97] M. Kloetzer and C. Belta. Reachability analysis of multi-affine systems. In *HSCC'06*, volume 3927 of *LNCS*, pages 348–362. Springer, 2006.

[98] M. Krstić, I. Kanellakopoulos, and P. Kokotović. *Nonlinear and Adaptive Control Design*. Wiley, 1995.

[99] J.M. Lane and R.F. Riesenfeld. Bounds on a polynomial. *BIT*, 21 :112–117, 1981.

[100] J.B. Lasserre. Linear programming with semi-definite matrices. *Technical report LAAS-94099, Laboratoire d'Analyse et d'Architecture des systèmes du CNRS*, 1995.

[101] J.B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM J. Optimization*, 11(3) :796–817, 2001.

[102] J.B. Lasserre. Semidefinite programming vs lp relaxations for polynomial programming. *Mathematics of operations research*, 27 No 2 :347–360, 2002.

[103] J.B. Lasserre. Polynomial programming : Lp-relaxations also converge. *SIAM J. Optimization*, 15(2) :383–393, 2004.

[104] J.B. Lasserre. *Moments, positive polynomials and their applications*, volume 1. World scientific publishing, 2010.

[105] J.B. Lasserre, D. Henrion, C. Prieur, and E. Trélat. Nonlinear optimal control via occupation measures and lmi relaxations. *SIAM J. Control Opt.*, 47(4) :1643–1666, 2008.

[106] D. Lazard. Résolution des systèmes d'équations algébriques. *Theor.Comput.Sci*, 15 :77–110, 1981.

[107] C. Le Guernic and A. Girard. Reachability analysis of hybrid systems using support functions. In *CAV'09*, volume 5643 of *LNCS*, pages 540–554. Springer, 2009.

[108] G. Leitmann. Guaranteed asymptotic stability for a class of uncertain linear dynamical systems. *Journal of Optimization theory and applications*, 27(1) :212–216, 1979.

[109] R. Leroy. *Certificats de positivité et minimisation polynomiale dans la base de Bernstein multivariée*. PhD thesis, Mathémathiques et applications, Université de Rennes 1, 2008.

[110] T.Y. Li. Numérical solution of multivariate polynomial systems by homotopy continuation method. *Acta Numerica*, 6 :399–436, 1997.

[111] Q. Lin and J. Rokne. Interval approximation of higher order to the ranges of functions. *Computers Math. Applic*, 31(7) :101–109, 1996.

[112] A.J. Lotka. Contribution to the theory of periodic reaction. *J. Phys. Chem*, 14(3), 1910.

[113] A. M. Lyapunov. *The General Problem of the Stability of Motion*. Kharkov Math. Soc , Kharkov, Russia, 1892.

[114] D. Henrion M. Ait Rami. A hierarchy of lmi inner approximations of the set of stable polynomials. *Automatica*, 47(7) :1455–1460, 2010.

[115] D. Henrion J. Zikmund M. Anderle, S. Celikovsky. Advanced lmi based analysis and design for acrobot walking. *International Journal of Control*, 83(8) :1641–1652, 2010.

[116] D. Henrion M. Mevissen, J. B. Lasserre. Moment and sdp relaxation techniques for smooth approximations of problems involving nonlinear differential equations. In *Proceedings of the IFAC World Congress on Automatic Control, Milan, Italy*, 2010.

[117] A. Shiu C. Conradi M. P. Millan, A. Dickenstein. Chemical reaction systems with toric steady states. *arXiv :1102.1590*, 2011.

[118] M.E. Magana and S.H. Zak. Robust output feedback stabilization of discrete time uncertain dynamical systems. *IEEE Transactions Automatic Control*, 33 :1082–1085, 1988.

[119] D. Manocha. Solving system of polynomial equations. volume 14, pages 46–55, 1994.

[120] R. Martin, H. Shou, I. Voiculescu, A. Bowyer, and G. Wang. Comparison of interval methods for plotting algebraic curves. *Computer Aided Geometric Design*, (19) :553–587, 2002.

[121] I. Mitchell and C. Tomlin. Level set methods for computation in hybrid systems. In *HSCC'00*, volume 1790 of *LNCS*, pages 310–323. Springer, 2000.

[122] A. Morgan. *Solving Polynomial Systems Using Continuation for Engineering an Scientific*. Prentice-Hall, 1987.

[123] B.S. Morgan. The synthesis of linear multivariable systems by state variable feedback. *IEEE Transactions Automatic Control*, 9 :405–411, 1964.

[124] H.M. Möller and H.J. Stetter. Multivariate polynomial equations with multiple zeros solved by matrix eigenproblems. *Numerishe Mathemathike*, 70 :311–329, 1995.

[125] P.S.V. Nataraj and M. Arounassalame. A new subdivison algorithm for the bernstein polynomial approach to global optimization. *International journal of automation and computing*, 4 :342–352, 2007.

[126] P.S.V. Nataraj and M. Arounassalame. Constrained global optimization of multivaraite polynomials using bernstein branch and prune algorithm. *Global optimization*, 49(2), 2009.

[127] N.S. Nedialkov, K.R. Jackson, and G.F. Corliss. Validated solutions of initial value problems for ordinary differential equations. *Applied Mathematics and Computation*, (105) :21–68, 1999.

[128] E. Noldus. Effectiveness of state constraints in controllability and least squares. *IEEE Trans. Automatic. Contr*, 22(4), 2009.

[129] N.Ramdani, N.Meslem, and Y.Candau. A hybrid bounding method for computing an over-approximation for the reachable space of uncertain nonlinear systems. *IEEE Transactions on Automatic Control*, 54(10) :2352–2364, 2009.

[130] N. Ozay, J. Liu, P. Prabhakar, and R. Murray. Computing augmented finite transition system to synthesize switching protocols for polynomial switched systems. In *American Control Conference*.

[131] P.A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California institute of technology, Pasadena, CA, 2000.

[132] P.A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming Ser. B*, 96(2) :293–320, 2003.

[133] G. Pataki. Cone-lp's and semi-definite programs :facial structure, basic solutions and the simplexe method. *Technical report, GSIA, Carnegie University, Pittsburgh, PA*, 1995.

[134] A. Platzer and E. M. Clarke. The image computation problem in hybrid systems model checking. In *HSCC'07*, volume 4416 of *LNCS*, pages 473–486. Springer, 2007.

[135] Balth. Van Der Pol and J. Van Der Marka. The heartbeat considered as a relaxation oscillation and an electrical model of the heart. 6, 1928.

[136] A. Polanski. On absolute stability analysis by polyhedral lyapunov functions. *Automatica*, 36(4) :573–578, 2000.

[137] V. Powers and B. Reznick. Summationsmethoden und momentfolgen i. *Math. Zeit*, 9 :74–109, 1921.

[138] V. Powers and B. Reznick. Polynomials that are positive on an interval. *Trans. Amer. Maths. Soc*, 352 :4677–4692, 2000.

[139] S. Prajna, A. Jadbabaie, and G. J. Pappas. A framework for worst-case and stochastic safety verification using barrier certificates. *IEEE Transactions on Automatic Control*, 52(8) :1415–1429, 2007.

[140] S. Prajna, P.A. Parrilo, and A. Rantzer. Nonlinear control synthesis by convex optimization. *IEEE Trans. on Autom. Control*, 49(2) :1–5, 2004.

[141] M. Putinar. Positive polynomials on comapct semi-algebraic sets. *Indiana Univ. Math*, 42 :969–984, 1993.

[142] M. Tartaglia R. Genesio and A. Vicino. On the estimation of asymptotic stability regions : State of the art and new proposals. *IEEE Transaction on Automatic Control*, 30(8) :747–755, 1985.

[143] L. Ramshaw. Blossoming : A connect the dots approach to splines. *Systems resarch center*, 1987.

[144] L. Ramshaw. Blossoms are polar forms. *Computer Aided Geometric Design*, 6 :323–358, 1989.

[145] A. Rantzer. A dual to lyapunov's stability theorem. *Systems and Control Letters*, 42 :161–168, 2001.

[146] S. Ray and P.S.V. Nataraj. An efficient algorithm for range computation of polynomials using the bernstein form. *Global optimization*, 45 :403–426, 2009.

[147] S. Ray and P.S.V. Nataraj. A new strategy for selecting subdivison point in the bernstein approach to polynomials optimization. *Reliable computing*, 14 :117–137, 2010.

[148] Z. V. Rekasius. Decoupling of multivariable systems by means of state variable feedback. In *Proc. 3rd -4nn. Allerton Conf on Circuit and System Theory, Urbana*, pages 439–447, 1965.

[149] B. Reznick. Some concrete aspects of hilbert's 17th problem. *Contemporay Mathemathics*, 253 :251–272, 2000.

[150] T.J. Rivlin. Bounds on a polynomial. *J. Res. Nat. Bur. Standards Sect. B*, 74B :47–54, 1970.

[151] R. Rockfellar. *Convex analysis*. Princeton landamarks in mathematics.

[152] J. Rokne. Bounds for an interval polynomial. *Computing Math*, 18(3) :225–240, 1977.

[153] J. Rokne. A note on the bernstein algorithm for bounds for interval polynomials. *Computing Math*, 21 :159–170, 1979.

[154] J. Rokne. Optimal computation of the bernstein algorithm for the bound of an interval polynomial. *Computing Math*, 28 :239–246, 1982.

[155] J. Rokne. The range of values of a complex polynomial over a compplex interval. *Computing Math*, 22 :153–169, 1999.

[156] B. Roszak and M. Broocke. Necessary and sufficient condition for reachability on a simplex. *Automatica*, 2007.

[157] S. Sankaranarayanan. Automatic invariant generation for hybrid systems using ideal fixed points. In *Hybrid Systems : Computation and Control*, pages 221–230, 2010.

[158] S. Sankaranarayanan, T. Dang, and F. Ivancic. Symbolic model checking of hybrid systems using template polyhedra. In *TACAS'08*, volume 4963 of *LNCS*, pages 188–202. Springer, 2008.

[159] S. Sankaranarayanan, T. Dang, and F. Ivančić. Symbolic model checking of hybrid systems using template polyhedra. In *Tools and algorithms for the construction and analysis of systems*, pages 188–202. Springer-Verlag, 2008.

[160] K. Schmudgen's. The k-moment problem for compact semi-algebraic sets. *Math. Ann*, 289 :203–206, 1991.

[161] H. Seidel. An introduction to polar forms. volume 13, pages 38–46, 1993.

[162] H.D. Sherali and C.H. Tuncbilek. A global optimization algorithm for polynomial programming using a reformulation-linearization technique. *Journal of Global Optimization*, 2 :101–112, 1991.

[163] H.D. Sherali and C.H. Tuncbilek. New reformulation-linearization/convexification relaxations for univariate and multivariate polynomial programming problems. *Operation Research Letters*, 21 :1,9, 1997.

[164] N.Z. Shor. Cut-off method with space extension in convex programming problems. *Cybermetics*, 13, 1977.

[165] S.Lang. *Differential and riemann manifolds*. Springer-Verlag New York,Inc.

[166] V. Stahl. *Interval Methods for Bounding the Range of Polynomials and Solving Systems of Nonlinear Equations*. PhD thesis, Research Institute for Symbolic Computation, Johannes Kepler University, A-4040 Linz, Austria, 1996.

[167] A. Steinberg and M. Corless. Output feedback stabilization of uncertain dynamical systems. *IEEE Transactions Automatic Control*, 30 :1025–1027, 1985.

[168] H.J. Stetter. Matrix eigenproblems are the heart of polynomial system solving. *ACM SIGSAM Bull*, 30(2) :22–25, 1996.

[169] H.J. Stetter. *Numerical polynomial algebra*. Siam, 2004.

[170] W. Tan and A. Packard. Stability region analysis using sum of squares programming. In *Proc. American Control Conf., Minneapolis*, page 2297–2302, 2006.

[171] W. Tan and A. Packard. Stability region analysis using polynomial and composite polynomial lyapunov functions and sum-of-squares programming. *Automatic Control, IEEE Transactions*, 53 :565–571, 2008.

[172] A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, 1948.

[173] B. Tibken and Y. Fan. Computing the domain of attraction for polynomial systems via bmi optimization methods. In *Proc. American Control Conf., Minneapolis*, page 117–122, 2006.

[174] A. Tiwari and G. Khanna. Nonlinear systems : Approximating reach sets. In *HSCC'04*, volume 2993 of *LNCS*, pages 600–614. Springer, 2004.

[175] U. Topcu and A. Packard. Local stability analysis for uncertain non linear systems. *Automatic Control, IEEE Transactions*, 54 :1042–1047, 2009.

[176] P. Seiler U. Topcu, A. Packard and T. Wheeler. Stability region analysis using simualtions and sum-of-squares programming. In *Proc. American Control Conf., New York*, 2007.

[177] R.J Vanderbei. *Linear programming : Foundations and extensions*, volume 114. Springer, 2008.

[178] L. Vanderberghe. Semidefinite programming. *Siam review*, 38(1) :49–95, 1996.

[179] A. Vannelli and M. Vidyasagar. Maximal lyapunov functions and domains of attraction for autonomous nonlinear systems. *Automatica*, 21(1) :69–80, 1985.

[180] F.H. Vasilescu. Spectral measures and moment problems. *Spectral theory and its applications*, Theta foundation :173–215, 2003.

[181] M. Vassilaki, J.C. Hennet, and G. Bitsoris. Feedback control of linear discrete-time systems under state and control constraints. *International Journal of control*, 47 :1727–1735, 1988.

[182] J. Verschelde. The phc pack, the database of polynomial systems. *MSRI Berkeley Preprint*, 1999.

[183] J. Verschelde. Polynomial homotopies for dense, sparse and determinantal systems. *Technical report, University of Illinois, Mathematics Departement, Chicago*, 2001.

[184] V. Visweswaran and C.A. Floudas. Unconstrained and constrained global optimization of polynomial functions in one variable. *J.Global Optim*, 2 :73,99, 1992.

[185] V. Volterra. Fluctuations in the abundance of a species considered mathematically. *Nature*, (118) :558–560, 1926.

[186] T.-C. Wang, S. Lall, and M. West. Polynomial level-set methods for nonlinear dynamical systems analysis. In *Allerton Conference on Communication, Control and Computing*, page 640–649, 2005.

[187] S. Wiggins. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*, volume 2. Springer-Verlag New York,Inc.

[188] W.A Wolovich. Linear multivariable systems. 11, 1974.

[189] W.M Wonham. *Linear multivariable control*, volume 101. Springer-Verlag New York, 1974.

[190] Chao Yan and Mark R. Greenstreet. Circuit level verification of a high-speed toggle. In *FMCAD'07*, pages 199–206, 2007.

[191] B. Yordanov and C. Belta. A formal verification approach to the design of synthetic gene networks. In *CDC*, pages 199–206, 2011.

[192] D.B. Yudin and A.S. Nemirovsky. Informational complexity and efficient methods for solving complex external methods. *Matekon*, 13 :25–45, 1977.

[193] K. Zhou and P.P. Khargonekar. On the stabilization of uncertain linear via bound invariant lyapunov functions. *SIAM Journal on Control and Optimization*, 26(6) :1431–1441, 1988.