



Intégration d'une contrainte logique dans les problèmes de contrôle optimal et résolution par la programmation mixte

Dorin Preda

► To cite this version:

Dorin Preda. Intégration d'une contrainte logique dans les problèmes de contrôle optimal et résolution par la programmation mixte. Modélisation et simulation. Institut National Polytechnique de Toulouse - INPT, 2004. Français. NNT : . tel-00008881v2

HAL Id: tel-00008881

<https://theses.hal.science/tel-00008881v2>

Submitted on 7 Apr 2005

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

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

THÈSE

présentée pour obtenir le titre de

**DOCTEUR DE L'INSTITUT NATIONAL
POLYTECHNIQUE DE TOULOUSE**

École Doctorale : Informatique & Télécommunications
Spécialité : Mathématiques Appliquées

par

Dorin PREDA

**INTÉGRATION D'UNE CONTRAINTE LOGIQUE
DANS LES PROBLÈMES DE CONTRÔLE OPTIMAL
ET RÉOLUTION PAR LA PROGRAMMATION MIXTE**

Soutenue publiquement le 14 Décembre 2004 devant le jury composé de :

MM.	P. Hansen Ph. Toint	<i>Rapporteurs</i>
MM.	J. Bernussou R. Epenoy P. Legendre	<i>Examineurs</i>
MM.	J.-B. Caillau J. Gergaud	<i>Invités</i>
M.	J. Noailles	<i>Directeur de thèse</i>

Table des matières

Introduction	v
1 Position du problème	1
1.1 Problème de contrôle optimal avec contraintes logiques	1
1.2 Discrétisation	4
1.3 Contraintes logiques	7
1.4 Problèmes mixtes	10
1.5 Problème en variables mixtes associé à une dynamique linéaire	11
1.6 Problème en variables mixtes associés à une dynamique non- linéaire. Transfert orbital	12
2 Contraintes logiques	15
2.1 Espace de recherche. Contraintes (C_Δ)	15
2.2 Notion de solution saturée	16
2.3 Réduction de l'espace de recherche	18
2.4 Élimination d'une solution déjà analysée	21
2.5 Matrice totalement unimodulaire	22
2.6 Conclusions	23
3 MINLP associé à une dynamique linéaire	25
3.1 Rappels relatifs à la programmation mixte	26
3.2 Modèle MIQP . Algorithmes et résultats	30
3.3 Modèle MINLP . Algorithme et résultats	36
3.4 Exemples	44
3.5 Conclusions	51
4 Optimisation globale et transfert orbital	53
4.1 La démarche Branch and Reduce	53
4.2 Transfert orbital 2D à masse constante	65
4.3 Convexification	68
4.4 La réduction du domaine	98
4.5 Résultats	100
4.6 Conclusions	103

Conclusion	105
Bibliographie	107

Introduction

Contexte. Dans le cadre de cette étude on se propose d'intégrer un type particulier de contrainte logique dans les problèmes de contrôle optimal. Plus précisément on considère un système dont l'évolution au cours du temps est décrite par des équations différentielles et qu'on veut ramener d'un état initial donné en un état final donné, dans un temps fixé, en minimisant l'énergie dépensée le long du processus. Ce type de problème est un cas important de problème de contrôle optimal et différentes approches pour le résoudre existent. La question à laquelle on essaye de répondre dans cette étude est de trouver la solution optimale du problème obtenu à partir du précédent en ajoutant une contrainte logique limitant la durée de contrôle. Ce dernier type de problème s'inscrit dans un cadre plus général, celui de l'optimisation globale. Plus spécifiquement encore, on modélise notre système comme un problème **MINLP** (Mixed Integer Non Linear Problem), problème en variables mixtes, contenant à la fois des variables continues et des variables contraintes à prendre des valeurs entières. Il s'agit là de la classe la plus générale des problèmes d'optimisation et deux facteurs rendent ce type de problème très difficile à résoudre : d'une part il n'existe pas de conditions nécessaires et suffisantes d'optimalité, ce qui oblige les algorithmes à trouver d'autres méthodes pour vérifier l'optimalité d'une solution; d'autre part, cette classe de problèmes est NP-difficile, ce qui signifie que la solution ne peut pas être trouvée en temps polynomial. Dans ces conditions, les algorithmes existants ne sont pas généraux, se limitant souvent à traiter des cas particuliers du domaine mixte (contraintes linéaires, problème convexes, etc). C'est la démarche qu'on va suivre nous aussi tout au long de ce travail consistant à adapter et prendre en compte au mieux la forme particulière du problème à résoudre, d'une part, et les spécificités des contraintes logiques, d'autre part. Quant aux problèmes de contrôle optimal traités, deux cas représentatifs sont analysés: le problème LQR (objectif quadratique séparable et dynamique linéaire par rapport aux variables continues) et le cas du transfert orbital 2D à masse constante (exemple significatif d'une dynamique fortement non linéaire).

Organisation du document. Ce manuscrit est divisé en quatre chapitres. Le premier chapitre a pour but de poser le problème, en partant

d'un problème de contrôle optimal et en exprimant les contraintes logiques qui nous intéressent. On donne ici les concepts utilisés pour obtenir une modélisation mixte: la discrétisation par collocation directe et les différentes façon qui s'offrent à nous pour intégrer les contraintes logiques. On propose aussi la formulation générale du problème dans deux cas intéressants qui méritent d'être distingués et analysés de façon indépendante: le cas où le système de départ a une dynamique linéaire et celui d'un système gouverné par une dynamique non-linéaire (illustré sur le problème du transfert orbital 2D à masse constante).

Le deuxième chapitre s'intéresse aux contraintes logiques; il s'agit d'une étude pouvant s'appliquer aux deux cas (dynamique linéaire et dynamique non linéaire). L'idée directrice du chapitre est de réduire l'espace de recherche induit par les contraintes logiques, mais aussi de caractériser au mieux les solutions parmi lesquelles se trouve l'optimum global. Plusieurs propriétés intéressantes sont énoncées et démontrées dans cette partie et leur rôle dans l'accélération d'un algorithme mixte est mis en valeur.

Le troisième chapitre s'intéresse de plus près au modèle quadratique; il faut préciser qu'il s'agit du cas où le problème de départ (avant l'intégration des contraintes logiques) est quadratique, c'est à dire gouverné par des équations différentielles linéaires et un objectif quadratique. On présente ici les algorithmes du domaine mixte et leur adaptation pour les deux modèles mixtes décrits auparavant. La dernière partie du chapitre est consacrée à l'analyse des différents résultats numériques obtenus en appliquant les algorithmes existants, mais aussi celui qu'on a développé nous-mêmes. Le cas de l'oscillateur harmonique est analysé plus en détail, comme exemple représentatif de la classe des problèmes à dynamique linéaire intégrant les contraintes logiques.

Au cœur du quatrième chapitre se trouve le cas non linéaire, le plus difficile à traiter; après avoir présenté la méthode qu'on utilise sur un exemple plus simple ayant un nombre petit de termes non convexes, on s'intéresse au cas du transfert orbital 2D à masse constante. Le cadre général étant celui de l'algorithme de Branch and Bound, le principal apport de ce chapitre étant dans les différentes façons d'obtenir une relaxation convexe du problème non convexe. En effet, la relaxation convexe joue un rôle très important dans l'algorithme de Branch and Bound/Reduce, en fournissant un minorant de l'optimum global du problème non convexe. Un autre apport important constitue l'intégration de l'analyse d'intervalle dans l'algorithme afin de réduire le domaine de recherche. On présente les différents résultats numériques obtenus et on conclut ce chapitre par une discussion concernant les limitations de la méthode proposée.

Motivation. Les contraintes logiques se modélisent naturellement en utilisant les variables booléennes, ce qui rend le problème de départ mixte

(variables continues et entières). C'est de cette façon qu'on a été amené à s'intéresser aux problèmes **MINLP**. D'autre part, la contrainte logique qu'on considère apparaît souvent dans le cas d'un engin dont le moteur est soumis à des limitations d'utilisation; ce sont les contraintes cumulatives sur la durée du contrôle. Une autre application possible est la prise en compte des éclipses dans les problèmes de transfert orbital. Outre l'intérêt pratique, les parties théorique et algorithmique représentent une motivation en soi: en l'absence de conditions nécessaires et suffisantes d'optimalité, trouver l'optimum global relève beaucoup de l'étude préalable afin de caractériser et d'exploiter la structure particulière du problème; prouver l'optimalité d'une solution trouvée devient un réel défi algorithmique. Ce n'est pas un travail facile, car l'implantation des algorithmes présentés ici est souvent lourde et demande la création de plusieurs composants logiciels élémentaires (comme l'implantation des méthodes directes pour résoudre le problème de transfert orbital 2D à masse constante en utilisant la collocation directe ou les différents problèmes primal et master dans les algorithmes de Décomposition de Benders et d'Approximation Extérieure). Le cas échéant, des bibliothèques d'optimisation ont été utilisées: ILOG CPLEX et IBM OSL (pour les excellents algorithmes MILP, MIQP et QP) et KNITRO (pour l'efficacité de son solveur non linéaire).

Collaborations. Cette étude s'inscrit dans le cadre d'une collaboration de longue date entre la Division Mathématiques Spatiales du CNES¹ de Toulouse (Département Mathématiques Appliquées et Analyse Numérique) et le LIMA², composante ENSEEIHT³ de l'IRIT⁴, Unité Mixte de Recherche CNRS 5505. Les rapports cités dans les références témoignent de cette relation contractuelle (contrats 871/94/CNES/1454 et 86/776/98/CNES/7462, 02/CNES/0257/00-DPI 500). Le Ministère de l'Éducation, de l'Enseignement Supérieur et de la Recherche est à l'origine du financement de ce travail (bourse MENRT no. 25INP00), qui s'est effectué au sein de l'équipe Algorithmes Parallèles et Optimisation du LIMA, rattachée à l'activité Informatique Numérique de l'IRIT.

1. Centre National d'Études Spatiales

2. Laboratoire d'Informatique et de Mathématiques Appliquées

3. École Nationale Supérieure d'Électronique, d'Électrotechnique, d'Informatique, d'Hydraulique et de Télécommunications

4. Institut de Recherche en Informatique de Toulouse

Chapitre 1

Position du problème

Ce chapitre a pour but d'introduire le problème qu'on se propose d'étudier au cours de cet exposé. On commence par formuler un problème de contrôle optimal et exprimer la contrainte logique dont on donne une motivation concrète. On fait le premier pas vers la modélisation mathématique du problème, en présentant la méthode de discrétisation choisie, celle de la collocation directe. Une première formulation (assez grossière) du problème nous permettra de mettre en évidence les difficultés et d'introduire les méthodes numériques susceptibles de s'appliquer avec succès pour résoudre cette formulation. On conclut ce chapitre en formulant les deux types de problèmes auxquels on va s'intéresser: les problèmes MINLP issus d'une dynamique linéaire et les problèmes MINLP issus d'une dynamique fortement non-linéaire.

1.1 Problème de contrôle optimal avec contraintes logiques

Considérons un système (1.1) dont l'évolution au cours du temps est donnée par une équation différentielle ordinaire, système se trouvant en x^0 au moment $t = t_0 = 0$ et qu'on veut ramener en x^f à $t = t_f$. Les états initial et final, ainsi que le temps final t_f sont connus; on veut minimiser la dépense d'énergie. Ce problème d'optimisation s'écrit:

$$\left\{ \begin{array}{ll} \min_{x(t), u(t)} & \frac{1}{2} \int_0^{t_f} |u(t)|^2 dt \\ \text{s.t.} & \dot{x}(t) = F(t, x(t), u(t)) \quad t \in [0, t_f] \\ & x(0) = x^0 \\ & x(t_f) = x^f \\ & |u(t)| \leq U_{max} \quad t \in [0, t_f] \end{array} \right. \quad (1.1)$$

où:

- $x(t)$ est l'état à l'instant t , $x(t) \in \mathbb{R}^n$;

- $u(t)$ est le contrôle à l'instant t , $u(t) \in \mathbb{R}^m$;
- $F : \mathbb{R}^{1+n+m} \mapsto \mathbb{R}^n$ est un vecteur des fonctions décrivant l'évolution du système au cours du temps;
- $x^0 \in \mathbb{R}^n$ et $x^f \in \mathbb{R}^n$ sont les états initial et final du système, supposés connus;
- U_{max} est la norme maximale développée par le moteur à tout instant $t \in [0, t_f]$, constante qu'on va souvent désigner par *poussée maximale*.

Le problème (1.1) est un problème de contrôle optimal, sa solution optimale étant le couple $(\bar{x}(t), \bar{u}(t))$ qui minimise l'objectif quadratique: $\int_0^{t_f} u^2(t) dt$. L'équation

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

va être désignée tout au long de ce manuscrit comme la *dynamique* du système. Ainsi, on parlera d'une *dynamique linéaire* quand la fonction F est linéaire par rapport aux $x(t)$ et $u(t)$ et d'une *dynamique non-linéaire* dans le cas contraire.

Les méthodes de résolution numérique du problème (1.1) peuvent être classifiées en méthodes **directes** et **indirectes**. Pour les premières, la solution optimale s'obtient par une discrétisation directe du problème; on ramène le problème à résoudre à un problème d'optimisation en dimension finie [13], [56]. Ce dernier est alors résolu, par exemple, en utilisant les algorithmes SQP [24] ou la méthode des Points Intérieurs [57], pour n'en citer que deux. Les méthodes indirectes sont basées sur le Principe du Maximum de Pontryagin, principe qui énonce les conditions nécessaires d'optimalité; ceci conduit à un problème au deux bouts [19]. Ce dernier est résolu, par exemple, en utilisant une méthode de tir [15, 14, 18].

Le point commun de ces techniques de résolution du problème de contrôle optimal est l'existence des conditions nécessaires d'optimalité: les conditions Karush-Kuhn-Tucker pour les méthodes directes et le Principe du Maximum pour les méthodes indirectes.

Le problème auquel on va s'intéresser est obtenu à partir de (1.1) en ajoutant la contrainte suivante (notée C_Δ):

Sur chaque intervalle de temps $[\tau, \tau + \delta] \subset [t_0, t_f]$ de longueur δ (où $\tau \in [t_0, t_f - \delta]$) la durée du contrôle ne peut pas dépasser δ_0 ($0 < \delta_0 < \delta$).

Ce type de contrainte cumulative peut s'interpréter comme une mesure de sécurité concernant le bon fonctionnement de l'organe de commande. Néanmoins, un autre type d'application a été à l'origine de l'étude présentée ici: pour un moteur solaire (comme cela peut-être le cas si notre système est un satellite en mouvement autour de la Terre) la durée de fonctionnement est conditionnée par la visibilité du Soleil. Cela interdit une utilisation prolongée de l'énergie fournie par le Soleil.

Outre l'intérêt pratique du problème, notre motivation a aussi un fort caractère théorique. Si pour le problème du contrôle (1.1), différentes techniques de résolution efficaces existent, en ajoutant la contrainte logique on rencontre un bon nombre de difficultés supplémentaires, issues notamment de la nature logique de cette contrainte. En effet, la modélisation passe par l'introduction des fonctions discontinues (traduites à leur tour par des variables entières après la discrétisation), ce qui complique considérablement la façon de traiter le problème ; les hypothèses nécessaires pour les conditions d'optimalité n'étant plus satisfaites, des nouveaux algorithmes (combinatoires vu la nature du problème) sont nécessaires pour trouver et surtout pour prouver l'optimalité de la solution.

Une façon de modéliser la contrainte logique passe par l'introduction de la fonction sgn associée au contrôle, définie comme suit:

$$sgn(u(t)) = \begin{cases} 0, & \text{si } |u(t)| = 0, \\ 1, & \text{sinon} \end{cases} \quad (1.2)$$

Avec cela, la contrainte C_Δ devient:

$$\int_t^{t+\delta} sgn(u(s))ds \leq \delta_0 \quad t \in [t_0, t_f - \delta]. \quad (1.3)$$

En introduisant cette équation (1.3) dans (1.1), notre problème d'optimisation s'écrit:

$$\left\{ \begin{array}{ll} \underset{x(t), u(t)}{Min} & \frac{1}{2} \int_0^{t_f} |u(t)|^2 dt \\ s.t. & \dot{x}(t) = F(t, x(t), u(t)) \quad t \in [0, t_f] \\ & \int_t^{t+\delta} sgn(u(s))ds \leq \delta_0 \quad t \in [0, t_f - \Delta] \\ & x(0) = x^0 \\ & x(t_f) = x^f \\ & |u(t)| \leq U_{max} \end{array} \right. \quad (1.4)$$

Malgré l'expression mathématique donnée en (1.4), ce modèle est difficilement exploitable en pratique car, d'une part la contrainte C_Δ est exprimée comme une contrainte sur le signe du contrôle comportant une expression intégrale et, d'autre part, la fonction sgn (introduite afin de formuler le problème) est discontinue, ce qui pose des problèmes du point de vue numérique.

Une méthode de résolution de ce modèle à été testée en reformulant le problème (1.4) comme un problème d'optimisation avec retard et en appliquant le principe de Pontryagin sur ce dernier. Comme attendu, la fonction sgn empêche la convergence numérique dans la résolution du BVP (Boundary Value Problem) et aucune solution n'a été obtenue [42].

D'autres tests utilisant des formulations continues [7] ont renforcés notre conviction que ce type de modélisation a des désavantages difficiles à contourner. La démarche qu'on propose est semblable à l'approche directe pour

les problèmes de contrôle optimal: on discrétise le problème (1.1) et on ajoute ensuite des variables binaires afin d'exprimer la contrainte C_Δ . De cette façon, les discontinuités de la fonction sgn vont se retrouver dans les modélisations à travers les variables binaires et le modèle ainsi obtenu fait partie de la classe de problèmes MINLP (problèmes d'optimisation en variable mixte).

1.2 Discrétisation

On commence par obtenir une formulation en dimension finie pour le problème (1.1). La méthode choisie pour discrétiser la formulation continue est celle de la **collocation directe** [56].

Dans ce but, soit $N+1$ le nombre de points de discrétisation de l'intervalle $[0, t_f]$ et t_i , $i = 0, \dots, N$ les instants discrétisés. On considère que le pas de discrétisation est constant $h = t_{i+1} - t_i$, sans restreindre la généralité du problème car les mêmes considérations présentées ici s'appliquent à une discrétisation à pas variable.

Dans le modèle qu'on propose, on approche l'état par des fonctions polynomiales de degré 3 sur chaque intervalle i , de discrétisation $[t_i, t_{i+1}]$ (avec $t_i = \frac{i}{N}t_f$, $i = 0, N$):

$$x(t) \approx c_i^3 \left(\frac{t - t_i}{h}\right)^3 + c_i^2 \left(\frac{t - t_i}{h}\right)^2 + c_i^1 \frac{t - t_i}{h} + c_i^0, t \in [t_i, t_{i+1}), i = 0, N-1 \quad (1.5)$$

où $c_i^0, c_i^1, c_i^2, c_i^3 \in \mathbb{R}^n$ sont des variables du problème, qui restent à être déterminées.

La fonction contrôle est à son tour discrétisée et approximée par des fonctions constantes par morceaux:

$$u(t) \approx u_i \in \mathbb{R}^m, \quad t \in [t_i, t_{i+1}) \quad (1.6)$$

Afin de déterminer les coefficients de la collocation (c_i) on utilise les conditions suivantes:

- $x(\cdot)$ est une fonction continue sur l'intervalle $[0, t_f]$:

$$\lim_{t \rightarrow t_{i+1}^-} x(t) = \lim_{t \rightarrow t_{i+1}^+} x(t) \Rightarrow c_i^3 + c_i^2 + c_i^1 + c_i^0 = c_{i+1}^0 \quad i = 0, N-1 \quad (1.7)$$

- $\dot{x}(\cdot)$ est une fonction continue sur l'intervalle $[0, t_f]$:

$$\lim_{t \rightarrow t_{i+1}^-} \frac{dx}{dt}(t) = \lim_{t \rightarrow t_{i+1}^+} \frac{dx}{dt}(t) \Rightarrow 3c_i^3 + 2c_i^2 + c_i^1 = c_{i+1}^1 \quad i = 0, N-1 \quad (1.8)$$

- $x(t), u(t)$ doivent respecter la dynamique du système au point t_i , pour $i = 0, N$:

$$\frac{dx}{dt}(t_i) = F(t_i, x(t_i), u(t_i)) \Rightarrow c_i^1 = hF_i(x_i, u_i) \quad (1.9)$$

où $F_i(x_i, u_i) \triangleq F(t_i, x_i, u_i)$ est le second membre de la dynamique à l'instant t_i .

Les équations (1.7), (1.8) et (1.9) avec la notation $x_i \triangleq x(t_i)$ nous permettent d'expliciter les expressions des coefficients c_i , $i = 0, N$:

$$\begin{cases} c_i^0 &= x_i \\ c_i^1 &= hF_i(x_i, u_i) \\ c_i^2 &= 3x_{i+1} - 3x_i - 2hF_i(x_i, u_i) - hF_{i+1}(x_{i+1}, u_{i+1}) \\ c_i^3 &= 2x_i - x_{i+1} + hF_i(x_i, u_i) + hF_{i+1}(x_{i+1}, u_{i+1}) \end{cases} \quad (1.10)$$

1.2.1 Équations de la collocation

Les équations discrétisées donnant l'évolution du système (1.1) au cours du temps s'obtiennent en imposant que la dynamique soit vérifiée en chaque point milieu $\frac{t_i + t_{i+1}}{2}$ des intervalles de discrétisation.

En explicitant:

- la dérivée de l'état:

$$\frac{dx}{dt}\left(\frac{t_{i+1} + t_i}{2}\right) = -\frac{3}{2h}x_i + \frac{3}{2h}x_{i+1} - \frac{1}{4}F_i(x_i, u_i) - \frac{1}{4}F_{i+1}(x_{i+1}, u_{i+1}) \quad (1.11)$$

- l'état au point milieu:

$$x\left(\frac{t_{i+1} + t_i}{2}\right) = \frac{1}{2}x_i + \frac{1}{2}x_{i+1} + \frac{h}{8}F_i(x_i, u_i) - \frac{h}{8}F_{i+1}(x_{i+1}, u_{i+1}) \quad (1.12)$$

la dynamique du système :

$$\dot{x}\left(\frac{t_{i+1} + t_i}{2}\right) = F\left(\frac{t_i + t_{i+1}}{2}, x\left(\frac{t_{i+1} + t_i}{2}\right), u(t_i)\right) \quad (1.13)$$

devient:

$$\begin{aligned} & -\frac{3}{2h}x_i + \frac{3}{2h}x_{i+1} - \frac{1}{4}F_i(x_i, u_i) - \frac{1}{4}F_{i+1}(x_{i+1}, u_{i+1}) = \\ & F\left(\frac{t_i + t_{i+1}}{2}, \frac{1}{2}x_i + \frac{1}{2}x_{i+1} + \frac{h}{8}F_i(x_i, u_i) - \frac{h}{8}F_{i+1}(x_{i+1}, u_{i+1}), u_i\right) \end{aligned} \quad (1.14)$$

En remplaçant dans (1.14) l'expression explicite de la dynamique (*i.e.* $F(t, x(t), u(t))$) et en développant cette équation on obtient "le système de collocation". Le problème d'optimisation issu de la discrétisation du problème de contrôle optimal (1.1) est de la forme:

$$\left\{ \begin{array}{ll} \min_{x,u} & \frac{1}{2} \sum_{i=0}^N |u_i|^2 \\ s.t. & F(x,u) = 0 \\ & x_0 = x^0 \\ & x_N = x^f \\ & |u|^2 \leq U_{max}^2 \end{array} \right. \quad (1.15)$$

où:

- x est le vecteur des variables d'état, $x \in \mathbb{R}^{n(N+1)}$
- u est le vecteur des variables de contrôle, $u \in \mathbb{R}^{m(N+1)}$
- $F : \mathbb{R}^{(N+1)(n+m)} \mapsto \mathbb{R}^n$ est la fonction obtenue en utilisant la collocation directe, $F = (F_0, F_1, \dots, F_N)$ où $F_i : \mathbb{R}^{(N+1)(n+m)} \mapsto \mathbb{R}$ est obtenue en explicitant l'équation (1.14).

Les deux exemples qui suivent explicitent l'équation (1.14) pour deux problèmes particuliers: un mobile unidimensionnel régi par une dynamique linéaire et le mouvement d'un satellite autour de la Terre, exprimé en coordonnées cartésiennes (dynamique fortement non-linéaire).

Dynamique discrète dans le cas linéaire. On considère le cas d'un mobile assujetti à un mouvement rectiligne sur l'axe x , ayant une position et une vitesse initiales données (x_0, v_0) et que l'on veut ramener au repos à l'origine $(x_f = 0, v_f = 0)$ en contrôlant l'accélération u (l'état x se traduit en x = position et y = vitesse dans ce cas). Le problème de contrôle optimal s'écrit dans ce cas:

$$\left\{ \begin{array}{ll} Min & \int_0^{t_f} u^2(t) dt \\ s.t. & \begin{array}{ll} \frac{dx}{dt}(t) = y(t) & t \in [0, t_f] \\ \frac{dy}{dt}(t) = u(t) & t \in [0, t_f] \\ x(0) = x^0, x(t_f) = x^f \\ y(0) = v^0, y(t_f) = v^f \end{array} \end{array} \right. \quad (1.16)$$

et il devient, après la discrétisation par la méthode de la collocation directe:

$$\left\{ \begin{array}{ll} Min & u^T u \\ s.t. & \begin{array}{ll} -2x_i + 2x_{i+1} - 2hy_i - h^2u_i = 0 & i = \overline{0, N-1} \\ 10x_i - 10x_{i+1} + 4hy_i + 6hy_{i+1} - h^2u_{i+1} = 0 & i = \overline{0, N-1} \\ x_0 = x^0, x_N = x^f, y_0 = v^0, y_f = v^f \\ x \in \mathbb{R}^{N+1}, y \in \mathbb{R}^{N+1}, u \in \mathbb{R}^{N+1} \end{array} \end{array} \right. \quad (1.17)$$

On remarque que le problème d'optimisation ainsi obtenu est quadratique.

Dynamique discrète pour le transfert orbital. Le système sur lequel on veut appliquer la collocation est donné par les équations du mouvement d'un satellite (en coordonnées cartésiennes: x, y, v_x, v_y qui représentent notre état x) contrôlé par u_x et u_y dans le cas 2D:

$$\begin{cases} \frac{dx}{dt}(t) &= v_x(t) & , t \in [0, t_f] \\ \frac{dv_x}{dt}(t) &= -k \frac{x(t)}{((x(t))^2 + (y(t))^2)^{\frac{3}{2}}} + u_x(t) & , t \in [0, t_f] \\ \frac{dy}{dt}(t) &= v_y(t) & , t \in [0, t_f] \\ \frac{dv_y}{dt}(t) &= -k \frac{y(t)}{((x(t))^2 + (y(t))^2)^{\frac{3}{2}}} + u_y(t) & , t \in [0, t_f] \end{cases} \quad (1.18)$$

On obtient ainsi les quatre équations suivantes:

$$\begin{cases} \frac{kh}{8} \frac{x_i}{((x_i)^2 + (y_i)^2)^{\frac{3}{2}}} - \frac{kh}{8} \frac{x_{i+1}}{((x_{i+1})^2 + (y_{i+1})^2)^{\frac{3}{2}}} - \frac{3}{2h} x_i + \frac{3}{2h} x_{i+1} - \frac{3}{4} v_i^x - \frac{3}{4} v_{i+1}^x - \frac{h}{8} u_i^x + \frac{h}{8} u_{i+1}^x &= 0 \\ \frac{kh}{8} \frac{y_i}{((x_i)^2 + (y_i)^2)^{\frac{3}{2}}} - \frac{kh}{8} \frac{y_{i+1}}{((x_{i+1})^2 + (y_{i+1})^2)^{\frac{3}{2}}} - \frac{3}{2h} y_i + \frac{3}{2h} y_{i+1} - \frac{3}{4} v_i^y - \frac{3}{4} v_{i+1}^y - \frac{h}{8} u_i^y + \frac{h}{8} u_{i+1}^y &= 0 \\ -\frac{3}{2h} v_i^x + \frac{3}{2h} v_{i+1}^x + \frac{k}{4} \frac{x_i}{((x_i)^2 + (y_i)^2)^{\frac{3}{2}}} + \frac{k}{4} \frac{x_{i+1}}{((x_{i+1})^2 + (y_{i+1})^2)^{\frac{3}{2}}} + k \frac{\pi_i^x}{((\pi_i^x)^2 + (\pi_i^y)^2)^{\frac{3}{2}}} - \frac{5}{4} u_i^x - \frac{1}{4} u_{i+1}^x &= 0 \\ -\frac{3}{2h} v_i^y + \frac{3}{2h} v_{i+1}^y + \frac{k}{4} \frac{y_i}{((x_i)^2 + (y_i)^2)^{\frac{3}{2}}} + \frac{k}{4} \frac{y_{i+1}}{((x_{i+1})^2 + (y_{i+1})^2)^{\frac{3}{2}}} + k \frac{\pi_i^y}{((\pi_i^x)^2 + (\pi_i^y)^2)^{\frac{3}{2}}} - \frac{5}{4} u_i^y - \frac{1}{4} u_{i+1}^y &= 0 \end{cases} \quad (1.19)$$

où:

$$\begin{cases} \pi_i^x &= \frac{1}{2} x_i + \frac{1}{2} x_{i+1} + \frac{h}{8} v_i^x - \frac{h}{8} v_{i+1}^x & i = \overline{0, N-1} \\ \pi_i^y &= \frac{1}{2} y_i + \frac{1}{2} y_{i+1} + \frac{h}{8} v_i^y - \frac{h}{8} v_{i+1}^y & i = \overline{0, N-1} \end{cases} \quad (1.20)$$

représentent les coordonnées (position) au point milieu de chaque intervalle de discrétisation.

1.3 Contraintes logiques

On va s'intéresser maintenant à la modélisation des contraintes C_Δ . La nature logique de ces contraintes et le fait de contrôler ou non le système à l'instant t_i , nous amènent à introduire des variables de décision binaires. Soit $y \in \{0,1\}^{N+1}$ le vecteur des variables binaires y_i , où $N+1$ est le nombre de points de discrétisation; le rôle de ces variables est d'exprimer le fait qu'à l'instant t_i on contrôle ($y_i = 1$) ou non ($y_i = 0$) le système, leur influence sur le problème pouvant s'écrire:

- $y_i = 0 \Rightarrow u_i = 0$ (on ne contrôle pas le système)
- $y_i = 1 \Rightarrow |u_i| \in [-U_{max}, U_{max}]$ (on contrôle le système)

Il faut remarquer que dans le cas d'un contrôle multidimensionnel ($m > 1$) le nombre de variables binaires est toujours $N + 1$, car si à l'instant t_i on ne veut pas contrôler le système toutes les composantes u_i^j , $j = 1, m$ du contrôle doivent s'annuler. Ceci est un fait important car le nombre de variables entières y (qui donnent au problème sa nature combinatoire) ne dépend pas de la dimension du contrôle mais seulement de la taille de la discrétisation.

On peut envisager deux façons différentes pour exprimer le rôle des variables y sous la forme des contraintes dans le problème d'optimisation. La première est de remplacer dans le problème de contrôle optimal discrétisé chaque occurrence de la variable u_i^j par le produit $u_i^j y_i$. Avec cette remarque, le problème d'optimisation (1.15) s'écrit:

$$\left\{ \begin{array}{l} \text{Min} \quad \frac{1}{2} \sum_{i=0}^N (u_i \odot y_i)^2 \\ \text{s.t.} \quad F(x, u \odot y) = 0 \\ x \in X^{n(N+1)} \\ u \in U^{m(N+1)} \\ y \in \{0,1\}^{N+1} \end{array} \right. \quad (1.21)$$

où l'opération (\odot) est définie par:

$$u \odot y = (u_0 y_0, \dots, u_i y_i, \dots, u_N y_N)$$

On remarquera que la formulation (1.21) modélise juste le problème de contrôle optimal et le rôle des variables discrètes y , la contrainte C_Δ n'étant pas encore intégrée dans les équations.

Cette façon d'exprimer le rôle des contraintes logiques a le désavantage de changer le type du problème, outre le fait de le transformer en un problème d'optimisation mixte. En effet, si le problème de contrôle optimal de départ a une dynamique linéaire alors le problème discrétisé est quadratique mais le problème (1.21) est non-convexe en (x, y, u) . C'est pour cette raison qu'on a cherché une autre modélisation intégrant les variables binaires.

Afin d'obtenir cette deuxième formulation, on va considérer le cas d'un contrôle unidimensionnel ($m = 1$) afin d'alléger les notations (le cas multidimensionnel se traite de façon analogue). On remarquera que, si $m = 1$:

$$\left\{ \begin{array}{l} y_i = 0 \Rightarrow u_i = 0 \\ y_i = 1 \Rightarrow u_i \in [-U_{max}, U_{max}] \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} u_i - U_{max} y_i \leq 0 \\ u_i + U_{max} y_i \geq 0 \end{array} \right. \quad (1.22)$$

Il s'agit donc de coupler les contraintes de bornes sur le contrôle avec le rôle des variables binaires, ce qui permet de formuler le problème d'optimisation

comme:

$$\left\{ \begin{array}{l} \text{Min} \quad \frac{1}{2}u^T u \\ \text{s.t.} \quad \bar{F}(x,u) = 0 \\ \quad \quad u_i - U_{max}y_i \leq 0 \quad i = \overline{0,N} \\ \quad \quad u_i + U_{max}y_i \geq 0 \quad i = \overline{0,N} \\ \quad \quad x \in X^{n(N+1)} \\ \quad \quad u \in U^{N+1} \\ \quad \quad y \in \{0,1\}^{N+1} \end{array} \right. \quad (1.23)$$

La formulation multidimensionnelle du contrôle nécessite en plus du cas $m = 1$ d'ajouter la contrainte $|u|^2 \leq U_{max}^2$, le reste du problème étant identique à (1.23).

L'avantage évident de cette modélisation est le fait qu'un problème de contrôle optimal de départ ayant une dynamique linéaire nous amène à un problème mixte ayant toujours une dynamique linéaire. Ceci est dû aux équations linéaires mixtes faisant intervenir le contrôle et les variables y , équations qui expriment le rôle des variable entières (on contrôle ou non le système à l'instant t_i). Quoi que non-convexe, à cause des variables binaires, pour ce dernier problème il existe des algorithmes spécifiques pouvant s'appliquer (on peut citer ici la Séparation et Évaluation, l'Approximation Extérieure et la Décomposition du Benders, algorithmes qu'on va analyser de plus près dans le chapitre 3).

Quelle que soit la formulation, (1.21) ou (1.23), les contraintes logiques C_Δ peuvent s'exprimer par des inégalités linéaires faisant intervenir seulement les variables entières:

$$\left\{ \begin{array}{ll} y_0 + y_1 + \dots + y_{\Delta-1} & \leq \Delta_0 \\ y_1 + y_2 + \dots + y_\Delta & \leq \Delta_0 \\ \dots & \\ y_i + y_{i+1} + \dots + y_{i+\Delta-1} & \leq \Delta_0 \\ \dots & \\ y_{N-\Delta} + y_{N-\Delta+1} + \dots + y_{N-1} & \leq \Delta_0 \\ y_{N-\Delta+1} + y_{N-\Delta+2} + \dots + y_N & \leq \Delta_0 \end{array} \right. \quad (1.24)$$

ou Δ et Δ_0 sont deux constantes entières approchant au mieux les paramètres δ et δ_0 .

Ce système d'inéquations linéaires (1.24) s'écrit matriciellement:

$$C_\Delta y \leq \Delta_0 \quad (\text{équations } C_\Delta), \quad (1.25)$$

C_Δ étant la matrice explicitée en (1.24), y le vecteur des variables binaires. Par abus de langage Δ_0 est ici un vecteur dont tous les éléments sont égaux à Δ_0 , nombre entier.

Dans le chapitre suivant, on analysera plus en détail la structure particulière que ces contraintes induisent sur l'ensemble des vecteurs binaires y qui satisfont les contraintes C_Δ .

1.4 Problèmes mixtes

Avant de regarder plus en détail les différents types de problèmes mixtes qui nous intéressent, on va donner dans ce paragraphe les différents types de problèmes mixtes et les principes généraux de leur résolution. On précise dans ce paragraphe introductif que la principale difficulté d'un tel algorithme est de prouver l'optimalité d'une solution réalisable. L'idée utilisée par tous les algorithmes existants est de générer, à chaque itération, un majorant (UBD) et un minorant (LBD) de la solution optimale; d'une itération à l'autre on essaye alors d'améliorer ces bornes LBD et UBD, en arrêtant l'algorithme quand $UBD - LBD \leq \varepsilon$. La différence entre les différents algorithmes du mixte réside dans la façon dont les minorants et les majorants sont calculés.

Une classification des problèmes d'optimisation **MINLP** est alors possible suivant la facilité avec laquelle on les résout.

Le problème mixte le plus simple est le problème **MILP** (Mixed Integer Linear Problem), caractérisé par un objectif et des contraintes affines en variables entières et réelles. Une autre classe de problèmes, **MIQP** (Mixed Integer Quadratic Problem) est semblable aux précédents avec comme seule spécificité, un objectif quadratique.

Le problème suivant est la formulation la plus générale d'un problème mixte, celui des **MINLP** (Mixed Integer Non-Linear Programming):

$$\left\{ \begin{array}{ll} \min_{x,y} & f(x,y) \\ \text{s.t.} & h(x,y) = 0 \\ & g(x,y) \leq 0 \\ & x \in X \subseteq \mathbb{R}^n \\ & y \in Y \text{ entier} \end{array} \right. \quad (1.26)$$

où X est un hypercube de \mathbb{R}^n et x est le vecteur des n variables continues, $y \in Y \subseteq \mathbb{Z}^m$ le vecteur des variables entières, $h(x,y) = 0$ désigne les contraintes d'égalité, $g(x,y) \leq 0$ les contraintes d'inégalité et $f(x,y)$ représente le coût à minimiser. Ce type de problème présente aussi quelques sous-types particuliers, si la relaxation continue est convexe ou non, si le problème d'optimisation continu en x , obtenu en fixant les variables entières à une combinaison particulière, est convexe ou non, si les contraintes et l'objectif sont séparables en x et y , pour n'en citer que quelques uns.

On conclut ce paragraphe en faisant le point sur ce qu'on peut appeler "problème facile" en optimisation mixte. Les problèmes continus linéaires (LP, Linear Programming), quadratiques (QP, Quadratic Programming) ou non-linéaires (NLP, Non-Linear Programming) sont considérés comme simples à résoudre, et cela dans un temps suffisamment raisonnable pour les intégrer comme une boucle interne à un algorithme de résolution MINLP. On se rend bien compte que cela n'est pas toujours le cas (exemple du contrôle

optimal dans le transfert d'orbite à faible poussée) mais dans bien des cas le temps de calcul d'un tel problème est effectivement négligeable devant celui d'un problème MINLP. Une autre classe de problèmes considérés simples est celui des **MILP** (problème mixte linéaire en x et y) et **MIQP** (problème mixte quadratique en x et y); bien qu'il s'agisse d'un problème mixte (et donc **NP-difficile**), les progrès théoriques effectués dans le domaine mixte linéaire et la qualité des solveurs existant sur le marché (CPLEX, OSL, EX-PRESS) nous permettent de dire qu'un tel problème MILP ou MIQP peut être résolu efficacement.

1.5 Problème en variables mixtes associé à une dynamique linéaire

On va distinguer par la suite le type de problème **MINLP** qu'on obtient suivant la nature de la dynamique. Cette distinction est nécessaire car on a des algorithmes différents dans chaque cas particulier.

Dans le cas d'une dynamique linéaire le problème de contrôle optimal s'écrit:

$$\left\{ \begin{array}{ll} \min_{x,u} & \frac{1}{2} \int_0^{t_f} |u(t)|^2 dt \\ \text{s.t.} & \frac{dx}{dt}(t) = A(t)x(t) + B(t)u(t) + C(t) \quad , t \in [0, t_f] \\ & x(0) = x^0 \\ & x(t_f) = x^f \end{array} \right. \quad (1.27)$$

Comme on l'a expliqué dans le paragraphe antérieur, on a deux façons différentes de modéliser le rôle des variables binaires y ; deux problèmes MINLP peuvent être alors formulés. La première s'obtient directement à partir du (1.21) en ajoutant les contraintes C_Δ (formule 1.25) et en tenant compte du fait que f est une fonction linéaire en x et u dans ce cas. Le problème MINLP est alors de la forme:

$$\left\{ \begin{array}{ll} \min_{x,u} & \frac{1}{2} u^T u \\ \text{s.t.} & Ax = b \quad \text{(Conditions aux deux bouts)} \\ & Cx + D(u \odot y) = d \quad \text{(Equations de la dynamique)} \\ & C_\Delta y \leq \Delta_0 \quad \text{(Contraintes logiques)} \\ & -U_{max} \leq u_i \leq U_{max} \quad , i = \overline{0, N} \quad \text{(Contrôle limite)} \end{array} \right. \quad (1.28)$$

où l'opération (\odot) dénote toujours:

$$u \odot y = (u_0 y_0, \dots, u_i y_i, \dots, u_N y_N)$$

Ce type de problème MINLP est non-linéaire et sa relaxation continue (obtenue en relaxant la contrainte $y \in \{0,1\}^{N+1}$ en $y \in [0,1]^{N+1}$) est non-convexe; en effet, c'est le terme $u \odot y$ apparaissant dans la dynamique du

problème qui rend le problème non-linéaire. Néanmoins, cette formulation mérite d'être distinguée du cas général **MINLP** car, en fixant les variables binaires à une combinaison particulière (autrement dit, en précisant la structure du contrôle par les instants où on peut contrôler le système) on obtient un problème quadratique, donc convexe. C'est cette propriété qu'on va exploiter dans le chapitre suivant en présentant un algorithme développé par nous, une variante de l'algorithme plus général de Benders (connue sous le nom de **GBD**, Generalized Benders Decomposition).

La deuxième modélisation s'obtient en exploitant les contraintes de boîte pour le contrôle (voir 1.22). Le problème MINLP ainsi obtenu est de la forme:

$$\left\{ \begin{array}{ll} \min_{x,u} & \frac{1}{2}u^T u \\ \text{s.t.} & Ax = b \quad (\text{Conditions aux deux bouts}) \\ & Cx + Du = d \quad (\text{Equations de la dynamique}) \\ & C_{\Delta}y \leq \Delta_0 \quad (\text{Contraintes logiques}) \\ & -U_{max}y_i \leq u_i \leq U_{max}y_i, \quad i = \overline{0,N} \quad (\text{Contraintes boîte}) \end{array} \right. \quad (1.29)$$

Il s'agit ici d'un problème **MIQP** (Mixed Integer Quadratic Problem, problème mixte quadratique), cas particulier de problème **MINLP**, pour lequel des différentes algorithmes existent: **GOA** (Generalized Outer Approximation, [25]), **GBD** (Generalized Benders Decomposition, [28]) et **BB** (Branch and Bound, [39]). On va présenter dans le chapitre suivant les principes de ces algorithmes et les résultats qu'on obtient en les appliquant au problème (1.29).

1.6 Problème en variables mixtes associés à une dynamique non-linéaire. Transfert orbital

On s'intéresse ici au problème de transfert orbital, dont la dynamique non-linéaire est donnée par l'équation de Newton dans le cas d'un satellite en mouvement autour de la Terre:

$$\ddot{r} = -\mu \frac{r}{|r|^3} + \frac{1}{m}u \quad (1.30)$$

(r étant le rayon vecteur, u la poussée contrôlant le système et m la masse du satellite, considérée constante tout au long du transfert).

Le problème de MINLP obtenu en ajoutant les contraintes C_{Δ} fait partie de la classe la plus générale des problèmes mixtes; en effet, en fixant les variables entières à une combinaison particulière, le problème d'optimisation en variable continue ainsi obtenu n'est pas convexe. En l'absence d'un algorithme spécifique pour traiter ce type de problème, le travail qu'on a effectué consiste à adapter l'algorithme générique de Branch and Reduce (une

variante de Branch and Bound, évaluation et séparation) à notre type de problème. Le point central de cette étude (exposée plus en détail en chapitre 4) est de trouver les meilleures relaxations convexes pour notre problème de transfert orbital, la relaxation convexe étant le concept clé dans l'algorithme de Branch and Reduce [50],[55].

Comme on va le voir dans le chapitre consacré à cette méthode, la démarche utilisée par l'algorithme de Branch and Reduce est la même dans le cas d'un problème d'optimisation non-convexe en variables continues et dans le cas d'un problème MINLP. C'est pour cette raison qu'au quatrième chapitre on laisse de côté les variables binaires, en nous intéressant de plus près à l'optimisation globale continue.

On va présenter dans ce chapitre les résultats obtenus sur un problème de test simple, ayant peu de termes non-convexes; ces résultats sont encourageants mais se dégradent vite avec le nombre de termes non-linéaires dans le problème. Ainsi, pour le problème de transfert orbital 2D à masse constante les résultats obtenus sont nettement inférieurs à ceux des problèmes issus d'une dynamique linéaire.

Chapitre 2

Contraintes logiques

Ce chapitre est consacré à l'analyse des contraintes logiques. On va s'intéresser successivement aux questions liées à l'espace de recherche induit par les contraintes logiques : quelles sont les classes de contrôle qui ne peuvent pas être optimaux, existe-t-il des solutions entières meilleures que d'autres, comment caractériser ces solutions entières et quelles sont les propriétés particulières de la matrice des contraintes logiques C_Δ ?

On commence par reformuler le problème en variables mixtes, en anticipant sur la façon dont les algorithmes existants essayent de le résoudre. On va se rendre compte de l'intérêt d'une analyse des contraintes logiques. Les "meilleurs" vecteurs entiers sont décrits ci-après et la structure de l'espace de recherche est réduite en se basant sur ces considérations. On présentera ensuite une façon élégante pour éviter l'exploration d'une solution entière déjà analysée et on finit par donner une propriété intéressante de la matrice de contraintes C_Δ .

2.1 Espace de recherche. Contraintes (C_Δ)

Avec le couple état-contrôle (x, u) variables continues et y variables de décision (binaires) notre problème MINLP peut se mettre sous la forme:

$$\left\{ \begin{array}{ll} \min_{y \in \{0,1\}^{N+1}} & \min_{x,u} \frac{1}{2} u^T u \\ s.t. & h(x, u, y) = 0 \\ & g(x, u, y) \leq 0 \\ & C_\Delta y \leq \Delta_0 \\ & x \in \mathbb{R}^{n(N+1)}, u \in \mathbb{R}^{m(N+1)}, y \in \mathbb{R}^{N+1}. \end{array} \right.$$

En définissant:

$$Y_\Delta = \{y \in \{0,1\}^{N+1} : C_\Delta y \leq \Delta_0\} \quad (2.2)$$

le problème devient:

$$\left\{ \begin{array}{ll} \min_{y \in Y_\Delta} & \min_{x, u} \quad \frac{1}{2} u^T u \\ & s.t. \quad h(x, u, y) = 0 \\ & \quad g(x, u, y) \leq 0 \\ & \quad x \in \mathbb{R}^{n(N+1)}, u \in \mathbb{R}^{m(N+1)} \end{array} \right.$$

Décomposé de cette façon, un problème **MINLP** peut se résoudre en explorant l'ensemble des vecteurs entiers Y_Δ ; fixer un certain $y^k \in Y_\Delta$ revient à résoudre le problème associé $P(y^k)$ (problème obtenu du (P) en fixant les valeurs des variables binaires à une combinaison particulière donnée; dans notre cas cela revient à fixer la structure du contrôle, en définissant les instants où l'on peut contrôler le système).

Cette démarche est adoptée par les algorithmes standards (**GBD**, **GOA**, **BB** présentés plus en détail dans le chapitre suivant), d'où l'importance de caractériser l'ensemble Y_Δ . De manière plus précise ce chapitre répond aux questions suivantes:

- une fois qu'on décide d'analyser une solution entière y^k , peut-on l'améliorer avant de résoudre le problème associé $P(y^k)$;
- existe-t-il des vecteurs entiers privilégiés, dont on sait qu'ils donnent des meilleures solutions; en conséquence, peut-on réduire l'ensemble Y_Δ ?
- une fois une solution y^k analysée existe-t-il une façon efficace de l'éliminer de l'espace de recherche?
- quelles sont les propriétés de la matrice des contraintes logiques (C_Δ)?

2.2 Notion de solution saturée

Supposons qu'à l'itération k d'un algorithme on décide d'analyser la solution y^k (c'est-à-dire de résoudre le problème associé $P(y^k)$). On veut savoir si, en regardant la structure de y^k on peut trouver une meilleure solution \bar{y}^k , i.e.: $P(y^k) \geq P(\bar{y}^k)$. Le théorème suivant apporte une première partie de réponse.

Théorème 2.2.1. *Si $y^1, y^2 \in Y_\Delta$ sont deux vecteurs entiers, avec $y_i^1 - y_i^2 \geq 0$, $i = \overline{0, N}$ alors $P(y^1) \leq P(y^2)$ (c'est-à-dire la solution du problème $P(y^1)$ est meilleure que celle du problème $P(y^2)$).*

Démonstration. La condition du théorème $y_i^1 - y_i^2 \geq 0, i = \overline{0, N}$ signifie qu'à chaque apparition d'un 1 sur une position i dans le vecteur y^2 on a un 1 sur la même position dans y^1 , sachant que le vecteur y^1 peut contenir d'autres composantes égales à 1. Cela signifie que le domaine de variation du contrôle du problème ($P(y^1)$) inclut le domaine de variation du contrôle du problème ($P(y^2)$), d'où le résultat. \square

Ce résultat est important car il donne le type des solutions entières à explorer (en réduisant ainsi l'espace de recherche): en partant de y^k on remplace chaque $y_i^k = 0$ par $y_i^k = 1$ tant que cette opération garde la faisabilité ($C_\Delta y^k \leq \Delta_0$). Cette transformation s'arrête après un nombre fini d'itérations obtenant ainsi un nouveau vecteur $\bar{y}^k \in Y$ qui ne peut plus être amélioré dans le sens du théorème; le pseudo-code suivant détaille cette procédure:

```

pour chaque composante  $y_i^k$  du vecteur  $y^k$ 
  si  $y_i^k = 0$  alors
    soit  $\tilde{y}^k = (y_0^k, y_1^k, \dots, y_{i-1}^k, 1, y_{i+1}^k, \dots, y_N^k)$ 
    si  $\tilde{y}^k \in C_\Delta$  alors  $y^k = \tilde{y}^k$ ;
  fin si
fin pour

```

Définition 2.2.1 (Solution saturée). Une solution $y \in Y_\Delta$ est dite *saturée* si et seulement si pour chaque $i \in \{0, \dots, N\}$ tel que $y_i = 0$, le vecteur entier obtenu du y en imposant $y_i = 1$ ne respecte pas les contraintes logiques (*i.e.* $y \notin Y_\Delta$).

Soit alors

$$\bar{Y}_\Delta = \{y \in Y_\Delta \mid y \text{ est saturée}\} \quad (2.4)$$

Définition 2.2.2 (Contrainte saturée). Une contrainte logique est dite saturée si et seulement si sa valeur est égale à Δ_0 .

Remarque 2.2.1. La saturabilité d'une solution n'entraîne pas automatiquement la saturation de toutes les contraintes (C_Δ).

Remarque 2.2.2. Le problème $P(\bar{y}^k)$ est une relaxation de tout problème $P(y^\alpha)$ pour lequel il existe une suite de saturations qui transforment y^α en \bar{y}^k .

Remarque 2.2.3. Malgré cette réduction de l'espace de recherche, le cardinal de \bar{Y}_Δ reste très grand dans une application réelle.

Cela nous amène à la question suivante: dans quelles conditions peut-on saturer une solution y^k ?

Proposition 2.2.1. *Une solution $y \in Y_\Delta$ est saturable si et seulement s'il existe un indice $i \in \{0..N\}$ telle qu'aucune des contraintes C_Δ dans lesquelles y_i apparaisse ne soit saturée (chaque contrainte a une valeur plus petite que Δ_0).*

La démonstration de cette proposition est évidente et s'obtient directement à partir de la définition d'une solution *saturée*.

Cette notion de saturation nous permet de diviser l'ensemble des vecteurs $\{0,1\}^{N+1}$ en trois sous-ensembles (voir Figure 2.1):

- l'ensemble des vecteurs entiers non-faisables (*i.e.* $y \notin Y_\Delta$);

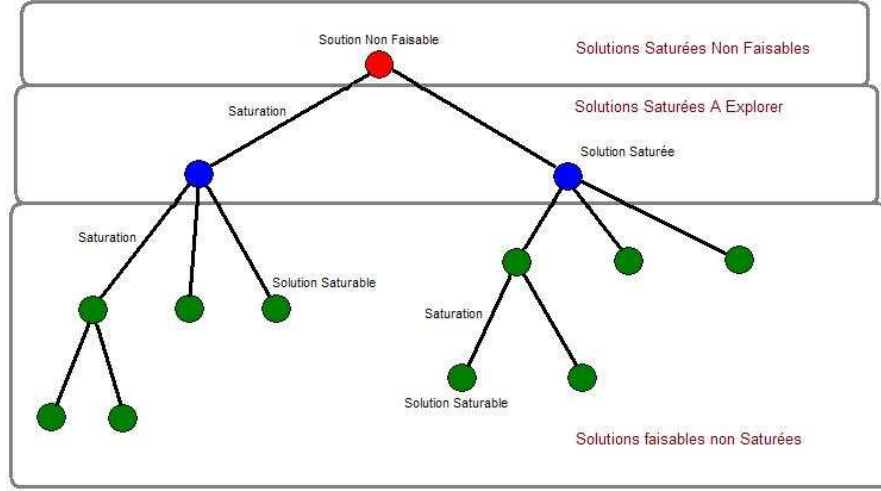


FIG. 2.1 – *Partition de l'ensemble des vecteurs entiers en trois sous-ensembles, en se basant sur la notion de saturation: les entiers non-faisables (en haut), les entiers saturés (au milieu) et les entiers non saturés (en bas). Seul l'ensemble des vecteurs saturés doit être exploré.*

- l'ensemble des vecteurs *saturés* (i.e. $y \in \bar{Y}_\Delta$); pour chaque vecteur appartenant à cet sous-ensemble le problème quadratique associé peut fournir la solution globale du problème de contrôle optimal discrétisé avec contraintes logiques;
- l'ensemble des vecteurs *non saturés* (i.e. $y \in Y_\Delta \setminus \bar{Y}_\Delta$); la solution optimale du problème à résoudre ne se trouve pas dans cet ensemble.

Avec cette division il est dans notre intérêt d'obtenir une caractérisation de l'ensemble \bar{Y}_Δ .

2.3 Réduction de l'espace de recherche

Une fois analysé le type de solution qu'on cherche, il nous reste à préciser la notion de "solution *saturée*", afin de l'intégrer dans notre problème d'optimisation.

Soit

$$\beta_i = \sum_{j=i}^{i+\Delta-1} y_j, \text{ pour } i = 0, N - \Delta + 1 \quad (2.5)$$

(β_i est la valeur de la contrainte i pour une solution y).

On va d'abord donner deux théorèmes (2.3.1 et 2.3.2) permettant de caractériser les solutions *saturées*.

Théorème 2.3.1 (Minorant pour chaque contrainte C_Δ). Si $y \in \bar{Y}_\Delta$ (donc y saturée) alors

$$\max(0, 2\Delta_0 - \Delta) \leq \beta_i \leq \Delta_0, \text{ pour } i = 0, N - \Delta + 1 \quad (2.6)$$

Le théorème suivant complète le précédent.

Théorème 2.3.2. Si $y \in \bar{Y}_\Delta$ (donc y saturée) alors:

- i) $\beta_0 = \Delta_0$ (première contrainte saturée)
- ii) $\beta_{N-\Delta+1} = \Delta_0$ (dernière contrainte saturée)

Afin de prouver les deux théorèmes précédemment énoncés, on va avoir besoin du lemme suivant.

Lemme 2.3.1. Soit:

- $y \in C_\Delta$ un vecteur entier saturé;
- $\beta_i(y)$ une contrainte non saturée, i.e. :

$$\beta_i(y) = y_i + y_{i+1} + \dots + y_{i+\Delta-1} < \Delta_0;$$

- r_i l'indice du n_i -ème zéro faisant partie de la contrainte $\beta_i(y)$ (en comptant avec le premier terme y_i de la somme), où $n_i = \Delta_0 - \beta_i(y)$;
- s_i l'indice n_i -ème zéro faisant partie de la contrainte $\beta_i(y)$ (en comptant à l'inverse à partir du dernier terme $y_{i+\Delta-1}$ de la somme),

alors $\beta_j(y) < \Delta_0$ pour chaque $j \in \{s_i - \Delta + 1, \dots, r_i\}$.

Démonstration. On remarque que la contrainte $\beta_i(y)$ contient $\Delta - \beta_i(y)$ zéros, ce qui veut dire que les indices r_i et s_i se trouvent à l'intérieur de la contrainte β_i .

Comme on peut le constater, la relation suivante est vérifiée:

$$\beta_{k+1}(y) = \beta_k(y) - y_k + y_{k+\Delta+1} \quad k \in \{1, 2, \dots, N - \Delta\}. \quad (2.7)$$

Cette relation implique que pour augmenter la valeur d'une contrainte β_{k+1} on doit enlever un 0 et récupérer un 1 dans l'équation (2.7):

$$\beta_{k+1}(y) = \beta_k(y) + 1 \Leftrightarrow y_k = 0 \text{ and } y_{k+\Delta+1} = 1. \quad (2.8)$$

Alors, afin d'obtenir une contrainte saturée en partant de la contrainte $\beta_i(y)$ on doit rencontrer au moins $\Delta_0 - \beta_i(y)$ zéros. Ce qui prouve que les contraintes $\beta_j(y)$, pour $j \in \{i, i+1, \dots, r_i\}$ ne sont pas saturées. Le même raisonnement, en obtenant la valeur de la contrainte $\beta_{k-1}(y)$ en fonction de $\beta_k(y)$ est appliqué pour démontrer que les contraintes $\beta_j(y)$, pour $j \in \{s_i - \Delta + 1, s_i - \Delta + 2, \dots, i\}$, ne sont pas saturées. \square

Démontrons maintenant les deux théorèmes énoncés auparavant.

Démonstration.

i) du théorème 2.3.2. Si la première contrainte $\beta_0(y)$ n'est pas saturée alors le premier 0 de cette contrainte ne fait pas partie d'aucune contrainte

$\beta_i(y) = \Delta_0$ pour tout $i \geq 0$ (en vertu du lemme). Ce qui implique que ce premier 0 ne fait pas partie d'aucune contrainte *saturée*; alors y n'est pas un vecteur *saturé* (car son premier 0 peut-être changé en 1 sans violer les contraintes C_Δ); ceci contredit l'hypothèse que y est un vecteur *saturé*.

ii) du théorème 2.3.2. Ce résultat s'obtient de façon analogue en raisonnant sur le dernier 0 de la dernière contrainte $\beta_{N-\Delta+1}(y)$.

Le théorème 2.3.1. Soit \mathcal{I} l'ensemble des zéros contenant les premiers $\Delta_0 - \beta_i(y)$ zéros de la contrainte $\beta_i(y)$ et \mathcal{J} l'ensemble des zéros contenant les derniers $\Delta_0 - \beta_i(y)$ zéros de la contrainte $\beta_i(y)$.

Supposons que l'ensemble $\mathcal{I} \cap \mathcal{J}$ n'est pas vide. Soit alors q_i l'indice d'un zéro ($y_{q_i} = 0$) se trouvant dans cet ensemble non vide $y_{q_i} \in \mathcal{I} \cap \mathcal{J}$. Alors

$$\begin{aligned} y_{q_i} \in \mathcal{I} &\Rightarrow q_i \leq r_i \\ y_{q_i} \in \mathcal{J} &\Rightarrow q_i \geq s_i \end{aligned}$$

et donc $s_i \leq q_i \leq r_i$. Les contraintes qui contiennent ce 0 sont: $\beta_{q_i-\Delta+1}(y), \dots, \beta_{q_i}(y)$. Comme $q_i - \Delta + 1 > s_i - \Delta + 1$ et $q_i \leq r_i$ alors le *lemme* montre qu'aucune de ces contraintes (contenant y_{q_i}) n'est *saturée*. Alors y n'est pas un vecteur entier *saturé* car $y_{q_i} = 0$ peut-être transformé en $y_{q_i} = 1$ sans violer les contraintes C_Δ . Cette contradiction implique que $\mathcal{I} \cap \mathcal{J} = \emptyset$. Alors le nombre total de zéros dans la contrainte $\beta_i(y)$ (qu'on va désigner par $z_i(y)$) est au moins $\text{card}(\mathcal{I}) + \text{card}(\mathcal{J})$. Donc:

$$z_i(y) \geq \text{card}(\mathcal{I}) + \text{card}(\mathcal{J}) = (\Delta_0 - \beta_i(y)) + (\Delta_0 - \beta_i(y)) = 2\Delta_0 - 2\beta_i(y). \quad (2.9)$$

Mais $z_i(y) = \Delta - \beta_i(y)$ et avec l'équation (2.9) on obtient:

$$\Delta - \beta_i(y) \geq 2\Delta_0 - 2\beta_i(y) \Rightarrow \beta_i(y) \geq 2\Delta_0 - \Delta,$$

ce qui prouve le théorème 2.3.1. \square

Les deux théorèmes énoncés auparavant et le fait que $|\beta_i - \beta_{i-1}| \leq 1$ nous permettent de reformuler les contraintes C_Δ qu'on ajoute dans toute modélisation mixte:

$$\begin{aligned} \Delta_0 &\leq y_0 + y_1 + \dots + y_{\Delta-1} && \leq \Delta_0 \\ \Delta_0 - 1 &\leq y_1 + y_2 + \dots + y_\Delta && \leq \Delta_0 \\ &\dots && \\ \max(0, -\Delta + 2\Delta_0) &\leq y_k + y_{k+1} + \dots + y_{k+\Delta-1} && \leq \Delta_0 \quad (2.10) \\ &\dots && \\ \Delta_0 - 1 &\leq y_{N-\Delta} + y_{N-\Delta+1} + \dots + y_{N-1} && \leq \Delta_0 \\ \Delta_0 &\leq y_{N-\Delta+1} + y_{N-\Delta+2} + \dots + y_N && \leq \Delta_0. \end{aligned}$$

Sans ambiguïté, on va désigner toujours par **contraintes** C_Δ le système obtenu en (2.10).

Remarque 2.3.1. En ajoutant ces contraintes dans les modélisations présentées au chapitre 1, on réduit l'espace de recherche Y_Δ . Il est toutefois

important de remarquer que les nouvelles contraintes C_Δ (2.10) ne sont pas suffisantes pour garantir que y est une solution saturée. La démarche de "saturation" donnée auparavant doit toujours être utilisée dans le cadre d'un algorithme afin de s'assurer qu'on analyse seulement les meilleures solution entières, celles qui sont saturées.

2.4 Élimination d'une solution déjà analysée

Afin d'éviter d'analyser à nouveau une solution (y^k) déjà rencontrée, la contrainte suivante peut être ajoutée:

$$y^T(e - 2y^k) \geq 1 - e^T y^k, \quad (2.11)$$

avec $e = (1, \dots, 1)^T$ (la contrainte 2.11 exprimant tout simplement $|y - y^k|^2 \geq 1$). Le désavantage de cette contrainte est qu'elle élimine seulement le vecteur entier y^k . Grâce à la notion de *saturation* on sait que si une solution *saturée* $y^k \in \bar{Y}_\Delta$ a été analysée alors on devrait aussi chercher à éliminer toute solution y^l qui se sature en y^k ; ceci car on a démontré que ces vecteurs entiers ne donnent pas une meilleure solution que celle obtenue à partir de y^k .

Le théorème suivant introduit cette notion de *coupe entière* éliminant l'ensemble de solutions se saturant dans un vecteur *saturé*, déjà exploré.

Théorème 2.4.1. *Si $y^k \in \bar{Y}_\Delta$ (donc y saturée) alors la contrainte suivante:*

$$y^T(e - y^k) \geq 1 \quad (2.12)$$

élimine seulement y^k et tout autre vecteur y se saturant en y^k .

Démonstration. Soient

$$I(k) = \{i \in \{0, \dots, N\} | y_i^k = 0\}$$

et

$$J(k) = \{j \in \{0, \dots, N\} | y_j^k = 1\},$$

(on a évidemment la relation $I(k) \cup J(k) = \{0, \dots, N\}$). Soit y^α un vecteur qui se sature en y^k . Alors $J(\alpha) \subset J(k)$, car la saturation d'un vecteur réduit son nombre de 0 et augmente son nombre de 1. Mais la relation (2.12) peut-être écrite comme:

$$\sum_{i \in \{0, \dots, N\}} y_i(1 - y_i^k) = \sum_{i \in I(k)} y_i(1 - y_i^k) + \sum_{i \in J(k)} y_i(1 - y_i^k) = \sum_{i \in I(k)} y_i. \quad (2.13)$$

En calculant la valeur de la contrainte (2.12) pour y^k et y^α on obtient:

$$(y^k)^T(e - y^k) = \sum_{i \in I(k)} y_i^k = 0 < 1$$

et

$$(y^\alpha)^T(e - y^k) = \sum_{i \in I(k)} y_i^\alpha \leq \sum_{i \in I(k)} y_i^k = 0,$$

ce qui implique que les vecteurs entiers y^k et y^α ne peuvent pas satisfaire la contrainte (2.12).

D'autre part, si y est un vecteur entier ne se saturant pas en y^k alors il existe une composante $y_{i_0} = 1$ et $y_k = 1$ ce qui implique que la contrainte (2.12) est satisfaite par le vecteur y . \square

2.5 Matrice totalement unimodulaire

On rappelle qu'une *matrice totalement unimodulaire* A est une matrice dont les éléments sont des 0,1 ou -1 et dont toute matrice carrée extraite de la matrice A a un déterminant égal à 0,1 ou -1. Ce type de matrice est très intéressant en optimisation combinatoire car toutes les solutions du système :

$$Ax \leq b, \text{ où } b \in \mathbb{Z}^m, \text{ et } A \in M_{m,n} \quad (2.14)$$

sont alors entières *cf.* [58].

Les matrices totalement unimodulaires ne représentent pas la classe la plus générale des matrices pour lesquelles le système antérieur a toutes ses solutions entières. On peut citer d'autres classes comme les matrices totalement balancées, par exemple. Toutefois, les résultats donnés dans cette section sont suffisants pour notre étude sur les conditions C_Δ . Pour une étude complète des polyèdres entiers le lecteur peut se rapporter à [58].

Théorème 2.5.1. *La matrice C_Δ est totalement unimodulaire et donc, tout sommet du polyèdre compact $\{y \in [0,1]^{N+1} | C_\Delta y \leq \Delta_0\}$ est entier.*

Démonstration: C_Δ est une matrice d'intervalle (sur chaque ligne, les 1 apparaissent de façon successive), donc elle est totalement unimodulaire [58]. La deuxième partie du théorème est une conséquence directe de la totale unimodularité de la matrice.

Le théorème précédent nous permet d'obtenir le résultat suivant:

Théorème 2.5.2. *Tout sommet solution optimale du problème linéaire*

$$\begin{cases} \min_{y \in [0,1]^n} & c^T y \\ \text{s.t.} & C_\Delta y \leq \Delta_0 \end{cases} \quad (2.15)$$

est une solution optimale du problème mixte (MILP):

$$\begin{cases} \min_{y \in \{0,1\}^n} & c^T y \\ \text{s.t.} & C_\Delta y \leq \Delta_0. \end{cases} \quad (2.16)$$

Démonstration: Le premier problème est évidemment la relaxation continue du deuxième. Étant donné que C_Δ est totalement unimodulaire, tous les sommets du polyèdre $C_\Delta y \leq \Delta_0$ sont entiers. Sachant que la solution du problème d'optimisation se trouve parmi ses sommets, il en résulte que la solution du problème relaxé est entière et donc cette solution optimale est aussi la solution optimale du problème mixte **MILP**. \square

On démontre ainsi que les contraintes C_Δ ont une structure particulière, tout problème MILP ayant comme matrice des contraintes C_Δ peut se résoudre comme un simple problème LP. On va revenir sur l'importance de ce théorème dans le chapitre présentant les résultats.

2.6 Conclusions

La structure particulière du notre problème et des contraintes logiques nous a permis de faire un premier pas important pour la résolution du problème mixte. On a donné une caractérisation des vecteurs entiers qui peuvent-être la solution entière du problème à résoudre (notion de saturation), on a montré une façon efficace d'éliminer les solution analysées et celles qui, quoique non analysées, ne sont pas meilleures et on a fourni une description de l'espace de recherche (les théorèmes 2.3.1 et 2.3.1). Ces propriétés vont être exploitées dans le chapitre suivant.

Chapitre 3

MINLP associé à une dynamique linéaire

Ce chapitre propose une méthode de résolution numérique des problèmes **MINLP** issus de problèmes de contrôle optimal à dynamique linéaire, coût quadratique, et intégrant les contraintes logiques C_Δ . Ce cas mérite d'être distingué du cas général des problèmes MINLP car la structure particulière des contraintes prises en compte nous permet de développer des algorithmes très spécifiques, plus performants du point de vue numérique.

On va commencer par présenter les algorithmes du domaine mixte, **GBD**, **GOA**, **BB**, en mettant en évidence les principes fondamentaux et surtout les domaines d'applicabilité de chaque algorithme (cette partie est un passage important car ces algorithmes peuvent s'appliquer à la modélisation **MIQP** du notre problème, modélisation introduite dans le chapitre précédent).

On poursuit par l'analyse des modélisations qu'on a proposées au premier chapitre. Le cas **MIQP**, *a priori* plus facile, est traité avec les algorithmes existants, modifiés pour intégrer les particularités des contraintes C_Δ (analysées dans le chapitre précédent). On présente les résultats obtenus sur un problème test et on discute les limites de la démarche.

La section suivante de ce chapitre est consacrée au deuxième modèle **MINLP** associé à notre problème de contrôle optimal. Malgré la formulation générale en tant que problème **MINLP**, une particularité importante va nous permettre de développer une variante de l'algorithme **GBD** ; après avoir présenté cette démarche on comparera les résultats obtenus pour les deux modèles (**MIQP** et **MINLP**). La fin du chapitre présente aussi les résultats obtenus sur des problèmes plus complexes, en analysant plus en détail le cas de l'oscillateur harmonique à une dimension et en présentant la forme des solutions obtenues pour d'autres problèmes (ajout des forces de frottement dans la dynamique du mobile unidimensionnel, un modèle bidimensionnel, etc).

3.1 Rappels relatifs à la programmation mixte

Cette section s'intéresse aux algorithmes et aux concepts de la programmation en variables mixtes. Cette partie donne seulement un survol des techniques et notions employées dans le domaine mixte, en mettant l'accent sur les algorithmes qu'on va utiliser par la suite (**GBD**, **GOA**, **BB**); pour une description détaillée, on peut consulter, par exemple, [27].

3.1.1 Formulation générale du MINLP

On commence par donner la modélisation générale d'un problème en variables mixtes en utilisant les mêmes notations, x est un vecteur qui rassemble les variables continues et y est le vecteur des variables discrètes.

Dans sa forme la plus générale, le problème **MINLP** peut s'écrire comme suit:

$$\left\{ \begin{array}{ll} \min_{x,y} & f(x,y) \\ s.t. & h(x,y) = 0 \\ & g(x,y) \leq 0 \\ & x \in X \subseteq \mathbb{R}^n \\ & y \in Y = \{0,1\}^m \end{array} \right. \quad (3.1)$$

où X est un hypercube de \mathbb{R}^n et $y \in Y \subset \mathbb{Z}^m$ le vecteur des variables discrètes, $h(x,y) = 0$ désigne les contraintes d'égalité, $g(x,y) \leq 0$ les contraintes d'inégalité et $f(x,y)$ représente le coût à minimiser. On remarquera que tout problème **MINLP** avec $Y \in \mathbb{Z}^m$ et Y borné peut se mettre sous la forme (3.1).

Afin d'illustrer les différentes démarches employées pour résoudre le problème (3.1), il convient de mettre ce problème sous la forme:

$$\begin{array}{ll} \min_y & v(y) \\ s.t. & y \in V \cap Y \end{array} \quad (3.2)$$

avec $v(y)$ définit par:

$$\begin{array}{ll} v(y) = \min_x & f(x,y) \\ s.t. & g(x,y) \leq 0 \\ & h(x,y) = 0 \\ & x \in X \subseteq \mathbb{R}^n, \end{array} \quad (3.3)$$

où V est l'ensemble $y \in Y$ pour lesquelles il existe une solution faisable du problème associé $P(y)$

$$V = \{y \in Y : \exists x \in X \text{ t.q. } g(x,y) \leq 0 \text{ et } h(x,y) = 0\}. \quad (3.4)$$

Remarque 3.1.1. Le problème (3.2) est la projection du problème (3.1) sur l'espace des y . D'autre part $v(y)$ et V ne sont connues que de façon implicite et il va falloir trouver une représentation explicite de ces deux ensembles.

Parmi les algorithmes existants [27] on va se concentrer sur trois d'entre eux: l'Approximation Extérieure (**GOA**, Generalized Outer Approximation, cf. [22], [25], [34]), la Décomposition de Benders (**GBD**, Generalized Benders Decomposition, cf. [28]) et l'algorithme d'Évaluation et Séparation (**BB**, Branch and Bound, cf. [5], [8]).

En l'absence de conditions nécessaires d'optimalité en programmation mixte analogues à celles existant en programmation mathématique classique, les algorithmes utilisent comme principe général de résolution numérique la minoration et la majoration de la solution optimale. A chaque itération, on génère un minorant et un majorant de la solution optimale; ces limites sont améliorées d'une itération à l'autre en arrêtant l'algorithme quand elles sont suffisamment proches. La principale différence entre les algorithmes réside dans la façon dont la majoration et la minoration est faite. Il convient toutefois de distinguer les algorithmes **GOA** et **GBD** de l'algorithme **BB**, les premiers utilisant le principe de **primal-master** (cf. sections suivantes), tandis que le dernier est plutôt une démarche générale, à adapter au cas par cas.

3.1.2 Approximation extérieure

L'algorithme de l'approximation extérieure a été proposé par Duran et Grossmann [22] et généralisé par Fletcher et Leyffer [25], [34] afin de prendre en compte les problèmes qui, pour un certain $y = y^k$ fixé, sont non-réalisables (n'admettent pas de point réalisable). Cet algorithme s'intéresse à la classe de problèmes mixtes ayant des contraintes d'égalité linéaires en (x, y) et des contraintes d'inégalité convexes en x, y et il est basé sur une approximation extérieure de l'ensemble V .

A l'itération k , on résout successivement les deux problèmes suivants:

- le problème (appelé **primal**), obtenu en fixant les variables binaires $y = y^k$:

$$P(y^k) \quad \begin{cases} \min_x & f(x, y^k) \\ s.t. & g(x, y^k) \leq 0 \\ s.t. & h(x, y^k) = 0 \\ & x \in X \subseteq R^n \end{cases} \quad (3.5)$$

Remarquons tout de suite que, dans notre cas, $P(y^k)$ est un problème **NLP** convexe (objectif quadratique et contraintes linéaires) et sa solution globale (x^k) peut être obtenue avec les algorithmes classiques. D'autre part si ce problème est réalisable, son objectif optimal est un majorant de l'objectif optimal du problème **MINLP**. Si le problème

primal est non-réalisable, alors un problème dit de "faisabilité" est formulé et sa solution optimale x^k obtenue [33].

- le problème (appelé **master**) qui est une relaxation du problème mixte en utilisant l'approximation extérieure:

$$\left\{ \begin{array}{ll} \min_{x,y,\mu} & \mu \\ \text{s.t.} & \mu \geq f(x^k, y^k) + \nabla f(x^k, y^k)^T \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} \quad k \in F \\ & 0 \geq g(x^k, y^k) + \nabla g(x^k, y^k)^T \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} \quad k \in F \\ & 0 \geq g(x^k, y^k) + \nabla g(x^k, y^k)^T \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} \quad k \in \bar{F} \\ & 0 = h(x, y) \\ & x \in X \\ & y \in Y, \end{array} \right. \quad (3.6)$$

avec F , l'ensemble des indices d'itérations pour lesquelles le problème **primal** est réalisable et \bar{F} , l'ensemble des indices d'itérations pour lesquelles le problème **primal** est non-réalisable.

Le rôle de ce problème **master** est, d'une part, de fournir un minorant de la valeur optimale du problème **MINLP** et, d'autre part, de fournir le vecteur entier y^{k+1} (pour lequel un problème **primal** va être résolu à l'itération suivante).

3.1.3 La Décomposition de Benders

L'algorithme de Benders (généralisé par Geoffrion en [28]) utilise le même principe **primal-master** mais en donnant une représentation duale de la fonction $v(\cdot)$. Le problème **primal** est obtenu de façon identique à celui de l'algorithme de Fletcher et Leyffer (problème 3.5), y compris le problème de non-faisabilité. On considère en plus que la solution duale optimale du problème **primal** (λ^k, μ^k si réalisable et $\bar{\lambda}^k, \bar{\mu}^k$ si non-réalisable) est disponible. Quant au **master**, à l'itération k , celui-ci s'écrit [27]:

$$(M_{GBD}) \left\{ \begin{array}{ll} \min_{y \in Y, \mu_B} & \mu_B \\ \text{s.t.} & \mu_B \geq \xi(y; \lambda^k, \mu^k) \quad \text{problème } \mathbf{primal} \text{ faisable} \\ & 0 \geq \bar{\xi}(y; \bar{\lambda}^k, \bar{\mu}^k), \quad \text{problème } \mathbf{primal} \text{ non-faisable,} \end{array} \right. \quad (3.7)$$

où ξ (respectivement $\bar{\xi}$) sont les fonctions support (fonctions duales, au sens du [39]) du problème **primal** faisable (respectivement **primal** non-faisable).

Le cas où l'algorithme de Benders peut-être appliqué (du point de vue pratique) est conditionné par l'obtention de l'expression explicite de la fonction duale, d'une part, et de l'obtention d'un problème **master** qu'on sait

résoudre de façon efficace. Plusieurs de ces cas ont été identifiés (voir par exemple [27]): les fonctions f, g et h sont linéairement séparables en x et y , le cas de la programmation par factorisation, etc.

3.1.4 La démarche Évaluation et Séparation

L'algorithme de Branch and Bound donne, sous certaines conditions, une démarche très générale de résolution. Son objectif principal est d'énumérer les alternatives d'un problème, sans en explorer toutes les instances possibles. Pour cela, il utilise les concepts de séparation, relaxation et élimination; une autre notion importante est la représentation des alternatives possibles comme un arbre binaire. Il faut préciser que les concepts de l'algorithme peuvent être appliqués non seulement aux problèmes mixtes (là où l'arbre est construit en prenant en compte les variables binaires) mais aussi aux problèmes non-convexes en variables continues; ce dernier cas est un Branch and Bound spatial, l'arbre binaire étant construit en divisant le domaine de définition des variables continues, domaine induit par les contraintes de boîte.

Séparation. Étant donné un problème on transforme celui-ci en deux problèmes différents dont la résolution équivaut à résoudre le problème de départ. Dans le cas mixte avec des variables binaires, on obtient ces deux problèmes en donnant les valeurs 0 et 1 à l'une des variables binaires. En appliquant de façon récursive cette démarche on obtient un arbre de problèmes à résoudre (chaque nœud de l'arbre contenant un problème). La force de l'algorithme consiste à résoudre seulement une partie infime des nœuds, les autres problèmes étant éliminés.

Relaxation. Le rôle de la relaxation est de fournir un minorant de la solution exacte du problème se trouvant dans un nœud. Dans la plupart des cas on relâche des contraintes d'où le nom de relaxation. L'intérêt de la relaxation est d'obtenir un minorant du problème contenu dans un nœud sans avoir à résoudre le problème de façon exacte (car ce problème est à son tour un problème mixte, donc NP-difficile).

Remarque 3.1.2. La relaxation n'est pas la seule technique employée pour obtenir un minorant, au cas par cas on fait une sous-estimation en essayant de combiner au mieux les contraintes suivantes (souvent antagonistes):

- l'obtention du minorant doit être non coûteuse en temps de calcul;
- le minorant doit être proche de la vraie valeur de l'objectif pour le nœud estimé.

Élimination L'élimination d'un nœud (et du sous-arbre associé) se fait quand on est dans un des cas suivants:

- le minorant du nœud est plus grand que le meilleur majorant déjà trouvé (on comprend donc l'importance d'un bon minorant);
- le problème associé au nœud est non-réalisable;
- la solution du nœud en question est déjà entière;

Quoique le cas le plus général de l'application du **BB** soit le problème **MINLP** (sous la réserve qu'un moyen de trouver une bonne estimation inférieure ait été trouvé), il est généralement utilisé de façon très efficace pour résoudre les problèmes **MILP** ou **MIQP**.

3.2 Modèle MIQP. Algorithmes et résultats

La section qui propose un survol des algorithmes MINLP nous montre que la résolution des problèmes mixtes non-convexes est très difficile. Comme on l'a déjà vu, il n'existe pas d'algorithme générique pour trouver la solution d'un tel problème. En fait, introduire des non-convexités dans un problème empêche en général de trouver une limite inférieure de l'optimum ce qui entraîne l'absence de condition d'arrêt de l'algorithme. On essaye ici de résoudre une classe de problèmes avec les conditions C_Δ en utilisant nos deux modèles.

On rappelle la forme du problème de contrôle optimal sur lequel on intègre les contraintes logiques:

$$\left\{ \begin{array}{l} \min_{x \in X, u \in U} \quad \frac{1}{2} \int_0^{t_f} |u(t)|^2 dt \\ s.t. \quad \dot{x}(t) = A(t)x(t) + B(t)u(t) + C(t) \quad , t \in [0, t_f] \\ x(0) = x^0 \\ x(t_f) = x^f \end{array} \right. \quad (3.8)$$

En discrétisant (3.8) par la méthode de la collocation directe on aboutit au système suivant:

$$\left\{ \begin{array}{ll} \min_{x, u} \quad \frac{1}{2} u^T u & \\ s.t. \quad Ax = b & \text{(Conditions aux deux bouts)} \\ \quad \quad Cx + Du = d & \text{(Equations de la dynamique)} \\ \quad \quad -U_{max} \leq u_i \leq U_{max} \quad , i = \overline{0, N} & \text{(Contrôle limite)} \end{array} \right. \quad (3.9)$$

On va analyser par la suite les algorithmes disponibles pour résoudre la première modélisation, voir les résultats obtenus et les limites d'une telle approche.

3.2.1 Démarche

On commence l'étude de ces problèmes en utilisant la modélisation suivante, dont l'obtention a déjà été décrite en §1.4 :

$$\left\{ \begin{array}{ll} \min_{x,u,y} & \frac{1}{2}u^T u \\ s.t. & Ax = b \\ & Cx + Du = d \\ & C_{\Delta}y \leq \Delta_0 \\ & -U_{max}y_i \leq u_i \leq U_{max}y_i, \quad i = \overline{0,N} \\ & x \in X, u \in U, y \in \{0,1\}^{N+1}, \end{array} \right. \quad (3.10)$$

L'intégration des contraintes logiques a été faite en couplant les contraintes sur la poussée maximale du contrôle avec le rôle des variables binaires.

Faisant partie de la classe des problèmes **MIQP** (objectif quadratique et contraintes linéaires), les algorithmes **BB** (Branch and Bound), **GOA** (Approximation extérieure) et **GBD** (Décomposition de Benders) peuvent s'appliquer. Le choix habituel fait par les logiciels d'optimisation (commerciaux ou non) est l'algorithme du **Branch and Bound**. Le **GOA** est une autre option car le problème est convexe en x, u et y (en faisant abstraction du domaine non-convexe des variables binaires). Finalement, le **GBD** est aussi un choix possible, car le problème (3.10) est linéairement séparable en variables continues (x, u) et variables discrètes y .

Avant de présenter les résultats obtenus en appliquant ces trois algorithmes sur un problème de test, voyons comment les propriétés particulières des contraintes C_{Δ} et du problème quadratique vont s'intégrer dans les algorithmes **GOA** et **GBD**.

La première observation qu'on peut faire concerne le problème **primal**. Celui-ci est identique dans les deux algorithmes et, dans notre cas, est un problème quadratique. D'autre part, seulement les problèmes **primaux** associés aux vecteurs entiers *saturées* sont analysés. Comme on n'a aucune garantie que la solution du problème **master** est déjà *saturée*, la procédure algorithmique donnée en §2.2 est utilisée sur la solution optimale entière du **master** afin de la saturer.

Une autre particularité est liée aux problèmes **primaux** non-faisables. **Pour l'algorithme GOA**, le problème **master** contient naturellement les contraintes issues de la dynamique du système; ceci a comme conséquence que **tout problème primal résolu pour une solution y du master est faisable**. Même en prenant en compte le fait qu'on sature la solution du **master**, le problème reste faisable. Donc, aucun traitement concernant la non-faisabilité n'est fait dans l'algorithme de GOA. Pour l'algorithme **GBD**, le traitement habituel, qui consiste à introduire un problème de faisabilité est pris en compte. Ceci nous amène à la question suivante: que faire si le premier problème (pour y^0) est non-faisable? Comme ce problème n'est pas issu d'un quelconque **master** on ne peut pas garantir sa faisabilité (pour l'algorithme **GOA**). Pour répondre à cette question on propose deux façons

distinctes :

i) On commence par résoudre le problème suivant:

$$\left\{ \begin{array}{ll} \max_{y \in \{0,1\}^{N+1}} & \sum_{i=0}^{i=N} y_i \\ \text{s.t.} & C_{\Delta} y \leq \Delta_0. \end{array} \right. \quad (3.11)$$

Comme prouvé dans la section §2.5, la totale unimodularité de la matrice C_{Δ} permet de résoudre le problème (3.11) aussi simplement qu'un problème de programmation linéaire. D'autre part, vu la forme de l'objectif on peut facilement prouver que la solution entière y^0 du problème (3.11) est toujours *saturée*. Si le problème **primal** associé à cette solution entière n'est pas faisable, alors la coupe entière (2.12) est ajoutée à (3.11) et ce problème résolu à nouveau. Cette procédure est répétée jusqu'au moment où un premier vecteur entier faisable est trouvé.

ii) La démarche précédente peut être coûteuse si beaucoup des vecteurs entiers sont non-faisables. Une autre méthode consiste à résoudre la relaxation continue du problème avec contraintes logiques, *i.e.* le problème (3.9). Si celui-ci est non-faisable, alors le problème intégrant les contraintes logiques est non-faisable, lui aussi. Sinon, dans le cas d'une relaxation faisable, la solution (x^0, u^0) est utilisée pour construire le premier **master** dans l'algorithme **GOA**.

Concernant le problème **master**, toujours pour l'algorithme **GOA**, celui-ci ne nécessite que la linéarisation de l'objectif, seule fonction non-linéaire intervenant dans notre problème mixte. Quant au **master** du **GBD**, celui-ci est construit en utilisant la théorie générale de la Décomposition de Benders [27],[28].

Finalement, afin d'accélérer les algorithmes, la *coupe entière* (2.12) est ajoutée à chaque itération au problème **master**. Il est intéressant de voir que cette contrainte à aussi pour rôle de garantir la terminaison des algorithmes modifiés. Elle permet d'éviter que la situation suivante ne se produise: à l'itération k de l'algorithme \bar{y}^{k+1} est obtenue comme solution du **master**; ce vecteur est saturé en y^{k+1} et le **primal** associé est résolu; le **master** est ensuite résolu et \bar{y}^{k+2} , sa solution entière, est saturée; mais en saturant on obtient toujours y^{k+1} et l'algorithme ne termine pas. Cette situation ne peut pas arriver car la coupe entière qu'on propose garantit que si \bar{y}^{k+2} se sature toujours en y^{k+1} alors elle est éliminée par la coupe entière associée au vecteur y^{k+1} .

3.2.2 Résultats

Problème de test. Le problème de test choisi est celui du mobile contraint à se déplacer dans un espace à une dimension, en partant d'une position et ayant une vitesse initiale donnée et qui doit être ramené au repos à l'origine.

En explicitant les équations de la collocation, on obtient le problème **MIQP** suivant:

$$\left\{ \begin{array}{ll} \text{Min} & \frac{1}{2}u^T u \\ \text{s.t.} & -2x_i + 2x_{i+1} - 2hx_{i+N} - h^2u_i = 0 & i = \overline{0, N-1} \\ & 10x_i - 10x_{i+1} + 4hx_{i+N} + 6hx_{N+i+1} - h^2u_{i+1} = 0 & i = \overline{0, N-1} \\ & -y_i U_{max} \leq u_i \leq y_i U_{max} & i = \overline{0, N} \\ & x_0 = x^0, x_N = x^f, x_N = v^0, x_{2N-1} = v^f \\ & C_{\Delta} y \leq \Delta_0 \end{array} \right. \quad (3.12)$$

Résultats. Les algorithmes (**BB**, **GBD**, **GOA**) ont été implémentés en utilisant la librairie d'optimisation CPLEX en langage **C** (une autre implémentation en utilisant la librairie OSL d'IBM a aussi été testée avec des résultats très semblables en ce qui concerne les temps de calcul).

Les résultats présentés dans le Tableau 3.1 suivant ont été obtenus sur un PC Pentium III, 933 MHz, 512M RAM:

TAB. 3.1 – *Résultats utilisant le modèle **MIQP**. Les notations suivantes ont été utilisées pour la légende: N est le nombre de points de discrétisation (il représente aussi le nombre de variables binaires), Δ, Δ_0 sont les paramètres des contraintes C_{Δ} , N_c est le nombre de variables continues, N_{con} dénote le nombre total de contraintes, t_{BB} , t_{GBD} , t_{GOA} sont les temps d'exécution des algorithmes **BB**, **GBD**, **GOA** exprimés en secondes et finalement (*) signifie que l'algorithme n'a pas convergé après une heure d'exécution.*

N	Δ	Δ_0	N_c	N_{con}	$t_{B\&B}(s)$	$t_{GBD}(s)$	$t_{GOA}(s)$
8	5	3	24	22	0.06	0.89	0.69
10	7	4	30	26	0.11	4.82	4.21
15	10	7	45	36	0.24	234.79	115.58
20	15	10	60	48	0.52	*	*
30	20	10	90	73	16.52	*	*
40	30	15	120	96	7.76	*	*
40	25	15	120	93	34.57	*	*
40	25	10	120	96	447.18	*	*
40	20	10	120	93	1009.04	*	*
50	15	10	150	108	428.97	*	*
50	30	20	150	113	43.88	*	*
50	40	20	150	123	144.74	*	*
50	40	30	150	113	4.54	*	*

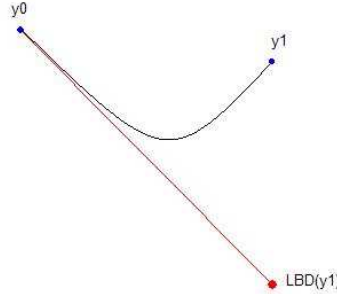


FIG. 3.1 – *Estimation inférieure dans la **GBD** et la **GOA**. On sous-estime la solution en y^1 en utilisant la solution en y^0 . La limite inférieure ainsi obtenue est très mauvaise.*

Analyse. Comme on peut le constater, seuls les problèmes très simples, ayant un petit nombre de variables entières, ont pu être résolus dans un temps raisonnable. Ces résultats sont la conséquence d'un mauvais mino- rant (**LBD**), dont l'explication est donnée dans le paragraphe suivant pour chaque algorithme traité :

BB. Dans le **BB**, le minorant est calculé comme la relaxation du problème mixte en problème quadratique. Or cette relaxation est trop forte, la solution de ce problème étant loin de la solution du problème mixte.

D'autre part, l'arbre du **BB** est très inadapté à notre problème : le nœud père est le même qu'un des deux nœuds fils, obtenus de lui ; un calcul supplémentaire (et inutile) est fait à chaque itération sans gagner beaucoup en efficacité.

Un autre inconvénient est celui qu'un algorithme de **BB** est de plus en plus gourmand en mémoire : ce qui signifie que plus le temps passe en exécution, moins il va vite à cause du nombre importants de nœuds se trouvant en mémoire.

GOA et GBD. On peut montrer dans ces cas que l'algorithme suit une direction de descente (de même type qu'un algorithme SQP, par exemple) : à l'itération k , on évalue $f(y^k)$ et on essaie de trouver un pas tel que $f(y^{k+1}) \leq f(y^k)$; mais dans un algorithme SQP ce pas est obtenu suite à un problème d'optimisation continue, tandis que dans notre cas il s'agit d'un problème mixte (le mastère) ; ayant moins de degrés de liberté, on obtient un pas trop grand et l'estimation faite est mauvaise, ce qui explique la mauvaise minoration, insuffisante pour donner un bon taux d'élimination des solutions entières (voir figure 3.1).

Soulignons toutefois que ce cas arrive suite à notre modélisation et ce n'est pas une caractéristique des algorithmes **GOA**, **GBD** (on est tout simplement dans un cas extrême).

La seule observation positive est celle que, dans la plupart des cas, après un nombre petit d'itérations, l'algorithme trouve un bon optimum local, il passe ensuite son temps à essayer de prouver qu'il est l'optimum global; on peut envisager d'utiliser ces algorithmes comme des méthodes heuristiques pour trouver un sous-optimum efficace.

Pour synthétiser, le **BB** est le meilleur algorithme, mais ces résultats ne nous permettent pas de dépasser la barrière de 60 variables entières, ce qui est loin d'être une bonne discrétisation pour des problèmes plus complexes.

Variations de la GBD

En essayant d'améliorer les performances des algorithmes on a concentré nos efforts sur l'algorithme de Benders.

Variations des multiplicateurs de Lagrange. Dans la **GBD** avec la séparabilité les fonctions supports sont obtenues à partir de la fonction duale du problème à estimer, cette fonction duale étant évaluée en (λ^k, μ^k) , les multiplicateurs du problème résolu:

$$\varphi(y; \lambda^k, \mu^k) = \min_{x \in X} L(x, y, \lambda^k, \mu^k) \quad (3.13)$$

Une des modifications qu'on a testées consiste à varier ces multiplicateurs avant de les introduire dans la fonction duale:

$$\begin{aligned} \overline{\lambda}_k &= \lambda_k(1 + \pi) \\ \overline{\mu}_k &= \mu_k(1 + \pi) \end{aligned}$$

et la limite inférieure est obtenue de la même façon:

$$\varphi(y; \overline{\lambda}^k, \overline{\mu}^k) = \min_{x \in X} L(x, y, \overline{\lambda}^k, \overline{\mu}^k) \quad (3.14)$$

Résultats. Seulement dans certains cas, des améliorations mineures ont été observées: à cause de la modélisation, les problèmes duaux associés à deux solutions entières distinctes sont très différentes.

Combinaison convexe des multiplicateurs de Lagrange Comme dans le paragraphe précédent, au lieu d'introduire les multiplicateurs de Lagrange tels quel dans la fonction duale on utilise la perturbation suivante:

$$\begin{aligned} \overline{\lambda}_k &= \sum_{i=1}^k \alpha_i \lambda_i \\ \overline{\mu}_k &= \sum_{i=1}^k \alpha_i \mu_i \end{aligned}$$

avec $\sum_{j=1}^k \alpha_j = 1$. Les nouveaux "multiplicateurs" sont obtenus comme des combinaisons convexes des multiplicateurs obtenus aux itérations $1, 2, \dots, k$. Le fait de choisir des combinaisons convexes et non pas simplement linéaire tient dans l'application de la **GBD**. Toutefois, une difficulté supplémentaire est introduite: à chaque itération la détermination des paramètres α se fait à travers un problème d'optimisation non-linéaire SQP.

On n'entre pas dans les détails de cet algorithme car les résultats obtenus en l'appliquant ne sont guère meilleurs que ceux obtenus en appliquant l'algorithme standard de Benders.

3.3 Modèle MINLP. Algorithme et résultats

On rappelle le modèle MINLP obtenu, problème qu'on désignera par la suite par (P) :

$$(P) \left\{ \begin{array}{ll} \min_{x,u,y} & \frac{1}{2} \sum_{i=0}^{i=N} u_i^2 \\ \text{s.t.} & Ax + B(u \odot y) = b \\ & Dx = d \\ & C_{\Delta} y \leq \Delta_0 \\ & -u_{\max} \leq u_i \leq u_{\max}, \quad i = \overline{0, N} \\ & x \in \mathbb{R}^{N+1}, y \in \{0, 1\}^{N+1}. \end{array} \right. \quad (3.15)$$

Dans cette section on utilisera les notations suivantes:

- $(P(y))$ est le problème d'optimisation obtenu en fixant y dans le problème (P) ; il s'agit du problème **primal**;
- le Lagrangien associé au problème **primal** s'écrit:

$$\begin{aligned} \mathcal{L}(y; x, u, \lambda^1, \lambda^2, \mu^1, \mu^2) = & \frac{1}{2} \sum_{i=0}^{i=N} u_i^2 + \\ & (\lambda^1)^T (Dx - d) + (\lambda^2)^T (Ax + B(u \odot y) - b) \\ & + (\mu^1)^T (u - u_{\max}) - (\mu^2)^T (u + u_{\max}); \end{aligned} \quad (3.16)$$

- la fonction support ou fonction duale, $\xi(y; \cdot, \cdot, \cdot, \cdot)$, associée au problème $(P(y))$ s'obtient à partir du Lagrangien:

$$\xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2) = \inf_{x, u} \mathcal{L}(y; x, u, \lambda^1, \lambda^2, \mu^1, \mu^2). \quad (3.17)$$

On utilisera aussi les hypothèses suivantes afin d'alléger les démonstrations qui suivent:

- (H1)** Pour chaque $u \in U \triangleq \{u \mid -u_{\max} \leq u_i \leq u_{\max}, i = 0, 1, \dots, N\}$ les contraintes du problème (3.15) définissent un unique $x \in \mathbb{R}^{n(N+1)}$.

On peut facilement prouver [56] que la discrétisation faite en utilisant la collocation directe permet de vérifier cette hypothèse pour le type de problèmes auxquels on s'intéresse;

(H2) *Pour chaque vecteur entier saturé y le système respectant la dynamique discrétisée est contrôlable (i.e. il existe un contrôle admissible u capable d'amener le système de l'état initial x^0 à l'état final x^f). Comme cette hypothèse ne peut-être facilement vérifiée en pratique, on montrera dans la section dédiée à l'algorithme comment on peut s'en affranchir. Pour l'instant on suppose l'hypothèse vérifiée afin d'illustrer la démarche générale sans avoir à s'inquiéter des problèmes non-faisables.*

Les propriétés suivantes sont faciles à vérifier pour le problème (P) auquel on s'intéresse et elles jouent un rôle important dans l'application de l'algorithme **GBD**:

- le problème **primal** $P(y)$ est un problème d'optimisation *convexe* quadratique; à partir de la formulation (3.15), et sous les hypothèses **(H1)** et **(H2)** il est facile de vérifier que $(P(y))$ a une solution unique (x_y, u_y) et que son objectif optimal $val(P(y))$ est fini;
- en prenant en compte la propriété précédente et en appliquant la théorie de la dualité, on a:

$$\begin{aligned} val(P(y)) &= \inf_{x,u} \sup_{\lambda^1, \lambda^2, \mu^1, \mu^2} \mathcal{L}(y; x, u, \lambda^1, \lambda^2, \mu^1, \mu^2) = \\ &= \sup_{\lambda^1, \lambda^2, \mu^1, \mu^2} \inf_{x,u} \mathcal{L}(y; x, u, \lambda^1, \lambda^2, \mu^1, \mu^2) = \\ &= \mathcal{L}(y; u_y, x_y, \lambda_y^1, \lambda_y^2, \mu_y^1, \mu_y^2), \end{aligned} \quad (3.18)$$

où x_y, u_y dénotent la solution optimale du $P(y)$ et $\lambda_y^1, \lambda_y^2, \mu_y^1, \mu_y^2$ sont les multiplicateurs optimaux pour ce même problème $P(y)$. Ces multiplicateurs représentent aussi la solution optimale du problème dual $D(y)$:

$$D(y) \left\{ \begin{array}{ll} \max_{\lambda^1, \lambda^2, \mu^1, \mu^2} & \xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2) \\ s.t. & \lambda^1 \in \mathbb{R}^p, \lambda^2 \in \mathbb{R}^q \\ & \mu^1 \in \mathbb{R}_+^{N+1}, \mu^2 \in \mathbb{R}_+^{N+1}, \end{array} \right. \quad (3.19)$$

où p est le nombre des conditions initiales et finales et q est le nombre d'équations de collocation.

L'algorithme GBD (cf. [27], [28]) génère, à chaque itération k , un mino- rant et un majorant pour la solution optimale du (P) :

- la limite supérieure nécessite la résolution du problème $(P(y^k))$; de cette façon on obtient aussi les multiplicateurs optimaux associés au problème $(P(y^k))$ $(\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k$;

- la limite inférieure est obtenue en résolvant le problème **master** suivant:

$$(PM^k) \begin{cases} \min_{y, \eta} & \eta \\ \text{s.t.} & \xi(y; (\lambda^1)^i, (\lambda^2)^i, (\mu^1)^i, (\mu^2)^i) \leq \eta \quad , i \in \mathcal{I}_k \\ & y^T(e - y^l) \geq 1 \quad \quad \quad l = 1, \dots, k \\ & C_{\Delta} y \leq \Delta_0 \\ & y \in \{0, 1\}^{N+1}, \end{cases} \quad (3.20)$$

où \mathcal{I}_k est l'ensemble des indices pour lesquels le problème **primal** correspondant a été résolu, ce **primal** est faisable et ses multiplicateurs optimaux sont connus $((\lambda^1)^i, (\lambda^2)^i, (\mu^1)^i$ et $(\mu^2)^i$).

Le mastère donné par (3.20) est le mastère standard de l'algorithme **GBD** auquel on a ajouté notre type particulier de coupes entières.

3.3.1 Algorithme. Convergence

Afin de prouver que l'algorithme converge on va prouver, dans l'ordre que :

- pour chaque k et entier faisable y , $\xi(y; (\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k)$ est finie, *i.e.*:

$$\xi(y; (\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k) > -\infty ;$$

- pour chaque k la valeur optimale de l'objectif dans le problème **master** est finie et fournit un minorant non trivial de l'objectif optimal du problème (P), *i.e.*:

$$-\infty < \text{val}(PM^k) \leq \text{val}(P).$$

On commence par expliciter la fonction duale $\xi(y; \cdot, \cdot, \cdot, \cdot)$ en utilisant les conditions KKT. On a:

$$\begin{cases} \nabla_x \mathcal{L}(y; x, u, \lambda^1, \lambda^2, \mu^1, \mu^2) = 0 \\ \nabla_u \mathcal{L}(y; x, u, \lambda^1, \lambda^2, \mu^1, \mu^2) = 0 \end{cases} \Rightarrow \begin{cases} D^T \lambda^1 + A^T \lambda^2 = 0 \\ u + (B^T \lambda^2) \odot y + \mu^1 - \mu^2 = 0. \end{cases} \quad (3.21)$$

En multipliant par x la première équation du (3.21) on élimine le terme correspondant dans l'expression du $\xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2)$. La deuxième équation du système (3.21) nous permet d'explicitier le contrôle optimal \bar{u} et de remplacer cette expression dans $\xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2)$.

Dans ce cas, la fonction support devient:

$$\xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2) = \begin{cases} -\frac{1}{2} u(\lambda, \mu)^T u(\lambda, \mu) - \pi(\lambda, \mu) & , \text{ si } D^T \lambda^1 + A^T \lambda^2 = 0 \\ -\infty & , \text{ sinon } , \end{cases} \quad (3.22)$$

où $\lambda = (\lambda^1, \lambda^2)$, $\mu = (\mu^1, \mu^2)$ et $u(\lambda, \mu)$, $\pi(\lambda, \mu)$ sont donnés par:

$$\begin{aligned} u(\lambda, \mu) &= -(B^T \lambda^2) \odot y - \mu^1 + \mu^2 \\ \pi(\lambda, \mu) &= d^T \lambda^1 + b^T \lambda^2 + u_{max}^T (\mu^1 + \mu^2). \end{aligned} \quad (3.23)$$

Avec ces considérations on peut expliciter le problème dual associé au $(P(y))$:

$$\left\{ \begin{array}{ll} \max_{\lambda^1, \lambda^2, \mu^1, \mu^2} & \xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2) \\ s.t. & D^T \lambda^1 + A^T \lambda^2 = 0 \\ & \lambda^1 \in \mathbb{R}^p \\ & \lambda^2 \in \mathbb{R}^q \\ & \mu^1 \in \mathbb{R}_+^{N+1} \\ & \mu^2 \in \mathbb{R}_+^{N+1}, \end{array} \right. \quad (3.24)$$

où l'expression de $\xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2)$ est donnée par (3.22).

On va utiliser ces résultats pour prouver les propositions suivantes.

Proposition 3.3.1. *Soient $(\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k$ les multiplicateurs optimaux obtenus en résolvant le problème $P(y^k)$. Alors, quelque soit $y \in Y_\Delta$, $\xi(y; (\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k)$ est une sous-estimation de l'objectif optimal du problème $P(y)$. En plus, cette fonction support prend la valeur finie:*

$$\xi(y; (\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k) = -\frac{1}{2} \|u(\lambda^k, \mu^k)\|^2 - \pi(\lambda^k, \mu^k). \quad (3.25)$$

Démonstration. Si $(\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k$ sont les multiplicateurs optimaux du problème $P(y^k)$, alors ces multiplicateurs sont dual-faisables, i.e. ils satisfont les contraintes du problème dual (3.24). Mais ces contraintes ne dépendent pas explicitement de y , le vecteur entier, de sorte que les multiplicateurs $(\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k$ sont dual faisables pour tout problème $(P(y))$.

Compte tenu du fait que le problème dual est un problème de maximisation on obtient (par dualité) :

$$val(P(y)) \geq \xi(y; (\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k),$$

où, $val(P(y))$ est l'objectif optimal du problème $P(y)$.

La fait que la fonction duale prend une valeur finie s'obtient en observant que la fonction duale devient $-\frac{1}{2} u(\lambda, \mu)^T u(\lambda, \mu) - \pi(\lambda, \mu)$ pour tout multiplicateur tel que $D^T \lambda^1 + A^T \lambda^2 = 0$ (ce qui est bien le cas pour $(\lambda^1)^k$ et $(\lambda^2)^k$). \square

L'importance de cette proposition est illustrée par le résultat suivant.

Proposition 3.3.2. *Pour chaque k , (PM^k) est un problème de programmation mathématique linéaire mixte et son objectif optimal $val(PM^k)$ est fini (et c'est donc une sous-estimation non triviale de $val(P)$).*

Démonstration. Étant donné que y_i est une variable binaire (i.e. $y_i^2 = y_i$, $i = 0, \dots, N$) la fonction support $\xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2)$ peut être exprimée comme une fonction linéaire en y :

$$\xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2) = \sum_{i=0}^N \left[-\frac{1}{2}(\alpha_i^2 + 2\alpha_i \gamma_i) y_i - \frac{1}{2} \gamma_i^2 - (\lambda_i^1 d_i + \lambda_i^2 b_i + u_{\max}(\mu_i^1 + \mu_i^2)) \right], \quad (3.26)$$

avec

$$\alpha \triangleq B^T \lambda^2 \text{ and } \gamma \triangleq \mu^1 - \mu^2.$$

Alors le problème **master** est un problème **MILP**.

D'autre part $val(PM^k)$ peut-être écrite comme:

$$val(PM^k) = \max_{j=1, \dots, k} \min_{y \in Y_\Delta} \xi(y; (\lambda^1)^j, (\lambda^2)^j, (\mu^1)^j, (\mu^2)^j).$$

Du fait que les opérateurs *min* et *max* sont pris sur un ensemble fini de valeurs (l'ensemble $\{1, \dots, k\}$ dans le premier cas et l'ensemble Y_Δ dans le deuxième cas) et que la fonction support $\xi(y; (\lambda^1)^j, (\lambda^2)^j, (\mu^1)^j, (\mu^2)^j)$ est finie on obtient que $val(PM^k)$ est finie.

D'autre part:

$$val(PM^k) \leq \sup_{\lambda, \mu \geq 0} \xi(y; \lambda, \mu) \quad , \quad y \in Y_\Delta,$$

ce qui implique

$$val(PM^k) \leq \min_{y \in Y_\Delta} \sup_{\lambda, \mu \geq 0} \xi(y; \lambda, \mu) = val(P).$$

Ces dernières relations prouvent que l'objectif optimal du **master** est un minorant de l'objectif optimal du problème (P) . \square

Ceci démontre que le **master** fournit une limite inférieure valide (non triviale) ce qui nous permet de construire un algorithme de type **GBD**. Ils nous reste à prouver la terminaison de l'algorithme dans un nombre fini d'itérations.

Proposition 3.3.3. *L'algorithme **GBD** proposé converge après un nombre fini d'itérations.*

Démonstration. A chaque itération k , le vecteur entier antérieur y^{k-1} est éliminé du **master** en utilisant une *coupe entière*. Comme l'ensemble des vecteurs entiers à explorer Y_Δ est fini, cela implique la convergence de l'algorithme après un nombre fini d'itérations. Dans le pire cas ce nombre est égal au $card(Y_\Delta)$. \square

Remarque 3.3.1. Dans le cas du **master GBD** standard (sans utiliser les coupes entières qu'on propose) l'algorithme converge. En fait, si à une certaine itération k on obtient un entier déjà analysé il a été prouvé que la limite inférieure est égale à la limite supérieure. Mais cette démarche standard nous empêche d'intégrer la notion de saturation, qui est un pas important dans notre algorithme (voir aussi la discussion sur la finitude de l'algorithme du §3.2.1).

Remarque 3.3.2. Même si l'expression linéaire donnée auparavant (3.26) est obtenue sans prendre en compte les contraintes de boîte sur les variables d'état x , la même démarche peut-être utilisée pour obtenir une sous-estimation linéaire si ce type de contrainte est ajoutée au problème. En fait, le cas le plus général qu'on peut traiter est celui où des contraintes quadratiques convexes séparables en x et u sont ajoutées au problème. Le lecteur pourra vérifier facilement que les considérations et les calculs faits restent valables dans ce cas.

Remarque 3.3.3. Le fait que les contraintes du problème dual ne dépendent pas explicitement de y est une caractéristique essentielle de notre démarche. En fait, si tel n'est pas le cas, on ne peut plus garantir que les multiplicateurs associés au problème primal pour un certain y fixé sont des multiplicateurs faisables dans les autres problèmes **primal** (pour les autres y fixés). Dans ce cas, deux cas peuvent se présenter: d'une part, la fonction duale prend une valeur infinie pour un certain nombre de variables y , ce qui nous amène à un **master** linéaire par morceaux mixte, difficile à résoudre; d'autre part si toutes les fonctions duales prennent une valeur infinie, le **master** fournit une sous-estimation triviale, infinie et l'algorithme ne converge pas. Les propositions qu'on a énoncées auparavant garantissent que ces situations ne peuvent pas se produire.

3.3.2 Implantation de l'algorithme

Un algorithme **GBD** basé sur l'idée du **primal-master** peut-être développé en prenant en compte les résultats théoriques du paragraphe précédent. Avant de donner cet algorithme, on va illustrer la façon dont on s'affranchit de l'hypothèse (**H2**), difficile à vérifier en pratique. Si le **primal** est non-faisable, alors en formulant un problème de non-faisabilité, comme dans l'algorithme **GBD** standard, on se rend compte que les contraintes duales font intervenir le paramètre y de façon explicite. Comme cela a déjà été expliqué, on ne peut pas obtenir une limite inférieure valide dans ce cas. C'est la raison pour laquelle, dans notre algorithme, un y^k non-faisable est tout simplement éliminé du **master** à l'aide de la coupe entière $y^T(e - y^k) \geq 1$.

Afin d'initialiser l'algorithme, le premier entier faisable est trouvé à l'aide du problème :

$$\begin{cases} \max_y & y^T e \\ \text{s.t.} & C_{\Delta} y \leq \Delta_0 \\ & y \in \{0,1\}^{N+1}. \end{cases} \quad (3.27)$$

Si la solution entière y^0 de ce problème est non-faisable, alors la coupe entière $y^T(e - y^0) \geq 1$ est ajoutée au problème (3.27) et ce dernier à nouveau résolu. Cette procédure est répétée tant qu'un vecteur entier non-faisable est obtenu.

Avec ces considérations, on peut formuler notre algorithme:

- Soient $k = 0$ et $y^k = y_0$ première solution *saturée* faisable.
Soient $\mathbf{LBD} = -\infty, \mathbf{UBD} = +\infty$
tant-que ($|\mathbf{UBD} - \mathbf{LBD}| < \varepsilon$)
 - Résoudre $P(y^k)$.
si **primal** est faisable **alors**
 - Actualiser **UBD**
 - Soit $(\lambda^1, \lambda^2, \mu^1, \mu^2) = ((\lambda^1)^k, (\lambda^2)^k, (\mu^1)^k, (\mu^2)^k)$
 - Ajouter la contrainte: $\eta \geq \xi(y; \lambda^1, \lambda^2, \mu^1, \mu^2)$ dans le **master**
 - fin**
 - Ajouter la contrainte $(y^T(e - y^k) \geq 1)$ dans le **master**
 - Résoudre le **master** (**MILP**). Soit \tilde{y}^k sa solution entière.
 - **STOP** si le **master** est non-faisable. Retourner la meilleure solution obtenue.
 - Saturer la solution du **master** \tilde{y}^k . Obtenir le nouveau y^k
- fin tant-que**

Comme on l'a déjà précisé, la solution du problème **master** à l'itération k , n'est pas nécessairement *saturée*. Ceci explique notre pas additionnel où cette solution est saturée. Peu importe la façon dont la saturation se fait, le paragraphe §2.4 démontre que la solution entière ainsi obtenue n'a pas été analysée auparavant. *C'est un pas important qui évite le bouclage et accélère la convergence de l'algorithme.* D'autre part, en saturant on augmente les chances d'obtenir un vecteur entier faisable.

3.3.3 Résultats

Le même exemple du mobile unidimensionnel présenté en §3.2.2 a été considéré. Le logiciel commercial CPLEX (8.1) est utilisé pour résoudre les deux type de problèmes, **primal** (QP) et **master** (**MILP**).

On peut remarquer (sur le Tableau 3.3.3) une nette amélioration quant aux résultats obtenus avec cette variante de **GBD** par rapport aux résultats obtenus avec le premier modèle **MIQP**. Le succès de la méthode peut-être expliqué par plusieurs facteurs qui jouent en notre faveur:

- premièrement, la limite inférieure issue du problème **master** est très efficace. Il s'agit là d'une conséquence de la façon dont cette limite à été obtenue, en évaluant la fonction duale d'un problème pour un ensemble des multiplicateurs faisables. Si les deux problèmes sont suffisamment semblables, la sous-estimation est de très bonne qualité et une bonne sous-estimation implique automatiquement un petit nombre d'itérations.

N	Δ	Δ_0	N_c	N_{con}	t_{exec}	N_{iter}
50	30	15	120	123	0.13	5
100	80	60	300	223	0.26	5
150	100	40	450	363	0.25	4
250	200	100	750	553	0.16	2
300	250	200	900	653	1.42	6
300	250	100	900	653	0.19	2
300	150	100	900	753	1.54	7
300	100	25	900	853	0.66	4
500	300	200	1500	1203	0.63	3
700	400	150	2100	1703	8.30	9

TAB. 3.2 – Résultats obtenus avec l’algorithme **GBD** sur le modèle **MINLP**. La légende est: N , le nombre de points dans la discrétisation (également le nombre de variables binaires); Δ, Δ_0 paramètres des contraintes logiques C_Δ ; N_c le nombre des variables continues (x et u); N_{con} le nombre total des contraintes (collocation et (C_Δ)); t_{exec} le temps d’exécution (en secondes); N_{iter} le nombre d’itérations nécessaires pour converger.

- les temps d’exécution raisonnables dépendent du temps nécessaire à l’algorithme afin de résoudre les problème **master** et **primal**. Notre **master** a une structure particulière (contenant une matrice totalement unimodulaire) et on a remarqué qu’un nombre très petit d’itérations est nécessaire afin de trouver la solution optimale (l’arbre du **BB** se termine après l’exploration de quelques nœuds). Cette affirmation a été vérifiée sur les différents problèmes de tests utilisés.

Les temps d’exécution ne dépendent pas des paramètres $N - \Delta$ et $\Delta - \Delta_0$. Ceci confirme le fait que l’algorithme est efficace même si la dimension de l’espace de recherche est très grande.

Dans certains cas, qui restent néanmoins très rares, on a remarqué que les solutions obtenues pour les différents problèmes **primal** sont très proches les unes des autres; ceci se traduit par un temps de calcul et un nombre d’itérations qui est très important et par une faible convergence. Dans ce cas l’algorithme a besoin d’explorer un grand nombre de solutions entières; comme ces cas sont plutôt rares, on préfère arrêter l’algorithme si les deux conditions suivantes sont satisfaites:

- la limite inférieure ne s’améliore de façon significative après un grand nombre d’itération consécutives ;
- la meilleure limite supérieure ne s’améliore pas après un grand nombre d’itérations consécutives .

Cette stratégie semble assez efficace car la solution locale obtenue dans ces

cas est proche de la solution optimale, souvent à moins de 5% :

$$\frac{UBD - LBD}{UBD} \leq 0.05 \ .$$

3.4 Exemples

3.4.1 L'oscillateur harmonique

On s'intéresse maintenant à un exemple plus pertinent pour les problèmes ayant une dynamique linéaire, celui de l'oscillateur harmonique à une dimension.

Le problème en dimension infinie s'écrit:

$$\left\{ \begin{array}{ll} \min_{x,u} & \frac{1}{2} \int_0^{t_f} u(t)^2 dt \\ s.t. & \dot{x}_1(t) = x_2(t) \quad , t \in [0, t_f] \\ & \dot{x}_2(t) = u(t) - \omega^2 x_1(t) \quad , t \in [0, t_f] \\ & x(0) = (x_1^0, x_2^0) \\ & x(t_f) = (x_1^f, x_2^f) \\ & -u_{max} \leq u \leq u_{max}. \end{array} \right. \quad (3.28)$$

Le problème qu'on traite s'obtient à partir de (3.28) en ajoutant les contraintes logiques; le modèle ainsi obtenu est analysé avec la méthode **GBD** décrite en §3.3.1.

Les tests effectués ont convergés pour l'ensemble des paramètres pris en compte. Généralement le nombre d'itérations de l'algorithme est petit et une fois que la solution optimale est trouvée l'algorithme s'arrête en 2 ou 3 itérations. Même des problèmes de très grande taille (jusqu'à 2000 variables entières) ont été résolus; ceci est surtout dû à la structure particulière du **master**, car en général les problèmes MILP de cette taille sont très difficiles à résoudre.

Les tests qu'on a effectués sont obtenus en faisant varier les conditions initiales et finales du problème (3.28) d'une part et les paramètres des contraintes (C_Δ), Δ et Δ_0 , d'autre part.

Dans la plupart des cas, le contrôle obtenu est périodique; à notre avis, les cas plus rares où on obtient un contrôle non périodique sont dus à des erreurs numériques.

Il y a une ressemblance évidente entre la structure du contrôle dans le cas où les contraintes logiques sont prises en compte et la structure du contrôle obtenu pour le cas non-contraint: si le contrôle est permis dans certains intervalles alors sa forme est proche de la forme du contrôle optimal sans contraintes logiques (3.28). Quant à la structure du contrôle, on a remarqué que si $\Delta_0 \geq \frac{\pi}{\omega}$ (on peut contrôler le système sur au moins une demi-période de l'oscillateur harmonique) alors les intervalles de non contrôle sont pratiquement centrés sur les zéros du contrôle optimal non restreint (Figure 3.2).

On a essayé de détruire cette structure en diminuant les paramètres Δ et Δ_0 (en faisant cela on s'assure que l'ancienne structure du contrôle viole les contraintes C_Δ). Les figures 3.3, 3.4 et 3.5 présentent la solution optimale dans certains de ces cas. Les intervalles de contrôle préférés sont ceux où le contrôle optimal du problème sans contraintes logiques a une influence importante sur le système.

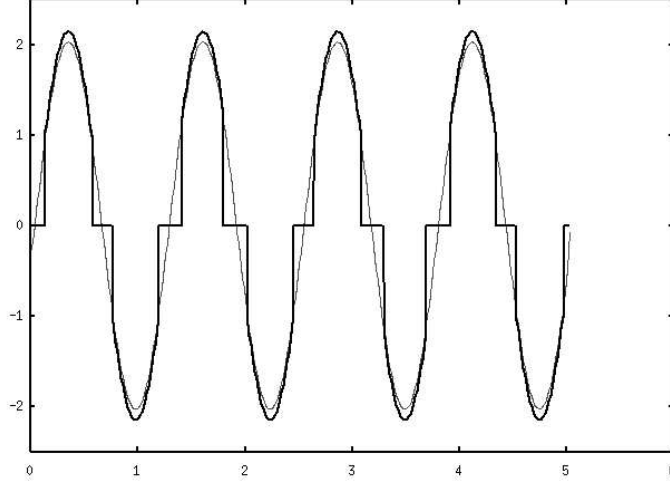


FIG. 3.2 – Contrôle optimal en fonction du temps ($N = 400$, $\Delta = 150$, $\Delta_0 = 100$). La ligne épaisse dénote le contrôle optimal obtenu en prenant en compte les contraintes logiques. La ligne plus fine est le contrôle optimale dans le cas relaxé (sans contraintes logiques).

3.4.2 D'autres exemples

Sur chacun des problèmes suivants on a ajouté les contraintes (C_Δ) et on a résolu la formulation ainsi obtenue par la méthode **GBD** développée et décrite dans le paragraphe précédent.

Problème no. 1 Le problème à résoudre est le suivant:

$$\left\{ \begin{array}{ll} \min_{x,u} & \frac{1}{2} \int_0^{t_f} u(t)^2 dt \\ \text{s.t.} & \dot{x}_1(t) = x_2(t) \quad , t \in [0, t_f] \\ & \dot{x}_2(t) = u(t) \quad , t \in [0, t_f] \\ & x(0) = (10.0, 10.0) \\ & x(t_f) = (0.0, 0.0) \\ & t_f = 10.0 \quad -U_{max} \leq u \leq U_{max} \end{array} \right. \quad (3.29)$$

Les figures (3.6), (3.7), (3.8) montrent les solutions optimales obtenues pour 3 cas différents testés.

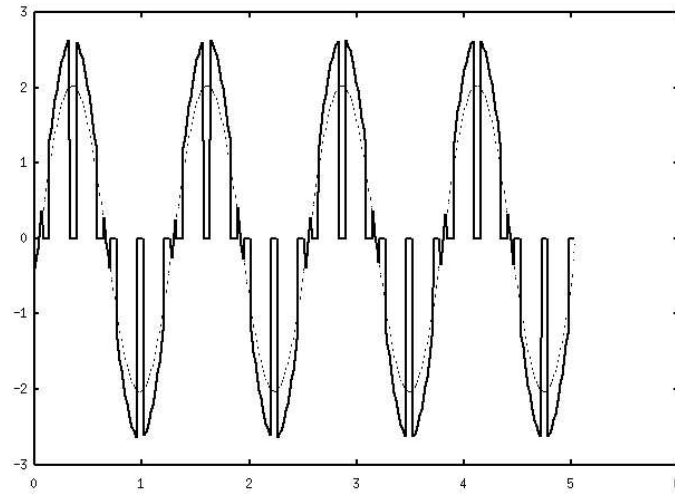


FIG. 3.3 – Contrôles optimaux en fonction du temps ($N = 400$, $\Delta = 20$, $\Delta_0 = 15$). La ligne épaisse montre le contrôle optimal avec contraintes logiques et la ligne pointillée représente le contrôle relaxé (sans contraintes logiques).

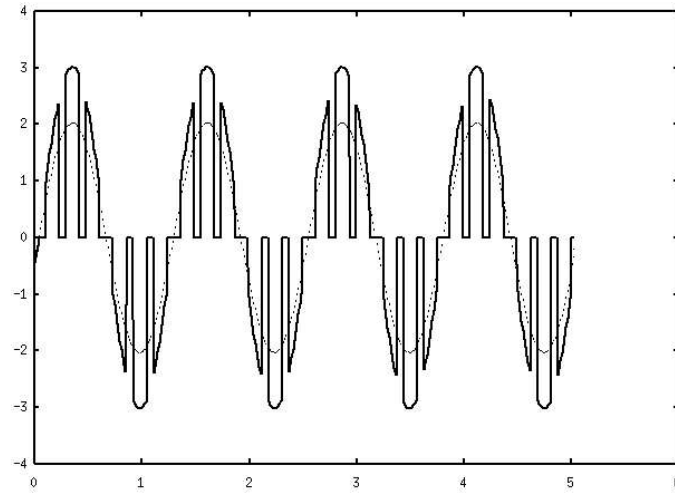


FIG. 3.4 – Contrôle optimal en fonction du temps ($N = 400$, $\Delta = 15$, $\Delta_0 = 10$). Le cas avec contraintes logiques (ligne épaisse) et le cas sans contraintes logiques (en pointillé).

Problème no. 2

$$\left\{ \begin{array}{l} \min_{x,u} \quad \frac{1}{2} \int_0^{t_f} u(t)^2 dt \\ s.t. \quad \dot{x}_1(t) = x_2(t) \quad , t \in [0, t_f] \\ \dot{x}_2(t) = u(t) - k_1 x_1(t) - k_2 x_2(t) \quad , t \in [0, t_f] \\ x(0) = (10.0, 10.0) \\ x(t_f) = (0.0, 0.0) \\ t_f = 10.0 \end{array} \right. \quad 46 \quad -U_{max} \leq u \leq U_{max} \quad (3.30)$$

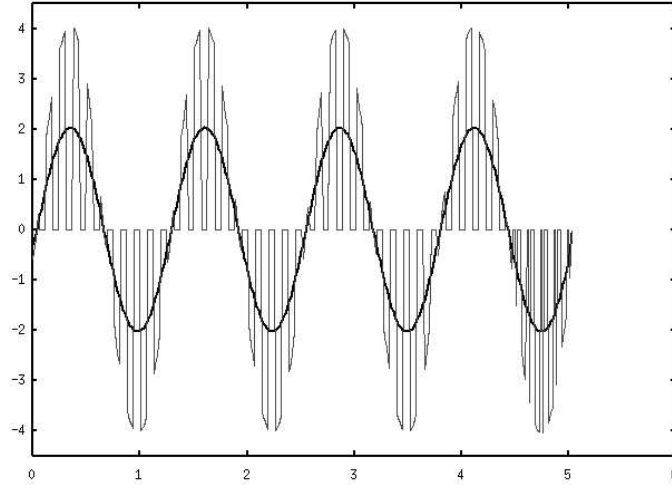


FIG. 3.5 – Contrôle optimal en fonction du temps ($N = 400$, $\Delta = 10$, $\Delta_0 = 5$). Le cas avec contraintes logiques (ligne épaisse) et le cas sans contraintes logiques (en pointillé).

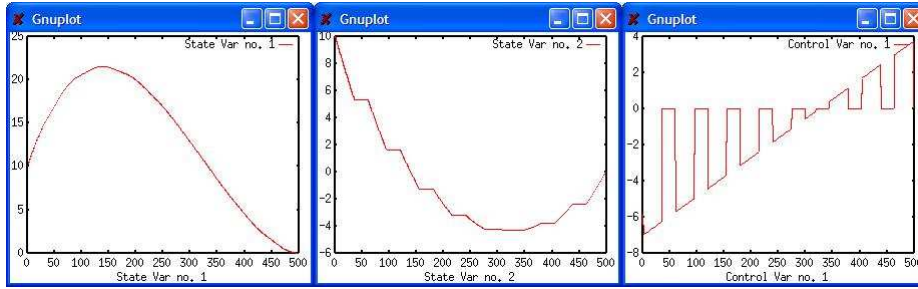


FIG. 3.6 – Résultats pour le problème no. 1 : position, vitesse, contrôle pour $N = 500, \Delta = 60, \Delta_0 = 35$

Les figures (3.9), (3.10), (3.11) montrent les solutions optimales obtenues pour 3 cas différents testés.

Problème no. 3

$$\left\{ \begin{array}{ll} \min_{x,u} & \frac{1}{2} \int_0^{t_f} u(t)^2 dt \\ \text{s.t.} & \dot{x}_1(t) = \omega x_2(t) \quad , t \in [0, t_f] \\ & \dot{x}_2(t) = -\omega x_1(t) + u(t) \quad , t \in [0, t_f] \\ & x(0) = (10.0, 10.0) \\ & x(t_f) = (0.0, 0.0) \\ & t_f = 10.0 \quad \quad \quad -U_{max} \leq u \leq U_{max} \end{array} \right. \quad (3.31)$$

Les figures (3.12), (3.13), (3.14) montrent les solutions optimales obtenues pour 3 cas différents testés.

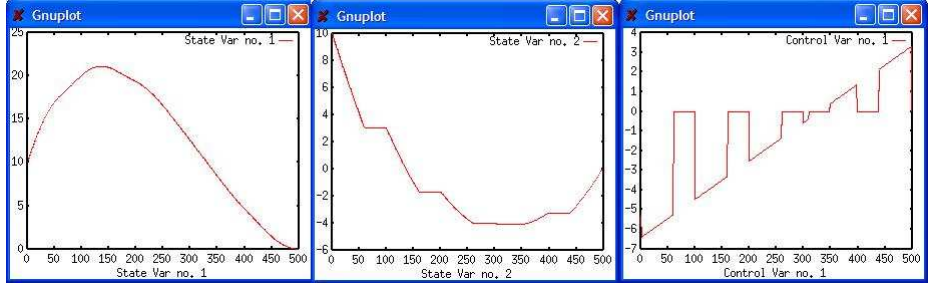


FIG. 3.7 – Résultats: position, vitesse, contrôle pour $N = 500, \Delta = 100, \Delta_0 = 60$

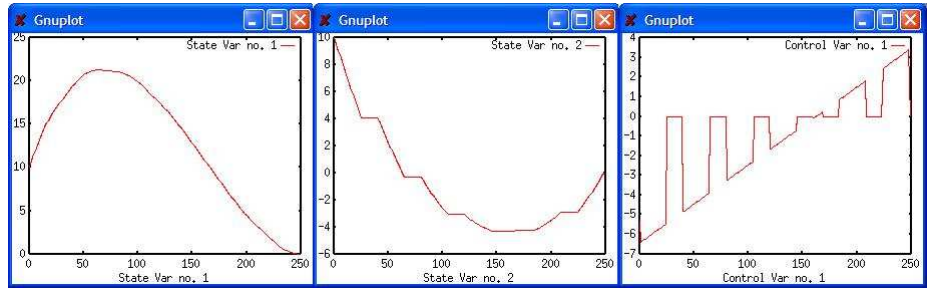


FIG. 3.8 – Résultats pour le problème no. 1 : position, vitesse, contrôle pour $N = 250, \Delta = 40, \Delta_0 = 25$

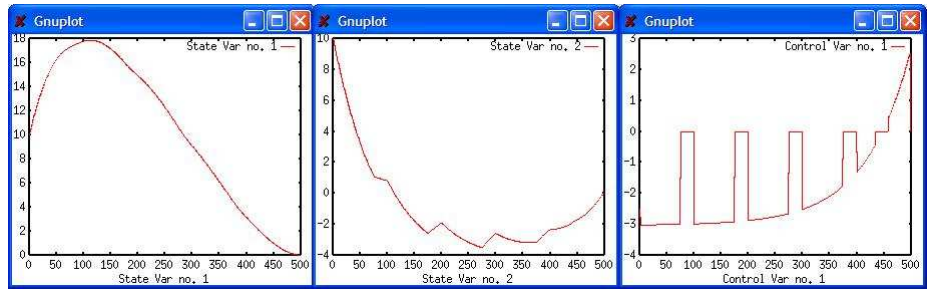


FIG. 3.9 – Résultats pour le problème no. 2: $N = 500, \Delta = 100, \Delta_0 = 75, k_1 = 0.6, k_2 = 0.0$

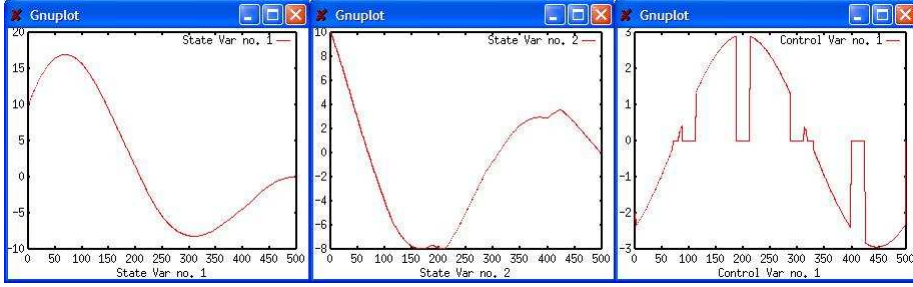


FIG. 3.10 – Résultats pour le problème no. 2: $N = 500, \Delta = 100, \Delta_0 = 75, k_1 = 0.0, k_2 = 0.4$

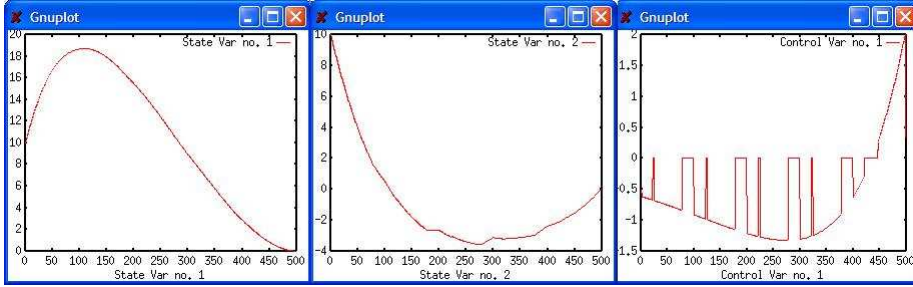


FIG. 3.11 – Résultats pour le problème no. 2: $N = 500, \Delta = 100, \Delta_0 = 75, k_1 = 0.1, k_2 = 0.6$

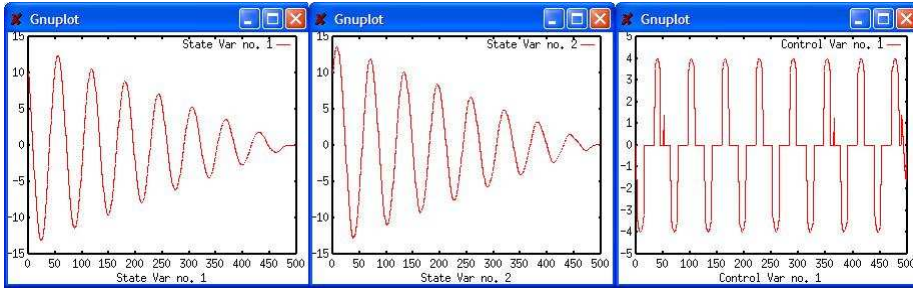


FIG. 3.12 – Résultats pour le problème no. 3: $N = 500, \Delta = 50, \Delta_0 = 25$

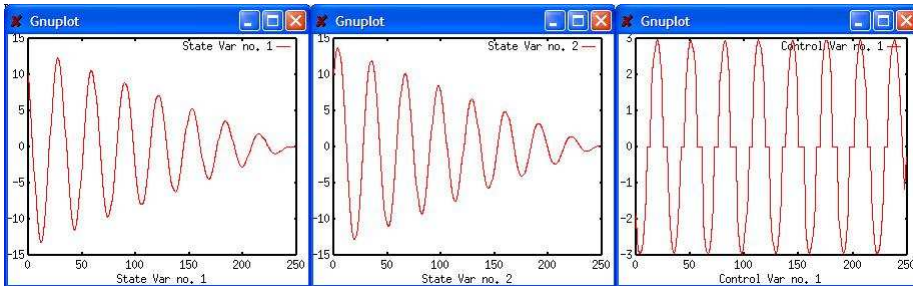


FIG. 3.13 – Résultats pour le problème no. 3: $N = 250, \Delta = 100, \Delta_0 = 70$

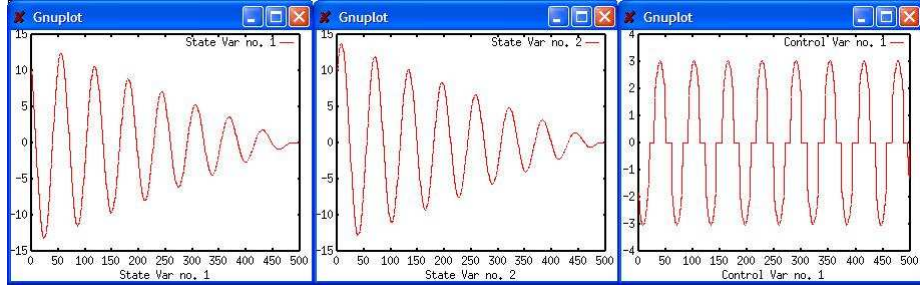


FIG. 3.14 – Résultats pour le problème no. 3: $N = 500, \Delta = 60, \Delta_0 = 40$

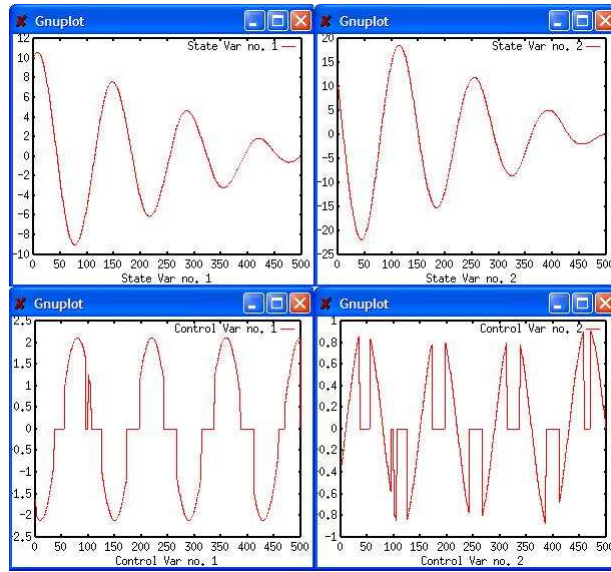


FIG. 3.15 – Résultats pour le problème no. 4: $N = 500, \Delta = 100, \Delta_0 = 75, \omega = 5$

Problème no. 4

$$\left\{ \begin{array}{l} \min_{x,u} \quad \frac{1}{2} \int_0^{t_f} u(t)^2 dt \\ \text{s.t.} \quad \dot{x}_1(t) = \omega x_2(t) + u_1(t) \quad , t \in [0, t_f] \\ \dot{x}_2(t) = -\omega x_1(t) + u_2(t) \quad , t \in [0, t_f] \\ x(0) = (10.0, 10.0) \\ x(t_f) = (0.0, 0.0) \\ t_f = 10.0 \end{array} \right. \quad -U_{max} \leq u_1(t), u_2(t) \leq U_{max} \quad (3.32)$$

Les figures (3.15), (3.16) montrent les solutions optimales obtenues pour 2 cas différents testés.

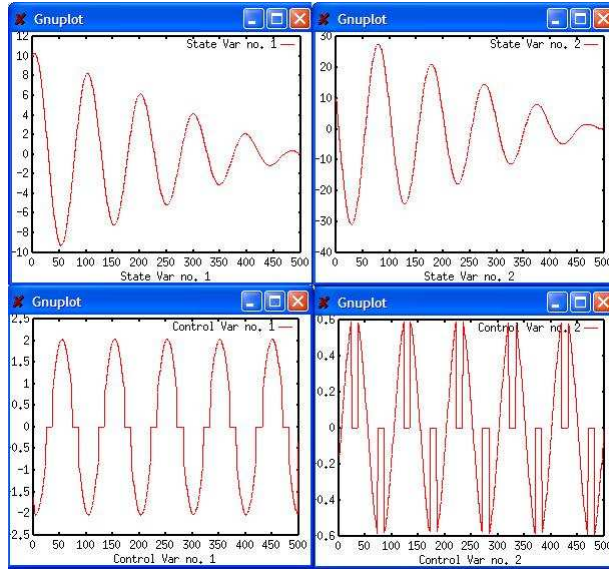


FIG. 3.16 – Résultats pour le problème no. 4: $N = 500, \Delta = 100, \Delta_0 = 75, \omega = 10$

3.5 Conclusions

Ce chapitre a été dédié au problème MINLP issu d’une dynamique linéaire dans le problème de contrôle optimal avec contraintes logiques. On a présenté et résolu deux modélisations différentes, la première de type **MIQP**, plus simple *a priori* et une deuxième **MINLP**. Les résultats pratiques nous montrent que le problème **MINLP** est résolu de façon plus efficace, non seulement en ce qui concerne la taille des problèmes, mais aussi par rapport aux temps d’exécution.

Les bons résultats s’expliquent par la manière dont on a su exploiter les particularités de notre problème: l’espace de recherche en introduisant la notion de solution *saturée*, le fait d’avoir un problème **primal** convexe et dont la fonction duale peut se calculer explicitement et le fait de travailler avec des variables binaires, ce qui nous a permis de formuler un problème **master** linéaire.

On a vu que les bons résultats obtenus concernent une classe plus large de problèmes, pas seulement les tests simples sur un mobile unidimensionnel, mais aussi les tests sur l’oscillateur harmonique (décrit dans ce chapitre) et d’autres problèmes où on a pu intégrer des forces de rappel élastiques ou de frottement dans le système. Le point commun de tous ces problèmes reste néanmoins la linéarité de la dynamique du système considéré.

Une autre remarque consiste la portée assez générale de l’algorithme **GBD** développé. On peut ajouter des contraintes non-linéaires quadratiques, séparables en contrôle et en état; dans ce cas, le **master** reste un

problème MILP et il est toujours une sous-estimation de la solution optimale du problème.

On conclut ce chapitre avec la remarque que, vu la vitesse de convergence de l'algorithme on peut espérer intégrer celui-ci dans des démarches plus compliquées, comme boucle interne; ce problème fait désormais partie (sous certains réserves néanmoins) de la classe des problèmes "faciles" à résoudre.

Dans le chapitre suivant on va essayer de s'affranchir d'une autre limitation, celle que pour tout y fixé le problème ainsi obtenu soit quadratique; on arrive ainsi à la classe la plus générale des problèmes d'optimisation, celle des problèmes non-convexes où on cherche l'optimum global.

Chapitre 4

Optimisation globale et transfert orbital

Dans cette partie on s'intéresse au cas général des dynamiques non-linéaires. La démarche qu'on propose d'appliquer est celle du Branch and Reduce, méthode qui réunit le Branch and Bound spatial (séparation sur les variables x et u), l'Analyse d'Intervalle et la Réduction de Domaine basée sur la dualité [55, 50]. Notre idée première en nous intéressant à l'approche Branch and Reduce était de traiter les problèmes de contrôle optimal à dynamique non-linéaire intégrant les contraintes (C_Δ). Cette démarche est la même que l'on traite de problèmes en variables mixtes (MINLP) où de problèmes uniquement en variables réelles à des fins d'optimisation globale. La difficulté prévisible, de traiter même des problèmes de taille modeste a restreint notre étude à ces derniers.

On commence par décrire la méthode de Branch and Reduce et ses éléments constitutifs : la convexification, la réduction de domaine et l'analyse d'intervalle. On illustre cette démarche sur un exemple, contenant un nombre limité de termes non-convexes. On analyse ensuite un problème beaucoup plus difficile en raison de son grand nombre de termes non-linéaires, celui du transfert orbital 2D à masse constante. En essayant d'adapter la démarche de Branch and Reduce à ce nouveau problème on remarquera l'importance et l'intérêt de la relaxation convexe. On décrit les différentes façons de convexifier le problème spatial et on fait une analyse comparative des différentes relaxations obtenues. La fin du chapitre utilise ces relaxations pour trouver la solution globale du problème (sans contraintes logiques) et on discute les limites de la méthode.

4.1 La démarche Branch and Reduce

Soit le problème d'optimisation non-convexe suivant, issu de la discrétisation par la collocation directe d'un problème de contrôle optimal à dynamique

non-linéaire :

$$(P) \left\{ \begin{array}{ll} \min_{x,u} & u^T u \\ \text{s.t.} & g(x,u) \leq 0 \\ & h(x,u) = 0 \\ & x \in X \\ & u \in U, \end{array} \right. \quad (4.1)$$

où $x \in \mathbb{R}^n$ est le vecteur des variables d'état, u le vecteur de variables de contrôle, g et h sont des fonctions non-linéaires.

La méthode de Branch and Reduce est une démarche déterministe pour la résolution de problèmes d'optimisation non-convexes. La méthode a été appliquée dans [1], [46], [47], [48] pour trouver l'optimum global des problèmes **MINLP** ou **NLP**. Néanmoins, son principal désavantage est le petit nombre de variables traitées dans le modèle, à cause de sa nature combinatoire.

Le cadre général de la méthode est celui du Branch and Bound. La séparation se fait en divisant successivement le domaine de définition des variables $X \cup U$. Sur chaque domaine réduit ainsi obtenu, $X_r \cup U_r$, l'évaluation d'un nœud consiste à obtenir un minorant (également appelé une estimation inférieure) et un majorant (estimation supérieure) de l'optimum global du problème (P) réduit au domaine $X_r \cup U_r$. L'estimation supérieure est calculée en résolvant le problème d'optimisation sur le domaine $X_r \cup U_r$ en appliquant une méthode locale (point intérieur, SQP, etc). L'estimation inférieure est obtenue en résolvant une relaxation convexe du problème, sur ce même domaine $X_r \cup U_r$. Un nœud est considéré exploré (analysé) soit quand ses limites inférieures et supérieures sont obtenues, soit quand la relaxation convexe est non-réalisable (pas de solution admissible). Dans ce dernier cas le nœud est éliminé. Une autre raison pour éliminer un nœud est que son minorant est plus grand que le meilleur majorant obtenu.

Pour chaque nœud, deux techniques de réduction du domaine sont appliquées : la première est basée sur la faisabilité et consiste à réduire le domaine de définition $X_r \cup U_r$ des variables en éliminant des domaines prouvés non faisables. La deuxième technique est appliquée après avoir sous-estimé le nœud en résolvant le relaxation convexe. Cette dernière technique consiste à réduire le domaine $X_r \cup U_r$ en éliminant des parties dont on sait qu'elles ne fournissent pas une meilleure solution que celle qu'on connaît à cet instant. Afin de réaliser ce type de réduction, la théorie de la dualité est appliquée au problème relaxé. L'algorithme de Branch and Reduce est illustré sur la Figure 4.1.

On va détailler dans les sections qui suivent les principaux pas de l'algorithme.

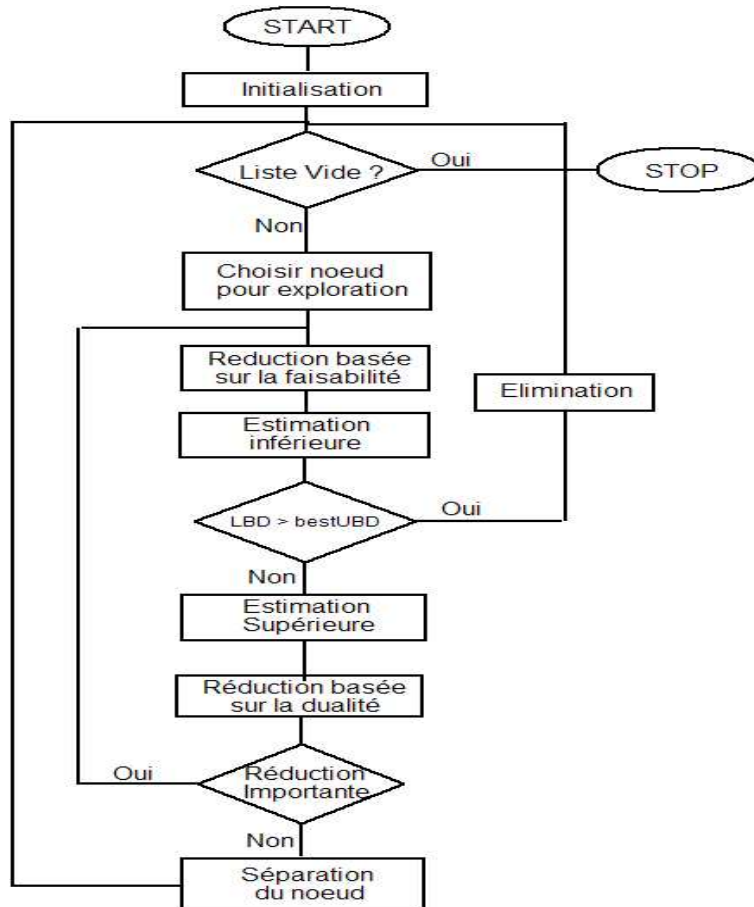


FIG. 4.1 – L’algorithme de Branch and Reduce avec ses différentes étapes : l’exploration d’un nœud en calculant un minorant et un majorant, la réduction du domaine (faisabilité et dualité) et l’élimination d’un nœud (à cause d’un mauvais minorant ou de l’absence de solution admissible dans le problème relaxé).

4.1.1 Majoration

Comme on l'a déjà précisé, le majorant de l'optimum global de (P) est obtenu en résolvant le problème (P) réduit à X_r . Si (P) est un problème d'optimisation non-convexe alors toute solution locale du problème (P) est un majorant de l'optimum global.

4.1.2 Minoration. Convexification

Afin d'obtenir un minorant pour l'objectif optimal du problème non-convexe, un problème relaxé et convexe est construit à partir du (P) . La démarche consiste à relaxer chaque terme non-linéaire intervenant dans le problème d'optimisation non-convexe.

Afin de mieux illustrer ce concept, supposons qu'un terme non-convexe $q(x,u)$ apparaisse dans une équation ou une inéquation appartenant aux contraintes de (P) . Une nouvelle variable y est alors introduite à sa place (l'ancienne équation est maintenant linéaire si le seul terme non-linéaire a été remplacé par y) et une nouvelle équation est ajoutée au système:

$$y = q(x,u), \quad (4.2)$$

où $y \in Y = [y^L, y^U]$, $x \in X = [x^L, x^U]$ et $u \in U = [u^L, u^U]$ sont des variables dans le problème d'optimisation et $q : X \times U \rightarrow \mathbb{R}$ est une fonction non-linéaire. Soient $q^{sup} : X \times U \rightarrow \mathbb{R}$ et $q^{inf} : X \times U \rightarrow \mathbb{R}$ les enveloppes respectivement concave et convexe de la fonction q , c'est à dire:

$$\begin{aligned} q^{sup}(x,u) &\geq q(x,u) \quad \forall (x,u) \in X \times U \\ q^{inf}(x,u) &\leq q(x,u) \quad \forall (x,u) \in X \times U \end{aligned} \quad (4.3)$$

Afin d'obtenir une relaxation convexe du problème on relaxe l'équation (4.2) en la remplaçant par les deux inégalités suivantes:

$$\begin{cases} +y - q^{sup}(x,u) &\leq 0 \\ -y + q^{inf}(x,u) &\leq 0 \end{cases} \quad (4.4)$$

Cette méthode est répétée pour chaque terme non-linéaire apparaissant dans le système d'équations:

- une nouvelle variable est ajoutée pour remplacer le terme non-linéaire
- deux inégalités convexes relaxent la contrainte d'égalité

Le nouveau problème ainsi obtenu est convexe (par construction) et son objectif optimal est un minorant du problème de départ (car les égalités ont été relaxées par des inégalités, comme en 4.4).

Remarque 4.1.1. La convexification nous amène à introduire de nouvelles variables et équations. Si le problème non-convexe contient beaucoup de termes non-linéaires nécessitant une convexification, alors l'obtention d'un minorant peut s'avérer coûteuse en temps de calcul (vu la taille considérable de la relaxation convexe obtenue).

Remarque 4.1.2. La qualité du minorant est donnée par la façon dont les enveloppes convexes et concaves sont obtenues. Plus les enveloppes se rapprochent de la fonction à approximer, mieux on approxime le problème non-convexe. Il convient de distinguer ici deux facteurs importants dans l'obtention des bonnes enveloppes convexes (ou concaves): d'une part la prise en compte des bornes de l'intervalle pour les variables et d'autre part, comme on peut facilement le supposer, le fait que certaines fonctions se prêtent mieux que d'autres à la convexification. Par exemple, la fonction $\sin L$ pour $L \in [0, 50\pi]$ est fortement non-linéaire; il est clair que la convexification d'une telle fonction doit être évitée (il vaut mieux trouver une reformulation du problème, en utilisant d'autres systèmes de coordonnées; cette démarche basée sur la reformulation du problème non-convexe va être détaillée plus tard, quand on s'intéressera au problème de transfert orbital 2D à masse constante). Quant au premier facteur, les bornes de l'intervalle, afin d'avoir de bonnes estimations on utilise en général l'analyse de l'intervalle pour les calculer.

On considère maintenant la convexification de quelques fonctions élémentaires qu'on va utiliser tout au long de ce chapitre.

Convexification de la fonction $q(x, y) = xy$. Soit le terme non-linéaire

$$q(x, y) = xy \text{ avec } x \in X = [x^L, x^U] \text{ et } y \in Y = [y^L, y^U]. \quad (4.5)$$

Cette fonction en x et y (en fait, bilinéaire) apparaît souvent dans les modèles qu'on analysera par la suite. La convexification bilinéaire qu'on présente ici est celle obtenue par McCormick [37]. Elle consiste à ajouter une nouvelle variable z et à convexifier en utilisant les équations linéaires suivantes:

$$\begin{aligned} z &\geq x^U y + y^U x - x^U y^U \\ z &\geq x^L y + y^L x - x^L y^L \\ z &\leq x^U y + y^L x - x^U y^L \\ z &\leq x^L y + y^U x - x^L y^U \end{aligned} \quad (4.6)$$

La figure (4.2) montre les enveloppes convexes, concaves et la fonction bilinéaire pour $x \in [-1, 1]$ et $y \in [0, 1]$.

Convexification de la fonction $q(x) = x^\alpha$. Suivant les valeurs du paramètre $\alpha \in \mathbb{R}_+$, la fonction $q(x) = x^\alpha$ est soit concave soit convexe. Les enveloppes convexes et concaves sont alors obtenues à l'aide des fonctions linéaires, en utilisant les sous-tangentes aux bornes de l'intervalle x^L et x^U . La figure 4.3 montre ces enveloppes pour le cas particulier $q(P) = P^{3/2}$. Ce cas se généralise facilement, non seulement pour toute valeur de n , mais aussi pour toute fonction d'une variable convexe ou concave (comme $\log(x)$, $\exp(x)$, etc).

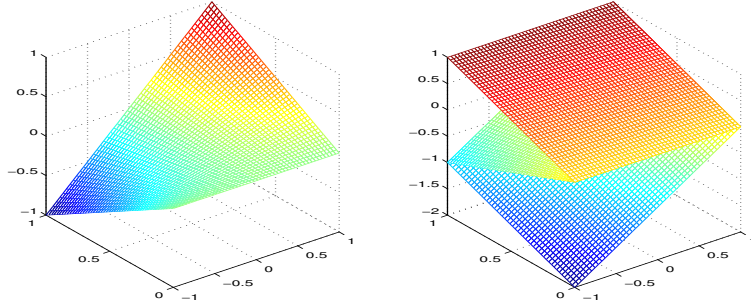


FIG. 4.2 – A gauche la fonction $z = xy$, pour $x \in [-1, 1]$ et $y \in [0, 1]$. A droite cette même fonction avec les enveloppes convexes et concaves (la fonction bilinéaire se trouve au centre du graphe). Pour plus de clarté, parmi les quatre enveloppes données par la convexification McCormick seulement deux sont montrées (l'une convexe en bas du z et l'autre concave en haut du z).

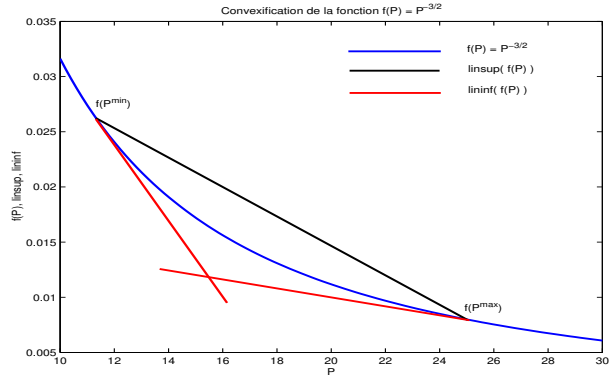


FIG. 4.3 – Enveloppes concaves et convexes de la fonction $q(P) = P^{-3/2}$. Dans ce cas les enveloppes sont linéaires: en bleu, la fonction q , en rouge les deux sous-estimations linéaires lininf et en noir la surestimation linéaire linsup . Toutes les droites sont des fonctions linéaires en P , dont les coefficients sont facilement obtenus comme expressions de P^L et P^U . Ici, $P^L = 11.5$ et $P^U = 25$

Remarque 4.1.3. Il est très important d'observer que si la relaxation convexe de (P) se fait en utilisant seulement des enveloppes convexes et concaves affines, alors le problème ainsi obtenu est quadratique. L'intégration des contraintes logiques sur la durée du contrôle nous amène au même type de problèmes que ceux traités dans le chapitre antérieur. En fait, la seule modification apportée à l'algorithme 4.1 afin de traiter le cas avec contraintes logiques consiste dans la façon dont la limite inférieure est obtenue: en résolvant un problème quadratique dans le cas continu et en résolvant un problème MINLP dans le cas mixte. C'est pour cette raison qu'on détaille uniquement le cas de l'optimisation globale en variables continues.

4.1.3 Réduction du domaine

Sahinidis et Tawalarmani proposent en [55] deux méthodes distinctes pour réduire le domaine de définition des variables $(X \cup U)$, dans notre cas). On présente ici les idées sur lesquelles ces deux techniques reposent, pour une description plus détaillée, on pourra consulter [55].

Réduction basée sur la faisabilité. Soit $X_r \cup U_r$ le domaine réduit caractérisant un nœud. S'il existe un domaine $D \subset X_r \cup U_r$ tel que le problème (P) est non-faisable sur le domaine D , alors celui-ci est éliminé (réduction) et le nouveau domaine de définition des variables associé au nœud devient :

$$D' = (X_r \cup U_r) \setminus D.$$

Les deux estimations inférieure et supérieure (minoration et majoration) se font seulement sur le domaine D' , ce qui augmente les chances d'avoir un bon minorant (respectivement majorant).

Quoique cette technique puisse être assez difficile à mettre en œuvre, dans des certains cas particuliers elle s'avère très utile. En ce qui nous concerne, l'exemple donné en 4.1.4 utilise l'analyse d'intervalle pour réduire le domaine de cette façon tandis que pour le problème de transfert orbital 2D à masse constante on développera un algorithme basé sur l'arithmétique d'intervalle afin de de réduire également le domaine de définition des variables.

Réduction basée sur la dualité. Soit (R) la relaxation convexe de (P) , LBD un minorant et UBD un majorant de la solution globale de (P) . Supposons que la variable x_j est saturée à sa limite supérieure, dans la solution du problème (R) et soit μ_j^{sup} le multiplicateur de Lagrange associé (*i.e.* $x_j = x_j^U$ et $\mu_j^{sup} > 0$). Alors pour tout $x_j < x_j^*$ on a $val(R) \geq UBD$, où :

$$x_j^* = x_j^U - \frac{UBD - LBD}{\mu_j^{sup}}$$

et $val(R)$ est l'objectif optimal du problème relaxé (R) .

Ce résultat simple (pour une démonstration voir [55]) permet de réduire le domaine de la variable x_j dans le cas où $x_j^* > x^L$. Le nouveau domaine ainsi obtenu pour la variable x_j est $[max(x^L, x_j^*), x^U]$.

De même la limite supérieure de la variable x_j est réduite si elle est saturée à sa limite inférieure. Dans ce cas:

$$x_j \in [x_j^L, min(x_j^*, x^U)], \text{ où } x_j^* = x^L + \frac{UBD - LBD}{\mu_j^{inf}}.$$

Dans le cas où la variable x_j n'est pas saturée ($\mu_j^{sup} = \mu_j^{inf} = 0$), Sahinidis et Tawalarmani proposent une technique de test (*probing*) afin de réduire le domaine. Cette méthode de *probing* étant coûteuse en temps de calcul (elle nécessite la résolution d'autres problèmes d'optimisation) elle ne va pas être utilisée par la suite. Nous allons nous limiter au cas où une ou plusieurs variables sont saturées dans le problème relaxé (R).

4.1.4 Exemple. Alunissage

Afin d'illustrer et de tester la méthode de Branch and Reduce, on commence par un problème assez simple, contenant un petit nombre de termes non-linéaires. Le problème de l'alunissage d'un véhicule à masse variable [19] est considéré :

$$(PL) \left\{ \begin{array}{ll} \min_{u,x,y,m} & \int_{t_0}^{t_f} u^2 dt \\ s.t. & \dot{x}(t) = y(t) \quad t \in [t_0, t_f] \\ & \dot{y}(t) = -g + \frac{u(t)}{m(t)} \quad t \in [t_0, t_f] \\ & \dot{m}(t) = -Ku(t) \quad t \in [t_0, t_f] \\ & x(0) = x_0, \quad x(t_f) = x_f \\ & y(0) = y_0, \quad y(t_f) = y_f \\ & m(0) = m_0, \quad m(t) \geq m_{vide}, t \in [0, t_f] \\ & 0 \leq u(t) \leq T_{max} \quad t \in [t_0, t_f], \end{array} \right. \quad (4.7)$$

où: le temps final t_f est fixé, $x(t), y(t), m(t)$ sont les variables d'état correspondant à la position, la vitesse et la masse du véhicule, $u(t)$ le contrôle (toujours positif), les positions et vitesses initiales et finales sont fixées (x_0, x_f, y_0, y_f), m_0 est la masse initiale du véhicule, m_{vide} sa masse au vide (sans carburant), g l'accélération gravitationnelle sur la Lune et K une constante caractérisant la consommation.

En raison du terme non-linéaire $\frac{u(t)}{m(t)}$, le problème (4.7) est non-convexe. Le modèle sur lequel on teste la méthode d'optimisation globale du Branch and Reduce est obtenue en discrétisant (PL) par collocation directe. Avec les considérations présentées en §1.2.1, les équations de la dynamique deviennent :

$$\begin{aligned}
-\frac{3}{2h}x_i + \frac{3}{2h}x_{i+1} - \frac{3}{4}y_i - \frac{3}{4}y_{i+1} - \frac{h}{8}\frac{u_i}{m_i} + \frac{h}{8}\frac{u_{i+1}}{m_{i+1}} &= 0 \\
-\frac{3}{2h}y_i + \frac{3}{2h}y_{i+1} - \frac{1}{4}\frac{u_i}{m_i} - \frac{1}{4}\frac{u_{i+1}}{m_{i+1}} - \frac{u_i}{\bar{m}_i} &= -\frac{3}{2}g \\
-\frac{3}{2h}m_i + \frac{3}{2h}m_{i+1} + \frac{5}{4}Ku_i + \frac{1}{4}Ku_{i+1} &= 0,
\end{aligned} \tag{4.8}$$

pour $i = 0, N-1$ et:

$$\bar{m}_i = \frac{1}{2}m_i + \frac{1}{2}m_{i+1} + \frac{5}{4}Ku_i + \frac{1}{4}Ku_{i+1}.$$

Convexification. La convexification s'obtient en utilisant les variables auxiliaires suivantes:

$$\begin{aligned}
T_i &= \frac{u_i}{m_i} & i = 0, N \\
M_i &= \frac{1}{2}m_i + \frac{1}{2}m_{i+1} + \frac{5}{4}Ku_i + \frac{1}{4}Ku_{i+1} & i = 0, N-1 \\
R_i &= \frac{u_i}{M_i} & i = 0, N-1.
\end{aligned} \tag{4.9}$$

Avec ces nouvelles variables les équations de la dynamique sont réécrites selon:

$$\begin{aligned}
-\frac{3}{2h}x_i + \frac{3}{2h}x_{i+1} - \frac{3}{4}y_i - \frac{3}{4}y_{i+1} - \frac{h}{8}T_i + \frac{h}{8}T_{i+1} &= 0 \\
-\frac{3}{2h}y_i + \frac{3}{2h}y_{i+1} - \frac{1}{4}T_i - \frac{1}{4}T_{i+1} - R_i &= -\frac{3}{2}g \\
-\frac{3}{2h}m_i + \frac{3}{2h}m_{i+1} + \frac{5}{4}Ku_i + \frac{1}{4}Ku_{i+1} &= 0, \\
\frac{1}{2}m_i + \frac{1}{2}m_{i+1} + \frac{5}{4}Ku_i + \frac{1}{4}Ku_{i+1} - M_i &= 0 \\
T_i m_i - u_i &= 0 \\
R_i M_i - u_i &= 0.
\end{aligned} \tag{4.10}$$

Le problème relaxé s'obtient en convexifiant les deux dernières équations de (4.10) par la méthode décrite en 4.1.2 (équations de McCormick). La relaxation ainsi obtenue est un problème quadratique.

Réduction du domaine. Parmi les deux méthodes de réduction de domaine, seule celle basée sur la faisabilité va être détaillée ici, car la réduction basée sur la dualité ne nécessite aucun effort particulier pour notre exemple.

Le domaine initial du problème (pour le nœud racine du Branch and Bound) est obtenu par analyse d'intervalle en utilisant les équations suivantes (provenant de système 4.8):

$$\begin{aligned}
m_{i+1} &= m_i - \frac{5Kh}{6}u_i + \frac{Kh}{6}u_{i+1} \\
y_{i+1} &= y_i + \frac{h}{6}\frac{u_i}{m_i} + \frac{h}{6}\frac{u_{i+1}}{m_{i+1}} + \frac{u_i}{\frac{3}{4h}m_i + \frac{3}{4h}m_{i+1} - \frac{3K}{16}u_i + \frac{3K}{16}u_{i+1}} - hg \\
x_{i+1} &= x_i + \frac{h}{2}y_i + \frac{h}{2}y_{i+1} + \frac{h^2}{12}\frac{u_i}{m_i} - \frac{h^2}{12}\frac{u_{i+1}}{m_{i+1}},
\end{aligned} \tag{4.11}$$

pour tout $i = 0, N - 1$.

Cette technique permet d'avoir un domaine initial réduit, améliorant ainsi la qualité de l'estimation inférieure (minorant) à la première itération de l'algorithme. Cette même technique est utilisée pour réduire le domaine après une séparation. Pour illustrer cela, soit x_i la variable choisie pour la séparation à une itération k et x_i^* le point de séparation. Les deux nouveaux domaines obtenus sont $[x_i^L, x_i^*]$ pour le premier nœud et $[x_i^*, x_i^U]$ pour le deuxième nœud. Pour chaque nœud les équations (4.11) permettent de réduire successivement les intervalles de définition pour les variables x_j, y_j, m_j , pour $j > i$. De même, le système (4.11) permet de réduire le domaine de définition des variables x_j, y_j, m_j , pour $j < i$.

Paramètres de l'algorithme. Avant de présenter les résultats et analyser l'efficacité de la méthode, on complète la description de l'algorithme par quelques remarques concernant les choix des paramètres dans la méthode de Branch and Bound.

Exploration des nœuds. Chaque fois qu'un nœud est subdivisé, les deux nœuds (fils) obtenus sont explorés (les estimations inférieures et supérieures sont calculées). Cela nous amène à la question du choix du nœud devant être subdivisé. Dans notre cas, on choisit toujours le nœud ayant l'estimation inférieure la plus petite; cette méthode rappelle la technique de **GBD**, ou le prochain vecteur entier à analyser était celui donnant le plus petit minorant. D'autre part, ce choix permet d'obtenir une suite croissante des minorants.

Séparation. Commençons par préciser que seul un petit nombre de variables est utilisé pour séparer les nœuds. Cette stratégie a le mérite de diminuer la complexité (combinatoire) du problème et elle est motivée par le fait qu'après une séparation, l'analyse d'intervalle permet de réduire considérablement les intervalles situés à proximité immédiate. Cette remarque est spécifique aux problèmes issus d'une discrétisation, lorsque entre les variables successives il existe un couplage fort issu de la dynamique du problème. D'autre part, en regardant le problème de départ on se rend compte que la variable m a une influence plus grande sur la qualité de la convexification, dans la mesure où elle est la seule variable d'état intervenant dans les

termes non-linéaires. Ces considérations nous amènent à une stratégie qui consiste à choisir plus souvent la variable m pour la séparation (par rapport aux autres variables x et y).

Pour chaque variable parmi celles acceptées pour la séparation d'un nœud on calcule la violation de la vraie contrainte (non-convexifiée) en utilisant la solution optimale du problème relaxé. La variable correspondant à la plus grande violation de la contrainte est choisie pour la séparation.

Le choix de la valeur utilisée pour séparer un nœud se fait en utilisant les règles suivantes (Sahinidis et Tawalarmani, [55]) :

- toutes les N_D itérations, le milieu de l'intervalle pour la variable choisie est utilisé pour diviser le nœud;
- la valeur de la solution optimale du problème relaxé est utilisée si la règle précédente ne s'applique pas.

Résultats. Le problème de test est défini par les valeurs suivantes des paramètres :

$$\begin{array}{llll} x_0 = 1.0 & x_f = 0.0 & & \\ y_0 = 1.0 & y_f = 0.0 & & \\ m_0 = 10.0 & m_{vide} = 6.0 & & \\ t_f = 50.0 & g = 0.1 & K = 0.01 & \end{array}$$

Le Tableau 4.1 montre les différents résultats obtenus sur le problème de test, avec une poussée maximale $T_{max} = 2$.

TAB. 4.1 – *Résultats pour le problème de test (poussée maximale $T_{max} = 2$ Newton). La colonne N_d représente le nombre de points de discrétisation, N_{var} est le nombre total de variables dans le modèle, $N_{nœuds}$ le nombre de nœuds explorés, N_p le nombre de problèmes résolus, T_{exec} le temps d'exécution total de l'algorithme, exprimé en secondes. La machine de test est un PC Linux, processeur Intel P4 2.8GHz avec 1Gb de mémoire vive.*

N_d	N_{var}	$N_{nœuds}$	N_p	$t_{exec}(s)$
25	100	3173	7346	145.00
40	160	14893	29786	2572.69
50	200	11835	23670	2930.14
60	240	16035	32070	7135.32
75	280	39005	78010	64284.24
80	360	105285	210570	$\approx 15h$

On observe que le nombre de points de discrétisation est beaucoup moins important que dans le cas linéaire avec contraintes logiques, la nature beaucoup plus combinatoire du problème étant un obstacle majeur. La plus

grande discrétisation utilisée ne dépasse pas les 100 points; même dans ce cas, le temps de calcul est très important (plusieurs jours) et la convergence de l'algorithme après un nombre raisonnable d'itérations est assurée en grande mesure par le choix d'un nombre plus restreint des variables pour la séparation d'un nœud (comme mentionné auparavant). En fait, *sans cette analyse d'intervalle*, l'algorithme peine à finir même pour un nombre limité de points de discrétisation. Les premiers tests effectués étaient plutôt décourageants dans ce cas, car on arrivait difficilement à dépasser les 15 points de discrétisation.

Une autre observation concerne le nombre important de problèmes résolus pour chaque test montré dans le Tableau 4.1. Résoudre 100000 problèmes d'optimisation dans un temps raisonnable implique d'une part l'utilisation de solveurs de bonne qualité, et d'autre part de garder un nombre limité de variables dans les deux types de problème d'optimisation utilisés : problème non-linéaire et relaxation convexe. Dans notre cas le solveur *KNITRO* a été utilisé pour calculer le majorant (problème NLP) et CPLEX pour résoudre la relaxation quadratique.

Un autre facteur très important est la qualité de la convexification. Pour cet exemple simple, le nombre de termes non-linéaires (induisant des non-convexités dans le problème de départ) est petit. Ceci a comme conséquence une convexification très bonne, *i.e.* qui donne une bonne sous-estimation de l'optimum global. Ceci se traduit dans notre exemple par un minorant qui dépasse souvent 85% du majorant dès les premières itérations.

L'un des inconvénient des algorithmes Branch and Bound est le nombre de nœuds actifs (explorés mais non-éliminés) gardés en mémoire. Notre algorithme a un comportement plutôt encourageant de ce point de vue car ce nombre de nœuds actifs ne dépasse presque jamais 30000.

Finalement, on présente l'influence de la poussée maximale T_{max} sur l'efficacité de l'algorithme. Intuitivement, une faible poussée maximale signifie un contrôle plus faible et un plus fort couplage entre les variables; cela rend l'analyse d'intervalle et la réduction de domaine encore plus efficace. Un autre avantage dû à une faible poussée est celui de pouvoir limiter de façon efficace le nombre de variables considérées pour la séparation. Les résultats du Tableau 4.2 étayaient ces affirmations.

On conclut cette partie introductive destinée à la méthode en mettant en évidence les différents facteurs qu'il faut prendre en compte afin d'accélérer la convergence de la méthode de Branch and Reduce. D'une part la qualité de la relaxation convexe est extrêmement importante, une mauvaise relaxation amenant à un mauvais minorant (trop éloigné du majorant) et à un faible taux d'élimination des nœuds. Un autre facteur important est lié à la réduction du domaine; dans notre cas, le fait de travailler avec un problème d'optimisation issu d'une discrétisation nous assure un fort couplage entre les différentes variables d'état. Cette caractéristique de notre problème doit être exploitée par la suite. Finalement, le grand nombre de nœuds explorés

TAB. 4.2 – *Influence de la poussée maximale T_{max} sur la résolution du problème de test. De plus grandes poussées impliquent un nombre plus important d'itération et donc, un nombre plus grand de problèmes d'optimisation (NLP et QP) résolus. Ceci est vérifié en général, mais il se peut que la stratégie (séparation, choix du nœud et de la variable) soit plus efficace dans un cas plutôt que dans l'autre. La seule différence entre les tests (outre la poussée maximale) est la façon dont la séparation d'un nœud est faite.*

N_d	$T_{max}(N)$	$N_{nœuds}$	N_p	$t_{exec}(s)$
25	2	3173	7346	145.00
25	2.5	4151	8302	202.85
25	3	19113	38226	984.02
25	4	24443	48886	2011.48
25	5	20783	41566	1284.30
25	10	66389	132778	3685.29

nous laisse penser à la nécessité d'avoir des solveurs robustes et efficaces pour résoudre la relaxation convexe, mais aussi le problème non-linéaire sur un domaine réduit des variables d'état.

4.2 Transfert orbital 2D à masse constante

On considère un satellite se trouvant sur une orbite initiale connue et qu'on veut ramener sur une orbite finale géostationnaire donnée en un temps final fixé t_f (voir Figure 4.4). Les positions initiale et finale du satellite sur l'orbite sont connues et on veut minimiser l'énergie dépensée le long du transfert.

Soit $u \in \mathbb{R}^2$ le contrôle, \mathbf{r} le rayon vecteur donnant la position du satellite en 2D et m sa masse. Alors, les équations du mouvement du satellite s'obtiennent en écrivant la loi du Newton (les forces impliquées sont l'attraction terrestre et le contrôle):

$$\ddot{\mathbf{r}} = -\mu \frac{\mathbf{r}}{r^3} + \frac{1}{m} \mathbf{u} \quad (4.12)$$

où μ est la constante d'attraction gravitationnelle et $\ddot{\mathbf{r}}$ la dérivée seconde par rapport au temps du rayon vecteur.

Afin de formuler le problème auquel nous nous intéressons en tant que problème d'optimisation on ajoute les conditions terminales (positions et vitesses initiales et finales fixées) et les contraintes de poussée (le contrôle est borné). L'objectif est approché en utilisant le carré de la norme L^2 du contrôle afin d'éliminer les difficultés issues de la non différentiabilité d'un

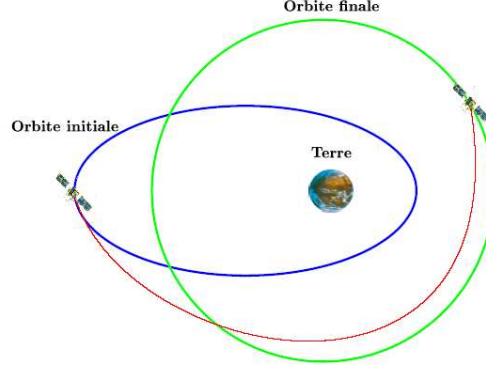


FIG. 4.4 – *Problème du transfert orbital 2D. Les position initiales et terminales sont données : orbite initiale et position du satellite sur orbite, orbite finale (géostationnaire) et la position du satellite sur l'orbite finale.*

objectif prenant en compte seulement la norme L. Le problème de contrôle optimal suivant est obtenu :

$$\left\{ \begin{array}{l} \min_{\mathbf{r}, \mathbf{u}} \quad \int_0^{t_f} u^2(t) dt \\ \text{s.t.} \quad \ddot{\mathbf{r}}(t) = -\mu \frac{\mathbf{r}(t)}{r^3(t)} + \frac{1}{m} \mathbf{u}(t) \quad t \in [0, t_f] \\ \mathbf{r}(0) = \mathbf{r}_0 \\ \mathbf{r}(t_f) = \mathbf{r}_f \\ \dot{\mathbf{r}}(0) = \mathbf{v}_0 \\ \dot{\mathbf{r}}(t_f) = \mathbf{v}_f \\ u^2(t) \leq T_{max}^2 \quad t \in [0, t_f] \end{array} \right. \quad (4.13)$$

Le problème (4.13) est un problème de contrôle optimal non-convexe, ce qui justifie les questions liées au caractère global de la solution obtenue.

4.2.1 Majoration.

Le majorant de l'optimum global du problème spatial s'obtient en appliquant une méthode locale (en utilisant toujours le principe que toute optimum local est un majorant de l'optimum global). Les méthodes locales peuvent être divisées en deux catégories : directes et indirectes.

Les approches *directes* consistent à discrétiser (4.13) en obtenant un problème d'optimisation en dimension finie. Les différents solveurs de programmation non-linéaire peuvent alors être utilisés afin de retrouver la solution locale [18], [13]. Le principal désavantage de cette démarche est la taille du problème obtenu après discrétisation. Par contre, l'avantage principal consiste en la grande diversité des contraintes continues qu'on peut ajouter

à (4.13), étant donné que les conditions d'optimalité de Karush-Kuhn-Tucker ne demandent aucune hypothèse quant aux variables d'état ou de contrôle.

Les méthodes *indirectes* sont basées sur le principe du maximum de Pontryagin [19]. Ce principe donne les conditions nécessaires d'optimalité, en transformant (4.13) en un problème aux conditions aux deux bouts. Un intégrateur combiné à une méthode de Newton peuvent être utilisés pour obtenir la solution (locale) de (4.13), cf. [15], [14], [10]. C'est l'approche la plus indiquée pour obtenir une solution locale du problème et l'indépendance de la méthode par rapport au nombre de points de discrétisation joue un rôle important. *A contrario*, le grand désavantage est le fait qu'on peut difficilement ajouter des contraintes sur les variables d'état dans les méthodes indirectes. Or, on a vu que les contraintes de bornes jouent un rôle important dans la méthode de Branch and Reduce. C'est la raison pour laquelle le minorant est obtenu en utilisant une méthode directe.

Afin d'obtenir un modèle en dimension finie on applique toujours la collocation directe, dont les principes ont été décrits dans le chapitre 1. Les notations suivantes vont être utilisées tout au long de ce chapitre:

- t_i est le i -ème point de discrétisation: $t_i = ih$, où $h = t_f/N$ est le pas de la discrétisation.
- x dénote les variables d'état et $x_i = x(t_i)$ est le vecteur contenant les variables d'état à l'instant t_i . (Par exemple $x_i = (x_i, y_i, v_i^x, v_i^y)$ dans le cas du problème spatial 2D exprimé en coordonnées cartésiennes).
- \bar{x} est le vecteur des variables d'état aux points milieux: $\bar{x}_i = x(\frac{t_i+t_{i+1}}{2})$
- u dénote le contrôle et u_i est le vecteur contrôle à l'instant t_i (ce vecteur est toujours de dimension 2 dans le cas du problème spatial 2D: $u_i = (u_i^x, u_i^y)$).
- F dénote le vecteur des seconds membres des équations différentielles et F_i est ce même vecteur pris à l'instant t_i . Dans le cas du problème spatial 2D on peut écrire en coordonnées cartésiennes:

$$F_i(x, y, v_i^x, v_i^y, u^x, u^y) = \begin{pmatrix} F_i^x(x, y, v_i^x, v_i^y, u^x, u^y), F_i^y(x, y, v_i^x, v_i^y, u^x, u^y), \\ F_i^{v^x}(x, y, v_i^x, v_i^y, u^x, u^y), F_i^{v^y}(x, y, v_i^x, v_i^y, u^x, u^y) \end{pmatrix}.$$

- \bar{F} est le vecteur des seconds membres des équations différentielles calculé aux points milieux $\bar{F}_i = \dot{x}(\bar{x}, u)$.

Le système d'équations ainsi obtenu peut-être divisé en deux parties :

- Les équations linéaires (et donc convexes) :

$$\begin{aligned} -\frac{3}{2h}x_i + \frac{3}{2h}x_{i+1} - \frac{1}{4}F_i + \frac{1}{4}F_{i+1} - \bar{F}_i &= 0 \\ \frac{1}{2}x_i + \frac{1}{2}x_{i+1} + \frac{h}{8}F_i - \frac{h}{8}F_{i+1} - \bar{x}_i &= 0 \end{aligned} \quad (4.14)$$

- Les équations non-linéaires directement issues de la dynamique du système :

$$\begin{aligned} F_i &= \frac{dx}{dt}(t_i, x_i, u_i) \\ \bar{F}_i &= \frac{dx}{dt}\left(\frac{t_i+t_{i+1}}{2}, \bar{x}_i, \bar{u}_i\right). \end{aligned} \quad (4.15)$$

4.3 Convexification

Les seules contraintes non-linéaires sont les équations décrivant l'évolution du système. La résolution du problème d'optimisation ainsi obtenu nous donne une solution locale, donc un majorant du minimum global. Par la suite on va donner différentes méthodes pour trouver un minorant; pour cela, on relaxera le problème en convexifiant tous les termes non-convexes qui apparaissent dans le système d'équations (4.15). Cette méthode de convexification a été décrite auparavant et va être maintenant analysée pour ce problème de transfert orbital 2D à masse constante. Pour chaque modèle qu'on propose, on présente les différentes façons de le convexifier et les résultats obtenus dans chacun des cas. Une analyse comparative de ces résultats est aussi effectuée. Le concept qu'on va utiliser est la reformulation du problème, c'est à dire la recherche du meilleur modèle, celui se prêtant le mieux à une bonne convexification.

4.3.1 Coordonnées cartésiennes

Le premier modèle qu'on a analysé est celui du problème de transfert orbital 2D à masse constante exprimé en coordonnées cartésiennes. Les variables d'état sont la position du mobile en 2D (x, y) et sa vitesse (v^x, v^y) . Le contrôle est à son tour décomposé dans le repère XY : $\mathbf{u} = (u^x, u^y)$. Le problème de contrôle optimal se met sous la forme:

$$\left\{ \begin{array}{ll} \min_{\mathbf{r}, \mathbf{v}, \mathbf{u}} & \int_0^{t_f} [u^x(t)]^2 + [u^y(t)]^2 dt \\ \text{s.t.} & \dot{x}(t) = v^x(t) \quad t \in [0, t_f] \\ & \dot{v}^x(t) = -\mu \frac{x(t)}{(x^2(t) + y^2(t))^{\frac{3}{2}}} + \frac{1}{m} u^x(t) \quad t \in [0, t_f] \\ & \dot{y}(t) = v^y(t) \quad t \in [0, t_f] \\ & \dot{v}^y(t) = -\mu \frac{y(t)}{(x^2(t) + y^2(t))^{\frac{3}{2}}} + \frac{1}{m} u^y(t) \quad t \in [0, t_f] \\ & [u^x(t)]^2 + [u^y(t)]^2 \leq T_U^2 \quad t \in [0, t_f] \\ & (x, y, v^x, v^y)(t=0) = (x_0, y_0, v_0^x, v_0^y) \\ & (x, y, v^x, v^y)(t=t_f) = (x_f, y_f, v_f^x, v_f^y) \end{array} \right. \quad (4.16)$$

Les travaux antérieurs effectués en collaboration avec le CNES sur les méthodes locales ont montré que les coordonnées cartésiennes ne sont pas le meilleur choix pour résoudre le problème de contrôle spatial. Cela est principalement dû au caractère oscillatoire des variables d'état (voir figure 4.5). En effet, même en l'absence du contrôle, le satellite reste sur la même orbite et ses caractéristiques cartésiennes varient (position et vitesse du satellite sur orbite ne sont pas constantes). Du point de vue convexification, deux difficultés sont la conséquence de cette représentation cartésienne. D'une part, le caractère fortement variable pour les états nous empêche de faire des suppositions sur le domaine de variation de ces variables; cela veut dire

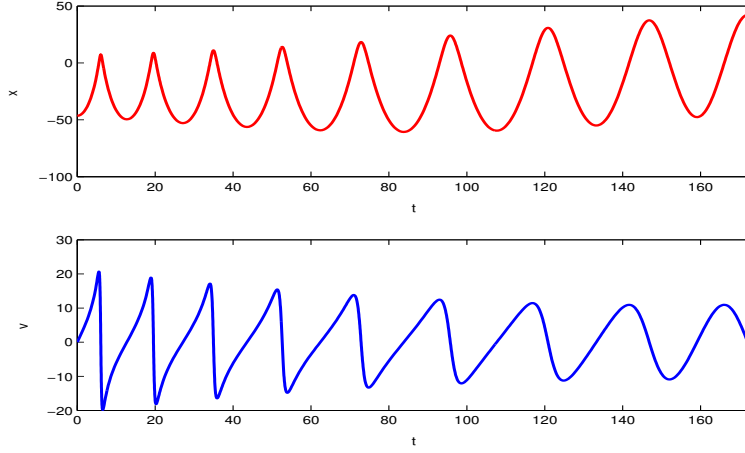


FIG. 4.5 – *Solution locale pour le problème de transfert orbital 2D à masse constante. Obtenu par une méthode directe, ce graphe montre le caractère oscillatoire des variables d'état exprimées en coordonnées cartésiennes. Le graphe du haut donne la composante x de la position en fonction du temps et le graphe du bas la composante v^x de la vitesse en fonction du temps.*

qu'on manque de précision quant aux bornes des intervalles utilisés, ce qui va nécessairement affecter la qualité de la convexification. D'autre part, l'obtention du majorant dans ce cas est plus difficile. En fait, afin de résoudre le cas spatial (en utilisant des méthodes directes ou indirectes, d'ailleurs) on préfère utiliser les coordonnées de Gauss, cela pose moins de problèmes numériques.

On peut alors se demander quel est l'intérêt de convexifier une telle représentation du problème. La réponse est donnée par l'apparition linéaire du contrôle dans les équations (4.16). On peut facilement remarquer que les seuls termes non-linéaires (nécessitant donc une convexification) sont ceux en x et y . Cette forme particulière du système nous a laissé penser qu'une bonne convexification (qui ne comporte pas beaucoup de variables) peut-être obtenue dans cette modélisation cartésienne.

Comme on l'a déjà précisé on utilise la collocation directe pour obtenir un problème d'optimisation en dimension finie. Avec les considérations et notations données en §4.2.1, on peut écrire le problème d'optimisation suivant :

$$\begin{aligned}
& \min \sum_{i=0}^N (u_i^x)^2 + (u_i^y)^2 \\
& -\frac{3}{2h}x_i + \frac{3}{2h}x_{i+1} - \frac{1}{4}F_i^x - \frac{1}{4}F_{i+1}^x - \bar{F}_i^x = 0 \quad i = 0, N-1 \\
& -\frac{3}{2h}y_i + \frac{3}{2h}y_{i+1} - \frac{1}{4}F_i^y - \frac{1}{4}F_{i+1}^y - \bar{F}_i^y = 0 \quad i = 0, N-1 \\
& -\frac{3}{2h}v_i^x + \frac{3}{2h}v_{i+1}^x - \frac{1}{4}F_i^{vx} - \frac{1}{4}F_{i+1}^{vx} - \bar{F}_i^{vx} = 0 \quad i = 0, N-1 \\
& -\frac{3}{2h}v_i^y + \frac{3}{2h}v_{i+1}^y - \frac{1}{4}F_i^{vy} - \frac{1}{4}F_{i+1}^{vy} - \bar{F}_i^{vy} = 0 \quad i = 0, N-1 \\
& \frac{1}{2}x_i + \frac{1}{2}x_{i+1} + \frac{h}{8}F_i^x - \frac{h}{8}F_{i+1}^x - \bar{F}_i^x = 0 \quad i = 0, N-1 \\
& \frac{1}{2}y_i + \frac{1}{2}y_{i+1} + \frac{h}{8}F_i^y - \frac{h}{8}F_{i+1}^y - \bar{F}_i^y = 0 \quad i = 0, N-1 \\
& \frac{1}{2}v_i^x + \frac{1}{2}v_{i+1}^x + \frac{h}{8}F_i^{vx} - \frac{h}{8}F_{i+1}^{vx} - \bar{F}_i^{vx} = 0 \quad i = 0, N-1 \\
& \frac{1}{2}v_i^y + \frac{1}{2}v_{i+1}^y + \frac{h}{8}F_i^{vy} - \frac{h}{8}F_{i+1}^{vy} - \bar{F}_i^{vy} = 0 \quad i = 0, N-1 \quad (4.17) \\
& F_i^x - v_i^x = 0 \quad i = 0, N \\
& F_i^y - v_i^y = 0 \quad i = 0, N \\
& F_i^{vx} - \mu \frac{x_i}{(x_i^2 + y_i^2)^{\frac{3}{2}}} - \frac{1}{m}u_i^x = 0 \quad i = 0, N \\
& F_i^{vy} - \mu \frac{y_i}{(x_i^2 + y_i^2)^{\frac{3}{2}}} - \frac{1}{m}u_i^y = 0 \quad i = 0, N \\
& \bar{F}_i^x - \bar{v}_i^x = 0 \quad i = 0, N \\
& \bar{F}_i^y - \bar{v}_i^y = 0 \quad i = 0, N \\
& \bar{F}_i^{vx} - \mu \frac{\bar{x}_i}{(\bar{x}_i^2 + \bar{y}_i^2)^{\frac{3}{2}}} - \frac{1}{m}u_i^x = 0 \quad i = 0, N \\
& \bar{F}_i^{vy} - \mu \frac{\bar{y}_i}{(\bar{x}_i^2 + \bar{y}_i^2)^{\frac{3}{2}}} - \frac{1}{m}u_i^y = 0 \quad i = 0, N
\end{aligned}$$

On remarque que la plupart des équations dans le problème (4.17) sont linéaires, ce qui confirme notre intérêt pour ce modèle. En effet, les seuls termes non-linéaires, qui donnent la non-convexité du problème, sont du type $\frac{x}{(x^2+y^2)^{\frac{3}{2}}}$ et $\frac{y}{(x^2+y^2)^{\frac{3}{2}}}$. Ces termes présentent néanmoins plusieurs inconvénients. D'une part leur valeur (comme fonction de x et y) tend vers l'infini quand $r = \sqrt{x^2 + y^2}$ tend vers 0. Cela rend les termes très difficiles à convexifier; c'est même impossible en l'état, car autour du point $(x, y) = (0, 0)$ la fonction n'admet pas d'enveloppe concave. La solution qu'on peut envisager dans un premier temps est de considérer la valeur de la fonction comme étant 0 sur une région qui ne présente pas d'intérêt physique (il est clair que la région $r \leq R_T$ n'est pas faisable pour une trajectoire du satellite, R_T étant le rayon de la Terre). On remplace alors cette fonction par la fonction suivante (R étant une constante convenablement choisie):

$$\begin{aligned}
& q : [-r^U, r^U] \times [-r^U, r^U] \rightarrow \mathbb{R} \\
& q(x, y) = \begin{cases} \frac{x}{(x^2+y^2)^{\frac{3}{2}}} & , x^2 + y^2 \geq R^2 \\ 0 & , x^2 + y^2 \leq R^2 \end{cases} \quad (4.18)
\end{aligned}$$

La figure suivante montre la fonction donnée en (4.18). Malgré le fait qu'il est dorénavant possible de la convexifier maintenant, il reste deux inconvénients difficilement surmontables: le premier est que la qualité de la

convexification dépend fortement du choix de la constante R ; en fait les plus grandes valeurs de la fonction s'obtiennent autour du cercle $x^2 + y^2 = R^2$, les régions éloignées ayant une valeur beaucoup plus faible. Le deuxième est que la convexification introduit de l'espace de recherche inutile car la région $r \leq R$ ne présente pas d'intérêt; elle devrait tout simplement être exclue du domaine de recherche.

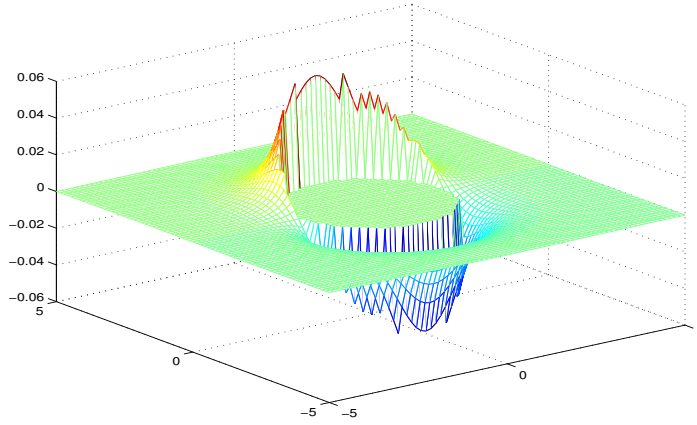


FIG. 4.6 – Le terme $\frac{x}{(x^2+y^2)^{\frac{3}{2}}}$ sur le domaine $[-5,5] \times [-5,5]$. Comme le domaine de définition contient une région qui ne présente pas d'intérêt pour la recherche des minima et que la fonction tend vers l'infini quand $r \rightarrow 0$ on a considéré 0 comme valeur de la fonction pour ce domaine.

La solution qu'on propose afin de s'affranchir de ces difficultés est de passer en coordonnées polaires une fois la discrétisation effectuée. Cela revient à remplacer le couple (x_i, y_i) par (r_i, α_i) , où r_i et α_i sont le module du rayon vecteur et l'angle polaire à l'instant t_i . Les formules de passage d'un modèle à l'autre sont :

$$\begin{aligned} x &= r \cos \alpha \\ y &= r \sin \alpha, \end{aligned} \quad (4.19)$$

de telle sorte que les termes non-linéaires à convexifier ont la forme suivante:

$$\begin{aligned} a \frac{\sin \alpha}{r^2} + br \sin \alpha \\ a \frac{\cos \alpha}{r^2} + br \cos \alpha \end{aligned} \quad (4.20)$$

où a et b sont deux constantes dépendantes du pas de la discrétisation h .

Modèle convexe. Il s'agit ici de trouver les enveloppes convexes pour les termes donnés en (4.20). Les fonctions convexes (respectivement concaves) qu'on recherche sont de la forme:

$$\begin{aligned} q^{sup}(\alpha, r) &= s_1 \alpha^2 + s_2 \alpha + s_3 r + s_4 \\ q^{inf}(\alpha, r) &= i_1 \alpha^2 + i_2 \alpha + i_3 r + i_4 \end{aligned} \quad (4.21)$$

où les constantes s_1, s_2, s_3, s_4 et i_1, i_2, i_3, i_4 restent à être déterminées en fonction des limites d'intervalle $(\alpha^L, \alpha^U, r^L, r^U)$ sur lesquels les fonctions (4.20) sont à convexifier.

Avoir choisi des fonctions de la forme (4.21) pour les enveloppes convexes et concaves est justifié par le fait que l'intégration des contraintes logiques dans un tel système peut-être traitée par notre variante de l'algorithme GBD. En effet, ce type de contrainte (quadratique séparable) donne au problème la structure particulière dont on a besoin pour prendre en compte les contraintes logiques. D'autre part, les variations des fonctions (4.20) par rapport au r sont faibles et sont bien approchées par des fonctions linéaires en r . De même, les variations par rapport à α de ces mêmes fonctions sont bien sous-estimées (sur-estimées) par des fonctions quadratiques. La figure (4.7) montre ces termes à convexifier sur le domaine $[0, 2\pi] \times [10, 30]$.

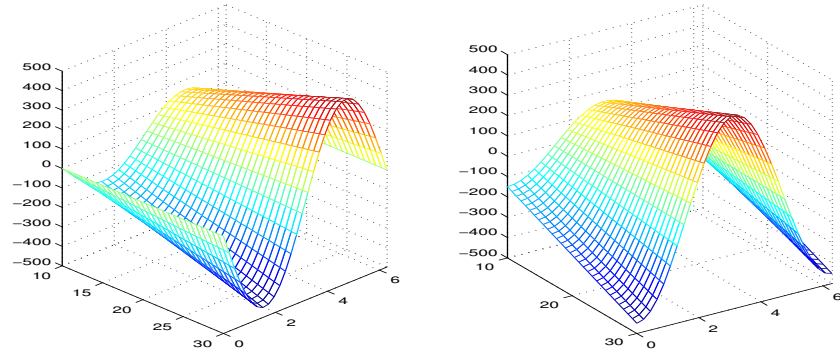


FIG. 4.7 – A gauche la fonction $a \frac{\sin \alpha}{r^2} + br \sin \alpha$ sur le domaine $[0, 2\pi] \times [10, 30]$. A droite la fonction $a \frac{\cos \alpha}{r^2} + br \cos \alpha$ sur ce même domaine. On remarque que ces fonctions (fortement non-linéaires) ont une faible variation en r (ce qui justifie les fonctions linéaires en r pour les convexifier) et une variation importante en α (d'où notre choix des fonctions quadratiques en α pour les enveloppes convexes et concaves)

La convexification des fonctions en sin et cos doit se faire pour tout domaine $D \subseteq [0, 2\pi] \times [10, 30]$. Autrement dit, afin d'avoir une convexification efficace, on doit analyser chaque cas possible ($[\alpha^L, \alpha^U] \subseteq [0, 2\pi]$). Plusieurs cas ont été identifiés et traités (le nombre total de convexifications différentes avoisine 70). Pour chaque cas, il s'agit d'expliciter les coefficients s_1, s_2, s_3, s_4 et i_1, i_2, i_3, i_4 en fonction de $\alpha^L, \alpha^U, r^L, r^U$. Afin de montrer la façon dont ces expressions sont obtenues, on détaille la procédure sur un cas particulier, celui du calcul de l'enveloppe convexe du terme $a \frac{\sin \alpha}{r^2} + br \sin \alpha$ sur $[\alpha^L, \alpha^U] \cup [r^L, r^U]$ avec $\alpha^L \in [0, \frac{\pi}{2}]$ et $\alpha^U \in [\pi, 2\pi]$.

Les coefficients i_1, i_2, i_3, i_4 s'obtiennent en utilisant les observations sui-

vantes :

- les minima des fonctions $q(\alpha, r)$ et $q^{inf}(\alpha, r)$ coïncident;

$$\begin{aligned} (q^{inf})'_\alpha(\frac{\pi}{2}, r^U) &= 0 \\ q^{inf}(\frac{\pi}{2}, r^U) - q(\frac{\pi}{2}, r^U) &= 0 \end{aligned}$$

- les fonction $q(\alpha, r)$ et $q^{inf}(\alpha, r)$ prennent la même valeur en (α^U, r^L) ;

$$q^{inf}(\alpha^U, r^L) - q(\alpha^U, r^L) = 0$$

- les fonctions $q(\alpha, r)$ et $q^{inf}(\alpha, r)$ ont la même tangente par rapport à r en $(\frac{\pi}{2}, r^U)$;

$$(q^{inf})'_r(\frac{\pi}{2}, r^U) - q'_r(\frac{\pi}{2}, r^U) = 0.$$

Ces quatre équations permettent d'obtenir les expressions explicites des paramètres i_1, i_2, i_3, i_4 en fonction de $r^U, r^L, \alpha^U, \alpha^L$. Les enveloppes convexes et concaves ainsi obtenues sont représentées sur le graphe (4.8).

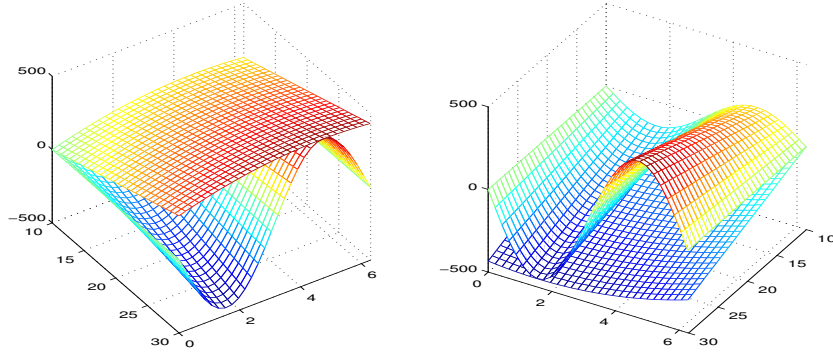


FIG. 4.8 – A gauche la fonction $a \frac{\sin \alpha}{r^2} + br \sin \alpha$ sur le domaine $[0, 2\pi] \times [10, 30]$ et son enveloppe concave. A droite la même fonction et son enveloppe convexe. Les enveloppes ont été obtenues en utilisant les équations présentées ci-avant.

Résultats pour le modèle cartésien

La même analyse que celle discutée auparavant est effectuée pour chaque cas possible. Afin d'obtenir les expressions analytiques des paramètres s et i , le calcul symbolique de Matlab a été utilisé; le logiciel d'optimisation non-linéaire KNITRO nous a permis de résoudre la relaxation convexe.

Les tests qu'on a effectués concernent un transfert orbital simple, chaque composante discrète r_i se trouvant dans le domaine $[10, 30]$ et chaque α_i dans l'intervalle $[0, 2\pi]$. Les intervalles utilisées pour la vitesse sont de la forme

$[-v^U, v^U]$ où v^U est la vitesse maximale obtenue en résolvant le problème spatial avec une méthode locale. La poussée maximale est de $60N$.

Dans tous les tests, l'objectif optimal de la relaxation convexe est 0!

Quels que soient les intervalles choisis pour borner les variables discrètes, il existe donc toujours un contrôle nul capable de satisfaire les contraintes du problème convexe.

Cette mauvaise limite inférieure peut-être expliquée en regardant de plus près l'équation de la dynamique qu'on est en train de convexifier:

$$\ddot{\mathbf{r}} = -\mu \frac{\mathbf{r}}{r^3} + \frac{1}{m} \mathbf{u}$$

Le terme le plus important en ce qui concerne l'ordre de grandeur est le terme en r , l'attraction terrestre. En effet ce terme est beaucoup plus grand que le contrôle qu'on utilise pour déplacer le satellite d'une orbite à l'autre, le contrôle ayant plutôt le rôle d'une perturbation:

$$|\mu \frac{\mathbf{r}}{r^3}| \gg |\frac{1}{m} \mathbf{u}|$$

Or, c'est justement pour ce terme en r qu'on cherche les enveloppes convexes et concaves q^{sup} et q^{inf} . Ce qui se traduit dans la relaxation convexe par le fait que les équations sont satisfaites même pour un contrôle nul, car

$$q^{sup} \gg |\frac{1}{m} \mathbf{u}| \quad \text{et} \quad q^{inf} \gg |\frac{1}{m} \mathbf{u}|$$

et les équations sont trop relaxées. Du point de vue physique, les équations n'expriment pas le fait que le satellite se trouve sur une orbite, mais quelque part autour d'une orbite; cela veut dire que le satellite peut passer d'une orbite à une autre sans qu'on contrôle le système.

Ces résultats, bien qu'infructueux en ce qui concerne la qualité de la limite inférieure, permettent d'exprimer une propriété cruciale que doit posséder la relaxation convexe:

Les contraintes du problème convexe, relaxation du problème spatial 2D à masse constante, doivent assurer le fait qu'en l'absence du contrôle le satellite reste sur la même orbite

Or cette propriété s'exprime naturellement en utilisant les coordonnées de Gauss, étant donnée qu'une partie des variables d'état sont des intégrales premières de mouvement. Les paragraphes suivants s'intéressent à ce type de modélisation.

4.3.2 Équations de Gauss en temps

Le principal avantage des équations du mouvement exprimées en coordonnées de Gauss est le fait d'utiliser, parmi les variables d'état, des intégrales premières du mouvement, qui expriment naturellement le fait que **si l'on ne contrôle pas le satellite celui-ci reste sur la même orbite**, une propriété que l'on va essayer de maintenir dans la relaxation convexe.

Équations du mouvement et variables d'état. Les nouvelles variables d'état sont (voir figure (4.9)):

- P , le paramètre de la conique; il est l'un des paramètres décrivant la forme de l'orbite
- e_x et e_y , composantes du vecteur excentricité; elles décrivent la forme de l'orbite (l'excentricité) mais aussi sa position dans le plan
- L , la longitude cumulée; elle donne la position du satellite sur l'orbite.

Quant au contrôle, il est exprimé dans le repère ortho-radial (figure 4.9).

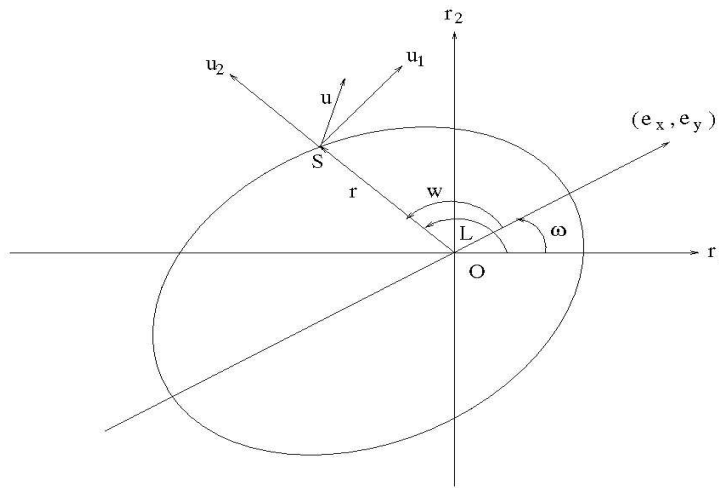


FIG. 4.9 – *Interprétation physique des paramètres orbitaux.* P , e_x et e_y modélisent la forme et la position de l'orbite dans le plan. Naturellement, si à l'instant t aucune poussée n'est exercée sur le satellite ces trois paramètres sont constants (l'orbite et sa position dans le plan ne souffrent aucune modification). L , qu'on va appeler la longitude cumulée précise la position du satellite sur l'orbite. Le contrôle (vecteur \mathbf{u}) est décomposé dans le repère ortho-radial: une composante le long du rayon vecteur \mathbf{r} et une composante normale à ce même vecteur.

Avec ces nouveaux paramètres les équations de mouvement du satellite s'écrivent:

$$\begin{aligned}
 \dot{P} &= \frac{2}{m\sqrt{\mu}} \frac{P^{\frac{3}{2}}}{W} u_2 \\
 \dot{e}_x &= \frac{1}{m\sqrt{\mu}} \sin LP^{\frac{1}{2}} u_1 + \frac{1}{m\sqrt{\mu}} P^{\frac{1}{2}} \frac{Q_1}{W} u_2 \\
 \dot{e}_y &= -\frac{1}{m\sqrt{\mu}} \cos LP^{\frac{1}{2}} u_1 + \frac{1}{m\sqrt{\mu}} P^{\frac{1}{2}} \frac{Q_2}{W} u_2 \\
 \dot{L} &= \sqrt{\mu} \frac{W^2}{P^{\frac{3}{2}}}
 \end{aligned} \tag{4.22}$$

où

$$\begin{aligned}
W &= 1 + e_x \cos L + e_y \sin L \\
Q_1 &= 2 \cos L + e_x(1 + \cos^2 L) + e_y(\sin L \cos L) \\
Q_2 &= 2 \sin L + e_x(\sin L \cos L) + e_y(1 + \sin^2 L)
\end{aligned} \tag{4.23}$$

Une première remarque concerne la variable W , donnée par les équations (4.23). Il est important dans la convexification d'obtenir des limites inférieures et supérieures sur cette variable. En explicitant l'expression de W en fonction de l'excentricité $e = \sqrt{e_x^2 + e_y^2}$, on obtient:

$$\begin{cases} e_x = e \cos \omega \\ e_y = e \sin \omega \end{cases} \Rightarrow W = 1 + e \cos(L - \omega) \tag{4.24}$$

et alors $W \in [1 - e^{max}, 1 + e^{max}]$. Comme $e \in [0, 1]$ alors $W > 0$ et les limites inférieures sur la variable W peuvent être obtenues en partant des limites d'intervalle pour les variables e_x et e_y . D'autre part, ces limites ne sont pas très précises et on va souvent être obligé de se contenter de cette approximation assez grossière; en fait, la seule façon de mieux borner W et de prendre en compte les contraintes de boîte sur la variable L (L^L, L^U). Malgré la meilleure précision obtenue de cette façon, cela complique considérablement les choses, en ajoutant un nouveau paramètre (L) à prendre en compte dans le Branch and Bound.

Cela nous amène aussi à un problème plus général: les termes en $\sin L$ et $\cos L$ sont très difficiles à convexifier, étant donnée leur nature très oscillatoire. Ce même type de fonction nous a posé des problèmes lors de la convexification du cas cartésien (on se rappelle le changement de variables et le cas polaire).

On remarque en regardant les équations (4.22) que si $|u| = 0$ alors $\dot{P} = 0$, $\dot{e}_x = 0$ et $\dot{e}_y dt = 0$, donc le satellite reste sur la même orbite. C'est la une propriété très importante du système qu'on va essayer de préserver lors de la convexification. Pour cela, considérons le cas général suivant:

$$F(x, u) = a(x)u_1 + b(x)u_2 \tag{4.25}$$

(ici F doit s'annuler quand $|u| = \sqrt{u_1^2 + u_2^2} = 0$)

On veut donc convexifier cette équation tout en gardant le fait que $|u| = 0 \Rightarrow F = 0$. On propose trois façons différentes de le faire:

i) en utilisant le module du contrôle ($|u|$):

$$\begin{aligned}
F &= a(x)u_1 + b(x)u_2 \Rightarrow |F| = |a(x)u_1 + b(x)u_2| \Rightarrow \\
|F| &\leq |a(x)||u_1| + |b(x)||u_2|
\end{aligned} \tag{4.26}$$

et la relaxation ainsi obtenue:

$$\begin{cases} F \leq |a(x)|^{max}|u_1| + |b(x)|^{max}|u_2| \\ F \geq -|a(x)|^{max}|u_1| - |b(x)|^{max}|u_2| \end{cases} \tag{4.27}$$

Cela revient à travailler avec des nouvelles variables de contrôle: $|u_1|$ et $|u_2|$. Il est facile maintenant de vérifier qu'un contrôle nul ne déplace pas le satellite sur une autre orbite.

ii) en utilisant les parties positives et négatives du contrôle:

$$\begin{cases} u_1 = u_1^+ - u_1^- \\ u_2 = u_2^+ - u_2^- \end{cases} \quad u_1^+, u_1^-, u_2^+, u_2^- \geq 0 \quad (4.28)$$

En effet, si la convexification du terme $a(x)u_1$ quand $u_1 \in [-T_{max}, T_{max}]$ n'assure pas que ce terme est nul quand $u_1 = 0$, la convexification du terme $a(x)u_1^+$ quand $u_1 \in [0, T_{max}]$ assure, elle, cette propriété. Afin de vérifier cela, le lecteur peut utiliser les équations de McCormick données en §4.1.2, formules (4.6).

Il est important de remarquer que ces substitutions de variable (passage de u_1, u_2 à $u_1^+, u_1^-, u_2^+, u_2^-$) ne changent pas la forme quadratique de l'objectif, son terme principal à l'instant t_i s'écrivant $(u_1^+)_i^2 + (u_1^-)_i^2 + (u_2^+)_i^2 + (u_2^-)_i^2$.

iii) en utilisant les variables booléennes: soit $y_1 \in \{0,1\}$ avec $y_1 = 0$ si $u_1 \leq 0$ et $y_1 = 1$ si $u_1 > 0$. De même, une variable binaire $y_2 \in \{0,1\}$ est associée à u_2 . Les deux équations suivantes expriment le rôle des variables binaires y_1, y_2 :

$$\begin{aligned} (2y_1 - 1)u_1 &\geq 0 \\ (2y_2 - 1)u_2 &\geq 0 \end{aligned} \quad (4.29)$$

Avec cela, la relaxation devient:

$$\begin{aligned} u_1[y_1 a^L(x) + (1 - y_1)a^U(x)] &+ u_2[y_2 b^L(x) + (1 - y_2)b^U(x)] &\leq F \\ u_2[y_1 a^U(x) + (1 - y_1)a^L(x)] &+ u_2[y_2 b^U(x) + (1 - y_2)b^L(x)] &\leq F \end{aligned} \quad (4.30)$$

La complexité des équations (4.29) et (4.30), surtout due au fait qu'on introduit des variables binaires, rendant ainsi le problème combinatoire, nous a amené par la suite à prendre en compte seulement les deux premiers modèles, i) et ii), pour convexifier le problème.

On conclut cette partie préalable dédiée à la convexification en regardant la solution locale obtenue pour le problème de contrôle spatial à masse constante, solution obtenue par une méthode directe. La figure (4.10) représente les évolutions des paramètres orbitaux en fonction du temps. Les faibles oscillations des paramètres P, e_x, e_y et L vont nous permettre de faire des suppositions quant aux bornes d'intervalle, en se basant sur la forme de la solution locale. C'est là un avantage qu'on n'avait pas en utilisant le modèle cartésien. La solution globale et la relaxation convexe du problème de transfert orbital 2D est cherchée sur le domaine réduit:

$$\begin{aligned} P_i &\in [P_i^{loc} - \Delta P, P_i^{loc} + \Delta P] &, \quad i = 0, N \\ (e_x)_i &\in [(e_x)_i^{loc} - \Delta e_x, (e_x)_i^{loc} + \Delta e_x] &, \quad i = 0, N \\ (e_y)_i &\in [-e_y^U, e_y^U] &, \quad i = 0, N \\ L &\in [L_i^{loc} - \Delta L, L_i^{loc} + \Delta L] &, \quad i = 0, N \end{aligned}$$

où l'indice loc fait référence à la solution locale, ΔP , Δe_x , e_y^U et ΔL sont des constantes judicieusement choisies, qui caractérisent la globalité de la solution.

Afin de pouvoir comparer les différentes convexifications entre elles, on compare les valeurs des objectifs des relaxations convexes. Mais une autre comparaison est aussi très intéressante: le minorant, exprimé comme fonction des paramètres ΔP , Δe_x et ΔL est une fonction décroissante. En effet, quand ΔP , Δe_x et ΔL tendent vers 0, la limite inférieure tend vers sa valeur maximale, la solution locale du problème. Cette fonction peut-être alors utilisée comme une mesure de la qualité de la relaxation convexe. De plus, si on prend la valeur relative par rapport à la solution locale du problème de transfert orbital à masse constante (UBD), on a une mesure quantitative de la limite inférieure:

$$\frac{LBD}{UBD} = fonction(\Delta P, \Delta e_x, \Delta L)$$

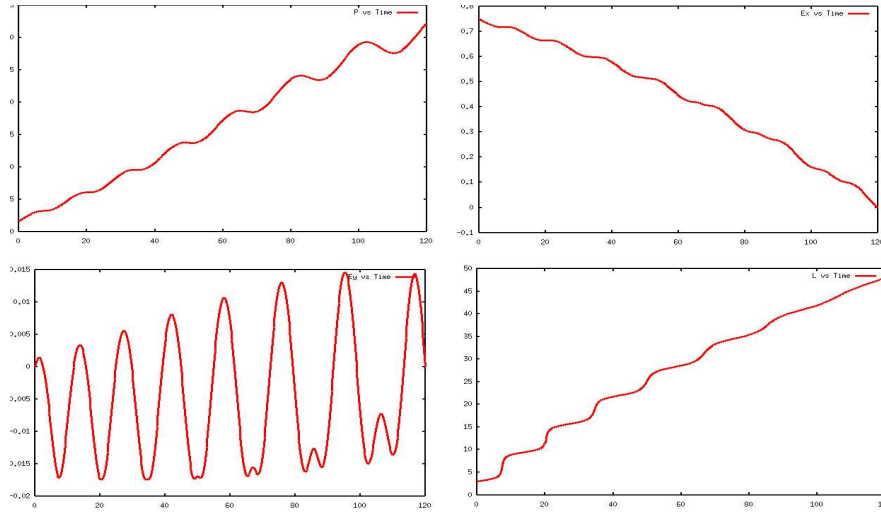


FIG. 4.10 – Une solution locale du problème de transfert orbital 2D à masse constante. En haut à gauche P en fonction du temps, les unités de mesure étant le Mégamètre (Mm) pour P et l'heure (h) pour le temps. En haut à droite e_x en fonction du temps, en bas à gauche e_y en fonction du temps et en bas à droite L en fonction du temps. Les oscillations sont petites, ce qui nous permet de faire de bonnes suppositions quant au minimum global, cela jouant en notre faveur pour la convexification car on réduit la taille des intervalles de variation pour les paramètres orbitaux.

Un premier modèle

On commence l'étude de la convexification des équations de Gauss en fonction du temps par un modèle simplifié. On regarde seulement les équations donnant la variation de P et L . Notre choix est motivé, d'une part, par le fait que ces équations ne contiennent pas de termes en $\sin L$ et $\cos L$, termes difficiles à bien convexifier comme on l'a vu dans le cas des équations cartésiennes. D'autre part, en simplifiant de cette façon le système d'équations on diminue le nombre de variables, en obtenant ainsi une relaxation convexe plus simple à résoudre.

Les équations prises en compte sont donc:

$$\begin{aligned}\frac{dP}{dt} &= \frac{2}{m\sqrt{\mu}} \frac{P^{\frac{3}{2}}}{W} u_2 \\ \frac{dL}{dt} &= \sqrt{\mu} \frac{W^2}{P^{\frac{3}{2}}} \\ W &= 1 + e_x \cos L + e_y \sin L\end{aligned}\tag{4.31}$$

On remarque que seule la composante u_2 du contrôle est considérée dans ce modèle. Cela affecte forcément la valeur de la limite inférieure donnée par la relaxation convexe (le terme $\int u_1^2(t)dt$ n'est pas pris en compte dans l'objectif), mais pas de façon dramatique. En fait, en analysant la contribution du terme en u_1 dans l'objectif, on s'est aperçu qu'elle dépasse rarement les 12% de la valeur totale de l'objectif (pour le jeu de conditions initiales et terminales qu'on utilise habituellement).

La démarche qu'on propose pour obtenir une relaxation convexe peut-être synthétisée comme suit:

- utiliser les intervalles des variables e_x et e_y pour déduire des intervalles sur la variables W à chaque instant t_i à l'aide des formules:

$$\begin{aligned}W^L &= 1 - e^U \\ W^U &= 1 + e^U \\ e^U &= \max_{e_x \in [e_x^L, e_x^U], e_y \in [e_y^L, e_y^U]} \sqrt{e_x^2 + e_y^2}\end{aligned}\tag{4.32}$$

(les indices i ayant été omis pour plus de clarté).

- les intervalles sur P et W sont utilisés pour limiter la variation de la variable d'état P , en utilisant le module du contrôle:

$$|F^P| = \left| \frac{dP}{dt} \right| = \frac{2}{m\sqrt{\mu}} \frac{P^{\frac{3}{2}}}{W} |u_2| \Rightarrow |F^P| \leq \frac{2}{m\sqrt{\mu}} \frac{(P^U)^{\frac{3}{2}}}{W^L} |u_2| \tag{4.33}$$

et donc la relaxation:

$$\begin{cases} F^P & \leq \frac{2}{m\sqrt{\mu}} \frac{(P^U)^{\frac{3}{2}}}{W^L} |u_2| \\ F^P & \geq -\frac{2}{m\sqrt{\mu}} \frac{(P^U)^{\frac{3}{2}}}{W^L} |u_2| \end{cases} \tag{4.34}$$

(encore une fois, l'omission des indices i est intentionnelle). La même démarche est utilisée pour obtenir les relaxations sur les variables \bar{F}^P , le second membre des équations en P aux points milieux de chaque intervalle de discrétisation, $\frac{t_i+t_{i+1}}{2}$;

- soit $\text{lin sup}(P^{-\frac{3}{2}})$ et $\text{lin inf}(P^{-\frac{3}{2}})$ les enveloppes concaves et convexes **linéaires** de la fonction $f : [P^L, P^U] \rightarrow \mathbb{R}$ $f(P) = P^{-\frac{3}{2}}$ (la façon dont ses enveloppes sont obtenues est illustrée sur la figure 4.3). Alors l'équation en L est relaxée en utilisant les formules suivantes ($F^L = \frac{dL}{dt}$):

$$\begin{cases} F^L - \sqrt{\mu}(W^L)^2 \text{lin sup}(P^{-\frac{3}{2}}) & \geq 0 \\ F^L - \sqrt{\mu}(W^U)^2 \text{lin inf}(P^{-\frac{3}{2}}) & \leq 0 \end{cases} \quad (4.35)$$

En utilisant ces différentes remarques le problème suivant est une relaxation du problème spatial 2D à masse constante:

$$\begin{aligned} & \min \sum_{i=0}^N |(u_2)_i|^2 \\ (C) \quad & \begin{cases} -\frac{3}{2h}P_i + \frac{3}{2h}P_{i+1} - \frac{1}{4}F_i^P - \frac{1}{4}F_{i+1}^P - \bar{F}_i^P & = 0 & i = 0, N-1 \\ -\frac{3}{2h}L_i + \frac{3}{2h}L_{i+1} - \frac{1}{4}F_i^L - \frac{1}{4}F_{i+1}^L - \bar{F}_i^L & = 0 & i = 0, N-1 \\ \frac{1}{2}P_i + \frac{1}{2}P_{i+1} + \frac{h}{8}F_i^P - \frac{h}{8}F_{i+1}^P - \bar{P}_i & = 0 & i = 0, N-1 \\ \frac{1}{2}L_i + \frac{1}{2}L_{i+1} + \frac{h}{8}F_i^L - \frac{h}{8}F_{i+1}^L - \bar{L}_i & = 0 & i = 0, N-1 \end{cases} \\ (M) \quad & \begin{cases} F_i^P - \frac{2}{m\sqrt{\mu}} \frac{(P_i^U)^{3/2}}{W_i^L} |(u_2)_i| & \leq 0 & i = 0, N \\ -F_i^P - \frac{2}{m\sqrt{\mu}} \frac{(\bar{P}_i^U)^{3/2}}{W_i^L} |(u_2)_i| & \leq 0 & i = 0, N \\ \bar{F}_i^P - \frac{2}{m\sqrt{\mu}} \frac{(\bar{P}_i^U)^{3/2}}{W_i^L} |(u_2)_i| & \leq 0 & i = 0, N \\ -\bar{F}_i^P - \frac{2}{m\sqrt{\mu}} \frac{(\bar{P}_i^U)^{3/2}}{W_i^L} |(u_2)_i| & \leq 0 & i = 0, N \end{cases} \\ (L) \quad & \begin{cases} F_i^L - \sqrt{\mu}(W_i^L)^2 \text{lin sup}(P^{-3/2}) & \geq 0 & i = 0, N \\ F_i^L - \sqrt{\mu}(W_i^U)^2 \text{lin inf}(P^{-3/2}) & \geq 0 & i = 0, N \\ \bar{F}_i^L - \sqrt{\mu}(\bar{W}_i^L)^2 \text{lin sup}(\bar{P}^{-3/2}) & \geq 0 & i = 0, N \\ \bar{F}_i^L - \sqrt{\mu}(\bar{W}_i^U)^2 \text{lin inf}(\bar{P}^{-3/2}) & \geq 0 & i = 0, N \end{cases} \\ (T) \quad & \{ |(u_2)_i| \leq T_U \quad i = 0, N \end{aligned} \quad (4.36)$$

où (C) est l'ensemble d'équations issues de la collocation, (M) les équations où la valeur absolue du contrôle a été utilisée pour assurer le fait que sans contrôle le satellite reste sur la même orbite et où (L) dénote l'ensemble d'équations relaxant l'évolution du paramètre d'état L . On rappelle que tous les termes utilisant la notation \bar{x} se réfèrent aux variables calculées aux points milieux des intervalles $(\frac{t_i+t_{i+1}}{2})$ pour x .

Résultats. La relaxation convexe décrite plus haut a été résolue à l'aide de la bibliothèque d'optimisation CPLEX, dans un algorithme qu'on a implanté

en langage C. Le solveur non-linéaire KNITRO calcule la solution locale (limite supérieure) du problème de départ. Le problème de test (transfert orbital 2D à masse constante) est caractérisé par les paramètres suivants:

$$\begin{aligned}
P^0 &= 36.625Mm & P^f &= 42.165Mm \\
e_x^0 &= 0.15 & e_x^f &= 0.00 \\
e_y^0 &= 0.00 & e_y^f &= 0.00 \\
L^0 &= 0.00 & L^f &= 100.00 \\
t^f &= 331.51h & T_U &= 30.00N & m &= 1500Kg
\end{aligned} \tag{4.37}$$

Le tableau suivant (4.3) présente les temps de calcul, correspondant aux différents nombres de points de discrétisation.

N_{dis}	N_{tot}	$t_{exec}(s)$	Rapport (%)
200	1796	0.2954	41.663
400	3596	0.6523	41.615
800	7196	1.4645	41.590
1000	8996	1.8967	41.585
1500	13496	3.5493	41.579
3000	26996	9.7214	41.574

TAB. 4.3 – Les temps d’exécution pour la relaxation convexe, problème quadratique obtenu en fixant les paramètres $\Delta P = 0.1$ et $\Delta e_x = 0.01$. Ici, N_{dis} est le nombre de points de discrétisation; N_{tot} , le nombre de variables dans le problème; t_{exec} , le temps d’exécution, exprimé en secondes et la colonne "Rapport" donne la valeur de la fonction $\frac{LBD}{UBD}$ obtenue dans chaque cas; le temps d’exécution représente le temps de calcul pour le problème d’optimisation plus le temps nécessaire à construire le problème et à calculer les limites d’intervalle pour les variables considérées. Les temps de calcul sont raisonnables, même pour les problèmes de grande taille (3000 points de discrétisation).

On peut remarquer que, même pour les problème de grandes taille, les temps de résolution de la relaxation convexe sont assez petits, le problème n’étant pas difficile, en raison d’un nombre de variables peu important. Bien sûr, le fait de travailler avec une relaxation quadratique est crucial, les algorithmes de résolution des problèmes quadratiques étant plus performants que ceux existant pour trouver la solution d’un problème non-linéaires; par exemple, le problème de contrôle optimal discrétisé prend plus de temps à être résolu que sa relaxation quadratique convexe.

Considérons maintenant l’influence des limites d’intervalle sur la qualité de l’estimation inférieure. Comme déjà précisé ΔP , Δe_x et ΔL dénotent les tailles des intervalles utilisées pour borner les variables d’état P, e_x et L . La Figure (4.11) présente la relation $\frac{LBD}{UBD}100 = fonction(\Delta P, \Delta e_x)$, le paramètre ΔL étant une constante dans ce cas (et égale à 1). Dans ces

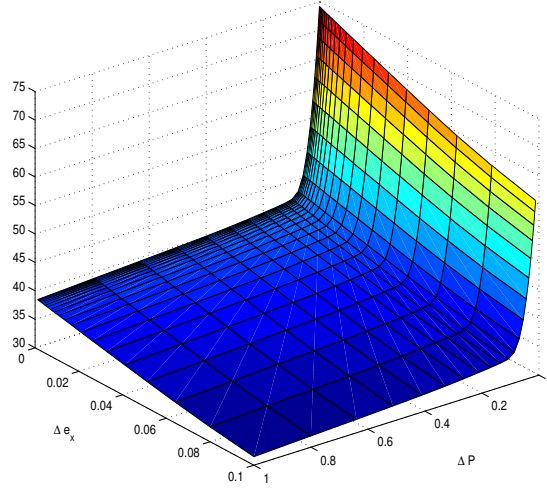


FIG. 4.11 – Le rapport $\frac{LBD}{UBD}$ fonction de $\Delta P \in [0.01, 1]$ et $\Delta e_x \in [0.1, 0.001]$. P est exprimé en Mm (Mégamètres). $\Delta L = 1$, constant dans ce cas. La meilleure sous-estimation dans ce cas ne dépasse pas 75%.

conditions, la meilleure limite obtenue ne dépasse pas les 75% de la limite supérieure, solution locale du problème spatial. Ce qui est encore plus important est le fait que cette limite est difficilement utilisable dans le cadre d'un algorithme de Branch and Bound. La figure (4.12) montre le rapport $\frac{LBD}{UBD}$ en fonction de ΔP (à Δe_x fixé) et ce même rapport en fonction de Δe_x (à ΔP fixé). Comme on peut le constater en analysant ces figures, on obtient une limite inférieure différente de 0, mais exploitable dans un algorithme seulement quand on est très près de la solution locale. Ceci est évidemment dû au fait que la relaxation qu'on utilise est trop forte. En effet, utiliser simplement les limites des intervalles est trop peu contraignant, la principale difficulté étant la mauvaise approximation faite pour obtenir les limites sur la variable W .

En conclusion de ce modèle, on obtient une limite inférieure non nulle, ce qui prouve que la démarche qu'on suit est la bonne. D'autre part, la convexification des fonctions $\sin L$ et $\cos L$ est la difficulté principale, dont on doit s'affranchir afin d'obtenir de meilleurs résultats.

Deuxième modèle

Ce modèle est en fait une autre façon de convexifier le problème en se concentrant toujours sur les deux équations en P et L . Quoiqu'on puisse affirmer *a posteriori* que les résultats obtenus avec ce modèle ne sont pas meilleurs que ceux du modèle précédent, son intérêt réside dans la façon dont il est obtenu. L'idée est de réduire l'indétermination introduite par

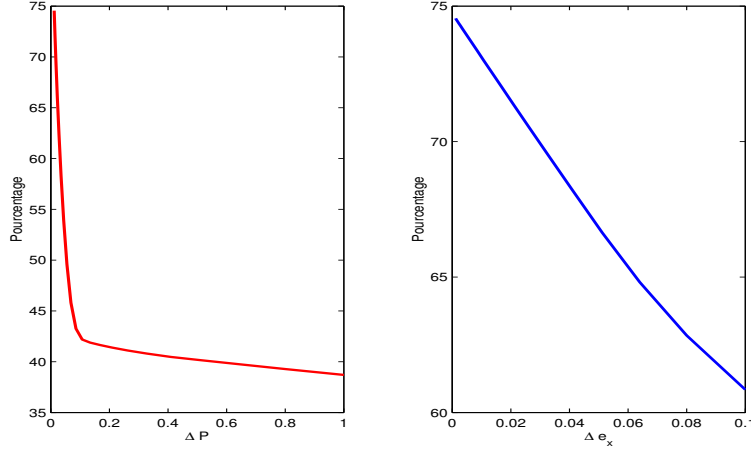


FIG. 4.12 – Pourcentage $\frac{LBD}{UBD}$ en fonction des paramètres ΔP (à gauche pour un $\Delta e_x = 0.001$ fixé) et Δe_x (à droite pour $\Delta P = 0.01$, fixé). L'influence de ΔP est plus petite pour de grands intervalles et devient importante dans le voisinage de la solution locale. Le paramètre Δe_x est plus important, sa variation ayant un impact plus important sur la qualité de l'estimation inférieure.

l'équation

$$\frac{dP}{dt} = \frac{2}{m\sqrt{\mu}} \frac{P^{3/2}}{W} u_2 \quad (4.38)$$

en utilisant une nouvelle variable d'état $R = P^{-1/2}$. On peut alors récrire cette équation :

$$\frac{dR}{dt} = -\frac{1}{m\sqrt{\mu}} \frac{1}{W} u_2 \quad (4.39)$$

L'intérêt de cette approche est de faire disparaître le terme en P dans la première équation. En fait, convexifier un terme bilinéaire (comme celui donné en (4.39)) est plus efficace en ce qui concerne les qualité de la sous-estimation (ou surestimation) qu'un terme trinéaire (celui donné en (4.38)), en raison d'un plus petit nombre de relaxations. Avec ces considérations, le nouveau système à convexifier s'écrit:

$$\begin{cases} \frac{dR}{dt} &= -\frac{1}{m\sqrt{\mu}} \frac{1}{W} u_2 \\ \frac{dL}{dt} &= \sqrt{\mu} W^2 R^3 \end{cases} \quad (4.40)$$

Remarquons qu'on est pas assuré d'avoir une meilleure relaxation avec ce modèle car, malgré l'amélioration trouvée pour la première équation, on a modifié la deuxième. En effet, la nouvelle fonction à convexifier est R^3 qui est moins bien approché par linéarisation que la fonction $P^{-3/2}$, en raison de sa plus grande courbure. Outre cette modification par rapport au modèle

précédent, on utilise aussi la séparation du contrôle en partie positive et négative. Dans un premier temps le problème suivant est obtenu:

$$\begin{aligned}
& \min \int_0^{t^f} (u_2^+)^2 + (u_2^-)^2 dt \\
(M) \quad & \left\{ \begin{array}{l} F^R - \frac{1}{m\sqrt{\mu}}T_1 + \frac{1}{m\sqrt{\mu}}T_2 = 0 \\ F^L - \sqrt{\mu}T_3 = 0 \end{array} \right. \\
(L) \quad & \left\{ \begin{array}{l} M_1 = R^3 \\ M_1 W^2 = T_3 \end{array} \right. \quad T_1 W = u_2^+ \quad T_2 W = u_2^- \\
(C) \quad & \left\{ \begin{array}{l} M_1 W^2 = T_3 \\ T_3 = M_1 W^2 \end{array} \right. \\
(T) \quad & \left\{ \begin{array}{l} u_2^+ \leq T_{max} \\ u_2^- \leq T_{max} \end{array} \right.
\end{aligned} \tag{4.41}$$

où:

- (L) dénote l'ensemble d'équations de Gauss considérées dans ce modèle
- (T) est la contrainte limitant la poussée
- M_1, T_1, T_2, T_3 sont des variables supplémentaires introduites pour la convexification et dont les relations de définition sont données par l'ensemble d'équations (C). Comme ces équations sont les seules non-linéaires du système, la convexification se concentre sur ces termes; on utilise les fonctions $\text{lininf}(R^3)$ et $\text{linsup}(R^3)$ pour l'équation $M_1 = R^3$; on utilise aussi les limites W^{min} et W^{max} pour relaxer les autres équations. Par exemple, dans la relaxation convexe, l'équation $T_1 W = u_2^+$ devient:

$$\left\{ \begin{array}{l} T_1 W^{min} \leq u_2^+ \\ T_1 W^{max} \geq u_2^+ \end{array} \right. \tag{4.42}$$

Avec ces remarques, le problème qu'on résout s'obtient à partir de (4.41) en discrétisant par collocation directe.

Résultats. On considère le même problème spatial afin de tester cette nouvelle méthode de convexification. La relaxation ainsi obtenue est toujours un problème quadratique (CPLEX a été utilisé comme solveur), quant au problème spatial discrétisé, sa solution est obtenue de façon identique, à l'aide du solveur KNITRO.

Les temps de calcul pour la limite inférieure dans ce cas sont supérieures à ceux obtenus précédemment, à cause du nombre de variables plus élevé (voir tableau 4.4):

Sur la figure (4.13) on a représenté l'évolution du rapport $\frac{LBD}{UBD}$ en fonction des paramètres ΔP et Δe_x .

On n'obtient pas de meilleurs résultats dans ce cas. Au contraire, une légère dégradation de la limite inférieure peut être remarquée. En se rapprochant de la solution locale du problème, notre sous-estimation avoisine les 72% de l'objectif optimal, solution locale du problème spatial (à comparer avec les 75% du premier modèle). D'autre part, la variation de la surface

N_{dis}	N_{tot}	$t_{exec}(s)$	Rapport (%)
200	3592	0.4489	40.6564
400	7192	1.0324	40.6148
800	14392	2.6121	40.5946
1000	17992	3.6361	40.59082
1500	26992	6.1373	40.5858
3000	53992	29.3399	40.5810

TAB. 4.4 – Les temps d'exécution pour la relaxation convexe, problème quadratique obtenu pour les paramètres $\Delta P = 0.1$ et $\Delta e_x = 0.01$. Les notations: N_{dis} , le nombre des points de discrétisation; N_{tot} , le nombre de variables dans le problème; t_{exec} , le temps d'exécution, exprimé en secondes et la colonne "Rapport" donne la valeur de la fonction $\frac{LBD}{UBD}$ obtenue dans chaque cas; les temps d'exécution sont supérieurs à ceux obtenus avec le modèle précédent, car le problème a plus de variables et de contraintes.

$\frac{LBD}{UBD}$ est brusque suivant la direction ΔP . L'avantage qu'on peut tirer de cet comportement est lié à la technique de Branch and Bound : il est plus avantageux de réduire la dimension de l'espace de recherche en divisant les intervalles $[P^L, P^U]$ que ceux de type $[e_x^L, e_x^U]$. En effet on a plus de chance d'obtenir une bonne limite inférieure en "séparant" les variables P que e_x . Toutefois, il existe un désavantage majeur : la limite inférieure n'est pas très efficace quand l'espace de recherche est très grand autour de la solution locale, elle ne dépasse pas les 50% de la limite supérieure. Ces remarques sont également vraies pour le modèle précédent étant donnée la similitude des résultats obtenus. Afin de mieux comparer les deux approches, on représente sur la figure (4.14) l'évolution de la limite inférieure dans les deux cas, en considérant dans un premier temps $\Delta P = 0.01$ fixé et puis $\Delta e_x = 0.001$ fixé.

Conclusions

On a présenté deux façons différentes de convexifier le problème de contrôle optimal en utilisant les équations de Gauss en fonction du temps. Une amélioration importante par rapport au modèle cartésien a été apportée, notamment aux techniques qu'on a utilisées pour exprimer cette propriété fondamentale du problème relaxé: si le satellite n'est pas contrôlé, il reste sur la même orbite. C'est là, l'apport principal de cette approche par rapport au cas de la convexification cartésienne. Dans nos efforts pour améliorer la limite inférieure on a développé plusieurs modèles différents, en combinant les différentes techniques présentées:

- la séparation du contrôle en partie positive et négative ou l'utilisation de la valeur absolue du contrôle
- l'utilisation ou non de la substitution $R = P^{-1/2}$.

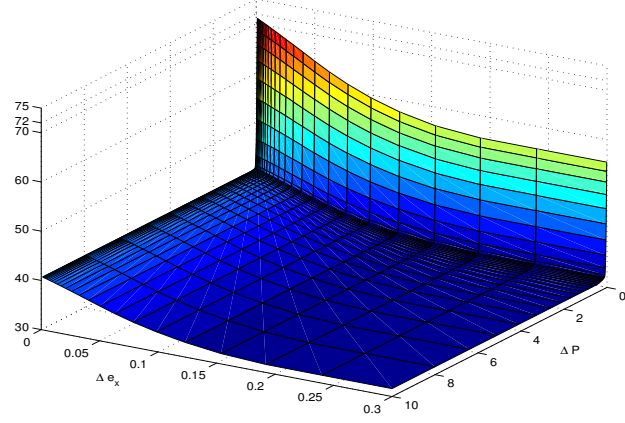


FIG. 4.13 – Le rapport $\frac{LBD}{UBD}$ fonction de $\Delta P \in [0.01, 10]$ et $\Delta e_x \in [0.3, 0.001]$. P est exprimé en Mm (Mégamètres). $\Delta L = 1$, constant dans ce cas. On observe une légère dégradation de LBD par rapport au modèle précédent. La meilleure estimation ne dépasse pas les 72% de la limite supérieure.

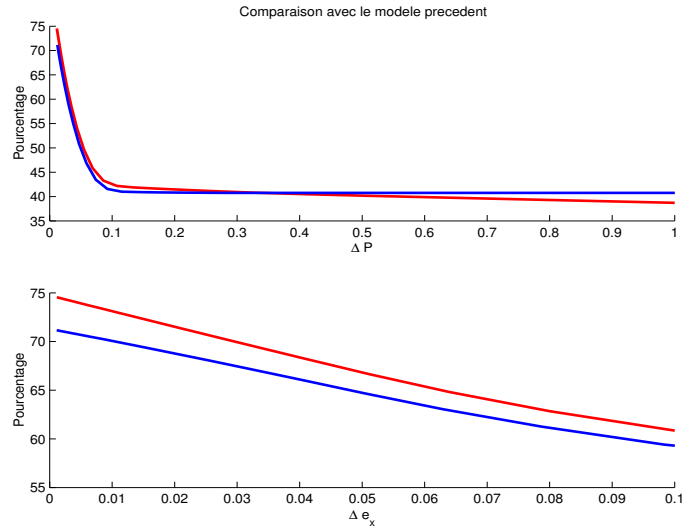


FIG. 4.14 – En haut, le rapport $\frac{LBD}{UBD}$ fonction de $\Delta P \in [0.01, 1]$ pour $\Delta e_x = 0.001$ fixé. En rouge le résultat pour le modèle précédent et en bleu, le résultat associé à la démarche décrite dans ce paragraphe. En bas, le rapport $\frac{LBD}{UBD}$ fonction de $\Delta e_x \in [0.001, 0.1]$ pour $\Delta P = 0.01$ fixé. Toujours en rouge le résultat pour le modèle précédent et en bleu, le résultat pour la démarche basée sur la substitution $R = P^{1/2}$. On observe un avantage dans la qualité de l'estimation pour le premier modèle, à cause d'une meilleure estimation de l'équation de Gauss $\frac{dL}{dt}$.

N'ayant pas obtenu d'amélioration notable par rapport aux résultats commentés ici, on ne détaille pas les autres modèles.

On a aussi essayé d'intégrer dans le système les équations $\frac{de_x}{dt}$ et $\frac{de_y}{dt}$, mais encore une fois, sans obtenir un quelconque gain concernant la qualité de la limite inférieure. Le problème principal reste les fonctions $\sin L$ et $\cos L$ qui, étant donnée leurs fortes variations sur des intervalles petits sont très difficiles à convexifier de façon efficace. Les sous-estimations et sur-estimations des équations $\frac{e_x}{dt}$ et $\frac{e_y}{dt}$ sont mauvaises et leur intégration dans le système n'apporte pas d'amélioration. Dans les paragraphes suivants on présente différentes modélisations nous permettant de nous affranchir de cet obstacle.

4.3.3 Equations de Gauss en L

Pour mieux convexifier les fonctions $\sin L$ et $\cos L$, l'idée est d'exprimer les équations de Gauss en fonction de L , la longitude cumulée, au lieu du temps. En fait, en discrétisant ces nouvelles équations, les instants L_i sont constants et connus à l'avance, ce qui transforme les fonctions $\sin L$ et $\cos L$ en constantes du problème.

La nouvelle dynamique du système est donnée par:

$$\begin{cases} \frac{dP}{dL} &= \frac{2}{m\mu} \frac{P^3}{W^3} u_2 \\ \frac{de_x}{dL} &= \frac{\sin L}{m\mu} \frac{P^2}{W^2} u_1 + \frac{1}{m\mu} \frac{P^2}{W^3} Q_1 u_2 \\ \frac{de_y}{dL} &= -\frac{\cos L}{m\mu} \frac{P^2}{W^2} u_1 + \frac{1}{m\mu} \frac{P^2}{W^3} Q_2 u_2 \end{cases} \quad (4.43)$$

où:

$$\begin{cases} W &= 1 &+ e_x \cos L &+ e_y \sin L \\ Q_1 &= 2 \cos L &+ e_x (1 + \cos^2 L) &+ e_y (\sin L \cos L) \\ Q_2 &= 2 \sin L &+ e_x (\sin L \cos L) &+ e_y (1 + \sin^2 L) \end{cases} \quad (4.44)$$

Un autre avantage évident de cette approche est le fait d'avoir transformé les expressions des paramètres W , Q_1 et Q_2 en relations affines par rapport à e_x et e_y . Cela évite de convexifier ces équations et simplifie le problème à résoudre.

Il y a toutefois un désavantage à cette technique: l'objectif du problème, jusque-là quadratique, est devenu une fonction non-convexe:

$$\int_{t_0}^{t_f} u^2(t) dt = \int_{L^0}^{L^f} u^2(L) \frac{dt}{dL} dL = \frac{1}{\sqrt{\mu}} \int_{L^0}^{L^f} u^2(L) \frac{P^{3/2}}{W^2} dL \quad (4.45)$$

Du point de vue de la convexification, cela signifie que l'objectif doit être convexifié à son tour. Au lieu de suivre cette voie, on propose la substitution suivante (ce qui rend à nouveau l'objectif quadratique):

$$\bar{\mathbf{u}} = \mathbf{u} \frac{P^{3/4}}{W} \quad (4.46)$$

Avec cette nouvelle variable ($\bar{\mathbf{u}}$), l'objectif devient: $\frac{1}{\sqrt{\mu}} \int_{L^0}^{L^f} \bar{u}^2(L) dL$. Néanmoins, la non-convexité se retrouve dans l'expression de la contrainte sur la poussée maximale: $u^2(t) \leq T_{max}^2$, inéquation convexe auparavant qui devient $\bar{u}^2 \frac{W^2}{P^{3/2}} \leq T_{max}^2$, désormais non-convexe.

Pour ne pas alourdir les notations, on va désigner toujours par \mathbf{u} le nouveau contrôle (il n'y a pas risque de confusion, étant donné que l'ancien contrôle ne sera pas utilisé dans les paragraphes qui suivent). On peut maintenant écrire la nouvelle forme des équations de Gauss en L et exprimer le problème d'optimisation qu'on désire relaxer :

$$\left\{ \begin{array}{l} \min \frac{1}{\sqrt{\mu}} \int_{L^0}^{L^f} (u_1^2 + u_2^2) dL \\ \frac{dP}{dL} = \frac{2}{m\mu} \frac{P^{9/4}}{W^2} u_2 \\ \frac{de_x}{dL} = \frac{\sin L}{m\mu} \frac{P^{5/4}}{W} u_1 + \frac{1}{m\mu} \frac{P^{5/4}}{W^2} Q_1 u_2 \\ \frac{de_y}{dL} = -\frac{\cos L}{m\mu} \frac{P^{5/4}}{W} u_1 + \frac{1}{m\mu} \frac{P^{5/4}}{W^2} Q_2 u_2 \\ (u_1^2 + u_2^2) \frac{W^2}{P^{3/2}} \leq T_{max}^2 \end{array} \right. \quad (4.47)$$

On conclut cette partie préalable à la convexification par deux remarques concernant la limite supérieure, solution locale du problème (4.47). Les expériences numériques avec les méthodes directes [15], [14] nous montrent que les équations de Gauss en L (formules 4.43) sont un bien meilleur choix par rapport au modèle cartésien ou aux équations de Gauss en temps; elles posent moins de problèmes numériques et on obtient de meilleurs temps de calcul. La deuxième observation est qu'aucun effort supplémentaire de développement n'est nécessaire à la suite du changement de variable (4.46) et au nouveau modèle obtenu (4.47). En effet, la solution locale du problème (4.43) est toujours une limite supérieure pour la solution globale du problème spatial 2D à masse constante. Les tests numériques qu'on a effectués sur quelques problèmes simples de contrôle orbital n'ont pas mis en évidence un avantage numérique à résoudre un modèle plutôt que l'autre.

Un modèle simplifié

On considère ici le même modèle simplifié que celui présenté au paragraphe §4.3.2, la motivation étant double: la simplicité du modèle (sa petite taille) favorise sa résolution dans un temps de calcul pratiquement négligeable et d'autre part, la sous-estimation de la contribution de u_2 dans l'objectif implique moins de convexification. L'équation prise en compte est:

$$\frac{dP}{dL} = \frac{2}{m\mu} \frac{P^{9/4}}{W^2} u_2 \quad (4.48)$$

On remplace la variable de contrôle par ses parties positives et négatives:
 $u_2 = u_2^+ - u_2^-$:

$$\frac{dP}{dL} = \frac{2}{m\mu} \frac{P^{9/4}}{W^2} u_2^+ - \frac{2}{m\mu} \frac{P^{9/4}}{W^2} u_2^- \quad (4.49)$$

et on suit la démarche décrite en §4.3.2:

- les bornes d'intervalle de e_x et e_y sont utilisées pour obtenir de bonnes estimations pour W (W^L et W^U); cette fois-ci, ces limites sont meilleures que dans les cas précédents.
- les termes de la forme $\frac{P^{9/4}}{W^2} u_2^+$ et $\frac{P^{9/4}}{W^2} u_2^-$ sont relaxés en utilisant les convexifications de McCormick; par exemple, pour le terme en u_2^+ les équations sont relaxées de la façon suivante:

$$\begin{aligned} T_1 = \frac{P^{9/4}}{W^2} &\Rightarrow \begin{cases} T_1 &\leq \frac{1}{(W^L)^2} \text{lin sup}(P^{9/4}) \\ T_1 &\geq \frac{1}{(W^U)^2} \text{lin inf}(P^{9/4}) \end{cases} \\ T_2 = T_1 u_2^+ &\Rightarrow \begin{cases} T_2 &\geq T_1^U u_2^+ + (u_2^+)^U T_1 - T_1^U (u_2^+)^U \\ T_2 &\geq T_1^L u_2^+ + (u_2^+)^L T_1 - T_1^L (u_2^+)^L \\ T_2 &\leq T_1^U u_2^+ + (u_2^+)^L T_1 - T_1^U (u_2^+)^L \\ T_2 &\leq T_1^L u_2^+ + (u_2^+)^U T_1 - T_1^L (u_2^+)^U \end{cases} \end{aligned} \quad (4.50)$$

où T_1 et T_2 sont les nouvelles variables introduites pour les besoins de la convexification.

Cette méthode a été testée sur le même problème spatial que les autres convexifications et les résultats sont similaires à ceux présentés auparavant. On donne ici les résultats sur un modèle légèrement amélioré, obtenu à partir de (4.48) en utilisant la substitution: $R = P^{-5/4}$. La nouvelle équation devient:

$$\frac{dR}{dL} = -\frac{5}{2m\mu} \frac{1}{W^2} u_2 \quad (4.51)$$

où le second membre de l'équation ne contient plus de termes en R , d'où l'intérêt de la substitution. La convexification de cette équation beaucoup plus simple que celle donnée en (4.48) s'écrit alors:

$$\begin{aligned} \frac{dR}{dt} &= -\frac{5}{2m\mu} T_1 + \frac{5}{2m\mu} T_2 \\ T_1 (W^L)^2 &\leq u_2^+ \\ T_1 (W^U)^2 &\leq u_2^+ \\ T_2 (W^L)^2 &\leq u_2^- \\ T_2 (W^U)^2 &\leq u_2^- \end{aligned} \quad (4.52)$$

où T_1 et T_2 sont des nouvelles variables du problème relaxé.

De la même façon, l'équation $((u_2^+)^2 + (u_2^-)^2) \frac{W^2}{P^{3/2}} \leq T_{max}^2$, exprimant la poussée maximale qui s'exerce à l'instant t sur le satellite, se transforme en:

$$\begin{cases} \frac{T_{max}}{W^L} P^{3/4} - u^+ &\geq 0 \\ \frac{T_{max}}{W^L} P^{3/4} - u^- &\geq 0 \end{cases} \quad (4.53)$$

et se convexifie en utilisant l'expression de $R = P^{-4/3}$ et les formules de McCormick pour les fonctions bilinéaires.

Résultats. Avec le même problème de test qu'en §4.3.2, cette convexification a été résolue en utilisant CPLEX. Le tableau 4.5 présente les temps des calculs en fonction du nombre de points de discrétisation. Malgré la simplicité du problème de départ (une seule équation a été retenue parmi les trois du système de Gauss), on voit que la relaxation convexe contient un nombre plus important de variables; les temps de calcul restent toutefois raisonnables, meilleurs que pour les convexifications des équations de Gauss en fonction du temps.

N_{dis}	N_{tot}	$t_{exec}(s)$	Rapport (%)
200	1996	0.138947	44.67957
400	3996	0.362702	44.87441
800	7996	0.974329	44.92768
1000	9996	1.323389	44.93424
1500	14996	2.353284	44.94165
3000	29996	6.521377	44.94693
10000	99996	59.308018	44.94941

TAB. 4.5 – Les temps d'exécution pour la relaxation convexe, problème quadratique obtenu pour les paramètres $\Delta P = 0.1$ et $\Delta e_x = 0.01$. La légende est: N_{dis} , le nombre des points de discrétisation; N_{tot} , le nombre de variables dans le problème; t_{exec} , le temps d'exécution, exprimé en secondes et la colonne "Rapport" donne la valeur de la fonction $\frac{LBD}{UBD}$ obtenue dans chaque cas. Les temps de calcul restent raisonnables et la valeur du rapport varie très peu d'une discrétisation à l'autre, ce qui est en notre avantage, le Branch and Bound ayant peu de chances d'aboutir pour un très grand nombre de variables.

Considérons maintenant l'évolution de la limite inférieure en fonction des paramètres ΔP et Δe_x (voir Figure 4.15).

La meilleure limite inférieure obtenue en utilisant ce modèle est de 75% de la limite supérieure donnée par une méthode locale. On espérait mieux pour cette convexification, mais il faut garder à l'esprit que, même si les limites W^L et W^U sont mieux calculées, on utilise une seule équation ($\frac{dR}{dL}$) par rapport à deux équations dans le cas des équations de Gauss en fonction du temps ($\frac{dR}{dt}$ et $\frac{dL}{dt}$). Une autre observation intéressante porte sur la façon dont la fonction $\frac{LBD}{UBD}$ évolue: pour de grands intervalles ΔP , l'influence des limites P^L et P^U sur la valeur de cette fonction est mineure. Ceci est dû au fait que le second membre de l'équation (4.52) ne contient pas de terme en R . Néanmoins, ces limites (P^L et P^U) deviennent brusquement importantes dans le voisinage de la solution locale, là où les contraintes de boîtes faisant inter-

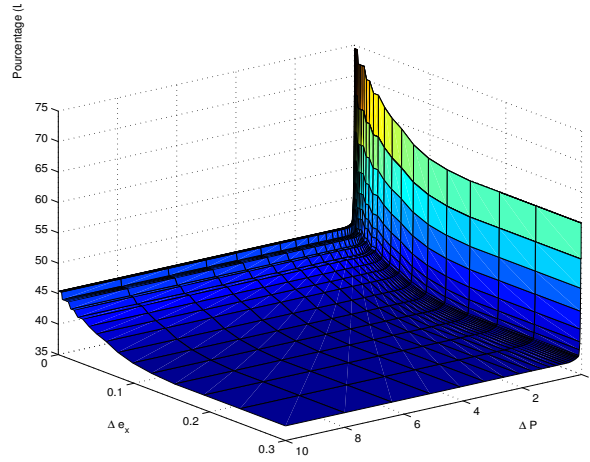


FIG. 4.15 – Pourcentage $\frac{LBD}{UBD}$ en fonction des paramètres ΔP et Δe_x sur le pavé $[0.01, 10.00] \times [0.30, 0.001]$. La meilleure limite inférieure atteint 75% de la limite supérieure. Les résultats sont meilleurs que ceux obtenus pour les équations de Gauss en fonction du temps, mais le gain n'est pas très important: à peine 2%.

venir ces paramètres sont souvent saturées. Ces considérations sont étayées par le graphe (4.16) montrant les évolutions du rapport $\frac{LBD}{UBD}$ en fonction de ΔP (respectivement Δe_x) à Δe_x (respectivement ΔP) fixé.

Un modèle complet

On présente maintenant un modèle prenant en compte toutes les équations de Gauss en fonction de la longitude (L). La principale difficulté rencontrée dans la convexification d'un tel système est la faible valeur de $\frac{de_x}{dt}$ et $\frac{de_y}{dt}$, ce qui fait que la convexification introduit trop d'espace de recherche, la limite inférieure ne dépassant pas les meilleurs résultats obtenus avec le modèle simplifié, décrit en §4.3.3. Afin de renforcer la relaxation convexe, on cherche une substitution de variable, nous permettant d'ajouter des relations (si possible linéaires) entre les différentes variables du modèle.

L'idée de ce modèle est de partir de la forme implicite de la dynamique [18]:

$$\frac{d(\frac{1}{P})}{dL} + \frac{d(\frac{e_x}{P})}{dL} + \frac{d(\frac{e_y}{P})}{dL} = 0, \quad (4.54)$$

équation qui peut être obtenue par calcul direct à partir du système donné en (4.47).

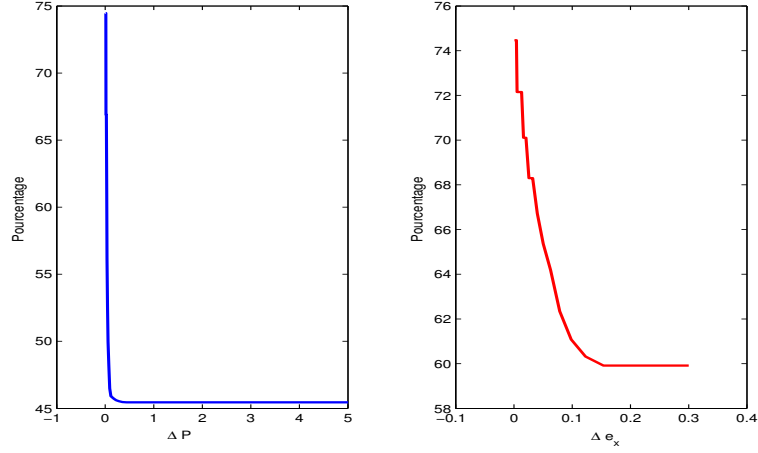


FIG. 4.16 – Pourcentage $\frac{LBD}{UBD}$ en fonction des paramètres ΔP (à gauche, en bleu pour un $\Delta e_x = 0.001$ fixé) et Δe_x (à droite, en rouge pour $\Delta P = 0.01$, fixé). L'influence de ΔP est petite pour des grandes intervalles et devient brusquement importante dans le voisinage de la solution locale. Le paramètre Δe_x est toujours important dans l'évolution de la fonction représentée car les contraintes faisant intervenir ce paramètre jouent un rôle actif dans la relaxation convexe.

Ceci nous amène à la substitution suivante:

$$\begin{cases} R &= \frac{1}{P} \\ E_x &= \frac{e_x}{P} \\ E_y &= \frac{e_y}{P} \end{cases} \quad (4.55)$$

Le système d'équations exprimé en utilisant les variables R , E_x , E_y s'écrit:

$$\begin{cases} \frac{dR}{dL} &= -\frac{2}{m\mu} \frac{R^{7/4}}{w^2} u_2 \\ \frac{dE_x}{dL} &= \frac{\sin L}{m\mu} \frac{R^{3/4}}{w} u_1 + \frac{1}{m\mu} \frac{R^{3/4}}{w^2} q_1 u_1 \\ \frac{dE_y}{dL} &= -\frac{\cos L}{m\mu} \frac{R^{3/4}}{w} u_1 + \frac{1}{m\mu} \frac{R^{3/4}}{w^2} q_2 u_1 \end{cases} \quad (4.56)$$

où:

$$\begin{cases} w &= R & + & E_x \cos L & + & E_y \sin L \\ q_1 &= 2R \cos L & - & E_x \sin^2 L & + & E_y \sin L \cos L \\ q_2 &= 2R \sin L & + & E_x \sin L \cos L & - & E_y \cos^2 L \end{cases} \quad (4.57)$$

Avant de commencer à convexifier le système (4.56) remarquons que les paramètres w , q_1 et q_2 deviennent maintenant des variables du problème à résoudre. Ceci n'est pas strictement nécessaire pour exprimer les équations de Gauss (il s'agit juste des notations pour simplifier la lecture des équations), mais dans la convexification le passage des paramètres aux variables est une étape obligatoire. Ceci a comme conséquence l'augmentation du

nombre de variables dans le problème d'optimisation, mais les expressions données en (4.57) sont linéaires par rapport aux autres variables du problème, R , E_x et E_y .

Afin d'obtenir la relaxation convexe du problème (4.56) plusieurs techniques déjà utilisées dans les paragraphes antérieurs vont être considérées:

- la séparation du contrôle en partie positive et négative assure le fait que sans contrôler le système le satellite reste sur la même orbite:

$$\begin{cases} u_1 = u_1^+ - u_1^- \\ u_2 = u_2^+ - u_2^- \end{cases}$$

Les nouvelles variables de contrôle sont u_1^+ , u_1^- , u_2^+ et u_2^- .

- les termes multilinéaires sont transformés en termes bilinéaires en introduisant de nouvelles variables. Par exemple:

$$v = v_1 v_2 v_3 \quad \text{se transforme en} \quad \begin{cases} v = v_1 v_4 \\ v_4 = v_2 v_3 \end{cases}$$

où v_4 est la nouvelle variable introduite pour les besoins de la convexification. Ses limites (v_4^L et v_4^U) s'obtiennent en utilisant l'analyse de l'intervalle sur la relation de définition: $v_4 = v_2 v_3$.

- les termes bilinéaires sont convexifiés en utilisant les relations de McCormick (§4.6).
- les termes de la forme x^n , ou $n \in \mathbb{R}$ sont convexifiés en utilisant les fonctions $\text{lin sup}(x^\alpha)$ et $\text{lin inf}(x^\alpha)$. Ceci nécessite encore une fois l'introduction d'une nouvelle variable, définie par $y = x^\alpha$.

En utilisant les équations (4.54), (4.56) et les remarques précédentes, le problème intermédiaire suivant est obtenu :

$$\begin{aligned} & \min \int_{L^0}^{L^f} (u_1^+)^2 + (u_1^-)^2 + (u_2^+)^2 + (u_2^-)^2 dL \\ (C) \quad & \begin{cases} F^R = -\frac{2}{m\mu}T_1 + \frac{2}{m\mu}T_2 \\ F^x = \frac{\sin L}{m\mu}T_3 - \frac{\sin L}{m\mu}T_4 + \frac{1}{m\mu}T_5 - \frac{1}{m\mu}T_6 \\ F^y = -\frac{\cos L}{m\mu}T_3 + \frac{\cos L}{m\mu}T_4 + \frac{1}{m\mu}T_7 - \frac{1}{m\mu}T_8 \end{cases} \\ (F) \quad & \begin{cases} F^R + F^x \cos L + F^y \sin L = 0 \end{cases} \\ (P) \quad & \begin{cases} w = R + E_x \cos L + E_y \sin L \\ q_1 = 2R \cos L - E_x \sin^2 L + E_y \sin L \cos L \\ q_2 = 2R \sin L + E_x \sin L \cos L - E_y \cos^2 L \end{cases} \\ (M) \quad & \begin{cases} S_1 = R^{7/4} & S_2 = R^{3/4} & S_3 = w^2 \\ M_1 S_3 = S_1 & M_2 w = S_2 & M_3 S_3 = S_2 \\ M_1 u_2^+ = T_1 & M_1 u_2^- = T_2 & M_2 u_1^+ = T_3 \\ M_2 u_1^- = T_4 & M_3 q_1 = M_4 & M_3 q_2 = M_5 \\ M_4 u_2^+ = T_5 & M_4 u_2^- = T_6 & M_5 u_2^+ = T_7 \\ M_5 u_2^- = T_8 & S_1 R = S_2 & M_3 w = M_2 \end{cases} \end{aligned} \tag{4.58}$$

dans lequel:

- (C) dénote l'ensemble d'équations issues directement du système de Gauss, $T_1, T_3, T_3, T_4, T_5, T_6, T_7, T_8$ étant les nouvelles variables correspondant aux termes:

$$\begin{aligned} T_1 &= \frac{R^{7/4}}{w^2} u_2^+ & T_2 &= \frac{R^{7/4}}{w^2} u_2^- & T_3 &= \frac{R^{3/4}}{w} u_1^+ & T_4 &= \frac{R^{3/4}}{w} u_1^- \\ T_5 &= \frac{R^{3/4}}{w^2} q_1 u_2^+ & T_6 &= \frac{R^{3/4}}{w^2} q_1 u_2^- & T_7 &= \frac{R^{3/4}}{w^2} q_2 u_2^+ & T_8 &= \frac{R^{3/4}}{w^2} q_2 u_2^- \end{aligned} \quad (4.59)$$

- (F) désigne l'ensemble d'équations linéaires exprimant la forme implicite;
- (P) dénote les équations définissant les paramètres (variables du problème) w, q_1, q_2
- (M) dénote l'ensemble d'équations qui restent à être convexifiées en utilisant les enveloppes de McCormick ou les fonctions *linsup* et *lininf*, suivant le cas. Les nouvelles variables introduites dans ce cas (S_1, S_2, S_3 et $M_1, M_3, M_3, M_4, M_5, M_6, M_7, M_8$) sont définies par les relations suivantes:

$$\begin{aligned} S_1 &= R^{7/4} & S_2 &= R^{3/4} & S_3 &= w^2 \\ M_1 &= \frac{R^{7/4}}{w^2} & M_2 &= \frac{R^{3/4}}{w} & M_3 &= \frac{R^{3/4}}{w^2} \\ M_4 &= \frac{R^{3/4}}{w^2} q_1 & M_5 &= \frac{R^{3/4}}{w^2} q_2 \end{aligned} \quad (4.60)$$

En utilisant les enveloppes convexes pour les termes bilinéaires on obtient la relaxation convexe directement à partir du système par discrétisation.

Résultats. Le même jeu de tests que pour les autres convexifications présentées antérieurement à été utilisé afin de comparer l'efficacité de la méthode avec les autres modèles.

On commence l'analyse des résultats en donnant les temps de calcul et le nombre de variables dans la relaxation du problème spatial. En effet, vu la forme des équations (4.58), cette convexification augmente considérablement la taille du problème (voir tableau 4.6). L'impact sur les temps d'exécution du problème est évident. Dès le début, pour des problèmes de petite taille (200 points de discrétisation) le temps de calcul est non négligeable (autour de 30s). C'est le prix à payer pour une bonne estimation inférieure. Il faut aussi préciser que les temps de calcul obtenus pour la convexification contiennent les temps effectifs d'optimisation plus le temps pour la construction de la relaxation convexe et le calcul des limites d'intervalle pour les variables nécessaires à la convexification. Ce dernier temps s'avère être extrêmement pénalisant; par exemple, pour le problème obtenu avec 10000 points de discrétisation, plus de 80% du temps de calcul est dédié à la construction du problème relaxé et à l'analyse par intervalle.

La figure (4.17) montre l'évolution du rapport $\frac{LBD}{UBD}$ en fonction de la taille des intervalles (ΔP et Δe_x). Pour les intervalles de variations considérés,

N_{dis}	N_{tot}	$t_{exec}(s)$	Rapport (%)
200	10775	23.6914	89.21262
400	21575	50.2818	89.51814
800	43175	122.9172	89.71111
1000	53975	166.7295	89.75155
1500	80975	299.1243	89.80697
3000	161975	888.0553	89.86389
10000	539975	$\approx 22h$	89.87212

TAB. 4.6 – Les temps d’exécution pour la relaxation convexe, problème quadratique obtenu pour les paramètres $\Delta P = 0.1$ et $\Delta e_x = 0.01$. La légende est: N_{dis} , le nombre des points de discrétisation; N_{tot} , le nombre de variables dans le problème; t_{exec} , le temps d’exécution, exprimé en secondes et la colonne "Rapport" donne la valeur de la fonction $\frac{LBD}{UBD}$ obtenue dans chaque cas; les temps d’exécution représente le temps de calcul pour le problème d’optimisation plus le temps nécessaire à construire le problème et à calculer les limites d’intervalle pour les variables considérées. Les temps de calcul ne sont plus négligeables, la bonne sous-estimation obtenue par cette méthode se faisant au détriment du temps de calcul. Même si les temps d’exécution ne sont pas mauvais, on peut s’attendre à un impact important sur le temps de calcul d’un algorithme qui résout ce problème à chaque itération.

la meilleure estimation inférieure est de 95% ce qui représente une nette amélioration par rapport aux modèles prenant en compte seulement l’équation en P dans le système de Gauss. A titre d’exemple, avec le modèle décrit en §4.3.3 on obtient pour $\Delta P = 0.1$ et $\Delta e_x = 0.01$ une estimation inférieure à 44% tandis qu’avec ce modèle l’estimation inférieure, obtenue pour la même valeur des paramètres, est de 89%, plus que le double. Le graphe (4.18) fait une comparaison des résultats de cette sous-estimation avec celle obtenue par la methode décrite en §4.3.3.

On conclut ce paragraphe concernant le modèle complet en faisant une analyse comparative de la limite inférieure obtenue avec des différentes poussées maximales T_{max} . On observe sur la figure (4.19) que les meilleures sous-estimations sont obtenues pour les poussées maximales les plus petites; ceci est un résultat prévisible, car en diminuant T_{max} on réduit automatiquement le domaine du variation du contrôle, l’analyse de l’intervalle étant beaucoup plus efficace. En contrepartie, le temps de calcul de la limite inférieure est plus important, comme le montre le tableau (4.7).

Conclusions

Avec ce dernier modèle complet, prenant en compte toutes les équations de Gauss en L , on finit l’étude des différentes méthodes de convexification pour le problème de transfert orbital 2D à masse constante. On a présenté

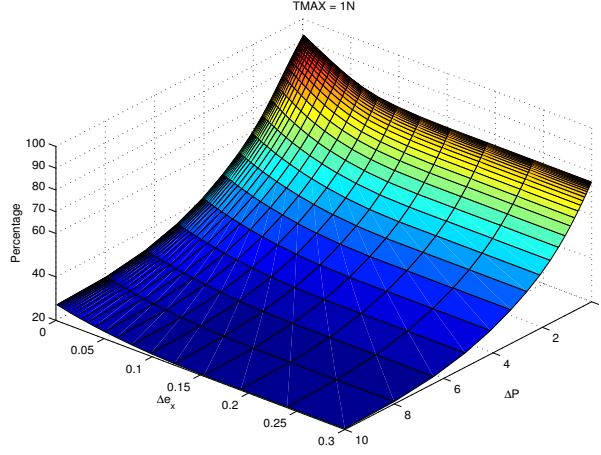


FIG. 4.17 – Pourcentage $\frac{LBD}{UBD}$ en fonction des paramètres ΔP et Δe_x sur le pavé $[0.01, 10.00] \times [0.30, 0.001]$. La meilleure limite inférieure atteint 95% de la limite supérieure. Les résultats sont évidemment meilleurs que ceux obtenus pour les équations de Gauss en fonction du temps.

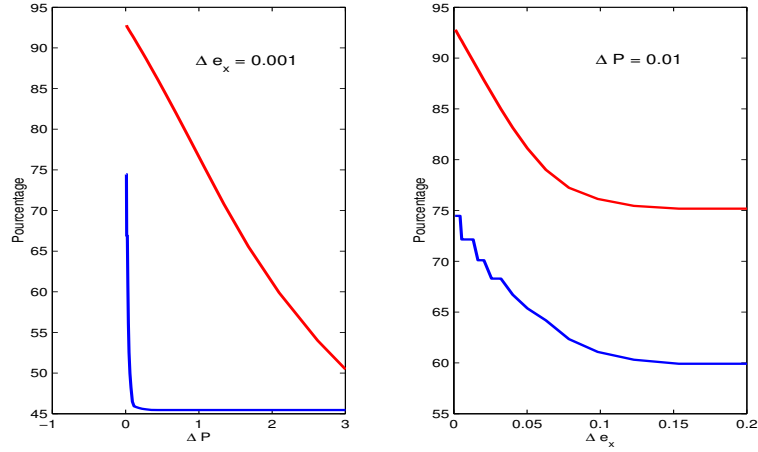


FIG. 4.18 – Comparaison entre les limites inférieures obtenues avec les modèles en P seulement et celui complet, décrit dans ce paragraphe. A gauche, l'évolution du rapport $\frac{LBD}{UBD}$ en fonction du ΔP , pour $\Delta e_x = 0.001$ fixé; en rouge le modèle complet, en bleu le modèle simplifié. A droite, l'évolution du rapport $\frac{LBD}{UBD}$ en fonction du Δe_x , pour $\Delta P = 0.01$ fixé; en rouge le modèle complet, en bleu le modèle simplifié. On observe une nette amélioration pour notre dernière convexification (en rouge).

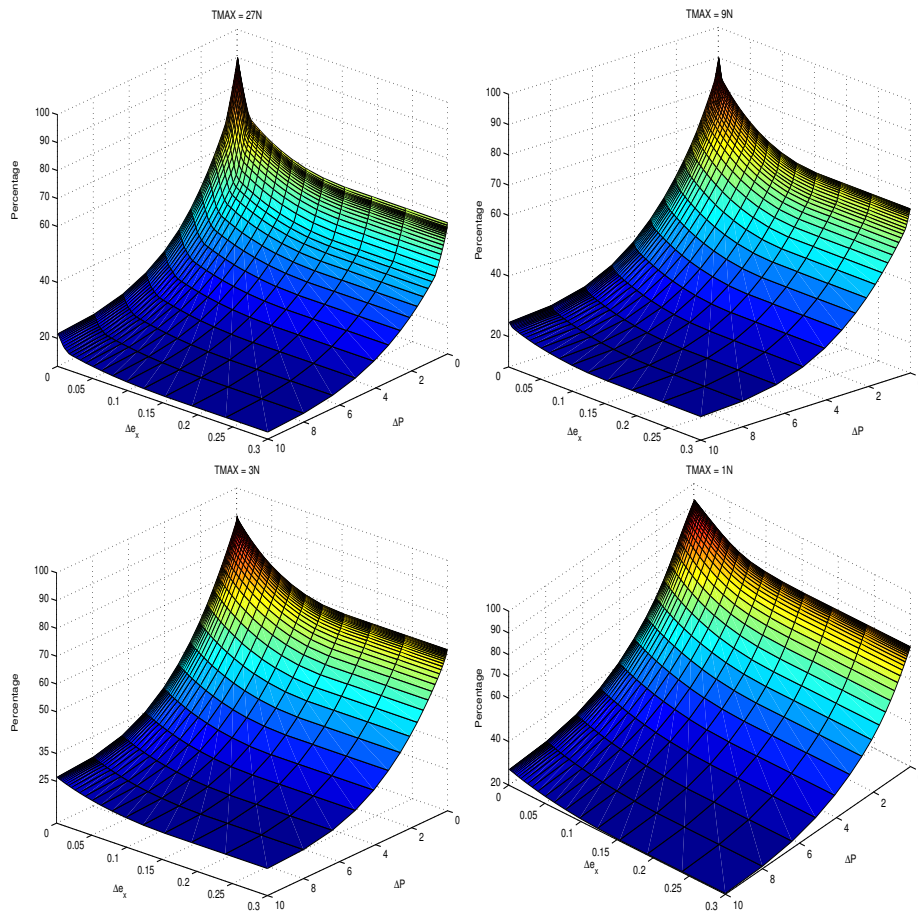


FIG. 4.19 – Comparaison entre les limites inférieures obtenues avec ce modèle complet en faisant varier la poussée maximale. Comme cela était attendu, la qualité de l'estimation inférieure augmente au fur et à mesure que T_{max} diminue.

$T_{max}(N)$	$t_{exec}(s)$
0.8	31.00
1	19.27
2	14.20
5	6.46
10	8.06
20	8.11
30	8.16

TAB. 4.7 – *Les temps d'exécution pour obtenir la limite supérieure, solution locale du problème spatial résolu par la méthode de la collocation directe avec le logiciel KNITRO. On utilise 1000 points de discrétisation et le problème à 7997 variables et 5995 contraintes. Le temps de calcul est plus important pour les petites poussées.*

et détaillé un certain nombre de modèles parmi ceux qu'on a développés et testés; on a choisi pour cette présentation les meilleurs mais aussi ceux permettant d'illustrer les différentes techniques employées.

Notre conclusion est que la meilleure façon de poser le problème est celle décrite en §4.3.3, car:

- ce modèle en L transforme les fonctions $\sin L$ et $\cos L$ en constantes après la discrétisation
- la séparation du contrôle en partie positive et négative assure que seulement un contrôle non nul peut satisfaire les conditions initiales et terminale
- le passage au variables R , E_x , E_y renforce le système d'équations à résoudre en ajoutant une équation linéaire (la forme implicite de la dynamique).

Néanmoins, le désavantage principal de ce modèle est le temps de calcul, très important compte tenu du fait que ce problème relaxé est à résoudre à chaque itération de l'algorithme d'optimisation globale. Ce temps de calcul élevé est justifié par deux facteurs: d'une part le grand nombre de variables dans le modèle et d'autre part par le grand nombre de calculs supplémentaires effectués pour construire le modèle (on parle bien entendu de l'analyse de l'intervalle, utilisée pour obtenir les limites de chaque variable auxiliaire).

4.4 La réduction du domaine

La section précédente nous montre que le dernier modèle présenté fournit la meilleure estimation inférieure, cette formulation se prêtant le mieux à une relaxation convexe. C'est sur cette même formulation qu'on va détailler la technique de la réduction du domaine qu'on a employée. Le principe qu'on

va utiliser est de propager la réduction de l'intervalle d'une variable d'état à l'instant i sur les intervalles de ces voisins (*i.e.* $i - 1, i + 1, i - 2, i + 2$, etc). Dans le début de chapitre, pour l'exemple de l'alunissage, on avait utilisé l'analyse d'intervalle pour cette réduction: la connaissance de l'état à l'instant i , ainsi que la connaissance du contrôle aux instants i et $i + 1$ permet d'obtenir l'expression explicite de l'état à l'instant $i + 1$. Pour le problème de transfert orbital, la meilleure formulation donnée en §4.3.3 s'écrit:

$$\begin{cases} \frac{dR}{dL} &= -\frac{2}{m\mu} \frac{R^2}{w^3} u_2 \\ \frac{dE_x}{dL} &= \frac{\sin L}{m\mu} \frac{R}{w^2} u_1 + \frac{1}{m\mu} \frac{R}{w^3} q_1 u_2 \\ \frac{dE_y}{dL} &= -\frac{\cos L}{m\mu} \frac{R}{w^2} u_1 + \frac{1}{m\mu} \frac{R}{w^3} q_2 u_2 \end{cases} \quad (4.61)$$

où, les variables w, q_1 et q_2 sont définies par:

$$\begin{cases} w &= R &+ E_x \cos L &+ E_y \sin L \\ q_1 &= 2R \cos L &- E_x \sin^2 L &+ E_y \sin L \cos L \\ q_2 &= 2R \sin L &+ E_x \sin L \cos L &- E_y \cos^2 L, \end{cases} \quad (4.62)$$

et le contrôle u_1 et u_2 est le vrai contrôle (sans utiliser une quelconque substitution).

La discrétisation de ce système ne permet plus de donner une expression explicite des états à $i + 1$ en fonction des états et du contrôle à l'instant i ; on a essayé de trouver cette expression en utilisant le calcul symbolique de Matlab, mais nos efforts n'ont pas eu de succès.

Pour réduire le domaine de faisabilité, on propose une méthode algorithmique. En fait, la discrétisation des équations (4.61) relie les états aux instants i et $i + 1$ par un vecteur de fonctions non-linéaires, de la forme:

$$F^i(x^i, x^{i+1}, u^i, u^{i+1}) = 0, \quad i = 0, N - 1, \quad (4.63)$$

avec $x = (R, E_x, E_y)$ l'état et $u = (u_1, u_2)$ le contrôle.

La démarche qu'on propose consiste à réduire successivement les bornes des intervalles de l'état à l'instant $i + 1$ en utilisant l'analyse d'intervalle. En considérant qu'à l'itération k de l'algorithme les intervalles en question sont: $x^i \in I_x^i, x^{i+1} \in I_x^{i+1}, u^i \in I_u^i$ et $u^{i+1} \in I_u^{i+1}$ la procédure donnée dans le Tableau 4.8 permet de réduire la limite supérieure x_{sup}^{i+1} de l'intervalle I_x^{i+1} .

Cette procédure consiste à découper successivement l'intervalle de départ I_x en éliminant des morceaux d'intervalle non-faisable. Une technique similaire est utilisée afin de réduire (en augmentant) la borne inférieure de l'intervalle.

Remarque 4.4.1. Dans le cas où l'état a plusieurs dimensions, la procédure de réduction est adaptée en analysant chaque intervalle à la fois: I_R^{i+1} est choisi en premier et réduit en considérant les autres intervalles fixes, puis on continue avec $I_{E_x}^{i+1}$ et $I_{E_y}^{i+1}$; on arrête au moment où une analyse de ces trois intervalles de R, E_x et E_y ne produit aucune réduction.

```

Soit  $J = I_x^{i+1}$ 
Soit  $J_s = \emptyset$ 
tant que  $\text{diam}(J) \geq \epsilon$  faire
    diviser l'intervalle  $J$  en deux moitiés:  $J = J_{inf} \cup J_{sup}$ 
    calculer l'intervalle  $I = F^i(I_x^i, J, I_u^i, I_u^{i+1})$ 
    si  $0 \in I$  alors
         $J = J_{sup}$  (moitié supérieure de cet intervalle)
    sinon
         $J = J_s$  (moitié inférieure de l'intervalle précédent)
    fin si
    Sauvegarde la moitié inférieure de cet intervalle  $J_s = J_{inf}$ 
fin tant que
Soit  $x_{sup}^{i+1} = \text{borne supérieure de } J$ 

```

TAB. 4.8 – Procédure de réduction de domaine

Remarque 4.4.2. Cette démarche ne réduit pas l'intervalle du contrôle. Ce choix est justifié par le fait qu'on travaille à poussée suffisamment faible (de l'ordre de 2 ou $3N$) et la réduction ainsi faite donne de bons résultats en pratique.

Remarque 4.4.3. Cet même algorithme pour réduire le domaine de faisabilité est utilisé pour réduire les intervalles des variables d'état à l'instant $i - 1$ en connaissant ceux de l'instant i .

On précise aussi que cette technique est utilisée d'une part, au début de l'algorithme pour faire une première réduction en utilisant les conditions initiales et finales du problème (on obtient ainsi le domaine initial) mais aussi après chaque séparation d'un nœud, afin de diminuer les domaines du chaque nœud fils obtenu par la séparation.

4.5 Résultats

Les différentes techniques et adaptations qu'on a présentées auparavant ont été testées sur quelques problèmes de transfert orbital 2D à masse constante. Comme on l'a précisé lors de l'analyse de la convexification, le logiciel commercial CPLEX est utilisé pour calculer le minorant de chaque nœud et KNITRO, le solveur non-linéaire utilisant une méthode des points intérieurs nous permet d'obtenir le majorant de chaque nœud.

Les premiers tests effectués sur des problèmes de transfert orbital ayant un petit nombre de points de discrétisations (ne dépassant pas 15) convergent,

mais les temps de calcul augmente considérablement par rapport à l'exemple traité en début de chapitre. On a ainsi remarqué des résultats plus modestes quant à la taille des problèmes résolus par rapport aux résultats du problème de l'alunissage.

Afin d'analyser l'efficacité de la méthode on va se concentrer sur le cas suivant, celui d'un transfert orbital comportant 3 orbites et ayant une poussée maximale petite ($T_{max} = 2.3N$). On a utilisé 10 points de discrétisation par orbite afin de garder une taille raisonnable pour le problème non-convexe:

$$\begin{aligned}
N &= 30 & T_{max} &= 2.3 \\
L^0 &= \pi & L^f &= L^0 + 6\pi \\
P^0 &= 36.625 & P^f &= 42.165 \\
e_x^0 &= 0.15 & e_x^f &= 0.00 \\
e_y^0 &= 0.00 & e_y^f &= 0.00
\end{aligned} \tag{4.64}$$

Après 100000 itérations et un temps de calcul important (de l'ordre de 10 jours) le plus petit minorant atteint 85% de la meilleure solution locale trouvée (*i.e.* $\frac{UBD-LBD}{UBD} \leq 15\%$). Malgré le petit nombre de points de discrétisation, le grand nombre de termes non-linéaires nous empêche d'obtenir un minorant très efficace, ce qui explique le nombre d'itérations important, d'une part, et le temps de calcul, d'autre part (on peut considérer, grossièrement, le temps de calcul pour une itération constant). La Figure 4.20 présente l'évolution du minorant en fonction du nombre d'itérations (sur les premiers 20000). On remarque au début une croissance importante, car l'arbre de Branch and Reduce a une petite taille, mais qui s'atténue vite avec l'augmentation du nombre des nœuds analysés (on retrouve vite la nature combinatoire du problème).

Une autre mesure importante afin d'analyser l'efficacité de l'algorithme est donnée par le nombre de nœuds éliminés. On se rappelle qu'un nœud peut-être éliminé soit si la relaxation convexe est non-faisable, soit si le minorant du nœud est plus grand que le meilleur majorant. La Figure 4.21 présente le pourcentage des nœuds éliminés en utilisant la sous-estimation. On remarque que l'élimination basé sur la minoration a le rôle le plus important; ceci est un résultat logique en considérant qu'avant de sous-estimer un nœud une analyse basée sur la faisabilité est effectuée et le domaine est réduit (dans la plupart des cas cette analyse détecte la non-faisabilité).

Finalement, la Figure 4.22 montre l'évolution des nœuds éliminés par rapport aux nœuds analysés. Ces résultats sont plutôt encourageants, car à peu près un nœud sur quatre analysés est éliminé. Ceci est dû à la fois à l'élimination basée sur la minoration, mais aussi à la réduction de domaine.

Comme on l'avait prévu, même si le minorant et le majorant s'obtient dans un temps raisonnable (de l'ordre de quelques secondes), l'impact sur le temps de calcul de l'algorithme est très important, en raison du grand nombre de problèmes devant être résolus. Ce qui explique que seulement des

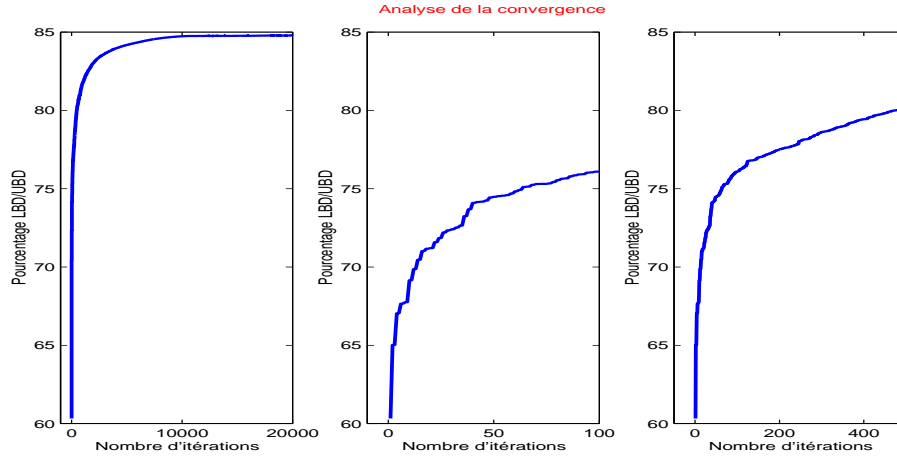


FIG. 4.20 – L'évolution du rapport $\frac{LBD}{UBD}$ en fonction du nombre d'itérations. LBD est le plus petit minorant obtenu à l'itération courante et UBD est le plus petit majorant, l'objectif optimal de la meilleure solution locale trouvée à cette même itération. On remarque une croissance importante au tout début de l'algorithme, là où la séparation est très efficace en raison du petit nombre de nœuds dans l'arbre de Branch and Bound, mais cette croissance s'atténue vite avec la croissance du nombre de nœuds.

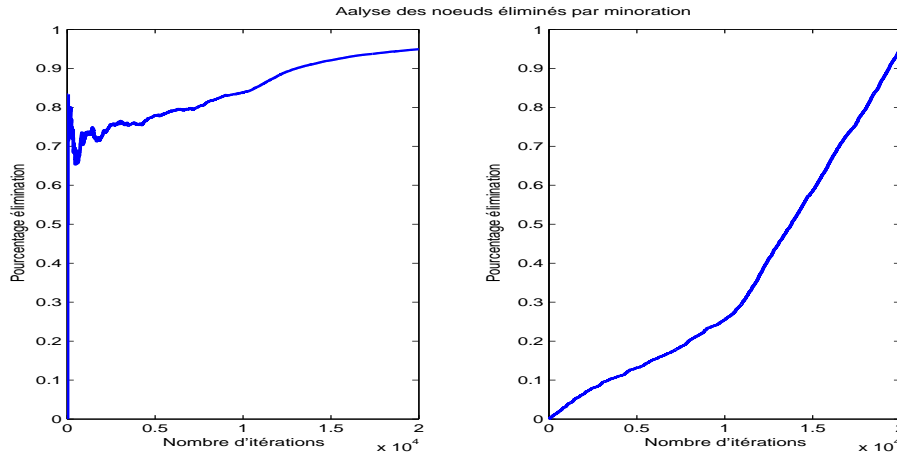


FIG. 4.21 – L'élimination des nœuds est principalement due à la minoration. A gauche, le nombre des nœuds éliminés à l'itération k (en utilisant la minoration) en fonction du nombre total des nœuds éliminés à cette même itération k . A droite, l'évolution du nombre des nœuds éliminés en utilisant la minoration par rapport au nombre total de nœuds éliminés (après 20000 itérations).

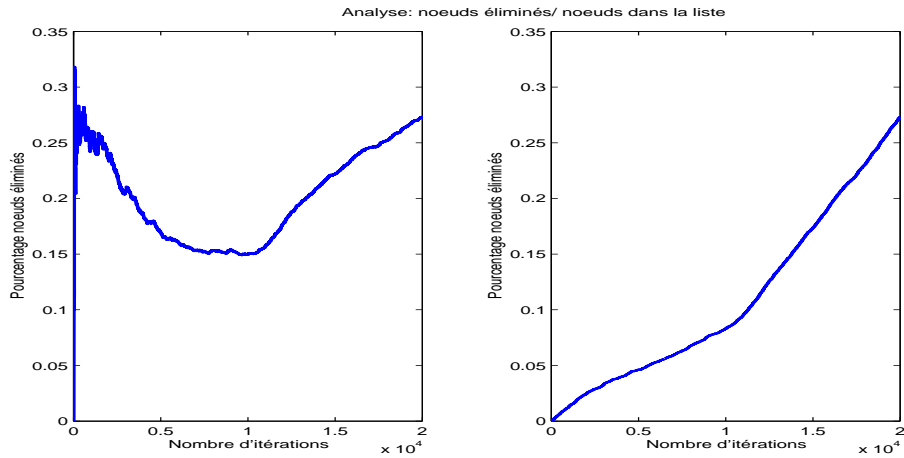


FIG. 4.22 – L’analyse du rapport nœuds éliminés / nœuds dans l’arbre de Branch and Reduce en fonction du nombre d’itérations. A gauche, le nombre de nœuds éliminés à l’itération k par rapport au nombre de nœuds dans l’arbre (à cette même itération k) en fonction du nombre d’itérations. A droite, ce même nombre de nœuds éliminés à l’itération k par rapport au nombre total de nœuds dans l’arbre après 20000 itérations.

problèmes de petite taille ont été résolus dans un temps raisonnable. C’est la limitation la plus importante de cette méthode, limitation qui est aussi la conséquence de la qualité de la minoration. A titre d’exemple, pour le problème de l’alunissage, la sous-estimation atteint 90% de la majoration dès les premières itérations, contre les 60% dans le cas du transfert orbital. Ceci étaye notre conviction que si le problème à résoudre contient un grand nombre de termes non-linéaires, l’efficacité de la méthode est fortement diminuée.

4.6 Conclusions

Ce chapitre a été dédié à l’optimisation globale dans les problèmes de contrôle optimal avec dynamique non-linéaire. Au centre de notre étude se trouve la méthode de Branch and Reduce, méthode susceptible de trouver l’optimum global des problèmes en variables mixtes ou seulement en variables continues. On a essayé d’adapter au mieux cette démarche à notre problème de contrôle optimal, le transfert orbital 2D à masse constante.

On a commencé par tester l’efficacité de la méthode sur un problème non-convexe simple, ayant un nombre limité de termes non-linéaires. Plusieurs techniques (la convexification pour obtenir un minorant, l’analyse d’intervalle pour réduire l’espace de recherche) ont été adaptées à ce problème simple et les résultats obtenus montrent que la taille des problèmes à résoudre peut atteindre une centaine de variables, résultats plutôt encourage-

geants.

Dans le cas du transfert optimal 2D à masse constante, les résultats obtenus sont préliminaires. En fait, à part quelques problèmes ayant peu d'intérêt pratique (10 à 15 points de discrétisation) la nature combinatoire du problème s'avère trop importante, nous empêchant de trouver la solution globale dans un temps de calcul raisonnable (*cf.* §4.5). Et cela, malgré une adaptation de l'algorithme à notre problème spécifique: une convexification obtenue après une analyse préalable de plusieurs modèles, une arithmétique d'intervalle effectuée à chaque itération pour réduire l'espace de recherche et finalement, l'utilisation de solveurs très efficaces afin d'obtenir les minorants et les majorants (CPLEX pour les problèmes quadratiques et Knitro pour les problèmes non-linéaires).

Les résultats obtenus nous laissent croire que des améliorations sont encore possibles (notamment au niveau des choix faits dans le Branch and Bound: la variable choisie pour la séparation, le choix du point de séparation), mais on ne peut guère gagner plus en ce qui concerne la taille des problèmes traités. A notre avis, une amélioration considérable passe obligatoirement par une meilleure sous-estimation de l'optimum global, car, comme on l'a constaté, le minorant qu'on obtient par les techniques de convexification n'est pas suffisant. Une approche basée sur la dualité peut-être envisagée dans ce cas, en espérant que le *saut de dualité* ait une petite valeur.

Conclusion

Bilan des contributions. On s'est proposé d'étudier comment on pourrait intégrer un type particulier de contrainte logique (limitant la durée du contrôle) dans les problèmes de contrôle optimal ayant un objectif quadratique. La démarche qu'on propose passe par la collocation directe pour discrétiser le problème et par la modélisation des problèmes en tant que **MINLP**.

La première analyse importante concerne la contrainte logique, prise indépendamment (ou presque) du problème d'optimisation. On a réussi à caractériser de façon précise le type des vecteurs entiers qu'on cherche et cela nous a permis de réduire l'espace de recherche des vecteurs entiers. En outre, le fait de travailler avec une matrice totalement unimodulaire s'avère très utile dans un algorithme.

On a divisé les problèmes traités en deux grandes classes, cette division étant motivée, *a posteriori*, par les algorithmes différents susceptibles d'être utilisés : d'une part, la classe des problèmes **MINLP** issus d'une dynamique linéaire et, d'autre part, ceux issus d'une dynamique non-linéaire. Pour la première classe, même si une modélisation facile, de type **MIQP** peut être obtenue, notre conclusion est qu'on arrive à traiter de façon très satisfaisante le modèle **MINLP**, le nombre de variables entières dans le modèle pouvant atteindre quelques milliers. La méthode qu'on a développée dans ce but entre dans un cadre assez général, celui de la décomposition de Benders; on exploite, à l'aide de la théorie de la dualité, le fait qu'à variables entières fixées, le problème à résoudre est convexe.

La deuxième classe des problèmes étudiés, issus d'une dynamique non-linéaire, est beaucoup plus complexe. La nature combinatoire se manifeste ici de deux façons distinctes: la non-convexité des équations en variables continues et le domaine discret des variables binaires. Malgré cela, la méthode pour traiter ces deux grands inconvénients est la même, l'algorithme de Branch and Reduce. C'est pour cette raison qu'on a concentré notre travail uniquement au cas des variables continues, en cherchant l'optimum global. Il mérite d'être souligné que l'intégration des contraintes logiques se fait à travers l'algorithme **GBD** développé pour les besoins du cas linéaire. Les résultats obtenus sont préliminaires: pour un exemple simple, au sens où le nombre de termes non-convexes est petit, la taille des problèmes traités

atteint une centaine de points de discrétisation. Dans le cas du problème de contrôle optimal spatial, transfert orbital 2D à masse constante, le nombre de variables reste modeste, de telle façon qu'on arrive difficilement à traiter des transferts plus longs qu'une ou deux orbites. Notre principal apport pour ce type de problème réside dans les méthodes développées pour convexifier le problème et réduire le domaine de définition des variables.

Perspectives. Concernant le cas issu d'une dynamique linéaire, l'ajout de contraintes convexes quadratiques nous semble la première amélioration qu'on peut apporter, et cela dans un souci de généraliser la démarche qu'on propose. Comme la théorie développée dans le chapitre 3 peut toujours s'appliquer dans ce cas, des tests numériques peuvent être envisagés afin de compléter notre étude. Ensuite, le cas où le même type de contrainte logique est intégré dans un problème convexe (non nécessairement quadratique) pourrait être analysé.

Le cas non-linéaire reste néanmoins le plus prioritaire, vu les résultats préliminaires obtenus. Afin d'améliorer les résultats on doit impérativement trouver une meilleure sous-estimation du problème non-convexe. La dualité nous semble une direction de recherche intéressante, mais ceci en supposant que le *saut de dualité* ait une faible valeur. Outre cette amélioration, différentes heuristiques peuvent être analysées afin d'améliorer la convergence de l'algorithme de Branch and Reduce: choix du nœud à diviser, choix de la variable à séparer, etc.

Une autre démarche où on approche successivement le problème non-convexe par des problèmes quadratiques peut-être envisagée; on s'était gardé de tester cette approche car, d'une part, la solution obtenue ne peut être que locale (vu la non-convexité du problème en variables continues) et d'autre part on peut s'attendre à ce que l'intégration des variables binaires empêche la convergence. Malgré ces inconvénients, faute d'une autre approche globale plus efficace, la tendance actuelle dans le domaine (au moins en ce qui concerne les logiciels d'optimisation) semble suivre cette voie [30].

Bibliographie

- [1] C. S. Adjiman, I.P. Androulakis, and C.A. Floudas. Global optimization of MINLP problems in process synthesis and design. *Computational Chemical Engineering*, 21:S445–S450, 1997.
- [2] C.S. Adjiman, S. Dallwig, C.A. Floudas, and A. Neumaier. A global optimization method, α BB for general twice-differentiable constrained NLPs-I. Theoretical Advances. *Computers and Chemical Engineering*, 22:1137–1158, 1998.
- [3] S.C. Aggarwal. An alternative method of integer solutions to linear fractional functionals by a Branch and Bound technique. *Z. Angew. Math. Mech.*, 57:52–53, 1977.
- [4] M. J. Bagajewicz and V. Manousiouthakis. On the Generalized Benders Decomposition. *Computers and Chemical Engineering*, 15(10):691, 1991.
- [5] E. Beale. Branch and bound methods for mathematical programming systems. *Annals of Discrete Mathematics*, 5:201, 1979.
- [6] J.F. Benders. Partitioning procedures for solving mixed variables programming problems. *Numer. Math.*, 4:238, 1962.
- [7] P. Bonneau. Modélisation de Contraintes logiques en Contrôle Optimal avec MISER 3. Master’s thesis, ENSEEIHT-INP Toulouse, Juiller 2003. Rapport de DEA.
- [8] B. Borchers and J. E. Mitchell. An improved branch and bound algorithm for mixed integer nonlinear programming. *Computers and Operations Research*, 4(21):359–367, 1994.
- [9] B. Borchers and J. E. Mitchell. An improved branch and bound algorithm for mixed integer nonlinear programming. *Computers and Operations Research*, 21(4):359–367, 1994.
- [10] J.-B. Caillaud. *Contribution à l’étude du contrôle en temps minimal des transferts orbitaux*. Ph.D. thesis, Institut National Polytechnique de Toulouse, Novembre 2000.
- [11] J.-B. Caillaud. Orbital transfer with maximisation of the mass. Direct Approach. <http://www.enseeiht.fr/caillaud/research/direct.pdf>, Juin 2003.

- [12] J.-B. Caillau, M. Denguir, J. Gergaud, T. Haberkorn, P. Martinon, J. Noailles, and L. Pialot. Trajectoires optimales à poussée continue. Technical Report Contrat 86/776/98/CNES/7462, ENSEEIHT-IRIT UMR CNRS 5505, Octobre 2001.
- [13] J.-B. Caillau, R. Dujol, J. Gergaud, T. Haberkorn, P. Martinon, J. Noailles, and D. Preda. Mise au point d’une méthode de résolution efficace pour les problèmes de contrôle optimal à solution bang-bang, application au calcul de trajectoires à poussé faible. Technical Report Contrat 02/CNES/0257/00 - DPI 500, ENSEEIHT-IRIT UMR CNRS 5505, Juin 2004.
- [14] J.-B. Caillau, J. Gergaud, T. Haberkorn, P. Martinon, J. Noailles, and D. Preda. Mise au point d’une méthode de résolution efficace pour les problèmes de contrôle optimal à solution bang-bang, application au calcul de trajectoires à poussé faible. Technical Report Contrat 02/CNES/0257/00 - DPI 500, ENSEEIHT-IRIT UMR CNRS 5505, Janvier 2003.
- [15] J.-B. Caillau, J. Gergaud, T. Haberkorn, P. Martinon, J. Noailles, D. Preda, and D. Ruiz. Mise au point d’une méthode de résolution efficace pour les problèmes de contrôle optimal à solution bang-bang, application au calcul de trajectoires à poussé faible. Technical Report contrat 772/01/CNES/3001/00, ENSEEIHT-IRIT UMR CNRS 5505, Juin 2002.
- [16] J.-B. Caillau, J. Gergaud, and J. Noailles. Tfmin: Short reference manual. *Optimization Online Digest*, 2002/07/511, 2002.
- [17] J.-B. Caillau and J. Noailles. Sensitivity analysis for time optimal orbit transfer. *Optimization*, 49(4):327–350, 2001.
- [18] J.-B. Caillau, R. Dujol, J. Gergaud, T. Haberkorn, P. Martinon, J. Noailles, and D. Preda. Mise au point d’une méthode de résolution efficace pour les problèmes de contrôle optimal à solution bang-bang, application au calcul de trajectoires à poussé faible. Technical Report Contrat 02/CNES/0257/00 - DPI 500, ENSEEIHT-IRIT UMR CNRS 5505, Octobre 2003.
- [19] L. Cesari. *Optimization—Theory and Applications*. Springer–Verlag, 1983.
- [20] CNES. *Mécanique Spatiale T.1.*, pages 934–937. Cépadulès-Editions Toulouse, 1995.
- [21] M. A. Duran and I. E. Grossmann. A mixed integer nonlinear programming algorithm for process systems synthesis. *AIChE Journal*, 32(4):592, 1986.
- [22] M. A. Duran and I. E. Grossmann. An outer approximation algorithm for a class of mixed integer nonlinear programs. *Mathematical Programming*, 36:307, 1986.

- [23] F. A. Al-Khayyal and H. D. Sherali. On finitely terminating Branch and Bound algorithms for some global optimization problems. *SIAM Journal on Optimization*, 10:1049–1057, 2000.
- [24] R. Fletcher. *Practical Methods of Optimization*. J. Wiley, 1987.
- [25] R. Fletcher and S. Leyffer. Solving mixed integer nonlinear programs by outer approximation. *Mathematical Programming*, 66(3):327–349, 1994.
- [26] R. Fletcher and S. Leyffer. Numerical experience with lower bounds for miqp branch and bound. *SIAM Journal on Optimization*, 8(2):604–616, 1998.
- [27] C. A. Floudas. *Nonlinear and Mixed-Integer Optimization*. Oxford University Press, 1995.
- [28] A. M. Geoffrion. Generalized benders decomposition. *Journal of Optimization Theory and Applications*, 10(4):237, 1972.
- [29] C. Guéret, C. Prins, and M. Sevaux. *Programmation Linéaire*. Editions Eyrolles, 2000.
- [30] S. Heipke. *Applications of optimization with XPRESS*. Dash Optimization, 2002.
- [31] K. Holmberg. On the convergence of the cross decomposition. *Mathematical Programming*, 47:269–316, 1990.
- [32] C.T. Le. *Contrôle optimal et transfert orbital en temps minimal*. PhD thesis, Institut National Polytechnique de Toulouse, 1999.
- [33] S. Leyffer. *Deterministic Methods for Mixed Integer Nonlinear Programming*. PhD thesis, University of Dundee, December 1993.
- [34] S. Leyffer. Generalized outer approximation. Numerical Analysis Report NA 178, Dundee University, 1997.
- [35] S. Leyffer. Integrating sqp and branch-and-bound for mixed integer nonlinear programming. *Computational Optimization and Applications*, 18:295–309, 2001.
- [36] G.P. McCormick. Converting general nonlinear programming problems to separable nonlinear programming problems. Technical Report T-267, The George Washington University, 1972.
- [37] G.P. McCormick. Computability of global solutions to factorable non-convex programs: Part i – convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.
- [38] G.P. McCormick. *Nonlinear Programming: Theory, Algorithms and Applications*. J. Wiley and Sons, 1983.
- [39] M. Minoux. *Programmation mathématique*, volume 1. Dunod, 1983.
- [40] Dash Optimization. *Xpress: Getting started*. Dash Optimization Ltd., 2003.

- [41] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. The mathematical theory of optimal processes. *Interscience*, viii:360, 1962.
- [42] D. Preda and J. Noailles. Approche par approximation extérieure des problèmes nonlinéaires mixtes. Rapport Technique RT/APO/03/04, LIMA-IRIT, Avril 2003.
- [43] D. Preda and J. Noailles. Logical Constraints and LQR Optimal Control Problem. ISMP-2003 Conference, Denmark, August 2003.
- [44] D. Preda and J. Noailles. Convex relaxations for minimum energy orbit transfer problem. MOPTA 2004 Conference, Hamilton, Canada, July 2004.
- [45] D. Preda and J. Noailles. Mixed Integer Programming for a Special Logic Constrained Optimal Control Problem. *Mathematical Programming*, Submitted April 2004.
- [46] H. S. Ryoo and N. V. Sahinidis. Global optimization of nonconvex NLPs and MINLPs with applications in process design. *Computers and Chemical Engineering*, 19:551–566, 1995.
- [47] H. S. Ryoo and N. V. Sahinidis. A branch-and-reduce approach to global optimization. *Journal of Global Optimization*, 8:107–139, 1996.
- [48] N. V. Sahinidis. Baron: A general purpose global optimization software package. *Journal of global optimization*, 8:201–205, 1996.
- [49] N. V. Sahinidis. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8:201–205, 1996.
- [50] N. V. Sahinidis. Baron: Branch and reduce optimization navigator, user’s manual. Technical Report Version 4.0, University of Illinois at Urbana-Champaign, June 2000.
- [51] N. V. Sahinidis and I. E. Grossmann. Convergence properties of generalized benders decomposition. *Computers and Chemical Engineering*, 15(7):481–491, 1991.
- [52] N. V. Sahinidis and M. Tawarmalani. Applications of global optimization to process and molecular design. *Computers and Chemical Engineering*, 24(2):2157–2169, 2000.
- [53] H. D. Sherali and P. J. Driscoll. Evolution and state-of-the-art in integer programming. *Journal of Computational and Applied Mathematics*, 124:319–340, 2000.
- [54] H. J. Sussmann. A maximum principle for hybrid optimal control problems. *38th IEEE Conference on Decision and Control*, December 1999.
- [55] M. Tawarmalani and N. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Kluwer, 2002.
- [56] Oskar von Stryk. *Solution of Optimal Control Problems by Direct Collocation*. Birkhauser, 1993 pp. 129–144.

- [57] R. A. Waltz and J. Nocedal. Knitro user's manual. Technical Report OTC 2003/5, Optimization Technology Center, September 2003.
- [58] L. A. Wolsey and G. L. Nemhauser. *Integer and Combinatorial Optimization*. J. Wiley and sons, 1988. Nemhauser 1988.
- [59] O. Zarrouati. *Trajectoires spatiales*. CNES - CEPADUES, 1987.

Résumé. On se propose d'intégrer un type particulier de contrainte logique dans les problèmes de contrôle optimal à coût quadratique. Une approche numérique directe par collocation conduit à des problèmes de programmation mathématique en variables mixtes. S'intéressant d'abord au cas des systèmes à dynamique linéaire, on propose une variante de la Décomposition de Benders qui nous permet de résoudre des problèmes mixtes de taille importante (au delà de mille variables binaires). Ces résultats sont obtenus grâce aux propriétés induites par la contrainte logique. Dans le cas des problèmes mixtes issus d'une dynamique non-linéaire, la démarche proposée (Branch and Reduce) traite de façon analogue les problèmes d'optimisation globale et ceux en variables mixtes. On s'est limité à l'optimisation globale sur le problème de transfert orbital, étudiant notamment les questions de la convexification et de la réduction du domaine. Les résultats obtenus sont partiels, seuls des problèmes de petite taille ayant été traités.

Mots-clés. contrôle optimal, contrainte logique, programmation mathématique en variables mixtes, Décomposition de Benders, Branch and Reduce

Classification MSC2000. 34K35, 34K60, 90C11, 49M27

Abstract. A cumulative logic constraint is integrated into a quadratic objective optimal control problem. Direct collocation method applied to this optimization problem yields a mixed integer non linear programming problem. We first discuss the case of a linear dynamic driven system and the Generalized Benders Decomposition variant that we propose allows us to solve important size problems (even with more than a thousand binary variables). These results are obtained due to some particular properties of the logical constraint. For the mixed integer non linear problems obtained from a non linear dynamics we used the Branch and Reduce framework. Given the analogous approach of this technique to solve both global continuous and mixed integer optimization problems, we focus on finding the global optimum of the continuous 2D orbit transfer problem. Furthermore, we adapt the proposed approach to our problem by studying different types of convexification and domain reduction. The results obtained are partial, only small size problems have been solved.

Key words. optimal control, logic constraints, mixed integer non linear programming, Generalized Benders Decomposition, Branch and Reduce

MSC2000 Classification. 34K35, 34K60, 90C11, 49M27

ENSEEIH-IRIT, UMR CNRS 5505, 2 rue Camichel, F-31071 Toulouse