UNIVERSITÉ DE NANTES

UFR DES SCIENCES ET TECHNIQUES

Sciences et Technologies
de l'Information et de Mathématiques

Année 2015

N° attribué par la bibliothèque

Algorithmes rigoureux pour l'optimisation nonlinéaire biobjectif

Rigorous algorithms for nonlinear biobjective optimization

THÈSE DE DOCTORAT
Discipline : Informatique
Spécialité : INFORMATIQUE

Présentée
et soutenue publiquement par

Benjamin MARTIN

le 22 Octobre 2014 au LINA, devant le jury ci-dessous

Président	:	Michel RUEHER, Professeur	Université de Nice Sophia-Antipolis
Rapporteurs	:	R. Baker KEARFOTT, Professeur	University of Louisiana at Lafayette
		Luc JAULIN, Professeur	ENSTA-Bretagne

Directeur de thèse : Pr. Laurent GRANVILLIERS, Université de Nantes
Co-encadrant de thèse : Dr. Alexandre GOLDSZTEJN, CNRS
Co-encadrant de thèse : Dr. Christophe JERMANN, Université de Nantes
Laboratoire : **L**ABORATOIRE D'**I**NFORMATIQUE DE **N**ANTES **A**TLANTIQUE.
2, rue de la Houssinière, BP 92 208 – 44 322 Nantes, CEDEX 3.

ALGORITHMES RIGOUREUX POUR L'OPTIMISATION NONLINÉAIRE BIOBJECTIF

Rigorous algorithms for nonlinear biobjective optimization

Benjamin MARTIN

⋈

favet neptunus eunti

Université de Nantes

Benjamin MARTIN

Algorithmes rigoureux pour l'optimisation nonlinéaire biobjectif

Rigorous algorithms for nonlinear biobjective optimization

xi+171 p.

Ce document a été préparé avec LaTeX2e et la classe `these-LINA` version v. 1.30 de l'association de jeunes chercheurs en informatique LOGIN, Université de Nantes. La classe `these-LINA` est disponible à l'adresse :

`http://login.univ-nantes.fr/`

Impression : sommaire.tex – 17/3/2015 – 9:56

Révision pour la classe : these-LINA.cls,v 1.30 2005/08/16 17:01:41 mancheron Exp

Résumé

L'Optimisation de critères nonlinéaires contradictoires, sous contraintes nonlinéaires, apparaît dans de nombreux problèmes, par exemple en ingénierie ou dans des problèmes de localisation. La résolution d'un problème avec m objectifs nécessite de calculer son ensemble de solutions dites Pareto optimales, formant des variétés continues de dimensions $m - 1$ potentiellement morcelées en plusieurs parties disjointes.

Dans cette thèse, nous nous intéressons aux algorithmes rigoureux, i.e. donnant des garanties de résultats, basés sur l'analyse par intervalle pour la résolution de problèmes biobjectifs. Nous proposons une méthode de continuation certifiée qui trace localement les variétés continues de solutions optimales. Cette méthode améliore d'autres techniques similaires de la littérature en proposant une meilleure adaptation à la forme de la variété tracée, ainsi que la prise en compte des contraintes d'inégalités du problème sources de singularités. De plus, nous proposons un algorithme de Branch & Bound (B&B) qui calcule globalement un encadrement vérifié des solutions optimales. Cette méthode intègre des techniques de propagation de contraintes, exploitant notamment les bornes sur les objectifs, afin d'accélérer la résolution. Elle généralise également d'autres approches similaires de la littérature. Enfin, nous discutons la perspective de coupler ces deux méthodes. Une telle approche est prometteuse dans la mesure où le BB converge globalement mais lentement. Ceci est dû aux efforts nécessaire pour couvrir totalement les variétés de solutions, tandis que la continuation est une méthode efficace, mais locale, pour effectuer ce travail.

Mots-clés : Optimisation nonlinéaire biobjectif, Analyse par intervalle, Satisfaction de contraintes numériques, Méthodes de continuation, Branch & Bound

Abstract

Many problems, such as in engineering design or in location problems, require the optimization of several conflicting nonlinear objectives subject to nonlinear constraints. Solving a multiobjective problem involving m objectives implies computing its set of Pareto-optimal solutions, that are in general $m - 1$ dimensional manifolds possibly made of several disjoint connected components.

In this thesis, we are interested in interval-based rigorous algorithms, i.e. with guaranteed results, to solve biobjective problems. We propose a certified continuation method that tracks locally a connected manifold of optimal solutions. This method supplements other techniques from the literature as it adapts finely to the shape of manifolds and deals with singularities resulting from inequality constraints in biobjective problems. We also propose an interval Branch & Bound (B&B) algorithm that globally computes a verified enclosure of the optimal solutions. This method integrates constraint propagation techniques, noticeably exploiting bounds on the objectives, in order to enhance the solving process. It also generalizes other similar approaches from the literature. Eventually, we discuss the perspective of coupling the two techniques. Such an hybrid approach is promising as the B&B converges globally, but slowly. It indeed spends many efforts for covering the manifold of solutions, whereas the continuation is an efficient, but local, technique for building such covering.

Keywords: Biobjective nonlinear optimization, Interval analysis, Numerical constraint satisfaction, Continuation methods, Branch & Bound

Remerciements

Tout d'abord, je souhaite remercier les professeurs Baker KEARFOTT, Luc JAULIN et Michel RUEHER, membre du jury de cette thèse. En particulier, j'exprime mes sincères gratitudes aux professeurs Baker KEARFOTT et Luc JAULIN pour avoir accepté d'être rapporteurs de ce manuscrit. Leur appréciation de mon travail me touche particulièrement. De même, je remercie Michel RUEHER qui a suivit mes travaux au long de ces trois années dans le cadre du comité de suivi de thèse, et qui a accepté d'être examinateur et président du jury. Enfin, je remercie Frédéric SAUBION, second membre du comité de suivi de thèse, qui a également suivi mes travaux durant ces trois ans.

Cette thèse n'aurai sans doute pas abouti aussi rapidement sans le travail d'encadrement de Laurent GRANVILLIERS, Alexandre GOLDSZTEJN et Christophe JERMANN. Mon profond respect va à Laurent GRANVILLIERS, directeur de cette thèse, pour la confiance et le soutien qu'il m'a accordé tout au long de ce travail, et même avant lors de mon stage de master. Je remercie Alexandre GOLDSZTEJN pour sa présence, son bouillonnement intellectuel et sa capacité à me remotiver dans les périodes difficiles. Enfin, mes gratitudes vont à Christophe JERMANN pour sa gentillesse et son souci du détail sans lequel ce manuscrit serait de bien moins bonne facture.

Je souhaite également remercier Brice CHEVALIER, ancien collègue de master qui a initié une partie des travaux sur le Branch & Bound intervalle multiobjectif au cours de son stage de master. Son travail a été un bon point de départ pour le chapitre 5 de cette thèse. Je remercie également (Dr.) Ellen DE SCHEPPER avec qui nous avons travaillé sur la modélisation d'un problème d'optimisation linéaire mixte multiobjectif liée a l'économie verte. Malheureusement, ces travaux sortent de la thématique de cette thèse et ne sont donc pas crédités ici. Néanmoins, l'expérience a été formidable notamment dû au fait qu'Ellen est une personne adorable.

Je remercie également tout le personnel permanent et non permanent de l'université de Nantes et du LINA que j'ai eu l'occasion de côtoyer au long de ces trois années, et particulièrement le personnel administratif qui a tout fait pour que cette thèse se déroule dans les meilleurs conditions. Je remercie en vrac Xavier, Anthony, Evgeny, les deux Florian, Frédéric, Anne-Françoise, les deux Annie, Élodie, Sabine, Séverine, et tous ceux que j'oublie.

Je salue et remercie tous les collègues doctorants (et anciens doctorants) pour tous les bons moments passés ensemble. Aurélien MEREL pour son humour de fort bon goût. Marie PELLEAU, sa gentillesse, ses qualités culinaires et ses coups de mains en LaTeX en milieu hostile. Thomas VINCENT, un homme aux idées lumineuses avec une excellente aptitude à déterminer ce qui ne va pas dans un travail manuel fastidieux une fois celui-ci fini. Audrey CERQUEUS, compagnonne des dits travaux manuels et qui, contrairement à ce qui a pu être écrit dans un autre manuscrit, a très bon goût en matière de thé. Ophélie LACROIX, pour les longs moments de pause à refaire le monde et à discuter GoT. Adrien BOUGOUIN, pour sa fausse naïveté et ses bons mots. La liste pourrait continuer encore plusieurs pages, aussi je remercie en vrac Liza, Bruno, Luis, Fabien, Rima, Amir, Prajol, Laurent, Olivier, Alban, Nicolo, et toutes les autres personnes non cités ici.

Enfin, je remercie tous mes amis et ma famille pour leur soutien et affection au quotidien. Je dédie ce manuscrit à ma nièce Faustine, née durant la rédaction de celui-ci. Sans qu'elle en ait conscience, son sourire m'a aidé à traverser les moments difficiles de la rédaction. Merci à toi petit ange.

Table of contents

Résumé de la thèse 1

1 Introduction 7
- 1.1 The problem . . . 7
- 1.2 Motivation and scope of the thesis . . . 7
- 1.3 Contribution . . . 8
- 1.4 Outline of the thesis . . . 9

Notations 11

2 Preliminaries on Interval Analysis 13
- 2.1 Introduction . . . 15
- 2.2 Basic Definitions . . . 15
  - 2.2.1 Interval arithmetic . . . 16
  - 2.2.2 Interval extension of functions . . . 17
  - 2.2.3 Rounded Computations . . . 19
- 2.3 Constraint satisfaction problems . . . 19
  - 2.3.1 Solving systems of equations . . . 21
  - 2.3.2 Contractors and constraint propagation . . . 32
  - 2.3.3 Search strategy . . . 39
  - 2.3.4 Convergence . . . 40
- 2.4 Global optimization . . . 41
  - 2.4.1 Pruning . . . 45
  - 2.4.2 Bounding . . . 46
  - 2.4.3 Search strategy . . . 47
  - 2.4.4 Convergence . . . 47
- 2.5 Conclusion . . . 48

3 Overview of nonlinear multiobjective optimization 51
- 3.1 Introduction . . . 53
- 3.2 Background on Nonlinear Multiobjective Optimization . . . 53
  - 3.2.1 Definitions and notations . . . 53
  - 3.2.2 Optimality conditions . . . 57
- 3.3 Solving multiobjective problems . . . 62
  - 3.3.1 Scalarization methods . . . 62
  - 3.3.2 Continuation methods . . . 67
  - 3.3.3 Global search methods . . . 70
  - 3.3.4 Performance assessment . . . 77
- 3.4 Conclusion . . . 79

4 Certified Parallelotope Continuation for one-manifold **81**
4.1 Introduction . . . 83
4.2 Contracting, Inflating and Certifying parallelotopes . . . 84
4.3 Parallelotope Continuation ParCont . . . 86
4.3.1 Algorithm Description . . . 86
4.3.2 Properties of the Algorithm . . . 91
4.3.3 Limitations of ParCont . . . 98
4.4 Experiments . . . 99
4.4.1 Influence of the manifold topology . . . 100
4.4.2 Influence of the conditioning . . . 101
4.4.3 Influence of the embedding space dimension . . . 103
4.4.4 Homotopy continuation . . . 104
4.4.5 Control synthesis . . . 106
4.4.6 Conclusion . . . 107
4.5 Adaptation to biobjective optimization . . . 107
4.5.1 Detecting rigorously changes of constraint activity . . . 107
4.5.2 Limitations . . . 110
4.6 Biobjective experiments . . . 111
4.6.1 Illustration of change of active constraints . . . 111
4.6.2 Following many changes of constraint activity . . . 113
4.6.3 Connectivity of the Pareto front through nonoptimal solutions . . . 114
4.7 Conclusion . . . 115

5 Biobjective Branch & Bound **117**
5.1 Introduction . . . 119
5.2 Implementing biobjective interval Branch & Bound . . . 119
5.2.1 Constraint propagation towards feasible nondominated solutions . . . 120
5.2.2 Bounding . . . 123
5.2.3 Search strategy . . . 124
5.2.4 Convergence . . . 125
5.3 Experiments . . . 127
5.3.1 Comparing dominance contractor . . . 128
5.3.2 Comparing search strategies . . . 134
5.4 Discussion and further inverstigations . . . 136
5.4.1 Lower and upper bounding . . . 136
5.4.2 Search strategy . . . 136
5.4.3 Hybridizing direct and inverse Branch & Bound . . . 137
5.5 Conclusion . . . 137

6 Conclusion and Perspectives **139**
6.1 Conclusion . . . 139
6.2 Perspectives . . . 140
6.2.1 Newton-based contractor and first-order conditions . . . 140
6.2.2 Integration of ParCont within Branch & Bound . . . 141
6.2.3 Towards multiobjective optimization . . . 141
6.2.4 Other perspectives . . . 143

A Biobjective Benchmark problems **147**
A.1 LZ3 Modified 147
A.2 Kim and DeWeck (KIM) 147
A.3 Osyczka (OSY) 148
A.4 NBI 148
A.5 SpeedReducer (SR) 148
A.6 MOP 148

B Detailed results of Section 5.3.2 (p. 134) **151**

Bibliography **155**
Bibliography 162

List of figures **163**

Liste of tables **165**

List of definitions **167**

Liste of examples **169**

Résumé de la thèse

La recherche de compromis efficaces entre plusieurs objectifs contradictoires apparaît dans de nombreux problèmes courants. Par exemple en ingénierie, les problèmes de design impliquent souvent la recherche d'un compromis entre l'efficacité de l'objet, sa structure (volume, poids) et son coût. Un autre exemple concerne les problèmes de placements où le choix d'ouvrir, par exemple, un restaurant d'une chaîne implique de prendre en compte les points de vues contradictoires des différents acteurs, notamment le propriétaire de la chaîne et le propriétaire du restaurant [126]. Des contraintes peuvent également s'ajouter aux problèmes. Dans le cas du problème d'ingénierie par exemple, des contraintes de structures doivent généralement être prises en compte.

De plus, trouver seulement un compromis n'est pas suffisant pour le décideur. Afin de prendre une décision, il doit pouvoir comparer divers compromis possibles. Le décideur est également intéressé par le fait qu'un compromis soit le meilleur possible. Autrement dit il recherche des décisions impliquant un compromis optimal, qu'on appelle alors Pareto-optimal, tel qu'il n'existe pas d'autres décisions possibles améliorant tout les objectifs en même temps. On parle alors de problèmes d'optimisation multiobjectifs. Quand un problème multiobjectif n'est pas trop complexe, une bonne connaissance de ce dernier ainsi qu'une part de bon sens permet de prendre une décision satisfaisante. En revanche, quand les problèmes deviennent trop complexes pour être humainement et efficacement traités, on fait appel à une modélisation mathématique du problème qui est ensuite donnée à un algorithme calculant les solutions Pareto-optimales. Cette modélisation exprime les objectifs et contraintes sous la forme d'expressions mathématiques fonctions des variables de décisions et compréhensibles pour une machine. Souvent, les objectifs ou contraintes modélisent des phénomènes complexes qui s'expriment sous la forme d'expressions non-linéaires. C'est à dire qu'il est difficile de savoir comment évolue ces expressions par rapport aux variations des variables de décisions. Par exemple, on peut observer qu'une décision est Pareto-optimale par rapport aux autres décisions voisines alors qu'il en existe une autre bien meilleure. Ceci complique grandement la résolution des problèmes non-linéaires par des algorithmes efficaces. De plus, ces expressions complexes conduisent à des imprécisions numériques sur les calculs effectués, ce qui peut être critique quant à la décision finale.

Motivation

Dans cette thèse, on s'intéresse à l'étude d'algorithmes pour la résolution de problèmes d'optimisation biobjectifs (avec deux critères à optimiser) contraints non-linéaires et où les variables de décision sont continues. On considère que la décision est prise a posteriori de la résolution. Autrement dit, l'ensemble des compromis Pareto-optimaux doit être calculé afin de les proposer au décideur.

La plupart des algorithmes de la littérature pour ce type de problèmes, principalement des méthodes de type métaheuritisque, calculent un ensemble fini de décisions représentant un ensemble de compromis homogène. En revanche, elles ne garantissent pas en général la qualité des résultats obtenus. Ceux ci peuvent être non Pareto-optimaux (parfois même dans leurs voisinages), et l'imprécision numérique des calculs n'est pas évaluée. Si ces algorithmes sont populaires, c'est prin-

cipalement pour deux raisons. La première est que quand les problèmes contiennent beaucoup de variables de décision, une résolution complète afin de trouver tous les compromis Pareto-optimaux globaux devient trop coûteuse en temps de calcul. La seconde est que le modèle sur lequel s'appuie la résolution est souvent approximatif, l'intérêt d'une résolution rigoureuse apparaissant alors limité. Ainsi, les solutions calculées servent au mieux à guider le décideur. Néanmoins, ces mêmes justificatifs peuvent être utilisés pour l'optimisation mono-critère. Or, de nombreuses méthodes de recherche globales et rigoureuses (calculant les optima globaux et prenant compte de l'imprécision numérique), comme les méthodes de Branch & Bound intervalle, existent et ont été étudiées de manière étendue dans la littérature [48, 61, 92, 131].

Il apparaît plutôt que la raison principale pour laquelle les méthodes globales rigoureuses sont si peu étudiées dans le cas multiobjectif est dû à la nature des solutions Pareto-optimales. En effet, celles ci forment des variétés continues (par exemple des courbes dans le cas biobjectif) potentiellement morcelées en plusieurs composants disjoints. Chaque composant représente un ensembles de compromis connexes. La Figure 3.1 (p. 55) illustre ces liens entre les variables de décision et les objectifs. Les méthodes globales rigoureuses convergent lentement vers ce type de solutions. Ajouter à cela le coût de la recherche globale fait que ces méthodes ne sont pas assez compétitives. Il est donc important de pouvoir au mieux s'attaquer aux spécificités des problèmes biobjectifs afin de les résoudre globalement et rigoureusement de manière efficace.

Notre but dans cette thèse est de développer des méthodes de résolutions rigoureuses et efficaces pour la résolution de problèmes non-linéaires biobjectifs. Le point de départ de ces travaux vient de l'idée d'exploiter la nature de variété localement connectées des solutions optimales, et ce de manière rigoureuse.

En étudiant la littérature sur ce type de méthodes, il apparaît que les techniques dites de continuation sont efficaces pour calculer localement des solutions Pareto-optimales connexes. Une méthode de continuation résout localement des systèmes d'équations sous-contraints, i.e. avec moins d'équations que de variables [1]. Les solutions de tels systèmes forment des variétés de dimension égale à la différence entre le nombre de variables et le nombre d'équations. Son application dans le cas biobjectif implique une transformation du problème d'optimisation en un problème de contraintes. Plus précisément, ce système de contraintes exprime des conditions nécessaires d'optimalité des solutions du problème d'optimisation, définie par (3.4) (p. 58). En principe, partant d'une solution (localement) Pareto-optimale, une méthode de continuation calcule localement de manière efficace toutes les autres solutions connectées (localement) Pareto-optimales représentant différents compromis autour de la solution initiale. La rigueur dans cette méthode a pour but d'éviter de sauter d'un composant connecté de solutions vers un autre disjoint, empêchant ainsi un calcul incomplet de tout le composant.

La deuxième idée est l'intégration de la méthode de continuation à l'intérieur d'une méthode de recherche globale rigoureuse. Cette méthode globale a pour but de calculer des solutions initiales pour la continuation. A terme, l'idée est d'éviter à la recherche globale d'effectuer de fastidieux calculs pour trouver les solutions facilement atteignables par continuation.

Contexte de la thèse

Ce manuscrit est construit de la manière suivante. Le Chapitre 2 introduit l'analyse par intervalle : un ensemble d'outils pour effectuer du calcul rigoureux [86, 90, 57, 61, 59]. L'analyse par intervalle comprend le calcul par intervalle : une arithmétique par intervalle remplaçant l'arith-

métique réelle standard, voir Section 2.2 (p. 15). Le but de l'analyse par intervalle est de pouvoir effectuer des calculs et analyses sur des ensemble de valeurs réelles, tout en prenant compte de l'imprécision des calculs numériques en machine. Ses applications sont diverses. Dans le Chapitre 2, nous nous attardons sur deux d'entre elles : la résolution de problèmes de satisfaction de contraintes numériques non-linéaires, voir Section 2.3 (p. 19), et la résolution globale de problèmes d'optimisation non-linéaires mono-objectifs, voir Section 2.4 (p. 41).

Pour la résolution problèmes de satisfaction de contraintes, on note l'existence de méthodes de Newton intervalle. Ces méthodes sont très utiles pour résoudre rigoureusement des systèmes d'équations notamment car elles permettent de construire des preuves d'existence et d'unicité de solutions de ces systèmes à l'intérieur d'intervalles. Elles sont notamment au cœur de la méthode rigoureuse de continuation de Kearfott et Xing [58]. Cette méthode construit pas à pas un pavage par vecteurs d'intervalles, autrement dit des boîtes, d'une variété définie implicitement comme l'ensemble de solutions d'un système d'équations sous-contraint. Une autre technique importante est le calcul par parallélotopes, des boîtes orientés, de Goldsztejn et Granvillers [39]. Ce domaine de calcul permet d'appliquer une méthode de Newton intervalle adaptée à la forme de la variété, gagnant ainsi en efficacité par rapport au domaine de calcul par boîtes. Ces deux travaux ont conduit à une première contribution de cette thèse : la méthode de continuation par domaines de parallélotopes ParCont, décrite plus bas.

Toujours dans la résolution de contraintes, les contracteurs sont des opérateurs intervalles permettant de retirer d'un domaine intervalle les valeurs qui ne satisfont pas un ensemble de contraintes. Ces contracteurs sont typiquement utilisés à l'intérieur d'un algorithme de Branch & Prune, une méthode de résolution globale de problèmes de satisfaction de contraintes. Cette méthode construit un pavage par des boîtes de l'ensemble de solutions d'un système de contraintes. Il est intéressant de noter que le principe du Branch & Prune s'étend naturellement pour la résolution de problèmes d'optimisation via la méthode de Branch & Bound. Le principe de ces méthodes est le même : décomposer le domaine des variables de décisions jusqu'à obtenir des domaines suffisamment petits contenant les solutions (optimales) du problème. La différence principale entre les deux approches est la considération d'une contrainte d'optimalité dans le cadre de l'optimisation. Cela nécessite en particulier le calcul de bornes sur chacun des sous-problèmes induits par la décomposition du domaine des variables de décisions afin d'éliminer de la recherche ceux dont on est sûr qu'ils ne contiennent pas de solutions optimales. De plus, ces bornes permettent de formuler des contraintes, et des contracteurs, permettant de retirer d'un domaine intervalle les valeurs conduisant à de moins bonnes solutions.

Le Chapitre 3 introduit le contexte et la théorie autour de l'optimisation non-linéaire multiobjectif, ainsi qu'un état de l'art des méthodes de résolution. En particulier, les méthodes de scalarisation transforment un problème multiobjectif en un problème mono-objectif paramétré. Ces paramètres visent à obtenir des compromis ciblés des différents objectifs. Leur utilisation pratique les rendent très proches des méthodes de continuation. Dans ce cadre, les travaux de Rakowska et al. [100] et de Hillermeier [53] ont posé les bases de l'application des méthodes de continuation usuelles aux problèmes d'optimisation multiobjectifs. Depuis, d'autres travaux ont suivis, par exemple [120, 50, 98, 74]. Un état de l'art de ces méthodes est proposé dans la Section 3.3.2 (p. 67). Certains de ces travaux proposent notamment des couplages de la continuation avec des méthodes globales mais non rigoureuses. Enfin, la résolution globale de problèmes multiobjectifs est abordée en Section 3.3.3 (p. 70). En particulier, nous présentons les méthodes rigoureuses de Branch & Bound intervalle de la littérature pour ce type de problèmes. Nous avons identifié deux types de méthodes : les méthodes directes construisant le pavage des solutions Pareto-optimales

depuis l'espace des variables de décisions, et les méthodes inverses effectuant ce travail depuis l'espace des objectifs. Ces dernières méthodes utilisent notamment des techniques de résolution de contraintes par intervalles afin de lier des boîtes dans l'espace des objectifs à des boîtes dans l'espace de décision. Les méthodes les plus abouties de la littérature sont la méthode directe de Fernández et Tóth [32, 126] et la méthode inverse de Kubica et Woźniak [66, 64, 65].

Partant de ces bases, plusieurs contributions ont été développées dans cette thèse, voir plus bas.

Contributions

ParCont : une méthode rigoureuse de continuation par parallélotopes

Dans le Chapitre 4, nous présentons une méthode de continuation rigoureuse par intervalle pour le calcul de variétés de dimension 1 définies par un système d'équations sous-contraint. Cette méthode appelée ParCont combine le principe de la méthode de continuation par intervalle de Kearfott et Xing [58] avec le domaine de calculs de parallélotopes issue de Goldsztejn et Granvilliers [39]. Le principe de ParCont est de construire localement un pavage d'une variété de solutions d'un système d'équations sous-contraint par des parallélotopes. L'avantage de ces parallélotopes est qu'ils sont orientés de sorte à s'adapter à la structure de la variété.

Au cœur de ParCont se trouvent des méthodes de Newton intervalle. Ces méthodes produisent des preuves d'existence et d'unicité d'une variété continue de solutions a l'intérieur des parallélotopes. Ainsi, comme dans [58], il est assuré que le pavage contient des solutions et qu'elles sont connectées. Les détails de l'implémentation de l'algorithme ParCont sont donnés dans la Section 4.2 (p. 84) et 4.3 (p. 86), ainsi qu'une étude théorique de la complexité, correction, convergence et terminaison de l'algorithme.

Nous avons ensuite testé expérimentalement ParCont sur divers systèmes d'équations dans la Section 4.4 (p. 99). Nous avons comparé l'usage des parallélotopes par rapport aux boîtes pour effectuer une continuation rigoureuse par intervalle. Il apparaît ainsi que ParCont s'adapte mieux à la forme des variétés, est moins sensible au mauvais conditionnement du problème, et son nombre de pas de continuation est peu affecté par le nombre de variables considérées. De plus, nous avons appliqué ParCont à deux types de problèmes impliquant la résolution d'un système d'équations sous-contraint : le calcul de racines de polynômes complexes par homotopie et le calcul de commandes garanties pour un robot. Dans le premier problème, ParCont s'avère compétitif par rapport à une autre méthode garantie, mais non basé sur l'analyse par intervalle, de la littérature [5]. Le dernier problème illustre une application dans laquelle le calcul rigoureux est nécessaire. Cette première version de ParCont à fait l'objet d'un article [79].

Finalement, dans la Section 4.5 (p. 107), ParCont est adapté pour traiter des problèmes non-linéaires biobjectifs. Cette adaptation concerne la prise en compte des contraintes d'inégalité du problème. En effet, il arrive assez régulièrement que des variétés de solutions Pareto-optimales traversent, ou quitte, les bordures de différentes contraintes d'inégalité. Les points où ont lieu ces changements forment des singularités qui ne peuvent être calculées par ParCont. De plus, la plupart des méthodes de continuation de la littérature s'attaquent à des problèmes d'optimisation biobjectifs sans contraintes d'inégalité. Pour les méthodes les traitant, par exemple [100, 98], des mécanismes sont utilisés pour prendre en compte correctement ces contraintes et éviter les singularités. L'idée que nous avons développée est une adaptation rigoureuse de la méthode de Rakowska et al. [100]. Le principe est de considérer l'ensemble des contraintes d'inégalité actives, i.e. pour

lesquelles la variété suit la frontière lors de la continuation, et de détecter rigoureusement quand la variété actuellement tracée traverse la frontière d'une contrainte inactive ou quitte la frontière d'une contrainte active. En ne considérant qu'un ensemble réduit de contraintes, les singularités liées aux changements de frontières disparaissent. Néanmoins, quand un changement est détecté, l'ensemble des contraintes actives est modifié en conséquence et la méthode de continuation repart depuis ce nouvel ensemble. La détection de ces changements se modélise sous la forme d'un système de contraintes résolu par un algorithme de Branch & Prune dédié.

L'adaptation de ParCont aux problèmes biobjectifs est finalement étudiée expérimentalement dans la Section 4.6 (p. 111). Comme il n'existe, à notre connaissance, pas d'autres méthodes de continuation rigoureuse dans la littérature pour ce type de problèmes, ces expérimentations servent davantage à illustrer comment fonctionne la méthode qu'à la positionner vis à vis de l'état de l'art. Tout d'abord, il apparaît que ParCont permet une analyse fine des ensembles de contraintes actives impliquées dans une variété de solutions Pareto-optimales. De plus, contrairement à d'autres méthodes de la littérature, ParCont calcule entièrement ces composants connexes de solutions tant qu'aucune autre singularité n'est rencontrée et tant que la précision de calcul est suffisante pour construire le pavage. Cela permet d'exploiter totalement la structure de ces variétés de solutions Pareto-optimales, parfois connectées par des solutions non Pareto-optimales. L'adaptation de ParCont aux problèmes non-linéaires biobjectifs a été le fruit d'un autre article [80].

Étude des methodes de Branch & Bound biobjectifs

Nous présentons dans le Chapitre 5 une étude de différentes implémentations des méthodes de Branch & Bound intervalle pour la résolution de problèmes non-linéaires biobjectifs. Principalement, nous proposons d'utiliser un schéma d'algorithme commun aux méthodes directes et inverses et généralisant précisément la version mono-objectif du Branch & Bound.

Ainsi, nous proposons un nouveau type de contracteurs prenant en compte efficacement des solutions bornantes calculées pendant la résolution, a l'instar du cas mono-objectif. Ces contracteurs permettent d'éliminer des valeurs non Pareto-optimales des boîtes de décisions. Ils généralisent d'autres contracteurs de la littérature, notamment la technique d'élagage issue de la méthode directe de Fernández et Tóth [32, 126]. De plus, nous proposons plusieurs améliorations à la méthode inverse de Kubica et Woźniak [66, 64, 65] permettant d'éviter à celle-ci de nombreux et lourds calculs au cours de la résolution. Enfin, nous proposons d'utiliser de nouvelles techniques de la littérature permettant de tester certaines conditions d'optimalité à l'intérieur de boîtes de décision [37]. Ces tests déterminent rigoureusement si une boîte de décision ne contient pas de solutions optimales.

Différentes implémentations sont testées expérimentalement et comparées à la littérature dans la Section 5.3 (p. 127). Ces résultats préliminaires montrent un réel apport des nouveaux contracteurs pour la résolution. De plus, les implémentations que nous proposons améliorent sensiblement la résolution par rapport aux méthodes de la littérature, particulièrement sur les problèmes contraints. Enfin, différentes stratégies de recherche sont évaluées, mais il apparaît pour le moment difficile d'établir lesquelles sont les plus robustes.

Ces analyses sont encore préliminaires. Ainsi, ces différents résultats ainsi que des études plus complètes à venir sont discutés dans la Section 5.4 (p. 136). Principalement, il est important d'étudier plus en détail l'influence de chaque composant du Branch & Bound entre eux. De plus, nous nous sommes pour le moment concentrés sur une comparaison en compétition des méthodes directes par rapport aux méthodes inverses. Les méthodes directes montrent en effet une plus

grande robustesse de résolution par rapport aux méthodes inverses. En revanche, ces dernières arrivent à obtenir assez rapidement un résultat préalable satisfaisant. Il semble alors opportun de voir comment on peut exploiter au mieux les forces des deux approches. Le schéma commun que nous proposons peut nous permettre de définir une méthode à mi-chemin entre une méthode directe et inverse dans un futur proche.

Perspectives

Les perspectives, détaillées dans la conclusion de cette thèse dans le Chapitre 6 sont nombreuses. Principalement, il convient d'étudier le couplage de ParCont avec un algorithme de Branch & Bound. Plusieurs idées émergent, en particulier l'utilisation des parallelotopes construits par ParCont comme régions d'exclusion. Si le Branch & Bound produit une boîte de décision à l'intérieur d'une région d'exclusion, celle-ci peut être éliminée de la recherche car ParCont a déjà trouvé les solutions Pareto-optimales dans cette région.

Nous nous sommes de plus concentrés dans cette thèse uniquement sur les problèmes biobjectifs. Il serait intéressant d'étendre les algorithmes que nous avons développés pour traiter un nombre quelconque d'objectifs, i.e. dont les solutions Pareto-optimales forment des variétés de dimension plus grande (des surfaces dans le cas de trois objectifs par exemple). Cela semble relativement aisée d'adapter nos implémentations du Branch & Bound intervalle pour de tels problèmes. En revanche, cela semble plus compliqué pour ParCont, principalement pour s'assurer que le pavage de ces variétés de plus grande dimension ne perde aucune solution.

Enfin, d'autres pistes à plus ou moins long terme sont ouvertes : l'introduction de contracteurs basés sur les méthodes de Newton intervalle dans le cas de contraintes d'inégalités, l'étude plus poussée d'autres stratégies de recherche dans le cadre du Branch & Bound, et la prise en compte a priori, ou interactivement, des préférences (en termes par exemple de qualité souhaitée sur les objectifs) du décideur au cours de la résolution.

CHAPTER 1

Introduction

1.1 The problem

Finding a trade-off between different conflicting goals or objectives naturally appears in many real-world problems, for example in engineering design or location problems, which in turn necessitate to take a decision based on those different possible compromises. In addition, one wants to find the optimal trade-offs to a given problem in order to be sure that there is no other decision that is strictly improving all goals or objectives. These decisions are called Pareto-optimal. Any decision must also, in general, respect several constraints. These problems are referred to multiobjective optimization problems. In order to find the Pareto-optimal trade-offs, and the induced decisions, a mathematical model of the problem is built and passed to an algorithm that computes these optimal decisions.

A mathematical model expresses the different objectives and constraints as functions of the decision variables. Many mathematical models have to represent complex phenomenon, for example physical aspects of a device in engineering, which are described by nonlinear expressions. Nonlinearity in the model of a problem generally implies a difficult resolution. Indeed, the behavior of the model is difficult to predict given the variation of the decisions variables. Hence, a decision can be seen as Pareto-optimal with respect to neighboring decisions, whereas there may exists another feasible decision that is strictly better on all objectives. In addition, having multiple objectives incites to explore all the possible compromising decisions, which also can be Pareto-optimal only in their local neighborhood. Eventually in practice, solving such problems come up with numerical imprecisions which can be critical for the application of the final decision.

1.2 Motivation and scope of the thesis

This thesis concentrates on biobjective optimization problems (only two objectives are considered) with nonlinear objectives and constraints, and continuous decision variables. In addition, a posteriori decision process is considered : the final decision takes place after the resolution of the problem, implying that a set of compromising decisions have to be computed and proposed to the decision maker.

In that field, most of the popular algorithms in the literature, in particular metaheuristics, focus on finding a representative set of decisions, in the sense that the decisions cover homogeneously the different possible trade-offs. However, they generally do not guarantee anything about the computed decisions : they may be far from the actual globally Pareto-optimal trade-offs (or even not Pareto-optimal in their local neighborhood), while the effects of the numerical imprecision is unknown. The popularity of these algorithms is mainly justified by two reasons :

1. solving globally an optimization problem implies exploring completely the space of feasible decisions which is computationally expensive if there are many decision variables.
2. the mathematical model approximates the original problem. Decisions obtained from the model can at best be used to guide the decision of the original problem.

Nevertheless, the same observations can be made for nonlinear single objective optimization problems although rigorous global algorithms, i.e. algorithms asserting the global optimality of the results such as interval Branch & Bound (B&B), are well developed and used in various applications in this field.

Actually, the main reason why rigorous global algorithms are not well developed in the context of multiobjective problems is intrinsic to the nature of the Pareto-optimal solutions. It is known in single objective optimization in the singular case where the optimal solutions form a continuous manifold, that the convergence of B&B is slow as it requires to globally converge to all the solutions composing the manifold. In multiobjective optimization, Pareto-optimal solutions generically form a manifold of several disjoint components representing different locally connected trade-offs of the objectives. Hence, the few rigorous global algorithms developed for nonlinear multiobjective problems, such as interval B&B, are not popular due to this slow convergence that superimpose to the cost of the rigorous global search.

In this thesis, we are interested in developing efficient rigorous global algorithms for solving nonlinear biobjective optimization problems. The key idea in this thesis is the development of an efficient local, but rigorous, algorithm that traces continuous manifolds of (locally) Pareto-optimal solutions, and to couple this algorithm with an efficient global one. Such couplings have already been proposed for non-rigorous methods and showed promising performances. It is expected that coupling this local algorithm with a rigorous global algorithm will avoid it to spend many computational efforts in computing each connected component of Pareto-optimal solutions, which is left to the local method, and to focus on searching all the disjoint parts of the manifold of Pareto-optimal solutions.

1.3 Contribution

The first aim of this thesis has been the development of an efficient rigorous technique for locally computing connected Pareto-optimal solutions of biobjective problems, specifically a continuation method [1]. Continuation in the context of multiobjective optimization has been studied in some key papers, e.g. Rakowska et al. [100] and Hillermeier [53]. These techniques are based on solving underconstrained systems of equations, whose solutions form manifolds.

Dedicated rigorous methods for solving such systems have been developed, in particular the method from Kearfott and Xing [58] which interleaves interval analysis and continuation for ensuring that all solutions to the system are effectively locally tracked. Another method different from continuation has been developed by Goldsztejn and Granvilliers [39]. It is a global interval-based method that adapts locally to the shape of the manifold of solutions. These two works have lead to the first contribution of this thesis : the rigorous continuation method ParCont that tracks locally one-dimensional manifolds, i.e. curves of solutions implicitly defined as the solutions to an underconstrained system of equations. It combines the locality and rigor of [58] and the robustness of [39]. This work has been published in [79].

This first version of ParCont was not directly related to solving biobjective problems. We then proposed to justify the usage of such methods for biobjective problems. A survey of continuation

methods in the context of multiobjective optimization has been conducted. In addition, we have added some components to ParCont for biobjective optimization problems in order to handle one limitation of many state-of-the-art approaches : the consideration of inequality constraints. This work has lead to a publication in [80].

Eventually, after the local technique ParCont has been precisely established and studied, we have focused on the design of a global method that would compute starting solutions for ParCont, such that all Pareto-optimal solutions of the biobjective problem are rigorously computed. The method we initially thought of consisted in reformulating the biobjective optimization problem into a constraint satisfaction problem whose discrete solution set would contain one solution on each connected component of the Pareto-optimal solutions. Indeed, such a method would have been the ideal complement to ParCont. It however turned out to raise several difficulties, in particular to guarantee that each component of the set of Pareto-optimal solutions is reached. Therefore, we have eventually focused on rigorous global solver, such as interval B&B. Few interval B&B have been developped for nonlinear biobjective optimization. Mainly two different methods have been proposed in the literature : the direct method from Fernández and Tóth [32, 126] and the inverse method from Kubica and Woźniak [66, 64, 65]. We have studied these different methods, and have derived some improvements. Our preliminary analyses and the first experiments we have conducted have arisen many questions, in particular, the influence on the performances of the different components of the B&B. Further investigations are hence required before a robust B&B approach to biobjective optimization can be defined.

1.4 Outline of the thesis

Chapter 2 introduces rigorous computation via interval analysis. In interval analysis, computations with real numbers is replaced by intervals. Dedicated set computations allows to evaluate functions on interval arguments. This gives some guarantees about the values contained in the interval arguments, for example for verifying the satisfaction of a constraint on an interval domain of values. Additionally and practically, interval analysis allows to take into account properly numerical imprecisions of the computations. We present two domains where interval analysis is effectively applied : the resolution of numerical constraint satisfaction problems and the resolution of nonlinear optimization problems. These two domains share many common principles as an optimization problem can be viewed as a particular constraint satisfaction problem. The different notions presented there are used throughout the thesis. In Chapter 3, we present an overview of the state-of-the-art of nonlinear multiobjective optimization. The necessary theoretical background is given as well as a description of well known local and global algorithms for solving such problems. Among local methods, we focus on recent developments of continuation methods that accurately capture locally connected components of optimal solutions. Among global methods, we describe mainly the rigorous interval-based algorithms from the literature. These two methods are the source of our developments. Chapter 4 describes ParCont, a rigorous interval-based continuation technique. ParCont solves locally underconstrained systems of equations whose solutions form a one-dimensional manifold. It uses techniques from interval analysis in order to compute an enclosure with parallelotopes of the manifold of solutions which asserts the existence, uniqueness and continuity of the solutions it contains. Moreover, ParCont advantageously adapts to the shape of manifold of solutions it tracks. Specific adaptation for biobjective optimization is added to ParCont in order to be applicable for biobjective problems containing inequality constraints.

Chapter 5 presents efficient implementations of interval B&B for solving biobjective optimization problems. Implementation of the different methods from the literature under a common scheme is described. Efficient processes that exploit the objectives are proposed. A numerical study of the performances of different possible interval B&B is presented. Finally, extensions of these interval B&B are discussed. The thesis in eventually concluded in Chapter 6.

Notations

Here are described some notations used throughout the thesis. In general, capital letters denote matrices. The transpose of a matrix A is denoted A^T. By default, vectors are column vectors. Capital calligraphic letters denote sets. Subscripts indicates components of a vector/matrix while superscripts denote different elements (e.g. particular vectors). There may be some exceptions to these rules, but notations should be clear enough in their context.

General

$\mathbb{N}$	set of positive integers
$\mathbb{R}$	set of reals numbers
$\mathbb{R}^n$	set of vectors of n real numbers
n, m, p, q	respectively number of decision variables, objectives, inequality constraints and equality constraints
$x = (x_1, \ldots, x_n)$	decision vector/solution
$y = (y_1, \ldots, y_m)$	vector of objective values
$y < y'$	$y_i < y'_i$ for all $i = 1, \ldots, m$
$y \leq y'$	$y_i \leq y'_i$ for all $i = 1, \ldots, m$
$y \lneq y'$	$y \leq y$ and $y \neq y'$
$f = (f_1, \ldots, f_m)$	vector of objective functions
$g = (g_1, \ldots, g_p)$	vector inequality constraint functions
$h = (h_1, \ldots, h_q)$	vector equality constraint functions
$\lambda = (\lambda_1, \ldots, \lambda_m)$	vector of objective multipliers
$r = (r_1, \ldots, r_p)$	vector of inequality constraint multipliers
$s = (s_1, \ldots, s_q)$	vector of equality constraint multipliers
$\mathcal{X}$	set of feasible decision vectors
$\mathcal{Y} = f(\mathcal{X})$	objective image of $\mathcal{X}$
$\mathcal{X}^*$	set of (Pareto) optimal decision vectors
$\mathcal{Y}^*$	objective image of $\mathcal{X}^*$
$\mathcal{Y}_U$	an upper bound set
$\mathcal{Y}_L$	a lower bound set
$f'(x)$	derivative/Jacobian of (vector) function f at x
$\nabla f(x) = (f'(x))^T$	gradient of f at x
$\nabla_i f(x)$	value of the gradient of f at x for the variable x_i, i.e. i^{th} row of the gradient of f at x.
$\nabla^2 f(x)$	Hessian matrix of f at x
$\mathrm{int}(\mathcal{U})$	interior of a set $\mathcal{U} \subseteq \mathbb{R}^n$
$\mathcal{B}(x, \delta)$	ball centered on x of radius $\delta > 0$

Interval analysis

$\mathbb{IR}$	set of real intervals
$\mathbb{IR}^n$	set of vectors of real intervals
$\boldsymbol{x} = [\underline{x}, \overline{x}]$	an interval. $\underline{x}$ and $\overline{x}$ represent resp. the lower and upper bound on this interval
$\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$	a vector of intervals / a box
$\boldsymbol{f} : \mathbb{IR}^n \to \mathbb{IR}^m$	interval extension of a (vector) function f
$[\underline{f(\boldsymbol{x})}, \overline{f(\boldsymbol{x})}] = \boldsymbol{f}(\boldsymbol{x})$	lower bound, resp. upper bound of the interval/box $\boldsymbol{f}(\boldsymbol{x})$
$\widehat{\boldsymbol{x}} = (C, \boldsymbol{w}, \tilde{x})$	a parallelotope, with characteristic (rotation) matrix C, auxiliary box $\boldsymbol{w}$ and center $\tilde{x}$
$\square\mathcal{U}$	interval hull of a set $\mathcal{U}$
$\mathbf{N}$	interval Newton operator
$\mathbf{K}, \mathbf{H}, \mathbf{\Gamma}$	resp. Krawczyk, Hansen-Sengupta and Gauss-Seidel interval Newton operator

CHAPTER 2

Preliminaries on Interval Analysis

2.1 Introduction 15
2.2 Basic Definitions 15
2.2.1 Interval arithmetic 16
2.2.2 Interval extension of functions 17
2.2.3 Rounded Computations 19
2.3 Constraint satisfaction problems 19
2.3.1 Solving systems of equations 21
2.3.2 Contractors and constraint propagation 32
2.3.3 Search strategy 39
2.3.4 Convergence 40
2.4 Global optimization 41
2.4.1 Pruning 45
2.4.2 Bounding 46
2.4.3 Search strategy 47
2.4.4 Convergence 47
2.5 Conclusion 48

2.1 Introduction

Interval analysis (IA) is a modern branch of numerical analysis born in the 60's [86]. It replaces computations with real numbers by computations with intervals of real numbers, providing a framework for handling uncertainties and verified computations, hence rigorousness. It is hence a powerful tool for dealing reliably with any problems implying real-valued variables such as numerical constraint satisfaction and nonlinear optimization.

Interval analysis can be used for different usages. It eases computations over sets of real values and builds verified enclosures of these computations. It can be used to define linear relaxations that bound below and above numerical nonlinear functions on interval domains [127]. It can be used to contract interval domains towards a feasible region defined by a system of numerical nonlinear constraints [6]. Eventually, it can be used to build existence and uniqueness proofs of solutions to nonlinear systems of equations [90].

This chapter presents the notations on interval analysis that will be used throughout the thesis. Then, operations on intervals, known as interval arithmetic, are detailed. Different usages of interval analysis are then described, noticeably numerical constraint satisfaction and global nonlinear optimization. The topics covered by this chapter give a broad overview of interval analysis. Most of the notions presented here, and many others, can be found in [90, 57, 61, 59].

2.2 Basic Definitions

The following gives a definition and notation of an interval.

Definition 2.2.1 (Interval)**.** An interval $\boldsymbol{x}$ is a closed connected subset of $\mathbb{R}$. It is defined, by a lower and an upper bound $\underline{x}, \overline{x} \in \mathbb{R}$, which define the set :

$$\boldsymbol{x} = [\underline{x}, \overline{x}] = \{x \in \mathbb{R} : \underline{x} \leq x \leq \overline{x}\} \tag{2.1}$$

We denote by $\mathbb{IR} = \{\boldsymbol{x} : \boldsymbol{x} \subseteq \mathbb{R}\}$ the set of intervals over the reals. A degenerated interval is a real x defined by the interval $[x, x]$.

The symbols $-\infty$ and $+\infty$ can be used to represent respectively left and right unbounded intervals. A box $\boldsymbol{x} \in \mathbb{IR}^n$ is a vector of intervals defined by a lower and an upper bound vectors $\underline{x}$ and $\overline{x}$. We can define similarly interval matrices as matrices of intervals : $\boldsymbol{A} = (\boldsymbol{a}_{ij})_{1 \leq i \leq n, 1 \leq j \leq m} = [\underline{A}, \overline{A}] \in (\mathbb{IR})^{n \times m}$. An interval or box can be empty, i.e. $\boldsymbol{x} = \emptyset$, e.g. if it results from an operation that discards all values in it.

We define the facets and ϵ-facets of a box the following way :

Definition 2.2.2 (Facets and ϵ-facets)**.** Given a box $\boldsymbol{x} \in \mathbb{IR}^n$ and a dimension i. We denote by $\boldsymbol{x}_i^-$ and $\boldsymbol{x}_i^+$ the ith left and right facets of $\boldsymbol{x}$ defined by :

$$\boldsymbol{x}_i^- = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, [\underline{x}_i, \underline{x}_i], \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n) \tag{2.2}$$

$$\boldsymbol{x}_i^+ = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, [\overline{x}_i, \overline{x}_i], \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n) \tag{2.3}$$

ϵ-facets are analogously defined replacing the interval on the ith component by respectively $[\underline{x}_i, \underline{x}_i + \epsilon]$ and $[\overline{x}_i - \epsilon, \overline{x}]$, with $\epsilon > 0$ a small real value such that ϵ-facets are included in $\boldsymbol{x}$. ϵ-facets can be seen as non-degenerated facets.

Interval and boxes can be used to enclose general sets of real values. The following hull operator defines a closure of a set by means of intervals.

Definition 2.2.3 (Interval hull)**.** The interval hull of a set $\mathcal{U} \subseteq \mathbb{R}^n$, denoted by $\square\mathcal{U}$, is the smallest box $\boldsymbol{x}$ enclosing $\mathcal{U}$. When $\mathcal{U}$ is bounded, its interval hull $\boldsymbol{x}$ is defined as :

$$\forall i,\ \underline{x}_i = \inf\{x_i : x \in \mathcal{U}\} \text{ and } \overline{x}_i = \sup\{x_i : x \in \mathcal{U}\} \tag{2.4}$$

Moreover, given two sets $\mathcal{U}_1, \mathcal{U}_2 \subseteq \mathbb{R}^n$, the operation $\mathcal{U}_1 \vee \mathcal{U}_2$ designates the hull of the union of the two sets.

The hull of a set corresponds to the smallest box enclosing this set. Eventually, we introduce several usual operations on intervals, boxes and interval matrices.

Definition 2.2.4 (Interval operations)**.** The center of an interval $\boldsymbol{x}$ is defined by $\mathrm{mid}(\boldsymbol{x}) := 0.5(\underline{x} + \overline{x})$. The mignitude and magnitude of an interval, respectively denoted $\mathrm{mig}(\boldsymbol{x})$ and $\mathrm{mag}(\boldsymbol{x})$, define respectively the lowest and largest absolute value within this interval, i.e. $\mathrm{mig}(\boldsymbol{x}) = \min_{x \in \boldsymbol{x}}(|x|)$ and $\mathrm{mag}(\boldsymbol{x}) = \max_{x \in \boldsymbol{x}}(|x|)$. The width of an interval is defined by $\mathrm{wid}(\boldsymbol{x}) = \overline{x} - \underline{x}$. The result of these operations are reals in the case of intervals, vectors in the case of boxes, and matrices in the case of interval matrices. The width of a box is generally turned into a single real value using the infinite norm, e.g. $\|\mathrm{wid}(\boldsymbol{x})\| = \max_i |\overline{x}_i - \underline{x}_i|$. The volume of a box, denoted $\mathrm{vol}(\boldsymbol{x})$ is the product of its component widths.

The interior of an interval (or box, or interval matrix) $\boldsymbol{x}$ is denoted by $\mathrm{int}(\boldsymbol{x}) = \{x : \underline{x} < x < \overline{x}\}$. Eventually, given another interval $\boldsymbol{x}'$, the intersection of the two intervals is the (possibly empty) interval defined by $\boldsymbol{x} \cap \boldsymbol{x}' = [\max(\underline{x}, \underline{x'}), \min(\overline{x}, \overline{x'})]$. It is defined componentwise for boxes or interval matrices.

We note that the union of intervals does not, in general, result in an interval, e.g. the union of intervals $[-1, 2] \cup [3, 5]$ is not continuous between 2 and 3. This may sometimes cause difficulties when interval operations result in a set made of the union of intervals. In such cases, the hull of the resulting union is usually computed, losing informations about its possible discontinuity. Union of boxes can be used to enclose sharply any set $\mathcal{U} \subseteq \mathbb{R}^n$. Such an enclosure is called a paving.

Definition 2.2.5 (Paving)**.** Consider a set $\mathcal{U} \subseteq \mathbb{R}^n$. A *paving* $\mathcal{P}$ of $\mathcal{U}$ is a set of boxes $\boldsymbol{x}$ such that :

$$\mathcal{U} \subseteq \bigcup_{\boldsymbol{x} \in \mathcal{P}} \boldsymbol{x} \tag{2.5}$$

A paving is *regular* if no box in the paving have overlapping interiors, i.e. $\mathrm{int}(\boldsymbol{x}) \cap \mathrm{int}(\boldsymbol{x}') = \emptyset$ for any $\boldsymbol{x} \neq \boldsymbol{x}' \in \mathcal{P}$.

From this definition, $\{\square\mathcal{U}\}$ is a regular paving of $\mathcal{U}$ containing one single box. Hence, the use of a paving is usually to represent $\mathcal{U}$ sharper than $\square\mathcal{U}$, noticeably as a paving allow to describe the possible discontinuity of $\mathcal{U}$.

2.2.1 Interval arithmetic

An arithmetic operation $\circ$ is extended to intervals in the following way :

$$[\underline{x}, \overline{x}] \circ [\underline{y}, \overline{y}] = \{x \circ y : x \in [\underline{x}, \overline{x}], y \in [\underline{y}, \overline{y}]\}. \tag{2.6}$$

In the case of the arithmetic operations $\{+, \cdot, -, /\}$, they can be simply defined as follows :

$$\boldsymbol{x} + \boldsymbol{y} = [\underline{x} + \underline{y}, \overline{x} + \overline{y}] \tag{2.7}$$

$$\boldsymbol{x} - \boldsymbol{y} = [\underline{x} - \overline{y}, \overline{x} - \underline{y}] \tag{2.8}$$

$$\boldsymbol{x} \cdot \boldsymbol{y} = [\min(\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}), \max(\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y})] \tag{2.9}$$

$$\boldsymbol{x}/\boldsymbol{y} = \boldsymbol{x} \cdot [1/\overline{y}, 1/\underline{y}], 0 \notin \boldsymbol{y} \tag{2.10}$$

They all yield to an interval, except the division operator when $0 \in \boldsymbol{y}$. In that situation, several cases have to be considered : $\boldsymbol{x}/[0,0] = \emptyset$; $\boldsymbol{x}/[0,\overline{y}] = \boldsymbol{x} \cdot [1/\overline{y}, +\infty]$ if $\overline{y} > 0$; $\boldsymbol{x}/[\underline{y},0] = \boldsymbol{x} \cdot [-\infty, 1/\underline{y}]$ if $\underline{y} < 0$; $\boldsymbol{x}/\boldsymbol{y} = [-\infty, +\infty]$ if $\underline{y} < 0$ and $\overline{y} > 0$. This latter case does not verifies (2.6), but this can done by considering an extended division that produces union of intervals, i.e. $1/\boldsymbol{y} = [-\infty, 1/\underline{y}] \cup [1/\overline{y}, +\infty]$. This extended division gives more information than the standard one, but implies manipulating unions of intervals. Hence, the extended division is usually dedicated for specific operation, see for example Section 2.3.2.2.

Most properties of the arithmetic operators are valid for interval arguments : commutativity, e.g. $\boldsymbol{x}+\boldsymbol{y} = \boldsymbol{y}+\boldsymbol{x}$ or associativity, e.g $(\boldsymbol{x} \cdot \boldsymbol{y}) \cdot \boldsymbol{z} = \boldsymbol{x} \cdot (\boldsymbol{y} \cdot \boldsymbol{z})$. However, distributivity is generally not satisfied. Precisely, $\boldsymbol{x}(\boldsymbol{y} + \boldsymbol{z}) \subseteq \boldsymbol{x} \cdot \boldsymbol{y} + \boldsymbol{x} \cdot \boldsymbol{z}$. This is due to the multiple occurrences of the interval $\boldsymbol{x}$ in the expression. Interval arithmetic does not take into account the dependency between the different occurrences of $\boldsymbol{x}$, yielding to an overestimation of the result. Similarly, $\boldsymbol{x} - \boldsymbol{x}$ is not equal to the degenerated interval 0 in general. The square of an interval cannot be computed as $\boldsymbol{x} \cdot \boldsymbol{x}$ without overestimations. A dedicated evaluation of the square (more generally of the power) function has to be derived. For example, $\boldsymbol{x} \cdot \boldsymbol{x}$, with $\boldsymbol{x} = [-3, 4]$, equals the interval $[-12, 16]$ while $\boldsymbol{x}^2$ equals the interval $[0, 16]$.

Continuous unary elementary functions like $\exp$, $\ln$, $\sin$, etc., are also extended to intervals similarly for interval arguments contained in their domains. Given a continuous unary function $f : \mathcal{U} \to \mathcal{V}$, then $f(\boldsymbol{x}) = \{f(x) : x \in \boldsymbol{x} \cap \mathcal{U}\}$ which is an interval since f is continuous. All these elementary interval extensions of arithmetic operators and unary function form the interval arithmetic. As real numbers are identified to degenerated intervals, the interval arithmetic actually generalizes the real arithmetic, and mixed operations like $1 + [1, 2] = [2, 3]$ are interpreted using (2.6).

2.2.2 Interval extension of functions

Let f be a function $f : \mathbb{R}^n \to \mathbb{R}^m$ [1]. An interval function $\boldsymbol{f} : \mathbb{IR}^n \to \mathbb{IR}^m$ is an interval extension of f if it satisfies the containment principle, i.e :

$$f(\boldsymbol{x}) = \{y : y = f(x), x \in \boldsymbol{x}\} \subseteq \square f(\boldsymbol{x}) \subseteq \boldsymbol{f}(\boldsymbol{x}) \tag{2.11}$$

Hence, the interval extension of a function f allows to compute a verified enclosure of the evaluation of f over an interval (or box) domain $\boldsymbol{x}$. Given $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, the values $\underline{y}$ and $\overline{y}$ gives respectively a lower and an upper bound of f over $\boldsymbol{x}$ (that can be noted $\underline{y} = \underline{f(\boldsymbol{x})}$ and $\overline{y} = \overline{f(\boldsymbol{x})}$). However, as stated previously, interval operations can generate overestimations. The quality of those bounds is subject to pessimism.

Nevertheless, additional properties of interval extension, such as convergence, can assert better enclosures.

1. For convenience, we will suppose in this thesis f is defined on $\mathbb{R}^n$ and its image on $\mathbb{R}^m$ although any results can be derived for general functions defined on any domain of definition.

Definition 2.2.6 (Convergent interval extension)**.** An interval extension $\boldsymbol{f}$ is *convergent* if for any bounded sequence of boxes $\boldsymbol{x}^{(k)}$:

$$\lim_{k\to\infty} \mathrm{wid}(\boldsymbol{x}^{(k)}) = 0 \Longrightarrow \lim_{k\to\infty} \mathrm{wid}(\boldsymbol{f}(\boldsymbol{x}^{(k)})) = 0 \tag{2.12}$$

In other words, a convergent interval extension $\boldsymbol{f}$ ensures that its overestimations reduces with the width of the interval arguments. Therefore, $\boldsymbol{f}(\boldsymbol{x})$ gets closer to $\square f(\boldsymbol{x})$, and implies that $\boldsymbol{f}(x) = f(x)$ for any degenerated interval x.

Another interesting property is the inclusion monotonicity of an interval extension.

Definition 2.2.7 (Inclusion monotonic interval extension)**.** An interval extension $\boldsymbol{f}$ is *inclusion monotonic* if :

$$\boldsymbol{x} \subset \boldsymbol{x}' \Longrightarrow \boldsymbol{f}(\boldsymbol{x}) \subset \boldsymbol{f}(\boldsymbol{x}') \tag{2.13}$$

An inclusion monotonic interval extension ensures that its overestimation follows a monotonic reduction.

The most straightforward interval extension of a function f is the natural extension. It consists in replacing all arithmetic and unary operators in the expression of f by their interval counterparts. The natural extension of f is denoted $\boldsymbol{f}^N$. However, we will generally assume throughout the thesis that the notation $\boldsymbol{f}$ for an interval extension will refer to the natural extension. If f is continuous, then the natural extension is convergent and inclusion monotonic. Moreover, the natural extension is an optimal extension, i.e. without overestimation [2], if each variable occurs only once in the formal expression of f.

Example 2.2.1 – Let the function $f(x) = x^2 + 4x + 4$, with $x \in \boldsymbol{x} = [-5, 3]$. Let the two natural extensions of f be :

$$\boldsymbol{f}^1(\boldsymbol{x}) = \boldsymbol{x}^2 + 4\boldsymbol{x} + 4$$
$$\boldsymbol{f}^2(\boldsymbol{x}) = (\boldsymbol{x} + 2)^2$$

The second expression has reduced the formal expression of f such that x appears only once. Evaluating these interval extension over $[-5, 3]$ yields to :

$$\boldsymbol{f}^1([-5,3]) = [-5,3]^2 + 4[-5,3] + 4 = [0,25] + [-20,12] + 4 = [-16,41]$$
$$\boldsymbol{f}^2([-5,3]) = ([-5,3] + 2)^2 = [-3,5]^2 = [0,25]$$

One can see that the latter expression gives less overestimation than the former. In addition, as x appears only once, the produced enclosure is optimal : the bounds 0 and 25 are attained respectively at $x = -2$ and $x = 3$.

Reducing the number of occurrences of variables in the formal expression of f is difficult in general. Hence, overestimations cannot be avoided. Nevertheless, other interval extensions have been derived so as to reduce overestimation.

The centered (or mean value) form $\boldsymbol{f}^c$ in a box $\boldsymbol{x}$ is an interval extension defined as

$$\boldsymbol{f}^c(x) = \boldsymbol{f}(\tilde{x}) + \boldsymbol{f}'(\boldsymbol{x})(x - \tilde{x}) \tag{2.14}$$

2. Precisely, the interval returned by the natural extension of f over a box $\boldsymbol{x}$ corresponds to the hull of the image of $\boldsymbol{x}$ by f. Thus, possible discontinuity of f are not considered.

with $\tilde{x} \in \boldsymbol{x}$ and with $\boldsymbol{f}$ and $\boldsymbol{f}'$ being interval extensions (usually natural) of respectively the function f and its derivative f'. Usually, $\tilde{x} = \mathrm{mid}(\boldsymbol{x})$. This interval extension gives rise to better enclosure than the natural extension when the interval arguments are small. Precisely, the overestimation of the natural extension reduces linearly with respect to the width of the interval arguments, and quadratically for the centered form. In general, it is expected that the natural extension is better suited for evaluating large intervals (or boxes) whereas the centered form is more accurate on small ones.

2.2.3 Rounded Computations

As real numbers are approximately represented by floating point numbers in computers [35], the interval arithmetic cannot match the property (2.6) exactly. In order to preserve the containment principle, the interval arithmetic has to be implemented using an outward rounding. For example, $[1,3]/[10,10] = [0.1, 0.3]$ while both 0.1 and 0.3 cannot be exactly represented with standard binary floating point numbers. Therefore, the computed result will be $[0.1^{\nabla}, 0.3^{\Delta}]$ where 0.1^{∇} (respectively 0.3^{Δ}) is a floating point number slightly smaller than 0.1 (respectively slightly greater than 0.3). Of course, a good implementation will return the greatest floating point number smaller than 0.1 and the smallest floating point number greater than 0.3. Some care have to be taken when implementing floating point operations within interval arithmetic. For example, the midpoint of an interval $\boldsymbol{x}$ is in general computed approximately by floating arithmetic. Thus, we have hence to ensure, and we assume, that at least $\mathrm{mid}(\boldsymbol{x}) \in \boldsymbol{x}$ holds [42].

Among implementations of IA, we can cite the C/C++ libraries PROFIL/BIAS [63] and Gaol [41], the Matlab toolbox INTLAB [112] and Mathematica [133]. The developments in this thesis involve real intervals. They all hold when implemented by a correctly rounded interval arithmetic.

2.3 Constraint satisfaction problems

Constraint satisfaction problems consist in finding the set of solutions to a constraint system defined by a set of equalities and inequalities. A Numerical Constraint Satisfaction Problem (NCSP) is defined as follows :

$$\left[\begin{array}{c} g(x) \leq 0, h(x) = 0 \\ x \in \boldsymbol{x}^{\mathrm{init}} \end{array}\right]. \tag{2.15}$$

where $g : \mathbb{R}^n \to \mathbb{R}^p$ is a vector of inequality constraints, $h : \mathbb{R}^n \to \mathbb{R}^q$ is a vector of equality constraints and the box $\boldsymbol{x}^{\mathrm{init}} \in \mathbb{IR}^n$ is the domain of variables. The solution set of the NCSP is denoted by $\mathcal{X} = \{x \in \boldsymbol{x}^{\mathrm{init}} : g(x) \leq 0, h(x) = 0\}$. Solving the NCSP (2.15) consists in computing $\mathcal{X}$ up to a prescribed precision. This is usually done by computing a (regular) paving of the set $\mathcal{X}$. Building this paving rigorously requires to test satisfaction of a constraint over a box domain. Thanks to the containment principle of interval extensions of functions, simply evaluating an interval extension of a constraint on a box allows to derive satisfaction results.

Definition 2.3.1 (Constraint satisfaction over intervals)**.** Let $\boldsymbol{x} \in \mathbb{IR}^n$ be a box. Consider an inequality constraint $g(x) \leq 0$ with $g : \mathbb{R}^n \to \mathbb{R}$ and an interval extension $\boldsymbol{g}$. Then the constraint g is :

- *certainly satisfied* over $\boldsymbol{x}$ if $\overline{g(\boldsymbol{x})} \leq 0$;

Algorithm 1: Branch & Prune

```
   Input: A NCSP, a box x^init
1.1  S ← {x^init} ;
1.2  S_out ← ∅ ;
1.3  while S ≠ ∅ do
1.4      x ← Extract(S) ;
1.5      x ← Prune(NCSP, x) ;
1.6      if x ≠ ∅ then
1.7          if x is terminal then
1.8              S_out ← S_out ∪ {x};
1.9          else
1.10             S ← S ∪ Split(x) ;
1.11         end
1.12     end
1.13 end
1.14 return S_out
```

- *certainly not satisfied* over $\boldsymbol{x}$ if $\underline{g(\boldsymbol{x})} > 0$;
- *possibly satisfied* over $\boldsymbol{x}$ otherwise.

Consider an equality constraint $h(x) = 0$ with $h : \mathbb{R}^n \to \mathbb{R}$ and an interval extension $\boldsymbol{h}$. Then the constraint h is :

- *certainly satisfied* over $\boldsymbol{x}$ if $\boldsymbol{h}(\boldsymbol{x}) = [0, 0]$;
- *certainly not satisfied* over $\boldsymbol{x}$ if $0 \notin \boldsymbol{h}(\boldsymbol{x})$;
- *possibly satisfied* over $\boldsymbol{x}$ otherwise.

Note that due to overestimation, an equality constraint is certainly satisfied only if $\boldsymbol{x}$ is a degenerated box, which rises difficulties in verifying the satisfiability of equality constraints over interval domains as described below.

Building the paving of $\mathcal{X}$ is usually done using Branch & Prune (B&P), see Algorithm 1. This algorithm implements the decomposition of the initial box $\boldsymbol{x}^{\text{init}}$ into a paving. Each box $\boldsymbol{x}$ of the paving that is certainly not satisfying a constraint is discarded by the procedure *Prune* at line 1.5 by returning an empty box. If $\boldsymbol{x}$ is not empty, it is tested for termination. A box $\boldsymbol{x}$ is terminal if it certainly satisfies all constraints (it is then called an inner box) or if the width of each of its interval components $\boldsymbol{x}_i$ drops below a prescribed precision ϵ_i (then $\boldsymbol{x}$ is called a boundary or outer box). Eventually, if $\boldsymbol{x}$ is not terminal, then it is split at line 1.10 so as to achieve a sharper paving of $\mathcal{X}$. This is usually done by bisecting one variable domain of $\boldsymbol{x}$.

The complexity of B&P is exponential in the number of variables due to the necessity of splitting. Reducing the number of splits can be done by constructing as quickly as possible inner boxes, or by narrowing boxes as much as possible. Detecting inner boxes in the case of equality constraints is in general not possible due to overestimations of interval computations. Instead, proofs of existence and uniqueness of solution to equations in a box $\boldsymbol{x}$ are performed using interval Newton methods. These methods are presented in Section 2.3.1. In order to accelerate the reduction of paving without splitting, the procedure *Prune* has to make use of contractors, operators that contract a box $\boldsymbol{x}$ towards the solutions it contains. Contractors are based on consistency notions of

constraints over a box domain. The consistency of a domain with respect to a NCSP asserts some property of satisfiability of the constraint system. A globally consistent domain for a NCSP (2.15) is any subset of $\mathcal{X}$. Considering a box domain $\boldsymbol{x}$, it is globally consistent only if it is an inner box for the NCSP. This is the aim of Branch & Prune, thanks to splitting, to produce such globally consistent boxes. Contractors are instead based on local consistency over interval domains. Contractors, are often used inside a constraint propagation algorithm that applies successively a set of contractors on a box so as to achieve more narrowing. Contractors, local consistency, and constraint propagation are presented in Section 2.3.2. Eventually, we discuss the search strategy, i.e how to split boxes and how to extract them, in Section 2.3.3. The search strategy is also critical for the performances of the B&P.

2.3.1 Solving systems of equations

Here we consider the restriction to NCSP with only equality constraints. Therefore, we consider the solving of the following system of nonlinear equations :

$$F(x) = 0, \tag{2.16}$$

with $F : \mathbb{R}^n \to \mathbb{R}^q$. Before presenting interval methods, we first briefly present numerical classical approaches on which interval methods are based on.

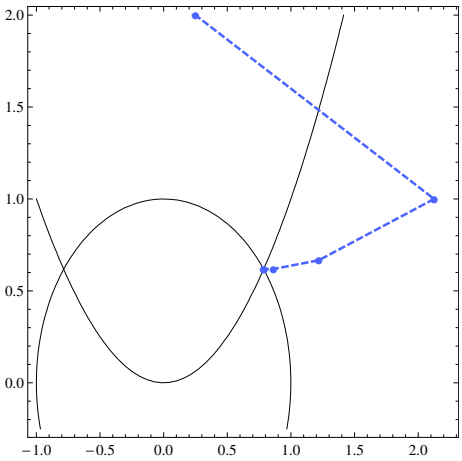
2.3.1.1 Classical numerical methods

When $q = n$, i.e. (2.16) is a square system of equations, a well known technique to solve it is the Newton method. The Newton method is a fixpoint algorithm. It aims at finding the solution x satisfying $x = \Phi(x)$. If the operator Φ is convergent, then it is expected that a sequence x_k of solutions such that $x_{k+1} = \Phi(x_k)$ will converge (approximately) to the fixpoint, i.e. there is a $\bar{k} > 0$ such that $x_{\bar{k}} \approx \Phi(x_{\bar{k}})$. The operator Φ is defined such that it consists in solving approximately and iteratively the linearized equations :

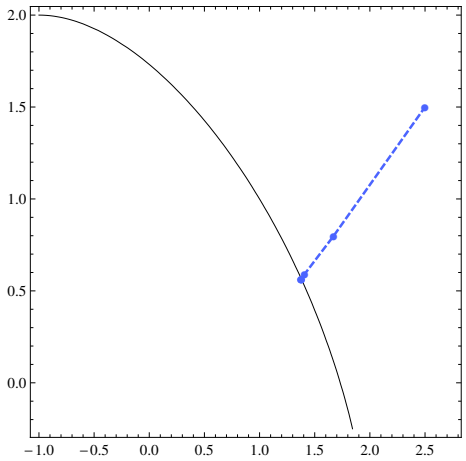
$$F(x_k) + F'(\tilde{x}_k)(x_{k+1} - x_k) = 0 \Leftrightarrow x_{k+1} = x_k - (F'(x_k))^{-1}F(x_k). \tag{2.17}$$

Hence, $\Phi(x) = x - (F'(x))^{-1}F(x)$. Therefore, the fix point of Φ is a zero of (2.16). It is known that in general if x_0 is not too far from a zero of (2.16), the sequence of x_k defined by (2.17) converges quadratically to this zero. Implementing the Newton method simply requires inverting the square matrices $F'(x_k)$. This method is illustrated on an example on Figure 2.1(a). The system (2.16) requires regularity at its solutions in order to ensure the convergence. Matrix inversion is computationally costly for large matrices (complexity of $O(n^3)$). Hence, either the inverse is not computed at each iteration (i.e. $(F'(x_k))^{-1}$ is reused for several iterations) or Quasi-Newton methods are used instead, in which an approximation of $(F'(x_k))^{-1}$ is maintained.

When the system (2.16) is rectangular and underconstrained, i.e. $q < n$, the solutions of (2.16) form, under regularity assumptions, a manifold of dimension $m = n - q$. The Newton method can be used to find some solutions, either by fixing m variables (applying the classical Newton method on the remaining q variables), or by using a generalized inverse of F' (e.g. Moore-Penrose pseudo-inverse). The core of the method remains the same. This is depicted on Figure 2.1(b). Once a solution x to (2.16) is obtained, one can find other solutions along the manifold using continuation methods.



(a) Solving $x_1^2 + x_2^2 - 1 = 0$ and $x_1^2 - x_2 = 0$. Newton method starting from $(0.25, 2)$. The sequence of iterates are in dashed blue lines



(b) Solving $x_1^2 + x_2^2 + x_1 x_2 - 3 = 0$. Newton method starting from $(2.5, 1.5)$ and using generalized inverse (dashed blue lines)

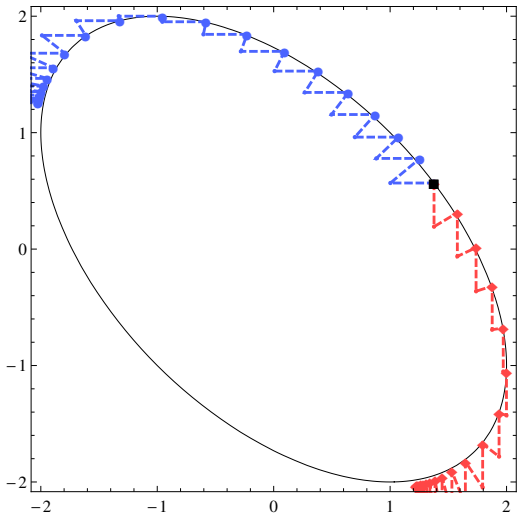
Figure 2.1 – Newton methods applied to a square and a rectangular system of equations

Numerical continuation methods are a class of techniques that track locally solutions to a underconstrained system of equations starting from an initial solution [1]. Under regularity assumptions, the implicit function theorem states that solutions to this system form a $m = n - q$ dimensional manifold : curves for $m = 1$, surfaces for $m = 2$, etc. The implicit function theorem can be stated as follows :

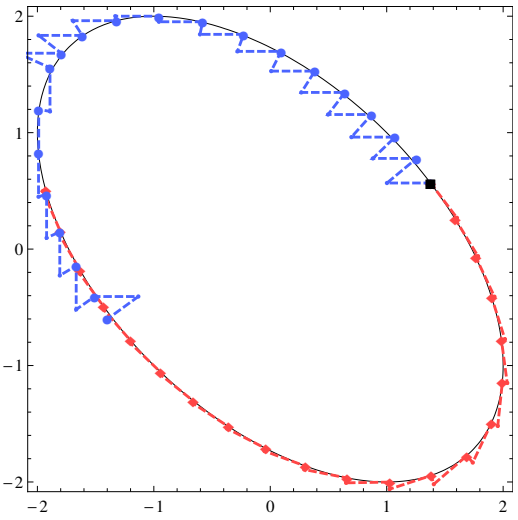
Theorem 2.3.1 (Implicit Function Theorem)**.** *Consider a system of equation as* (2.16) *with* $F : \mathbb{R}^{q \times m} \to \mathbb{R}^q$*. Let* x *be a solution to the system, i.e.* $F(x) = 0$*, and suppose that* F *is continuously differentiable around* x*. If* $F'(x)$ *is full row rank, then there exist* q *linearly independent column vectors in* $F'(x)$*. Without loss of generality, we suppose they are the first* q *column vectors. Consider the decomposition* $x = (x', x'')$ *where* $x' \in \mathbb{R}^q$ *and* $x'' \in \mathbb{R}^m$*. Then, there exist open neighborhoods* $\mathcal{U} \subseteq \mathbb{R}^q$ *and* $\mathcal{V} \subseteq \mathbb{R}^m$*, with* $x' \in \mathcal{U}$ *and* $x'' \in \mathcal{V}$*, and a unique function* $\gamma : \mathcal{U} \to \mathcal{V}$ *such that* $F(\gamma(x''), x'') = 0$*, for all* $x'' \in \mathcal{V}$*.*

In other words, the solutions to the underconstrained system (2.16) can be parameterized by functions of m variables around nonsingular solutions (see Definition 2.3.2 below).

There are two main categories of continuation methods : Piecewise-Linear (PL) continuation, and Predictor-Corrector (PC) continuation. In PL continuation, the manifold is approximated by means of simplicial decomposition of the search space. Starting from the initial solution, the method determines an initial simplex and solutions at the intersection of the simplex and the manifold. From these solutions, neighboring intersecting simplices are determined and the process is repeated. In PC continuation, two steps are performed successively. Starting from a solution, the predictor step builds a point along an approximate tangent direction to the manifold, at a given step length. The corrector step applies Newton iterations to correct the predicted point back to the manifold. The process is repeated from this new corrected solution. We can note that PC continuations are most widely used as they are better suited for a high embedding dimension. Some PC continuation methods select a priori m variables as parameters. While fixing parameters has a sense in some applications, and eases some computations of the continuation (e.g. the tangent computa-



(a) PC continuation with fixed parameter : x_1 in blue dots, x_2 in red diamond. Turning points are encountered.



(b) PC continuation with adaptive parameterization : parameter switching in blue dots, arc-length in red diamond.

Figure 2.2 – Solving $x_1^2 + x_2^2 + x_1x_2 - 3 = 0$ with PC continuation after the computation of an initial solution as in Figure 2.1(b). Dashed broken lines show the path taken by predicted and corrected points.

tion), it may raise some difficulties like turning points. This is depicted on Figure 2.2(a). To avoid turning points, parameter switching can be performed, i.e. adaptively selecting variables as parameters, or arclength continuation which uses the implicit arclength parameterization of the manifold yielding even sharper adaptations. Those two methods are illustrated on Figure 2.2(b). Among applications of continuation methods are nonlinear eigenvalue problem [9], path-planning in robotics [97], homotopy techniques for polynomial root finding [5] or nonlinear optimization [46, 47] ; and parametric optimization [99, 101]. We can also note that while many use of continuation from the literature concern one-dimensional manifold, i.e. $m = 1$, there exist several continuation methods in order to deal with higher dimensional manifold via linear/polyhedral approximations, see e.g. [12, 52, 11]. As the manifold of solutions can be made of several disjoint parts, it is important to ensure the continuity between two solutions produced by continuation, such that the continuation tracks a single connected part of the solution set. Asserting the continuity can be done by selecting an appropriate step length for continuation, see e.g. [29, 5].

Finally, when the system is overconstrained, i.e. $q > n$, there is in general no solution to (2.16). If any, solutions in that case are the cause of (locally) redundant equations, hence numerically instable [3]. In general, singular solutions of the system (2.16) cannot be attained via Newton methods. Singularity is defined as follows :

Definition 2.3.2 (Singularity in system of equations)**.** Let a system of equation $F(x) = 0$ with $F : \mathbb{R}^n \to \mathbb{R}^q$. This system is singular at a solution x if $F'(x) = (\nabla F(x))^T$ is not full row rank.

3. Changing slightly one of the equations causes the loss of the solution

When approaching a singular solution, the derivative of the system becomes more and more ill conditioned yielding numerical errors and instability. Moreover, we see that any overconstrained system is singular.

Interval analysis can be used to solve (2.16) when solutions are nonsingular. Precisely, interval Newton operators [90] can be used on a box $\boldsymbol{x}$ in order to iteratively build a tight enclosure of a solution to (2.16). Two usages of Newton operators can be considered : contract a box $\boldsymbol{x}$ towards a solution to (2.16) and/or perform an interval search for a solution to (2.16). A very interesting feature of interval Newton operators is that they can build certificates, i.e. numeric proofs of existence and uniqueness of solutions to (2.16) within a box $\boldsymbol{x}$. This is of critical importance for solving safely systems of equations. In the next two subsections, interval Newton operators are presented for respectively square and underconstrained systems.

2.3.1.2 Interval methods for square systems of equations

Suppose here a system of equations (2.16) with $q = n$. Given a box $\boldsymbol{x}_k$ and $\tilde{x}_k \in \boldsymbol{x}_k$ (usually $\tilde{x}_k = \text{mid}(\boldsymbol{x}_k)$), the interval Newton operator consists in enclosing the solutions to the interval linearization $\boldsymbol{F}(\tilde{x}_k) + \boldsymbol{F}'(\boldsymbol{x}_k)(x_{k+1} - \tilde{x}_k) \ni 0$ of (2.17), giving rise to a new box $\boldsymbol{x}_{k+1}$. Precisely, we denote by $\mathbf{N}(F, \boldsymbol{x})$ an interval Newton operator, and define the sequence of boxes $\boldsymbol{x}_k$ by

$$\boldsymbol{x}_{k+1} = \mathbf{N}(F, \boldsymbol{x}_k). \tag{2.18}$$

The different iterates $\boldsymbol{x}_k$ are shifted and inflated or contracted towards one solution to (2.16). Interval Newton is usually performed via the Krawczyk or Hansen-Sengupta operators. The Krawczyk operator is defined as follows :

$$\mathbf{K}(F, \boldsymbol{x}) = \tilde{x} - \boldsymbol{F}(\tilde{x}) - (\boldsymbol{F}'(\boldsymbol{x}) - I)(\boldsymbol{x} - \tilde{x}). \tag{2.19}$$

This operator is somehow an interval version of the Jacobi method. The Hansen-Sengupta operator is defined by :

$$\mathbf{H}(F, \boldsymbol{x}) = \tilde{x} + \boldsymbol{\Gamma}(\boldsymbol{F}'(\boldsymbol{x}), \boldsymbol{x} - \tilde{x}, -\boldsymbol{F}(\tilde{x})) \tag{2.20}$$

$$\text{with } \boldsymbol{\Gamma}_i(\boldsymbol{A}, \boldsymbol{x}, \boldsymbol{b}) = \frac{1}{\boldsymbol{a}_{ii}}\Big(\boldsymbol{b}_i - \sum_{j \neq i} \boldsymbol{a}_{ij}\boldsymbol{x}_j\Big), \tag{2.21}$$

where $\boldsymbol{\Gamma}$ denotes the interval Gauss-Seidel operator for linear systems. The Hansen-Sengupta operator is hence a nonlinear version of Gauss-Seidel. It is known that that the Hansen-Sengupta operator performs better than Krawczyk [90]. Nevertheless, the Krawczyk operator can be efficiently implemented by fast interval matrix computations (like e.g. INTLAB [112]). In the following, the interval Newton $\mathbf{N}(F, \boldsymbol{x})$ will denote either the Krawczyk or Hansen-Sengupta operator. It is in general required to precondition the system in order to accelerate the convergence of the interval Newton [90]. Usually, left preconditioning is used. The system (2.16) is multiplied on its left by an adequate matrix P. In general a good choice is $P = (\text{mid}(\boldsymbol{F}'(\boldsymbol{x}_k)))^{-1}$. This matrix is called inverse midpoint preconditioning matrix.

These interval Newton operators have the following key properties [90].

Theorem 2.3.2. *Consider a square system as (2.16) and let a sequence $\boldsymbol{x}_k$ of boxes defined by (2.18). Then :*

$$\forall x \in \boldsymbol{x}_k, F(x) = 0 \Rightarrow x \in \boldsymbol{x}_{k+1} \tag{2.22}$$

This property ensures that the interval Newton cannot lose any solution to the system once it is contained in a box $\boldsymbol{x}_k$. The next property states the existence and uniqueness of fixpoint for the Newton iteration, i.e. the existence and uniqueness of solutions within a box $\boldsymbol{x}$. The Brouwer fixpoint theorem is used to build this property.

Theorem 2.3.3 (Brouwer fixpoint theorem)**.** *Consider a continuous function $\phi : \mathcal{U} \to \mathcal{U}$ where $\mathcal{U}$ is a closed subset of $\mathbb{R}^n$ homeomorphic to the unit closed ball in $\mathbb{R}^n$. Then, ϕ has a fixpoint $x \in \mathbb{R}^n$, i.e. $\phi(x) = x$.*

If ϕ is the function corresponding to the Newton iteration, then if ϕ is continuous and maps a closed set $\mathcal{U}$ onto itself, then $\mathcal{U}$ contains a solution to the corresponding system of equation. Another important existence theorem is the Miranda's theorem.

Theorem 2.3.4 (Miranda's theorem)**.** *Consider a vector of continuous function $F = (h_1, \cdots, h_n)$ defined on a box $\boldsymbol{x} \subseteq \mathbb{R}^n$. If*

$$\max\{xy : x \in h_i(\boldsymbol{x}_i^-), y \in h_i(\boldsymbol{x}_i^+)\} \leq 0 \tag{2.23}$$

for each i then there is an $x \in \boldsymbol{x}$ such that $F(x) = 0$.

The Miranda's theorem extends the intermediate value theorem to vector functions and is known to be equivalent to the Brouwer theorem. The existence and uniqueness tests of the interval Newton can now be stated :

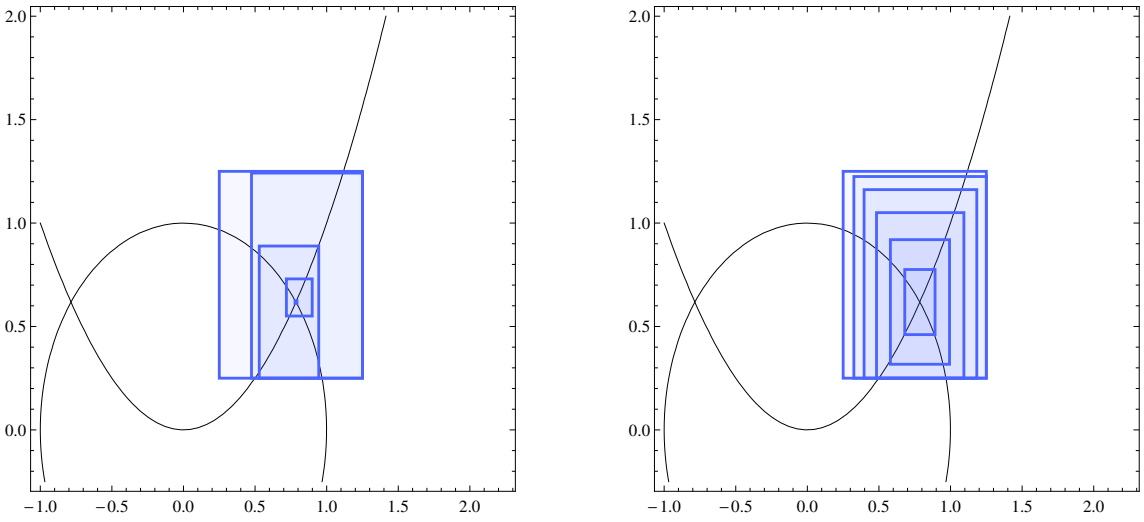
Theorem 2.3.5 (Interval Newton existence and uniqueness test)**.** *Consider a square system as (2.16) and let a sequence $\boldsymbol{x}_k$ of boxes defined by (2.18). Then :*

$$\emptyset \neq \boldsymbol{x}_{k+1} \subseteq \text{int}(\boldsymbol{x}_k) \Rightarrow \begin{cases} \boldsymbol{F}'(\boldsymbol{x}_k) \text{ is regular (full rank)} \\ \exists! x \in \boldsymbol{x}_k, F(x) = 0 \end{cases} \tag{2.24}$$

The existence of the solution is due to the Brouwer fixpoint theorem for the Krawczyk operator, applicable as the Krawczyk is the centered form extension of the function $x - F(x)$ that is proved to map $\boldsymbol{x}_k$ onto itself ; and the Miranda's theorem for the Hansen-Sengupta operator (see e.g. Theorem 1.17 and 1.18 from [61]). Regularity of $\boldsymbol{F}'(\boldsymbol{x})$, implying uniqueness of the solution, is due to for example Lemma 1.20 and 1.21 from [61]. Note that the regularity of $\boldsymbol{F}'(\boldsymbol{x}_k)$ implies that $F'(x)$ is full rank for all $x \in \boldsymbol{x}_k$.

This latter property of the interval Newton operator allows it to produce certificates : proofs that a unique solution to the system is captured by a box. It is detected by a strict contraction of a box by the interval Newton, meaning that all solutions within $\boldsymbol{x}_k$ are convergent with respect to the Newton operator. This proof asserts that the interval Newton is converging to an unique solution to the equations within $\boldsymbol{x}_k$. Hence, considering a general NCSP those equality constraints form the system (2.16), and assuming $\boldsymbol{x}_k$ is additionally certainly satisfying all inequality constraints, then $\boldsymbol{x}_k$ is called a safe box, and can be considered as terminal for B&P.

Remark 2.3.1 – We have here used natural extensions of $\boldsymbol{F}$ and $\boldsymbol{F}'$ to defined the different interval Newton operators. We can note that the evaluations $\boldsymbol{F}(\tilde{x}_k)$ and $\boldsymbol{F}'(\boldsymbol{x}_k)$ can be equivalently replaced by respectively $\boldsymbol{b}$ and $\boldsymbol{A}$ provided $F(\tilde{x}_k) \in \boldsymbol{b}$ and $F'(x_k) \in \boldsymbol{A}$ for all $x_k \in \boldsymbol{x}_k$. This allows using possibly sharper interval extensions that may enhance the Newton operator. Theorem 2.3.2 and 2.3.5 still hold in that case.



(a) Hansen-Sengupta contracting operator

(b) Krawczyk contracting operator

Figure 2.3 – Newton contractor applied to a square system of equations

If the interval Newton is used to build proof of existence and uniqueness of solutions within an initial box $\boldsymbol{x}_0$, it must be asserted that the different iterates of the interval Newton remains inside $\boldsymbol{x}$. This is done by intersecting each box $\boldsymbol{x}_{k+1}$ with its predecessor $\boldsymbol{x}_k$, i.e the Newton iteration becomes $\boldsymbol{x}_{k+1} = \mathbf{N}(F, \boldsymbol{x}_k) \cap \boldsymbol{x}_k$.

Example 2.3.1 – Consider the square system of equations

$$F(x) = \begin{pmatrix} x_1^2 + x_2^2 - 1 \\ x_1^2 - x_2 \end{pmatrix} = 0, \tag{2.25}$$

and consider the initial box $\boldsymbol{x}_0 = ([0.25, 1.25], [0.25, 1.25])$ and its midpoint $\tilde{x}_0 = \text{mid}(\boldsymbol{x}_0) = (0.75, 0.75)$. We have :

$$\boldsymbol{F}(\tilde{x}_0) = \begin{pmatrix} 0.125 \\ -0.1875 \end{pmatrix}, \ \boldsymbol{F'}(\boldsymbol{x}_0) = \begin{pmatrix} [0.5, 2.5] & [0.5, 2.5] \\ [0.5, 2.5] & [-1, -1] \end{pmatrix}, \ P = \begin{pmatrix} 0.26667 & 0.4 \\ 0.4 & -0.4 \end{pmatrix}.$$

The interval Newton operators give $\mathbf{H}(P \cdot F, \boldsymbol{x}_0) \cap \boldsymbol{x}_0 = ([0.475, 1.25], [0.25, 1.24167])$, and $\mathbf{K}(P \cdot F, \boldsymbol{x}_0) \cap \boldsymbol{x}_0 = ([0.325, 1.25], [0.25, 1.225])$. The enclosure produced by the Hansen-Sengupta operator is sharper than that of the Krawczyk. Five iterations of both contracting operators are shown on Figure 2.3. The certificate (2.24) is observed at $k = 2$ (i.e. $\boldsymbol{x}_3 \subset \text{int}(\boldsymbol{x}_2)$) for the Hansen-Sengupta operator and at $k = 4$ for the Krawczyk operator.

If one wants to search with certificate for a solution to the system given any starting box $\boldsymbol{x}$, then it is important to interleave Newton iteration and inflation of box iterates in order to fasten the observation of strict contraction (e.g. when the initial box is very small). An inflated box $\boldsymbol{x}'_k$ for $\boldsymbol{x}_k$ is typically obtained by $\boldsymbol{x}'_k = \delta(\boldsymbol{x}_k - \tilde{\boldsymbol{x}_k}) + \tilde{\boldsymbol{x}_k} + \chi[-1, 1]$, with $\delta > 1$ and $\chi > 0$ [111]. This process is called ϵ-inflation. Then, the Newton iteration is replaced by $\boldsymbol{x}_{k+1} = \mathbf{N}(F, \boldsymbol{x}'_k)$, while strict contraction must be observed within $\boldsymbol{x}'_k$. The values of δ and χ perturb the convergence of the interval Newton. Hence, they must be fixed to a reasonable value, e.g. $\delta = 1.1$ and χ very small (depending on the machine precision).

Finally, note that the interval Newton can theoretically converge to a solution x with infinite precision. In practice, we are faced with rounding errors (handled properly by an interval arithmetic library). Hence, the interval Newton converges to x up to the machine precision. In general, we stop the interval Newton operator after a finite number of iterations, once the convergence rate is not significant (in case of slow convergence or divergence), or when a prescribed precision is reached.

2.3.1.3 Interval methods for underconstrained system of equations

Suppose here the system of equations (2.16) with $q < n$. Solutions to such systems form, under regularity assumptions, manifolds of dimension $m = n - q$. Hence, in order to deal with underconstrained systems with intervals, methods have to produce enclosures of manifolds of solutions.

Box domains The interval Newton operators can be easily extended to square parametric system of equations $F(z, a) = 0$, i.e. $F : \mathbb{R}^q \times \mathbb{R}^m \to \mathbb{R}^q$. Transforming an underconstrained system $F(x)$ into a parametric one $F(z, a)$, with $x = (z, a)$, requires fixing m variables in x as parameters. For convenience we consider that the parameters a are the last m components of x, up to a variable permutation. Here, free variables are denoted by z and parameters by a. The parameters have fixed domains, such that only free variables are affected by the Newton operator. A parametric interval Newton operator will be denoted $\mathbf{N}(F, \boldsymbol{z}, \boldsymbol{a})$. It is used to define the following sequence of boxes

$$(\boldsymbol{z}_{k+1}, \boldsymbol{a}) = (\mathbf{N}(F, \boldsymbol{z}_k, \boldsymbol{a}), \boldsymbol{a}). \tag{2.26}$$

Precisely, parametric interval Newton operator can be defined as the parametric Krawczyk by :

$$\mathbf{K}(F, \boldsymbol{z}, \boldsymbol{a}) = \text{mid}(\boldsymbol{z}) - \boldsymbol{F}(\tilde{z}, \boldsymbol{a}) - (\boldsymbol{F}'_z(\boldsymbol{z}, \boldsymbol{a}) - I)(\boldsymbol{z} - \text{mid}(\boldsymbol{z})). \tag{2.27}$$

Analogously, the parametric Hansen-Sengupta operator is defined by :

$$\mathbf{H}(F, \boldsymbol{z}, \boldsymbol{a}) = \tilde{z} - \boldsymbol{\Gamma}(\boldsymbol{F}'_z(\boldsymbol{z}, \boldsymbol{a}), \boldsymbol{z} - \tilde{z}, -\boldsymbol{F}(\tilde{z}, \boldsymbol{a})). \tag{2.28}$$

These operators differ from the case of square systems only in the evaluation of $\boldsymbol{F}$. Here, the interval extension of F is evaluated on the extension point $(\tilde{z}_k, \boldsymbol{a})$ where $\tilde{z}_k \in \boldsymbol{z}_k$, i.e. all parameter values are considered. In other words, it solves the linearization (2.17) over all parameter values within $\boldsymbol{a}$ [4]. Analogously to the case of square systems, left preconditioning with respect to the free variables z enhances the convergence of the parametric interval Newton.

The properties of the parametric Newton operators are similar to the ones in the case of square systems, except that they are valid for all parameter values within $\boldsymbol{a}$ [36, 39]

Theorem 2.3.6. *Consider an underconstrained system as* (2.16) *with* $m < n$. *Let a sequence* $(\boldsymbol{z}_k, \boldsymbol{a})$ *of boxes defined by* (2.26). *Then :*

$$\forall (z, a) \in \boldsymbol{z}_k \times \boldsymbol{a}, F(z, a) = 0 \Rightarrow z \in \boldsymbol{z}_{k+1} \tag{2.29}$$

This proposition is a direct consequence of the interval Newton on square systems. Certificates can also be constructed in this parametric context.

4. Natural or centered form evaluation with respect to the parameters can be used

Theorem 2.3.7 (Parametric interval Newton existence and uniqueness test)**.** *Consider an underconstrained system as* (2.16) *with* $q < n$*. Let a sequence* $(\boldsymbol{z}_k, \boldsymbol{a})$ *of boxes defined by* (2.26)*. Then :*

$$\emptyset \neq \boldsymbol{z}_{k+1} = \mathbf{N}(F, \boldsymbol{z}_k, \boldsymbol{a}) \subseteq \text{int}(\boldsymbol{z}_k) \Rightarrow \begin{cases} \boldsymbol{F'}_z(\boldsymbol{z}, \boldsymbol{a}) \text{ is regular (full rank)} \\ \forall a \in \boldsymbol{a}, \exists! z \in \boldsymbol{z}_k \text{ with } F(z, a) = 0 \end{cases} \tag{2.30}$$

Moreover, there exists a unique continuous and differentiable function $\gamma_k(a) : \boldsymbol{a} \to \mathbb{R}^m$*, verifying* $F(\gamma_k(a), a) = 0, \forall a \in \boldsymbol{a}$*.*

The existence of a unique continuously differentiable curve γ is a consequence of the implicit function theorem and the regularity of F'_z inside the box $(\boldsymbol{z}_k, \boldsymbol{a})$. This implies that certificates in the parametric case ensure the existence of a continuous and regular parameterization, with respect to the parameters a, of the manifold of solutions to the system within $(\boldsymbol{z}_k, \boldsymbol{a})$. Hence, certificates prove the existence and continuity of a part of the manifold of solutions.

Remark 2.3.2 – As in the case of square systems, we have used natural extensions of $\boldsymbol{F}$ and $\boldsymbol{F'}$ to define the different parametric interval Newton operators. The evaluations $\boldsymbol{F}(\tilde{z}_k, \boldsymbol{a})$ and $\boldsymbol{F'}(\boldsymbol{z}_k, \boldsymbol{a})$ can be equivalently replaced by respectively $\boldsymbol{b}$ and $\boldsymbol{A}$ provided $F(\tilde{z}_k, a) \in \boldsymbol{b}$ for all $a \in \boldsymbol{a}$ and $F'(z_k, a) \in \boldsymbol{A}$ for all $z_k \in \boldsymbol{z}_k$ and $a \in \boldsymbol{a}$. Theorem 2.3.6 and 2.3.7 still hold in that case.

As for the case of square systems, each box $\boldsymbol{z}_{k+1}$ must be intersected with its predecessor $\boldsymbol{z}_k$ in order to prove existence and uniqueness of solutions within an initial box $\boldsymbol{x}_0$.

Example 2.3.2 – Consider the underconstrained system

$$F(x) = (x_1^2 + x_2^2 + x_1 x_2 - 3) = 0, \tag{2.31}$$

whose solution set is an ellipse. The variable $a = x_2$ is set as parameter, $z = x_1$ is let free. Thus, consider the initial box $\boldsymbol{x}_0 = (\boldsymbol{z}_0, \boldsymbol{a}) = ([0.5, 2.25], [0.25, 0.75])$, and midpoint $\tilde{z}_0 = \text{mid}(\boldsymbol{z}_0) = 0.875$. We have :

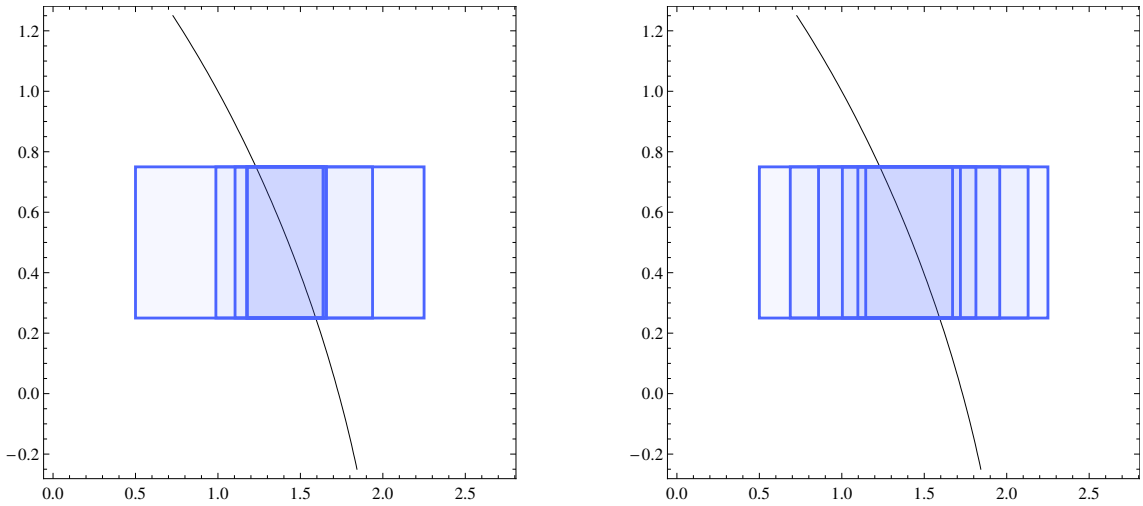
$$\boldsymbol{F}(\tilde{z}_0, \boldsymbol{a}) = \left([-1.953125, -1.015625]\right), \ \boldsymbol{F'}_z(\boldsymbol{z}_0, \boldsymbol{a}) = \left([1.25, 5.25]\right), \ P = \left(0.307692\right).$$

The Newton operators give $\mathbf{H}(P \cdot F, \boldsymbol{z}_0, \boldsymbol{a}) \cap \boldsymbol{z}_0 = ([0.9875, 1.9375])$, and $\mathbf{K}(P \cdot F, \boldsymbol{z}_0, \boldsymbol{a}) \cap \boldsymbol{z}_0 = ([0.6875, 2.12981])$. Again, the enclosure produced by the Hansen-Sengupta operator is sharper than that of the Krawczyk. However certificates are obtained after the first application of both Newton operators. Five iterations of both contracting operators are shown on Figure 2.4.

Similarly, principle of ϵ-inflation can also be extended to underconstrained system. Only the component z needs to be inflated for the purpose of building a certificate from any initial box.

Transforming an underconstrained system into a parametric one requires selecting appropriately which variables are turned into parameters. Ideally, in order to ease and ensure the convergence of the parametric Newton, the projection of the manifold onto the parameters must be unique and complete, i.e. for each parameter values a in $\boldsymbol{a}$ there exists a unique solution (z, a). This is usually not known beforehand, therefore the parameters have to be selected heuristically. For example, one can use the derivative of F over all the variables x in $\boldsymbol{x}$ and select via an interval Gauss elimination the free variables z, the others being set as parameters. Another heuristic is to compute the tangent space of $F'(\tilde{x})$, and to set as parameters the variables having the biggest influence in this tangent space [58].

Another strategy is to build an auxiliary space in which the manifold of solutions is locally transversal to the parametric subspace. This idea of expressing an auxiliary space has lead to the definition of parallelotopes [39]. The construction of such auxiliary space can be seen as a form of right preconditioning of the system.



(a) Hansen-Sengupta contracting operator

(b) Krawczyk contracting operator

Figure 2.4 – Newton contractor applied to an underconstrained system of equations

Parallelotope domains A parallelotope domain is, roughly speaking, an affine transformation of a box domain, oriented specifically. The aim of the orientation is to adapt the parallelotope to the shape of the solution set. Here, as we consider solutions to underconstrained systems of equations, such linear transformation of boxes can be seen as a right-preconditioning of the system. We will use the following definition of a parallelotope.

Definition 2.3.3 (Parallelotope)**.** Given a matrix $C \in \mathbb{R}^{n\times n}$, a point $\tilde{x} \in \mathbb{R}^n$ and a box $\boldsymbol{w} \in \mathbb{IR}^n$, a parallelotope $\widehat{\boldsymbol{x}}$ is the image of a box $\boldsymbol{w}$ through the affine map $w \to Cw + \tilde{x}$, i.e.

$$\widehat{\boldsymbol{x}} = \{Cw + \tilde{x} \in \mathbb{R}^n : w \in \boldsymbol{w}\} \tag{2.32}$$

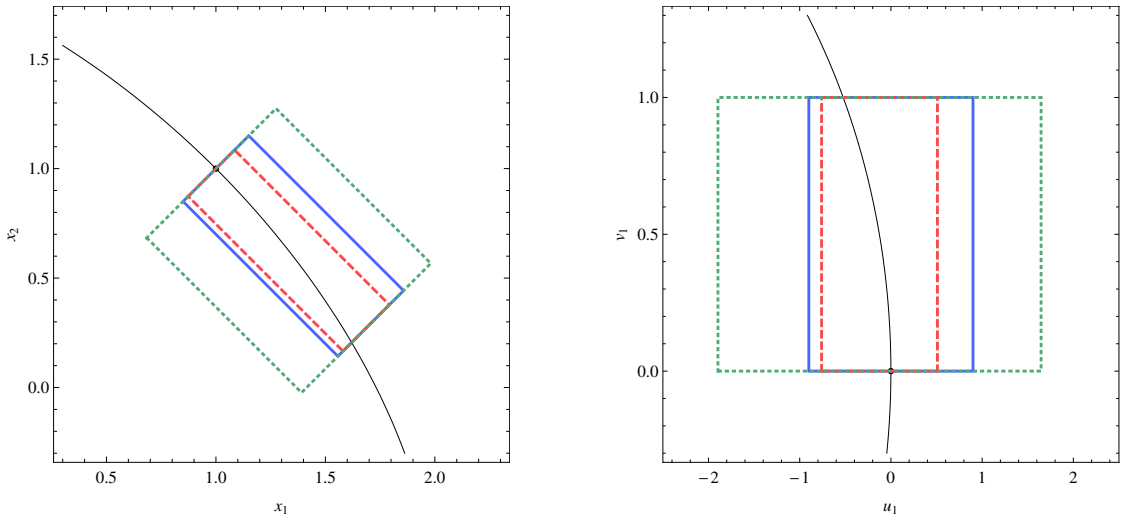
A parallelotope $\widehat{\boldsymbol{x}}$ is then defined by a triplet $(C, \boldsymbol{w}, \tilde{x})$, where C, $\boldsymbol{w}$ and $\tilde{x}$ are respectively called the characteristic matrix, box and vector of $\widehat{\boldsymbol{x}}$.

The interval hull $\square\widehat{\boldsymbol{x}}$ of the parallelotope $\widehat{\boldsymbol{x}} = (C, \boldsymbol{w}, \tilde{x})$ is computed as $C\boldsymbol{w}+\tilde{x}$. The midpoint of a parallelotope is $\operatorname{mid}(\widehat{\boldsymbol{x}}) := C \operatorname{mid}(\boldsymbol{w}) + \tilde{x}$.

Parallelotopes are used in [39] in conjunction with interval analysis in order to enclose and certify m-manifolds defined by a system $F(x) = 0$ of q equations and n unknowns with $m = n-q > 0$. The cornerstone of this approach is to apply usual interval analysis techniques dedicated to boxes within the auxiliary space of the parallelotope, where the original system of equations becomes $G(w) = 0$ with $G(w) = F(Cw + \tilde{x})$, and whose derivative is $G'(w) = F'(Cw + \tilde{x})C$. Inside this auxiliary basis, the last m components of w are identified to parameters (w is split into $w = (u, v)$ with $u \in \mathbb{R}^q$ and $v \in \mathbb{R}^m$), and the aim is to build parallelotopes $\widehat{\boldsymbol{x}} = (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$ that contain a solution for each parameter value in $\boldsymbol{v}$, i.e. satisfying

$$\forall v \in \boldsymbol{v},\ \exists u \in \boldsymbol{u},\ G(u, v) = 0. \tag{2.33}$$

This property expresses that the manifold crosses the whole parallelotope along the parameter subspace. In order to certify Property (2.33), [39] applies a parametric interval Newton operator



(a) Parallelotope enclosure : Original basis

(b) Parallelotope enclosure : Auxiliary basis

Figure 2.5 – Application of the interval Krawczyk operator for parallelotopes in Example 2.3.3

to the auxiliary system $G(u, v) = 0$: The interval Newton for parallelotopes takes as input a system of equations and a parallelotope, and outputs a new domain for the non parametric auxiliary variables u. It can be instantiated using the Krawczyk or the Hansen-Sengupta interval operators :

$$\mathbf{K}(F, \widehat{\boldsymbol{x}}) := \mathrm{mid}(\boldsymbol{u}) - \boldsymbol{b} - (\boldsymbol{A}_u - I)(\boldsymbol{u} - \mathrm{mid}(\boldsymbol{u})) \tag{2.34}$$

$$\mathbf{H}(F, \widehat{\boldsymbol{x}}) := \mathrm{mid}(\boldsymbol{u}) + \Gamma(\boldsymbol{A}_u, \boldsymbol{u} - \mathrm{mid}(\boldsymbol{u}), -\boldsymbol{b}), \tag{2.35}$$

where $\widehat{\boldsymbol{x}} = (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$, $\boldsymbol{A}_u \supseteq \{G'_u(w) : w \in \boldsymbol{w}\}$, $\boldsymbol{A}_v \supseteq \{G'_v(\mathrm{mid}(\boldsymbol{u}), v) : v \in \boldsymbol{v}\}$ (note that $\boldsymbol{A}_u \in \mathbb{IR}^{m\times m}$ and $\boldsymbol{A}_v \in \mathbb{IR}^{m\times(n-m)}$) and $\boldsymbol{b} = \boldsymbol{G}(\mathrm{mid}(\boldsymbol{w})) + \boldsymbol{A}_v(\boldsymbol{v} - \mathrm{mid}(\boldsymbol{v}))$. In the following, $\mathbf{N}(F, \widehat{\boldsymbol{x}})$ will denote either of these interval operators. Since $G(\mathrm{mid}(\boldsymbol{u}), v) \in \boldsymbol{b}$ holds for all $v \in \boldsymbol{v}$, the properties of the parametric interval Newton hold for the auxiliary system $G(w) = 0$. In particular, Theorems 2.3.6 and 2.3.7 hold inside the auxiliary space induced by a parallelotope. Every solution contained in the initial parallelotope has to belong to the computed parallelotope $(C, (\mathbf{N}(F, \widehat{\boldsymbol{x}}), \boldsymbol{v}), \tilde{x})$, and $\mathbf{N}(F, \widehat{\boldsymbol{x}}) \subseteq \boldsymbol{u}$ implies (2.33), with uniqueness in addition to the existence statement.

The partial derivatives enclosures $\boldsymbol{A}_u$ and $\boldsymbol{A}_v$ can be computed in several ways. The most obvious is to evaluate the derivatives of F over the interval hull of the parallelotopes : $\boldsymbol{A}_u = \boldsymbol{F}'(C\boldsymbol{w} + \tilde{x})C_u$ and $\boldsymbol{A}_v = \boldsymbol{F}'(C(\mathrm{mid}(\boldsymbol{u}), \boldsymbol{v}) + \tilde{x})C_v$, where $C = (C_u \,|\, C_v)$. However, this does not take full benefit of parallelotopes. When formal simplifications can be performed, e.g. for polynomial systems, the expression $G(w) = F(Cw + \tilde{x})$ can be formally simplified before performing an automatic differentiation to obtain $\boldsymbol{A}_u$ and $\boldsymbol{A}_v$. When F is twice differentiable, a third option is to use a mean-value, or centered form evaluation of the derivatives of G. This sensibly improves the evaluation over the hull of the parallelotope, but requires evaluating second order derivatives which can turn out to be too expensive for large systems.

Example 2.3.3 – Consider the underconstrained system (2.31), whose solution set is an ellipse, and the parallelotope $(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$ with $\tilde{x} = (1., 1.)$, $\boldsymbol{u} = [-0.9, 0.9]$, $\boldsymbol{v} = [0, 1]$ and

$$C = \begin{pmatrix} 0.16666 & 0.70710 \\ 0.16666 & -0.70710 \end{pmatrix}. \tag{2.36}$$

The solution set and the parallelotope are depicted in full line in Figure 2.5(a). The system expressed in the auxiliary parallelotope basis is $G(w) = 0$ with $G(w) = F(Cw + \tilde{x})$, where $w = (u, v) \in \mathbb{R}^2$. The auxiliary system solution set, which is now approximately aligned with the vertical axis corresponding to the parameter v, and the box $(\boldsymbol{u}, \boldsymbol{v})$ are depicted in full line in Figure 2.5(b).

The explicit expression of $G(w)$ is

$$(C_{11}u+C_{12}v+\tilde{x}_1)^2+(C_{21}u+C_{22}v+\tilde{x}_2)^2+(C_{11}u+C_{12}v+\tilde{x}_1)(C_{21}u+C_{22}v+\tilde{x}_2)-3. \tag{2.37}$$

Using automatic differentiation with this expression gives rise to the following enclosure of $G'(w) : (\boldsymbol{A}_u, \boldsymbol{A}_v) = ([0.496, 1.504], [-1.637, 2.637])$, which is equivalent to evaluating $F'(x)$ on the interval hull of the parallelotope using automatic differentiation. The Krawczyk operator used with this expression returns $\boldsymbol{u}' = [-1.897, 1.647]$, which is not included inside $\boldsymbol{u}$ hence failing to certify the curve within the parallelotope. The parallelotope $(C, (\boldsymbol{u}', \boldsymbol{v}), \tilde{x})$ is depicted in dotted green line in Figure 2.5.

Expanding (2.37), collecting together the occurrences of u and v, and differentiating with respect to w gives rise to the new expression

$$([1^-, 1^+] + [0.166^-, 0.166^+]u + [0^-, 0^+]v, [0^-, 0^+] + [0^-, 0^+]u + [1^-, 1^+]v), \tag{2.38}$$

where $[a^-, a^+]$ denotes the interval enclosing a obtained using rounded interval arithmetic. Using automatic differentiation with this expression gives rise to the following enclosure of $G'(w) : (\boldsymbol{A}_u, \boldsymbol{A}_v) = ([0.849, 1.151], [-0.001, 1.001])$, which is much sharper than the evaluation over the interval hull of the parallelotope. The Krawczyk operator used with this expression returns $\boldsymbol{u}'' = [-0.761, 0.511]$, which is now included inside $\boldsymbol{u}$ hence certifying that a unique curve crosses the parallelotope. The parallelotope $(C, (\boldsymbol{u}'', \boldsymbol{v}), \tilde{x})$ is depicted in dashed red line in Figure 2.5. This system being quadratic, its derivative is linear and the mean-value extension of the derivatives yields the same enclosure as the formal simplification.

In order to contract, the interval Newton for parallelotopes requires that $\boldsymbol{A}_u$ is close to the identity and that $\boldsymbol{A}_v$ is close to zero. To this end, the characteristic matrix C proposed in [39] is chosen so that its first q columns are an approximation of a generalized inverse of $F'(\tilde{x})$ and the remaining $n - q$ columns approximate the kernel of $F'(\tilde{x})$, where $\tilde{x}$ is close to the center of the parallelotope. Provided the parallelotope is small enough so that the manifold remains relatively close to its tangent, this characteristic matrix fixes the orientation of the parallelotope similarly to the one of the manifold. Therefore, the characteristic matrix acts like a right-preconditioning of the system.

Parallelotopes are used in [39] to build proofs of existence and uniqueness of solutions within boxes produced by B&P, and to perform some pruning. The idea is, given a box $\boldsymbol{x}$ to build an enclosing parallelotope. The parallelotope is then contracted to the solutions inside the box while linear constraints corresponding to the initial box domains are also considered. Contractors (see Section 2.3.2) are applied on these additional constraints to reduce the size of the parallelotope and enhance the Newton iterations.

Interval Continuation Given an initial solution or certified box solving the underconstrained system (2.16), continuation can be performed with certification by locally constructing and certifying boxes along the manifold of solutions. Interval analysis is used to derive a verified step length of continuation in [29] asserting that solutions produced are connected. However, no enclosure of the solution set is constructed by this method. In addition, it is mainly devoted to quadratic systems of low dimension : while the maximal step size has a simple expression for quadratic systems, it requires heavy formal manipulations for non quadratic systems. In addition, this maximal step size decreases quadratically with the dimension of the problem, restricting it to low dimensional systems. This maximal step size is proportional to the inverse of the entrywise sum of the absolute value of the system's Hessian, which decreases quadratically with the dimension for non sparse systems leading to dramatically small step size. Eventually, overall algorithm or implementation certifying the complete curve connectivity is proposed in [29].

Kearfott and Xing [58] have proposed a continuation method that builds locally a set of boxes along a continuous component of the solution to an underconstrained system as (2.16) with $m = n - q = 1$. This method acts as an interval version of the PC continuation with parameter switching. Consecutive boxes computed on the manifold of solutions are connected by the solution contained in their joint facets on their parametric part. Each iteration of the method tries to construct a certified box on the manifold. A trial box is determined heuristically and parametric interval Newton is performed. If Theorem 2.3.7 is observed, then the method proceeds with the next iteration. Otherwise the different parameters controlling the size of the trial box or the step length of continuation are updated and the iteration of continuation restarted. The construction of the trial box can be seen as a particular "generate and try" inflation process : a box is constructed and one interval Newton iteration, that must strictly contract, is performed. The method used to build this trial box is difficult to implement and tune compared to the ϵ-inflation presented above. Boxes returned by this process are proved to enclose a connected manifold of solutions to (2.16). An example is depicted on Figure 2.6(a) where boxes are shown in plain or dashed lines if x_1, respectively x_2, is taken as parameter.

We propose in Chapter 4 a paralletope-based continuation that implements an intervalization of the arclength continuation, see Figure 2.6(b). The use of parallelotopes allows a better adaptation to the shape of the manifold than that with boxes. All details can be found in Chapter 4.

2.3.2 Contractors and constraint propagation

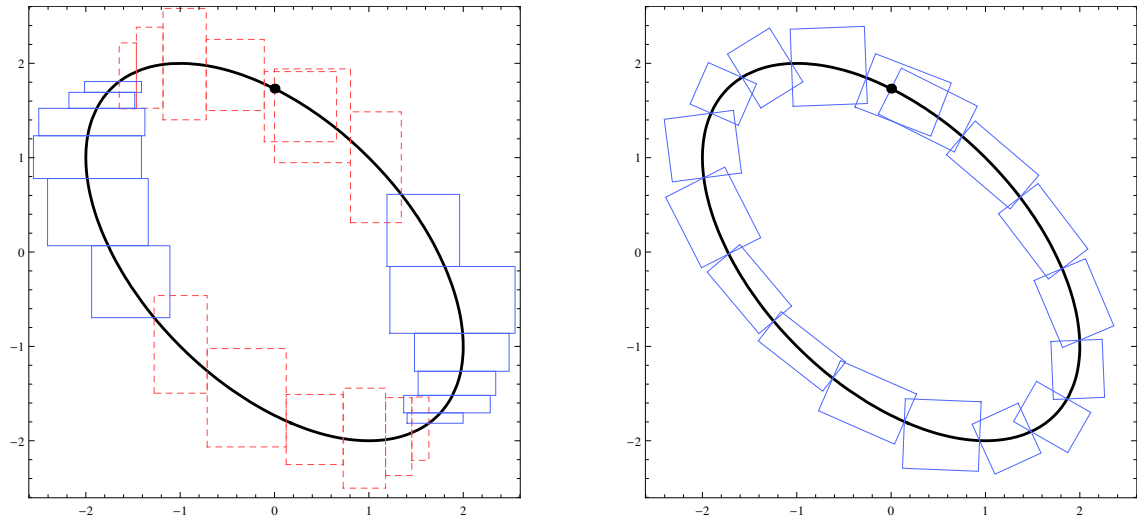
The pruning step of Algorithm 1 is typically done by applying contractors : operators that narrow a box $\boldsymbol{x}$ towards the solution to a given constraint it contains [14]. A contractor is generally based on a constraint c : a relational operation over a set of variables that is either a single constraint of the NCSP, i.e. an inequality or equality constraint (e.g. $c(x) := g_i(x) \leq 0$) or a conjunction of several constraints.

Definition 2.3.4 (Contractors)**.** Given a constraint c, a contractor $\theta : \mathbb{IR}^n \to \mathbb{IR}^n$ for c is an interval operator satisfying :

(i) $\theta(\boldsymbol{x}) \subseteq \boldsymbol{x}$

(ii) $\forall x \in \boldsymbol{x} \backslash \theta(\boldsymbol{x})$, x is not satisfying c.

In other words, a contractor is an operator that contracts a box domain $\boldsymbol{x}$, and all the discarded elements in $\boldsymbol{x}$ are not satisfying the constraint c. Note that a contractor cannot contract along variables not involved in a constraint. Note also that the pruning step we have introduced previously



(a) Using box domains (b) Using parallelotope domains

Figure 2.6 – Interval-based continuation on an underconstrained problem. The initial solution is depicted by a point.

for the B&P consists of a binary contractor : if an individual constraint is certainly not satisfied, then an empty box is produced (the box is entirely contracted) otherwise the box is unchanged.

Most contractors consider either individual constraints of the NCSP or a constraint being the conjunction of all individual constraints. The methods behind are based on principles of local consistency of a domain of variables values with respect to the considered constraint. Arc-consistency, for example, states that a domain of variables $\boldsymbol{x} \subset \mathbb{R}^n$ is arc-consistent with respect to a constraint c if for all variable x_i, it is possible to build a feasible solution by picking any values in the domain of x_i. In other words, there is a solution satisfying c for all values in the domain of $\boldsymbol{x}_i$. As we are dealing with interval domains, attaining arc-consistency is not possible in general as the set of arc-consistent values is often discontinuous. Therefore, consistency notions for interval domains, and induced contractors, have been derived [70, 7, 6, 17].

2.3.2.1 Local consistency on interval domains

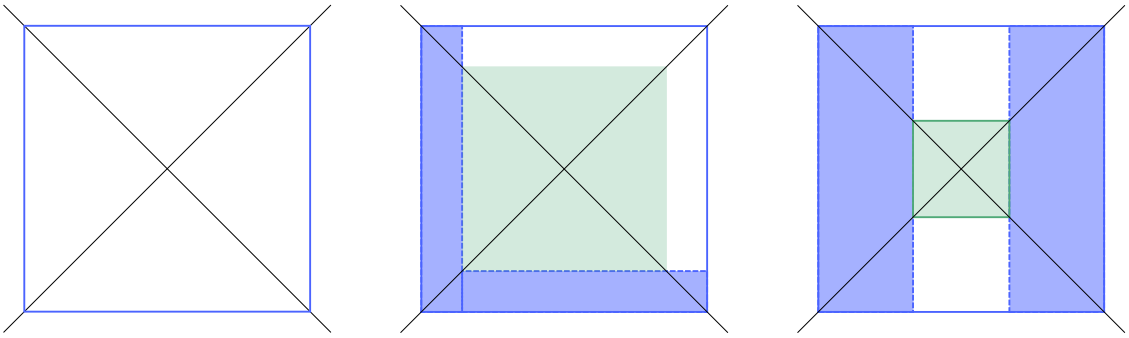
A contractor θ on a constraint c removes from a box $\boldsymbol{x}$ variable values that are inconsistent, yielding to a contracted consistent box. Different local consistency notions have been introduced for interval domains. We do not present an exhaustive list of local-consistencies for interval domains, only those from which the most used contractors are extended.

Hull-consistency (or 2B-Consistency) is version of arc-consistency which considers arc-consistency only for the bounds of a box.

Definition 2.3.5 (Hull-consistency)**.** Consider a NCSP and let c be one of its constraint over the variables $(x_1, \ldots, x_n)$. The constraint c is *hull-consistent* over a box $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ if

$$\forall i, \boldsymbol{x}_i = \square\{x_i \in \boldsymbol{x}_i : \forall j \neq i, \exists x_j \in \boldsymbol{x}_j, (x_1, \ldots, x_n) \text{ is satisfying } c\} \tag{2.39}$$

If all constraints of the NCSP are hull-consistent over $\boldsymbol{x}$, the NCSP is said hull-consistent over $\boldsymbol{x}$.



(a) The box in blue is both hull and box-consistent

(b) The box is not 3B-consistent as ϵ-facets are not hull-consistent.

(c) The box is not CID-consistent, considering a decomposition into three sub-boxes.

Figure 2.7 – Consistency of the NCSP $x_1 - x_2 = 0$ and $x_1 + x_2 = 0$.

Hull-consistency over a constraint c can be achieved by projecting the hull of the solution set of this constraint onto each variable domain. In practice, such projections are difficult to perform accurately when variables occur multiple times in the expression of the constraint [17]. This is due to the loss of dependency between the different occurrences of a variable in interval arithmetic.

Box-consistency is a notion weaker than Hull-consistency but more easily reachable in general.

Definition 2.3.6 (Box-consistency)**.** Consider a NCSP and let c be one of its constraint over the variables $(x_1, \ldots, x_n)$. Consider a box $\boldsymbol{x}$ and let $\boldsymbol{x}_i^-$ and $\boldsymbol{x}_i^+$ be the ith left and right facets (or ϵ-facets) of $\boldsymbol{x}$, and an interval extension $\boldsymbol{c}$ of c. The constraint c is *box-consistent* over a box $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ if

$$\forall i,\ c \text{ is certainly or possibly satisfied on } \boldsymbol{x}_i^- \text{ and } \boldsymbol{x}_i^+, \tag{2.40}$$

where certain and possible satisfaction are determined with respect to $\boldsymbol{c}$ as in Definition 2.3.1 (p. 19). If all constraints of the NCSP are box-consistent over $\boldsymbol{x}$, the NCSP is box-consistent.

A box-consistent box can be obtained by locating leftmost and rightmost solutions of the constraint along each variable domain given the other variables are evaluated on their whole domain. Hull-consistency implies box-consistency. This is reciprocally true if no variable occurs more than once in any constraint of the NCSP [17].

Both hull and box-consistency consider each constraint of the NCSP independently. Hence, possible relations between constraints are not taken into account. This gives rise to situations where large boxes are hull or box consistent over each constraint. For example, if we consider the simple NCSP with two equality constraints depicted on Figure 2.7(a), the box is hull and box consistent for both constraints but clearly far from sharply enclosing the solution to the NCSP.

In order to deal with such situations, higher order consistencies have been stated so as to consider all constraints of the NCSP simultaneously. 3B-consistency generalizes Hull-consistency to all constraints of a NCSP. It is defined as follows :

Definition 2.3.7 (3B-consistency)**.** Consider a NCSP as (2.15) and a box $\boldsymbol{x}$. Let $\boldsymbol{x}_i^-$ and $\boldsymbol{x}_i^+$ the ith left and right facets (or ϵ-facets) of $\boldsymbol{x}$. The NCSP is *3B-Consistent* over $\boldsymbol{x}$ if $\forall i$, the NCSP is hull-consistent on $\boldsymbol{x}_i^-$ and $\boldsymbol{x}_i^+$.

The box depicted on Figure 2.7(b) is clearly not 3B-consistent as the facets are not hull-consistent. Removing these facets would allow to obtain the hull-consistent box in green. Replacing hull-consistent facets by box-consistent ones in the definition of 3B-consistency yields to bound-consistency [17]. Note that kB-Consistency is straightforwardly defined : a box $\boldsymbol{x}$ is kB-consistent for a NCSP if all its facets (or ϵ-facets) are $(k-1)$B-Consistent.

Obtaining a 3B-Consistent or bound-consistent box requires computing leftmost and rightmost hull or box-consistent facets. This implies in principle that contractors based on 3B or bound-consistency require more computations than contractors based on hull or box-consistency.

Eventually The concept of 3B-consistency and bound-consistency can also be extended by considering not ϵ-facets but a decomposition of the box that have to be consistent. This principle yield to Constructive Interval Disjunction consistency, i.e. CID-consistency [128].

Definition 2.3.8 (CID-consistency)**.** Consider a NCSP as (2.15) and a box $\boldsymbol{x}$. Consider in addition a regular paving $\mathcal{P}$ of $\boldsymbol{x}$, and consider for all $\boldsymbol{x}' \in \mathcal{P}$ the smallest consistent box $\boldsymbol{x}'' \subseteq \boldsymbol{x}'$ for the NCSP. Then $\boldsymbol{x}$ is CID-consistent for the NCSP with respect to the paving $\mathcal{P}$ if $\boldsymbol{x}$ is equal to the hull of all boxes $\boldsymbol{x}''$.

This consistency is dependent on the way the paving of $\boldsymbol{x}$ is built but also on the consistency considered for the boxes $\boldsymbol{x}''$ (in general, hull or box-consistency). Figure 2.7(c) depicts a box not CID-consistent with respect to a paving made of three sub-boxes along variable x_1. The left and right sub-boxes are not hull-consistent. The middle one contains the green hull-consistent box. Obtaining a CID-consistent box can be done by decomposing the box (in general splitting regularly along variables), reducing the boxes of the decomposition and then build the hull of reduced boxes.

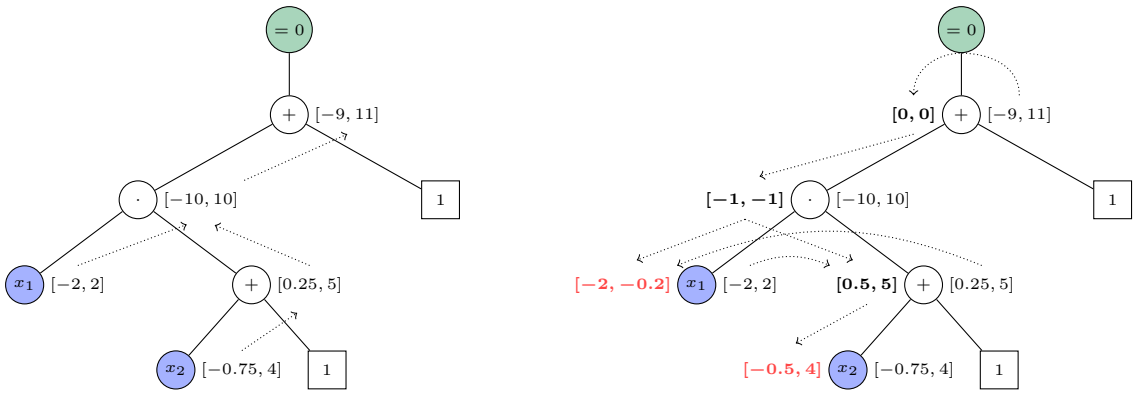
2.3.2.2 Contractors

We present here contractors based on the local consistency presented previously. First, we present usual contractors based on hull and box-consistency, then contractors based on 3B or bound-consistency.

The method HC4 [6] is a contractor that tries to attain hull-consistent boxes. Given a constraint c of a NCSP (2.15) and a box $\boldsymbol{x}$, HC4 exploits the expression tree of the constraint. It performs a forward evaluation of all the elementary sub-expressions of the constraint (i.e. arithmetic operations and unary constraints), intersects the value of the constraint expression at the root with the satisfying value ($[-\infty, 0]$ or $[0, 0]$ if the constraint is an inequality or equality) and propagates backward to the variable domains at the leaves. Forward evaluation is simply done by evaluating an interval extension of all sub-expressions. Backward propagation is done by performing relative, or inverse, interval operation of the corresponding child sub-expressions.

Example 2.3.4 – Consider the constraint $x_1 + x_2 = 0$ and the box $\boldsymbol{x}$ with $\boldsymbol{x}_1 = [-3, 1]$ and $\boldsymbol{x}_2 = [-2, 5]$. The sum of the two variables over the domain $\boldsymbol{x}$ yield to $[-5, 6]$, which contains 0. The constraint is then potentially satisfied over $\boldsymbol{x}$. HC4 propagates the value 0 to the variable domains by replacing the expression of the constraints by its relative expressions, namely $x_1 = 0 - x_2$ and $x_2 = 0 - x_1$. The new contracted box domains $\boldsymbol{x}'$ can be computed as $\boldsymbol{x}'_1 = ([0, 0] - \boldsymbol{x}_2) \cap \boldsymbol{x}_1 = [-3, 1]$ and $\boldsymbol{x}_2 = ([0, 0] - \boldsymbol{x}_1) \cap \boldsymbol{x}_2 = [-1, 3]$. Here, only the domain of x_2 has been contracted.

There are interval relative operations for all elementary operations : arithmetic operations and unary interval functions. Note that the relative operation of the multiplication is the extended division, possibly returning an union of intervals whose hull has to be eventually computed.



(a) Forward evaluation (b) Backward projection

Figure 2.8 – Example of an application of HC4 contractor procedure

Example 2.3.5 – Consider the constraint $x_1(x_2+1)+1=0$ and the box $\boldsymbol{x}$ with $\boldsymbol{x}_1=[-2,2]$ and $\boldsymbol{x}_2=[-0.75,4]$. The forward evaluation of the expression tree of the constraint is depicted on Figure 2.8(a). Each node are evaluated using natural interval extension. Backward propagation is depicted on Figure 2.8(b). When the node for the operation $\cdot$ is treated, the relative operation, namely the extended division, is performed. Denoting $z=x_2+1$ (the right child of the $\cdot$ node), we have that $x_1=[-1,-1]/z$ and $z=[-1,-1]/x_1$. We obtain new domains for the child nodes of $\cdot$, respectively

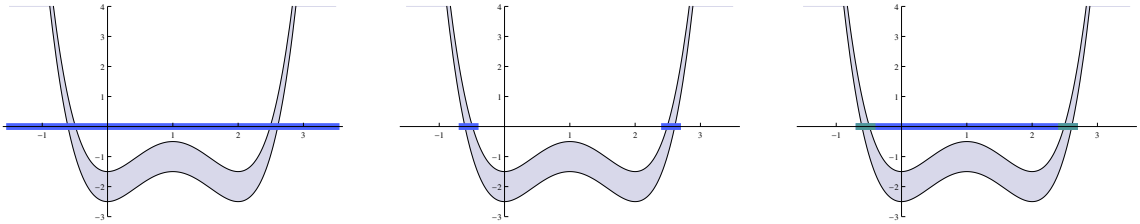
$$\boldsymbol{x}_1' = ([-1,-1]/[0.25,5]) \cap \boldsymbol{x}_1 = [-4,-0.2] \cap [-2,2] = [-2,-0.2],$$

$$\boldsymbol{z}' = ([-1,-1]/[-2,2]) \cap \boldsymbol{z} = ([-\infty,-0.5] \cup [0.5,+\infty]) \cap [0.25,5] = [0.5,5].$$

The domain of both variables have been eventually reduced. The new contracted box $\boldsymbol{x}'$ is here hull-consistent for the constraint. Note that not using the extended division would have failed to contract the domain of x_2.

In the case of variables occurring multiple times in the expression of a constraint, the method HC4 can produce at best box-consistent box [17]. The method has however a small computational cost and can be improved by representing the expression of a constraint by a Directed Acyclic Graph (DAG), allowing to share multiple-occurrences of a sub-expression. In general, obtaining a hull-consistent box for a NCSP is done by applying HC4 contractors on all constraints individually. A hull contractor on all constraints can be also obtained by applying HC4 on a DAG containing all the expressions of all constraints [116, 132].

Contracting methods based on box-consistency [131, 6, 38] can produce stronger narrowing than HC4, but are generally more computationally expensive. There are different possible instantiations of a narrowing operator based on box-consistency. They are based on equivalent principles. We briefly present here the BC3 procedure [131]. Given a constraint c of a NCSP and a box $\boldsymbol{x}$, consider the univariate constraint c_i which is c except that all variables x_j, $j \neq i$, are replaced by their domain $\boldsymbol{x}_j$. The BC3 contractor locates the leftmost and rightmost solutions of c_i in the domain $\boldsymbol{x}_i$. This is usually done by bisecting the domain of $\boldsymbol{x}_i$, checking satisfiability of leftmost and rightmost parts, and the application of univariate Newton to precisely enclose those extremal solutions.



(a) Initial state

(b) Location of the leftmost and rightmost consistent intervals.

(c) Hull of the two consistent intervals.

Figure 2.9 – Narrowing operator based on box-consistency applied on Example 2.3.6

Example 2.3.6 – Let the constraint $x_2 + (x_1^2 - 2x_1)^2 - 2 = 0$. Let $\boldsymbol{x} = ([-1.5, 3.5], [-0.5, 0.5])$ and suppose we want to contract $\boldsymbol{x}$ using box-consistency, with respect to variable x_1. Hence, the constraint c_1 is $\boldsymbol{x}_2 + (x_1^2 - 2x_1)^2 - 2 = 0$, and depicted Figure 2.9(a). Several bisections of $\boldsymbol{x}_1$ allows to find with univariate Newton the leftmost consistent interval $[-0.65, -0.45]$ and rightmost consistent interval $[2.45, 2.65]$ as depicted on Figure 2.9(b) (at a given precision ϵ). Eventually, the hull of those two intervals is constructed, see Figure 2.9(c).

A contractor based on box-consistency typically implements the above mentioned procedure along a single variable (i.e. for each constraint c_i). A box-consistent contractor for the constraint c is then obtained by applying contractors on each constraint c_i. Contractors based on box-consistency are in general better for contracting than ones based on hull-consistency (e.g. using HC4), noticeably when there are multiple occurrences of variables in the constraint expressions. On the other hand, contractors for box-consistency are more computationally expensive.

As we have previously seen, contractors based on hull or box-consistency can sometimes not contract large boxes far from sharply enclosing solutions of all constraints of a NCSP. Contractors based on 3B-consistency able to produce further reduction in such situations.

Shaving or peeling [70, 71] is a method implementing such a contractor. It somehow resembles the contracting operator based on box-consistency, but applied to the aggregation of all constraints of the NCSP. The idea is to eliminate iteratively inconsistent ϵ-facets of a box $\boldsymbol{x}$ over the whole NCSP, using a contractor on all constraints. This elimination is typically based on the application of a contractor on each individual constraint. Controlling the size of ϵ-facets to eliminate is difficult and can result in poor performances of the process (high computational cost compared to the effective contracting power).

Contractor based on CID-consistency are analogously implemented. A CID contractor decompose the input box, apply a contractor over the whole NCSP on each sub-box of the decomposition (as for the elimination of ϵ-facets in peeling) and construct the hull of contracted sub-boxes. With this decomposition, it is expected that a contractor is more effective on smaller sub-boxes, while enabling a global narrowing of $\boldsymbol{x}$. Returning the hull of all reduced sub-boxes is necessary in order to manipulate only interval domains (and not union of intervals). Hence, the contractor can discard an inconsistent sub-box "inside" $\boldsymbol{x}$ while the hull operation can reintroduce it into the returned box.

Eventually, for equality constraints of a NCSP, interval Newton can be used as a contractor. It is indeed its usual application in constraint satisfaction problem. A contracting operator is simply obtained by intersecting interval Newton iterations by $\boldsymbol{x}_k$ as presented in Section 2.3.1. Without preconditioning, the Newton contractor would not be able to further narrow the box on Figure 2.7(a), as the interval Newton operates the different equations independently. Nevertheless,

Algorithm 2: Constraint Propagation

Input: A set of contractors Θ, a box $\boldsymbol{x}$

2.1 $\mathcal{S} \leftarrow \Theta$;
2.2 $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}$;
2.3 $k \leftarrow 0$;
2.4 **while** $\mathcal{S} \neq \emptyset$ **do**
2.5 $\quad \theta \leftarrow \text{Extract}(\mathcal{S})$;
2.6 $\quad \boldsymbol{x}_{k+1} \leftarrow \theta(\boldsymbol{x}_k)$;
2.7 $\quad \mathcal{S} \leftarrow \text{Update}(\mathcal{S}, \Theta, \boldsymbol{x}_k, \boldsymbol{x}_{k+1})$;
2.8 $\quad k \leftarrow k + 1$;
2.9 **end**
2.10 **return** $\boldsymbol{x}_k$

preconditioning acts as a global reformulation of the equations, therefore allowing to contract very fast to the solution of the problem on Figure 2.7. Newton contractors additionally check existence and uniqueness of solutions within a box satisfying equations. Note however that if the solution lies on the boundary of the initial box, such certificate cannot be obtained as strict contraction cannot be observed. This can be done by accepting a slight shift of the box, but this will violate the definition of a contractor [5].

2.3.2.3 Constraint Propagation

Given a set Θ of contractors of a NCSP, one would use them efficiently in the pruning step of the B&P. As two contractors can operate on different constraints, their order of application may produce different contracted boxes. Moreover, interleaving different contractors may enable a single idempotent contractors, i.e. $\theta(\theta(\boldsymbol{x})) = \theta(\boldsymbol{x})$, to produce further narrowing. Finally, one would generally be certain that a box returned by the pruning step cannot be more contracted. Managing the applications of different contractors to this end is called constraint propagation, performed by algorithms called propagators.

A propagator is an algorithm that applies a set of contractors in a given order to a box. Algorithm 2 presents a propagator, which can be viewed as an AC3-like algorithm [77]. The set of contractors Θ must contain contractors for all constraints of the NCSP. Different strategies of propagation can be considered using different implementation of the procedures *Extract* and *Update*. The former selects which contractor in $\mathcal{S}$ to apply (this contractor is removed from $\mathcal{S}$), the latter inserts back into $\mathcal{S}$ contractors from Θ. A propagation algorithm could simply consist of sequentially applying all the contractors once (i.e. the procedure *Update* does nothing and the algorithm terminates once $\mathcal{S} = \emptyset$). Usually, a propagator is implemented as a fixpoint procedure : contractors are applied until $\boldsymbol{x}_{k+1} \approx \boldsymbol{x}_k$. In that case, at each iteration, the procedure *Update* insert contractors from Θ that can produce further narrowing. Which contractors to insert depends on which variable domains have been sufficiently reduced. In general, the domain of a variable $\boldsymbol{x}_{k,i}$ is said to be significantly reduced if its width is greater than a prescribed precision ϵ_i (usually the same as in termination criteria of B&P) and/or the ratio of contraction $\text{wid}(\boldsymbol{x}_{k,i})/\text{wid}(\boldsymbol{x}_{k-1,i})$ is below a prescribed threshold μ_i in order to avoid slow convergence. Contractors that are based

5. Such certificates can be made at the end of the B&P, possibly resulting in a non-regular paving

on constraints involving variables whose domain has been sufficiently reduced are inserted back into $\mathcal{S}$. Hence, Algorithm 2 terminates once all variables domains cannot be sufficiently reduced.

Contractors and propagators are excellent procedures for reducing the domains of boxes in B&P. They do not avoid splitting though, therefore efficient splitting strategies must be used. Eventually, note that a propagator is also a contractor, based on the constraints involved by the contractors in Θ. For example, propagators are often applied for contracting/discarding ϵ-facets and boxes in the peeling or CID contractors.

2.3.3 Search strategy

The search strategy concerns how boxes are split and how they are extracted. Splitting strategies in the B&P is critical for obtaining good performances. Usually, the interval domain of a variable is bisected at its middle yielding two boxes. The selection of the variable to bisect is important as it can speed up the detection of non-satisfaction of constraints or improve the pruning of boxes after bisection. Several selection strategies have been proposed in the literature, see [60, 18, 127]. We briefly describes the most used ones :

- Round Robin : each variable is selected in a fixed order. This ensures that all variables are fairly selected for bisection.
- Max Domain : a variable i is selected if its domain $\boldsymbol{x}_i'$ is the larger one. This yields to reducing efficiently the overestimations of (natural) interval evaluations.
- Max Smear : the smear of a variable x_i evaluates its influence on the variation of the constraints expressions. It is defined as

$$\max_j \quad \mathrm{mag}(\boldsymbol{a}_{ji})\mathrm{wid}(\boldsymbol{x}_i), \tag{2.41}$$

  where $\boldsymbol{A} = (\boldsymbol{a}_{ij})$ is the interval Jacobian matrix of the constraints expression in the NCSP (2.15) evaluated at $\boldsymbol{x}'$. This criterion is motivated for enhancing interval Newton operations as it tends to reduce the overestimations of mean value extensions.

Other strategies heuristically mix different criteria [43]. Selecting the best selection criterion is hence highly dependent on the NCSP, and on the different contractors that are used. Adapting the strategy to the NCSP is a difficult task. More often, an overall robust strategy for a set of problems is used and changed only if a problem is really more difficult to solve. Note that in order to ensure the convergence of B&P, see below, the selection strategy must be balanced : all variables must be asymptotically selected infinitely many times by the splitting strategy given an initial box $\boldsymbol{x}$. Termination is ensured in practice if variables whose domain width is smaller than the prescribed precision used for the termination criterion are never selected.

The strategy for extracting boxes can be critical for the memory consumption of the algorithm, which also depends on the problem considered. For example if the equations of the NCSP form a (regular) square system, then it is likely that there is a finite number of punctual solutions to the NCSP. In that case, a depth-first search, i.e. implementing the set of boxes $\mathcal{S}$ as a stack, is more convenient. This helps to focus on the identification of boxes containing solutions, and reduce them until termination criterion is met. The splitting strategy shall also be selected to accelerate the convergence to terminal boxes.

If on the contrary no equality constraints are considered, then the paving of the solutions to the NCSP may be dense. In that case, breadth-first search, i.e. implementing $\mathcal{S}$ as a queue, can

be convenient so as to build a uniform paving anytime during the algorithm. The *Max Domain* splitting strategy associated to this extraction strategy is well suited for building such uniform paving.

2.3.4 Convergence

We state here the asymptotic convergence of the B&P algorithm when the precision used for termination criteria is 0. Additionally, we suppose here that there is no termination criterion based on certain constraint satisfiability. In order to prove the convergence of Algorithm 1, we first need notions of accumulation point for infinite sequence of boxes, and notions of convergent sequence of boxes.

Definition 2.3.9 (Accumulation point for sequence of boxes)**.** Let $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$ be an infinite sequence of boxes $\boldsymbol{x}^{(k)} \in \mathbb{IR}^n, \forall k$. A point $x \in \mathbb{R}^n$ is an *accumulation point of the sequence of boxes* $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$ if there exists points $x^{(k)} \in \boldsymbol{x}^{(k)}$ for which x is an accumulation point of the sequence of points $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$.

Definition 2.3.10 (Convergent infinite sequence of boxes)**.** Let $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$ be an infinite sequence of boxes $\boldsymbol{x}^{(k)} \in \mathbb{IR}^n, \forall k$. This sequence of boxes is *convergent* if $\lim_{k\to\infty} \mathrm{wid}(\boldsymbol{x}^{(k)}) = 0$.

We can note that when precision is set to 0, Algorithm 1 is producing infinite sequence of boxes via the splitting step. If the splitting strategy is balanced, these sequences of boxes are convergent. The convergence of the B&P can now be stated.

Theorem 2.3.8 (Convergence of Branch & Prune)**.** *Given a NCSP as* (2.15) *and consider Algorithm 1 is using a breadth-first search and balanced selection criterion as search strategy. Denote by $\mathcal{U}$ the set of accumulation point for each infinite sequence of boxes obtained by successive splitting of $\boldsymbol{x}^{\text{init}}$ in Algorithm 1. Suppose that $\boldsymbol{g}$ and $\boldsymbol{h}$ are convergent interval extensions. Then $\mathcal{U} = \mathcal{X}$.*

Proof.

First, we prove $\mathcal{U} \subseteq \mathcal{X}$. Suppose there is an $x \in \mathcal{U}$ such that $x \notin \mathcal{X}$. Since the splitting selection is balanced, then there exists an infinite sequence of boxes obtained by successive splitting $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$ with $\lim_{k\to\infty} \boldsymbol{x}^{(k)} = x$. Since the interval extension of the constraints are convergent, then $\lim_{k\to\infty} \boldsymbol{g}(\boldsymbol{x}^{(k)}) = g(x)$ and $\lim_{k\to\infty} \boldsymbol{h}(\boldsymbol{x}^{(k)}) = h(x)$. Since $x \notin \mathcal{X}$, there is a i or j such that $g_i(0) > 0$ or $h_j(0) \neq 0$. Thus, there exists a $\overline{k} > 0$ such that $\underline{g_i(\boldsymbol{x}^{(\overline{k})})} > 0$ or $0 \notin \boldsymbol{h}_j(\boldsymbol{x}^{(\overline{k})})$, yielding $\boldsymbol{x}^{(\overline{k})}$ to be discarded by the pruning step contradicting the infiniteness of this sequence of boxes, hence $x \in \mathcal{U}$.

Now we prove $\mathcal{X} \subseteq \mathcal{U}$. The pruning step, through the properties of contractors, yields that if there is an $x \in \mathcal{X}$, then there is a $\boldsymbol{x} \in \mathcal{S}$ such that $x \in \boldsymbol{x}$, i.e the solution x is not lost. Due to the breadth first search strategy, there is an infinite sequence of boxes obtained by successive splitting $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$ containing x, and since the splitting strategy is balanced, $\lim_{k\to\infty} \boldsymbol{x}^{(k)} = x$. Thus $x \in \mathcal{U}$.

□

Hence, the boxes produced by B&P accumulates on the solutions to the NCSP. Note that it is easy to deduce from this proof that if $\mathcal{X} = \emptyset$, then Algorithm 1 does not produce asymptotically infinite sequence of boxes, i.e. it terminates proving $\mathcal{X} = \emptyset$. Note also that if inner boxes are considered as terminal, hence stored in $\mathcal{S}_{out}$, we can see that $\mathcal{U}$ equals the border of $\mathcal{X}$ while the union of boxes in $\mathcal{S}_{out}$ converges to $\mathrm{int}(\mathcal{X})$.

2.4 Global optimization

NCSP are rigorously solved via B&P which builds a regular paving of the solutions with boxes. We have presented several techniques to improve the pruning step : interval Newton certificates and contractors. In a NCSP, there is no preference on the solutions to compute. Many problems, namely global optimization problems, contain a nonlinear objective function that discriminates between solutions satisfying a set of constraints. The goal of global optimization is to find the feasible solutions that are the best with respect to the objective function.

We consider then the problem of finding the minimum of a (nonlinear) objective function $f : \mathbb{R}^n \to \mathbb{R}$ over a set $\mathcal{X}$ defined by a set of constraints [6]. This optimization problem is posed as follows :

$$\begin{bmatrix} \min & f(x) \\ \text{s.t} & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in \mathbb{R}^n \end{bmatrix}, \tag{2.42}$$

where $g : \mathbb{R}^n \to \mathbb{R}^p$ is a vector of inequality constraints, $h : \mathbb{R}^n \to \mathbb{R}^q$ is a vector of equality constraints and $\mathcal{X} = \{x \in \mathbb{R}^n : g(x) \leq 0, h(x) = 0\}$ denotes the set of feasible solutions. Solving (2.42) requires computing globally optimal solutions.

Definition 2.4.1 (Optimality)**.** A feasible solution $x \in \mathcal{X}$ is globally optimal if there is no other $x' \in \mathcal{X}$ such that $f(x') < f(x)$. We note $\mathcal{X}^*$ the set of all globally optimal solutions and $y^* = f(x), x \in \mathcal{X}^*$, the optimal objective value.

The optimal value y^* is unique and $\mathcal{X}^*$ is often a singleton (a single optimal solution). Due to the nonlinearity of the objective function and constraints, there may also exist locally optimal solutions.

Definition 2.4.2 (Local optimality)**.** A feasible solution x is locally optimal if there exists a $\delta > 0$ such that there is no other $x' \in \mathcal{B}(x, \delta) \cap \mathcal{X}$ with $f(x') < f(x)$.

The presence of local optima raises difficulties in solving the optimization problem in many solving methods. In the case of convex problems, i.e with a convex objective and convex feasible set, we have that any locally optimal solution is a globally optimal solution.

In general solving methods only assert local optimality of the solution they compute. Asserting the (local) optimality of a solution can be done by checking necessary and sufficient condition for optimality. First, we present the Fritz-John necessary conditions for optimality [8].

Theorem 2.4.1 (Fritz John necessary optimality condition)**.** *Consider the optimization problem (2.42). A necessary condition for a solution $x \in \mathcal{X}$ to be (locally) optimal is that there exist multipliers $0 \leq \lambda \in \mathbb{R}$, $0 \leq r \in \mathbb{R}^p$ and $s \in \mathbb{R}^q$ (λ, r, s not all 0) such that :*

$$F(x, \lambda, r, s) = \begin{pmatrix} \nabla f(x)\lambda + \nabla g(x)r + \nabla h(x)s \\ (\forall i = 1, \ldots, p)\ g_i(x)r_i \\ (\forall i = 1, \ldots, q)\ h_i(x) \end{pmatrix} = 0. \tag{2.43}$$

6. Maximizing f is equivalent to minimizing $-f$.

The system (2.43) is underconstrained with $n+p+q+1$ variables and $n+p+q$ equations. An additional normalization constraint for the multipliers can be considered, such as $\lambda^2 + r^T r + s^T s - 1 = 0$. Note that solutions satisfying this system must also satisfy the constraint of the optimization problem and the domain of multipliers. This condition is necessary for (local) optimality but not sufficient. Local maxima or saddle solutions, in general, also satisfy these conditions. Nevertheless, these conditions can still be used to perform some filtering. We will further refer to solutions solving the system (2.43) as stationary solutions.

Definition 2.4.3 (Stationary solutions)**.** A solution $x \in \mathcal{X}$ is *stationary* for problem (2.42) if there exist multipliers $0 \leq \lambda \in \mathbb{R}$, $0 \leq r \in \mathbb{R}^p$ and $s \in \mathbb{R}^q$ (λ, r, s not all 0) solving system (2.43).

Note that if a stationary solution $x \in \mathcal{X}$ with multipliers (λ, r, s) is singular for the system 2.43, it is numerically difficult to assert its stationarity. Singularity of the system 2.43 follows Definition 2.3.2 (p. 23).

Definition 2.4.4 (Singular stationary solution)**.** A stationary solution $x \in \mathcal{X}$ with multipliers (λ, r, s) is nonsingular for (2.43) if its Jacobian given by :

$$F'(x,\lambda,r,s)) = \begin{pmatrix} \mathcal{L} & \nabla f(x) & \nabla g(x) & \nabla h(x) \\ r_1(\nabla g_1(x))^T & & & \\ \vdots & 0 & \mathrm{G} & 0 \\ r_p(\nabla g_p(x))^T & & & \\ (\nabla h(x))^T & 0 & 0 & 0 \end{pmatrix} \tag{2.44}$$

is full rank, where $\mathcal{L}$ is the matrix defined as

$$\nabla^2 f_i(x)\lambda + \sum_{i=1}^{p} \nabla^2 g_i(x) r_i + \sum_{i=1}^{q} \nabla^2 h_i(x) s_i, \tag{2.45}$$

and G is the diagonal matrix with $G_{ii} = g_i(x)$.

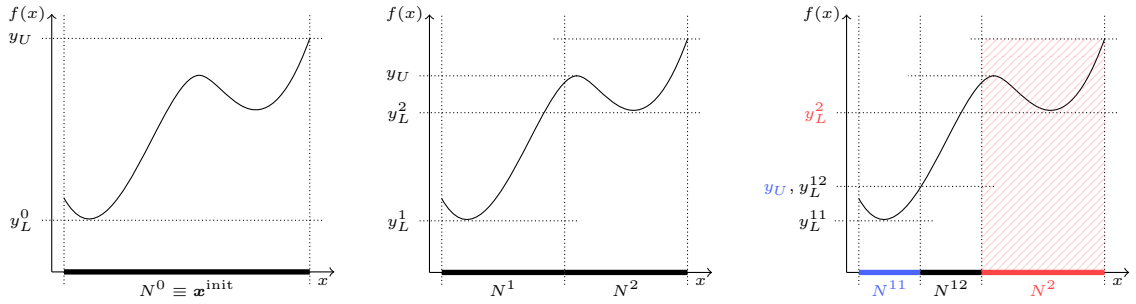
Another well known first-order necessary conditions for optimality are the Karush-Kuhn-Tucker (KKT). These conditions differ from Fritz John as they do additionally suppose the solution to satisfy constraint qualifications. This concept is recalled hereafter. First, we need to introduce the following notations :

Definition 2.4.5 (Active constraints)**.** Consider a solution $x \in \mathcal{X}$ to problem (2.42). An inequality constraint g_i is said active at x if $g(x) = 0$. The set $\mathcal{A}(x) = \{i : g_i(x) = 0\}$, of size $\dot{p}$, denotes the index of active constraints at x. The vector function of active inequalities at x will be denoted $\dot{g} : \mathbb{R}^n \to \mathbb{R}^{\dot{p}}$.

Note that in the system (2.43), if $i \in \mathcal{A}(x)$ then $r_i \geq 0$, otherwise $r_i = 0$. This observation allows some simplification or reformulation of the system (2.43).

Constraint qualification can now be introduced. We present here the Linear Independence Constraint Qualification (LICQ). Other constraint qualification exists but LICQ is the most used and most restrictive one.

Definition 2.4.6 (Linear Independent Constraint qualification)**.** Consider a solution $x \in \mathcal{X}$ to problem (2.42). Linear Independence Constraint Qualifications (LICQ) holds at x if the gradients $\nabla \dot{g}(x), \nabla h(x)$ are linearly independent.



(a) Initial sub-problem N^0.

(b) Splitting the sub-problem N^0 in two. The upper-bound can be updated by selecting N^1.

(c) Splitting of N^1. Sub-problem N^{11} yields to update the upper-bound which allows to discard N^2 from the search.

Figure 2.10 – Illustration of B&B on a single variable unconstrained problem.

At a solution not satisfying LICQ, some (active) constraints are either locally redundant, or a constraint gradient is zero. This yields that some constraints can be locally ignored which complicates the analysis of the (local) optimality of the solution. Not that solutions not satifying LICQ are singular for the system (2.43). Eventually, the KKT conditions can be stated [8].

Theorem 2.4.2 (Karush-Kuhn-Tucker necessary optimality condition)**.** *Consider the optimization problem* (2.42)*. Let $x \in \mathcal{X}$ be a solution satisfying LICQ. Then a necessary condition for x to be (locally) optimal is that it is stationary. In addition, the multiplier λ can be set to* 1*.*

As the value of the multiplier λ is fixed, the system (2.43) can be considered square in the case of solutions satisfying LICQ. Note that the KKT conditions are also sufficient in the case of convex problems. In order to check whether a solution satisfying KKT is locally optimal, sufficient conditions for local optimality based on second-order informations, using Hessians of the objective and constraints, are used [8].

Theorem 2.4.3 (Sufficient condition for optimality)**.** *Consider a solution $x \in \mathcal{X}$ for a problem* (2.42) *satisfying KKT, i.e with induced multipliers r and s. Let $\mathcal{L}$ be the matrix defined by* (2.45)*. A sufficient condition for x to be a (locally) optimal solution is :*

$$u^T \mathcal{L} u > 0, \ \forall u \in \{u : h'(x)u = 0\} \cup \{u : g_i'(x)u \leq 0, \forall i \in \mathcal{A}(x)\} \tag{2.46}$$

These conditions test the local convexity of the problem around a solution satisfying KKT. When no constraint is active at x (and no equalities in the problem), this condition checks whether the Hessian of the objective is positive definite, i.e. the objective function is convex around x.

The purpose of global optimization is to search for global optimal solutions. Ensuring that one of the global optima is found cannot be guaranteed in general. However, we can guarantee that the global optima are not lost during the solving process. As one can see, problem (2.42) resembles a constraint satisfaction problem with an additional constraint on the optimality of the solutions. The interval Branch & Bound (B&B) is a global search method that constructs a regular paving, up to a prescribed precision, of the set of global optimal solutions $\mathcal{X}^*$ [48, 61, 92, 131]. The algorithm can be viewed as a variant of the Branch & Prune in which boxes of the paving that certainly do not contain optimal solutions are discarded. The overall algorithm is stated in Algorithm 3.

Algorithm 3: Interval Branch & Bound

Input: Initial box $\boldsymbol{x}^{\text{init}}$; Optimization problem (2.42) ;
Output: List $\mathcal{S}_{out}$ of boxes containing the optimal solutions

3.1 $N_0 \leftarrow (\boldsymbol{x}^{\text{init}}, y_L^0)$;
3.2 $\mathcal{S} \leftarrow \{N_0\}$;
3.3 $y_U \leftarrow InitializeUB()$;
3.4 **while** $\mathcal{S} \neq \emptyset$ **do**
3.5 $\quad N = (\boldsymbol{x}, y_L) \leftarrow Extract(\mathcal{S})$;
3.6 $\quad Prune(\boldsymbol{x}, y_U)$;
3.7 $\quad UpdateBounds(\boldsymbol{x}, y_U, y_L)$;
3.8 $\quad$ **if** $\boldsymbol{x} \neq \emptyset$ **then**
3.9 $\quad\quad$ **if** N *is terminal* **then**
3.10 $\quad\quad\quad \mathcal{S}_{out} \leftarrow \mathcal{S}_{out} \cup \{N\}$;
3.11 $\quad\quad$ **else**
3.12 $\quad\quad\quad \mathcal{S} \leftarrow \mathcal{S} \cup Split(N)$;
3.13 $\quad\quad$ **end**
3.14 $\quad$ **end**
3.15 **end**
3.16 **return** $\mathcal{S}_{out}$

Given an initial variable domain $\boldsymbol{x}^{\text{init}}$ containing $\mathcal{X}$, the principle of B&B is to select a sub-problem N (initially the original problem with variable domain $\boldsymbol{x}^{\text{init}}$ and initial lower bound y_L^0), to try to discard the sub-problem by pruning its variable domain (using the lower bound y_L and upper bound y_U and the constraints of the problem), to update bounds on the objective on the sub-problem and finally to check some termination criteria. If these criteria are met, the sub-problem can be stored as possibly containing optimal solutions (its variable domain being part of the paving of the solutions), otherwise the sub-problem is decomposed by splitting its variable domain. We can also view this global search as exploring a tree of sub-problems, whose root node is the initial problem with $\boldsymbol{x}^{\text{init}}$ as domain of variables, and each child node is a sub-problem generated from its father node. The leaf nodes are either terminal sub-problems, for which the aim is to detect which one contains optimal solutions, or discarded sub-problems. In particular, a sub-problem can be safely pruned/discarded if its lower bound y_L is worse than the known upper bound y_U. Hence, having accurate lower bounds and a good upper bound allow to discard quickly during the search uninteresting sub-problems. Figure 2.10 illustrates few iterations of B&B on a single variable unconstrained problem, where lower and upper bounds are obtained by evaluating the objective function on the decision interval induced by each sub-problems.

Analogously as in B&P, sub-problems that do not contain optimal solutions have to be discarded from the search as soon as possible so as to avoid unnecessary splitting. To do so, critical steps have to be efficient : the pruning step (at line 3.6), the bounding step (at line 3.7) and the search strategy (extraction of sub-problems, splitting strategy and termination criteria). Each of these step are detailed in the following sections.

2.4.1 Pruning

The pruning step of the B&B tries to narrow the domain $\boldsymbol{x}$ of a sub-problem. To this end, several informations are available : a lower bound y_L of the objective function over the solutions in $\boldsymbol{x}$ and an upper bound y_U consisting of the objective value of the best feasible solution found so far. They are updated in the bounding step, see Section 2.4.2.

First, the sub-problem is checked for optimality. This is done via binary contractors which either discard entirely a box, or let it unchanged. Those binary contractors will be referred to discarding tests. A simple discarding test checks the inferiority of the known upper bound with respect to the lower bound of the sub-problem, i.e. $y_U < y_L$. Indeed, as all objective values of the solutions in $\boldsymbol{x}$ are bounded by y_L, then they are all worse than the best known upper bound. Thus, the sub-problem can safely be discarded, i.e. the pruning step returns an empty set.

Another usual discarding test checks the monotonicity of the objective function in $\boldsymbol{x}$ [61]. Given an interval evaluation $\nabla \boldsymbol{f}(\boldsymbol{x})$ of the gradient of f over $\boldsymbol{x}$, if there is a variable i such that $\nabla_i \boldsymbol{f}(\boldsymbol{x}) < 0$ or $\nabla_i \boldsymbol{f}(\boldsymbol{x}) > 0$, then, due to the containment principle of interval evaluations, the objective is monotone (either decreasing or increasing) along x_i. Hence, the best solution in $\boldsymbol{x}$ is on the induced facet $\boldsymbol{x}_i^+$ or $\boldsymbol{x}_i^-$ respectively. The node can be discarded if in addition the facet does not intersect the boundary of the solution set, as it cannot be stationary. Note that this test cannot be applied with equality constraints.

Eventually, there are discarding tests based on the verification of necessary or sufficient condition for optimality. Concerning first-order necessary conditions, three tests have been proposed in [37] that are generalized to multiobjective optimization, or vector optimization. Hence, these tests are fully described in the next chapter in Section 3.3.3 (page 70). Briefly, they check whether it is possible within $\boldsymbol{x}$ to find (normalized) multipliers λ, r and s satisfying the n first equations of (2.43) at different level of analysis. They use the notion of potentially active inequalities.

Definition 2.4.7 (Potential constraint activation)**.** Given a box $\boldsymbol{x}$, a constraint $g_i(x) \leq 0$ is said potentially active in $\boldsymbol{x}$ if $0 \in \boldsymbol{g}_i(\boldsymbol{x})$.

The system (2.43) is restricted to the potentially active constraints in $\boldsymbol{x}$ (including equality constraints) in the discarding tests from [37].

Tests based on second-order sufficient conditions consider, in general, only the unconstrained case (or the case of inner boxes with respect to the constraints of the problem). If there are possibly stationary solutions within an inner box $\boldsymbol{x}$, a simple second-order test is then to check whether the objective function is certainly nonconvex in $\boldsymbol{x}$. A necessary condition is that there exist a variable x_i such that $\nabla^2_{ii} \boldsymbol{f}(\boldsymbol{x}) < 0$ [61, 48]. Sharper analysis of the nonconvexity of f in $\boldsymbol{x}$ can be performed [61, 91].

Constraint propagation on the constraints of problem (2.42), as in constraint satisfaction problems, allows pruning the domain $\boldsymbol{x}$ efficiently [48]. In general, the constraint $f(x) - y_U \leq 0$ is inserted into the constraint system in order to remove from $\boldsymbol{x}$ solutions that are necessarily worse than the upper bound. Note that discarding tests are generally not inserted into the list of contractors for the propagator but applied prior to it so as to avoid propagating if the sub-problem can be discarded.

Additionally, the first-order system (2.43) can be introduced in the constraints of the problem, provided new variables (and domains) for the multipliers are introduced. Applying contractors on this system (e.g. via interval Newton) is an usual approach for discarding a box via first-order conditions [48, 61]. The use of interval Newton on equations (2.43) requires computing Hessians of the objective and constraints.

Remark 2.4.1 – Computing all stationary solutions can be viewed as a way of globally solving (2.42) as a NCSP, the stationary solutions being filtered as a post-process with respect to their objective values.

Eventually, it can be important to construct exclusion regions : regions of the search space for which local optimal solutions are proved to be contained in a box $\boldsymbol{x}$ [115]. Computing an exclusion region can be made after proving the existence (and uniqueness) of solutions to (2.43) in $\boldsymbol{x}$ using interval Newton methods.The idea is to find the largest box around $\boldsymbol{x}$ that is converging to the solution in $\boldsymbol{x}$ via interval Newton. This can be done using ϵ-inflation in order to inflate $\boldsymbol{x}$ to this limit box. In [115] a more efficient approach for building such exclusion region is proposed. This technique uses third-order informations to produce exclusion regions. As the optimal solutions in an exclusion region are proved to be contained in a (small) box $\boldsymbol{x}$, then any sub-problem whose variable domain lies within the exclusion region can be safely discarded. This technique avoids possible cluster effects [24] in the final paving around optimal solutions : possibly large cluster of boxes which cannot be discarded. This cluster effect slows down the convergence of the B&B dramatically.

2.4.2 Bounding

The bounding step updates the lower bound of the considered sub-problem and tries to improve the upper bound, i.e. the best known feasible solution found. A lower bound of the objective f over $\boldsymbol{x}$ can be easily obtained by evaluating an interval extension of f over $\boldsymbol{x}$. This bound may be however not of good quality as the interval extension may contain large overestimation and as the box $\boldsymbol{x}$ may contain infeasible solutions. A better way of improving the lower bound is via linear relaxations of the optimization problem. Linear optimization problems are easy to solve via the well known simplex algorithm. The difficulty is to ensure that a linear relaxation actually safely bounds below the original problem. This can be handled via interval analysis, see e.g [68, 127].

Updating the upper bound requires finding a feasible solution whose objective value improves y_U. As the upper bound is used to discard sub-problems, its feasibility has to be stated rigorously. A simple strategy for generating a new solution is to pick the midpoint of $\boldsymbol{x}$. Its feasibility can be assessed evaluating the constraints on the induced degenerated interval. The same is done for the objective. In practice, such interval evaluations ensure the rigor of the potentially new upper bound even in the presence of numerical imprecisions. If there are equality constraints, certifying their satisfaction is compulsory and can be done using interval Newton methods (see Section 2.3.1). Once computed and found feasible, this new solution replaces the upper bound if improving its objective value. Other easily computable points from $\boldsymbol{x}$ can be computed, such as corner points (e.g. $\overline{x}$).

This technique is simple and does not cost many computations, but it does not improve efficiently the upper bound. Another strategy implies to use a local search procedure, an optimization method that computes by local means a locally optimal solution, for example Newton methods (for optimization) or interior point methods [134]. Local search implies many computations that make it maybe not reasonable to use at every sub-problem. Nevertheless, it is worth spending some efforts in finding a good quality upper bound before the search proceeds (in the method *InitializeUB* in Algorithm 3). In any case, those solutions still need to be asserted to satisfy the constraints. In the case of equality constraints, numerical proofs are necessary (see Section 2.3.1)

Note that usually, if the upper bound is successfully improved, sub-problems from $\mathcal{S}$ (and $\mathcal{S}_{out}$) whose lower bound are worse than the new upper bound are directly discarded, i.e. without

waiting to be removed by the pruning step. In practice, this save memory by avoiding to store many uninteresting sub-problems at any time of the algorithm.

2.4.3 Search strategy

The remaining part of the B&B that needs to be carefully designed concern the overall search strategy : ordering the boxes in $\mathcal{S}$ (thus the extracting strategy), splitting strategy and termination condition.

A usual and efficient ordering of $\mathcal{S}$ is by increasing lower bound of sub-problems y_L [61, 48]. Sub-problems having lowest lower bounds are most promising in containing a globally optimal solution. This strategy tries to locate as quickly as possible a global optimal solution of (2.42). This eases and accelerates the computation of good quality upper bound, and improves significantly the pruning of other sub-problems. The performances induced by this strategy depend on the quality of the computed lower bounds, i.e. on the bounding process.

As in the B&P, selecting the right variables to split in B&B can accelerate the convergence of the method. In general, bisection is performed along a selected variable x_i. The same selection strategies as in Section 2.3.3 can be used, except for the smear criterion which has to consider exclusively the objective function in the bound-constrained (or unconstrained) case [60, 18] or to consider the objective function and constraints alltogether [127]. Note that several variants of the smear criterion are proposed in [127] : max smear as in Section 2.3.3, sum of smear, and variants using a relative smear measure. Although not always the best strategies, selection criteria based on smear metrics are in general more robust. Note that in order to ensure convergence, the selection strategy must be balanced, i.e. ensure that all variables are asymptotically selected infinitely many times. This can be ensured by not selecting variables whose domain width is too small.

Eventually, termination criteria has to be selected so as to avoid unnecessary computations once a sufficiently satisfying and rigorous result is produced. As for the B&P, a basic termination rule is with respect to a precision on the variable domains. Once a sub-problem with domain $\boldsymbol{x}$ satisfies $\mathrm{wid}(\boldsymbol{x}_i) < \epsilon_i$, then the sub-problem is considered as terminal. The range of the objective value over $\boldsymbol{x}$ can also be used. One can also test the precision on the objective value in addition to the precision of variable domains, i.e. whether $\mathrm{wid}(\boldsymbol{f}(\boldsymbol{x})) < \epsilon_f$ [61]. The width of $\boldsymbol{f}(\boldsymbol{x})$ can be replaced by the distance between the lower bound of the sub-problem y_L and an upper bound over $\boldsymbol{x}$, or by the distance between y_L and y_U. These latter termination criteria suppose a prescribed precision on the objective value ϵ_f. One last termination criterion can be considered when interval Newton is applied on first-order systems (2.43). If a proof of existence and uniqueness of a solution to the system within $\boldsymbol{x}$ is made, the induced sub-problem is considered as terminal. Note that it may happen that sub-problems can be found terminal before finding an upper bound y_U that could discard them. Therefore, at the end of the algorithm, a final filtering of the sub-problems in $\mathcal{S}_{out}$ should be made.

2.4.4 Convergence

Convergence for the B&B algorithm is analogous to the B&P as the two algorithms are essentially the same. The main difference is that we consider a search strategy for which sub-problems with smaller lower bounds are extracted first until termination. Given a precision 0 is set for the termination criteria, infinite sequence of decision boxes are obtained only towards sub-problems containing the global optimal solutions. We consider here that no termination criteria is used.

Theorem 2.4.4 (Convergence of Branch & Bound)**.** *Consider an optimization problem as (2.42), and consider Algorithm 3 with balanced splitting strategy and extraction based on best lower bound. Denote by $\mathcal{U}$ the set accumulation points of each infinite sequence of decision boxes obtained by successive splitting of $\boldsymbol{x}^{\text{init}}$ in Algorithm 3. Suppose that $\boldsymbol{f}$, $\boldsymbol{g}$ and $\boldsymbol{h}$ are convergent interval extensions. Then $\mathcal{U} = \mathcal{X}^*$.*

Proof.

First, we prove that $\mathcal{U} \subseteq \mathcal{X}^*$. From the proof of Theorem 2.3.8, it is easy to see that any $x \in \mathcal{U}$ is feasible. We have to prove it is optimal. Suppose not. Then there is an $x' \in \mathcal{X}$ such that $f(x') < f(x)$. Due to the contractors properties, x' cannot be discarded hence it belongs to some sub-problems in $\mathcal{S}$. Denote by $\boldsymbol{x}'$ the decision box associated to such a sub-problem, and suppose its lower bound is $\underline{f(\boldsymbol{x}')}$ (hence $\underline{f(\boldsymbol{x}')} < f(x')$). By definition of x, there is an infinite and convergent sequence of boxes obtained by successive splitting $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$ such that $\lim_{k\to\infty} \boldsymbol{x}^{(k)} = x$. Since the interval extension $\boldsymbol{f}$ is convergent, we have that $\lim_{k\to\infty} \boldsymbol{f}(\boldsymbol{x}^{(k)}) = f(x)$. Therefore, there must exists an index $\overline{k}$ such that $f(x') < \underline{f(\boldsymbol{x}^{(\overline{k})})}$. Therefore, $\underline{f(\boldsymbol{x}')} \leq lbf(\boldsymbol{x}^{(\overline{k})})$ and due to the search strategy, the sup-problem involving $\boldsymbol{x}'$ is selected prior to the one involving $\boldsymbol{x}^{(\overline{k})}$, which is not selected anymore. Thus, this sequence of boxes is not infinite contradicting x is an accumulation point in $\mathcal{U}$. Thus, $\mathcal{U} \subseteq \mathcal{X}^*$.

We prove now that $\mathcal{X}^* \subseteq \mathcal{U}$. Consider $x \in \mathcal{X}^*$. Since no contractors can remove a solution in $\mathcal{X}^*$, there is a sub-problem in $\mathcal{S}$ whose decision box $\boldsymbol{x}$ contains x at any iteration of the algorithm. Thus due to the extraction strategy, x belongs to an infinite sequence of decision boxes obtained by successive splitting $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$, which is convergent since the splitting strategy is balanced. Hence, $\lim_{k\to\infty} \boldsymbol{x}^{(k)} = x$, and thus $x \in \mathcal{U}$.

□

Due to the properties of contractors used in the B&B, it is known that at any iteration of the algorithm $\mathcal{X}^* \subseteq \mathcal{S}$. This convergence theorem ensures that the decision boxes from the set of sub-problems $\mathcal{S}$ asymptomatically converge to $\mathcal{X}^*$, i.e reducing the precision of the decision boxes will eventually yields to enclose $\mathcal{X}^*$ tighter and tighter.

2.5 Conclusion

In this chapter, we have presented a broad overview of interval analysis and its application for solving rigorously and globally numerical problems such as nonlinear constraint satisfaction and global optimization. Two methods have been presented, namely Branch & Prune and Branch & Bound which share many common subprocesses such as pruning of variables domains via constraint propagation and decomposition of the search space. Branch & Bound is an efficient technique for the rigorous solving of nonlinear optimization problems. However, when optimal solutions are not regular, it can be difficult to converge quickly. Noticeably, some singular cases cause the existence of manifolds of optimal solutions which have to be computed [108].

Example 2.5.1 – Consider the two variables (linear) optimization problem with $f(x) = -(x_1 + x_2)$ and constraints $g_1(x) = x_1 \leq 1$, $g_2(x) = x_2 \leq 1$ and $g_3(x) = x_1 + x_2 - 1 \leq 0$. The global optimal solutions form a manifold satisfying $g_3(x) = 0$. The optimal solutions are singular as, for example, a slight change of the objective function change the nature of the optimal solutions (either $x = (1, 0)$ or $x = (0, 1)$).

These cases cause slow convergence of the Branch & Bound. In the following chapter is presented the field of nonlinear multiobjective optimization, in which optimal solutions are manifold under regularity assumptions. Hence, when we eventually design an interval Branch & Bound in this context, this difficulty has to be handled via some approaches similar to [108], i.e. computing efficiently and locally a paving of the manifold of solutions, and forbidding the B&B to search further in this paving (e.g. via the use of exclusion regions).

CHAPTER 3

Overview of nonlinear multiobjective optimization

3.1 Introduction **53**
3.2 Background on Nonlinear Multiobjective Optimization **53**
3.2.1 Definitions and notations 53
3.2.2 Optimality conditions 57
3.3 Solving multiobjective problems **62**
3.3.1 Scalarization methods 62
3.3.2 Continuation methods 67
3.3.3 Global search methods 70
3.3.4 Performance assessment 77
3.4 Conclusion **79**

3.1 Introduction

In this chapter is presented an overview of Nonlinear Multiobjective Optimization. Nonlinear multiobjective problems imply optimizing several conflicting objectives with respect to real-valued variables subject to constraints, where objectives and constraints can be nonlinear. This field has been widely studied over the last decades as many problems require to optimize different conflicting objectives. Due to these conflicts, many intuitions from single objective optimization cannot be used in a multiobjective context. For example, the notions of optimality are radically different as two solutions can be incomparable : a solution can be better than another with respect to one objective while the latter may be better on another objective.

This chapter presents first the necessary theoretical background on nonlinear multiobjective optimization that will be used throughout the thesis. Different notations are introduced as well as the most important definitions and theorems. Then, a state-of-the-art on multiobjective optimization methods is presented. Among the described methods are scalarization methods, well known techniques that transform the multiobjective problem into a parametric single objective one. Then, continuation methods are presented. These local approaches ave recently encountered a growing interest in the literature and generalize a traditional use of scalarization methods. Eventually, global approaches are presented with a particular focus on rigorous and complete methods such as multiobjective interval B&B. A short introduction to performance assessment in multiobjective optimization is also presented. The chapter is finally concluded on the necessity to combine continuation methods with B&B.

3.2 Background on Nonlinear Multiobjective Optimization

NonLinear (Constrained) MultiObjective Optimization (NLMOO) is the problem of simultaneously minimizing several criteria [25, 85] and can be formally defined as :

$$\begin{bmatrix} \min & f(x) \\ \text{s.t} & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in \mathbb{R}^n \end{bmatrix} \tag{3.1}$$

with $x \in \mathbb{R}^n$ the decision variables, $f : \mathbb{R}^n \to \mathbb{R}^m$ the objective functions, $g : \mathbb{R}^n \to \mathbb{R}^p$ inequality constraints and $h : \mathbb{R}^n \to \mathbb{R}^q$ equality constraints. The feasible region $\mathcal{X}$ is the set of decision vectors that satisfy all the constraints, i.e., $\mathcal{X} := \{x \in \mathbb{R}^n : g(x) \leq 0, h(x) = 0\}$. Its image $\mathcal{Y} = f(\mathcal{X})$ in the objective space is called the feasible objective region. This problem is nonlinear if at least one of the objective or constraint functions is nonlinear.

Solving Problem (3.1) implies finding the solutions yielding to an optimal trade-off of the objectives ; in other words, solutions for which there is no other solution strictly improving one of the objective.

3.2.1 Definitions and notations

As noted in introduction, comparing solutions to Problem 3.1 requires comparing their vector of objectives values. Thus, it is necessary to define some orders on vector values. As minimization problems are considered, we are interested in lower objective values. Throughout the thesis, we will use the following relations :

Definition 3.2.1 (Vector comparison and dominance relation)**.** Let y and y' be two vectors in $\mathbb{R}^m$. The following notations are used :

(i) $y < y' \equiv y_i < y'_i \quad \forall i = 1, \ldots, m$

(ii) $y \lneq y' \equiv y_i \leq y'_i \quad \forall i = 1, \ldots, m, \text{ and } y \neq y'$

(iii) $y \leq y' \equiv y_i \leq y'_i \quad \forall i = 1, \ldots, m$

Moreover in the case (i), we say that y *strictly dominates* y', and in case (ii) that y *dominates* y'.

Dominance is stated in the objective space, where the criteria are compared although we will later say that a solution x dominates another solution x' if $f(x)$ dominates $f(x')$. These relations can only be used to define a partial order of vector of objective values. Indeed, two vectors y and y' can be incomparable : no one dominating the other. From a multiobjective point of view, and without preferences on the criteria, two incomparable objective vectors have an equivalent quality. Of course if some preferences are considered, e.g. a lexicographic order of the objectives, it is possible to discriminate further two objective vectors. Hence, solving Problem (3.1) consists of finding the set of nondominated solutions, i.e solutions for which there is no other solution that improves all objectives (their objective values are not dominated by any other feasible objective value).

This concept of nondominance defines the concept of optimality in multiobjective problems. The following notations and definitions of optimality are used :

Definition 3.2.2 (Nondominance, weak nondominance, and optimality)**.** Consider Problem (3.1). Let y be a vector in the feasible objective space $\mathcal{Y} \subseteq \mathbb{R}^m$. Then y is called a *nondominated* objective vector of $\mathcal{Y}$ if there is no other $y' \in \mathcal{Y}$ such that $y' \lneq y$. The objective vector y is *weakly nondominated* if there is no other $y' \in \mathcal{Y}$ such that $y' < y$. A solution $x \in \mathcal{X}$ is called globally *Pareto optimal* if there is no other $x' \in \mathcal{X}$ such that $f(x') \lneq f(x)$, i.e. $f(x)$ is nondominated. Analogously, a solution $x \in \mathcal{X}$ is *weakly Pareto optimal* if there is no other $x' \in \mathcal{X}$ such that $f(x') < f(x)$, i.e. $f(x)$ is weakly nondominated. A Pareto-optimal solution (respectively nondominated objective vector) is also weakly Pareto-optimal (respectively weakly nondominated objective vector).

The set of all Pareto optimal solutions is denoted by $\mathcal{X}^*$. Its image in the objective space is called the *nondominated set* or *Pareto front* [1], and denoted $\mathcal{Y}^*$. The set of all weak Pareto optimal solutions is denoted $\mathcal{X}^*_W$ and its image in the objective space denoted $\mathcal{Y}^*_W$.

One can note that other optimality notions and terms are sometimes used in the literature, e.g. the notion of *efficiency* [25, 85]. The optimality definitions presented here are sufficient for the rest of the thesis. Dominance relations and optimality are illustrated on Figure 3.1. We can note that in general, Pareto optimal solutions form a manifold of dimension $m - 1$, possibly disconnected. Figure 3.1 ($m = 2$) shows such curves of Pareto optimal solutions.

Example 3.2.1 – Consider the two variable biobjective problem with $f_1(x) = -(x_1 + x_2)^2 - x_1 + x_2$, $f_2(x) = -(x_1 + x_2)^2 + x_1 - x_2$ and $g(x) = x_1^2 + 2x_2^2 - 1 \leq 0$. The decision and objective spaces of this problem are illustrated on Figure 3.1, where $f(x^i) = y^i$. The objective vector y^1, hence solution x^1, dominates all objectives in its upper right corner. Here, y^1 dominates y^3 but is dominated by y^4. The two vectors y^1 and y^2 are not comparable. The Pareto optimal set $\mathcal{X}^*$ (in plain lines) is composed of two disconnected components, each being a continuous curve. Its

1. If $m = 2$. Otherwise, it is often called the *Pareto surface*

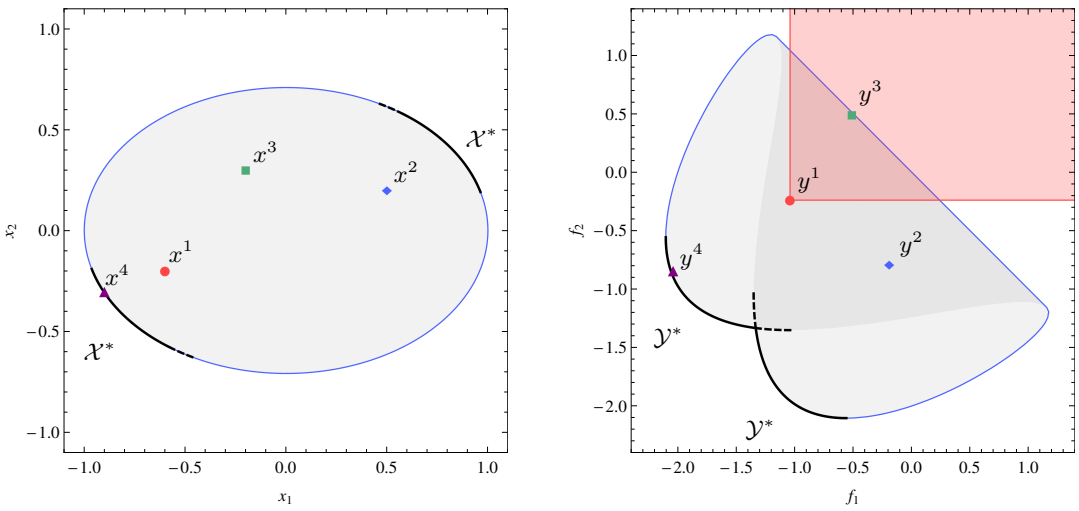


Figure 3.1 – Decision space (left) and objective space (right) of a biobjective problem from Example 3.2.1.

image $\mathcal{Y}^*$ is the Pareto front, and is connected. Local Pareto optimal solutions (in dashed lines) are present.

As in the single objective case, a NLMOO problem may contain several local optimal solutions (see Figure 3.1). These solutions define sets of locally Pareto optimal solutions.

Definition 3.2.3 (Local optimality)**.** A solution $x \in \mathcal{X}$ is *locally Pareto optimal* if there exists $\delta > 0$ such that x is Pareto optimal in $\mathcal{B}(x, \delta) \cap \mathcal{X}$.

In practice, and as in the single objective case, solutions that are computed by a solver can at best be asserted as locally Pareto optimal. However, global complete methods, see Section 3.3.3, ensures returning the global optima, among with possible local optima that could not be discarded.

Nevertheless, when the NLMOO problem is convex, i.e. when objectives are convex functions and $\mathcal{X}$ is a convex set, every local Pareto optimal solutions are also Pareto optimal.

Theorem 3.2.1 (Optimality in the convex case [85, Theorem 2.2.3])**.** *If the multiobjective problem* (3.1) *is convex, then any locally Pareto optimal solution is also (globally) Pareto optimal.*

Lower and upper bound on the Pareto optimal solutions give important informations that are well exploited by many optimization methods. In order to perform bounding, the notions of ideal, anti-ideal and nadir point have been introduced [25, 85].

Definition 3.2.4 (Ideal, anti-ideal and nadir points)**.** Suppose $\mathcal{Y}$ is bounded. The ideal point y^I of Problem (3.1) is defined as :

$$y_i^I = \min_{x \in \mathcal{X}} f_i(x) = \min_{y \in \mathcal{Y}} y_i \quad \forall i$$

Analogously, the anti-ideal point y^A is defined as :

$$y_i^A = \max_{x \in \mathcal{X}} f_i(x) = \max_{y \in \mathcal{Y}} y_i \quad \forall i$$

These two points respectively correspond to a lower bound (resp. upper bound) of all objective vectors $\mathcal{Y}$, and are generally infeasible.

The nadir point y^N of Problem (3.1) is defined as :

$$y^N = \max_{x \in \mathcal{X}^*} f_i(x) = \max_{y \in \mathcal{Y}^*} y_i$$

This point is an upper bound of all Pareto optimal objectives $\mathcal{Y}^*$.

The ideal and anti-ideal can be obtained by computing the individual minima $\hat{x}^i \in \mathcal{X}$ (maxima for the anti-ideal) of each objective f_i. The nadir is however more complicated to compute as it necessitate knowing $\mathcal{Y}^*$. Individual minima are not sufficient for the computation of the nadir, noticeably when an objective have several minima or when there are more than two objectives. Nevertheless, individual minima are sometimes used to estimate the nadir through the use of a payoff table [25, 85].

Example 3.2.2 – Consider the problem from Example 3.2.1, whose objective space is depicted on Figure 3.2(a). The individual minima $\hat{y}^1$ and $\hat{y}^2$ of each objective help constructing the ideal points y^I. Analogously, individual maxima are used to construct the anti-ideal y^A. In this example, the individual minima $\hat{y}^i$ are sufficient to build the nadir point y^N as they are unique and only two objectives are considered. On the other hand, changing the objectives to $f_1(x) = x_1^2$ and $f_2(x) = x_1 - x_2$ (the new objective space is illustrated on Figure 3.2(b)), then the nadir is more complicated to compute as there are several weakly Pareto-optimal solutions corresponding to the minimum of f_1, whose objective vectors have minimal value on f_1 but distinct value on f_2.

Nevertheless, these two points (ideal and nadir) are not good bounds as they are in general far away from the Pareto front due to the conflicting objectives. Better bounds can be obtained by the use of bound sets [26]. Bound sets generalize the concept of bounds to multiobjective problems.

Definition 3.2.5 (Bound sets)**.** Consider Problem (3.1). Let $\mathcal{Y}_L \subset \mathbb{R}^m$ be a set of points not dominating each other. This set is a *lower bound set* of $\mathcal{Y}^*$ if it satisfies :

$$\mathcal{Y}^* \subseteq \{y \geq y' : y' \in \mathcal{Y}_L\}$$

Similarly, let $\mathcal{Y}_U \subset \mathbb{R}^m$ be a set of points not dominating each other. This set is an *upper bound set* of $\mathcal{Y}^*$ if it satisfies :

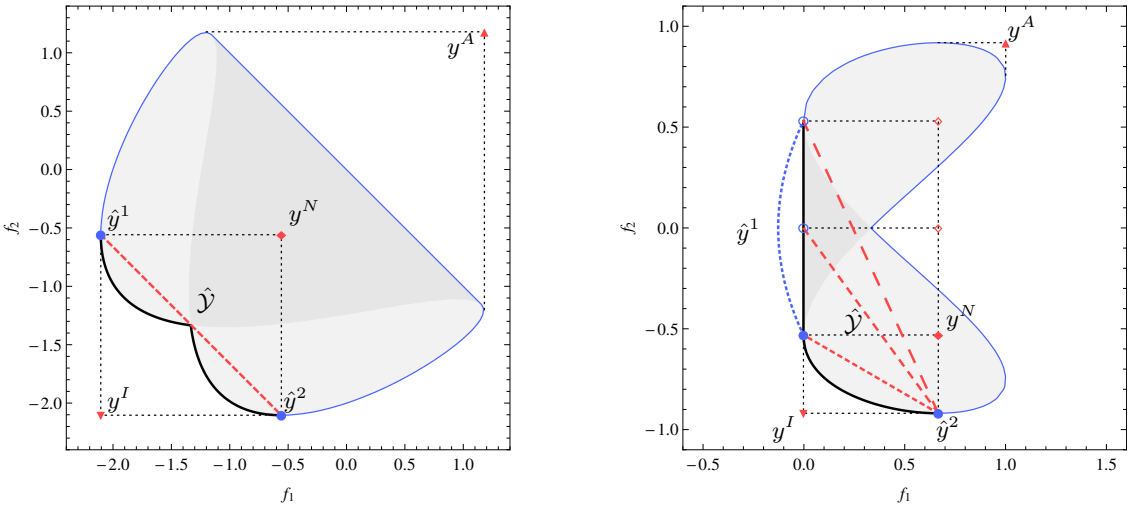
$$\mathcal{Y}^* \subseteq \mathbb{R}^m \backslash \text{int}\{y \geq y' : y' \in \mathcal{Y}_U\}$$

The sets $\{y^I\}$ and $\{y^N\}$ are respectively a lower and an upper bound set of $\mathcal{Y}^*$. In addition, any set of feasible objective points not dominating each other form an upper bound set of $\mathcal{Y}^*$.

In some solving techniques, an approximation of the shape of the Pareto front can be used efficiently. A usual approximation is the Convex Hull of Individual Minima (CHIM) taken from [19, 82].

Definition 3.2.6 (Convex Hull of Individual Minima)**.** Let $\hat{x}^i \in \mathcal{X}$ be an individual minima of the objective f_i of Problem (3.1). Let $\hat{y}^i$ be their corresponding objective vectors. The Convex Hull of Individual Minima is defined as :

$$\hat{\mathcal{Y}} = \{y : y = \sum_{i=1}^{m} w_i \hat{y}^i, \sum_{i=1}^{m} w_i = 1\} \tag{3.2}$$



(a) Straightforward computation

(b) Weakly nondominated solutions complicates the computation of the nadir and the CHIM

Figure 3.2 – Constructing ideal, anti-ideal and nadir points ; and the CHIM in dashed line.

Given weights w_i such that $\sum_i^m w_i = 1$, we also define points of the convex hull as :

$$\hat{\mathcal{Y}}(w) = \sum_{i=1}^{m} w_i \hat{y}^i \tag{3.3}$$

The CHIM is often referred as the utopia plane, and is used for approximating $\mathcal{Y}^*$. Again, as the individual minima can be weakly Pareto-optimal if they are not unique, it is difficult to obtain an optimal CHIM that is as close as possible to $\mathcal{Y}^*$. Moreover, the approximation of the nondominated set can be rough when considering more than two objectives [19].

Example 3.2.3 – Consider the problem from Example 3.2.1, whose objective space is depicted on Figure 3.2(a). As the individual minima of each objective are unique, the CHIM $\hat{\mathcal{Y}}$ is uniquely defined. On the other hand, the modified problem depicted on Figure 3.2(b) allows different construction of the CHIM due to the presence of weakly nondominated solutions. The optimal CHIM is obtained for the $\hat{y}^1$ having the smallest value on the second objective (point represented by a filled circle).

3.2.2 Optimality conditions

When the objectives and constraints of Problem (3.1) are differentiable [2], necessary and sufficient conditions for (local) optimality can be stated. These conditions are important as they can be used to help searching for, or asserting, (local) Pareto optimal solutions.

First, we present Fritz John first-order necessary conditions for optimality.

2. For nondifferentiable objectives and constraints, generalized gradients are used [85]

Theorem 3.2.2 (Fritz John necessary optimality condition [85, Theorem 3.1.1])**.** *Consider the NLMOO problem* (3.1)*. A necessary condition for a solution $x \in \mathcal{X}$ to be (locally) Pareto optimal is that there exist multipliers $0 \leq \lambda \in \mathbb{R}^m$, $0 \leq r \in \mathbb{R}^p$ and $s \in \mathbb{R}^q$ (λ, r, s not all 0) such that :*

$$F(x, \lambda, r, s) = \begin{pmatrix} \nabla f(x)\lambda + \nabla g(x)r + \nabla h(x)s \\ (\forall i = 1, \ldots, p)\ g_i(x)r_i \\ (\forall i = 1, \ldots, q)\ h_i(x) \end{pmatrix} = 0. \tag{3.4}$$

The multipliers can be normalized, e.g. using $\lambda^T\lambda + r^T r + s^T s = 1$, such that they are, in general, unique. These conditions are necessary for (local) Pareto optimality and weak Pareto optimality, but not sufficient. Solutions satisfying (3.4) are called *stationary*.

Definition 3.2.7 (Stationary solutions in multiobjective problems)**.** A solution $x \in \mathcal{X}$ is *stationary* for problem (3.1) if there exist multipliers $0 \leq \lambda \in \mathbb{R}^m$, $0 \leq r \in \mathbb{R}^p$ and $s \in \mathbb{R}^q$ (λ, r, s not all 0) solving system (3.4).

As in single-objective case, a stationary solution can be a (locally) Pareto optimal solution, or another extrema (saddle or maximal solution). Stationary solutions can be used to discriminate extrema from other non-stationary solutions. As stationary solutions are defined by a system of $n + m + p + q$ variables and $n + p + q + 1$ equations (counting an additional normalization equation), stationary solutions form a $m - 1$ dimensional manifold embedded in $\mathbb{R}^{n+m+p+q}$ under regularity assumptions, see Section 2.3.1 (p. 21). Note that regularity of the system (3.4) follows Definition 2.4.4 (p. 42), with the difference that in the multiobjective case, $\mathcal{L}$ denotes the matrix

$$\sum_{i=1}^{m} \nabla^2 f_i(x)\lambda_i + \sum_{i=1}^{p} \nabla^2 g_i(x) r_i + \sum_{i=1}^{q} \nabla^2 h_i(x) s_i \tag{3.5}$$

A stationary solution $x \in \mathcal{X}$ of a multiobjective problem is nonsingular if the system (3.4) derivative is full row rank at x and its induced multipliers λ, r and s.

Another well known first-order necessary conditions for optimality are the Karush-Kuhn-Tucker (KKT). These conditions differ from Fritz John as they do additionally suppose the solution to satisfy constraint qualifications (see Section 2.4). For the system 3.4, this also implies that if x satisfies constraint qualification, then the multipliers on the objectives λ are not all zero. This ensures that the objectives are actually involved in the first-order conditions. The KKT first-order optimality conditions can now be stated.

Theorem 3.2.3 (Karush-Kuhn-Tucker necessary optimality condition [85, Theorem 3.1.5])**.** *Consider the NLMOO problem* (3.1)*. Let $x \in \mathcal{X}$ be a solution satisfying LICQ. Then a necessary condition x to be (locally) Pareto optimal is that it is stationary. In addition, the multipliers λ of the objectives are not all* 0*.*

Another simpler normalization of the multipliers can be used for these conditions, as $\lambda \neq 0$. Indeed, it is enough to have $\sum_{i=1}^{m} \lambda_i = 1$. The assumption in KKT that a solution has to satisfy constraint qualification allows to derive a sufficient condition for (local) Pareto optimality in the convex case.

Theorem 3.2.4 (Sufficient optimality condition in the convex case [85, Theorem 3.1.7])**.** *Let a solution $x \in \mathcal{X}$ be stationary for which LICQ holds. If Problem* (3.1) *is convex, then x is (weakly) Pareto optimal.*

Therefore, in the case of convex problems, the KKT conditions are a proper characterization of the Pareto optimal solutions. For example, they can be used to derive additional constraints (complementarity constraints) in order to ease the solving of the problem [69]. In the general case, the optimality conditions have also been used to express the Pareto optimal solutions in function of the multipliers [3].

Exploiting the first-order conditions from the system (3.4) with numerical stability requires regularity, or nonsingularity, at its solutions. Similar to the single objective case, the stationarity of a solution is difficult to numerically assert when singular. Due to the specificities of the system (3.4), necessary and sufficient conditions for nonsingularity can be derived. The following theorem is an adaptation of Theorem 2.1 from [96], exhibiting nonsingularity conditions in parametric optimization problems.

Theorem 3.2.5 (Necessary and sufficient condition for nonsingularity)**.** *Let x be a (weakly) Pareto optimal solution to problem (3.1), and its set of active inequalities $\mathcal{A}(x)$, and vector of active constraints $\dot{g}$. Then let $0 \leq \lambda \in \mathbb{R}^m$, $0 \leq r \in \mathbb{R}^p$ and $s \in \mathbb{R}^q$ (λ, r, s not all 0) be the multipliers solving (3.4) with the additional normalization equation $\lambda^T\lambda + r^T r + s^T s - 1 = 0$. Further, note that $r_i = 0$ for each $i \notin \mathcal{A}(x)$. Denote the tangent space of objectives and (active) constraints by :*

$$\mathcal{T} = \{\tilde{x} \in \mathbb{R}^n : f'(x)\tilde{x} = 0, \dot{g}'(x)\tilde{x} = 0, h'(x)\tilde{x} = 0\} \tag{3.6}$$

Finally, denote by $\mathcal{L}$ the matrix defined as (3.5). Then, (x, λ, r, s) is nonsingular for (3.1) if and only if the following holds :

(i) The sets $\mathcal{A}(x)$ and $\{i : r_i > 0\}$ are equal (constraint complementarity) ;

(ii) LICQ holds at x (constraint qualification) ;

(iii) The matrix $\mathcal{L}$ is not singular on the space $\mathcal{T}$ (regularity on tangent space) [3].

Proof.

The proof is analogous to the one of Theorem 2.1 from [96]. Let $F(x, \lambda, r, s)$ be the system (3.4). The Jacobian $F'(x, \lambda, r, s)$ follows (2.44) (p. 42), with an additional row $(0, 2\lambda, 2r, 2s)$ corresponding to the derivative of the normalization function. Then, $F'(x, \lambda, r, s)$, is full row rank if and only if :

$$\underbrace{\begin{pmatrix} \mathcal{L} & r_1\nabla g_1(x) \dots r_p \nabla g_p(x) & \nabla h(x) & 0 \\ (\nabla f(x))^T & 0 & 0 & 2\lambda \\ (\nabla g(x))^T & G & 0 & 2r \\ (\nabla h(x))^T & 0 & 0 & 2s \end{pmatrix}}_{=(F'(x,\lambda,r,s))^T} u = 0 \Longrightarrow u = 0, \tag{3.7}$$

where G is a diagonal matrix with $G_{ii} = g_i(x)$. In the following, we will note $u = (\tilde{x}, \tilde{r}, \tilde{s}, \tilde{\lambda})$, $\tilde{x} \in \mathbb{R}^n$, $\tilde{r} \in \mathbb{R}^p$, $\tilde{s} \in \mathbb{R}^q$ and $\tilde{\lambda} \in \mathbb{R}$. Separating the active and inactive constraints, and noting that $g_i(x) = 0$ for any active constraint g_i and $r_i = 0$ for any inactive ones, the system of

3. The projection of $\mathcal{L}\tilde{x}$ onto $\mathcal{T}$ of any nonzero vector $\tilde{x} \in \mathcal{T}$ is not zero.

equations $(F'(x,\lambda,r,s))^T u = 0$ can be written as

$$\mathcal{L}\tilde{x} + \sum_{i=1}^{p} r_i \nabla g_i(x)\tilde{r}_i + \nabla h \tilde{s} = 0 \tag{3.8}$$

$$(\nabla f(x))^T \tilde{x} + 2\lambda\tilde{\lambda} = 0 \tag{3.9}$$

$$(\forall i \in \mathcal{A}(x))(\nabla g_i(x))^T \tilde{x} + 2r_i\tilde{\lambda} = 0 \tag{3.10}$$

$$(\forall i \notin \mathcal{A}(x))(\nabla g_i(x))^T \tilde{x} + g_i(x)\tilde{r}_i = 0 \tag{3.11}$$

$$(\nabla h(x))^T \tilde{x} + 2s\tilde{\lambda} = 0. \tag{3.12}$$

First, we prove that (i), (ii) and (iii) hold implies (x,λ,r,s) is nonsingular by contradiction. Assuming (i), (ii) and (iii) hold, we suppose (x,λ,r,s) is singular, i.e. $F'(x,\lambda,r,s)$ is not full row rank. Then there is a $u \neq 0$ such that $(F'(x,\lambda,r,s))^T u = 0$.

By performing a linear combination of equations (3.9), (3.10) and (3.12) with respective multipliers λ,r_i $(i \in \mathcal{A}(x))$ and s, we observe that

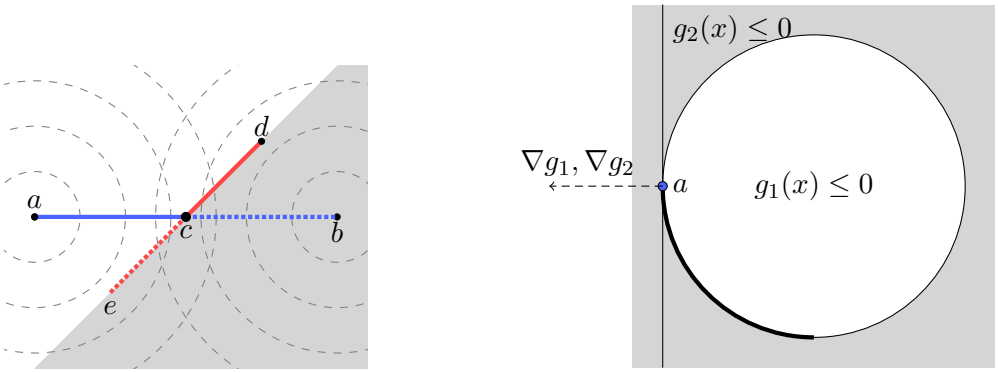
$$\begin{aligned}
&\lambda^T((\nabla f(x))^T\tilde{x} + 2\lambda\tilde{\lambda}) + \sum_{i\in\mathcal{A}(x)} r_i((\nabla g_i(x))^T\tilde{x} + 2r_i\tilde{\lambda}) \\
&+ s^T(\nabla h(x))^T\tilde{x} + 2s\tilde{\lambda}) \\
&= \tilde{x}^T \underbrace{(\nabla f(x)\lambda + \nabla \dot{g}(x) r + \nabla h(x) s)^T}_{=0 \text{ since } (x,\lambda,r,s) \text{ solves (3.4)}} + 2\tilde{\lambda} \underbrace{\left(\lambda^T\lambda + \sum_{i\in\mathcal{A}(x)} r_i^2 + s^T s\right)}_{=1 \text{ due to the normalization}} \\
&= 2\tilde{\lambda} = 0
\end{aligned}$$

This implies $\tilde{\lambda} = 0$. Hence, equations (3.9), (3.10) and (3.12) can be written respectively as $(\nabla f(x))^T\tilde{x} = 0$, $(\nabla g_i(x))^T\tilde{x} = 0 \quad (\forall i \in \mathcal{A}(x))$ and $(\nabla h(x))^T\tilde{x} = 0$ which are solved simultaneously only if $\tilde{x} \in \mathcal{T}$. From (3.8), we observe that :

- as $\mathcal{L}$ is nonsingular on $\mathcal{T}$, i.e. (iii) holds, and $\tilde{x} \in \mathcal{T}$, the projection of $\mathcal{L}\tilde{x}$ onto $\mathcal{T}$ is nonzero. Therefore, $\mathcal{L}\tilde{x} \notin \text{span}(\nabla\dot{g}(x), \nabla h(x))$, implying $\tilde{x} = 0$;
- as LICQ holds, i.e. (ii) is satisfied, there is no $\tilde{r}_i r_i, i \in \mathcal{A}(x)$ and $\tilde{s}$ all nonzero, such that $\sum_{i\in\mathcal{A}(x)} r_i \nabla g_i(x)\tilde{r}_i + \nabla h\tilde{s} = 0$.

Thus, the only solution to (3.8) is $\tilde{x} = 0$, $\tilde{r}_i r_i = 0$ for $i \in \mathcal{A}(x)$ and $\tilde{s} = 0$. Since constraint complementarity, i.e. condition (i), holds, then $r_i > 0, \forall i \in \mathcal{A}(x)$. Hence, $\tilde{r}_i = 0$ for $i \in \mathcal{A}(x)$. In addition, from (3.11) and due to $\tilde{x} = 0$ and $g_i(x) < 0$ for $i \notin \mathcal{A}(x)$, $\tilde{r}_i = 0$. This implies that the only vector satisfying all equations is $u = 0$, contradicting $F'(x\lambda, r, s)$ not full row rank. Therefore, conditions (i), (ii) and (iii) implies the nonsingularity of (x,λ,r,s).

To prove the reciprocity, we assume that (x,λ,r,s) is nonsingular and suppose one of the condition (i), (ii) or (iii) is violated. First, suppose (i) does not hold. Then there is a $k \in \mathcal{A}(x)$ such that $r_k = 0$ and $g_k(x) = 0$. Hence, the vector u with $\tilde{r}_k > 0$ and all other coordinates being 0 contradicts (3.7), therefore (i) must hold for (x,λ,r,s) to be nonsingular. Hence, we assume that (i) holds. Suppose now that (ii) is violated. Then, there exist vectors $\tilde{r} \neq 0$, with $\tilde{r}_i = 0$



(a) Violation of constraint complementarity

(b) Violation of LICQ

Figure 3.3 – Illustration of singular stationary solutions

for $i \notin \mathcal{A}(x)$, and $\tilde{s} \neq 0$ such that $\sum_{i=1}^{p} r_i \nabla g_i(x)\tilde{r}_i + \nabla h \tilde{s} = 0$. Therefore, picking $\tilde{x} = 0$ and $\tilde{\lambda} = 0$ solves equations (3.8) to (3.12), contradicting (3.7). Thus, (ii) must hold for (x, λ, r, s) to be nonsingular. Hence, we assume (i) and (ii) holds. Eventually, suppose that (iii) is violated. Then there is a nonzero vector $\tilde{x} \in \mathcal{T}$ such that $\mathcal{L}\tilde{x}$ is in the orthogonal space of $\mathcal{T}$, contained in $\text{span}(\nabla \dot{g}(x) | \nabla h(x))$ by definition. Thus, equation (3.8) can be solved with nonzero $(\tilde{x}, \tilde{r}, \tilde{s})$. The other equations are solved by picking $\tilde{\lambda} = 0$ and $\tilde{r}_i = 0$, $i \notin \mathcal{A}(x)$, contradicting (3.7). Thus, (iii) must hold for (x, λ, r, s) to be nonsingular. Therefore, (x, λ, r, s) is nonsingular implies conditions (i), (ii) and (iii), hence completing the proof.

□

We can note that singularities due to a violation of the first condition, i.e. complementarity of active constraints, are easily triggered when inequality constraints are considered. Noticeably for linear biobjective problems, the parametric simplex [25] implicitly deals with these critical points as they trigger changes in the set of basic variables. Such a singularity corresponds to a local change of the set of active constraints around a stationary solution. The following example illustrates typical violation of the conditions of Theorem 3.2.5 leading to singular stationary solutions.

Example 3.2.4 – A solution not satisfying (i), i.e. complementarity of the active constraints, is a solution at the junction of two manifolds of stationary solutions, different from their set of active constraints. For example, let a two variable biobjective problem be defined by $f_1(x) = (x_1 + 1)^2 + x_2^2$, $f_2(x) = (x_1 - 1)^2 + x_2^2$ and inequality constraint $g(x) = x_1 - x_2 \leq 0$, whose Pareto set is illustrated in plain lines on Figure 3.3(a). The stationary solution c with $x = (0, 0)$ and with multipliers $(\lambda_1, \lambda_2, r) = (\sqrt{2}/2, \sqrt{2}/2, 0)$ is singular as the constraint g is active and its multiplier r is zero. Due to the constraint $g(x)r = 0$ in (3.4), there are two paths of stationary solutions : one where $r = 0$ (from a to b), i.e. g is inactive, and the other one where $g(x) = 0$ (from d to e), i.e. g is active. The point c is at the bifurcation of these two paths with different sets of active constraints. Note that due to the positive sign of r and the constraint $g(x) \leq 0$, the paths from c to b and from c to e do not satisfy the first-order conditions.

A solution not satisfying (ii), i.e. not satisfying LICQ, occurs when constraints are locally redundant, or when a constraint has a null gradient at a solution. For example, let the two variables biobjective problem be defined by $f_1(x) = x_1$, $f_2(x) = x_2$ with inequality constraints $g_1(x) = x_1^2 + x_2^2 - 1 \leq 0$ and $g_2(x) = -x_1 \leq 1$ whose Pareto set is illustrated on Figure 3.3(b). Clearly at the Pareto optimal solution $x = (-1, 0)$, the gradient of the constraints g_1 and g_2 are collinear, i.e. the constraints are redundant at x. Note that the satisfaction of g_1 enforces the satisfaction of g_2, hence g_2 is globally redundant. In order to avoid violation of (ii), removing all globally redundant constraints is necessary.

A solution violating (iii), i.e. singular on the tangent space of objectives and constraints, is a consequence of a particular structure of the problem at this solution. For example let the three variables unconstrained biobjective problem be defined by $f_1(x) = x_1 + (x_2 - \alpha(x_1))^2$ and $f_2(x) = 1 - x_1^2 + (x_3 - \beta(x_1))^2$, where α and β are two univariate functions. Consider the stationary solution a with $x = (0, \alpha(x_1), \beta(x_1))$ and multipliers $\lambda = (0, 1)$. By construction,

$$\mathcal{T} = \text{span} \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \mathcal{L} = \begin{pmatrix} -2 + 2(\beta'(x_1))^2 & 0 & -2\beta'(x_1) \\ 0 & 0 & 0 \\ -2\beta'(x_1) & 0 & 2 \end{pmatrix}.$$

We can see that the vector $\tilde{x} = (0, 1, 0)^T$ in $\mathcal{T}$ is singular for $\mathcal{L}$ as $\mathcal{L}\tilde{x} = 0$. Thus, a violates (iii) and is singular for the system (3.4).

Many approaches from the literature in single objective optimization solves first-order systems, using Newton-like methods, such as interior point methods (e.g. [134]) or Sequential Quadratic Programming (SQP) methods (e.g. [10]). In multiobjective optimization, we note the existence of a Newton method [33] for unconstrained problems, not based on a single-objective reformulation, and continuation methods (see Section 3.3.2). We later focus on such approaches as exploiting the system (3.4) of first-order conditions is important for the rigorous solving of multiobjective problems.

3.3 Solving multiobjective problems

Solving the NLMOO Problem (3.1) means searching its Pareto optimal solutions $\mathcal{X}^*$. When preferences on the objectives are known or when the decision maker is implied during the solving process, then only a subset of Pareto optimal solutions are sought. Otherwise, in what it is called a posteriori decision process, one is looking for all the solutions $\mathcal{X}^*$. In this thesis, we are interested in such processes. We can note that as $\mathcal{X}^*$ is in general uncountable (as $\mathcal{X}^*$ consists of manifolds), only a discrete approximation of $\mathcal{X}^*$ is in general computed by solving methods. In the following, different approaches that aims at achieving such a result are presented.

Note that the literature is prolific and many methods that are not important enough for the understanding of this thesis are not described here. For the interested reader, we refer to the following surveys compiling many different solving methods [113, 78].

3.3.1 Scalarization methods

When one is faced to a NLMOO problem as (3.1), it can be interesting to use well known and efficient single objective solver. Hence, a popular approach when tackling a NLMOO problem is *scalarization*. Scalarization is a process that transforms a NLMOO problem into a parametric

single-objective problem. This single-objective problem is defined such that fixing the parameters to appropriate values yields to a specific solution from $\mathcal{X}^*$. Thus, it is expected that solutions obtained from different parameter values approximate sufficiently well the set $\mathcal{X}^*$ of Pareto optimal solutions.

Well-known scalarization methods are the Weighted-Sum (WS) and the ε-constraint (εC), see [25, 85]. The WS scalarization is defined as follows

$$\left[\begin{array}{ll} \min & \sum_{i=1}^{m} \lambda_i f_i(x) \\ \text{s.t} & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in \mathbb{R}^n \end{array}\right], \tag{WS}$$

with $\sum_{i=1}^{m} \lambda_i = 1$, $\lambda_i \geq 0$. The WS introduces m parameters although, due to their normalization, one can be fixed with respect to the others leading to $m-1$ parameters. This scalarization consists of optimizing a linear aggregation of the objectives, and is very popular as its principle is easy to understand and to use. However, it suffers from an important drawback : it is not able to determine solutions whose objective values are outside the convex hull of $\mathcal{Y}$. Hence, WS is best suited for convex problems. Solutions to (WS) are always weakly Pareto optimal for Problem (3.1), and Pareto optimal if the weights λ are strictly positive [25, 85]. This scalarization is illustrated on Figure 3.4(a) on a biobjective problem.

The εC scalarization is based on a different principle. The idea is to optimize one objective while the others are transformed into constraints.

$$\left[\begin{array}{lll} \min & f_m(x) & \\ \text{s.t} & f_i(x) \leq \varepsilon_i & (i = 1, \ldots, m-1) \\ & g(x) \leq 0 & \\ & h(x) = 0 & \\ & x \in \mathbb{R}^n & \end{array}\right] \tag{εC}$$

Such additional parameters ε_i are then upper bounds on these additional constraint. This scalarization ables to reach any Pareto optimal solution regardless of the nonconvexity of the Pareto front. On the other hand, having additional constraints tend to increase the difficulty of solving the problem. One can note that any optimal solution to (εC) is a weakly Pareto optimal solution to (3.1) (Pareto optimal if the solution is unique). Reciprocally, any Pareto optimal solution to (3.1) is an optimal solution to (εC), given a specific value of ε_i [25, 85]. However, two different parameter values possibly yield to the same solutions, implying a smart selection of the parameter values in order to avoid redundant computations. The εC scalarization is illustrated on Figure 3.4(b) on a biobjective problem. The WS and εC scalarization can be used in combination for generating all Pareto optimal solutions within an hybrid formulation, e.g the approach from Kim and De Weck [62].

Some scalarization methods, known as weighted metrics [85], make use of a reference point. For example, we present here the Weighted Tchebycheff (WT) scalarization :

$$\left[\begin{array}{ll} \min & (\sum_{i=1}^{m} \lambda_i |f_i(x) - y_i^I|^p)^{\frac{1}{p}} \\ \text{s.t} & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in \mathbb{R}^n \end{array}\right], \tag{WT}$$

with $\sum_{i=1}^{m} \lambda_i = 1$, $\lambda_i \geq 0$ and $1 \leq p \leq \infty$. This problem consists of minimizing the distance to the ideal point, using a p-norm, in the objective space. Varying the weights helps finding different Pareto optimal solutions. However, in order to reach all Pareto optimal solution, p has to be taken to a high value [4]. Indeed, one can note that $p = 1$ corresponds to the WS scalarization. Higher value of p adapts to higher level of nonconvexity of the Pareto front. This scalarization requires the computation of the ideal point, although an overapproximation can be used. This scalarization is illustrated on Figure 3.4(c), using different p values.

As previously said, one of the goal of scalarization techniques is to find a representative discrete approximation of the set of Pareto optimal solutions, i.e. their objective values must be well distributed along $\mathcal{Y}^*$. This distribution can be difficult to obtain as the extra parameters of the scalarization may not be appropriate for describing the Pareto front of a given problem. For example, the distance in the objective space between two Pareto optimal solutions may be large compared to the distance between the parameters values used to find them. In order to have a better correlation between parameter values and the induced solutions in the objective space, several other scalarization techniques have been studied in the literature. Noticeably, many of them use parameters related to the CHIM, yielding them to better describes the Pareto front [5].

The Normal Boundary Intersection (NBI) from Das and Dennis [19] defines such a scalarization. The single-objective problem is defined as follows :

$$\left[\begin{array}{ll} \max & t \\ \text{s.t} & \hat{\mathcal{Y}}(\lambda) + tu = f(x) \\ & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in \mathbb{R}^n \end{array}\right], \tag{NBI}$$

where u is a vector quasi normal to the CHIM $\hat{\mathcal{Y}}$, $\hat{\mathcal{Y}}(\lambda)$ follows (3.3) with $\sum_{i=1}^{m} \lambda_i = 1$ and t is an additional variable. This scalarization can be seen as the problem of finding the boundary point of $\mathcal{Y}$ the furthest from a point of the CHIM along a direction u. This is illustrated on Figure 3.4(d). It is expected with such a scalarization that an evenly spread set of parameters λ will lead to an evenly spread set of Pareto points. One drawback of the NBI scalarization is that although it can capture any Pareto optimal solutions, non Pareto optimal solutions can also be obtained. Moreover, the additional equality constraints may be difficult to handle numerically.

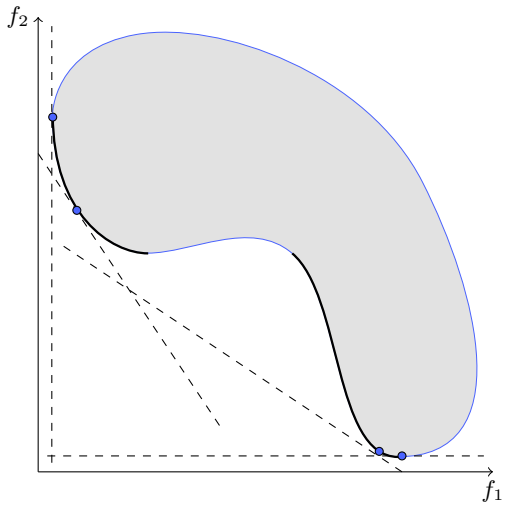
These drawbacks have been partially treated in the Normal Constraint (NC) scalarization from Messac et al. [82, 84]. Instead of adding equality constraints as in NBI, the NC scalarization works in an εC fashion : one objective is minimized while the others are transformed into constraints. The difference with εC is that these constraints are constructed with respect to the CHIM :

$$\left[\begin{array}{ll} \min & f_m(x) \\ \text{s.t} & (f(\hat{x}^m) - f(\hat{x}^i))^T (f(x) - \hat{\mathcal{Y}}(\lambda)) \leq 0, \forall i = 1, \ldots, m-1 \\ & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in \mathbb{R}^n \end{array}\right], \tag{NC}$$

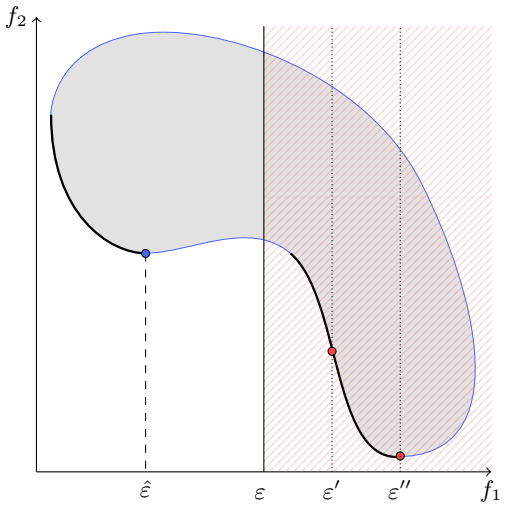
where $\sum_{i=1}^{m} \lambda_i = 1$, $\lambda \geq 0$. Under certain circumstances, due to the shape of $\mathcal{Y}$, the NC scalarization may still generate dominated points [82]. On the other hand, it has a more stable behavior

4. $p = \infty$ ensures all Pareto optimal solutions can be reached, but transform the problem into a min max one.

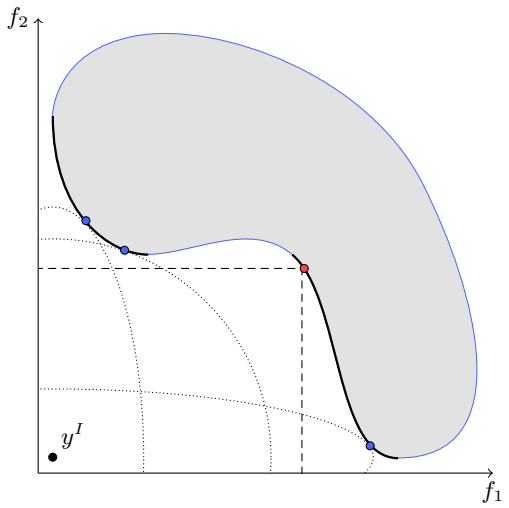
5. With the additional difficulty of computing an accurate CHIM beforehand, i.e. computing individual minima that ideally must not be weakly Pareto optimal



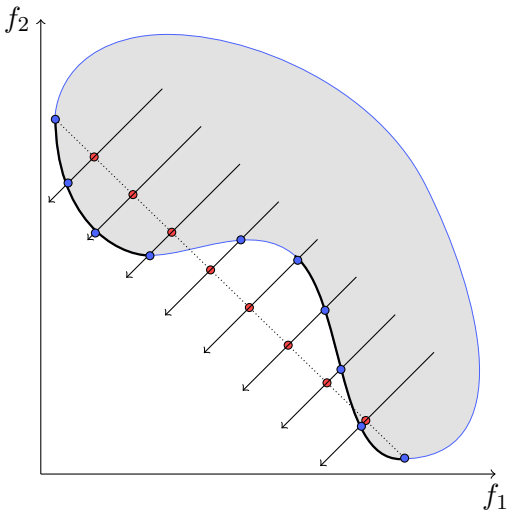
(a) WS Scalarization. Different directions of the linear aggregation of the objectives are displayed.



(b) εC scalarization. Different parameter values (e.g. ε', ε'') yield to distinct points. With at paremeter between $\hat{\varepsilon}$ and ϵ yield the same to solution.



(c) WT scalarization. Minimizing the weighted distance to the ideal using a 2-norm with different weights (in dotted line), and using ∞-norm (in dashed lines).



(d) NBI scalarization. Solutions are found along the quasi-normal to the utopia plane.

Figure 3.4 – Figure depicting different scalarization.

than NBI. Both methods however share a drawback when more than two objectives are considered [19, 82]. As the CHIM is defined by a convex combination of m points, it can not capture the extreme points outside the edge of the CHIM [6]. Improvements and discussions about these two scalarization methods can be found in [73, 88, 114].

All the scalarization methods presented previously introduce a single parameter [7] which corresponds to the dimension of the Pareto frontier of biobjective problems. Extra parameters can be used to direct more accurately the solving process. For example the Directed Search Domain (DSD) [28] is a method that is inspired by preceding work on other scalarizing methods known as Physical Programming [83, 130]. It proposes to orient cones along the CHIM that identify a specific region of the Pareto front. Orienting these cones allows to reach any Pareto optimal solutions (even in the case of more than two objectives) and may avoid redundant computations. Setting these parameters in order to reach a specific solution on the Pareto front is however not straightforward.

Scalarization methods can be seen as a way of integrating the decision maker preferences before or during the resolution of (3.1), by fixing the parameters accordingly. In a posteriori decision making process, solving (3.1) through a scalarized formulation is clearly related to solving a parametric optimization problem. In a context of finding the globally Pareto optimal solutions, it is hence required to solve globally several scalarized problems for different parameter values. In practice however, we observe that 1) global solvers are computationally expensive and 2) shifting slightly the parameters of a scalarized problem generally tends to correspond to a slight change of its optimal solution. Therefore, a popular use of scalarized problems is to use as initial solution of a local solver a previously computed optimal solution in order to solve each subsequent scalarized problem, i.e. by shifting appropriately the parameter values, see e.g. [19, 95]. This practical approach raises some difficulties :

1. it is no more global ;
2. it may be difficult, given the scalarization, to find an appropriate shift of parameter values so as to achieve an evenly spread set of nondominated points ;
3. its performances strongly depend on the selected scalarization, i.e. the parameterization of the Pareto front that is used (and fixed).

The former limitation is theoretical as many practical applications do not require finding all global Pareto optimal solutions. Still, some globality can be achieved if this continuation by scalarization is coupled with a global search algorithm, see e.g. [123, 106]. The second difficulty has been tackled with some success, for example in [27, 95]. For the latter difficulty, selecting the most appropriate scalarization for a given NLMOO problem is, to our knowledge, not easy to do as it depends on the problem structure which is not known beforehand.

A look at the literature on parametric optimization reveals another class of approaches, which are close in spirit to this usage of scalarization. Indeed, some approaches in parametric optimization consists in applying numerical continuation techniques [1] (see also Section 2.3.1) to a characterization of the optimal solutions, namely the KKT optimality conditions represented by the system (3.4) [99, 101]. These approaches have mostly been studied for single parameter problems, which can be assimilated to biobjective optimization, as remarked in [100]. In this work, the authors proposed a continuation method based on the WS scalarization of (3.1), whose first-order system is similar to the one of the biobjective problem. This method starts from a (local) Pareto

6. The points that are Pareto optimal with respect to each pair of objectives
7. More precisely, the number of parameters can be reduced to one

optimal solutions satisfying (3.4). It then tracks locally connected solutions satisfying (3.4), which are potential locally Pareto optimal solutions. This process is numerically sensitive to singularities, for example as consequence of loss of active constraints complementarity (see Theorem 3.2.5 and [96]). Therefore, the method also implements a strategy for detecting change inside the set of active constraints, a required feature to deal with general inequality constraints. The other possible source of singularities can still occur, although it is possible to detect and handle them approximately, see [76]. Moreover, this method uses second order optimality conditions to track (local) minima of the WS scalarization. Hence, it cannot track optimal solutions that are on the non-convex parts of the Pareto front. The advantage of this approach with respect to classical use of scalarization is that it does not fix a priori the parameter used for continuation : the parameters λ introduced by the WS scalarization are in fact considered as variables. Hence, it adapts better and automatically to the shape of the (local) Pareto optimal solutions as seen in Section 2.3.1 (p. 21).

More recently, Hillermeier has proposed a general framework for numerical continuation in NLMOO [53, 54]. This work has lead to a brand new set of methods for solving, locally, NLMOO problems.

3.3.2 Continuation methods

From the work on continuation methods in parametric optimization, Hillermeier [53, 54] has proposed a general scheme of continuation approaches for nonlinear multiobjective optimization. Although it does not consider inequality constraints, any number of objectives can be taken into account. The process consists in applying a continuation method on the system (3.4) of KKT conditions, starting from an initial Pareto optimal solution. Uniqueness of the multipliers, which is required for efficiently solving (3.4), is obtained by adding the normalization equation $\sum_i \lambda_i = 1$ to the system (3.4). In the biobjective case, this continuation technique can be viewed as an arc-length continuation. It does not suffer from possible turning points. Hence, the continuation process in [54] is a local technique that takes full benefits of this first-order characterization of Pareto optimal solutions, but it requires an initial one. Note that [54] also requires the system to be regular, in particular constraint qualification must hold along the manifold.

One of the first offspring of this scheme is the recovering algorithm of Schütze et al. [120]. Continuation is used as a repair operator inside a global subdivision method [122, 118]. It tracks solutions inside a decomposition of the search space into hyper-rectangles. Hence, the step of continuation is adapted with respect to the mesh of this decomposition. The same continuation technique has been applied within another global search algorithm, namely a particle swarm optimizer [119], see Section 3.3.3. As for Hillermeier [54], this continuation technique has not been applied to problems with inequality constraints.

Harada et al. in [50] have also proposed to combine a global search, namely a genetic algorithm (see Section 3.3.3), with a continuation method. This continuation process, called Pareto Path Following (PPF), is a predictor-corrector algorithm. The predictor step constructs a new solution using the gradient of the objectives as an initial direction of continuation, which is repaired by the Pareto-descent repair operator [51] and a gradient projection method for handling constraints. The corrector step uses the Pareto-descent local search [49]. Although the three objectives and the constrained case are discussed in [50], only one experiment on a biobjective bound-constrained problem is shown.

As stated in [1], solving a system as (3.4) by continuation is equivalent to solving a specific system of ordinary differential equations. Pereyra [94] has proposed to transform biobjective un-

constrained convex problems into 2-point boundary value problems. In that case, the first-order optimality conditions can be defined as :

$$(1-\lambda)\nabla f_1(x) + \lambda \nabla f_2(x) = 0, \tag{3.13}$$

with $\lambda \in [0,1]$. The problem consists in finding the parametric curve $x(\lambda) : [0,1] \to \mathbb{R}^n$ such that $(x(\lambda), \lambda)$ satisfies (3.13) for all λ. Since only convex problems are considered, this curve represents the Pareto optimal solutions. It is computed as the solution curve of the following two-points boundary-value problem :

$$\dot{x}(\lambda) = -H_x^{-1}(x) H_\lambda(x),\ x(0) = \hat{x}^1,\ x(1) = \hat{x}^2, \tag{3.14}$$

where $H_x(x) = (1-\lambda)\nabla^2 f_1(x) + \lambda \nabla^2 f_2(x)$ and $H_\lambda(x) = -\nabla f_1(x) + \nabla f_2(x)$. As for the NBI scalarization, solving (3.14) requires to find the minimum of each objective $\hat{x}^1$ and $\hat{x}^2$. The proposed continuation process is a predictor-corrector. The prediction increments the parameter λ by δ_λ, and uses the previous corrected solutions as an initial guess. The correction step is the application of the Newton method on the system (3.13). This process produces a set of points that covers the Pareto-optimal curve. As the time step parameter is λ, the technique suffers the same drawback as the WS, hence the method is restricted to convex problems. In addition to this continuation method, Pereyra [94] has proposed additional constraints ensuring an homogeneous coverage, in the decision or objective space. A parallel algorithm inspired by the continuation process is also proposed, and shows similarity with NBI. Finally, although the three objectives case is discussed, all these techniques are experimentally assessed only on biobjective problems.

Another method based on solving a system of differential equations by continuation is the method by Potschka et al. [98]. This method is based on the NBI or NC scalarization of (3.1) in the biobjective case. More precisely, the considered scalarized problem is as follows :

$$\begin{bmatrix} \min & f_2(x) \\ \text{s.t} & g(x) \leq 0 \\ & h(x) = 0 \\ & u^T(f(x) - \hat{f}(v)) = 0 \end{bmatrix}, \tag{3.15}$$

with $\hat{f}(v) = v f(\hat{x}^1) + (1-v) f(\hat{x}^2)$, i.e. a point of the CHIM, and $u = f(\hat{x}^1) - f(\hat{x}^2)$. The parameter v varies within $[0,1]$. As in [100], an active set of constraints $\mathcal{A}$ is used to handle the change of activity of inequalities. Therefore, active inequalities are considered as equalities, while inactive ones are not considered. Let $h_e(x,v)$ be the additional equality $u^T(f(x) - \hat{f}(v))$. Let $L(x, r_\mathcal{A}, s, s_e, v) = f_2(x) + r_\mathcal{A} g_\mathcal{A}(x) + s h(x) + s_e h_e(x,v)$ the Lagrangian of (3.15) considering the active set $\mathcal{A}$, with $g_\mathcal{A}$ and $r_\mathcal{A}$ being the vector of active constraints and multipliers, leading to the following first-order conditions :

$$F(x, r_\mathcal{A}, s, s_e, v) = \begin{pmatrix} \nabla_x L(x, r_\mathcal{A}, s, s_e, v) \\ g_\mathcal{A}(x) \\ h(x) \\ h_e(x,v) \end{pmatrix} = 0. \tag{3.16}$$

Denoting $y = (x, r_\mathcal{A}, s, s_e)$, Potschka et al. [98] have proposed to determine the curve of solutions of (3.16), parameterized by v, solving the following ordinary differential equation :

$$\dot{y}(v) = (\nabla_y F(y,v))^{-1} (\nabla_v F(y,v)). \tag{3.17}$$

The system is solved by continuation using an integrator applied on the parameter v. The set of active constraints $\mathcal{A}$ is managed such that the process builds feasible and (locally) Pareto optimal vectors by using the event detection of the integrator which triggers changes of the active set on conditions similar to [100] (i.e. studying the change of the sign of a constraint or of its multiplier). When a change is detected at a solution x, the active set is updated according to the constraints that are active at x. As in [94], this continuation technique moves along a fixed parameter, here v. Hence, it can stop tracking the curve at turning points when the front is perpendicular to the line connecting $f(\hat{x}^1)$ and $f(\hat{x}^2)$.

One limitation of continuation methods is that they usually require to build the tangent space of the tracked manifold at each iterate. From a system of n equations $F(x) = 0$, the complexity of such an operation is $O(n^3)$. Although the complexity is polynomial, it tends to be very expensive to build tangent space of problems involving thousands of variables. Ringkamp et al. [107] have proposed a tangent space approximation technique in the context of continuation for multiobjective optimization. This technique reduces the complexity of the tangent computation to $O(n^2)$. This approximation method can be applied for any problems where continuation can be used.

Other continuation approaches to multiobjective optimization not based on solving (3.4) also exist. All the continuation methods presented before require the use of the gradient (or Hessian) of objectives and constraints. In the case where gradients are not available, Schütze et al. [121] have proposed to use the descent method HCS from Lara et al. [67] in order to perform the continuation on unconstrained multiobjective problems. This technique is based on the observation that, without constraints, there are less chances in finding a direction that improves all objectives at a solution x if this solution gets closer to (locally) Pareto optimal solutions [13]. On the contrary, the chances of finding a direction of trade-off (i.e. that improves one objective and deteriorates the other) increases. The idea in [121] is to build a predictor-corrector technique which uses the HCS to build trade-off directions for predicting new solutions along the Pareto optimal curve, and to build improving directions for correcting the predicates. The HCS used in [121] constructs these two kind of directions by using gradient approximation techniques. These approximations are also used to determine the length of the continuation step. Although this can, in theory, be used to solve unconstrained problems with any number of objectives, only the biobjective case is experimentally assessed.

Finally, Lovison [74] has studied the global characterization of Pareto optimal manifolds, through first-order conditions for unconstrained problems. This characterization consists of a piecewise linear approximation of these manifolds using a Delaunay tessellation of the search space, i.e. a decomposition of the search space in simplices. Piecewise linear continuation is used to determine the simplices cut by the manifold. In addition, a process shrinking simplices around the manifold is proposed. The treated problems are unconstrained, but the methods can be adapted to deal with equalities. Any number of objectives can be considered. The technique has some limitations. First, the complexity of the Delaunay tessellation is exponential with the number of variables, hence the process is limited in problem size. Second, the initial tessellation must be thin enough to guarantee that each connected component of the Pareto-optimal manifold is cut by at least one simplex. Some issues have been partially answered by Lovison [75]. The author indeed proposes a simpler globally convergent version of the algorithm using a decomposition of the search space by regular (equilateral) triangles, dedicated to solve unconstrained problems with 2-variables and 2-objectives. The method shares some similarity with [120] : the former captures the manifold of Pareto optimal solutions by means of regular triangles, the latter by means of hyper-rectangles.

We can see there is a growing interest in the recent literature on continuation methods for solving NLMOO problems. These techniques appear to be really efficient and have a robust behavior for tracking (locally) Pareto optimal solutions. Hybridization with global search algorithms have been proposed and are appealing. Most of these approaches are however only dedicated to biobjective and unconstrained problems [8].

3.3.3 Global search methods

Global search methods aim at determining the set of globally Pareto optimal solutions. This is done though a global exploration of the space $\mathcal{X}$. In this thesis, we consider as a global search a method that both :

- Constructs an (approximate) representation of the Pareto optimal solutions by a discrete set of solutions, or an enclosure of the Pareto optimal set and front ;
- Explores the whole feasible space either stochastically (possibly guaranteeing complete exploration asymptotically) or deterministically (complete exploration).

Among the most popular approaches in the literature are population based metaheuristics [34, 16], like Genetic Algorithm (GA), Particle Swarm Optimization (PSO), etc. These methods apply stochastic breeding procedures on a population of solutions with a maximal size. These procedures are often inspired by nature : evolutionary concepts for GA, movement of flock of birds or fishes for PSO. Well known implementations of population based metaheuristics are for example NSGA-II [20], MOEA/D [72] or SMPSO [89]. These methods aim at evolving/directing each generation of the population of solutions towards the Pareto optimal set. Practical advantages of these methods are their applicability to a wide variety of problems and their intuitive concepts. They remain one of the most popular answer when one is faced to a NLMOO problem. On the other hand, they tend to converge poorly to the Pareto optimal set as the different operators focus in priority on the diversification of the population, i.e. its spread along the Pareto front. Thus, advanced population based metaheuristics are in general coupled with local search techniques to help pushing the solutions towards (locally) Pareto optimal solutions [16, 124]. Another step forward can be made by using continuation techniques as local search. Indeed, continuation can be used to efficiently recover large portions of connected (locally) Pareto optimal solutions. This lets the metaheuristic focus on searching all the disconnected parts of the Pareto optimal solutions, and not to focus on the spread along the front, as in [50]. Noticeably in the biobjective case, Harada et al. [50] have proposed to use curves of (locally) Pareto optimal solutions as the atomic element of the population [9]. Thus, breeding elements of the population aims at reaching undiscovered curves of Pareto optimal solutions.

Another drawback of population based metaheuristics is that as the global search is stochastic and strongly depends on the initial population, there is no guarantee that the solutions returned once the process stops after a finite time are Pareto optimal. Instead, convergence to the global Pareto optimal front can be guaranteed by complete methods with deterministic exploration. Branch & Bound are a class of methods that do such an exploration, see Section 2.4.

In the context of unconstrained biobjective problems, Scholz [117] has proposed the Small Cube Big Cube method, which is actually a general B&B algorithm. Bounds associated to each sub-problems are computed as approximation of local ideal (as lower bound) and local anti-ideal

8. Precisely, problems without inequality constraints
9. Curves are represented by a discrete set of solutions.

(as upper bound). Solutions are instantiated during the search and allow removing sub-problems using dominance. A sub-problem is considered terminal once the distance between its lower and upper bounds drops below a prescribed precision. The computation of bounds is left to the user. In [117], the method is applied on a set of location problems for which there are known techniques to compute lower and upper bounds for each objective.

Another method that resembles a B&B is the sub-division algorithm from Schütze et al. [122, 118]. The principle of this method is to decompose the search space into hyper-rectangles. Each iteration operates a thinner mesh of this decomposition. An hyper-rectangle is kept if it contains at least one Pareto optimal solution. Detecting the presence of Pareto optimal solutions depends on the use of local searches [10] : descent algorithms or evolutionary techniques [122]. These different local searches may fail to detect Pareto optimal solutions. Hence, an hyper-rectangle can be removed while it in fact contains Pareto optimal solutions. Thus, this global search is not complete in practice. Nevertheless, a repairing operator, for recovering missing hyper-rectangles, has been proposed, in particular an operator based on continuation techniques [120]. Problems considered by the sub-division techniques contain only bound-constraints.

In NLMOO, few interval based B&B methods have been derived. The basic principles of interval B&B remain the same as in the single objective case presented in Section 2.4 (p. 41). The different operations, used to narrow domains of sub-problems or to decide whether a sub-problem can be discarded, have to be redefined for considering multiple objectives. Lower and upper bounds change their nature, i.e. bounding sets, see [26] and Definition 3.2.5, have to be considered. We can note however that, to our knowledge, there is no proposed B&B in the literature that uses proper lower bound sets : the lower bound of a sub-problem is an over approximation of a local ideal point. Finally, note that it is convenient for multiobjective interval B&B to associate an objective box $\boldsymbol{y}$ to each sub-problem. This objective box stores the objective values associated to a sub-problem and can be used for splitting (see below).

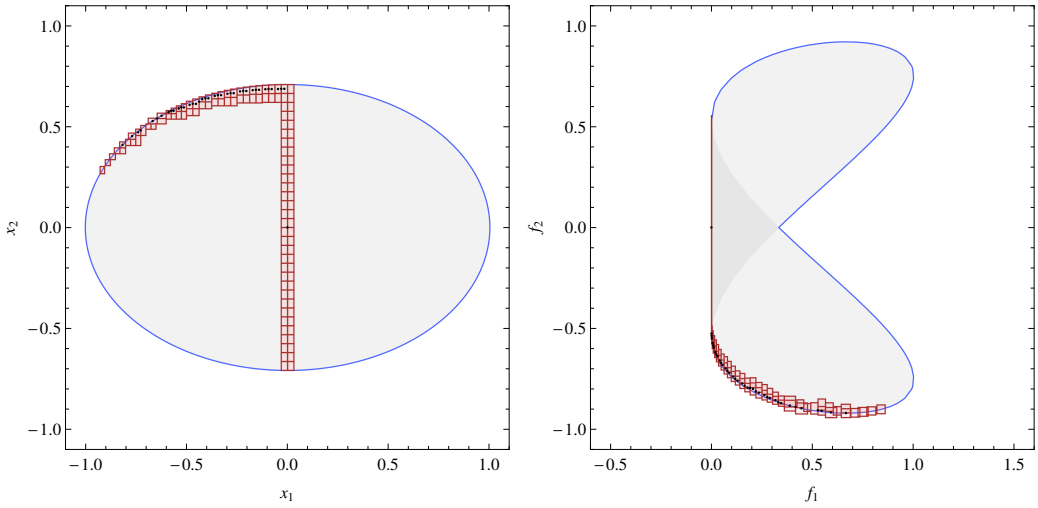
We can distinguish two types of multiobjective interval-based B&B in the literature : direct methods [109, 32] and inverse methods [4, 66]. The difference between the two lies in how sub-problems are generated : the former decompose variable domains (they produce a paving in the decision space) whereas the latter decompose objective domains (they produce a paving in the objective space). Direct methods are the natural extension to multiobjective problems of single objective B&B. Inverse methods see the solving process more as supporting the decision from the objective space. Direct and inverse solving are illustrated on Figure 3.5(a) and 3.5(b) respectively.

Example 3.3.1 – Consider the modified biobjective problem from Example 3.2.2 (p. 56). We observe the differences between direct and inverse approaches on Figure 3.5. The points on these figures represent feasible solutions (and their objective values) found during the search.

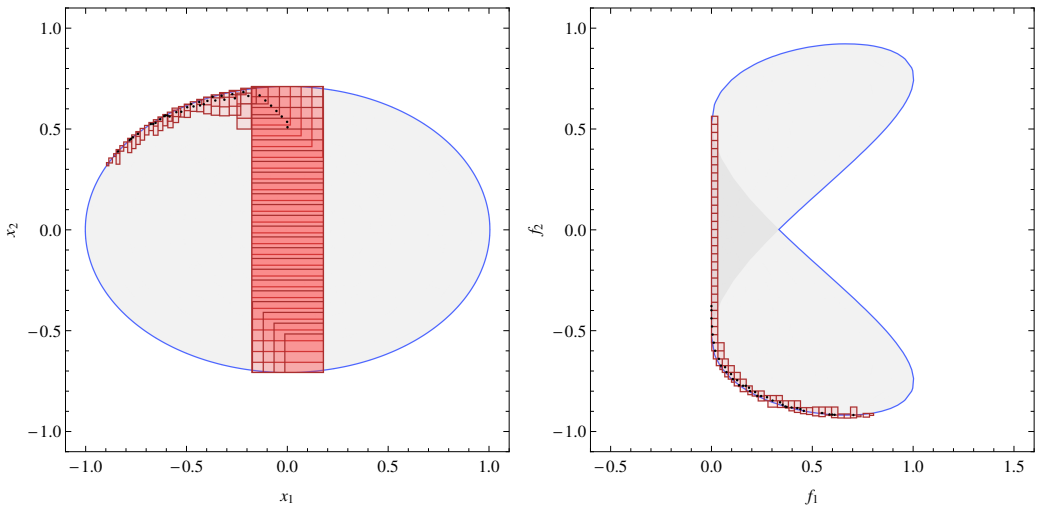
An application of direct B&B with precision of boxes set to 0.05, is shown on Figure 3.5(a). Splitting is performed in the decision space, hence the paving of $\mathcal{X}^*_W$ is regular while the paving overlaps in the objective space, noticeably around the minimum of f_2.

In inverse methods, depicted on Figure 3.5(b), the paving is constructed in the objective space, with precision 0.05, while interval constraint techniques propagate the objective boxes to the decision space. Boxes in the decision space can overlap. On this example, the boxes around weakly Pareto optimal solutions are overlapping. A thinner paving of the decision space can be obtai-

10. The local search is viewed as a dynamical system. The detection of Pareto optimal solutions corresponds to the detection of an invariant of this dynamical system



(a) Direct : paving of the set of Pareto optimal solutions



(b) Inverse : paving of the nondominated outcomes

Figure 3.5 – Direct vs inverse interval Branch & Bound

ned by also splitting the decision boxes. This is even necessary when distinct equivalent solutions (solutions having same objective values) are Pareto-optimal.

As inverse methods, Barichard and Hao have developed PICPA [4], namely Population and Interval Constraint Propagation Algorithm, which is a B&B-like algorithm whose set of sub-problem has a limited size. It builds a fixed-size paving of the nondominated outcomes. A population of boxes in the objective space is constructed by bisecting an initial box $\boldsymbol{y}$ containing the feasible objective values $\mathcal{Y}$ until a population size is reached. Interval constraint propagation methods are applied on these boxes in order to associate variable values (represented by boxes in the decision space) to these objective boxes. Interval constraint propagation can determine whether an objective box is infeasible. Finally, once a maximal size paving is build, instantiation of feasible solutions within the nondominated boxes are attempted. If such a solution is successfully instantiated within an objective box, then all corresponding dominated boxes in the population are removed. The process stops once a maximal population of undiscardable objective boxes is reached without being able to discard one of its element. The set of objectives boxes and the set of nondominated instantiated solutions found are returned. Having a limited number of boxes to build the paving, the complexity of PICPA is practically lower than that of classical B&B, in trade-off with difficulties in asserting the quality of the returned solutions with respect to the population size. Moreover, the instantiation procedure is not described in [4], though its computational efforts can be tuned suggesting it can be a computationally expensive process. Eventually, problems considered do not contain equality constraints.

From PICPA, Kubica and Woźniak [66, 64, 65] have proposed an inverse B&B procedure for inequality constrained multiobjective problems. Contrarily to PICPA, no population size is used. Instead, the termination criterion for a sub-problem is based on a precision on the objective boxes similar to usual termination criterion in interval B&B as seen in Section 2.4 (p. 41). Associating variable domains to objective boxes and instantiation of solutions are done using a set inversion technique called SIVIA [57]. For this purpose, the decision box $\boldsymbol{x}$ associated to each sub-problem is instead replaced by a triplet of set of decision boxes $(\mathcal{S}_{in}, \mathcal{S}_{bound}, \mathcal{S}_{unchecked})$ corresponding to a paving of the set $f^{-}1(\boldsymbol{y})$ and $g^{-}1([-\infty, 0])$, i.e. the inverse image of $\boldsymbol{y}$ satisfying the constraints. The set $\mathcal{S}_{in}$ stores inner boxes whose image by the objective function is proved to lie within $\boldsymbol{y}$ and are certainly satisfying all the inequality constraints. The set $\mathcal{S}_{bound}$ stores boundary boxes, i.e. small decision boxes that are potentially on the boundary of the inverse set. The $\mathcal{S}_{unchecked}$ stores temporarily unchecked boxes. The whole SIVIA-like process used in [66, 64, 65] is shown in Algorithm 4.

Breaking-SIVIA, i.e. with the boolean *breaker* set to true, replaces the pruning step of the B&B. In breaking-SIVIA, Algorithm 4 stops once an inner box is found. In such situation, the induced sub-problem can be considered as instantiated : there exist a solution box $\boldsymbol{x}$ certainly satisfying all constraints and verifying $f(\boldsymbol{x}) \subseteq \boldsymbol{y}$. When instantiated, a sub-problem can be used to discard any sub-problem it dominates, i.e. any sub-problem with objective box $\boldsymbol{y}'$ verifying $\overline{y} \leq \underline{y}'$. If the procedure does not manage to build an inner box (all produced boxes are boundary boxes), then the sub-problem is considered as terminal. Algorithm 4 is enhanced in [64] by applying pruning techniques at line 4.3, namely component-wise Newton contractor and/or Gauss-Seidel on constraints $f(x) = y$ and system of first order conditions. Moreover, a discarding test based on monotonicity detection is proposed in [64] and is defined as follows :

Theorem 3.3.1 (Monotonicity test from [66])**.** *Consider problem* (3.1) *and a decision box* $\boldsymbol{x}$*. If there is a coordinate* x_i *such that* $\nabla_i \boldsymbol{f}_j(\boldsymbol{x}) < 0$ *or* $\nabla_i \boldsymbol{f}_j(\boldsymbol{x}) > 0$ *for all* $j = 1, \ldots, m$ *(i.e. all*

Algorithm 4: SIVIA-like procedure

Input: Objective box $\boldsymbol{y}$; Triplet set of decision boxes $\mathcal{S}_{in}$, $\mathcal{S}_{bound}$, $\mathcal{S}_{unchecked}$; NLMOO problem (2.42) ; Boolean *breaker* ; Precision ϵ

4.1 **while** $\mathcal{S}_{unchecked} \neq \emptyset$ **do**
4.2 $\quad \boldsymbol{x} \leftarrow Extract(\mathcal{S}_{unchecked})$;
4.3 $\quad \boldsymbol{x} \leftarrow Pruning(\boldsymbol{x})$;
4.4 $\quad$ **if** $\emptyset \neq (\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{g}(\boldsymbol{x})) \subseteq (\boldsymbol{y}, [-\infty, 0])$ **then**
4.5 $\quad\quad \mathcal{S}_{in} \leftarrow \mathcal{S}_{in} \cup \{\boldsymbol{x}\}$;
4.6 $\quad\quad$ **if** *breaker* **then return**;
4.7 $\quad$ **else if** $\mathrm{wid}(\boldsymbol{x}) \leq \epsilon$ **then** $\mathcal{S}_{bound} \leftarrow \mathcal{S}_{bound} \cup \{\boldsymbol{x}\}$;
4.8 $\quad$ **else** $\mathcal{S}_{unchecked} \leftarrow \mathcal{S}_{unchecked} \cup Split(\boldsymbol{x})$;
4.9 **end**
4.10 **return**

objectives are either decreasing or increasing along x_i) and such that all constraints involving the variable x_i are strictly satisfied within $\boldsymbol{x}$, then $\boldsymbol{x}$ does not contain any Pareto-optimal solution.

Eventually in [65], bisections heuristics are proposed and used at line 4.8 so as to accelerate the detection of inner boxes. At the end of the breaking-SIVIA, the box $\boldsymbol{y}$ is intersected with the interval evaluation of the objective on the boxes in the sets $\mathcal{S}_{in}$, $\mathcal{S}_{bound}$ and $\mathcal{S}_{unchecked}$. When constraints are considered, this allows to refine the objective boxes to feasible objective values. Finally, at the end of the B&B, a non-breaking SIVIA is applied to each terminal sub-problem, such that the associated triplet of sets covers accurately the inverse image of the objective and constraints. We can note that this inverse B&B does not considers explicit lower and upper bound sets, although $\underline{y}$ acts as a lower bound of a sub-problem and the set of all $\overline{y}$ among all sub-problems having an inner box, and filtered by dominance, acts as an upper bound set. Although the methods from Kubica and Woźniak appears efficient on unconstrained multiobjective problems with a couple of variables and difficult objectives, the number of required bisections for converging increases dramatically fast when tackling larger problems with constraints [65].

In the context of direct methods, Ruetsch [109] has defined a B&B which uses a discarding test based on optimality conditions. This test checks whether a decision box can contain (locally) Pareto optimal solutions. Otherwise, the corresponding sub-problem can be discarded. This technique is however difficult to apply on constrained problems, and only experimented on biobjective problems. This work has been patented [110]. Note that no details on this B&B method is given, although it seems not to use pruning techniques other than this discarding test.

Fernández and Tóth [32, 126] have proposed a generic direct B&B for biobjective optimization that uses discarding tests based on monotonicity and a technique pruning variable domains based on dominance with respect to the upper bound set. The first monotonicity test proposed in [32], whose adaptation to multiobjective, is defined as follows :

Theorem 3.3.2 (Monotonicity test from [32])**.** *Consider problem (3.1) and a decision box $\boldsymbol{x}$. Suppose there is a coordinate x_i such that $(\forall\ j = 1, \ldots, m)\ \nabla_i \boldsymbol{f}_j(\boldsymbol{x}) < 0$ or $\nabla_i \boldsymbol{f}_j(\boldsymbol{x}) > 0$. Then if the facet $\boldsymbol{x}_i^+$ (respectively $\boldsymbol{x}_i^-$) certainly satisfies all constraints, then there is no Pareto-optimal solution in the interior of $\boldsymbol{x}$.*

If the facet intersects some boundary of the search space, $\boldsymbol{x}$ can be reduced to the facet, otherwise the corresponding sub-problem can be discarded.

Another monotonicity test, called generalized monotonicity test, has been proposed in [32]. Dedicated to biobjective problems, this test checks whether there is a direction within $\boldsymbol{x}$ strictly improving both objectives.

Theorem 3.3.3 (Generalized monotonicity test from [32])**.** *Consider a biobjective problem* (3.1) *and a decision box* $\boldsymbol{x}$*. Suppose there is no variable satisfying Theorem* 3.3.2*. If one of the following conditions holds :*

(i) $\nabla_i \boldsymbol{f}_1(\boldsymbol{x}) > 0$, $\nabla_j \boldsymbol{f}_2(\boldsymbol{x}) > 0$ *and* $\underline{\frac{\nabla_i f_2(\boldsymbol{x})}{\nabla_j f_2(\boldsymbol{x})}} - \underline{\frac{\nabla_i f_1(\boldsymbol{x})}{\nabla_j f_1(\boldsymbol{x})}} > 0$ *;*

(ii) $\nabla_i \boldsymbol{f}_1(\boldsymbol{x}) < 0$, $\nabla_j \boldsymbol{f}_2(\boldsymbol{x}) > 0$ *and* $\underline{\frac{\nabla_i f_1(\boldsymbol{x})}{\nabla_j f_1(\boldsymbol{x})}} - \overline{\frac{\nabla_i f_2(\boldsymbol{x})}{\nabla_j f_2(\boldsymbol{x})}} > 0$ *;*

(iii) $\nabla_i \boldsymbol{f}_1(\boldsymbol{x}) > 0$, $\nabla_j \boldsymbol{f}_2(\boldsymbol{x}) < 0$ *and* $\overline{\frac{\nabla_i f_1(\boldsymbol{x})}{\nabla_j f_1(\boldsymbol{x})}} - \underline{\frac{\nabla_i f_2(\boldsymbol{x})}{\nabla_j f_2(\boldsymbol{x})}} > 0$ *;*

(iv) $\nabla_i \boldsymbol{f}_1(\boldsymbol{x}) < 0$, $\nabla_j \boldsymbol{f}_2(\boldsymbol{x}) < 0$ *and* $\overline{\frac{\nabla_i f_2(\boldsymbol{x})}{\nabla_j f_2(\boldsymbol{x})}} - \overline{\frac{\nabla_i f_1(\boldsymbol{x})}{\nabla_j f_1(\boldsymbol{x})}} > 0$ *;*

Then there is a direction v*, with* $v_k = 0$ *for all* $k \neq i, j$ *and* $v_i, v_j \neq 0$ *(positive or negative depending which of the previous condition is satisfied), for which both objectives are increasing. If in addition the induced facets* $\boldsymbol{x}_i^+$ *(or* $\boldsymbol{x}_i^-$*) and* $\boldsymbol{x}_j^+$ *(or* $\boldsymbol{x}_j^-$*) satisfy all constraints, then there is no Pareto-optimal solution in the interior of* $\boldsymbol{x}$*.*

If the two induced facets intersect the boundary of the search space, the box $\boldsymbol{x}$ *can be reduced to the union of the two facets.*

The lower bound set of a sub-problem consists of an outer-approximation of the local ideal point, obtained by interval evaluations of the objectives. The upper bound set is a set of feasible midpoints (or corner points), filtered by dominance, computed from decision boxes $\boldsymbol{x}$ at each sub-problem. Solutions from the upper bound set are used inside a pruning procedure. Given a box $\boldsymbol{x}$, a variable x_i and solution $\hat{x}$, with objectives $\hat{y}$, from the upper bound set, this pruning operator contracts $\boldsymbol{x}_i$ towards solutions satisfying the constraints $f(x) \leq \hat{y}$. This operator uses gradient information and resembles the contracting process based on box-consistency. It actually extract from $\boldsymbol{x}_i$ the portions not satisfying the constraints. Technically, given the box $\tilde{\boldsymbol{x}}^i = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \mathrm{mid}(\boldsymbol{x}_i), \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n)$, the value $\underline{f(\tilde{\boldsymbol{x}}^i)}$ and $\nabla \boldsymbol{f}(\boldsymbol{x})$ the gradient of f over $\boldsymbol{x}$, the process builds univariate lower bounding functions of each objectives along x_i. The intersection of the lower bounding functions with $\hat{y}$ are built, and the portion $\boldsymbol{x}^j$ of $\boldsymbol{x}_i$ yielding $f_j(x) > \hat{y}$ determined for each j. Eventually, the intersection of all the $\boldsymbol{x}^j$ (if not empty) is discarded from $\boldsymbol{x}_i$, possibly bisecting it if the nonempty intersection lies in the interior of $\boldsymbol{x}_i$. The whole process is illustrated on a single variable biobjective problem on Figure 3.6. As this pruning operator can bisect the domain of a variable, it is also seen as a splitting strategy. The pair of variable x_i and vector $\hat{y}$ for the application of the pruning operator are selected as the one yielding the largest discarded part.

Except this pruning operator, the B&B from [32, 126] does not use other contracting procedures in order to remove infeasible solutions. Hence, constraints are simply exploited through the use of feasibility tests. Two main different termination criteria are proposed. One simply considering a precision on the relative width of the interval evaluation of the objectives or a precision on the width of the decision boxes. Another one, well suited for decision making purposes, requiring additionally to a precision on the interval evaluation of the objectives that each returned terminal sub-problem contains a feasible solution. The set of sub-problem $\mathcal{S}$ is ordered by either increasing $\underline{f_1(\boldsymbol{x})}$, $\underline{f_2(\boldsymbol{x})}$ or with respect to a weighted sum of the normalized objectives. This B&B has been

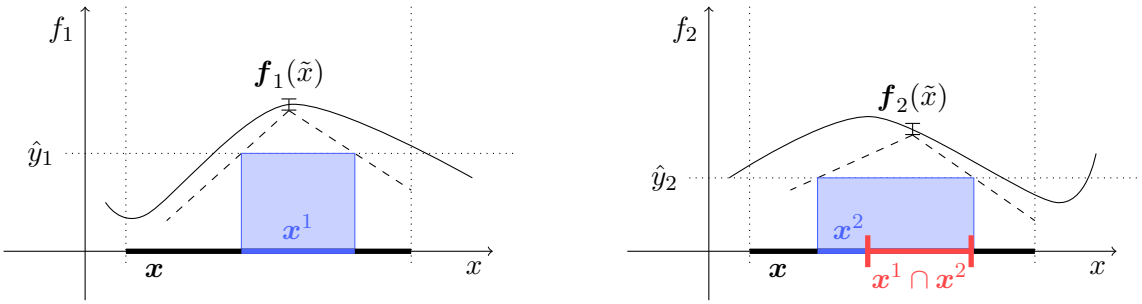


Figure 3.6 – Pruning method from [32] on single variable biobjective problem. The portion $\boldsymbol{x}^1 \cap \boldsymbol{x}^2$ contains dominated solutions and can be removed from $\boldsymbol{x}$.

numerically experimented on a class of competitive location problems. These problems, and the B&B itself, assume that the biobjective problem does not contain equality constraints. We can finally note that the authors have also developed an interval-based global solving algorithm based on εC [31], which uses some of the B&B components.

Eventually, Goldsztejn et al. [37] have proposed discarding tests based on first-order optimality conditions for optimization problems with any number of objectives. These tests have been briefly stated in the previous chapter and are here detailed. Three tests are proposed in [37], which check, given a box $\boldsymbol{x}$, if their exist multipliers solving the first n equations of the first-order system. Let $\boldsymbol{G}(\boldsymbol{x})$ be the interval matrix defined by :

$$\boldsymbol{G}(\boldsymbol{x}) = (\nabla \boldsymbol{f}(\boldsymbol{x}) | \nabla \dot{\boldsymbol{g}}(\boldsymbol{x}) | \nabla \boldsymbol{h}(\boldsymbol{x})) \tag{3.18}$$

where $\dot{g}$ denotes the vector of potentially active inequalities in $\boldsymbol{x}$. Note the gradients can be equivalently replaced by generalized gradients in the following. The first test in [37] is as follows :

Theorem 3.3.4 (Full rank test)**.** *Consider problem* (3.1) *and a decision box* $\boldsymbol{x}$*. If* $\boldsymbol{G}(\boldsymbol{x})$ *is full column rank, then* $\boldsymbol{x}$ *does not contain (locally) Pareto optimal solutions.*

Note that LICQ necessarily holds in $\boldsymbol{x}$ for $\boldsymbol{G}(\boldsymbol{x})$ to be full column rank. Checking if an interval matrix is full column rank can be made, for example, using interval Gauss elimination [90]. It is easy to see that if $\boldsymbol{G}(\boldsymbol{x})$ is full column rank, there are no multipliers λ, $\dot{r}$ and s such that $\boldsymbol{G}(\boldsymbol{x})(\lambda, \dot{r}, s) = 0$, hence the system (3.4) cannot be solved within $\boldsymbol{x}$.

This first test does not consider the signs of the multipliers. The second test from [37] checks signs of row entries of the matrix $\boldsymbol{G}(\boldsymbol{x})$.

Theorem 3.3.5 (Multipliers sign test)**.** *Consider problem* (3.1) *without equality constraints and a decision box* $\boldsymbol{x}$*. If there is a row of* $\boldsymbol{G}(\boldsymbol{x})$ *such that all its interval entries do not contain* 0 *and have the same sign, then* $\boldsymbol{x}$ *does not contain (locally) Pareto optimal solutions.*

For the system (3.4), multipliers λ and r are positive, but no all zero. Thus, if $\boldsymbol{x}$ satisfies this test, the corresponding row of the system (3.4) cannot be zero.

Eventually, the latter test checks the domains of the multipliers themselves. For this test, the matrix $\boldsymbol{G}_*(\boldsymbol{x})$ is considered and defined as :

$$\boldsymbol{G}_*(\boldsymbol{x}) = \begin{pmatrix} \nabla \boldsymbol{f}(\boldsymbol{x}) & \nabla \dot{\boldsymbol{g}}(\boldsymbol{x}) & \nabla \boldsymbol{h}(\boldsymbol{x}) \\ 1^T & 0^T & 0^T \end{pmatrix}, \tag{3.19}$$

where 1^T and 0^T denote horizontal vectors of ones and zeros respectively. This matrix corresponds to $\boldsymbol{G}(\boldsymbol{x})$ considering an additional line which is used for normalizing the multipliers λ following $\sum_{i=1}^{m} \lambda_i = 1$: the normalization of the KKT conditions, valid only if LICQ holds. The following theorem is used to build domains of multipliers, which in turn can be used to check whether (locally) Pareto optimal solutions lies within $\boldsymbol{x}$.

Theorem 3.3.6 (Multipliers domain test)**.** *Consider problem (3.1) and a decision box $\boldsymbol{x}$. Let $k = \dot{p} + q$, $C \in \mathbb{R}^{(m+k)\times(n+1)}$ be a matrix (e.g. inverse midpoint preconditioner of $\boldsymbol{G}_*(\boldsymbol{x})$) and consider $\boldsymbol{A} = C\boldsymbol{G}_*(\boldsymbol{x})$. Given :*

$$C = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}, \quad \boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{pmatrix}, \tag{3.20}$$

with $C_{11} \in \mathbb{R}^{m\times n}$,$C_{12} \in \mathbb{R}^{m\times 1}$,$C_{21} \in \mathbb{R}^{k\times n}$,$C_{22} \in \mathbb{R}^{k\times 1}$, $\boldsymbol{A}_{11} \in \mathbb{IR}^{m\times m}$, $\boldsymbol{A}_{12} \in \mathbb{IR}^{m\times k}$, $\boldsymbol{A}_{21} \in \mathbb{IR}^{k\times m}$ and $\boldsymbol{A}_{22} \in \mathbb{IR}^{k\times k}$, i.e. with

$$\boldsymbol{A}_{11} = C_{11}\nabla \boldsymbol{f}(\boldsymbol{x}) + C_{12}1^T \tag{3.21}$$
$$\boldsymbol{A}_{12} = C_{11}(\nabla \dot{\boldsymbol{g}}(\boldsymbol{x})|\nabla \boldsymbol{h}(\boldsymbol{x})) \tag{3.22}$$
$$\boldsymbol{A}_{21} = C_{21}\nabla \boldsymbol{f}(\boldsymbol{x}) + C_{22}1^T \tag{3.23}$$
$$\boldsymbol{A}_{22} = C_{21}(\nabla \dot{\boldsymbol{g}}(\boldsymbol{x})|\nabla \boldsymbol{h}(\boldsymbol{x})). \tag{3.24}$$

If $\boldsymbol{A}_{22}$ is strictly diagonally dominant [90], then :

(i) LICQ holds in $\boldsymbol{x}$. The multipliers $(\lambda, \dot{r}, s)$ can be normalized according to $\sum_{i=1}^{m} \lambda_i = 1$.

(ii) The normalized multipliers $(\lambda, \dot{r}, s)$ must satisfy $G_*(\lambda, \dot{r}, s) = e$ and $A(\lambda, \dot{r}, s) = Ce$ for a $G_* \in \boldsymbol{G}_*$ and a $A \in \boldsymbol{A}$ where $e = (0, \ldots, 0, 1) \in \mathbb{R}^{n+1}$.

(iii) The initial domains of normalized multipliers are $\boldsymbol{\lambda}_i = [0, 1]$ [11] for the objective multipliers. The initial domain of multipliers $(\dot{r}, s)$ can be set to $(\dot{\boldsymbol{r}}, \boldsymbol{s}) = C_{22} + [-1, 1] \cdot \|C_{22} - \boldsymbol{A}_{21}\boldsymbol{\lambda} - \boldsymbol{A}_{22}C_{22}\|$. The domains $\dot{\boldsymbol{r}}$ are intersected with $[0, +\infty]$.

(iv) The normalized multipliers domains can be improved by applying interval solving techniques to the interval linear systems $\boldsymbol{G}_*(\boldsymbol{x})(\lambda, \dot{r}, s) \ni e$ and $\boldsymbol{A}(\lambda, \dot{r}, s) \ni Ce$, using e.g. the interval Gauss-Seidel. If this interval method proves one of the multiplier domain is empty, then $\boldsymbol{x}$ does not contain (locally) Pareto optimal solutions.

This theorem is useful for both eliminating sub-problems but also for providing domains for the multipliers that can be used for other purposes, such as contracting with interval Newton methods on the system (3.4). These three tests combined are also efficient in reducing the cluster effect [37] possibly encountered when solving globally the problem (3.1) with intervals.

3.3.4 Performance assessment

Let $\hat{\mathcal{X}}_1$ and $\hat{\mathcal{X}}_2$ (with images $\hat{\mathcal{Y}}_1$ and $\hat{\mathcal{Y}}_2$) be two sets of solutions of a multiobjective problem not dominating each other returned by two different solvers. The performance of these solvers on this problem can be evaluated by checking the computational effort (time, memory, evaluation of functions, etc) required to find $\hat{\mathcal{X}}_1$ or $\hat{\mathcal{X}}_2$, and the respective quality of $\hat{\mathcal{X}}_1$ and $\hat{\mathcal{X}}_2$.

11. If a single objective is considered, $\boldsymbol{\lambda} = [1, 1]$.

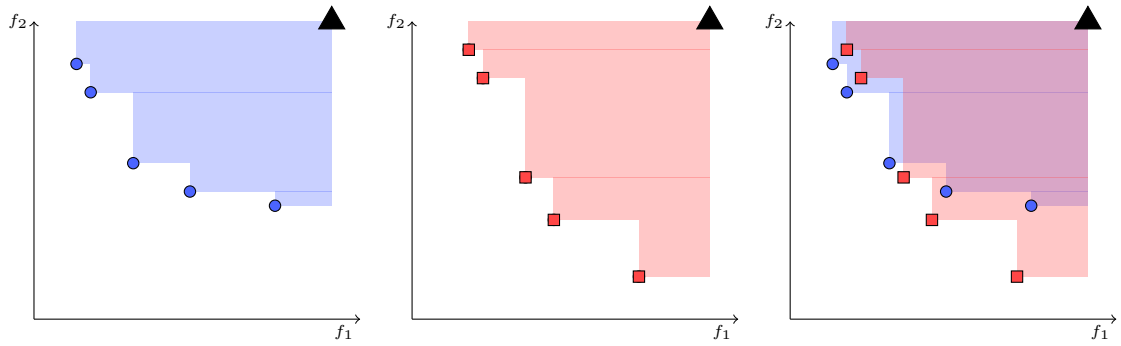


Figure 3.7 – Hypervolumes computed from two sets of nondominated points with respect to a reference point (in black triangle). The set in red dots has a larger hypervolume than the one in blue square, but some areas covered by the latter are not covered by the former.

In single objective optimization, asserting the quality of the solution returned by the solver is straightforward, but asserting the overall performances is a multicriteria issue. The solver returning the best solution with least computational effort will be preferred, but one solver may return the global optima with huge computational efforts while another may find a solution of poor quality with few efforts. Some preferences are then specified in order to merge those two measures. For example, a desired quality level for the solution may be used, such that the solver reaching this quality level with the least computational effort is preferred.

Evaluating the quality of sets of nondominated solutions is difficult [137]. Dominance relations allow cases where $\hat{\mathcal{X}}_1$ does not dominate $\hat{\mathcal{X}}_2$ entirely. These dominance relations are also not sufficient to measure the difference between two sets of nondominated solutions. Therefore, different quality measures have been developed in order to evaluate sets of nondominated solutions [138]. The hypervolume, for example, is the volume dominated by a set of nondominated points (images in the objective space of nondominated solutions) with respect to a reference point, usually taken as the anti-ideal y^A of the considered multiobjective problem. Figure 3.7 illustrates the hypervolume measure on two sets of nondominated points. One can note that even if a set of nondominated points has a larger hypervolume than another one, it does not necessarily mean it is better as it may still contains solutions dominated by the other set (see Figure 3.7). Therefore, comparing only the hypervolumes of two sets can only determine if one set is at least as good as the other one. A finer analysis can be made if the two hypervolumes are considered simultaneously, analyzing which area are covered by one set and not covered by the other set. This latter measure considering two sets at once is referred to a binary measure [138]. The hypervolume is a convenient measure for evaluating both closeness to the Pareto optimal set and the spread along the Pareto front. There are others measures aiming at evaluating these two elements separately, see [138]. We do not present these other measurements here as we will later derive a measure based on hypervolume that will be sufficient for our analysis.

Performances of a solver of course vary from one problem to another. Several suite of benchmark problems have been defined in the literature like the ZDT [136], DTLZ [22], CTP [21], LZ [72] or WFG [55]. These problems have different characteristics : convex Pareto front, constrained problems, nondifferentiable problems, etc. They have a known Pareto front, hence making it possible to assert the ability of a single solver to tackle them.

3.4 Conclusion

We have presented in this chapter the theoretical background of nonlinear multiobjective optimization. We have also detailed many approaches from the literature, describing their strengths and weaknesses.

From this state of the art, we observe a recent growing interest for continuation methods. These techniques are indeed very appealing as they exploit the local structure of Pareto optimal solutions, hence yielding to a fast and efficient recovering of (locally) Pareto optimal solutions. These methods are however not global and require a starting (locally) Pareto optimal solution. Hence they have to be coupled with global search algorithms. Such couplings have been done, see e.g. [119, 50]. However, these hybridizations have mostly concerned stochastic and asymptotically complete global search algorithms and unconstrained problems.

Complete global search methods, such as B&B, are the only methods able to approximate accurately, and deterministically, the global Pareto optimal set with additional rigor from the use of interval analysis. These methods are however computationally demanding as the cost of a complete search is exponential. They are hence disapproved in the literature and in applications. There is nevertheless a good potential in improving these techniques as they remain an adequate process for solving small sized but highly nonlinear problems. As solutions generally form manifolds, a promising idea is the hybridization of B&B with continuation techniques. Indeed, continuation is able to recover efficiently connected manifolds of locally Pareto optimal solutions. Hence, such a coupling will help the global search to focus on reaching the different disconnected components of the Pareto set while the continuation recover these components. Continuation can also be used to update efficiently the upper bound of the B&B.

Interval based approaches are best suited for generic B&B algorithms. Bounding has to be rigorous (e.g. solutions from the upper bound have to be feasible), therefore if we want to use continuation, this latter has to give some guarantees. As continuation is based on solving the system of equations (3.4), we want to be sure that every solution found by continuation solves this system, i.e. is not roughly approximated. Numerical proofs of existence of solutions within an enclosure can be obtained by interval analysis, see Section 2.3.1 (p. 21). Interval analysis can also be used to enclose manifolds of locally Pareto optimal solutions. Ensuring the continuity of local Pareto solutions, helps understanding the problem structure, for example distinguishing reliably the different disconnected components. These enclosures can also be used to avoid the B&B to spend searching efforts within them, e.g. via exclusion regions.

CHAPTER 4

Certified Parallelotope Continuation for one-manifold

4.1 Introduction 83
4.2 Contracting, Inflating and Certifying parallelotopes 84
4.3 Parallelotope Continuation ParCont 86
4.3.1 Algorithm Description 86
4.3.2 Properties of the Algorithm 91
4.3.3 Limitations of ParCont 98
4.4 Experiments 99
4.4.1 Influence of the manifold topology 100
4.4.2 Influence of the conditioning 101
4.4.3 Influence of the embedding space dimension 103
4.4.4 Homotopy continuation 104
4.4.5 Control synthesis 106
4.4.6 Conclusion 107
4.5 Adaptation to biobjective optimization 107
4.5.1 Detecting rigorously changes of constraint activity 107
4.5.2 Limitations 110
4.6 Biobjective experiments 111
4.6.1 Illustration of change of active constraints 111
4.6.2 Following many changes of constraint activity 113
4.6.3 Connectivity of the Pareto front through nonoptimal solutions 114
4.7 Conclusion 115

4.1 Introduction

Continuation methods [1] allow exploring step by step a solution manifold, usually defined as the solution set of an under-constrained system of equations $F(x) = 0$ with $F : \mathbb{R}^n \to \mathbb{R}^q$ and $q < n$. This chapter focuses on systems where $n - q = 1$, whose solution manifolds are curves. It has a wide range of applications, e.g. polynomial root finding via homotopy [5], nonlinear eigenvalue problems [9], robot path planning [81], ..., and biobjective optimization [53, 119]. The most simple and effective continuation method is the predictor corrector algorithm. It includes the simple embedding method and several of its improvements such as the parameter switching [103, 104] and the pseudo-arclength method, that have the definite advantage over the simple embedding method that they naturally track folds [23]. They are all however subject to jumping between disconnected components without any prompt. While this may be acceptable in some contexts since solutions are anyway computed, such jumps contradict the essence of continuation. Furthermore, connectivity is mandatory in some applications, e.g. robot control synthesis where such jumps would result in a non-feasible control path, or in homotopy methods where such jumps would prevent computing all the solutions of the original system.

Several methods have been proposed to certify the connectivity between continuation steps. Smale's α-theory [125] has been used in [5] to derive some maximal step size that certifies the connectivity of a predictor corrector continuation in a homotopy method for computing roots of square polynomial systems in the fields of complex numbers (it is implemented in the software Macaulay2 [45]). A simplified Kantorovich theorem argument has been proposed and used in [29] in order to derive a maximal step size that certifies the connectivity of a general predictor corrector continuation. Note that Smale's α-theory and Kantorovich theorem are known to be closely related [105]. In the more general context of multi-parameter continuation, the reach of a manifold [1] is used in [11] in order to build an isotopic triangulation of the manifold. Although this theoretical framework is not exactly a continuation method, it is close in spirit and it adds connectivity certification to the multi-parameter continuation method proposed in [52]. No implementation of [11] is available.

Interval analysis, see Chapter 2 and Section 2.3.1 (p. 21), allows certifying the existence of solutions to systems of equations through the verification of the hypotheses of existence theorems, e.g. the Brouwer fixed point theorem, the Poincaré-Miranda theorem, or the interval Newton existence test. A continuation method with certified connectivity based on interval analysis was investigated in [58], where an embedding method coupled with a strategy of parameter switching was proposed to build a sequence of boxes that encloses the solution path. The main drawback of [58] is intrinsic to the usage of boxes, which do not accurately capture the shape of the manifold. This key issue was tackled in [39] where parallelotopes are used instead of boxes, see Section 2.3.1.3. This allows applying a parametric interval Newton test in an auxiliary basis that better fits the manifold orientation. In [39], parallelotope computations were used in a Branch & Prune framework that globally explores a manifold, entailing a computational complexity exponential in the number of variables as is the case of any global search in nonlinear problems. Continuation potentially allows drastically reducing this complexity by instead following locally the manifold, leading to a polynomial time complexity, in the number of variables, considering a fixed manifold dimension.

1. The reach of a manifold has been introduced in [30] and characterizes the size of a neighborhood of the manifold within which any point has a unique projection on the manifold. Roughly speaking, the manifold has a simple behavior within this neighborhood.

Computations with parallelotopes are used in Section 4.2 and 4.3 to define a continuation algorithm with certified existence and connectivity : The continuation step consists of an intervalization of the predictor corrector process, where a parametric interval Newton operator is used to build a parallelotope that contains one unique portion of the curve. Continuation steps are interleaved with several checks that ensure the connectivity and the absence of backtrack, while tracking the presence of a loop and certifying it when the curve is actually periodic. The result of this process is illustrated in Figure 2.6(b) (p. 33). Even on this simple example, parallelotopes show a better adaptation to the curve than boxes, which will become a key advantage for more difficult problems. The simplicity of this algorithm is a strong advantage for its implementation and for studying its properties (its correctness, its termination and its asymptotic convergence are investigated in Subsection 4.3.2). Experimental results presented in Section 4.4 show a drastic efficiency improvement with respect to an interval embedding method with parameter switching close to the one proposed in [58], and demonstrate that the proposed method is quite competitive with respect to the Macaulay2 [45] implementation of [5] when applied to homotopy continuation for complex polynomial root finding. Eventually, this method is applied to nonlinear biobjective optimization problems, through the use of first-order optimality conditions in Section 4.5. Necessary adaptations of the original algorithm have to be made in order to handle properly inequality constraints. The adapted method is applied to a set of illustrative biobjective optimization problems in Section 4.6.

Two papers have been published on this algorithm [79, 80]. The first one treating the certified continuation algorithm, the second one its adaptation to biobjective problems.

4.2 Contracting, Inflating and Certifying parallelotopes

In Section 2.3.1.3 (p. 29), parallelelotopes domains are introduced as a way of rigorously computing manifolds of solutions to underconstrained systems of equations. This preliminary section introduces algorithms that apply the techniques introduced in Chapter 2, and that will be used for the design of the certified continuation algorithm presented in Section 4.3. Two algorithms are required : one for contracting parallelotopes to the curve of solutions, one for certifying a parallelotope to enclose curve of solutions starting from an initial guess.

Algorithm 5 applies the interval Newton for parallelotopes inductively until some fixed point is reached or a maximum of iterations is reached (by default $\overline{k} = 15$). It aims at reducing a parallelotope without loosing any solution to $F(x) = 0$. Formally, it satisfies

$$\forall x \in \widehat{\boldsymbol{x}}, F(x) = 0 \implies x \in \text{Contract}(F, \widehat{\boldsymbol{x}}), \tag{4.1}$$

which is a direct consequence of Theorem 2.3.6 (p. 27) for the parametric interval Newton operator.

It turns out to be also useful to inflate a box in such a way that the interval Newton operator strictly contracts it. This is even critical in the context of certified continuation where the certification has to be performed from a initial non-rigorous guess. In the context of certification of manifolds using parallelotopes, inflating a parallelotope consists in inflating its characteristic box $\boldsymbol{u}$ corresponding to the non-parametric dimensions. To this end, Algorithm 6 implements a two stage inflation process adapted from a box-inflation process proposed in [40, 56] : The main inflation is performed by the interval Newton operator itself at Line 6.4, applying it without intersecting the previous domain so as to allow shifting and inflating the box. When this inflation

Algorithm 5: Contract

Input: $F : \mathbb{R}^n \to \mathbb{R}^q$; $(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}) \in \mathbb{R}^{n\times n} \times \mathbb{IR}^n \times \mathbb{R}^n$;
Parameters: $\overline{k} \in \mathbb{N}$;

5.1 $k \leftarrow 0$;
5.2 **repeat**
5.3 $\quad \boldsymbol{u}' = \boldsymbol{u}$;
5.4 $\quad \boldsymbol{u} \leftarrow \mathbf{N}(F, (C, (\boldsymbol{u}', \boldsymbol{v}), \tilde{x})) \cap \boldsymbol{u}'$;
5.5 $\quad k \leftarrow k + 1$;
5.6 **until** $\boldsymbol{u}' = \boldsymbol{u}$ **or** $k \geq \overline{k}$;
5.7 **return** $(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$

process succeeds (informally the initial parallelotope has to be small enough and close enough to a regular manifold), it often converges to a limit parallelotope from the inside, which does not allow observing the strict contraction necessary to certify the manifold. Therefore, a static (relative and absolute) inflation is performed at Line 6.9, before applying the interval Newton operator, in order to allow the latter to be strictly contracting (such a static inflation is similar to the ϵ-inflation described in [111]). A larger static inflation eases the interval Newton contraction, but also deteriorates the overall convergence of the inflation iteration. Although the behavior of the process seems to be not very sensitive to reasonable changes of the static inflation parameters δ and χ, experiments have shown that $\delta = 1.1$ (which is compatible with the value recommended in [111]) and $\chi = 10^{-12}$ represent a good setting in general (although χ obviously depends on the machine and the problem). Finally, the iteration is stopped as soon as $\boldsymbol{u} \subseteq \text{int}(\boldsymbol{u}')$, i.e. such that the property (2.33) holds, or when divergence is observed, either by monitoring the distance between successive iterates and enforcing a decrease of at least $\overline{\mu}$ (by default $\overline{\mu} = 1.0$, aiming only at detecting divergence), or by enforcing a safeguarding maximum number of steps (by default $\overline{k} = 15$). Algorithm 6 returns a pair of results composed of a boolean indicating whether it was able to strictly contract $\boldsymbol{u}$, i.e., to certify the parallelotope ; and the resulting parallelotope. The latter is typically quite large due to the double inflation process, which is in fact advantageous since the existence and uniqueness of the solutions hold inside its whole region. When necessary, tighter parallelotopes could be computed by bisecting and contracting them as done in [39]. Note that each iteration of Algorithm 6 involves the interval evaluation of a function, that of a Jacobian matrix and $O(n^3)$ interval operations for the right-preconditioning of the interval Jacobian. The properties of Algorithm 6 are summarized in the following theorem (whose proof is derived straightforwardly from [36, 39, 56] and from Theorem 2.3.7.

Theorem 4.2.1. *Let* $\widehat{\boldsymbol{x}} = (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$ *and* $(\mathbf{true}, (C, (\boldsymbol{u}', \boldsymbol{v}), \tilde{x})) = N\text{–}Inflate(F, \widehat{\boldsymbol{x}})$. *First,* $x \in \widehat{\boldsymbol{x}}$ *and* $F(x) = 0$ *implies* $x \in \widehat{\boldsymbol{x}}' = (C, (\boldsymbol{u}', \boldsymbol{v}), \tilde{x})$. *Second, there exists a unique function* $\gamma : \boldsymbol{v} \to \boldsymbol{u}'$, *differentiable inside* $\text{int}(\boldsymbol{v})$, *such that*

$$\forall v \in \boldsymbol{v},\ F(C(\gamma(v), v) + \tilde{x}) = 0. \tag{4.2}$$

Finally, $\forall x \in \widehat{\boldsymbol{x}}$, $F'(x)$ *is full rank.*

Algorithm 6: N–Inflate

Input: $F : \mathbb{R}^n \to \mathbb{R}^q$; $(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}) \in \mathbb{R}^{n\times n} \times \mathbb{IR}^n \times \mathbb{R}^n$;
Parameters: $\overline{k} \in \mathbb{N}$; $\overline{\mu} \in \mathbb{R}$ (s.t. $0 < \overline{\mu} < 1$) ; $\delta \in \mathbb{R}$ (s.t. $1 < \delta$) ; $\chi \in \mathbb{R}$ (s.t. $0 < \chi$)

6.1 $k, \mu \leftarrow 0$;
6.2 **repeat**
6.3 $\quad k \leftarrow k + 1$;
6.4 $\quad \boldsymbol{u}' \leftarrow \mathbf{N}(f, (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}))$;
6.5 $\quad success \leftarrow \boldsymbol{u}' \subseteq \mathrm{int}(\boldsymbol{u})$;
6.6 $\quad$ **if** $\neg success$ **then**
6.7 $\quad\quad$ **if** $k \geq 2$ **then** $\mu \leftarrow d_H(\boldsymbol{u}, \boldsymbol{u}')/d$;
6.8 $\quad\quad d \leftarrow d_H(\boldsymbol{u}, \boldsymbol{u}')$;
6.9 $\quad\quad \boldsymbol{u} \leftarrow \mathrm{mid}(\boldsymbol{u}') + \delta(\boldsymbol{u}' - \mathrm{mid}(\boldsymbol{u}')) + \chi[-1, +1]$;
6.10 $\quad$ **end**
6.11 **until** $success$ **or** $k > \overline{k}$ **or** $\mu > \overline{\mu}$;
6.12 **return** $(success, (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}))$

4.3 Parallelotope Continuation ParCont

The method we propose, called ParCont, interleaves local curve certifications using Algorithm 6 with several checks, which intend in particular to certify the connections between successive parallelotopes and to ensure no backtrack. It is formally defined in Algorithm 7 whose description resides in Subsection 4.3.1. It takes as inputs the system F whose solution set is the curve to follow, an initial point x_0 on (or very close to) this curve, and an initial box domain $\boldsymbol{x}^{\text{init}}$ within which the curve is to be followed. A direction $d \in \{-1, 1\}$ for the continuation is also provided, in order to be able to perform two continuations processes in different direction (see the definition of C_k in Subsection 4.3.1.1). It iteratively builds two sequences of parallelotopes $(\widehat{\boldsymbol{x}}_k)$ and $(\widehat{\boldsymbol{y}}_k)$. Each parallelotope $\widehat{\boldsymbol{x}}_k$ is crossed from one input side to the opposite output side by one unique component of the curve. Each parallelotope $\widehat{\boldsymbol{y}}_k$ encloses the single solution of the curve on the output side of $\widehat{\boldsymbol{x}}_k$. The parallelotopes $\widehat{\boldsymbol{y}}_k$ enclose a unique solution and are therefore tiny (usually of width approximately 10^{-14}). The connection between two consecutive parallelotopes $\widehat{\boldsymbol{x}}_k$ and $\widehat{\boldsymbol{x}}_{k+1}$ is ensured since both enclose the certified solution within parallelotope $\widehat{\boldsymbol{y}}_k$. A special case is $\widehat{\boldsymbol{y}}_0$ which encloses the solution of the curve on the *input* side of $\widehat{\boldsymbol{x}}_1$. It is used to detect loops, i.e., connection between the first and last parallelotopes in the sequence $(\widehat{\boldsymbol{x}}_k)$. The correctness, halting and asymptotic convergence of ParCont are investigated in Subsection 4.3.2. Finally, the limitations of ParCont are emphasized in Subsection 4.3.3.

4.3.1 Algorithm Description

Algorithm 7 gives a formal definition of ParCont. It consists of one main loop, which iteratively builds a trial parallelotope whose length corresponds to the step size and tries inflating it using Algorithm 6.

In case of successful certification of the parallelotope, a tiny parallelotope enclosing the unique solution within its output side is built using Algorithm 5, which is used to certify the connection with the next parallelotope. Then several checks are performed in order to validate the absence of

Algorithm 7: ParCont

Input: $F : \mathbb{R}^n \to \mathbb{R}^{n-1}$; $\boldsymbol{x}^{\text{init}} \in \mathbb{IR}^n$; $\tilde{x}_0 \in \mathbb{R}^n$; $d \in \{-1, 1\}$
Parameters: $\underline{h}, h \in \mathbb{R}$ (s.t. $0 < \underline{h} < h$) ; $\alpha, \beta \in \mathbb{R}$ (s.t. $0 < \alpha < 1 < \beta$)

7.1 $k \leftarrow 1$;
7.2 *stop* $\leftarrow$ **false**;
7.3 **while** $\neg$*stop* **do**
7.4 $\quad \widehat{\boldsymbol{x}}_k \leftarrow (C_k, \boldsymbol{w}_k, \tilde{x}_{k-1})$ using Eq. (4.3) ;
7.5 $\quad (\mathit{success}, \widehat{\boldsymbol{x}}_k) \leftarrow$ N–Inflate$(F, \widehat{\boldsymbol{x}}_k)$;
7.6 $\quad$ **if** *success* **then**
7.7 $\quad\quad$ **if** $k = 1$ **then** $\widehat{\boldsymbol{y}}_0 \leftarrow$ Contract$\big(F, \text{InputSide}(\widehat{\boldsymbol{x}}_k)\big)$;
7.8 $\quad\quad \widehat{\boldsymbol{y}}_k \leftarrow$ Contract$\big(F, \text{OutputSide}(\widehat{\boldsymbol{x}}_k)\big)$;
7.9 $\quad$ **end**
7.10 $\quad$ *valid* $\leftarrow$ *success* **and** (4.11a) **and** (4.11b) **and** (4.11c);
7.11 $\quad h \leftarrow$ Step size updating using (4.14);
7.12 $\quad$ **if** *valid* **then**
7.13 $\quad\quad \tilde{x}_k \leftarrow \text{mid}(\widehat{\boldsymbol{y}}_k)$;
7.14 $\quad\quad k \leftarrow k + 1$;
7.15 $\quad$ **end**
7.16 $\quad$ *stop* $\leftarrow$ ($\neg$(4.11a)) **or** (4.15a) **or** (4.15b) **or** (4.15c) **or** (4.15d);
7.17 **end**
7.18 **return** $(\widehat{\boldsymbol{y}}_0, \ldots, \widehat{\boldsymbol{y}}_{k-1})$, $(\widehat{\boldsymbol{x}}_1, \ldots, \widehat{\boldsymbol{x}}_{k-1})$;

backtrack, the inclusion within the initial domain and the looping status. The step size is eventually updated, and tests decide whether to perform a new step or not.

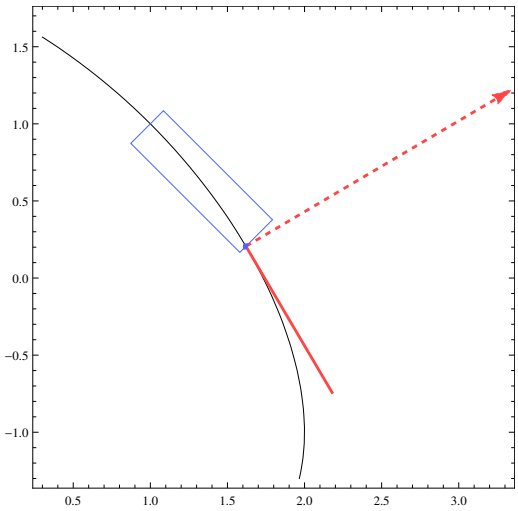
Notations used within the algorithm : The parallelotope $\widehat{\boldsymbol{x}}_k$ characteristic matrix and box are respectively denoted by C_k and $\boldsymbol{w}_k = (\boldsymbol{u}_k, \boldsymbol{v}_k)$. The parallelotope $\widehat{\boldsymbol{y}}_k$ is tiny and contains the unique solution within the output side of the parallelotope $\widehat{\boldsymbol{x}}_k$, which is denoted by y_k. For $k \geq 1$, the (approximate) midpoint of the parallelotope $\widehat{\boldsymbol{y}}_k$ is denoted $\tilde{x}_k$, so $\tilde{x}_k$ and y_k are very close to each other. Provided that $\tilde{x}_0$ is an accurate enough approximate solution, it is also very close to y_0. The distance between of $\tilde{x}_k$ and y_k for $k \geq 1$ has no impact on the correctness of the algorithm, but it provides a useful picture of its normal behavior.

4.3.1.1 Trial Parallelotope Construction and Inflation

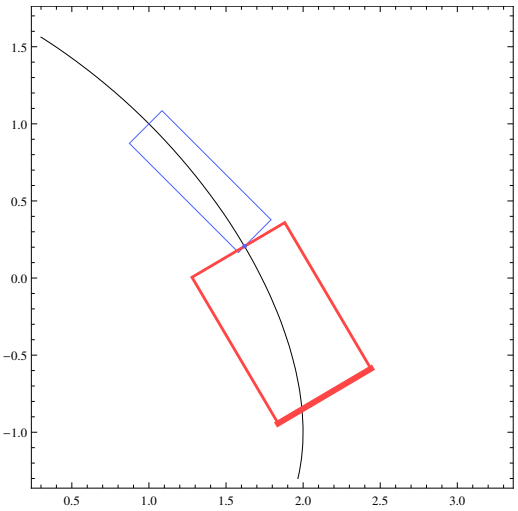
Line 7.4 of Algorithm 7 builds a trial parallelotope $\widehat{\boldsymbol{x}}_k = (C_k, \boldsymbol{w}_k, \tilde{x}_{k-1})$, which intends to be a segment tangent to the curve at $\tilde{x}_{k-1}$ of length h. In practice, it is approximately tangent (due to acceptable rounding errors in the derivatives evaluation at $\tilde{x}_{k-1}$ and the approximate null space computation), and is slightly enlarged so as to rigorously contain the tiny parallelotope $\widehat{\boldsymbol{y}}_{k-1}$ and its solution y_{k-1} :

$$C_k := \begin{pmatrix} F'(\tilde{x}_{k-1}) \\ \ker(F'(\tilde{x}_{k-1}))^T \end{pmatrix}^{-1} \tag{4.3a}$$

$$\boldsymbol{w}_k := \begin{cases} \big(0, \ldots, 0, [0, h]\big) & \text{if } k = 1 \\ \big(C_k^{-1}(\square\widehat{\boldsymbol{y}}_{k-1} - \tilde{x}_{k-1})\big) \vee \big(0, \ldots, 0, [0, h]\big) & \text{if } k \geq 2, \end{cases} \tag{4.3b}$$



(a) In thick red lines, the trial Parallelotope $\widehat{\boldsymbol{x}}_2$ ($F'(\tilde{x}_1)$ is represented by the dashed arrow).



(b) In thick red lines, the inflated Parallelotope $\widehat{\boldsymbol{x}}_2$. In bold, its output side.

Figure 4.1 – Construction of the parallelotope in Example 4.3.1.

where $\ker(F'(\tilde{x}_{k-1}))$ is one single vector of norm 1 and oriented so that the sign of $\det C_k^{-1}$ is $d \in \{-1, 1\}$ (in order to maintain the same direction for the continuation, see [1]). The characteristic box $\boldsymbol{w}_k$ built using (4.3b) entails that the trial parallelotope $\widehat{\boldsymbol{x}}_k$ actually contains the unique solution within $\widehat{\boldsymbol{y}}_{k-1}$ (indeed $\boldsymbol{w}_k \supseteq C_k^{-1}(\square\widehat{\boldsymbol{y}}_{k-1} - \tilde{x}_{k-1})$ implies $\square\widehat{\boldsymbol{y}}_{k-1} \subseteq \widehat{\boldsymbol{x}}_k$).

Floating point computations can be used in (4.3a), since these components of $\widehat{\boldsymbol{x}}_k$ can be computed approximately. However, interval computations have to be used in (4.3b) so that the last solution of the previous parallelotope provably belongs to the current parallelotope. This entails using a certified enclosure of the inverse of C_k, e.g. using the enclosure defined by (4.13) in Section 4.3.1.3. Note also that in case the previous iteration failed, it is not necessary to recompute (4.3a). This is not shown explicitly in Algorithm 7 for the sake of simplicity.

Example 4.3.1 – Consider the system (2.31), whose solution set is an ellipse, as in Example 2.3.3 (p. 30). Applying Algorithm 7 with step size $h = 1$ starting from $\tilde{x}_0 = (1, 1)$ results in the parallelotope $\widehat{\boldsymbol{x}}_1 = (C_1, \boldsymbol{w}_1, \tilde{x}_0)$, which has been computed in Example 2.3.3, together with a tiny parallelotope $\widehat{\boldsymbol{y}}_1 = (C_1, (\boldsymbol{u}_1, 1), \tilde{x}_0)$, where $\boldsymbol{u}_1 = [-0.5227744249483427, -0.5227744249483349]$, containing the unique solution in its output side. Also, $\tilde{x}_1 = \mathrm{mid}(\widehat{\boldsymbol{y}}_1) = (1.6199777103618245, 0.2057641479887291)$, which is a good approximation of the solution within $\widehat{\boldsymbol{y}}_1$, and the step size is increased, say to $h = 1.1$.

A new loop of Algorithm 7 starts by evaluating the derivatives at x_1, using standard double computations, leading to $F'(x_1) \approx (1.72286, 1.01575)$, whose normalized and correctly oriented null space is approximately $(-0.507877, 0.86143)$. This leads to the trial parallelotope $\widehat{\boldsymbol{x}}_2 = (C_2, \boldsymbol{w}_2, \tilde{x}_1)$ with

$$C_2 \approx \begin{pmatrix} 0.215357 & 0.507877 \\ 0.126969 & -0.86143 \end{pmatrix} \tag{4.4}$$

and

$$\begin{aligned} \boldsymbol{w}_2 &= (0,[0,h]) \vee \left(C_2^{-1}(\square\widehat{\boldsymbol{y}}_1 - \tilde{x}_1)\right) && (4.5)\\ &= ([-8.12\times10^{-15}, 7.90^{-15}], [-1.87\times10^{-15}, 1.1]). && (4.6)\end{aligned}$$

This trial parallelotope, which is depicted in thick red lines in Figure 4.1(a), contains the unique solution y_1 that lies in the output side of $\widehat{\boldsymbol{x}}_1$.

The trial parallelotope contains the previous parallelotope output side solution y_{k-1} for $v \approx 0$. It is inflated aiming to enclose a unique solution for all values of $v \in \boldsymbol{v}_k$. This is done using Algorithm 6 at Line 7.5, which furthermore ensures by Theorem 4.2.1 that the inflated parallelotope also contains y_{k-1} (since no solution can be lost during Algorithm 6).

Example 4.3.2 – Algorithm 6 successfully inflates the trial parallelotope of Example 4.3.1 giving rise to $\widehat{\boldsymbol{x}}_2 = (C_2, \boldsymbol{w}_2, \tilde{x}_1)$ with

$$\boldsymbol{w}_2 = ([-1.583, 1.211], [-1.87\times10^{-15}, 1.1]), \tag{4.7}$$

which is depicted in Figure 4.1(b), showing it indeed contains a unique solution for each $v \in \boldsymbol{v}_2$.

The trial characteristic box built using (4.3b) is generally very thin, which entails too many iterations of Algorithm 6. Making the reasonable assumption that two consecutive parallelotopes will be similar, this can be sensibly improved at low cost using the previous parallelotope characteristic box to enlarge it :

$$\boldsymbol{w}_k \leftarrow (\boldsymbol{u}_k \vee \boldsymbol{u}_{k-1}, \boldsymbol{v}_k). \tag{4.8}$$

This heuristic is used within Line 7.4, just after (4.3b) for $k \geq 2$. It is used only when the previous parallelotope was successfully build, since a failure entails a decrease of the step size, for which the previous characteristic box is most probably too large.

4.3.1.2 Output Side Parallelotopes

Once the parallelotope $\widehat{\boldsymbol{x}}_k$ is successfully built, the solution on its output side has to be enclosed sharply inside $\widehat{\boldsymbol{y}}_k$. To this end, $\widehat{\boldsymbol{y}}_k$ is computed at Line 7.8 by applying Contract (Algorithm 5) to contract the parallelotope corresponding to the output side of $\widehat{\boldsymbol{x}}_k$. During the first iteration of Algorithm 7, the input solution is similarly enclosed inside $\widehat{\boldsymbol{y}}_0$ at Line 7.7. Parallelotopes corresponding to the input and output sides of a given parallelotope are determined as follows :

$$\begin{aligned} \mathrm{InputSide}(C,(\boldsymbol{u},\boldsymbol{v}),\tilde{x}) &= (C,(\boldsymbol{u},\underline{v}),\tilde{x}) && (4.9)\\ \mathrm{OutputSide}(C,(\boldsymbol{u},\boldsymbol{v}),\tilde{x}) &= (C,(\boldsymbol{u},\overline{v}),\tilde{x}). && (4.10)\end{aligned}$$

By Theorem 5.2.2 and Theorem 5.2.5 of [90], addressing respectively the Krawczyk and Hansen-Sengupta operators, these contractions allow computing arbitrarily sharp enclosure of the solutions, according to the computational precision.

Example 4.3.3 – The output side of the parallelotope (4.7) built in Example 4.3.2 is $(C_2, ([-1.583, 1.211], 1.1), \tilde{x}_1)$, which is depicted in bold in Figure 4.1(b). It is contracted at Line 7.8 to $\widehat{\boldsymbol{y}}_2 = (C_2, ([-0.86220253621524, -0.86220253621520], 1.1), \tilde{x}_1)$. The resulting parallelotope is a very sharp enclosure of the unique solution within the output side of $\widehat{\boldsymbol{x}}_2$.

4.3.1.3 Validation of Successfully Inflated Parallelotopes

The parallelotope successfully inflated has to be validated at Line 7.10 for ensuring no backtrack on the curve happens (4.11a) (starting from iteration $k = 2$), for deciding the looping status of the curve (4.11b) (the parallelotope is valid if the presence of a loop is either rigorously disproved by $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_k = \emptyset$ or proved by $\widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_k$, starting from iteration $k = 2$) and for checking its inclusion within the initial domain (4.11c) (the parallelotope is valid if rigorously proved to either remain fully inside the domain, or to leave the domain) :

$$k = 1 \ \textbf{or} \ \big(\widehat{\boldsymbol{y}}_k \cap \widehat{\boldsymbol{x}}_{k-1} = \emptyset \ \textbf{and} \ \widehat{\boldsymbol{y}}_{k-2} \cap \widehat{\boldsymbol{x}}_k = \emptyset\big), \tag{4.11a}$$

$$k = 1 \ \textbf{or} \ \widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_k = \emptyset \ \textbf{or} \ \widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_k, \tag{4.11b}$$

$$\square\widehat{\boldsymbol{y}}_k \cap \boldsymbol{x}^{\text{init}} = \emptyset \ \textbf{or} \ \square\widehat{\boldsymbol{x}}_k \subseteq \boldsymbol{x}^{\text{init}}. \tag{4.11c}$$

Testing whether two parallelotopes $\widehat{\boldsymbol{x}}$ and $\widehat{\boldsymbol{x}}'$ do not intersect is a simple linear program. Alternatively, we will use the following sufficient condition :

$$C^{-1}\big((C'\boldsymbol{w}' + \tilde{x}') - \tilde{x}\big) \cap \boldsymbol{w} = \emptyset \implies \widehat{\boldsymbol{x}} \cap \widehat{\boldsymbol{x}}' = \emptyset. \tag{4.12}$$

A certified enclosure of the inverse of C has to be used in (4.12), e.g.

$$C^{-1} \in \tilde{C}^{-1} + \frac{\nu}{1-\nu}[-|\tilde{C}^{-1}|, |\tilde{C}^{-1}|], \tag{4.13}$$

with $\nu = \|\tilde{C}^{-1}C - I\|_\infty$, $\tilde{C}^{-1}C - I$ being computed using interval arithmetic in order to provide a rigorous upper bound on the norm, and where $\tilde{C}^{-1}$ designates an approximate inverse of C (see Theorem 4.1.11 in [90]). The sufficient condition (4.12) is quite accurate when $\widehat{\boldsymbol{x}}'$ is very small with respect to $\widehat{\boldsymbol{x}}$.

Remark 4.3.1 – Since the correct direction is used by checking the sign of the determinant of C_k, (4.11a) is always true provided that C_k is computed with good enough accuracy (evaluation of the derivatives at $\tilde{x}_{k-1}$, kernel computation, and matrix inversion). However, these tests are included in Algorithm 7 in order to enforce its correctness regardless of the accuracy of C_k and of its determinant sign computation.

Example 4.3.4 – The parallelotope built in Example 4.3.2 is validated as follows : First there is no backtrack since $\widehat{\boldsymbol{y}}_2 \cap \widehat{\boldsymbol{x}}_1 = \emptyset$ and $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_2 = \emptyset$ (see Figure 4.1(a)). Second, there is no loop since $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_2 = \emptyset$. Finally, (4.11c) depends on the initial domain $\boldsymbol{x}^{\text{init}}$, but will succeed provided that it is large enough.

4.3.1.4 Step Size Update

A step size adjustment is performed by updating h at Line 7.11. The only requirement on a step size adjustment strategy is that it is *fair*, i.e., infinitely many consecutive unsuccessful iterations entail that h converges toward zero. ParCont implements the following simple step size control strategy :

$$h \leftarrow \begin{cases} \beta h & \text{if } \textit{valid} \\ \alpha h & \text{otherwise.} \end{cases} \tag{4.14}$$

This strategy is obviously fair provided that $0 < \alpha < 1$, while $1 < \beta$ allows increasing h in case of validated success. A *pessimistic* strategy with small values for α and β, which intends

making smaller steps with easy success of N–Inflate, sounds realistic taking into account that calls to N–Inflate have the same cost whether they succeed or not. Experiments with various value combinations on several problems have confirmed this hypothesis and they have shown that $\alpha = 0.5$ and $\beta = 1.1$ represent an efficient setting.

4.3.1.5 Stopping Criteria

Five stopping conditions can stop the algorithm : First, the algorithm stops whenever (4.11a) is false, i.e. the absence of backtrack could not be proved. In this case, the parallelotope was not validated, but decreasing the step size will not improve the situation, hence the algorithm has to stop. The last four stopping conditions are given below :

$$\textit{valid} \ \textbf{and} \ k \geq 2 \ \textbf{and} \ \widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_k \tag{4.15a}$$

$$\textit{valid} \ \textbf{and} \ \square\widehat{\boldsymbol{y}}_k \cap \boldsymbol{x}^{\text{init}} = \emptyset \tag{4.15b}$$

$$\textit{valid} \ \textbf{and} \ \inf(\textbf{dist}(\square\widehat{\boldsymbol{y}}_{k-1}, \square\widehat{\boldsymbol{y}}_k)) \leq \underline{h} \tag{4.15c}$$

$$h \leq \underline{h}. \tag{4.15d}$$

Whenever (4.15a) is satisfied (starting from iteration $k = 2$), the curve is proved to loop (see the proof of Theorem 4.3.2) and the algorithm stops as a complete connected component of the followed curve has been rigorously covered. The condition (4.15b) allows stopping the algorithm as soon as the curve is proved to leave the initial domain. The conditions (4.15c) and (4.15d) enforce a minimal step size, respectively enforcing a minimal actual step size and avoiding infinitely many successive failures. These last two conditions stand for guaranteeing the termination of the algorithm (see Corollary 4.3.3 and its proof), when failing covering the manifold because e.g. of a singularity or a too strong curvature.

4.3.2 Properties of the Algorithm

The three statements provided in this subsection involve real interval arithmetic. The first two, related to the correctness and termination of Algorithm 7, remain correct in outwardly rounded floating interval arithmetic. The third statement is about the asymptotic convergence of Algorithm 7, which is valid only in real interval arithmetic, but provides a good insight on the normal behavior of the floating point implementation of Algorithm 7. All parameters of the algorithms are assumed to lie inside the domains provided in each algorithm description.

4.3.2.1 Correctness

Theorem 4.2.1 shows that each validated parallelotope $\widehat{\boldsymbol{x}}_k$ is crossed by a piece of solution curve $\gamma_k(t)$. The following Theorem 4.3.2 is dedicated to show that they can be glued together to a global solution curve $\gamma(t)$ (using the solutions y_k in parallelotopes $\widehat{\boldsymbol{y}}_k$ that are common to successive parallelotopes $\widehat{\boldsymbol{x}}_k$), hence ensuring the connectivity of the continuation, and that there is no unwanted backtrack during the continuation process (using the validating conditions (4.11a)). Before stating this theorem, we need to prove a property on parametric curves solving the system $F(x) = 0$.

Lemma 4.3.1. *Let $\alpha : [0, L_\alpha] \to \mathbb{R}^n$ and $\beta : [0, L_\beta] \to \mathbb{R}^n$ be continuously differentiable curves (whenever t is some interval endpoint, right or left derivatives are considered), and $F : \mathbb{R}^n \to \mathbb{R}^{n-1}$ be continuously differentiable. Suppose that :*

(i) For all t *in* $[0, l]$ *with* $l = \min\{L_\alpha, L_\beta\}$ *:*

(a) $F(\alpha(t)) = 0$ *and* $F(\beta(t)) = 0$ *;*

(b) $F'(\alpha(t))$ *and* $F'(\beta(t))$ *are full (row) rank ;*

(c) $\|\alpha'(t)\| = 1$ *and* $\|\beta'(t)\| = 1$.

(ii) $\alpha(0) = \beta(0)$ *and* $\alpha'(0) = \beta'(0)$.

Then, $\alpha(t) = \beta(t)$*, and thus* $\alpha'(t) = \beta'(t)$*, holds for all* $t \in [0, l]$.

Proof.

First, we prove that if $\alpha(t^*) = \beta(t^*) = x^*$ and $\alpha'(t^*) = \beta'(t^*)$ both hold for some $t^* \in (0, l)$ then $\alpha(t) = \beta(t)$ (and therefore $\alpha'(t) = \beta'(t)$) holds in a neighborhood of t^*. Since $F'(x^*)$ is full rank, it contains a nonsingular $n-1 \times n-1$ sub matrix. Without loss of generality, suppose that the last $n-1$ columns of $F'(x^*)$ form a nonsingular square matrix, i.e. $F'(x^*) = (a|A)$ with $A \in \mathbb{R}^{n-1 \times n-1}$ nonsingular and $a \in \mathbb{R}^{n-1}$. Therefore, by the implicit function theorem, there exists a neighborhood $\mathcal{U}$ of x^* and a differentiable function $\phi : \mathbb{R} \longrightarrow \mathbb{R}^{n-1}$ such that $x \in \mathcal{U}$ and $F(x) = 0$ is equivalent to $x = (x_1, \phi(x_1))$.

Since $F(\alpha(t)) = 0$ we have $(a|A)\,\alpha'(t^*) = 0$, and thus $\alpha'_{>1}(t^*) = -\alpha'_1(t^*)A^{-1}a$ (where the vector α' is split into $(\alpha_1, \alpha'_{>1})$). Therefore $\alpha'_1(t^*) = 0$ entails $\alpha'(t^*) = 0$ which contradicts $\|\alpha'(t^*)\| = 1$. As a conclusion, $\alpha'_1(t^*) \neq 0$ and the inverse function theorem proves that exists a neighborhood $\mathcal{U}_1$ of x_1^* and a differentiable function α_1^{-1} defined inside $\mathcal{U}_1$ such that $\alpha_1(\alpha_1^{-1}(x_1)) = x_1$. Since α, α_1^{-1} and β are continuous, there exists a neighborhood $\mathcal{T}^*$ of t^* such that $\beta_1(\mathcal{T}^*) \subseteq \mathcal{U}_1$, $\alpha(\alpha_1^{-1}(\beta_1(\mathcal{T}^*))) \subseteq \mathcal{U}$ and $\beta(\mathcal{T}^*) \subseteq \mathcal{U}$. Fix an arbitrary $t \in \mathcal{T}^*$ and write $u := \alpha(\tau(t))$ with $\tau(t) := \alpha_1^{-1}(\beta_1(t))$ and $v := \beta(t)$. Then $u_1 = v_1$ while $F(u) = 0$ and $F(v) = 0$ entail $u = (u_1, \phi(u_1))$ and $v = (v_1, \phi(v_1))$. Therefore $u = v$, i.e. $\alpha(\tau(t)) = \beta(t)$.

Differentiating $\alpha(\tau(t)) = \beta(t)$, we obtain $\alpha'(\tau(t))\tau'(t) = \beta'(t)$ for all $t \in \mathcal{T}^*$. Since both $\|\alpha'(\tau(t))\| = 1$ and $\|\beta'(t)\| = 1$, we have $|\tau'(t)| = 1$. Note that $\alpha_1(t^*) = \beta_1(t^*) = x_1^*$ entails $\tau(t^*) = t^*$, and then and $\alpha'(t^*) = \beta'(t^*)$ entails $\tau'(t^*) = 1$. Since τ' is continuous we have $\tau'(t) = 1$ for all $t \in \mathcal{T}^*$, and eventually $\tau(t) = t$ and $\alpha(t) = \beta(t)$ (and therefore $\alpha'(t) = \beta'(t)$).

This argument based on the implicit function theorem also holds for $t = 0$ and $t = l$, and furthermore proves that α and β can be extended to a neighborhood of $[0, l]$ with the same properties.

Second, define $\mathcal{T} = \{t \in [0, l] : \alpha(t) = \beta(t), \alpha'(t) = \beta'(t)\}$. By the first part of the proof, $\mathcal{T}$ is open in the topology of $[0, l]$. By the continuity of $\alpha(t) - \beta(t)$ and its derivative, $\mathcal{T}$ is also closed. Since $[0, l]$ is connected, it has only two clopen subsets, namely $\emptyset$ and $[0, l]$. Since $0 \in \mathcal{T}$, we finally have $\mathcal{T} = [0, l]$.

□

In other words, Lemma 4.3.1 states that two unit-speed curves of solutions to $F(x) = 0$ having the same value and derivative at their starting point are equivalent on their common domain of definition. The theorem asserting the correctness of Algorithm 7 can be stated as follows.

Theorem 4.3.2. *Suppose* $K \geq 1$. *Let* $(\widehat{\boldsymbol{x}}_1, \ldots, \widehat{\boldsymbol{x}}_K)$ *and* $(\widehat{\boldsymbol{y}}_0, \ldots, \widehat{\boldsymbol{y}}_K)$ *be some parallelotopes sequences produced by Algorithm 7. There exist* $L > 0$ *and* $\gamma : [0, L] \to \cup_{k=1}^{K} \widehat{\boldsymbol{x}}_k$ *such that :*

(i) $\forall t \in [0, L]$, $F(\gamma(t)) = 0$ *and* $F'(\gamma(t))$ *is full rank.*

(ii) γ *is differentiable in the interior of* $[0, L]$*, and* $\|\gamma'(t)\| = 1$.

(iii) Either $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_K = \emptyset$ *or* $\widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_K$*. Furthermore :*

(iii-a) If $K = 1$ *or* $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_K = \emptyset$ *then* γ *is injective.*

(iii-b) If $K > 1$ *and* $\widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_K$ *then* γ *is an injective loop* [2].

Proof.

Let $\mu_k : \boldsymbol{v}_k \to \boldsymbol{u}_k$ be the function given by Theorem 4.2.1 applied to the parallelotope $\widehat{\boldsymbol{x}}_k$, and define $\tilde{\gamma}_k : \boldsymbol{v}_k \to \widehat{\boldsymbol{x}}_k$ by $\tilde{\gamma}_k(v) = C_k(\mu_k(v), v) + \tilde{x}_{k-1}$, which is also differentiable in the interior of $\boldsymbol{v}_k$. Hence, $x \in \widehat{\boldsymbol{x}}_k \wedge F(x) = 0 \iff \exists v \in [\underline{v}_k, \overline{v}_k], x = \tilde{\gamma}_k(v)$. Furthermore, $\tilde{\gamma}_k'(v) \neq 0$ obviously holds, hence $\tilde{\gamma}_k$ can be reparametrized to $\gamma_k : [0, l_k] \to \widehat{\boldsymbol{x}}_k$ with unit speed (i.e. $\|\gamma_k'(t)\| = 1$), so l_k is the length of the solution path that crosses $\widehat{\boldsymbol{x}}_k$. The functions γ_k obviously inherit the property

$$\forall x \in \widehat{\boldsymbol{x}}_k, F(x) = 0 \iff \exists t \in [0, l_k], x = \gamma_k(t). \tag{4.16}$$

Note that $\tilde{\gamma}_k$ is injective since C_k is nonsingular, and hence so is γ_k.

Now, γ is defined piecewise by connecting the γ_k using the solutions $y_k \in \widehat{\boldsymbol{y}}_k$ shared by consecutive parallelotopes : By construction, $y_{k-1} \in \widehat{\boldsymbol{x}}_k$ and $y_k \in \widehat{\boldsymbol{x}}_k$ for $k \in \{1, \ldots, K\}$. Hence $y_{k-1}, y_k \in \gamma_k([0, l_k])$, and $\underline{t}_k := \gamma_k^{-1}(y_{k-1})$ and $\overline{t}_k := \gamma_k^{-1}(y_k) = l_k$ are well defined. We now prove that $\underline{t}_k < \overline{t}_k$ holds for all $k \in \{1, \ldots, K\}$. For all such k, y_k is the output solution of the parallelotope $\widehat{\boldsymbol{x}}_k$ and hence $\underline{t}_k \leq \overline{t}_k$ holds. The equality $\underline{t}_k = \overline{t}_k$ contradicts (4.15d) and $\underline{h} > 0$ for $k = 1$, or contradicts $\widehat{\boldsymbol{y}}_k \cap \widehat{\boldsymbol{x}}_{k-1} = \emptyset$ in (4.11a) for $k \geq 2$ (since $\widehat{\boldsymbol{y}}_{k-1} \subseteq \widehat{\boldsymbol{x}}_{k-1}$). Define T_k for $k \in \{0, \ldots, K\}$ by $T_0 = 0$ and $T_k = T_{k-1} + \overline{t}_k - \underline{t}_k$, so $0 = T_0 < T_1 < \cdots < T_K$. The function $\gamma : [0, T_K] \to \cup_{k=1}^K \widehat{\boldsymbol{x}}_k$ is finally defined as follows : If $t \in [T_{k-1}, T_k[$ then $t + \underline{t}_k - T_{k-1} \in [\underline{t}_k, \overline{t}_k[$ and

$$\gamma(t) := \gamma_k(t + \underline{t}_k - T_{k-1}) \tag{4.17}$$

is well defined. As a consequence, γ is differentiable within $]0, T_K[\backslash\{T_1, \ldots, T_{K-1}\}$. For all $t \in [0, T_K]$, $\gamma(t) \in \cup_{k=1}^K \widehat{\boldsymbol{x}}_k$ and $F(\gamma(t)) = 0$ hold by construction, and $F'(\gamma(t))$ is full rank by Theorem 4.2.1, which prove *(i)* for any $0 < L \leq T_K$.

To prove *(ii)*, first note also that γ is continuous with $\gamma(T_k) = y_k$: Indeed $\gamma(T_k) = \gamma_{k+1}(\underline{t}_{k+1}) = y_k$ while

$$\lim_{t \to T_k^-} \gamma(t) = \lim_{t \to T_k^-} \gamma_k(t + \underline{t}_k - T_{k-1}) = \gamma_k(\overline{t}_k) = y_k. \tag{4.18}$$

Furthermore, γ is differentiable in $]0, T_K[$, so define $g_k^{\pm} := \gamma'(T_k^{\pm})$ (evaluations at $T_k^{\pm}$ corresponding to limits from above and below respectively). As a consequence, we just need to prove that $g_k^- = g_k^+$ for $k \in \{1, \ldots, K-1\}$. Since $F'(\gamma(t))\gamma'(t)$ and $\|\gamma'(t)\|$ are both continuous and respectively identically 0 and 1 inside $]0, T_K[\backslash\{T_1, \ldots, T_{K-1}\}$, both $F'(\gamma(T_k))g_k^{\pm} = 0$ and $\|g_k^{\pm}\| = 1$ hold. Therefore, $g_k^- = \pm g_k^+$ (since $\ker F'(\gamma(T_k))$ has dimension 1). Now suppose that $g_k^- = -g_k^+$. Both $\gamma(T_k + t) = \gamma_{k+1}(\underline{t}_k + t)$ and $\gamma(T_k - t) = \gamma_k(\overline{t}_k - t)$ hold inside $[0, \min\{T_{k+1} - T_k, T_k - T_{k-1}\}]$, and hence both are differentiable in the interior of this time domain. Furthermore, both have the same value (namely $\gamma(T_k)$) and derivative at $t = 0$ (since $g_k^- = -g_k^+$), so Lemma 4.3.1 proves that they are equal on the whole time domain. Therefore, $\gamma(T_k + \min\{T_{k+1} - T_k, T_k - T_{k-1}\}) = \gamma(T_k - \min\{T_{k+1} - T_k, T_k - T_{k-1}\})$. Finally, either

2. I.e. it is injective inside $[0, L[$, $\gamma(0) = \gamma(L)$, and $\gamma'(0^+) = \gamma'(L^-)$, evaluations at $t^{\pm}$ corresponding to one-sided limits

$T_{k+1} - T_k \leq T_k - T_{k-1}$ then $\gamma(T_{k+1}) = \gamma(T_k - (T_{k+1} - T_k))$ and $T_k - (T_{k+1} - T_k) \in [T_{k-1}, T_k]$, which contradicts $\widehat{\boldsymbol{y}}_{k+1} \cap \widehat{\boldsymbol{x}}_k = \emptyset$ in (4.11a). Or, $T_{k+1} - T_k > T_k - T_{k-1}$ then $\gamma(2T_k - T_{k-1}) = \gamma(T_{k-1})$ and $2T_k - T_{k-1} \in [T_k, T_{k+1}]$, which contradicts $\widehat{\boldsymbol{y}}_{k-1} \cap \widehat{\boldsymbol{x}}_{k+1} = \emptyset$ in (4.11a). Therefore $g_k^- = g_k^+$ and γ is eventually differentiable in $]0, T_K[$, proving *(ii)* for any $0 < L \leq T_K$.

The property *(iii)* is a direct consequence of the validating condition (4.11b). For *(iii-a)* and *(iii-b)*, first note that *(iii-a)* is trivial if $K = 1$. Thus suppose $K > 1$, and consider $0 \leq t_1 < t_2 \leq T_K$ such that $\gamma(t_1) = \gamma(t_2) = x_*$. As previously, note that $F'(x_*)$ being full rank, its kernel is one dimensional and hence $\gamma'(t_1) = \pm\gamma'(t_2)$ (left or right derivatives are considered whether $t_1 = 0$ or $t_2 = T_K$). Suppose first that $\gamma'(t_1) = -\gamma'(t_2)$. Then both $\gamma(t_1+t)$ and $\gamma(t_2-t)$ are defined on $[0, t_2-t_1]$, and have the same value and derivative at $t = 0$. Therefore, Lemma 4.3.1 proves that they are equal on $[0, t_2-t_1]$. As a consequence, $\gamma'(\frac{1}{2}(t_2+t_1)) = -\gamma'(\frac{1}{2}(t_2+t_1))$, which entails $\gamma'(\frac{1}{2}(t_2+t_1)) = 0$, a contradiction since $\|\gamma'(\frac{1}{2}(t_2+t_1))\| = 1$. Therefore $\gamma'(t_1) = \gamma'(t_2)$. Then both $\gamma(t_1-t)$ and $\gamma(t_2-t)$ are defined on $[0, t_1]$, and have the same value and derivative at $t = 0$. Therefore, Lemma 4.3.1 proves that they are equal on $[0, t_1]$, and in particular $\gamma(0) = \gamma(t_2-t_1)$ and $\gamma'(0^+) = \gamma'((t_2 - t_1)^-)$ (evaluations at $t^\pm$ corresponding to limits from above and below respectively). Now, $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_k = \emptyset$ is enforced by Algorithm 7 for all $k \in \{2, \ldots, K-1\}$, hence $\gamma(t_2 - t_1) \notin \widehat{\boldsymbol{x}}_k$ for any such k. Note furthermore that $t_2 - t_1 \notin [0, T_1]$ since otherwise $\gamma_1(0) = \gamma_1(t_2 - t_1)$, which is impossible since γ_1 is injective. Therefore, $t_2 - t_1 \in [T_{K-1}, T_K]$ so

$$y_0 = \gamma(0) = \gamma(t_2 - t_1) = \gamma_K(t_2 - t_1 + \underline{t}_k - T_{K-1}). \tag{4.19}$$

Two cases arise depending on the halting status of Algorithm 7 : First, if $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_K = \emptyset$ then $t_2-t_1 \in [T_{K-1}, T_K]$ is contradicted and γ is injective inside $[0, L]$ with $L = T_K$, hence proving *(iii-a)*. Second, if $\widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_K$ then $y_0 \in \widehat{\boldsymbol{x}}_K$ which entails $t_2-t_1 = \gamma_K^{-1}(y_0)+T_{K-1}-\underline{t}_k$ by (4.19), which has been proved to belong to $[T_{K-1}, T_K]$. So by defining $L = \gamma_K^{-1}(y_0)+T_{K-1}-\underline{t}_k \leq T_K$, we have γ injective inside $[0, L[$ and $\gamma(0) = \gamma(L)$, which eventually proves *(iii-b)*.

□

4.3.2.2 Termination

Define $\mathcal{R} := \{x \in \boldsymbol{x}^{\text{init}} : F(x) = 0, F'(x) \text{ full rank}\}$ and $\mathcal{R}(y_0)$ as the connected component of $\mathcal{R}$ that contains y_0. Three cases arise when the algorithm terminates at iteration $k = K + 1$ depending on the condition that has stopped the algorithm : First, if (4.15a) has stopped Algorithm 7 then $\mathcal{R}(y_0)$ is proved to be a loop. Second, if (4.15b) has stopped Algorithm 7 then the last solution y_k is proved to be outside the domain $\boldsymbol{x}^{\text{init}}$, and therefore Algorithm 7 has fully enclosed $\mathcal{R}(y_0)$ from y_0 to one of its sides, depending on the direction chosen for the continuation. The second part of $\mathcal{R}(y_0)$ can be computed by starting again Algorithm 7 from y_0 in the opposite direction. Finally, if (4.15d) has stopped Algorithm 7 then either a singularity or a too strong curvature prematurely stopped the algorithm.

Since γ has a unit speed parametrization, L is the length of the 1-manifold between $\gamma(0)$ and $\gamma(L)$, and the following Corollary can be proved.

Corollary 4.3.3. *If the step size update is fair and $\mathcal{R}(y_0)$ has a finite length then Algorithm 7 terminates.*

Proof.

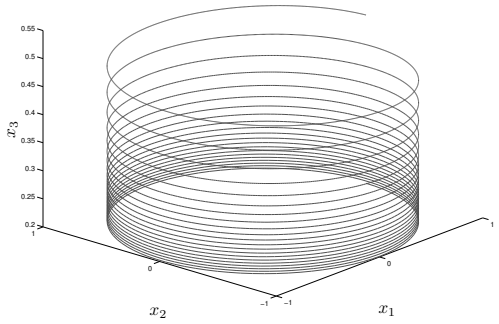


Figure 4.2 – Bounded manifold of infinite length

Suppose Algorithm 7 does not terminate. Since Algorithm 6 and Algorithm 5 terminate, Algorithm 7 does not terminate if it executes infinitely many times the main while loop. Thus the algorithm is either producing infinitely many consecutive failed iterations or infinitely many successful ones (separated by finitely many failures).

If Algorithm 7 is producing infinitely many consecutive failures, then the fairness of the step size control implies that h converges to zero. However, this contradicts the stopping criteria (4.15d), since $\underline{h} > 0$. Hence, Algorithm 7 cannot produce infinitely many failures, and therefore produces infinitely many successful iterations.

At iteration $K \geq 2$, Theorem 4.3.2 can be applied : there exists a curve $\gamma : [0, L_K] \to \cup_{k=1}^{k} \widehat{\boldsymbol{x}}_k$, with L_K the length of the curve between y_0 and y_K, where $F'(\gamma(t))$ is full rank, $\gamma(t)$ connected to y_0 and $\gamma(t)$ is in $\boldsymbol{x}^{\text{init}}$ for all $t \in [0, L_K]$. Therefore, $\gamma(t) \in \mathcal{R}$ for all $t \in [0, L_K]$. The stopping criteria (4.15c) ensures that the length of the manifold between two consecutive certified solutions y_{i-1} and y_i is greater than $\|y_i - y_{i-1}\| \geq \underline{h}$ (the length of the curve between two points is longer than the length of the direct line connecting them, which is bounded by the algorithm). Hence, $L_K \geq K\underline{h}$. Therefore, with infinitely many successful iterations, K tends to infinity and so does the length of the curve L_K. This contradicts the assumption that $\mathcal{R}$ has a finite length.

□

Algorithm 7 may not terminate whenever the regular manifold connected to y_0 has an infinite length. For example, consider the system $F(x) = (x_1 - \cos\left(2\pi/x_3^2\right), x_2 - \sin\left(2\pi/x_3^2\right)) = 0$ whose solution manifold is depicted on Figure 4.2. The manifold is a bounded spiral that collapses to a circle toward $x_3 = 0$, without reaching it.

However, when finite precision floating point arithmetic is used and $\boldsymbol{x}^{\text{init}}$ is bounded, such a bounded curve with infinite length has some accumulation point outside the curve, and parallelotopes with floating point characteristics will eventually contain some singular points entailing the stop of Algorithm 7.

4.3.2.3 Asymptotic Convergence

The previous two theorems do not state anything about the ability of Algorithm 7 to actually enclose the curve. Indeed, Algorithm 7 may stop failing to build any parallelotope because $\tilde{x}_0$

is too close to a singularity. The following analysis shows that under appropriate hypotheses, Algorithm 7 actually succeeds in computing the component connected to y_0.

It is conducted for the simplest version of Algorithm 7 : The Krawczyk operator is used since it is easier to analyze than the Hansen-Sengupta operator (the latter being known to be more efficient, see [90]). Furthermore, derivatives are all evaluated on the interval hull of the parallelotope. As a consequence, the Newton operator has here the following form :

$$\mathbf{K}(F, \widehat{\boldsymbol{x}}) = (\mathrm{Id}_E - \boldsymbol{G})(\mathrm{mid}\boldsymbol{w}) - (\boldsymbol{A}(\boldsymbol{w}) - E)(\boldsymbol{w} - \mathrm{mid}\boldsymbol{w}), \tag{4.20}$$

where $\widehat{\boldsymbol{x}} = (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$, $\boldsymbol{A}(\boldsymbol{w}) = (\boldsymbol{A}_u, \boldsymbol{A}_v) = \boldsymbol{F}'(\square\widehat{\boldsymbol{x}})C$, $G(w) = F(Cw + \tilde{x})$, $E = (I \,|\, 0)$ and $\mathrm{Id}_E(w) = Ew$ is the truncated identity map associated to E.

For the asymptotic analysis, we neglect rounding errors due to the floating point arithmetic : We consider that C_k and $\boldsymbol{G}(\mathrm{mid}(\boldsymbol{w}))$ are computed with no rounding error, so $F'(\tilde{x}_{k-1})C_k$ is exactly the $n-1$ first lines of the identity matrix and $\boldsymbol{G}(\mathrm{mid}(\boldsymbol{w})) = G(\mathrm{mid}(\boldsymbol{w}))$. Furthermore, we consider that $\widehat{\boldsymbol{y}}_k = \tilde{x}_k$, and hence $F(\tilde{x}_k) = 0$, which is justified by the fact that Algorithm 5 converges unconditionally and very quickly to the unique solution contained in the parallelotopes input and output sides. As a consequence, for any trial parallelotope built using Equation (4.3) we have $F(\tilde{x}_k) = 0$ (the heuristic (4.8) is not considered here). Note furthermore that since $\widehat{\boldsymbol{y}}_{k-1} = \tilde{x}_{k-1}$, we have $\boldsymbol{w}_k = (0, \ldots, 0, [0, h])$ in (4.3b) at all steps.

Then, the following theorem shows that there exists a threshold step size h_*, which depends only on the Lipschitz constant of $\boldsymbol{F}'$ and the norm of C, under which the certification procedure implemented in Algorithm 6 succeeds.

Theorem 4.3.4. *Consider an initial parallelotope $\widehat{\boldsymbol{x}} = (C, (0, \ldots, 0, [0, h]), \tilde{x})$ built as in Algorithm 6, i.e. using Equation (4.3). Note that by hypothesis $1 \leq \delta$ and $0 < \chi$. Suppose that the interval extensions $\boldsymbol{F}'$ of the derivatives of F are λ-Lipschitz continuous [3], with $\lambda > 0$ [4], $\|C\| \leq \kappa$ and $\chi \leq \frac{1}{6\kappa^2\lambda\delta}$, and define h_* as the greatest root of the quadratic polynomial $\kappa^2\lambda\delta(2h)^2 + \chi = (2h)$:*

$$h_* := \frac{1 + \sqrt{1 - 4\kappa^2\lambda\delta\chi}}{4\kappa^2\lambda\delta}. \tag{4.21}$$

Note that h_* satisfies $\frac{1}{4\kappa^2\lambda\delta} \leq h_* \leq \frac{1}{2\kappa^2\lambda\delta}$. Finally suppose that $\left(\frac{3}{4}\right)^{\overline{k}} < \frac{\chi}{4h_*}$ (such a $\overline{k}$ exists since since $\chi > 0$) and $\overline{\mu} \geq \frac{3}{4}$. Then $h \leq h_*$ implies N–Inflate$(F, \widehat{\boldsymbol{x}})$ succeeds.

Proof.

Denote the kth box computed by Algorithm 6 by $\boldsymbol{u}_k$, and write generally $\boldsymbol{w} = (\boldsymbol{u}, [0, h])$ and $\widehat{\boldsymbol{x}} = (C, \boldsymbol{w}, \tilde{x})$, so $\boldsymbol{w}_k = (\boldsymbol{u}_k, [0, h])$ and $\widehat{\boldsymbol{x}}_k = (C, \boldsymbol{w}_k, \tilde{x})$. Then $\boldsymbol{u}_0 = 0$ and $\boldsymbol{u}_{k+1} = \mathbf{K}_{\delta,\chi}(\boldsymbol{u}_k)$ with

$$\begin{aligned} \mathbf{K}_{\delta,\chi}(\boldsymbol{u}) &= \mathrm{mid}(\mathbf{K}(F, \widehat{\boldsymbol{x}})) + \delta(\mathbf{K}(F, \widehat{\boldsymbol{x}}) - \mathrm{mid}(\mathbf{K}(F, \widehat{\boldsymbol{x}}))) + \chi[-1, 1] && (4.22) \\ &= (\mathrm{Id}_E - \boldsymbol{G})(\mathrm{mid}(\boldsymbol{w})) - \delta(\boldsymbol{A}(\boldsymbol{w}) - E)(\boldsymbol{w} - \mathrm{mid}(\boldsymbol{w})) + \chi[-1, 1], && (4.23) \end{aligned}$$

Since $\boldsymbol{F}'$ is λ-Lipschitz continuous, $\boldsymbol{A}(\boldsymbol{w})$ is $\kappa^2\lambda$-Lipschitz continuous. As a consequence,

$$\|\mathrm{wid}(\boldsymbol{A}(\boldsymbol{w}))\| \leq 2d\big(\boldsymbol{A}(\boldsymbol{w}), \boldsymbol{A}(\mathrm{mid}(\boldsymbol{w}))\big) \leq 2\kappa^2\lambda d(\boldsymbol{w}, \mathrm{mid}(\boldsymbol{w})) = \kappa^2\lambda\|\mathrm{wid}(\boldsymbol{w})\|. \tag{4.24}$$

3. The natural interval extension of an expression involving elementary functions that are Lipschitz continuous is Lipschitz continuous, see Theorem 2.1.1 of [90].

4. The limit case $\lambda = 0$, corresponding to a system of linear equations, is uninteresting and can be captured by considering arbitrarily small Lipschitz constants, entailing $h_* = \infty$ for nonsingular linear systems.

Now define $\mathcal{U} := \{\boldsymbol{u} \in \mathbb{IR}^{n-1} : 0 \in \boldsymbol{u} \wedge \|\mathrm{wid}(\boldsymbol{u})\| \leq h_*\}$, which is a complete metric space for the usual distance for intervals. We are going to apply the Banach fixed point theorem to $\mathbf{K}_{\delta,\chi}$ inside $\mathcal{U}$.

First, we prove that $\mathbf{K}_{\delta,\chi}(\mathcal{U}) \subseteq \mathcal{U}$ by considering an arbitrary $\boldsymbol{u} \in \mathcal{U}$ and proving that $\mathbf{K}_{\delta,\chi}(\boldsymbol{u}) \in \mathcal{U}$. Note that $0 \in \boldsymbol{u} \Longleftrightarrow c \in (C, (\boldsymbol{u}, [0,h]), \tilde{x})$, and by Theorem 4.2.1 $F(c) = 0$ implies $c \in (C, (\mathbf{K}_{\delta,\chi}(\boldsymbol{u}), [0,h]), \tilde{x})$. So eventually $0 \in \mathbf{K}_{\delta,\chi}(\boldsymbol{u})$. This furthermore entails $E = F'(c)C \in \boldsymbol{A}(\boldsymbol{w})$, and hence using (4.24)

$$\|A(\boldsymbol{w}) - E\| \leq \|\mathrm{wid}(\boldsymbol{A}(\boldsymbol{w}))\| \leq \kappa^2 \lambda \|\mathrm{wid}(\boldsymbol{w})\|. \tag{4.25}$$

Using standard rules for computing the radius of intervals, we obtain

$$\begin{aligned} \|\mathrm{wid}(\mathbf{K}_{\delta,\chi}(\boldsymbol{u}))\| &= \delta \|A(\boldsymbol{w}) - E\| \, \|\mathrm{wid}(\boldsymbol{w})\| + \chi & (4.26)\\ &\leq \kappa^2 \lambda \delta \|\mathrm{wid}(\boldsymbol{w})\|^2 + \chi & (4.27)\\ &\leq \kappa^2 \lambda \delta \, h_*^2 + \chi. & (4.28) \end{aligned}$$

Inequality (4.27) is a consequence of (4.25), and Inequality (4.28) is a consequence of $\|\mathrm{wid}(\boldsymbol{w})\| = \|\mathrm{wid}((\boldsymbol{u}, [0,h]))\|$, $\|\mathrm{wid}(\boldsymbol{u})\| \leq h_*$ and $h \leq h_*$. Now, recall that h_* is defined as the greatest root of the quadratic polynomial $\kappa^2\lambda\delta(2h)^2 + \chi = (2h)$. Since $\kappa^2\lambda\delta h_*^2 + \chi = \frac{1}{4}\big(\kappa^2\lambda\delta(2h_*)^2 + \chi\big) + \frac{3}{4}\chi$, we obtain $\|\mathrm{wid}(\mathbf{K}_{\delta,\chi}(\boldsymbol{u}))\| \leq \frac{1}{2}h_* + \frac{3}{4}\chi$. Finally, $\frac{3}{4}\chi \leq \frac{1}{8\kappa^2\lambda\delta} \leq \frac{1}{2}h_*$, eventually entailing $\|\mathrm{wid}(\mathbf{K}_{\delta,\chi}(\boldsymbol{u}))\| \leq h_*$.

Second, we prove that $\mathbf{K}_{\delta,\chi} : \mathcal{U} \to \mathcal{U}$ is contracting. Consider some arbitrary $\boldsymbol{u}, \boldsymbol{u}' \in \mathcal{U}$, and define $\tilde{\boldsymbol{u}} = \mathbf{K}_{\delta,\chi}(\boldsymbol{u})$ and $\tilde{\boldsymbol{u}}' = \mathbf{K}_{\delta,\chi}(\boldsymbol{u}')$. Then

$$d(\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{u}}') = \|\mathrm{mid}(\tilde{\boldsymbol{u}}) - \mathrm{mid}(\tilde{\boldsymbol{u}}')\| + \|\mathrm{rad}(\tilde{\boldsymbol{u}}) - \mathrm{rad}(\tilde{\boldsymbol{u}}')\|. \tag{4.29}$$

We bound the two summands separately. For the first summand,

$$\begin{aligned} \|\mathrm{mid}(\tilde{\boldsymbol{u}}) - \mathrm{mid}(\tilde{\boldsymbol{u}}')\| &= \|(\mathrm{Id}_E - G)(\mathrm{mid}(\boldsymbol{w})) - (\mathrm{Id}_E - G)(\mathrm{mid}(\boldsymbol{w}'))\| & (4.30)\\ &\leq \Big(\max_{\substack{\xi = (1-t)\mathrm{mid}(\boldsymbol{w}) \\ + t\mathrm{mid}(\boldsymbol{w}') \\ t \in [0,1]}} \|E - G'(\xi)\| \Big) \|\mathrm{mid}(\boldsymbol{w}) - \mathrm{mid}(\boldsymbol{w}')\|. & (4.31) \end{aligned}$$

Now, $\|E - G'(\xi)\| = \|G'(0) - G'(\xi)\| \leq \kappa^2\lambda\|\xi\| \leq \kappa^2\lambda h_* \leq \frac{1}{2}$, hence eventually

$$\|\mathrm{mid}(\tilde{\boldsymbol{u}}) - \mathrm{mid}(\tilde{\boldsymbol{u}}')\| \leq \tfrac{1}{2}\|\mathrm{mid}(\boldsymbol{w}) - \mathrm{mid}(\boldsymbol{w}')\|. \tag{4.32}$$

For the second summand, $\|\mathrm{rad}(\tilde{\boldsymbol{u}}) - \mathrm{rad}(\tilde{\boldsymbol{u}}')\| = \delta\||\boldsymbol{A}(\boldsymbol{w}) - E|\mathrm{rad}(\boldsymbol{w}) - |\boldsymbol{A}(\boldsymbol{w}') - E|\mathrm{rad}(\boldsymbol{w}')\| + \chi - \chi$, which, by adding the vector $0 = -|\boldsymbol{A}(\boldsymbol{w}) - E|\mathrm{rad}(\boldsymbol{w}') + |\boldsymbol{A}(\boldsymbol{w}) - E|\mathrm{rad}(\boldsymbol{w}')$ within the norm, and using the triangular inequality, is less than

$$\begin{aligned} & \delta\||\boldsymbol{A}(\boldsymbol{w}) - E|(\mathrm{rad}(\boldsymbol{w}) - \mathrm{rad}(\boldsymbol{w}'))\| + \delta\|\big(|\boldsymbol{A}(\boldsymbol{w}) - E| - |\boldsymbol{A}(\boldsymbol{w}') - E|\big)\mathrm{rad}(\boldsymbol{w}')\| & (4.33)\\ \leq\ & \delta\|\boldsymbol{A}(\boldsymbol{w}) - E\|\|\mathrm{rad}(\boldsymbol{w}) - \mathrm{rad}(\boldsymbol{w}')\| + \delta d\big(\boldsymbol{A}(\boldsymbol{w}) - E, \boldsymbol{A}(\boldsymbol{w}') - E\big)\|\mathrm{rad}(\boldsymbol{w}')\| & (4.34)\\ \leq\ & \delta\kappa^2\lambda\|\mathrm{wid}(\boldsymbol{w})\|\|\mathrm{rad}(\boldsymbol{w}) - \mathrm{rad}(\boldsymbol{w}')\| + \delta\kappa^2\lambda\|\mathrm{rad}(\boldsymbol{w}')\| d(\boldsymbol{w}, \boldsymbol{w}') & (4.35)\\ \leq\ & \tfrac{1}{2}\|\mathrm{rad}(\boldsymbol{w}) - \mathrm{rad}(\boldsymbol{w}')\| + \tfrac{1}{4} d(\boldsymbol{w}, \boldsymbol{w}'), & (4.36) \end{aligned}$$

where (4.34) follows from $|\,|\boldsymbol{a}| - |\boldsymbol{a}'|\,| \leq d(\boldsymbol{a}, \boldsymbol{a}')$ (see Proposition 1.7.3 in [90]) ; (4.35) follows from (4.25), $d\big(\boldsymbol{A}(\boldsymbol{w}) - E, \boldsymbol{A}(\boldsymbol{w}') - E\big) = d\big(\boldsymbol{A}(\boldsymbol{w}), \boldsymbol{A}(\boldsymbol{w}')\big)$ and the Lipschitz constant of

$\boldsymbol{A}(\boldsymbol{w})$; (4.36) follows from the fact that both $\|\mathrm{wid}(\boldsymbol{w})\|$ and $\|\mathrm{wid}(\boldsymbol{w}')\| = 2\|\mathrm{rad}(\boldsymbol{w}')\|$ are less than $h_* \leq \frac{1}{2\kappa^2\lambda\delta}$. Summing the upper bounds of the two summands yields $d(\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{u}}') \leq \frac{3}{4}d(\boldsymbol{w}, \boldsymbol{w}')$, while $d(\boldsymbol{w}, \boldsymbol{w}') = d(\boldsymbol{u}, \boldsymbol{u}')$. We have proved that $\mathbf{K}_{\delta,\chi}$ is contracting, and hence that $\boldsymbol{u}_k$ converges toward its unique fixed point.

Finally, we prove that Algorithm 6 stops with success. Since $\mathbf{K}_{\delta,\chi}$ is $\frac{3}{4}$-contracting, $d(\boldsymbol{u}_{k+2}, \boldsymbol{u}_{k+1}) \leq \frac{3}{4}d(\boldsymbol{u}_{k+1}, \boldsymbol{u}_k)$ so the test $\mu > \overline{\mu}$ at Line 6.11 of Algorithm 6 never fails and the main loop of Algorithm 6 only stops when *success* is true. We now prove that *success* is actually true for $k = \overline{k}$ satisfying the constraint given in the statement. Define $\boldsymbol{u}'_{k+1} = \mathbf{K}(\boldsymbol{u}_k)$ so that

$$\boldsymbol{u}_{k+1} = \mathrm{mid}(\boldsymbol{u}'_{k+1}) + \delta(\boldsymbol{u}'_{k+1} - \mathrm{mid}(\boldsymbol{u}'_{k+1})) + \chi[-1, 1], \tag{4.37}$$

as defined at Line 6.4 and Line 6.9 of Algorithm 6. We shall prove that $\boldsymbol{u}'_{k+1} \subseteq \mathrm{int}(\boldsymbol{u}_k)$, which is equivalent to $\mathrm{rad}(\boldsymbol{u}'_{k+1}) + |\mathrm{mid}(\boldsymbol{u}'_{k+1}) - \mathrm{mid}(\boldsymbol{u}_k)| < \mathrm{rad}(\boldsymbol{u}_k)$. Now, $d(\boldsymbol{u}_k, \boldsymbol{u}_{k+1}) \leq (\frac{3}{4})^k d(\boldsymbol{u}_0, \boldsymbol{u}_1) \leq (\frac{3}{4})^k h_* =: \epsilon$, hence, using (4.37), $\|\mathrm{mid}(\boldsymbol{u}_k) - \mathrm{mid}(\boldsymbol{u}'_{k+1})\| \leq \epsilon$ and $\|\mathrm{rad}(\boldsymbol{u}_k) - (\delta\mathrm{rad}(\boldsymbol{u}'_{k+1}) + \frac{\chi}{2})\| \leq \epsilon$, which implies $\mathrm{rad}(\boldsymbol{u}'_{k+1}) \leq \mathrm{rad}(\boldsymbol{u}_k) + \epsilon - \frac{\chi}{2}$. We obtain finally, $\mathrm{rad}(\boldsymbol{u}'_{k+1}) + |\mathrm{mid}(\boldsymbol{u}'_{k+1}) - \mathrm{mid}(\boldsymbol{u}_k)| \leq \mathrm{rad}(\boldsymbol{u}_k) + \epsilon - \frac{\chi}{2} + \epsilon$, which is strictly less than $\mathrm{rad}(\boldsymbol{u}_k)$ provided that $2\epsilon < \frac{\chi}{2}$, holding for $k = \overline{k}$ by assumption.

□

Note that when the map F is close to singular, $\|C\|$ is very large. Therefore both χ and h_* have to be small so as to satisfy $\chi \leq \frac{1}{6\kappa^2\lambda\delta}$ and $h_* \leq \frac{1}{2\kappa^2\lambda\delta}$. On the other hand, if the system is nonsingular inside a neighborhood of the manifold connected to $\tilde{x}_0$ and included inside $\boldsymbol{x}^{\mathrm{init}}$ then it is compact and $\|C\|$ reaches its lower bound, which therefore has to be strictly positive. As a consequence, there exists a minimal step size which will allow continuing the full curve. Note finally that the bound

$$\overline{k} = \left\lceil \frac{\log \chi - \log 4h_*}{\log 4 - \log 3} \right\rceil \tag{4.38}$$

provided by Theorem 4.3.4 is quite pessimistic. On the one hand, it shows that the static absolute inflation $\chi > 0$ is enough to enforce the certification. On the other hand, the static relative inflation $\delta \geq 1$, although not necessary, strongly speeds up the certification.

4.3.3 Limitations of ParCont

Algorithm 7 is restricted to certify nonsingular curves : Indeed, the certification of existence comes with a certification of regularity of the system's derivatives. If the curve contains a singular point then Algorithm 6 will fail each time a parallelotope contains this singular point. Therefore, the step size will keep on decreasing leading to a sequence of smaller and smaller parallelotopes, which will converge to this singularity, eventually terminating when the prescribed minimal step size is reached.

When the curve is regular, Theorem 4.3.4 shows that there exists a step size h_* that enforces the success of all certifications. However, this analysis does not take into account the finite precision floating point computations. In practice, the required step size could be too small with respect to the computational precision, leading to a failure similar to the singular case. When using standard floating point computations, the computational precision depends on the magnitude of the involved numbers, hence translating a problem can impact the resolution using floating point numbers. This is however inherent to all floating point implementations of numerical algorithms.

Non certified continuation methods can benefit of quasi-Newton like algorithms, which allow handling very large systems. Each step of Algorithm 7 involves operations of cubic complexity with respect to the number of variables, which are involved in the interval Newton steps. This cubic complexity restricts the scope of Algorithm 7 to smaller problems than non certified methods.

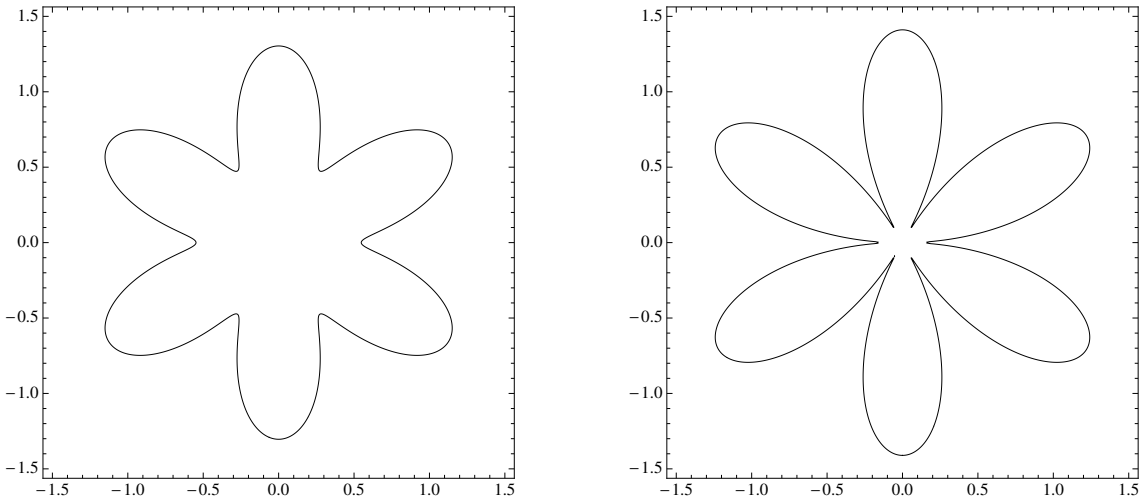
The heuristic we proposed for updating the step length is simple. In terms of complexity, the best would be to have at each iteration of the ParCont the step length that leads to a success of certification at a reasonable cost (few interval Newton iterations). There is a balance between the step length (so as to move quickly along the manifold) and the cost of the iteration (which is likely to be high if the step length is close to the maximal step length for succeeding the certification). We have tried a strategy to build over the iterations of ParCont an approximated model associating a step length to the number of iterations of the procedure N–Inflate, and also a limit step length for which N–Inflate fails. However, we have failed to find a model, and update procedures, that is close enough to the behavior of N–Inflate. This strategy turned out to be much less efficient than the simple strategy presented above.

Eventually, although this is not the scope of this chapter, the adaptation of ParCont to general m-manifolds, $m > 1$, is interesting but not straightforward. There is no theoretical limitation in producing paralleletopes in order to certify solutions of systems inducing m-manifolds. However if $m > 1$, there are m input and output facets of a certified parallelotope, each containing solutions forming a $m-1$ manifold. Orienting and connecting new parallelotopes with respect to those facets in order to be certain that at the end of the algorithm no solutions are loss (i.e. there are no gaps between certified parallellotopes) is difficult.

4.4 Experiments

ParCont has been implemented in C++ using the RealPaver [44] API, implementing routines such as interval newton methods and many other constraint solving techniques, which uses Gaol [41] for interval arithmetic, and Lapack [2] for linear algebra (matrix inversions, kernel computations, etc). All the experiments have been run on a machine under Linux Ubuntu version 11.10, with processor Intel i5-2400 3.10GHz and 4Gb of RAM. In the following experiments, the parameters of N–Inflate and Contract are set as described in Section 4.2 (i.e. $\chi = 10^{-12}$, $\delta = 1.1$, $\overline{k} = 15$, $\overline{\mu} = 1$) – except when explicitly mentioned – and as described in Section 4.3.1 for parameters of the main algorithm (i.e. $\alpha = 0.5$, $\beta = 1.1$, $\underline{h} = 10^{-8}$). The derivatives in N–Inflate are by default computed using the box hull of the input parallelotope. In addition, some results are shown when considering two other ways of computing the derivatives, namely : mean value form of the derivatives (hence using the second derivatives of F evaluated at $\square\widehat{\boldsymbol{x}}$) and formal form (a formal condensed expression of $G(w)$ is computed beforehand).

Measuring the performances of ParCont by counting the number of produced parallelelotopes is not relevant. Indeed, an iteration of the algorithm makes use of the iterative procedures N–Inflate and Contract whose induced computational cost is not fixed. Moreover, unsuccessful steps have to be counted since they cost as much as successful ones. Hence, it is more suitable to count the number of iterations performed by N–Inflate and Contract. The time complexity of both N–Inflate and Contract is $O(n^3)$ per iteration, which corresponds to the interval matrix multiplication performed when evaluating the derivatives in the auxiliary basis. The complexity of Contract can be reduced to $O(n^2)$ by reusing the last interval evaluation of the derivatives in N–Inflate. Thus, the number of



(a) $\epsilon = 0.7$ (b) $\epsilon = 0.99$

Figure 4.3 – Solution manifold of Equation (4.39).

N–Inflate iterations is used as a performance measure. In the following, both computational times and number of N–Inflate iterations are presented.

Our goal is to illustrate the strengths (and limitations) of parallelotopes with respect to boxes. To this end, we propose a comparison of ParCont with a continuation based on boxes as in [58]. The algorithm in [58] can be seen as an intervalization of the parameter-embedding continuation : boxes are constructed along the manifold by first selecting one of its components as a parameter. The certification process is based on a parametric interval Krawczyk operator and a generate-and-try inflation. However, from the information provided in [58] we have not been able to tune the inflation process correctly in order to reach any satisfying results. Therefore, we have implemented a version of this algorithm, called BoxCont, replacing the inflation process by the one defined in Section 4.2, and using a heuristic for guessing good initial boxes as in Equation (4.8).

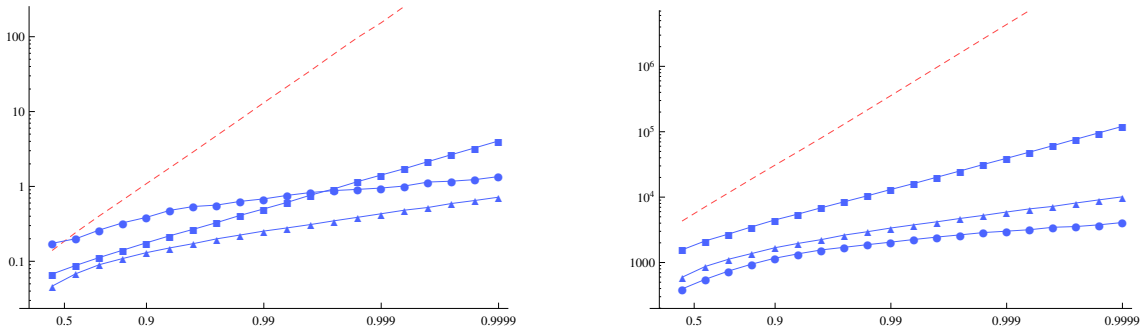
Different experiments are presented in the following. First, the two methods are applied to a problem with increasingly difficult topology. Second, another problem with increasingly difficult conditioning is proposed. Third, the algorithms are applied to track a particular manifold embedded in higher and higher dimensional space. Last, two applications of certified continuation are presented : homotopy continuation for tracking the roots of a complex polynomial - results are compared with another certified method from the literature - and robot control synthesis.

4.4.1 Influence of the manifold topology

Consider the following system depending on the parameter $\epsilon \in [0, 1]$:

$$x_1^8-(1-\epsilon)x_1^6+4x_1^6x_2^2-(3+15\epsilon)x_1^4x_2^2+6x_1^4x_2^4-(3-15\epsilon)x_1^2x_2^4+4x_1^2x_2^6-(1+\epsilon)x_2^6+x_2^8. \quad (4.39)$$

The solution manifold of this system has a six-petals-flower shape. The parameter ϵ controls the length of the petals, hence the acuity of the curvature between two consecutive petals (see Figure 4.3). In particular $\epsilon = 0$ corresponds to a circle and $\epsilon = 1$ leads to six petals connected by a singularity at (0,0). We solve several instances of (4.39) using varying ϵ values closer and



(a) Time versus ϵ (b) N–Inflate iterations versus ϵ

Figure 4.4 – Comparison of ParCont (plain lines) with BoxCont (dashed line) on the manifold induced by Equation (4.39) (logarithmic scales). Square, triangle and circle markers represent different evaluation of the derivatives, respectively : hull of parallelotopes (default mode), mean value form and formal form.

closer to 1, and starting from the initial solution $(\sqrt{1-\epsilon}, 0)^T$. The non-linearity of this problem is strong enough to allow using $\chi = 0$ in N–Inflate and $\underline{h} \simeq 0$, hence enabling tackling very strong curvatures.

Figure 4.4 confirms that as ϵ increases, more computations are required. This is seen as the number of N–Inflate iterations and hence computational times are increasing with ϵ. The continuation is indeed adapting to the acuity of the curvature by reducing the average step length, hence increasing the number of iterations. BoxCont appears to adapt badly to the difficult topology, whereas ParCont handles them well. Indeed, the growth factor in number of iterations as ϵ increases is much higher when considering simple box continuation. On the other hand, manipulating parallelotopes is computationally more costly, hence the computation times for the problems with low ϵ are quite comparable. Nevertheless, Figure 4.4 clearly shows that parallelotopes asymptotically have better timings : $t_{\text{par}} \approx 0.062(1-\epsilon)^{-0.45}$ while $t_{\text{box}} \approx 0.099(1-\epsilon)^{-1.06}$, which are quite accurate approximations in view of the almost linear curves in the log scale graphic.

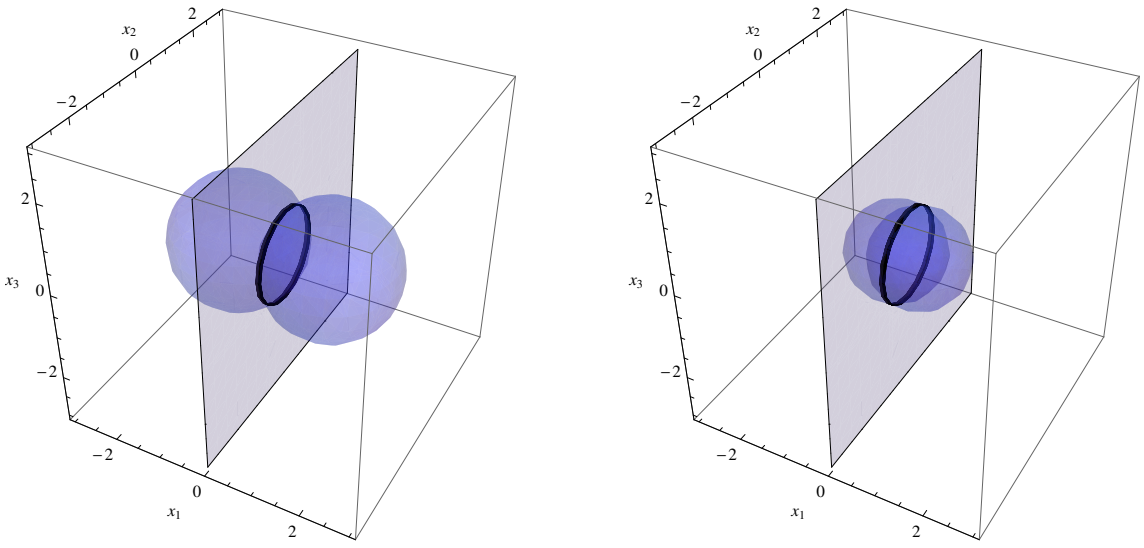
Exploiting better parallelotopes using more accurate derivative computations improves the asymptotic behavior of ParCont as shown on Figure 4.4. Computing derivatives using a formal auxiliary expression of (4.39) strengthen the adaptation to strong curvatures. On the other hand, this expression is complex, hence its evaluation and differentiation are more computationally expensive. This way of computing derivatives is worth using only when the curvatures are very sharp. Computing the derivatives using a mean form requires second derivatives of (4.39). Nevertheless, the small number of variables makes this technique computationally reasonable, and overall very efficient.

4.4.2 Influence of the conditioning

Consider the following problem :

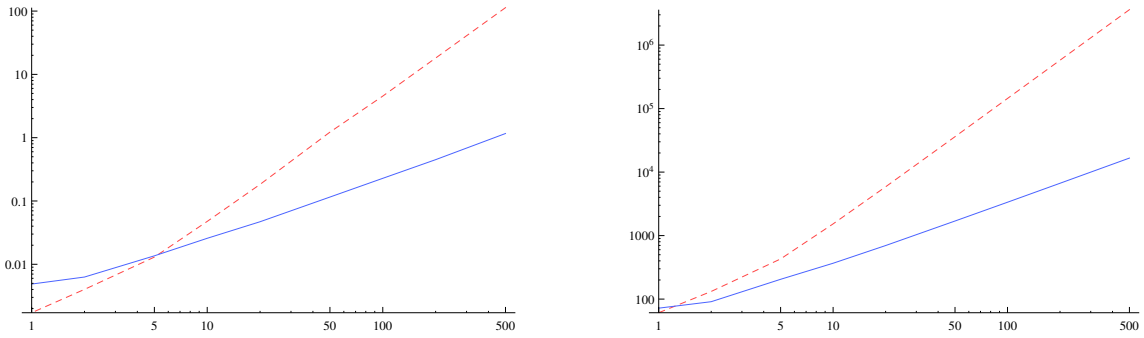
$$\begin{pmatrix} (x_1+\epsilon)^2 + x_2^2 + x_3^2 - (1+\epsilon^2) \\ (x_1-\epsilon)^2 + x_2^2 + x_3^2 - (1+\epsilon^2) \end{pmatrix} = 0. \tag{4.40}$$

The problem in (4.40) consists of two spheres, at distance 2ϵ from each other, whose intersection is the unit circle in the plane (x_2, x_3). By shifting ϵ towards 0, the system becomes more and



(a) $\epsilon = 1$ (b) $\epsilon = 0.25$

Figure 4.5 – Solution manifold of Problem (4.40).



(a) Time versus $1/\epsilon$ (b) N–Inflate iterations versus $1/\epsilon$

Figure 4.6 – Comparison of ParCont (plain line) with BoxCont (dashed line) on the manifold induced by Equation (4.40) (logarithmic scales).

more singular, see Figure 4.5. The norm of matrices C are increasing, following a factor $\simeq 1/\epsilon$, whereas the manifold of solutions remains the same. We solve this problem for different ϵ values, with $(0, 1, 0)$ as starting points, and compare the performances of BoxCont and ParCont.

Figure 4.6 shows, as awaited, that performances deteriorate as the problem becomes more singular. However, BoxCont suffers more from this difficulty than ParCont. Whereas the usage of BoxCont seems reasonable when the problem is well defined, showing better timings than ParCont, its performances significantly reduce with the conditioning of the system. The growths in computational time and in N–Inflate iterations in ParCont are indeed much lower than in BoxCont. Results for sharper derivatives evaluations are not reported here. There is in fact no gain in using these techniques here since the manifold to track is simple.

4.4.3 Influence of the embedding space dimension

Consider the following system depending on the parameters $n \in [2, +\infty)$ and $\epsilon \in (0, 1]$:

$$\begin{pmatrix} x^T Q' x - 1 \\ A' x \end{pmatrix} = 0. \tag{4.41}$$

with $Q' = M^T Q M$, $A' = AM$, $Q = \begin{bmatrix} I_{n-1\times n-1} & 0 \\ 0 & \epsilon \end{bmatrix}$, $A = (I_{n-2\times n-2}|0|0)$ and M an orthonormal matrix. With $M = I$, the solution manifold of (4.41) consists in a 2D ellipse embedded in the subspace spanned by (e_{n-1}, e_n) of $\mathbb{R}^n$ and whose shape is controlled by ϵ ($\epsilon = 1$ yields a circle, while $\epsilon \to 0$ yields an ellipse more and more stretched towards infinity). Applying an isometry with $M \neq I$ enforces the different variables to be non-trivially involved in the equations, and preserves the length and shape of the manifold. The following results are given upon a set of 10 random transformation matrices M per tested dimension, obtained by applying a Gramm-Schmidt orthogonalization to random matrices. Initial solutions are $M^T(0, 0, \ldots, 0, 1, 0)$. For each dimension, Figure 4.7 shows the average timings and number of N–Inflate iterations.

Consider first the results for $\epsilon = 1$ (i.e. a circle) reported in Figure 4.7(a) and 4.7(b). They show that the two methods have similar computational time as the dimension increases. However, ParCont performs less steps than BoxCont. The latter also shows a growth of the number of N–Inflate iterations as the dimension increases that impacts its computational time (ParCont runs faster for dimensions above 32). The number of N–Inflate iterations of ParCont remains overall stable.

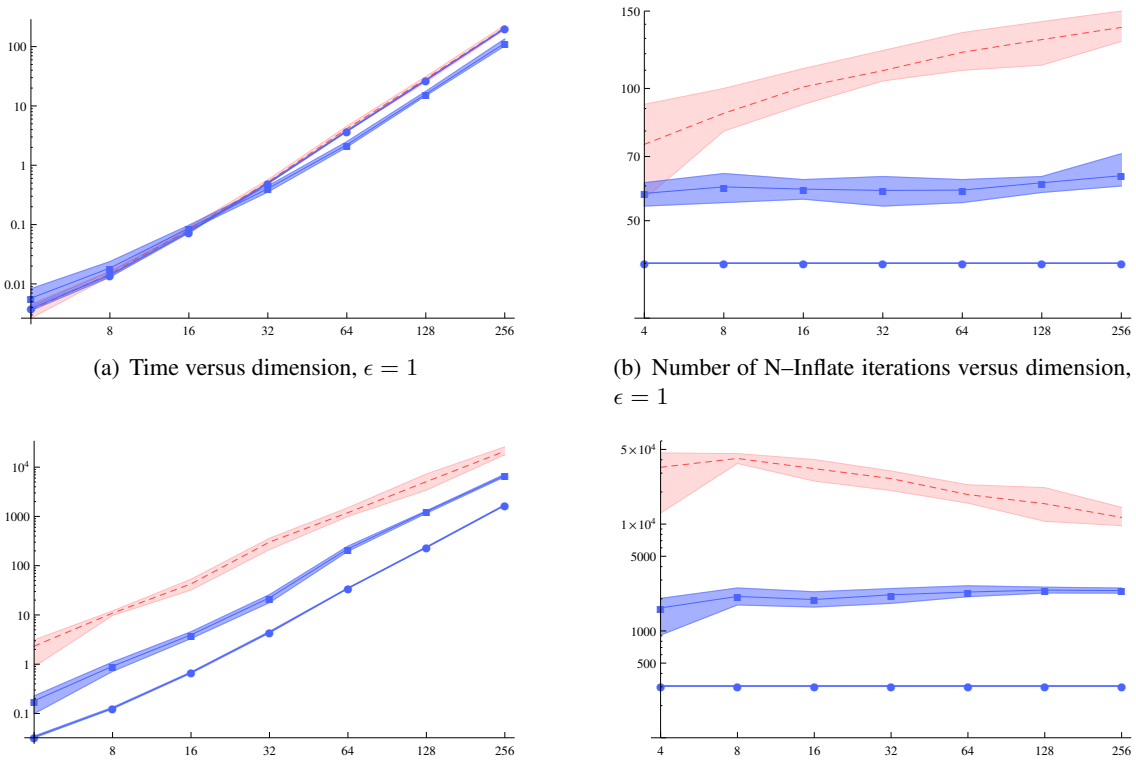
Consider now the results for $\epsilon = 10^{-3}$ (i.e. a stretched ellipse) on Figure 4.7(c) and 4.7(d). It appears ParCont requires less computation time than BoxCont and its number of N–Inflate iterations remains stable. However, BoxCont shows a growth of computation time slightly lower than ParCont. This is explained by a surprising decrease of the number of steps as the dimension increases. This effect is due to the parameter embedding approach used in BoxCont : moving along one parameter with step size h entails that the certified manifold inside a box produced by N–Inflate is proportional to approximately $h\sqrt{n}$ (the diagonal of a n-dimensional box of width h). Hence, since the total length of the manifold remains constant by construction and the average step length remains quite stable as the dimension increases, the corresponding reduction of the number of N–Inflate iterations, due to a reduction of the number of steps, can be observed.

Computing derivatives using a mean form, hence computing second-order derivatives, is not reasonable here due to the large dimension of the problems (it is in fact much slower than the default ParCont after dimension 8). On the other hand, on this particular problem, we can construct a very simple formal expression of the derivatives in the auxiliary space of parallelotopes, using

$$G'(w) = \begin{pmatrix} 2w^T(C^T Q' C) + (\tilde{x}^T Q')C \\ A'C \end{pmatrix}. \tag{4.42}$$

This latter expression involves more matrix-matrix multiplications than $F'(\square\widehat{\boldsymbol{x}})C$, hence increasing the cost of the derivative evaluation. However the results become totally stable : For any rotation and any dimension, the number of N–Inflate iterations (and number of steps) remains the same. Furthermore, the overall computational time is sensibly decreased with respect to the default derivative evaluation.

To conclude, using parallelotopes instead of boxes guarantees stability in following a manifold pushed into several dimensions, or, otherwise said, the behavior of ParCont depends essentially



(a) Time versus dimension, $\epsilon = 1$

(b) Number of N–Inflate iterations versus dimension, $\epsilon = 1$

(c) Time versus dimension, $\epsilon = 10^{-3}$

(d) Number of N–Inflate iterations versus dimension, $\epsilon = 10^{-3}$

Figure 4.7 – Results of ParCont (plain lines) and BoxCont (dashed-lines) on the manifold induced by Equation (4.41) (axis y in logarithmic scale). Colored part represents the range (maximum and minimum) of the different measures among the 10 random transformation matrices M. Square and circle markers represent respectively the differentiation using the hull of parallelotopes (default mode) and a formal expression of $G'(u)$.

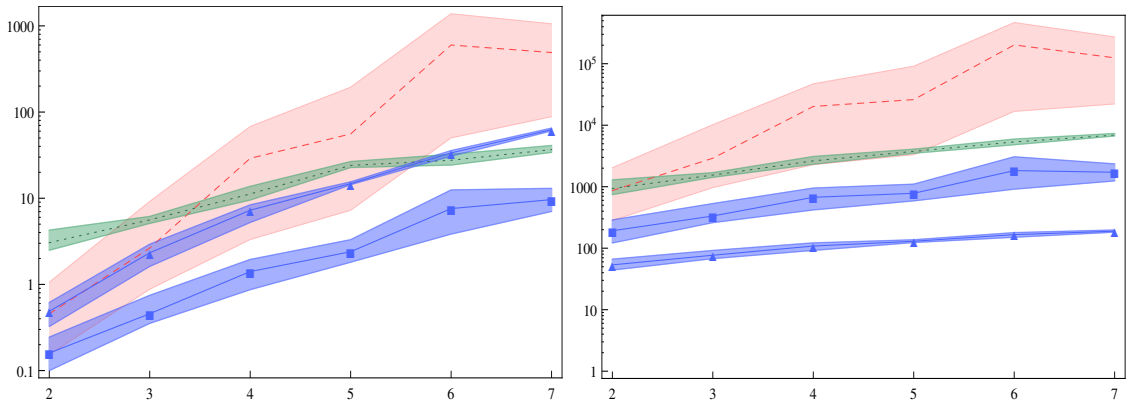
on the manifold shape and not on its embedding dimension. Coupled with the results from the previous experiments, the use of parallelotopes ensures the stability of the continuation process, adapts well to difficult topologies and is less sensitive to the conditioning of the system.

4.4.4 Homotopy continuation

ParCont is used here to solve a polynomial system through homotopy. Its results are compared against the method NAG4M2 [5], a certified homotopy continuation implemented in the software Macaulay2 [45].

The problem $\mathrm{Katsura}_n$ is one of the standard scalable benchmarks for homotopy solving methods. It consists of a system of $n+1$ variables $(z_0, z_1, \ldots, z_n) \in \mathbb{C}^{n+1}$. For instance $\mathrm{Katsura}_2(z)$ consists of :

$$\mathrm{Katsura}_2(z) = \begin{pmatrix} 1 - z_0 - 2z_1 - 2z_2 \\ z_1 - 2z_0z_1 - 2z_1z_2 \\ z_0 - z_0^2 - 2z_1^2 - 2z_2^2 \end{pmatrix} = 0 \tag{4.43}$$



(a) Time versus dimension (b) Number of steps versus dimension

Figure 4.8 – Comparison of the different methods on the manifold induced by Equation (4.44) (axis y in logarithmic scale). Plain, dashed and dotted lines corresponds respectively to ParCont, BoxCont and NAG4M2. Colored part represents the range (maximum and minimum) of the different measures among the μ values. Square and triangle markers represent respectively differentiation using the hull of parallelotopes (default mode) and mean value form.

The aim is to find all the complex roots of $\mathrm{Katsura}_n$. Therefore, the system to solve is the following linear homotopy :

$$(1-t)h_n(z) + t\mu\mathrm{Katsura}_n(z) = 0 \tag{4.44}$$

where $\mu \in \mathbb{C}$ is randomly selected on the complex circle, $t \in [0,1]$ the induced homotopy variable and h_n an initial system matching the degree of $\mathrm{Katsura}_n$, that is : $h_n(z) = \left(z_0 - 1, z_1^2 - 1, \ldots, z_n^2 - 1\right)^T$. In order to solve the system with ParCont, the real and imaginary parts have to be separated, doubling the number of variables and equations. Hence, with z in $\mathbb{C}^{n+1}$, there are $2(n+1)+1$ variables (real and imaginary part of z plus t) and $2(n+1)$ equations. Input solutions are roots of h_n (there are 2^n such roots), and $t = 0$. The aim is to follow the paths of solutions of (4.44) from $t = 0$ in order to reach solutions to $\mathrm{Katsura}_n$ at $t = 1$. Hence, the domain of the variables is not limited, except for t which must remain in $[0,1]$.

The results obtained by ParCont are compared against those of the certified homotopy tracking method NAG4M2 from [5]. There is no performance measure in NAG4M2 that can be compared with the number of N–Inflate iterations. Therefore, we instead compare the computational time and the number of steps of the main algorithm (counting also unsuccessful ones for ParCont). Indeed, a step of NAG4M2 builds a new solution, whereas a step of ParCont is an attempt to build a new parallelotope. We randomly selected a sample of five values of μ such that no one leads to an incomplete homotopy or diverging paths. For each value of μ and each root of the polynomial system, we measured the number of steps (including unsuccessful steps for ParCont) and computation time. Figure 4.8 reports mean, min and max values of these measures for different problem sizes.

Figure 4.8 shows that the performances of BoxCont are worse, in average, than ParCont and NAG4M2 with n greater than 4. Noticeably, BoxCont shows high instability for the different values of μ both in iterations and computation time. Undeniably, the difficulty to adapt well to the topology of solution paths makes the use of boxes unadvised compared to parallelotopes on

this benchmark. Considering ParCont and NAG4M2, Figure 4.8 shows that the former requires less steps and computational time than the latter. However, whereas the growth rate of steps are approximately proportional between the two methods, computational time of ParCont appears to increase quicker than NAG4M2 when the dimension increases. In addition, NAG4M2 appears to be more stable than ParCont between the different values of μ.

Formal expressions in auxiliary space of (4.44) are too large to be handled by our implementation, making it impossible to exploit better derivatives with this technique. Mean value form can still be used. The improved derivative computation depicted on Figure 4.8 appears to be more stable than default ParCont. However, computing second-order derivatives appears too computationally expensive when the dimension increases : computation times are always higher than default ParCont, and higher than NAG4M2 after $n = 6$.

All in all, one can see that ParCont is competitive with NAG4M2. The latter makes use of techniques well suited for homotopy on polynomial systems (complex arithmetic computations and use of homogeneous systems and projective spaces) ensuring a more stable behaviour with respect to the value of μ. Hence, it seems promising to incorporates these techniques in ParCont, for instance by using complex interval arithmetic as in [102]. Finally, better derivative computations increase the stability of ParCont, but would require some dedicated and computationally cheaper evaluation techniques to handle high dimensional problems.

4.4.5 Control synthesis

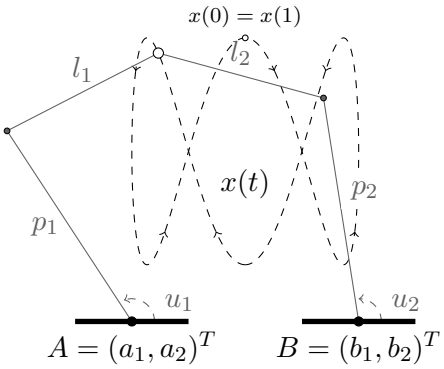
A robot is defined by a system $F(x, u) = 0$ called the kinematic model, where $F : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$, $x \in \mathbb{R}^n$ is the pose of the end effectors of the robot, and $u \in \mathbb{R}^n$ are the control variables. Given a trajectory for the effector defined by $x : [0, 1] \to \mathbb{R}^n$ (from $x(0)$ to $x(1)$), the aim of control synthesis is to find the control function $u(t)$ required to follow this trajectory, that is all the solutions to $F(x(t), u) = 0$, starting from one given input control $u(0) = u^0$ satisfying $F(x(0), u^0) = 0$. Obtaining certified continuous control function is critical in order to avoid false trajectory tracking, or inconsistent sequences of controls. Therefore, ParCont is suitable to solve this problem

Consider the robot presented in Figure 4.9(a), called RRRRR [15]. The end effector is controlled by two articulated arms, each made of two parts linked by a revolute joint. The two arms are controlled by their angles u_1 and u_2. One can easily see that the pose of the end effector may be attained by two different controls per articulated arm. The kinematic model of this robot is :

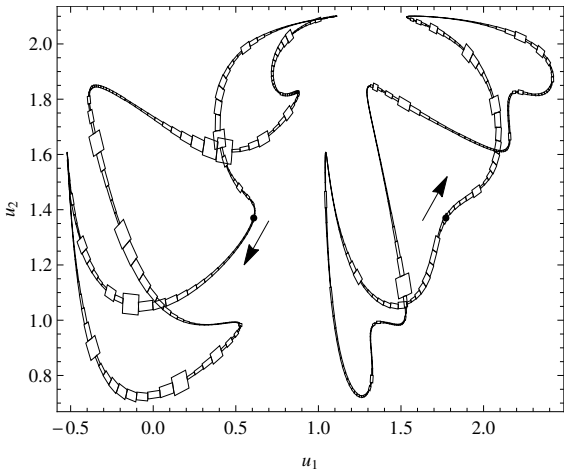
$$F(x, u) = \begin{pmatrix} (x_1 - (\cos(u_1)p_1 + a_1))^2 + (x_2 - (\sin(u_1)p_1 + a_2))^2 - l_1^2 \\ (x_1 - (\cos(u_2)p_2 + b_1))^2 + (x_2 - (\sin(u_2)p_2 + b_2))^2 - l_2^2 \end{pmatrix} = 0, \tag{4.45}$$

where $x = (x_1, x_2)^T$ is the position of the end effector of the robot, $A = (a_1, a_2)^T$ (respectively $B = (b_1, b_2)^T$) is the position of the fixed joint of the first (respectively second) articulated arm. The first and second part of the first arm are of respectively length p_1 and l_1 (p_2 and l_2 for the second arm). For this benchmark, $x(t) = (\sin(2\pi t), \cos(3(2\pi t)) + 1.5)$, with t in $[0, 1]$ (shown in Figure 4.9(a)).

The geometric parameters of the robot are $A = (-1, 0)^T$, $B = (1, 0)^T$, p_1 and p_2 set to 2, l_1 and l_2 set to 1.5. Given the system $F(x, u)$ and the initial pose $x(0) = (0, 2.5)^T$, ParCont is applied with two initial controls $u^0 = (0.6085, 1.3699)^T$ and $u^0 = (1.7716, 1.3699)^T$. The sequence of controls following the trajectory obtained with ParCont are represented on Figure 4.9(b), and are



(a) Scheme of the robot. The trajectory $x(t)$ is represented in dashed.



(b) Computed control sequence projected in the control space. Initial controls u^0 are represented by points.

Figure 4.9 – Example of a two armed robot RRRRR.

fully certified to be continuous. Each control sequence has been computed in approximately 0.14 seconds.

4.4.6 Conclusion

ParCont presents a better behavior compared to BoxCont : it adapts better to the topology of the tracked manifold, its less sensitive on the conditioning and on the dimension of the solved system. We also see that ParCont is competitive in rigorously finding complex roots of polynomial systems by homotopy, although improvements are required so as to handle complex numbers and projective spaces that avoids divergent paths as in [5]. Eventually, a simple direct application of ParCont, for which certification is compulsory, is shown. Now, we would like to apply ParCont for solving nonlinear biobjective problems via their first-order system of equations. Some adaptations are necessary in order to properly handle the inequality constraints of the biobjective problem.

4.5 Adaptation to biobjective optimization

In Section 3.3.2 (p. 67), we have presented several continuation methods for nonlinear multiobjective problems based on a characterization of Pareto-optimal solutions via the first-order optimality conditions. We propose here to adapt ParCont in order to solve such problems. Noticeably, the adaptation handles singularties resulting from loss of constraint complementarity, see Theorem 3.2.5 (p. 59).

4.5.1 Detecting rigorously changes of constraint activity

Consider a nonlinear biobjective problem as (3.1) (p. 53) and its system F of first order-conditions as (3.4) (p. 58), involving variables $x \in \mathbb{R}^n$ and multipliers $\lambda \in \mathbb{R}^2_+$, $r \in \mathbb{R}^p_+$ and $s \in \mathbb{R}^q$. We denote by $z = (x, \lambda, r, s)$ the vector of these variables and multipliers, and consider

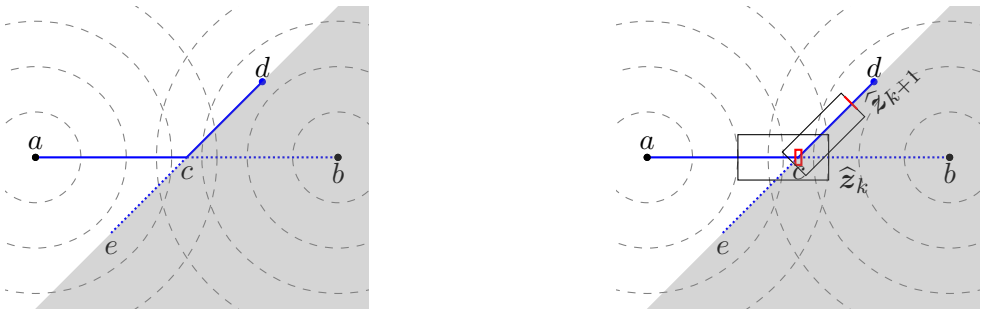


Figure 4.10 – Bi-objective problem of Example 4.5.1.

that ParCont iterates on parallelotopes $\widehat{z}_k$. We consider the Fritz John conditions instead of KKT conditions and additionally consider the normalization equation $\lambda^T\lambda + r^T r + s^T s = 1$ (instead of $\sum_i \lambda_i = 1$ for KKT), which are valid regardless of any constraint qualification.

As seen previously, ParCont requires the regularity of the manifold in order to be able to produce parallelotopes. Three possible sources of singularity in the system of first-order conditions have been stated in Theorem 3.2.5 (p. 59). In particular, the loss of constraint complementarity are very common, provided the biobjective problem contains inequality constraints. Thus, in order to treat inequality constraints, ParCont is adapted to handle such singularities.

They occur at solutions z where there is a constraint i with $g_i(x) = 0$ and $r_i = 0$. The equation $g_i(x)r_i = 0$ actually consists of the product of two zero quantities. Hence, two paths are possible, one where $g_i(x) \neq 0$ and $r_i = 0$ (only the part $g_i(x) \leq 0$ is feasible), the other where $g_i(x) = 0$ and $r_i \neq 0$ (only the part $r_i \geq 0$ is feasible).

Example 4.5.1 – Consider the biobjective problem with $f_1(x) = (x_1 + 1)^2 + x_2^2$, $f_2(x) = (x_1 - 1)^2 + x_2^2$ and inequality constraint $g(x) = x_1 - x_2 \leq 0$, first presented in Example 3.2.4 (p. 61). Variables of F are here $z = (x_1, x_2, \lambda_1, \lambda_2, r)$. This problem is illustrated in the left hand side graphic of Figure 4.10 in the plane (x_1, x_2). The solution $c = (0, 0, \sqrt{2}/2, \sqrt{2}/2, 0)$ is singular for F. The curve of solutions bifurcates at this point in two paths : the path from $a = (-1, 0, 1, 0, 0)$ to $b = (1, 0, 0, 1, 0)$ satisfies $r = 0$ but is infeasible from c to b ($g(x) > 0$); the path from $d = (0.5, 0.5, 0, \sqrt{2}/2, \sqrt{2}/2)$ to $e = (-0.5, -0.5, \sqrt{2}/2, 0, -\sqrt{2}/2)$ satisfies $g(x) = 0$ but is infeasible from c to e ($r < 0$).

Remark 4.5.1 – The x-projection depicted in Figure 4.10 can be misleading because it hides the multipliers space. In particular, the case where one constraint is activated and another constraint is disactivated at the same x is stable : in fact, two successive different single activation changes occur, separated by a Pareto curve along the multiplier space with fixed x (see Section 4.6).

In order to handle these singularities, we propose to use the approach from [100], originally in the context of parametric optimization. This technique considers instead of the system (3.4) of first-order conditions, a subsystem restricted to the active constraints. Given a set $\mathcal{A}$ of active constraints, this subsystem can be stated as :

$$F_{\mathcal{A}}(x, \lambda, r, s) = \begin{pmatrix} \nabla f(x)\lambda + \nabla g_{\mathcal{A}}(x) r_{\mathcal{A}} + \nabla h(x) s \\ (\forall i \in \mathcal{A})\ g_i(x) \\ (\forall i = 1, \ldots, q)\ h_i(x) \end{pmatrix} = 0, \tag{4.46}$$

with $x \in \mathcal{X}$ (i.e. satisfying all constraints of the biobjective problem), $r \geq 0$ and $\lambda \geq 0$, and the normalization equation $\lambda^T\lambda + r^T r + s^T s = 1$. Note that the multipliers r_i for $i \notin \mathcal{A}$ are fixed to 0. Given an initial solution $z = (x, \lambda, r, s)$ of the first-order system (3.4), and let the set $\mathcal{A}$ be $\mathcal{A}(x)$, the continuation can equivalently be applied on the system (4.46), provided the set $\mathcal{A}$ is updated correctly during the continuation. In addition, as the active constraints are considered as equality constraints in the system (4.46), there is no singularity due to loss of constraint complementarity in this restricted system, which can be now treated by ParCont provided it manages the set of active constraints (detecting activation and disactivation of inequality constraints) rigorously. As in [100], detecting activation/disactivation requires observing whether a constraint g_i, $i \notin \mathcal{A}$, verifies $g_i(x) = 0$ (activation of g_i) or if the multiplier r_i, $i \in \mathcal{A}$ associated to an active constraint equals 0 (the active constraint g_i disactivates).

Let $\mathcal{A}_k$ be the set of active constraints at iteration k of ParCont, and $\widehat{\boldsymbol{z}}_k = (C_k, (\boldsymbol{u}_k, \boldsymbol{v}_k), \tilde{z}_k)$ a parallelotope certified to contain solutions of (4.46). In order to maintain a correct active set for the next iteration of ParCont, it is required to determine changes in $\mathcal{A}_k$ that occur on the solution curve contained in $\widehat{\boldsymbol{z}}_k$. To this end, we introduce the following Numerical Constraint Satisfaction Problems (NCSP) for each constraint g_i :

$$\left[\begin{array}{l} z = C_k(u, v) + \tilde{z}_k,\ F_{\mathcal{A}_k}(z) = 0,\ \omega_i(z) = 0 \\ u \in \boldsymbol{u}_k, v \in \boldsymbol{v}_k \end{array}\right], \tag{4.47}$$

whose variables u and v take their values in $\boldsymbol{u}_k$ and $\boldsymbol{v}_k$ respectively, i.e. the characteristic box of $\widehat{\boldsymbol{z}}_k$. The virtual variables z, and the first constraint, express the transition from the parallelotope basis. The second constraint enforces (4.46). The third constraint deals with the activation/disactivation of inequality g_i : the function $\omega_i(z)$ is defined as $g_i(x)$ if $i \notin \mathcal{A}_k$ (i.e. we try to activate g_i), or as r_i if $i \in \mathcal{A}_k$ (i.e. we try to disactivate g_i). Since the parallelotope is certified for (4.46) in $\widehat{\boldsymbol{z}}_k$, there is a unique solution to the equations $F_{\mathcal{A}_k}(z) = 0$ for each $v \in \boldsymbol{v}_k$. Hence, solving (4.47) can be done by performing an unidimensional search in the domain $\boldsymbol{v}_k$ of variable v, whose complexity is generally low. In addition, since only the first change of activity is of interest, a single solution is computed, the one closest to $\underline{v}_k$.

Changes in the active set must be certified for ParCont to remain rigorous. To this end, we propose to solve (4.47) with an interval-based Branch and Prune (B&P) method, adapted to the specificities of this problem (i.e. it branches only on the domain $\boldsymbol{v}_k$ of the variable v, and uses only the interval Newton to solve the constraints). For each CSP (4.47), the method returns either no solution, thus proving no change of activity occurs in $\widehat{\boldsymbol{z}}_k$ for this constraint ; or the first encountered certified solution box $(\boldsymbol{u}_k^i, \boldsymbol{v}_k^i)$ identifying a verified change ; or the first encountered non-certified solution box if the computational precision is insufficient. The process is applied at each iteration of ParCont, once a parallelotope $\widehat{\boldsymbol{z}}_k$ has been certified. The overall technique for detecting changes in the active set of constraints is presented in Algorithm 8.

Given a certified (by ParCont) parallelotope $\widehat{\boldsymbol{z}}_k = (C_k, (\boldsymbol{u}_k, \boldsymbol{v}_k), \tilde{z}_k)$, this algorithm maintain a parallelotope $\widehat{\boldsymbol{z}}_c$ containing the change of constraint activity closest to $\underline{v}_k$, and two flags *change* and *unique* indicating respectively whether $\widehat{\boldsymbol{z}}_c$ contains a change of active constraints, and if this change is unique. Initially, $\widehat{\boldsymbol{z}}_c$ is set to the output edge of $\widehat{\boldsymbol{z}}_k$, and we try to find changes of the active set between the input edge of $\widehat{\boldsymbol{z}}_k$ and the output edge of $\widehat{\boldsymbol{z}}_c$ (i.e. the first encountered active set change). On this portion of $\widehat{\boldsymbol{z}}_k$, for all inequality constraint g_i such that the interval evaluation of the constraint ω_i contains 0 (i.e. the constraint g_i potentially activates/disactivates), then the NCSP (4.47) is solved on the domain $(\boldsymbol{u}_k, [\underline{v}_k, \overline{v}_c])$ using the adapted B&P, yielding to a solution box $(\boldsymbol{u}_k^i, \boldsymbol{v}_k^i)$. If this box is empty, the constraint g_i does not change its activity status wi-

Algorithm 8: Check Active Set Change

Input: Certified Parallelotope $\widehat{\boldsymbol{z}}_k = (C_k, (\boldsymbol{u}_k, \boldsymbol{v}_k), \tilde{z}_k)$, $F_{\mathcal{A}_k}$, constraints g_i
Output: A parallelotope $\widehat{\boldsymbol{z}}_c$ and two flags *change* and *unique*

8.1 $(\boldsymbol{u}_c, \boldsymbol{v}_c) \leftarrow (\boldsymbol{u}_k, [\overline{v}_k, \overline{v}_k])$;
8.2 *change* $\leftarrow$ **false** ;
8.3 *unique* $\leftarrow$ **true** ;
8.4 **foreach** i *with* $0 \in \boldsymbol{\omega}_i(C_k(\boldsymbol{u}_k, [\underline{v}_k, \overline{v}_c]) + \tilde{z}_k)$ **do**
8.5 $((\boldsymbol{u}_k^i, \boldsymbol{v}_k^i), \mathit{certificate}) \leftarrow$ B&P(NCSP (4.47), $(\boldsymbol{u}_k, [\underline{v}_k, \overline{v}_c])$) ;
8.6 **if** $(\boldsymbol{u}_k^i, \boldsymbol{v}_k^i) \neq \emptyset$ **then**
8.7 *change* $\leftarrow$ **true** ;
8.8 **if** *certificate* **and** $\boldsymbol{v}_k^i \cap \boldsymbol{v}_c = \emptyset$ **then** *unique* $\leftarrow$ **true** ;
8.9 **else** *unique* $\leftarrow$ **false** ;
8.10 $(\boldsymbol{u}_c, \boldsymbol{v}_c) \leftarrow (\boldsymbol{u}_k^i, \boldsymbol{v}_k^i)$;
8.11 **end**
8.12 **end**
8.13 **return** $((\boldsymbol{u}_c, \boldsymbol{v}_c), \mathit{change}, \mathit{unique})$

thin $(\boldsymbol{u}_k, [\underline{v}_k, \overline{v}_c])$. Otherwise, the constraint g_i (potentially) changes its constraint activity within $(\boldsymbol{u}_k^i, \boldsymbol{v}_k^i)$. If a certificate (existence and uniqueness) has been produced for this solution box and if it does not overlap the previously found change, then this change of constraint activity is proved to exist and be unique.

At the end of the algorithm, if both flags *change* and *unique* are true, there is a constraint g_i that is proved to change its activity within $\widehat{\boldsymbol{z}}_c$, and there is no other changes occurring within $\widehat{\boldsymbol{z}}_c$ and this change is the first encountered along the manifold of solutions within $\widehat{\boldsymbol{z}}_k$. The output edge $\widehat{\boldsymbol{y}}_k$ of $\widehat{\boldsymbol{z}}_k$ is then set to $\widehat{\boldsymbol{z}}_c$. If *change* is false, no constraint change its activity within $\widehat{\boldsymbol{z}}_k$. The iteration of ParCont follows as usual. Otherwise, the parallelotope is not validated. If constraint activity is validated and all other validation criteria are satisfied (see Section 4.3.1.3), then the step of ParCont is successful. The set $\mathcal{A}_{k+1}$ can be obtained given $\mathcal{A}_k$ and the constraint changing its activity (if there is one). If a constraint has changed its activity, the direction of continuation is modified according to $g_i(x) \leq 0$ if g_i has just been disactivated, or $r_i \geq 0$ if g_i has just been activated. Note that the step of ParCont following a change of active constraint certainly contains the corresponding inverse change of activity. For this particular constraint, the second (and not the first) change of constraint activity is checked within Algorithm 8. Two parallelotopes built following this whole process are shown on Figure 4.10.

This strategy for maintaining the correct active set of constraints has been incorporated in ParCont. The stopping criterion and validation criterion of ParCont based on the search domain only considers the domains of λ_1 and λ_2 (i.e. $\lambda_1, \lambda_2 \geq 0$), the non-negativeness of the multipliers r_i being maintained by the active set strategy.

4.5.2 Limitations

There are some limitations for the application of ParCont to biobjective problems. The procedure we propose for managing the set of active constraints cannot rigorously detect several changes at the same time. Such situations are indeed unstable, and correspond to solutions to an

overconstrained system of equations (for example with F, γ_i and γ_j), whose solutions cannot be certified by interval Newton procedures.

Another drawback is that ParCont still suffers from the other sources of singularities such as loss of constraint qualification. These singularities can be detected by some numerical technique, see [76], but cannot be certified. For particular cases of loss of constraint qualification, a preprocessing can be made in order to remove globally redundant constraints.

Eventually, ParCont requires an initial solution whose decision values, multipliers and active constraints must be certified to solve the system of first-order conditions. In general, numerical solvers for single objective problems provide approximate informations about these values that can be used as an initial guess for certification via interval Newton methods.

4.6 Biobjective experiments

The adaptation of ParCont to biobjective problems has been implemented in C++ using the RealPaver [44] API, implementing routines such as interval Newton methods and many other constraint solving techniques, and using Gaol [41] for interval arithmetic. These experiments have been run on a computer under Linux Ubuntu version 13.10 64-bit, with processor Intel i7-3520M 2.90GHz and 8Gb of RAM. As parallelotopes produced by ParCont lie in both decision and multiplier spaces, they are difficult to represent. Thus, we depict Pareto optimal solutions, and their image, as joined midpoint of solutions boxes sampled from each parallelotope [5]. Such a sampling is easy to obtain and is accurate as the parallelotopes are certified.

4.6.1 Illustration of change of active constraints

We illustrate the active set management on simple problems, showing different cases of constraint activation from the point of view of the decision, multiplier and objective spaces.

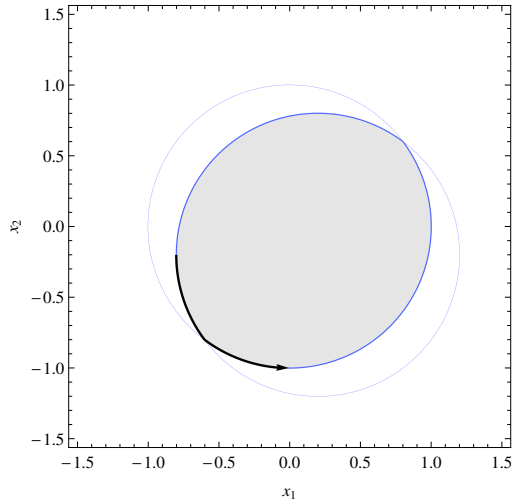
Consider the following biobjective problem with two variables and two constraints :

$$\left[\begin{array}{lll} \min & f_1(x) = x_1 & \\ & f_2(x) = x_2 & \\ \text{s.t} & g_1(x) = x_1^2 + x_2^2 - 1 & \leq 0 \\ & g_2(x) = (x_1 - 0.2)^2 + (x_2 + 0.2)^2 - 1 & \leq 0 \end{array}\right]. \tag{4.48}$$

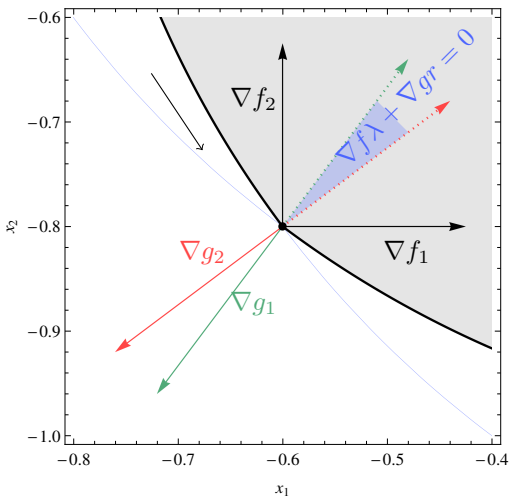
The objective space is identical to the decision space. The set of Pareto-optimal solutions obtained with ParCont, starting from the minimum of f_1 at $\hat{x}^1 = (-0.8, -0.2)^T$ with multipliers $\lambda = (0.8944, 0)^T$ and $r = (0, 0.4472)^T$ (the constraint g_2 is activated), is depicted on Figure 4.11(a). As the problem is convex, the produced set of solutions is the set of globally Pareto optimal solutions, and the continuation proceeds from the $\hat{x}^1$ to $\hat{x}^2$, the minimum of f_2. It requires 19 parallelotopes (including failures), two changes of the set of active constraints and 0.03 second to construct this Pareto optimal set.

We can see that the manifold of Pareto solutions is broken at $x = (-0.6, -0.8)^T$, with multipliers $\lambda = (0.7155, 0.5367)^T$ and $r = (0, 0.4472)^T$, as the constraint g_1 activates on the path. At this solution, the two constraints are activated. Since there are two variables, the active constraints form a square system of equations. Thus, there is no degree of freedom for the continuation to move in the space of the decision variables x. The continuation only proceeds in the multiplier

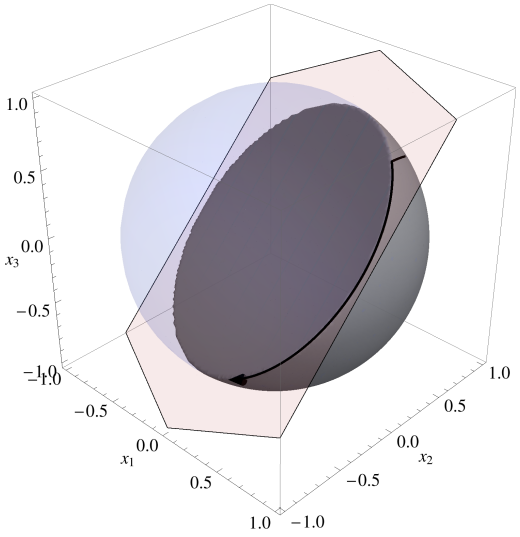
5. Parallelotopes allow a cheap and certified sampling of the solutions they include.



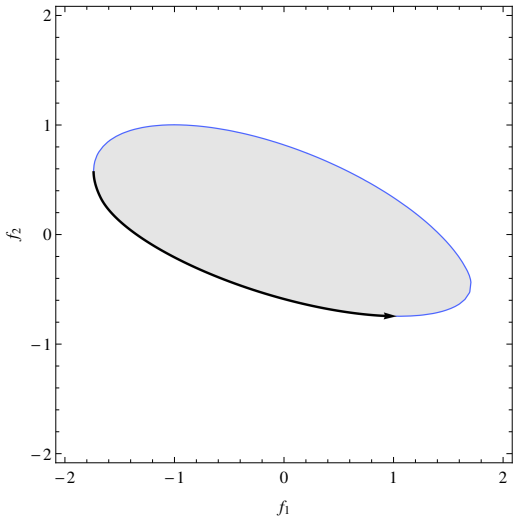
(a) Pareto optimal solutions computed by ParCont in black arrow.



(b) Zoom on the break point : the movement in the multiplier space corresponds to a movement of the weighted gradients in the cone.



(c) Decision space



(d) Objective space

Figure 4.11 – Illustration of problem 4.48

space. This is illustrated on Figure 4.11(b) from the point of view of the decision space. The weighted gradients of the objectives $\nabla f(x)\lambda$ must remain in the negative part of the weighted gradient of the constraints $\nabla g(x)r$. Once activating g_1, $\nabla f(x)\lambda$ is collinear with the negative gradient of g_2 (i.e. $r_1 = 0$). The continuation tracks the value of the multipliers such that $\nabla f(x)\lambda$ follows $\nabla g(x)r$ with increasing r_1 and decreasing r_2. Eventually, the multipliers $\lambda = (0.5367, 0.7155)^T$ and $r = (0.4472, 0)^T$ is reached, hence the constraint g_2 disactivates. The continuation can then move in the decision space, and stops tracking new solutions once reaching $\hat{x}^2 = (0, -1)^T$.

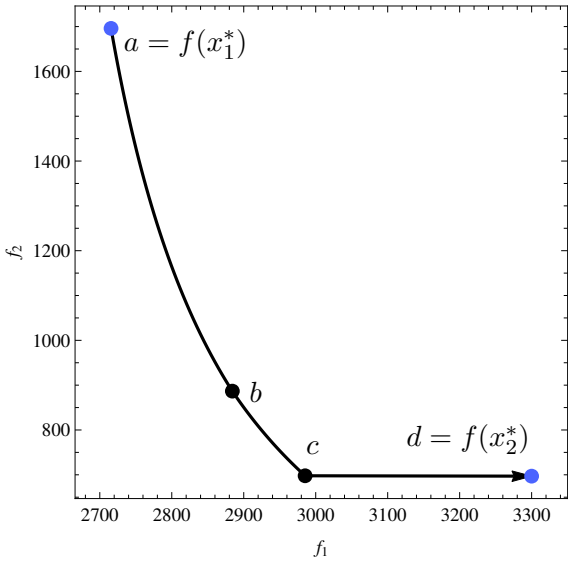


Figure 4.12 – Captured Pareto-optimal curve in the objective space for problem (A.5) (p. 148).

Consider now the following biobjective problem with three variables and two constraints :

$$\left[\begin{array}{lll} \min & f_1(x) = -x_1 - x_2 - x_3 & \\ & f_2(x) = x_2 & \\ \text{s.t} & g_1(x) = x_1^2 + x_2^2 + x_3^2 - 1 & \leq 0 \\ & g_2(x) = -0.5x_1 - x_2 + x_3 & \leq 0 \end{array}\right]. \tag{4.49}$$

The decision and objective spaces of this problem are depicted respectively on Figure 4.11(c) and 4.11(d). ParCont is applied starting from $\hat{x}^1 = (0.5735, 0.5735, 0.5735)^T$ with multipliers $\lambda = (0.7559, 0)^T$ and $r = (0.6546, 0)^T$ (the constraint g_1 is activated). As the problem is convex, all solutions found with ParCont are globally Pareto optimal, and it proceeds until reaching $\hat{x}^2$. It requires 20 parallelotopes, one change of active constraint and 0.04 second for ParCont to compute this set of solutions.

As in the previous experiment, the path of Pareto solutions is breaking at $x = (0.6667, 0.3333, 0.6667)^T$, with multipliers $\lambda = (0.7428, 0.3714)^T$ and $r = (0.5571, 0)^T$, as the constraint g_2 activates. As the problem is convex, the path followed by the multipliers λ keeps its orientation, and since there is never a square system of active constraints, the path followed by the decision variables never stops. Thus, no break is observed in the objective space. ParCont stops at $\hat{x}^2 = (0.2981, -0.7453, -0.5963)^T$ with multipliers $\lambda = (0, 0.8650)^T$ and $r = (0.3224, 0.3845)^T$. This experiment shows that activation or disactivation of constraints is not always observable in the objective space, the shape of the captured Pareto front depending on the variation of the multipliers λ.

4.6.2 Following many changes of constraint activity

We consider the biobjective problem (A.5) from [135] with 7 variables and 11 constraints, for the design of a speed reducer and presented in Appendix A (p. 147). In addition, domains of variables are considered as inequality constraints : constraints are hence denoted g_i, $i = 1 \ldots 11$ and

Table 4.1 – Change of $\mathcal{A}$: $+$ means activated, $-$ means disactivated

Constraint	$a \to a$	$a \to b$	$b \to b$	$b \to c$	$c \to c$	$c \to d$	$d \to d$
g_8	$-$	$-$	$+$	$+$	$+$	$+$	$+$
$\underline{g}_{x_4}$	$+$	$+$	$+$	$-$	$-$	$-$	$-$
$\underline{g}_{x_6}$	$+$	$-$	$-$	$-$	$-$	$-$	$-$
$\overline{g}_{x_6}$	$-$	$-$	$-$	$-$	$+$	$+$	$+$
$\underline{g}_{x_3}$	$+$	$+$	$+$	$+$	$+$	$-$	$-$
g_{10}	$-$	$-$	$-$	$-$	$-$	$-$	$+$

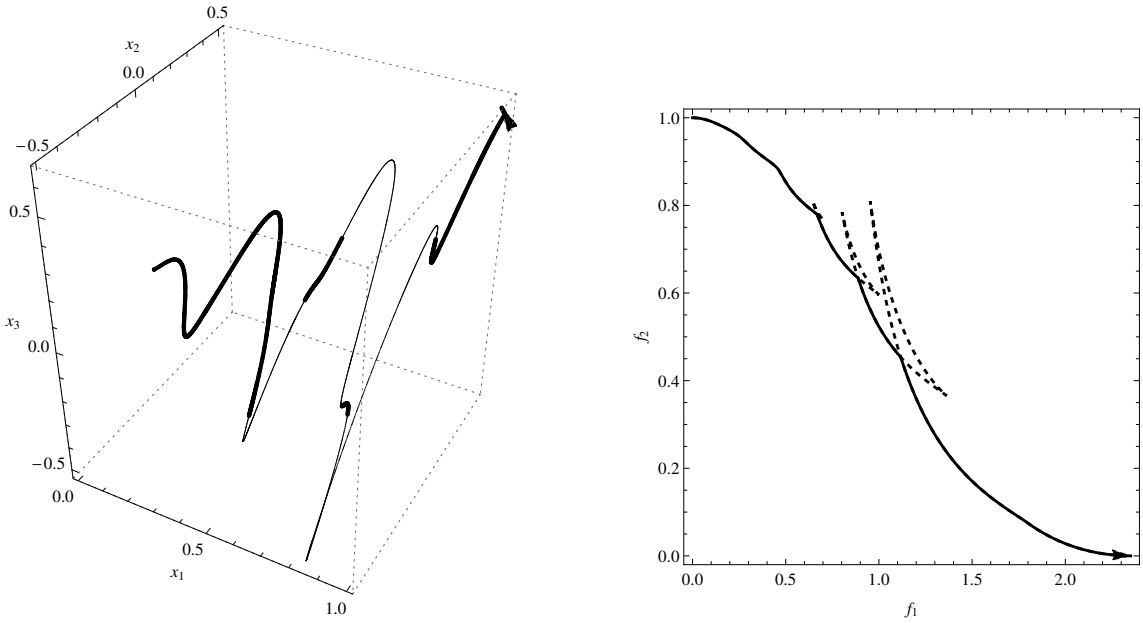
bound constraints $\underline{g}_{x_i}$ and $\overline{g}_{x_i}$ for respectively the lower and upper bound of x_i. There is hence a total of 25 constraints. We apply ParCont starting from the minimizer of f_1, and it takes approximately 0.25 seconds to produce 27 parallelotopes (including failures) enclosing the whole Pareto curve. The captured Pareto-front is shown in Figure 4.12. The points a, b, c and d correspond to solutions on which a change of $\mathcal{A}$ occurs. ParCont starts at solution a and tracks solutions until it reaches d. The obtained solutions show that the constraints $g_7, g_9, \underline{g}_{x_2}$ and $\underline{g}_{x_7}$ are always active. Constraints reported in Table 4.1 change activity during the process, the others are inactive along the Pareto front.

Consider first the path from a to c. Table 4.1 shows that 7 constraints, for 7 variables x, are active at its start point a. Hence, the active set of constraints form a square system of equations uniquely defining the values of x, as in the experiment in Section 4.6.1. In such situation, the Pareto manifold remains constant in x, but moves in the multipliers space $(\lambda, r_{\mathcal{A}})$. The disactivation of $\underline{g}_{x_6}$ is finally observed, allowing the continuation to exit a. On the path $a \to b$, 6 constraints are active, hence the continuation tracks other solutions in the decision space, and in the objective space. When reaching b, g_8 activates. Again, the system of active constraints is square such that a path $b \to b$ is followed in the multiplier space until the detection of the disactivation of $\underline{g}_{x_4}$. The path $b \to c$ has 6 active constraints, and reaches c by activating the constraint $\overline{g}_{x_6}$. ParCont leaves c by disactivating $\underline{g}_{x_3}$. It can hence further move on the path from $c \to d$. At d, g_{10} activates and the set of active constraints form a square system of equations : a path in the multiplier space is followed. The process stops at d by detecting negative objective multipliers ($\lambda < 0$).

We can see that ParCont allows to determine the set of critical constraints that change their activity along the Pareto front. This helps to determine which constraints involved which part of the front. For this problem, as the Pareto front is connected in the decision space and convex, continuation methods based on scalarization, i.e. fixing a prior parametrization of the front, would work well on this problem and would also provide such analysis of critical constraints. However, ParCont performs this analysis with rigor and, as it is performing an arclength continuation, is sure to adapt finely to the structure of the front. An example is given in the next experiment.

4.6.3 Connectivity of the Pareto front through nonoptimal solutions

Consider the problem (A.1) adapted from [72] and presented in Appendix A (p. 147) with $n = 10$. Here, we are not interested in studying changes of constraint activity (the problem is here simply bound constrained), but in illustrating particular shape of the Pareto front, namely cusps. Starting from the minimum of f_1, ParCont is applied and returns the front shown on Figure 4.13(a) (in the space (x_1, x_2, x_3)) and on Figure 4.13(b) in the objective space. It has taken 2807 parallelotopes (including failures) and 20.52 seconds for ParCont to build this set of solutions.



(a) Space of decision variables (x_1, x_2, x_3) (b) Objective space

Figure 4.13 – Pareto optimal solution of problem (A.1) (p. 147) with $n = 10$

The path followed by ParCont contains globally Pareto optimal solutions (in plain thick lines on Figure 4.13) but also other stationary solutions (in dashed thin line on Figure 4.13). It also contains cusps : particular folds in the objective space. These cusps are here the result of a switch from (locally) Pareto optimal and non-Pareto optimal solutions (actually locally Pareto optimal when maximizing the objectives) [74]. Those solutions are not interesting as results of the problem. Nevertheless, the cusps here connect the whole set of globally Pareto optimal solutions which is well exploited by ParCont, and help understanding the problem structure. In any case, those unwanted solutions can be easily filtered in a post-process.

We can note that, to our knowledge, no scalarization-based continuation that have been proposed in the literature can trace this path of Pareto-optimal solutions like ParCont and other approaches based on adapting parameterization, such as [100, 53], do. Cusps form turning points in the multiplier space λ but also in the objective space. If the multiplier λ are taken as parameters, or if a parameterization based on an approximation of the front is selected (like NBI), the induced continuation process stops at these cusps while, as we can see, they can be exploited.

4.7 Conclusion

We have presented ParCont a certified parallelotope-based continuation for tracking one-manifolds implicitly defined as the solutions to an underconstrained system of equations. We first assessed the merits and weaknesses of the approach on a set of problems inducing an underconstrained system. Then we have proposed an adaptation of ParCont to solve rigorously biobjective optimization problems via the system of first order optimality conditions. The method uses a process to detect and treat changes of constraints activity, necessary to handle problems with inequality constraints. As it is not based on a fixed parameterization, the method allows capturing locally

the structure of the Pareto optimal set, hence exploiting fully the connectivity of these solutions, contrarily to scalarization-based continuation.

ParCont and its adaptation to biobjective problems is a first step towards rigorous local searches for multiobjective optimization. Dealing with more than two objectives is currently out of reach as ParCont cannot be trivially adapted to track general m-manifolds. Nevertheless on biobjective problems, ParCont appears as an efficient local technique for rigorously recovering Pareto optimal solutions. It only needs now to be coupled with (rigorous) global search techniques, so as to attain each disconnected part of Pareto optimal solutions.

CHAPTER 5

Biobjective Branch & Bound

5.1 Introduction 119
5.2 Implementing biobjective interval Branch & Bound 119
5.2.1 Constraint propagation towards feasible nondominated solutions 120
5.2.2 Bounding 123
5.2.3 Search strategy 124
5.2.4 Convergence 125
5.3 Experiments 127
5.3.1 Comparing dominance contractor 128
5.3.2 Comparing search strategies 134
5.4 Discussion and further inverstigations 136
5.4.1 Lower and upper bounding 136
5.4.2 Search strategy 136
5.4.3 Hybridizing direct and inverse Branch & Bound 137
5.5 Conclusion 137

5.1 Introduction

Interval Branch and Bound are well known and most effective methods for solving rigorously and globally nonlinear optimization problems [48, 61, 92, 131]. Although they have been widely developed in the context of single optimization problems, few B&B exist for multiobjective optimization. We have identified in Section 3.3.3 (p. 70) two categories of such interval B&B : direct and inverse methods.

Direct methods are the straightforward adaptation of classical single objective B&B to multiobjective problems. Direct methods pave the set of weakly Pareto optimal solutions $\mathcal{X}_W^*$, i.e. in the decision space. The most advanced direct method presented in the literature is the method from Fernández and Tóth [32, 126] for biobjective problems. Inverse methods intend to view the multiobjective problem as capturing accurately the different trade-offs of the objectives. They pave the Pareto front, i.e. in the objective space, while solutions in the decision space are built with the help of interval set inversion, e.g. as Algorithm 4 (p. 74). Kubica and Woźniak [66, 64, 65] have proposed such inverse B&B.

To our knowledge, these two methods have never been compared together. Moreover, they lack important processes such as proper constraint propagation. In addition, we observe that the key component that separates the two methods is how the objective space is considered for discarding/narrowing sub-problems. In inverse methods, it is sufficient to find a feasible solution associated to an objective box of a sub-problem in order to discard every other dominated sub-problem. Since the paving in the objective space is regular, this analysis is easy to perform provided that such solutions are easily constructed. In direct method, one would like as in the single objective case to use efficiently upper bounds, e.g. via constraint contractors as seen in Section 2.4.1 (p. 45). But contrarily to the single objective case where contracting towards solutions better than an upper bound consists of introducing a single constraint, in multiobjective case this implies considering a disjunction of constraints, one for each objective multiplied by the number of feasible points in the upper bound set.

This chapter is intended to study the implementation of the two kinds of approaches and compare their performances in the case of biobjective optimization. Section 5.2 proposes an implementation of both approaches under a similar framework. Noticeably, we propose a way of handling properly upper bound solutions so as to introduce efficient narrowing processes. We will see that this technique is helpful in both direct and inverse methods. In addition, we propose some improvements of the approaches from the literature. Experiments of the proposed implementation are shown in Section 5.3. Note that the experiments proposed here are preliminary results as many aspects of the implementation of B&B are not yet investigated. Hence, in Section 5.4, we eventually discuss further investigations for improving the performances of biobjective B&B.

5.2 Implementing biobjective interval Branch & Bound

The core of the interval Branch & Bound algorithm follows Algorithm 3 (p. 44). We consider additionally that for each sub-problem with decision box $\boldsymbol{x}$ and lower bound set $\mathcal{Y}_L$ is associated an objective box $\boldsymbol{y}$, which is set for the initial sub-problem to $\boldsymbol{f}(\boldsymbol{x}^{\text{init}})$, where $\boldsymbol{x}^{\text{init}}$ is a box domain enclosing the set of feasible solutions. An upper bound set $\mathcal{Y}_U$ is maintained throughout the search [1]. The use of objective boxes allows to consider under the same framework direct and

1. The bound sets $\mathcal{Y}_L$ and $\mathcal{Y}_U$ replace the bounds y_L and y_U in Algorithm 3 (p. 44)

inverse B&B. Note however that for considering the inverse method from [66, 64, 65], the decision boxes are replaced by a triplet of set of decisions boxes as seen in Section 3.3.3 (p. 70).

Each box $\boldsymbol{y}$ stores the objective values of the induce sub-problem that is of interest. For example one can remove from $\boldsymbol{y}$ values that are dominated by a solution from the upper bound set. In this section, we will see first how we can exploit such box $\boldsymbol{y}$ in order to reduce the domain of decision variables in $\boldsymbol{x}$ accordingly. The idea is to generalize the pruning method from [32, 126] to the use of any kind of contractors, similar to what is done in single objective optimization. The technique we propose is first thought for direct methods but can also be used within inverse techniques after some modifications of the algorithm from [65]. Then bounding step and search strategies are discussed.

5.2.1 Constraint propagation towards feasible nondominated solutions

Given a sub-problem with decision box $\boldsymbol{x}$ and objective box $\boldsymbol{y}$, the following NCSP can be used to build contractors

$$\left[\begin{array}{c} f(x) = y,\ g(x) \leq 0,\ h(x) = 0 \\ x \in \boldsymbol{x}, y \in \boldsymbol{y}. \end{array}\right] \tag{5.1}$$

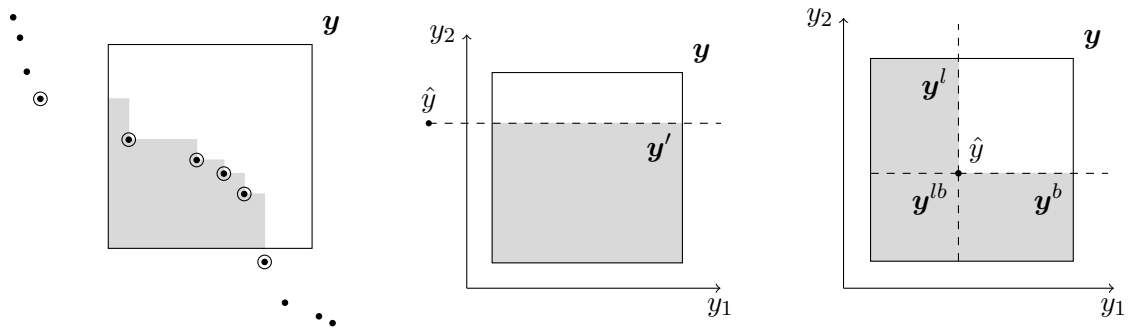
Applying contractors with respect to this NCSP on $(\boldsymbol{x}, \boldsymbol{y})$ allows pruning the objective and decision boxes. If $f(\boldsymbol{x}) \subseteq \boldsymbol{y}$, the first constraints $f(x) = y$ do not yield narrowing of $\boldsymbol{x}$. Therefore, in order to exploit these constraints, $\boldsymbol{y}$ needs to be first reduced by external means. In single objective optimization, $\boldsymbol{y}$ would be intersected with $[-\infty, \hat{y}]$, where $\hat{y}$ is a known feasible objective, such as the upper bound. As seen in Section 2.4.1 (p. 45), in the single objective case this constraint can be simply written $f(x) \leq \hat{y}$. In the biobjective case, given an objective vector $\hat{y}$ from the upper bound set $\mathcal{Y}_U$, such a rewriting induce the following disjunctive constraint [2]

$$f_1(x) \leq \hat{y}_1 \vee f_2(x) \leq \hat{y}_2. \tag{5.2}$$

This constraint has to be considered only if $\hat{y} < \overline{y}$. If $\hat{y} < \underline{y}$, then the sub-problem is dominated by $\hat{y}$ and can be discarded. Otherwise, if for example $\hat{y}_2 < \underline{y}_2$ and $\hat{y}_1 \in \boldsymbol{y}_1$, then the second constraint of the disjunction in (5.2) cannot be satisfied, thus only the first one is of interest and the disjunction reduces to the satisfaction of a single constraint. Eventually, if $\hat{y} \in \boldsymbol{y}$, the disjunction of the two constraints must be considered.

Constructive Interval Disjunction (CID) [128] appears as the most suitable approach in order to apply contractors on disjunctive constraints. Considering the NCSP (5.1) and an objective vector $\hat{y}$ from the upper bound set $\mathcal{Y}_U$, the principle is to decompose $\boldsymbol{y}$ such that each sub-box induces a single constraint of the disjunction and to apply a constraint propagator on the resulting NCSP. Eventually, the hull of the contracted sub-boxes is returned. As there are many possible objective vectors in the upper bound set, it is first required to select the ones that are relevant to be used for this decomposition. This is illustrated on Figure 5.1(a). The objective vectors of the upper bound set of interest are any vector in $\boldsymbol{y}$ and the two other vectors immediately outside $\boldsymbol{y}$ dominating $\overline{y}$. These two extreme vectors can be used to peel the box $\boldsymbol{y}$ directly. As depicted on Figure 5.1(b), this corresponds to a decomposition of $\boldsymbol{y}$ in two boxes with only one being nondominated. The vectors inside $\boldsymbol{y}$ yield to a decomposition in multiple sub-boxes as shown on Figure 5.1(c).

2. The pruning method from [32, 126] actually solves the converse of this disjunctive constraint $f(x) > \hat{y}$.



(a) Selection from $\mathcal{Y}_U$ (b) Peeling by dominance (c) Full dominance decomposition

Figure 5.1 – Cutting $\boldsymbol{y}$ in the biobjective case from $\mathcal{Y}_U$.

In the inverse method from [65], the upper bound set is implicitly given as the set of vectors $\overline{y}$ filtered by dominance taken from all sub-problems having an inner decision box. Since the produced paving in the objective space is regular, given an objective vector $\hat{y}$ from the upper bound set, a sub-problem is either entirely dominated or non-dominated, i.e. the decomposition of $\boldsymbol{y}$ is not required. Introducing explicitly different but sharper bounds in the upper bound set would allow applying the decomposition of $\boldsymbol{y}$. Assuming such a bounding strategy is given (see Section 5.2.2), it is possible to discard from $\boldsymbol{y}$ parts dominated by an objective vector from the upper bound set. We can note here that the decomposition in several sub-boxes is here considered as a splitting strategy in the context of inverse method.

Implementing the decomposition of $\boldsymbol{y}$ by dominance consists of two steps : 1) selection of the objective vectors from $\mathcal{Y}_U$ to use for the decomposition and 2) application of the decomposition. As there may be many possible candidates from $\mathcal{Y}_U$, implementing a CID in direct methods or a splitting strategy in inverse ones may yield to many sub-boxes to process which in turn can be inefficient. Vectors $\hat{y}$ that simply yield to peel the objective box $\boldsymbol{y}$, i.e. as in Figure 5.1(b), can always be selected as they induce creating only one sub-box. For the vectors $\hat{y}$ with $\hat{y} \in \boldsymbol{y}$, we propose to select the one yielding the largest dominated area, and to apply the decomposition only if the width of this part is larger than a threshold. Denote by $\boldsymbol{y}^{\succ}$ the part of $\boldsymbol{y}$ dominated by $\hat{y}$, then the decomposition of $\boldsymbol{y}$ is performed if

$$\mathrm{vol}(\boldsymbol{y}^{\succ}) \geq \rho \cdot \mathrm{vol}(\boldsymbol{y}), \tag{5.3}$$

where $0 \leq \rho < 1$. The threshold is based on the volume of the dominated part $\boldsymbol{y}^{\succ}$ and depends on a portion ρ of the volume of $\boldsymbol{y}$. Eventually, we can see that a single vector $\hat{y}$ can yield to three sub-boxes as depicted on Figure 5.1(c). The number of sub-boxes can be reduced by considering merging $\boldsymbol{y}^l$ or $\boldsymbol{y}^b$ with $\boldsymbol{y}^{lb}$. In that case, two boxes are produced. By default, we merge $\boldsymbol{y}^{lb}$ with $\boldsymbol{y}^l$ or $\boldsymbol{y}^b$ given which one has smallest volume, so as to balance the size of the two resulting sub-boxes. We will later refer to the process of peeling objective boxes as *dominance peeler*, to the CID with respect to a decomposition in two sub-boxes as *dominance CID* and to the bisection version for inverse method as *dominance bisection*, and to the CID with respect to a full decomposition as *dominance CID-full*.

The computational time complexity of the dominance decomposition depends solely on the complexity of the selection of the appropriate objective vector $\hat{y}$ from the upper bound set $\mathcal{Y}_U$. Since we consider only two objectives, ordering the objective vectors in $\mathcal{Y}_U$ (that are not domi-

Algorithm 9: Dominance contractor

Input: Sub-problem $(\boldsymbol{x}, \boldsymbol{y}, \mathcal{Y}_L)$; upper bound set $\mathcal{Y}_U$ set ; A biobjective problem
Output: A narrowed box

9.1 **if** *The sub-problem can be discarded* **then return** $\emptyset$;
9.2 $\mathcal{S} \leftarrow DominanceDecomposition(\boldsymbol{y}, \mathcal{Y}_U)$;
9.3 $\mathcal{S}' \leftarrow \emptyset$;
9.4 **for** $\boldsymbol{y}' \in \mathcal{S}$ **do**
9.5 $\quad (\boldsymbol{x}', \boldsymbol{y}') \leftarrow Contract((\boldsymbol{x}, \boldsymbol{y}'), NCSP(5.1))$;
9.6 $\quad \mathcal{S}' \leftarrow \mathcal{S}' \cup \{(\boldsymbol{x}', \boldsymbol{y}')\}$;
9.7 **end**
9.8 **return** $\square\mathcal{S}'$

nating each other) by increasing order of the first objective enforces them to be also ordered in decreasing order on the second objective. This property of biobjective problems allows to store the vectors in $\mathcal{Y}_U$ within a binary search tree ordered ascendantly with respect to one objective. Hence, inserting a new vector in $\mathcal{Y}_U$ has a $O(\log(|\mathcal{Y}_U|))$ time complexity for finding the position of insertion of this vector, and an additional $O(k)$, where $k \leq |\mathcal{Y}_U|$ designates the number of dominated objective vectors in $\mathcal{Y}_U$, in order to maintain nondominating objective vectors. For the selection of an objective vector from $\mathcal{Y}_U$ for dominance decomposition, the principle is the same : we try to insert $\overline{y}$ in $\mathcal{Y}_U$ and retrieve the k interesting objective vectors for a complexity similar to the insertion of an element in $\mathcal{Y}_U$. After the decomposition, contractors on the NCSP (5.1) are applied on each box obtained by dominance decomposition. Hence, the cost of the application of the contractors is multiplied by the number of sub-boxes to proceed. The whole process (dominance decomposition plus contractors) is further referred to *dominance contractor* [3]. Algorithm 9 depicts the dominance contractor. The set $\mathcal{S}$ in this algorithm stores the sub-boxes of the decomposition while $\mathcal{S}'$ stores these sub-boxes after the application of contractors. The hull of the sub-boxes in the set is eventually returned so as to replace the decision and objective boxes of the considered sub-problem.

The contractor used at line 9.5 follows a propagator as given by Algorithm 2 (p. 38), and that we propose to implement as follows. Contractors used are HC4 for all constraints, and BC3 for constraints with variables occurring multiple times. HC4 contractors are applied first in an arbitrary order, followed by BC3 contractors. The procedure *Update* let all the contractors be called once. Then, when the list of contractors to proceed is empty, the contractors involving variables whose domain have been sufficiently reduced are inserted in the set of contractors to proceed. The process is repeated until no domain of variables has been sufficiently reduced.

Before applying the propagator and dominance decomposition at line 9.1, several discarding tests are applied : dominance test, i.e. checking whether there exist an objective vector $\hat{y}$ from $\mathcal{Y}_U$ that strictly dominates the lower bound set $\mathcal{Y}_L$; the three first-order tests from [37] and the two monotonicity tests from [32], all presented in Section 3.3.3 (p. 70). For inverse B&B, this propagator and those discarding tests are used in the SIVIA-like process given in Algorithm 4 (p. 74).

3. Note that dominance contractors have a sense only for direct B&B. For inverse B&B, we consider possibly the application of dominance peeler followed by breaking-SIVIA, which acts as a dominance contractor. Dominance bisection is viewed as a splitting strategy although it prunes the domains of $\boldsymbol{y}$.

5.2.2 Bounding

Updating the upper bound set is done by generating feasible solutions from the different encountered sub-problems. At the difference to single objective optimization, a set of feasible solutions not dominating each other is computed by biobjective local solver. The efforts spent by a local solver clearly depend on the number of computed solutions. Currently, we do not consider having a proper biobjective local solver being applied at each encountered sub-problem. Instead, for direct methods, we propose to simply update the upper bound set as in [32, 126], i.e. taking the midpoint of the decision box $\boldsymbol{x}$ in a sub-problem. If this midpoint is feasible and is not dominated by any element from the upper bound set $\mathcal{Y}_U$, then the midpoint is inserted in $\mathcal{Y}_U$, and the other elements dominated by the midpoint are removed from $\mathcal{Y}_U$. In the presence of equality constraints, the parametric Hansen-Sengupta, with ϵ-Inflation, is used to build proofs of existence of solutions satisfying the equalities. The q variables that are not selected as parameters are determined via a Gauss elimination on the interval matrix $\nabla \boldsymbol{h}(\boldsymbol{x})$. The remaining variables are selected as parameters.

In the inverse method from [66, 64, 65], no explicit upper bound set is maintained, and the implicit upper bound set used depends on the ability to find inner boxes in the SIVIA-like process given by Algorithm 4 (p. 74). Finding quickly inner boxes avoids many bisections of the decision space in the SIVIA-like process. However, this also strongly depends on the precision given to the decision boxes. If this precision is too large with respect to the precision of the objectives and constraints, it may be difficult to produce inner boxes. Nevertheless in principle, we can note there is no need in the method from [66, 64, 65] to produce such inner boxes. Indeed, it is simply required to find solutions satisfying the constraints and whose objective lies within the objective box $\boldsymbol{y}$. Therefore, we propose to update an explicit upper bound set inside the SIVIA-like procedure. As in direct methods, midpoints are generated at each processed decision box. If a feasible midpoint [4] is found, breaking-SIVIA stops and the midpoint is inserted in $\mathcal{Y}_U$, if not dominated by any element in it. This bounding strategy hence helps to both reducing drastically the number of bisections performed by breaking-SIVIA but also to integrate bounds that can be used for dominance decomposition as seen in Section 5.2.1.

Once the upper bound set is updated, the newly inserted solution can be used to discard any sub-problem that it dominates, without passing through the pruning step. This reduces the memory consumption of the B&B at any time of the search. However, it does not avoid searching within $\mathcal{Y}_U$ for objective vectors for dominance decomposition at each treated sub-problem. This selection can be used trivially and without extra computational cost to check if the sub-problem is dominated by an objective vector from $\mathcal{Y}_U$. Hence, discarding directly sub-problems additionally yields extra computations.

The lower bound set $\mathcal{Y}_L$ at each sub-problem consists simply on the singleton containing the objective vector $\underline{y}$. This does not differ from the current biobjective B&B from the literature. To improve the lower bound, the box $\boldsymbol{y}$ is intersected with $\boldsymbol{f}(\boldsymbol{x}) \cap \boldsymbol{f}^c(\boldsymbol{x})$, i.e. using natural and centered form interval extensions of the objectives. In inverse methods, $\boldsymbol{y}$ is intersected with the hull of the evaluation of all the decision boxes contained the associated triplet.

4. In the case of equality constraints, the same certification process as in direct methods is used.

5.2.3 Search strategy

5.2.3.1 Termination

If a sub-problem cannot be discarded, then it is checked whether it is terminal or not. Criterion for termination reflects a desired accuracy of the results. Usually, this is given by a precision on the decision box and/or a precision on the distance between the lower bound of a sub-problem and an upper bound (on the sub-problem or based on $\mathcal{Y}_U$). In [32] is proposed a termination criterion that does not consider a precision on the decision box $\boldsymbol{x}$, but rather that $\boldsymbol{x}$ is proved to contain a feasible solution. Additionally, a precision on the interval evaluation of the objectives is considered. This termination criterion can be seen as bounding the errors on the evaluation of the objectives of the feasible solution. Such a feasible solution is generated by either checking the feasibility of the midpoint of $\boldsymbol{x}$ or of one of its corner points. It has shown to be sufficient for the experiment conducted in [32], but of course it is not sufficient in the general case in which certification must be used to ensure existence of feasible solutions. In the inverse methods from [66, 64, 65], the termination criterion is simply based on a precision on the objectives, while a precision on the decision boxes controls the termination of the SIVIA-like process. Note that it can be difficult to determine a precision on objectives and decision variables at the same time as the relation between the width of the decision boxes and the width of their interval evaluation of objectives is not straightforwardly known. For example, it can be necessary to reduce the width of a decision box beyond the its given precision so as to build an inner box. On the other hand, a small precision on the decision boxes can induce numerous bisections in the SIVIA-like process. For termination of direct B&B, we propose the two following criteria :

TC1. $\mathrm{wid}(\boldsymbol{x}_i) \leq \epsilon_{x_i}$;

TC2. $\mathrm{wid}(\boldsymbol{f}_j) \leq \epsilon_{f_j}$ or $\mathrm{wid}(\boldsymbol{x}_i) \leq \epsilon_{x_i}$.

For inverse methods, we use the termination criteria from [66, 64, 65]. Having different precisions for each variable and/or objective allows considering different scaling.

5.2.3.2 Splitting

If a sub-problem is not terminal, then it is split into several sub-problems which are inserted back into the list $\mathcal{S}$ of sub-problems to treat. Different splitting strategies are possible. The most used one in B&B is the bisection of a variable domain selected with respect to a particular criterion. In the direct B&B from [32], the variable domain to bisect is the largest one. In the inverse method from [66, 64, 65], no selection criterion is given for the objective box $\boldsymbol{y}$, but different selection criteria are proposed for the decision boxes in the SIVIA-like process among which two selection criteria based on the objectives dedicated to accelerate the detection of inner boxes [65]. Nevertheless, these two criteria are shown not to supplement a selection based on largest domain [65]. We then consider the following splitting criteria for decision boxes in both direct and inverse B&B.

SC1. Round Robin ;

SC2. Max Domain, i.e. the variable x_i whose $\mathrm{wid}(\boldsymbol{x}_i)$ is the largest ;

SC3. Max Sum of Relative Smear (based on objectives and potentially active constraints).

The latter criterion follows the bisection strategy from [127] shown to be robust for single objective problems. Given an interval matrix $\boldsymbol{A}$ containing the derivatives of the objectives and potentially

active constraint on $\boldsymbol{x}$, the sum of relative smear of a variable x_i is computed following

$$SumSmearRel(x_i) = \sum_j \frac{\mathrm{mag}(A_{ji})\mathrm{wid}(\boldsymbol{x}_i)}{\sum_k \mathrm{mag}(A_{jk})\mathrm{wid}(\boldsymbol{x}_k)}. \tag{5.4}$$

In the context of inverse methods, we propose to bisect objective boxes $\boldsymbol{y}$ in inverse B&B using the max domain splitting criteria SC2.

The splitting criteria SC1 and SC2 are balanced for any problem (trivially for SC1 and proof for SC2 is given in [32]), but not SC3.

Example 5.2.1 – Consider the unconstrained biobjective problem $f_1(x) = x_1$ and $f_2(x) = -x_1x_2$ with $x_1, x_2 \in [2, 4]$. For this problem, $\boldsymbol{A} = \begin{pmatrix} 1 & 0 \\ -\boldsymbol{x_2} & -\boldsymbol{x_1} \end{pmatrix}$. The relative smear with respect to the first objective is 1 for x_1 and 0 for x_2. On the second objective, the relative smear of both x_1 and x_2 is lower than 1 but greater than 0. Hence, on this problem, the sum of relative smear is enforced to be greater than one for x_1 but smaller than one for x_2. Therefore, x_2 is never selected for splitting, regardless the width of $\boldsymbol{x}_2$.

On the other hand, variables whose domain has reached the precision given by the termination criterion are not selectable for splitting. Therefore, although not balanced, termination of B&B is ensured given SC3.

5.2.3.3 Extracting

The set of sub-problems $\mathcal{S}$ is ordered with respect to some criteria, and the first sub-problem is extracted at the beginning of each iteration of the B&B. Usually, this criteria is based on the lower bound set $\mathcal{Y}_L$ of sub-problems. As each of these lower bound sets is composed of the singleton $\{\underline{y}\}$, criteria are based on this point as in [32, 126]. In the inverse method from [66, 64, 65], no ordering is given, although the ordering seems to verify that the selected sub-problem has a lower bound not dominated by any other lower bound of the set of sub-problem. We propose the following possible ordering :

OC1. increasing value of $\underline{y}_1$

OC2. increasing value of $\underline{y}_2$

OC3. increasing value of $\underline{y}_1 + \underline{y}_2$

OC4. decreasing value of hypervolume of the point $\underline{y}$.

The latter ordering requires a reference point to compute hypervolumes. Such reference point can be obtained by simply evaluating the objectives on the initial decision box. These strategies attempt to focus the search towards some specific regions of the objective space. Note that ordering with respect to a single objective (as in [32]) may yield to extract first weakly nondominated solutions, if any. All these ordering criteria allows the set of sub-problems $\mathcal{S}$ to be implemented as a binary search tree, with a constant time complexity for extracting a sub-problem (as the first one is extracted) and a $O(\log(|\mathcal{S}|))$ time complexity for insertion of sub-problems in the set.

5.2.4 Convergence

In the direct B&B from [32], convergence properties are stated. The implementation of direct B&B we propose here generalizes the pruning aspects of the B&B from [32]. Convergence of the

proposed direct B&B can then be stated analogously to [32]. On the other hand, no convergence property is stated in [66, 64, 65].

Here, we consider that the termination criterion TR2 is used with $\epsilon_f, \epsilon_x = 0$, i.e. we study the asymptotic convergence when the termination criterion never holds. The contractors and discarding tests we previously considered cannot remove any weakly Pareto optimal solutions. This implies that at any iteration of the biobjective B&B algorithm, the set $\mathcal{X}_W^*$ is contained in the sub-problem stored in $\mathcal{S}$. We want to prove here that the B&B asymptotically converge to $\mathcal{X}_W^*$. The difficulty, as it has been stated in [32], is that the ordering criteria focuses on specific part of the objective space and cannot ensure that the convergence is homogeneous at every weakly Pareto optimal solutions. This implies that B&B asymptotically accumulates on these particular regions. A trick is proposed in [32], that modifies slightly the extraction procedure so that every K iterations, the sub-problem having the largest decision box is extracted from $\mathcal{S}$ instead of the first given the ordering criteria. This however supposes that any accumulation point that is not weakly Pareto optimal can be discarded by dominance test based on a feasible solution from the upper bound set, which supposes that the upper bound set can always be successfully updated by feasible solutions. However, it is not true in the general case, e.g. in case of singularities where no proof of existence of feasible solutions can be made. Hence the proof of convergence from [32] is false due to the necessity to assert feasibility of solutions used to update the upper bound set.

On the other hand, the proposed trick highlights that the extraction strategy must yield to focus on the whole set of of weakly Pareto optimal solutions in order to ensure an anytime asymptotic convergence. To this end, consider the following ordering criteria :

OC5. sub-problems are ordered in decreasing decision box size given $\underline{y}$ is not strictly dominated by any other $\underline{y'}$ from the set of sub-problems $\mathcal{S}$.

This criterion enforces to focus on all the weakly nondominated region of the objective space. Selecting the weakly nondominated sub-problem having the larger decision box enforces that all sub-problems containing weakly Pareto optimal solutions are selected one after the other infinitely many times during the B&B. The proof of convergence of direct B&B can then be stated.

Theorem 5.2.1 (Convergence of direct biobjective Branch & Bound)**.** *Consider a biobjective optimization problem as* (3.1) *(p. 53), and consider the biobjective direct version of Algorithm 3 (p. 44) as previously described. Consider the splitting strategy is balanced and the ordering OC5 is used. In particular for each sub-problem, $\underline{y} = \underline{f(\boldsymbol{x})}$. Denote by $\mathcal{U}$ the set of accumulation points of each infinite sequence of decision boxes obtained by successive splitting of $\boldsymbol{x}^{\text{init}}$ in Algorithm 3. Eventually, suppose that $\boldsymbol{f}$, $\boldsymbol{g}$ and $\boldsymbol{h}$ are convergent interval extensions. Then $\mathcal{U} = \mathcal{X}_W^*$.*

Proof.

We first prove that $\mathcal{U} \subseteq \mathcal{X}_W^*$. From the proof of Theorem 2.3.8 (p. 40), it is easy to see that any $x \in \mathcal{U}$ is feasible, i.e. $U \subseteq \mathcal{X}$. Suppose $x \in \mathcal{U}$ such that $x \notin \mathcal{X}_W^*$, i.e. their exist a solution $x' \in \mathcal{X}_W^*$ such that $f(x') < f(x)$. Since none of the discarding tests or contractors can discard sub-problems containing weakly Pareto optimal solutions, there exist a sub-problem in $\mathcal{S}$ with decision box $\boldsymbol{x}'$ such that $x' \in \boldsymbol{x}'$. Its lower bound vector is $\underline{f(\boldsymbol{x}')}$, and verifies $\underline{f(\boldsymbol{x}')} < f(x')$. Since $x \in \mathcal{U}$ and the splitting strategy is balanced, there exist an infinite sequence of decision boxes obtained by successive splitting $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$ (and induced sub-problems) such that $\lim_{k\to\infty} \boldsymbol{x}^{(k)} = x$. In addition, since $\boldsymbol{f}$ is convergent, we have that $\lim_{k\to\infty} \boldsymbol{f}(\boldsymbol{x}^{(k)}) = f(x)$. Thus, there exist an index $\overline{k}$ such that $f(x') < \underline{f(\boldsymbol{x}^{(\overline{k})})}$, thus $\underline{f(\boldsymbol{x}')} < \underline{f(\boldsymbol{x}^{(\overline{k})})}$. Therefore, the

ordering criteria cannot select the sub-problem containing x infinitely many times, contradicting $x \in \mathcal{U}$. Hence, $\mathcal{U} \subseteq \mathcal{X}_W^*$.

Now, we prove that $\mathcal{X}_W^* \subseteq \mathcal{U}$. Consider $x \in \mathcal{X}_W^*$. Then, since no discarding tests discards weakly Pareto optimal solutions, there is a sub-problem with decision box $\boldsymbol{x}$ such that $x \in \boldsymbol{x}$. As it contains a weakly non-dominated solution, this sub-problem will be selected since it will be one of the nondominated sub-problem and it will be of maximum size when all the other nondominated sub-problems have been selected and their decision boxes reduced. Thus, and since the splitting strategy is balanced, there is an infinite sequence of decision boxes obtained by successive splitting $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$, such that $x \in \boldsymbol{x}^{(k)}, \forall k$. Hence, x is an accumulation point of the algorithm and belongs to $\mathcal{U}$.

□

Note that the ordering criteria OC5 used for the convergence proof cannot be efficiently implemented in the B&B algorithm as it requires updating the dominance relations between all sub-problems in $\mathcal{S}$ each time a sub-problem is extracted and each time a sub-problem is inserted in this set.

For inverse methods, proof of convergence of the algorithm must take into account the convergence of the SIVIA-like process. As it is mainly used as an instantiation procedure with breaking-SIVIA, we need to ensure it discovers an inner box after a finite number of iterations. Implementing the extraction process as a breadth-first search in the breaking-SIVIA allows to finitely detect inner boxes in most of the situations. However, if the inverse set to compute has an empty interior, breaking-SIVIA cannot terminate. This happens, for example, if the objective box $\boldsymbol{y}$ intersects the feasible objective space at a single point : breaking-SIVIA can at best asymptotically converge to the inverse image of this point, but never terminates. Hence, convergence of the inverse method cannot be stated in the general case.

5.3 Experiments

We have implemented the different B&B algorithms in C++ using the RealPaver [44] API, implementing routines such as constraint contractors, and using Gaol [41] for interval arithmetic. This experiment has been run on a computer under Linux Ubuntu version 13.10 64-bit, with processor Intel i7-3520M 2.90GHz and 8Gb of RAM.

The implementation follows the description given in Section 5.2. The portion ρ used in the dominance CID, dominance CID-full and dominance bisection is set to 0.125, which turns out appropriate for the considered problems. For the propagator, the improvement factor used to determine sufficiently reduced variable domains – see Section 2.3.2.3 (p. 38) – is set to 0.75 and also considers the precision on the decision variables (dependent on the problem solved). We aim here at evaluating the performances of different possible instantiations of direct and inverse B&B for biobjective problems. For direct B&B, we will only consider the termination criteria TC1.

We apply the different methods on the 5 problems given in Table 5.3, with different characteristics. These problems are detailed in Appendix A (p. 147). We set different precisions for the decision variables and objectives. The precisions on the objectives have been selected so as to reflect a maximal width of interval evaluation of the objectives on the decision boxes returned by direct B&B under termination criteria TC1. Two different precision settings is proposed for OSY : the one denoted by OSY* corresponds to the setting used for testing the inverse B&B from [65].

Table 5.1 – Problem characteristics

Problem	n	p	q	ϵ_x	ϵ_f	Comment
KIM [62]	2	0	0	0.00125	0.05	Objectives contains multiple occurrences of variables
OSY [93]	6	6	0	0.00125	0.5	Quadratic problem
OSY* [93]	6	6	0	0.25	4	Quadratic problem
SR [135]	7	11	0	0.00125	10	Applied relaxed problem
NBI [19]	5	1	2	0.00125	0.05	Contains equality constraints
MOP [120]	20	0	0	0.0125	1	Simple objectives, numerous variables

For each problem, an initial decision box $\boldsymbol{x}^{\text{init}}$ is given as bound constraints. An initial objective box $\boldsymbol{y}^{\text{init}}$ is obtained by setting $\boldsymbol{y}^{\text{init}} = \boldsymbol{f}(\boldsymbol{x}^{\text{init}})$, and is not unbounded on the tested problems. Hence, the objective vector $\overline{y}^{\text{init}}$ can be used as a reference point for computing hypervolumes, see Section 3.3.4 (p. 77). Therefore, for each tested implementation, we measure the normalized, with respect to the width of $\boldsymbol{y}^{\text{init}}$, difference of hypervolume between the lower bound set obtained by combining and filtering by dominance the lower bound $\mathcal{Y}_L$ of all sub-problems in $\mathcal{S}_{out}$, and the hypervolume of the upper bound set $\mathcal{Y}_U$. This difference of hypervolume measures the global distance between the lower bound set and the upper bound set : the lower the difference is, the better is the enclosure of the weakly Pareto optimal solutions. We also measure the CPU time, the size of $\mathcal{S}_{out}$, the number of treated sub-problems and the size of $\mathcal{Y}_U$. For inverse methods, we also count the number of decision boxes associated with the sub-problems in $\mathcal{S}_{out}$ and the decision boxes produced and treated by the SIVIA-like processes on each sub-problem during the B&B. For all the methods and problems, we have used a timeout of 3600 seconds.

5.3.1 Comparing dominance contractor

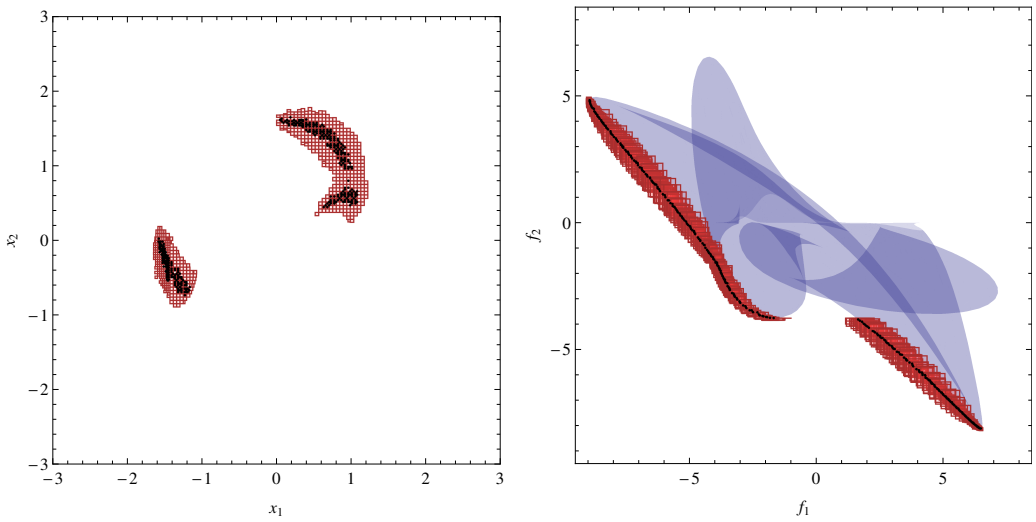
In this experiment, we evaluate the performances of the dominance contractor in the solving process of both direct and inverse B&B. Figure 5.2 depicts a paving obtained by direct and inverse B&B using dominance peeler and dominance CID (dominance bisection for inverse method).

5.3.1.1 Direct Branch & Bound

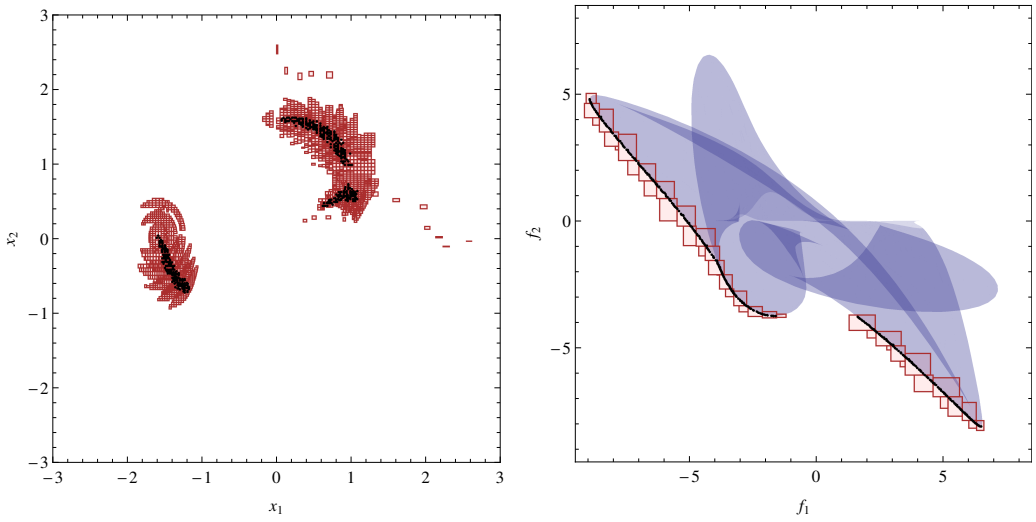
We compare here the following implementations of direct B&B :

- basic : direct B&B using OC3 and SC2 without dominance contractor, i.e. only considering contractors on the constraints of the problem (the constraint $f(x) = y$ is not taken into account in the propagator) ;
- basic+FT : basic B&B using the pruning method from Fernández and Tóth [32] presented in Section 3.3.3 (p. 70) ;
- basic+peel : basic B&B using dominance peeler (hence considers the constraint $f(x) = y$ in the propagator) ;
- basic+peel+cid : basic B&B using dominance peeler and dominance CID
- basic+peel+cid : basic B&B using dominance peeler and dominance CID-full

We also compare with our implementation of the direct B&B from [32], in two different version. These implementations are equivalent to basic+FT except that no contractors on constraints are used and the two version differ only in the usage of first-order discarding tests (not used in [32]). These two methods will be referred to as FT and FT+FO for respectively the version without and



(a) Direct method



(b) Inverse method

Figure 5.2 – Direct and inverse interval B&B with dominance peeler and dominance CID (bisection for inverse method) on KIM problem with $\epsilon_x = 0.05$ and $\epsilon_f = 1.1$. Black dots are points from the upper bound set.

with first-order discarding tests. Note that the version with these discarding tests is equivalent to basic+FT on unconstrained problems, in particular KIM and MOP. Hence, only FT is reported for these two problems. Note also that the direct B&B from [32] does not consider equality constraints, hence we do not experiment these two methods on the problem NBI. A paving of basic+peel+cid is shown on Figure 5.2(a).

Results for the direct B&B are shown in Table 5.2. Methods written in boldface are those yielding the best performances on the considered problem. On the KIM problem, we can see that none of the approaches using dominance contractors, and neither FT, performs better than the basic algorithm. Although the more advanced techniques reduce the number of treated sub-problems, this reduction is not significant compared to their extra computational cost. There is also no gain in terms of quality of the enclosure of solutions. This is explained by the complex expression of the objectives that yield large overestimation of interval evaluations, and yield a low efficiency of the contractors on the objectives in the NCSP (5.1). As FT is equivalent to basic+FT but without first-order tests, we can note that the monotonicity tests on this problem perform exactly as first-order discarding tests. Hence, using the two tests is not useful.

Considering now the OSY problem, we can note here that applying dominance peeler and dominance CID improves much the performances of the B&B, both in terms of computation time, number of treated sub-problems and quality of the result. It is not worth using dominance CID-full instead of simple dominance CID, as the latter does not reduce the number of treated sub-problems. We note also that basic+FT yields a good enclosure of the Pareto optimal solutions, but at a huge computational cost. On the other hand, the method FT+FO requires the same computation time but it treats more sub-problems and produces a worse enclosure. Constraint contractors clearly helps pruning efficiently the sub-problems, and are better than first-order tests alone while both requiring almost the same computation time. The method FT however cannot terminate before the timeout. The problem OSY* shows the same conclusions, except here that FT+FO is more computationally expensive than basic+FT. Clearly, the method FT yields poor results with about 100 times more required computation time and a worse enclosure than the best method : basic with dominance CID.

On problem NBI, the exclusive use of dominance peeler performs overall better than any other techniques. All methods have approximately similar quality of their produced enclosure. Nevertheless and as in the previous problem, the pruning technique from [32] yields to a high computational cost. Similarly, the method using exclusively dominance peeler performs better on the SR problem. It also gives better enclosures than basic algorithm and the algorithm basic+FT. We also see that neither FT or FT+FO is able to terminate the timeout. Contrarily to the problem OSY, the first-order tests cannot replace the constraint contractors on this problem.

Eventually for the MOP problem, it appears necessary to use either dominance CID or dominance CID-full to be able to terminate before the timeout. The objectives of this problem are simple, hence contractors applied on constraints $f(x) = y$ efficiently narrow the domain of every variable. However, peeling alone is not efficient since the B&B takes time to find feasible solutions to insert in $\mathcal{Y}_U$ that can be used for dominance peeler. The situation can be unlocked if initial solutions are introduced in the upper bound set. The pruning technique from [32], although applicable, is not efficient as it reduces only the domain of one variable at a time. Hence, FT and basic+FT cannot terminate before the timeout.

From the experiments reported here, we can deduce the following properties of the dominance contractors :

1. complex objective expressions (i.e. with numerous occurrences of each variable) cannot be efficiently handled by the dominance contractors.
2. constrained problems are solved efficiently with the use of the proposed dominance contractors (dominance peeler and dominance CID). They generalize the use of constraint propagation and perform better than using a given pruning technique alone, such as the one from [32].
3. when a problem has many variables but simple objective expressions, the stronger are dominance contractors (i.e. dominance CID or dominance CID-full), the quicker is the solving process.

We can note also that when dominance CID performs well, dominance CID-full does not perform much better while requiring more computation time. It is hence in general suggested to use dominance CID instead of dominance CID-full. We can also note that the original B&B method from [32] appears inefficient on these problems. In addition, the pruning technique from [32] performs clearly worse than dominance contractors. Therefore, we can conclude that dominance contractors, and in general constraint propagation techniques, improve the solving of biobjective problems with direct B&B, except when the expression of the objectives or constraints are too complex for constructing efficient contractors.

5.3.1.2 Inverse Branch & Bound

We compare here different implementation of inverse B&B :

- inverse basic : inverse B&B using OC3 and SC2 without dominance contractor ;
- inverse+peel : inverse basic B&B using dominance peeler ;
- inverse+peel+bis : inverse basic B&B using dominance peeler and dominance bisection.

In the original implementation of the inverse B&B from [65], the only constraint contractors considered are a component-wise interval Newton on the constraint $f(x) = y$ and an interval Newton on the first-order system of equations. It does not use first-order discarding tests. Currently, we do not consider an implementation of interval Newton on the first-order system of equations as it involves some difficulties for constrained problems that are not taken into account in [65]. These difficulties are developed in Section 6.2.1 (p. 140). Hence, for comparing the original inverse B&B to our implementations, we give the best results reported in [65] on the tested biobjective problems KIM and OSY* under the method name KW. Note that no hypervolume is given in the results from [65], but only the total volume of the objectives boxes. We have normalized the reported volumes the same way as the hypervolume we computed so as to evaluate the different enclosure quality, although the two measures are not directly comparable. Indeed, given a paving of $\mathcal{Y}^*_W$ obtained by inverse B&B and an upper bound set whose objective vectors is composed of the vectors $\overline{y}$ from the paving, then the total volume of the paving is less than the hypervolume difference between the global lower bound set and the upper bound set. This is caused by the additional area computed by the hypervolume due to the reference point $\overline{y}^{\text{init}}$. Computational times are also not directly comparable to the one obtained by our implementations. These two measures are then noted in italic. A paving of inverse+peel+bis is shown on Figure 5.2(b).

Results for inverse B&B are reported in Table 5.3. We have here reported the measures immediately after the main loop of the B&B, and after post-process consisting of the last call to non-breaking SIVIA over all resulting sub-problems. The method written in boldface corresponds

Table 5.2 – Result of direct methods with different use of dominance contractors.

Problem	Methods	CPU time	hypervolume	$\lvert\mathcal{S}_{out}\rvert$	# Sub-problems	$\lvert\mathcal{Y}_U\rvert$
KIM	FT	89.23	2.10E-4	62 241	307 041	81 173
	basic	45.26	2.10E-4	62 237	310 896	81 180
	basic+FT	94.24	2.10E-4	62 241	307 042	81 173
	basic+peel	75.95	2.11E-4	62 145	303 170	80 836
	basic+peel+cid	107.09	2.13E-4	62 724	298 434	78 483
	basic+peel+cid-full	133.50	2.13E-4	62 733	298 150	78 452
OSY	FT	T.O	-	-	-	-
	FT+FO	2763.68	1.50E-4	836 350	6 807 811	27 703
	basic	1072.03	1.45E-4	794 608	5 920 500	80 623
	basic+FT	2742.02	4.87E-5	875 694	3 551 810	65 204
	basic+peel	195.64	8.68E-5	180 270	1 080 528	58 859
	basic+peel+cid	185.65	6.35E-5	156 828	965 554	57 300
	basic+peel+cid-full	208.54	6.34E-5	175 121	1 045 888	58 451
OSY*	FT	77.81	2.24E-2	222 105	867 588	97
	FT+FO	2.68	2.33E-2	1 748	18 614	65
	basic	0.90	1.99E-2	1 188	6 484	245
	basic+FT	1.23	1.99E-2	1 154	5 182	225
	basic+peel	0.83	1.65E-2	909	3 900	156
	basic+peel+cid	0.62	1.57E-2	853	3 698	138
	basic+peel+cid-full	0.66	1.58E-2	853	3 708	148
NBI	basic	181.78	2.76E-7	82 027	701 260	367 288
	basic+FT	2205.11	3.14E-7	81 150	693 044	364 162
	basic+peel	153.64	2.64E-7	79 618	680 458	355 757
	basic+peel+cid	180.37	2.75E-7	76 108	632 574	328 159
	basic+peel+cid-full	169.64	3.17E-7	76 439	634 066	328 143
SR	FT	T.O	-	-	-	-
	FT+FO	T.O	-	-	-	-
	basic	168.00	5.71E-4	53 917	599 506	3 215
	basic+FT	1322.29	6.45E-4	52 228	620 906	2 910
	basic+peel	108.76	3.93E-4	38 674	385 430	3 775
	basic+peel+cid	127.76	3.92E-4	38 686	376 382	3 816
	basic+peel+cid-full	139.51	3.94E-4	38 757	374 444	3 807
MOP	FT	T.O	-	-	-	-
	basic	T.O	-	-	-	-
	basic+FT	T.O	-	-	-	-
	basic+peel	T.O	-	-	-	-
	basic+peel+cid	2123.35	4.92E-6	77 936	3229 520	75 727
	basic+peel+cid-full	2131.81	4.92E-6	77 936	3229 520	75 727

to the one yielding the best results on the considered problem. In Table 5.3, we first observe that the new dominance contractors greatly improve the overall performances. In terms of computational time and quality of enclosures, the two approaches using either the dominance peeler alone or additionally the dominance bisection are almost similar. However, none of the approaches are able to terminate before the timeout on the problem MOP. We can also see that most of the computational efforts of the inverse B&B are spent by the post-process, required to enclose accurately the set of weakly Pareto optimal solutions. On the KIM problem, we observe that our best implementation of inverse methods yields to reduce the number of decision boxes produced by the SIVIA-like process compared to the original approach from [65]. Although the comparison of CPU time and hypervolume difference must be taken with some care, we observe that enclosures quality is better in our proposed implementation, while requiring potentially more CPU time. On the problem OSY*, we can see that contractors on constraints and the new version of breaking-SIVIA we proposed have drastically reduced the number of decision boxes generated by the SIVIA-like process

Table 5.3 – Result of inverse methods with different use of dominance contractors.

Problem	Methods	CPU time	hypervolume	$\lvert\mathcal{S}_{out}\rvert$/# Dec Box	# Sub-problems/# Dec Box	$\lvert\mathcal{Y}_U\rvert$
KIM	**KW**	*75.00*	*2.47E-3*	727 / 251 491	2 736 / 864 326	415
	inverse	62.93	2.76E-3	776 / 8 893	3 700 / 364 952	904
	+ post-process	201.29	1.02E-3	776 / 63 444	3 700 / 1 184 318	48 567
	inverse+peel	36.56	3.31E-3	706 / 7433	2 384 / 111 580	1 031
	+ post-process	157.04	9.32E-4	706 / 59 681	2 384 / 539 216	41 700
	inverse+peel+bis	39.57	3.39E-3	771 / 8 407	2 330 / 118 200	1 002
	+ post-process	158.12	1.10E-3	771 / 59 547	2 330 / 549 292	42 043
OSY	inverse	230.46	4.16E-4	832 / 193 636	3 582 / 1 224 388	609
	+ post-process	T.O	-	-	-	-
	inverse+peel	19.17	3.48E-4	775 / 26 110	2 616 / 134 176	972
	+ post-process	3030.48	7.08E-5	775 / 4 460 306	2 616 / 20 603 414	48 959
	inverse+peel+bis	20.82	3.91E-4	789 / 26 126	2 620 / 135 686	954
	+ post-process	2919.65	6.70E-5	789 / 4 460 176	2 620 / 20 601 492	49 284
OSY*	KW	*72.00*	*5.40E-3*	144 / 975 999	512 / 2 260 626	29
	inverse	6.89	2.24E-2	110 / 9 225	380 / 43 144	49
	+ post-process	9.91	2.07E-2	110 / 6 682	380 / 59 112	245
	inverse+peel	0.98	1.13E-2	109 / 2 364	294 / 4 192	111
	+ post-process	2.29	1.03E-2	109 / 2 154	294 / 10 138	135
	inverse+peel+bis	0.87	1.13E-2	119 / 2 774	292 / 4 276	110
	+ post-process	2.04	1.02E-2	119 / 2 296	292 / 10 460	131
NBI	inverse	91.59	2.54E-6	362 / 6 123	1 794 / 621 336	593
	+ post-process	326.01	9.13E-7	362 / 100 424	1 794 / 2 297 456	367 419
	inverse+peel	39.96	3.35E-6	360 / 4 346	1 238 / 222 488	602
	+ post-process	210.25	1.83E-6	360 / 92 942	1 238 / 1 275 658	353 879
	inverse+peel+bis	44.60	3.08E-6	511 / 6 018	1 242 / 249 262	619
	+ post-process	241.52	3.47E-7	511 / 96 623	1 242 / 1 420 104	358 806
SR	inverse	177.64	2.22E-3	255 / 130 762	1 066 / 553 436	287
	+ post-process	1139.86	7.47E-4	255 / 108 408	1 066 / 4 338 380	3 240
	inverse+peel	18.25	1.86E-3	232 / 95 324	590 / 32 490	292
	+ post-process	151.18	4.79E-4	232 / 40 138	590 / 411 332	4 345
	inverse+peel+bis	17.60	1.86E-3	227 / 91 506	578 / 31 522	282
	+ post-process	160.23	4.79E-4	227 / 42 166	578 / 435 968	4 352
MOP	inverse	T.O	-	-	-	-
	inverse+peel	T.O	-	-	-	-
	inverse+peel+bis	T.O	-	-	-	-

compared to the original inverse B&B from [65]. This allows our implementations to terminate faster.

From this experiment, we can conclude that dominance contractors improve overall the performances of inverse B&B. The use of dominance bisection seems not to improve much the performances, but does not clearly deteriorate them neither. As it is a kind of splitting strategy, its effect within the search strategy must be further studied. Moreover, the proposed new version of breaking-SIVIA and the use of general contractors on constraints have reduced the computational efforts of the algorithm compared to the original method from [65]. Eventually, we can see that most of the search efforts are spent by the post-process.

5.3.1.3 Direct vs. Inverse

If we compare the results between direct (in Table 5.2) and inverse (in Table 5.3) methods, we can see that direct B&B performs overall better on all problems. However, we can also see that before applying the post-process on inverse methods, computation times are small. At this step, the objective boxes enclose well the shape of $\mathcal{Y}_W^*$, while information about the decision space is

mainly contained in the solutions stored in $\mathcal{Y}_U$ [5]. The enclosure of the set of weakly Pareto optimal solutions $\mathcal{X}_W^*$, i.e. in the decision space, is not sharp until the call to the nonbreaking-SIVIA, which greatly increases the computational time as many bisections of decision boxes are necessary. This also yields to improve the upper bound set. Hence, although inverse B&B allows to construct quickly, except for the problem MOP, an enclosure of the Pareto front, it is not well suited for enclosing sharply the set of weakly Pareto optimal solutions. In addition, it is difficult for a given problem to find the precisions ϵ_f and ϵ_x so as to ease the computation of inner boxes/feasible solutions and avoid unnecessary decomposition of the decision space. Direct B&B is then more reliable for the purpose of solving biobjective problems from both the decision and objective space.

5.3.2 Comparing search strategies

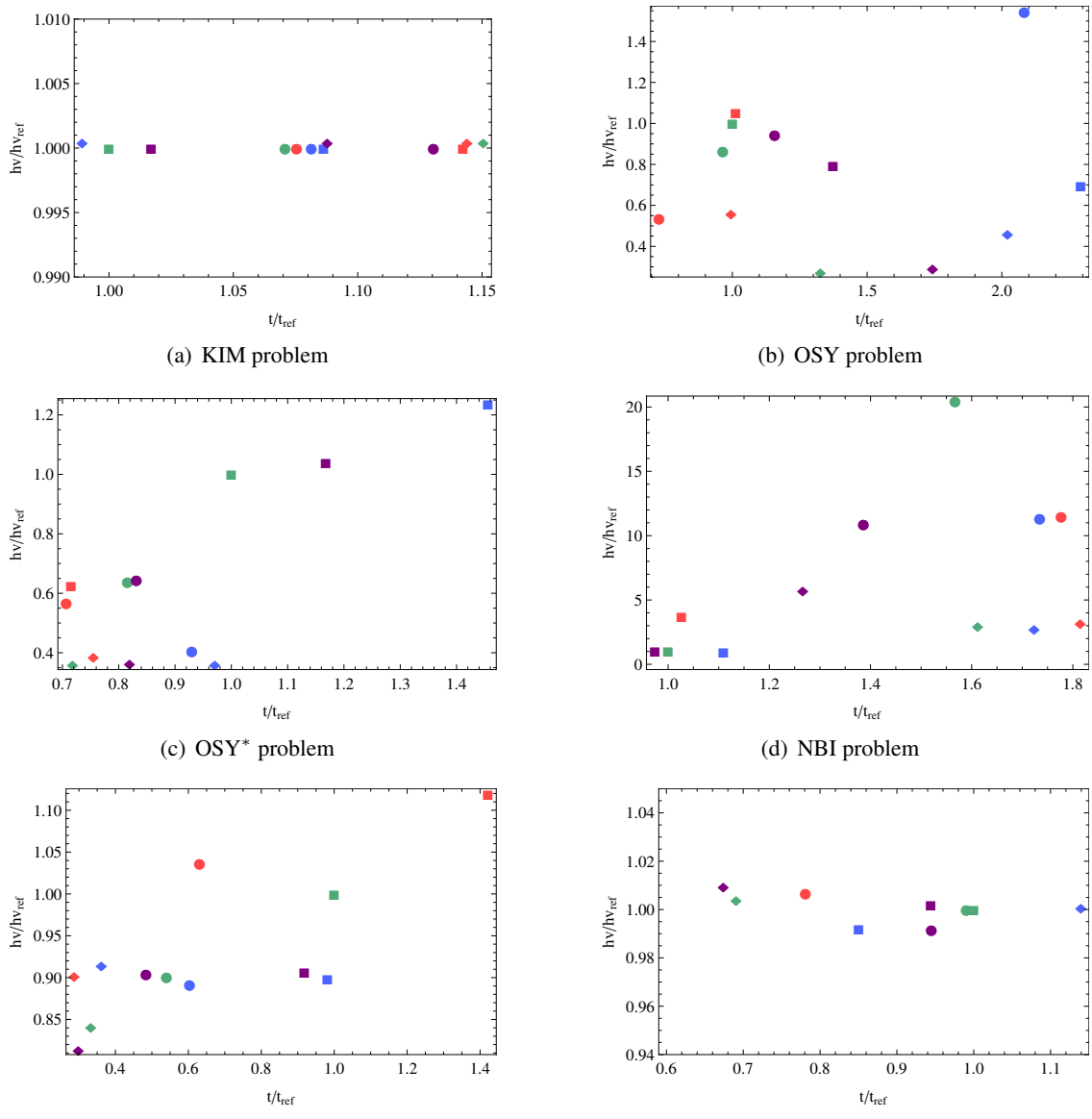
We compare here the different search strategies on direct B&B, i.e. ordering and splitting criteria. For each tested problem, we have selected the direct B&B implementation that has lead to the best results in the previous experiment. These implementations are written in boldface in Table 5.2. The different ordering criterion considered here are the OC1, OC2, OC3 and OC4 presented in Section 5.2.3.3. The splitting criteria are SC1, SC2 and SC3, which are presented in Section 5.2.3.2. The different elements and parameters of the B&B are the same. Full detailed results are presented in Appendix B (p. 151). Figure 5.3 reports the obtained computational times and hypervolumes of the considered search strategies expressed as a ratio with respect to the reference search strategy OC3+SC2 used if the previous experiment. If the timeout is attained for a method, its results are not shown.

Results obtained for the KIM problem are shown on Figure 5.3(a). Here, no search strategy appears better than another concerning enclosure quality. Maximal gain in terms of computational time with respect to OC3+SC2 is about 1%, and maximal deterioration is about 15%. On the OSY problem, whose results are shown on Figure 5.3(b), we can see that OC2, i.e. ordering with respect to the second objective, gives the best computational time especially when coupled with SC1, i.e. a round robin selection. Three times lower timing with respect to the worst possible strategy is gained, and about 25% with respect to the reference strategy. On the other hand, OC1 yields the worst timings. The best enclosure is obtained with respect to either OC3 or OC4 coupled with SC3, the relative smear splitting criterion. The gain in terms of hypervolume is about 80%. Overall, we can see that given an ordering criterion, SC3 yields better hypervolumes while the other splitting strategies give better timings, except for the ordering OC1. With a more permissive precision, i.e. on the OSY* problem on Figure 5.3(c), we can observe similar results except that SC3 yields better timings (they are all faster than the reference strategy).

On the NBI problem whose results are shown on Figure 5.3(d), the use of SC2 appears critical for obtaining overall the best performances, regardless of the extraction strategy. The use of SC1 is on the contrary yielding to high computational time, and lower enclosure quality. Consider now the SR problem whose results are shown on Figure 5.3(e). We can see that the splitting strategy is clearly affecting the CPU time, regardless the ordering criterion. SC3 yields overall better timings and hypervolumes, with a respective gain of about 65% of computation time and more than 10% of hypervolume with respect to the reference strategy.

Eventually, on the MOP problem on Figure 5.3(f), we observe that OC1 and OC2 are not stable with respect to the splitting criterion. Indeed, the combination of SC1 with OC1 does not terminate before the timeout while it does so when combined with OC2. On the other hand, SC2 or SC3

5. Without the post-process, inverse B&B resemble the inverse B&B-like method PICPA [4].



(a) KIM problem

(b) OSY problem

(c) OSY* problem

(d) NBI problem

(e) SR problem

(f) MOP problem

Figure 5.3 – Comparison of the search strategies. OC1, OC2, OC3 and OC4 are depicted in respectively blue, red, green and purple marks. Splitting criteria SC1, SC2 and SC3 are depicted respectively by circle, square and diamond marks.

combined with OC1 does terminate but not when combined with OC2. Nevertheless, regardless the splitting criteria, OC3 and OC4 always terminate before the timeout. These latter search strategies are all equivalent in terms of quality of enclosure, while SC3 allows to terminate about 30% faster with respect to the reference strategy.

It is here difficult to derive general conclusions from this experiment. Nevertheless, we observe that OC1 and OC2, i.e. ordering the sub-problems with respect to either the first and second objective, tends to be unstable. One of this criterion can lead opposite performances compared to the

other, depending on the splitting criterion used. The instability of these ordering can be explained as after the individual minima are captured by terminal sub-problems, these orderings may allow extracting in priority sub-problems that do not contain Pareto optimal solutions. It is expected that the upper bound set is well updated so as to discard those sub-problems quickly. But given the shape of the Pareto front and the splitting strategy, this cannot be ensured. On the other hand, OC3 and OC4 seem overall more robust. The best splitting strategy changes from a problem to another. In order to derive more intuitions about their influence on the performances, more experiments must be conducted.

5.4 Discussion and further inverstigations

From the previous experiments, we have deduced that direct B&B is better than inverse B&B for simultaneously enclosing both Pareto optimal solutions and the Pareto front. The efficiency of the dominance contractors clearly depends on the problem solved, but we have found some hints predicting how they behave given the problem characteristics. In addition, we have exhibited some properties of the different search strategies, although further experiments must be considered in order to assess the observations we made. We discuss here possible improvements of biobjective B&B.

5.4.1 Lower and upper bounding

Improving the lower bound $\mathcal{Y}_L$ of each sub-problem improves the detection of dominated sub-problems. It also enhances the performances induced by the search strategy, in particular the ordering criteria as it allows to focus accurately on the most promising regions of the search space, i.e. those containing Pareto optimal solutions. In addition, it helps obtaining an overall tighter enclosure of the Pareto front. For this purpose, verified linear relaxation of a sub-problem can be performed, for example as in [127] for single objective problems. In biobjective problems, such linearization leads to a biobjective linear problem that can be efficiently solved via the parametric simplex algorithm [25]. The set of solutions of the linear relaxation form a lower bound set which is expected to be tighter than the singleton made of the approximate local ideal $\underline{y}$ at a sub-problem. Consequently, the ordering criterion used must be based on a measure considering lower bound sets not made of a singleton.

The upper bounding can be improved by using more advanced local search techniques, such as continuation methods, throughout the run of the B&B. As these approaches can be computationally costly if applied at each sub-problem, heuristics must be defined so as to decide when it is best suited for applying local search. Better upper bounds yield better enclosure of the Pareto front, and more efficient dominance contractors. Local search efforts can be spent for generating a good initial upper bound set $\mathcal{Y}_U$ before running the B&B. We tried such an approach given the problems presented in the previous experiments. However, results are surprising. Either a slight improvement of performances is obtained, or performances are deteriorated. Whether the results are deteriorated depends on the search strategy. There is hence an issue between the search strategy and the exploitation of the upper bound set which needs to be studied precisely.

5.4.2 Search strategy

A drawback of the search strategies we have proposed, in particular the ordering criteria, is that they do not yield to an anytime search : specific parts of the Pareto front are explored until termination of the induced sub-problems. We have seen that the ordering criterion OC5 used in the proof of convergence is anytime, but it is difficult to efficiently implement it.

Nevertheless, a kind of anytime strategy can be simulated by considering different levels of precision for termination. Consider the termination criterion TC1, the idea is to define sequences $\epsilon_{x_i}^{(k)}$ of precision for the decision variables [6], with $k \leq \overline{k}$, such that $\epsilon_{x_i}^{(k+1)} \leq \epsilon_{x_i}^{(k)}$ and $\epsilon_{x_i} = \epsilon_{x_i}^{(\overline{k})}$ where ϵ_{x_i} designates the usual user defined precision. The B&B starts at $k = 0$. Once a sub-problem reaches the precision $\epsilon_{x_i}^{(0)}$, it is stored temporarily in $\mathcal{S}_{out}$. When there are no more sub-problems to treat, k is incremented and all the sub-problems in $\mathcal{S}_{out}$ inserted back into $\mathcal{S}$. Then, the B&B runs the new level of precision. When $k = \overline{k}$, the B&B stops.

This approach can be used in combination with any of the possible ordering and splitting criteria presented in Section 5.2.3. At the end of each precision level, the coverage of the weakly Pareto optimal solutions and of the Pareto front is homogeneous. Intuitively, this precision level approach tends to deteriorate the convergence speed compared to the original "single" precision level because of the possibility to split non-interesting sub-problems early. However, experiments suggested that this is not always the case as the whole search strategy is affected by this approach. The relations between the precision level approach and the other criteria composing the search strategy must be further investigated.

5.4.3 Hybridizing direct and inverse Branch & Bound

Until know, we have only considered direct and inverse B&B as concurrent methodologies although they have a complementary behavior. By working directly in the decision space, direct B&B paves accurately the set $\mathcal{X}_W^*$ and avoid many redundant splitting of decision boxes, contrarily to inverse B&B. On the other hand, inverse B&B can pave quickly the weakly nondominated outcomes $\mathcal{Y}_W^*$. This is mostly due to the fact that the dimension of the decision space is in general lower than the dimension of the decision space.

We can expect the two approaches to combine well. For example, we can consider applying first inverse B&B for identifying interesting objective boxes (e.g. given decision maker preferences as suggested in [65]). Then, some local search (e.g. via ParCont) can be applied for improving globally the upper bound set and filter more the set of sub-problems. Eventually, on a subset, or all, sub-problems, the induced decision boxes are filtered by dominance and exploited by direct B&B in order to accurately enclose the set of weakly Pareto optimal solutions. An immediate and basic implementation of this hybridization consists of replacing the post-process of the inverse B&B by a direct B&B. Another possibility is to use a splitting strategy that considers decision and objective boxes simultaneously, with more chances of splitting along an objective early during the search while splitting in the decision variables operates more often once the set $\mathcal{Y}_W^*$ is sufficiently well covered by the objective boxes. A study of the behavior and performances of these possible approaches seems necessary.

6. Analogously, sequence $\epsilon_{f_i}^{(k)}$ of precision for the objectives.

5.5 Conclusion

We have presented several implementations of direct and inverse B&B for nonlinear biobjective problems. We have proposed dominance contractors that are used to exploit the upper bound set via contractors on constraints based on the objective functions. Dominance contractors generalize proposed pruning techniques from the literature, noticeably those for single objective optimization, and perform better than the related pruning technique from [32]. The implementations we have proposed make use of the recent advances from the literature, such as first-order discarding tests. Eventually, we have improved the inverse method from [66, 64, 65] by integrating the proposed dominance contractors and by enhancing breaking-SIVIA with a generation of inner solutions.

Both direct and inverse approaches have been experimented on a small but representative set of biobjective benchmark problems. These experiments have helped to determine which dominance contractor is best suited for tackling given problem characteristics. We have also observed that inverse approaches, while interesting for quickly capturing the structure of the Pareto front, perform worse than direct ones for computing a sharp enclosure of the weakly Pareto optimal solutions. A study of the effect of search strategies on the performance of direct B&B is also proposed. Some intuitions have been derived, such as the instability of ordering the sub-problems with respect to a single objective. Further experiments are required to confirm these observations.

Eventually, we have discussed several possible improvements and further investigations on biobjective B&B. A depth study of search strategies is required as experiments have shown nonintuitive relations between ordering criteria, splitting criteria and the exploitation of the upper bound. Concerning bounds, it is possible to improve their quality using linear relaxation (for lower bounds sets) and local search (for the upper bound set). Better bounds improves the quality of the enclosure of the weakly Pareto optimal solutions. Moreover, considering proper lower bound sets can yield to derive other criterion for ordering the set of sub-problems. Eventually, one can think of hybridizing inverse and direct B&B in order to take advantage of their respective strengths. These further developments would help to derive more efficient implementations of biobjective B&B.

CHAPTER 6

Conclusion and Perspectives

6.1 Conclusion

In this thesis, we have presented a state-of-the-art of rigorous computation via interval analysis. Interval analysis is an efficient tool for solving with guaranteed results many different problems, in particular numerical constraint satisfaction problems and nonlinear optimization problems. Respectively, these problems are solved via Branch & Prune and Branch & Bound : global algorithms that decompose the search space into smaller and smaller parts until termination. We have also presented contractors, interval operators that narrow interval domains to consistent ones with respect to some constraints or optimality conditions. For systems of equations, we have described interval Newton methods, interval solving techniques that can assert the existence and uniqueness of solutions to the system within interval domains.

An overview of multiobjective optimization has been proposed. A rather exhaustive theoretical background is presented, giving for example notions of Pareto optimality and conditions for Pareto optimality. The traditional solving techniques based on scalarization are described. We observe in practice a continuation-like usage of scalarization techniques and a link to parametric optimization, in particular with continuation methods. Hence, we have studied the literature on the recent developments of continuation methods for multiobjective optimization problems. Eventually, global solving methods from the literature have been presented, with a particular focus on interval B&B for multiobjective problems. Two categories of interval B&B were extracted from the literature : direct and inverse B&B. The overview of this literature allowed us to see that the combination of local continuation methods with global searches has shown promising results, but such combinations have never been considered for rigorous methods such as interval B&B.

Therefore, after presenting all the necessary background on interval analysis and multiobjective optimization, we have proposed mainly two algorithms. First a local certified continuation method called ParCont, which uses interval computations via parallelotope domains and interval Newton methods to track rigorously and efficiently continuous manifold of (locally) Pareto optimal solutions of biobjective problems. Compared to other similar approaches, ParCont adapts better to the shape of the tracked manifold of solutions. It also handles inequality constraints that are source of singularities. Although limited to biobjective problems, the properties of ParCont appears ideal for a combination with global search techniques.

Second we propose different implementations of interval B&B for biobjective problems, following and improving the ones proposed in the literature. The proposed implementation allows to generalize the two different methods from the literature. In order to be able to apply general

contractors based on the objectives, we have proposed to extend the upper bound constraints in single objective optimization to biobjective problems. These new techniques are called dominance contractors. Several improvements of the inverse B&B are also proposed. The different dominance contractors and implementations of B&B are experimentally tested on a representative set of benchmark problems. We have shown that our proposed B&B performs better than the direct and inverse B&B from the literature, especially on constrained problems. We have observed that direct B&B are overall more reliable for solving biobjective problems, and hence have studied the effect of different search strategies on the performances of direct B&B. Eventually, we have discussed these preliminary results and future investigations for improving biobjective B&B.

This thesis is hence making a step forward to linking rigorous interval-based methods with nonlinear biobjective optimization solving algorithms, from both local and global point of view.

6.2 Perspectives

6.2.1 Newton-based contractor and first-order conditions

In the implementation of biobjective B&B proposed in Chapter 5, first-order conditions of the biobjective problem are only exploited via the first-order discarding tests from [37]. While efficient, the first-order conditions can be better exploited using contractors as in [64] and in single objective optimization. In particular interval Newton methods can be applied in order to build proof of existence and uniqueness of stationary solutions. If such a proof is obtained in a given sub-problem, it is no more necessary to split this sub-problem anymore. The proof hence acts like a termination criterion. Note that it is necessary to have an initial domain for the multipliers in order to apply contractors on the first-order system. Such an initial domain is computed with the application of the third discarding test based on first-order conditions from [37] shown in Theorem 3.3.6 (p. 77).

The difficulty of applying interval Newton methods on the system of first-order conditions is due to the presence of singularities, noticeably the ones involved by loss of constraint complementarity as seen in Theorem 3.2.5 (p. 59). These singularities are often encountered when considering problems containing inequality constraints. In Chapter 4, we have extended the certified continuation ParCont so as to handle such singularities. Ideally, we would like to derive a similar approach for applying interval Newton methods : determine all the different sets of possibly active constraints and apply the interval Newton on the different induced sub-systems given by (4.46) (p. 108). Detecting whether an inequality constraint g_i shall belong to the active set or not depends on if the initial domain of its induced multiplier $\boldsymbol{r}_i$ contains 0. Note that this initial domain can falsely detect that a constraint may be inactive for the first-order condition inside a decision box $\boldsymbol{x}$.

Once all possible sub-systems are obtained, the question is how to apply the interval Newton method. This clearly corresponds to solving the disjunction of all the sub-systems of equations via interval Newton, and building proof of existence and uniqueness of solutions of each constraint of this disjunction within a decision box $\boldsymbol{x}$ and induced domain of multipliers $(\boldsymbol{\lambda}, \boldsymbol{r}, \boldsymbol{s})$. The design of such interval Newton for disjunctive system of equations must be clearly established before dealing with loss of constraint complementarity. On the other hand, when no initial domain of a multiplier of a potentially active constraints within $\boldsymbol{x}$ contains 0, equivalently if no inequality constraints are considered, the sub-system (4.46) is equivalent to the original system of first-order

equations. Interval Newton can be applied directly on this system, and proof of existence of uniqueness of stationary solutions can be expected and used as a termination criterion.

6.2.2 Integration of ParCont within Branch & Bound

The convergence speed of B&B for biobjective problems is clearly affected by the necessity to cover the manifold of Pareto optimal solutions. ParCont is efficient for locally covering rigorously connected components of Pareto optimal solutions. Hence, a smart hybridization of ParCont with B&B would let the latter to focus solely on finding globally the different disconnected curves of Pareto optimal solutions. Such a proper integration is however not straightforward.

A potential approach is to construct exclusion regions from the parallelotopes produced by ParCont : regions of the search space that are proved to contain a unique manifold of locally Pareto optimal solutions. The intersection of an exclusion region with the decision box of a sub-problem can be safely discarded from the sub-problem, allowing to avoid recomputing with the B&B the Pareto optimal solutions found by ParCont. The difficulty here is that parallelotopes are certified on sub-systems of equations (4.46) (p. 108), but not on the general system of first-order optimality conditions. Hence, when constructing the exclusion region, particular attention has to be taken in order to avoid excluding Pareto optimal solutions that do not correspond to the considered sub-system, especially near stationary solutions at which there is a change in the set of active inequalities.

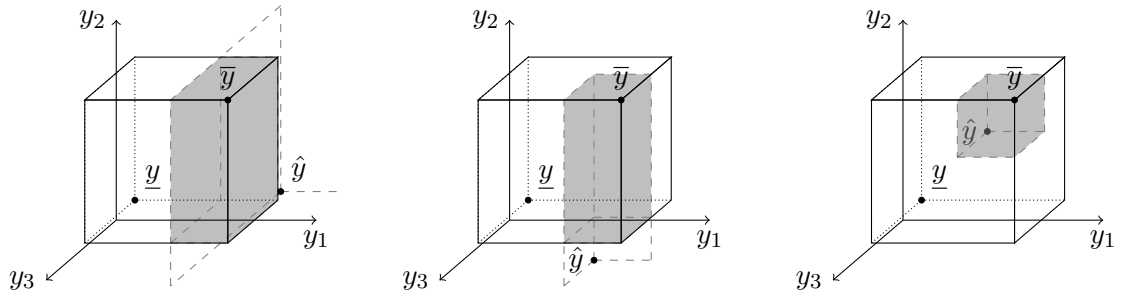
Constructing an exclusion region can be done via ϵ-inflation and interval Newton method, i.e. inflating iteratively parallelotopes and checking if an iteration of interval Newton is strictly contracting. Otherwise, exclusion regions can be computed using the technique from [115], which must be adapted to parallelotopes and biobjective problems. Note also that this latter technique requires the objectives and constraint functions to be three times differentiable. Further investigations are required so as to determine all the difficulties that are involved by these techniques, and what approach is the most appropriate.

In addition to the construction of exclusion regions, it can be interesting to design search strategies, that focus on the regions of the search space not discovered by ParCont. For example, an approach similar in principle to the two phase method [129] used for mutliobjective linear combinatorial problems seems promising.

In any case, a possible immediate collaboration of ParCont with biobjective B&B is to use the former as a local search technique for improving the upper bound set. As the solutions obtained with ParCont are rigorously computed, they can be safely inserted in the upper bound set. Moreover, the certified enclosure with parallelotopes allows to sample solutions as desired. Nevertheless, the use of ParCont as local search must be compared to other possible local search techniques in order to assess the relevance of ParCont for that purpose.

6.2.3 Towards multiobjective optimization

Extending the B&B implementations we have proposed to any number of objectives can be achieved quite easily. Discarding tests are actually applicable to the multiobjective case, except the generalized monotonicity test from [32], described in Theorem 3.3.3 (p. 75). The termination criteria, ordering strategy and splitting strategy that have been presented in Section 5.2.3 (p. 124) can be immediately translated to the multiobjective case.



(a) Objectives f_2 and f_3 are improved by $\hat{y}$. $\boldsymbol{y}$ can be decomposed along y_1 into a single nondominated sub-box.

(b) Objective f_2 is improved by $\hat{y}$. $\boldsymbol{y}$ can be decomposed by cutting along y_1 and y_3.

(c) None objective are improved by $\hat{y}$. The box $\boldsymbol{y}$ must be decomposed along all its components.

Figure 6.1 – Cutting $\boldsymbol{y}$ in the three-objective case from a vector $\hat{y}$. Dominated part are in gray.

The difficulties reside first in the implementation of efficient data structures for storing and retrieving solutions within $\mathcal{Y}_U$. To this end, we can follow the studies on the storage of archive of solutions for evolutionary algorithm as [87]. The second problem is to adapt the constraint formulation of the objectives presented in Section 5.2.1. The principles of cutting through a box $\boldsymbol{y}$ in the objective space can be derived directly, but implies numerous possible strategies, whose impact must be accurately studied.

Let suppose that m objectives are given, and let a box $(\boldsymbol{x}, \boldsymbol{y})$ with $\boldsymbol{y} \subset \mathbb{R}^m$. Given a solution $\hat{x}$ from the upper bound set $\mathcal{Y}_U$ with objective values $\hat{y} \in \mathbb{R}^m$. Assuming that $\hat{y}$ dominates $\overline{y}$ but not $\underline{y}$, and that $\hat{y}$ improves $\underline{y}$ on $0 \leq k \leq m$ objectives. This implies that a portion of $\boldsymbol{y}$ is dominated by $\hat{y}$. Decomposing $\boldsymbol{y}$ into nondominated sub-boxes depends on the objectives that are improved by $\hat{y}$. The three objective case is illustrated in Figure 6.1.

The case where $\hat{y}$ improves all but one objective of $\underline{y}$ on Figure 6.1(a) can be seen as a candidate for the dominance peeler, as the induced nondominated part of $\boldsymbol{y}$ is a single sub-box. On Figure 6.1(b), $\hat{y}$ improves only the second objective. This case can be handled as in the biobjective case, observing that this consists of applying similar cuts within the projection on (y_1, y_3). On Figure 6.1(c), $\hat{y}$ improves none of the objectives, i.e. is inside $\boldsymbol{y}$. A naive decomposition of $\boldsymbol{y}$, similar to dominance CID-full for two objectives, into 7 non-dominated sub-boxes can be made. With m objectives, this decomposition would yields to $2^m - 1$ sub-boxes. Thus, it seems to be inefficient to use such a decomposition with higher number of objectives. In addition, we have seen in the biobjective case that a complete decomposition does not improve a simpler one. Hence, another decomposition, assimilated to dominance CID in the biobjective case, would be to decompose $\boldsymbol{y}$ into m sub-boxes $\boldsymbol{y}^j$ defined as [1] :

$$\boldsymbol{y}_i^j = \begin{cases} [\underline{y}_i, \hat{y}_i] & \text{if } i = j \\ \boldsymbol{y}_i & \text{if } i > j \\ [\hat{y}_i, \overline{y}_i] & \text{if } i < j \end{cases} \tag{6.1}$$

We expect this latter decomposition to scale better on the number of objectives, although, it has to be experimented in more depth.

1. This definition considers without loss of generality a lexicographic ordering of the objectives

Extending ParCont to mulitobjective problems is however more difficult. We have discussed the problem of building a paving of general m-manifolds without gaps in Section 4.3.3 (p. 98). Other difficulties have to be considered for handling constraints activity. For example if we consider the three-objective case, a single change of constraint activity corresponds to a switch between two surfaces. Hence, the entire curve joining the surfaces contains the change of constraint activity, and must be detected. In addition, two simultaneous changes can occur at a single point while still being regular. This point corresponds to a junction of three surfaces : the one implicitly defined by the current set of active constraints and the other two by the other possible sets of active constraints. The certification of such junctions and the application of the induced changes of constraint activity is challenging.

6.2.4 Other perspectives

A short term perspective is a more complete and detailed study of the relations between the different components of the direct B&B. For example, we have observed non-intuitive relations between search strategies and the exploitation of the upper bound set. It is important to finely understand what is happening and what relates the search strategies to the other components of the B&B. This will perhaps necessitate to define new benchmark problems. Eventually, once all components of B&B have been sufficiently well studied and a stable and efficient B&B is derived, we could consider applying the method to real-world problem.

Eventually in this thesis, we have considered a posteriori decision process. Hence, a last, but not least, interesting further research direction would be to study how preferences of the decision maker, both from priori or interactive decision process [85], can be used to guide both the B&B and ParCont to the targeted solutions.

Appendix

APPENDIX A

Biobjective Benchmark problems

A.1 LZ3 Modified

The following problem is an adaptation of problem LZ3 from [72]. The adaptation yields that each variable is involved in both objectives, discarding the particular structure causing singularities in the stationary solutions, see Example 3.2.4 (p. 61).

$$\left[\begin{array}{ll} \min & f_1(x) = x_1 + \frac{2}{n-1}\sum_{i=2}^{n}(x_i - 0.8x_1\cos(6\pi x_1 + \frac{i\pi}{n}))^2 \\ & f_2(x) = 1 - x_1^2 + \frac{2}{n-1}\sum_{i=2}^{n}(x_i - 0.8x_1\sin(6\pi x_1 + \frac{i\pi}{n}))^2 \end{array}\right], \tag{A.1}$$

with $0 \leq x_1 \leq 1$ and $-1 \leq x_i \leq 1$, $i = 2, \ldots, n$.

A.2 Kim and DeWeck (KIM)

A problem with 2 objectives and 2 variables with bound constraints taken from [62] :

$$\left[\begin{array}{lll} \min & f_1(x) = & -(3(1-x_1)^2\exp(-x_1^2-(x_2+1)^2) \\ & & -10(x_1/5.0 - x_1^3 - x_2^5)\exp(-x_1^2-x_2^2) \\ & & -3\exp(-(x_1+2)^2 - x_2^2) + 0.5(2x_1+x_2)) \\ \min & f_2(x) = & -(3(1+x_2)^2\exp(-x_2^2-(1-x_1)^2) \\ & & -10(-x_2/5.0 + x_2^3 + x_1^5)\exp(-x_1^2-x_2^2) \\ & & -3\exp(-(2-x_2)^2 - x_1^2)) \end{array}\right], \tag{A.2}$$

with $-3 \leq \{x_1, x_2\} \leq 3$.

A.3 Osyczka (OSY)

The following problem with six constraints is taken from [93].

$$\left[\begin{array}{lll} \min & f_1(x) = & -(25(x_1-2)^2+(x_2-2)^2+(x_3-1)^2 \\ & & +(x_4-4)^2+(x_5-1)^2) \\ \min & f_2(x) = & x_1^2+x_2^2+x_3^2+x_4^2+x_5^2+x_6^2 \\ \text{s.t} & g_1(x) = & -(x_1+x_2-2);\ g_2(x) = -(-x_1-x_2+6) \\ & g_3(x) = & -(x_1-x_2+2);\ g_4(x) = -(-x_1+3x_2+2) \\ & g_5(x) = & -(-(x_3-3)^2-x_4+4);\ g_6(x) = -((x_5-3)^2+x_6-4) \end{array}\right], \tag{A.3}$$

with $0 \leq \{x_1, x_2, x_6\} \leq 10$, $1 \leq \{x_3, x_5\} \leq 5$ and $0 \leq x_4 \leq 6$.

A.4 NBI

The following problem with equality constraints is taken from from [19] :

$$\left[\begin{array}{lll} \min & f_1(x) = & x_1^2+x_2^2+x_3^2+x_4^2+x_5^2 \\ \min & f_2(x) = & 3x_1+2x_2-x_3/3+0.001(x_4-x_5)^3 \\ \text{s.t} & h_1(x) = & x_1+2x_2-x_3-0.5x_4+x_5-2=0 \\ & h_2(x) = & 4x_1-2x_2+0.8x_3+0.6x_4+0.5x_5^2=0 \\ & g_1(x) = & x_1^2+x_2^2+x_3^2+x_4^2+x_5^2-10 \leq 0 \end{array}\right], \tag{A.4}$$

with $-50 \leq x_i \leq 50$.

A.5 SpeedReducer (SR)

The following problem is taken from [135], it models the design of a speed reducer.

$$\left[\begin{array}{ll} \min & f_1(x) = 0.7854x_1x_2^2(\frac{10x_3^2}{3}+14.933x_3-43.0934) \\ & -1.508x_1(x_6^2+x_7^2)+7.477(x_6^3+x_7^3)+0.7854(x_4x_6^2+x_5x_7^2) \\ & f_2(x) = \sqrt{(745x_4/x_2x_3)^2+1.69\times10^7}/0.1x_6^3 \\ \text{s.t} & g_1(x) = \frac{1}{x_1x_2^2x_3}-\frac{1}{27} \leq 0;\ g_2(x) = \frac{1}{x_1x_2^2x_3^2}-\frac{1}{397.5} \leq 0 \\ & g_3(x) = \frac{x_4^3}{x_2x_3x_6^4}-\frac{1}{1.93} \leq 0;\ g_4(x) = \frac{x_5^3}{x_2x_3x_7^4}-\frac{1}{1.93} \leq 0 \\ & g_5(x) = x_2x_3-40 \leq 0;\ g_6(x) = \frac{x_1}{x_2}-12 \leq 0 \\ & g_7(x) = 5-\frac{x_1}{x_2} \leq 0;\ g_8(x) = 1.9-x_4+1.5x_6 \leq 0 \\ & g_9(x) = 1.9-x_5+1.1x_7 \leq 0;\ g_{10}(x) = f_1(x)-3300 \leq 0 \\ & g_{11}(x) = \sqrt{(745x_5/x_2x_3)^2+1.575\times10^8}/0.1x_7^3-1100 \leq 0 \end{array}\right], \tag{A.5}$$

with $2.6 \leq x_1 \leq 3.6$, $0.7 \leq x_2 \leq 0.8$, $17 \leq x_3 \leq 28$, $7.3 \leq \{x_4, x_5\} \leq 8.3$, $2.9 \leq x_6 \leq 3.9$ and $5 \leq x_7 \leq 5.5$.

A.6 MOP

The following scalable problem is taken from [120] :

$$\left[\begin{array}{lll} \min & f_1(x) = & \sum_{j\neq 1}(x_j-1)^2+(x_1-1)^4 \\ \min & f_2(x) = & \sum_{j\neq 2}(x_j+1)^2+(x_2+1)^4 \end{array}\right], \tag{A.6}$$

with $-5 \leq x_1, x_2 \leq 5$.

APPENDIX B

Detailed results of Section 5.3.2 (p. 134)

This appendix contains the detailed results used for the figures in Section 5.3.2 (p. 134).

Table B.1 – Comparing search strategies : KIM problem

Ordering	Splitting	CPU time	hypervolume	# $\mathcal{S}_{out}$	# Sub-problems	# $\mathcal{Y}_U$
OC1	SC1	48.94	2.10E-4	62 223	308 284	80 565
	SC2	49.16	2.10E-4	62 237	310 896	81 180
	SC3	44.77	2.11E-4	62 232	334 426	83 694
OC2	SC1	48.67	2.10E-4	62 223	308 284	80 565
	SC2	51.70	2.10E-4	62 237	310 896	81 180
	SC3	51.77	2.11E-4	62 232	334 420	83 695
OC3	SC1	48.46	2.10E-4	62 223	308 284	80 565
	SC2	45.26	2.10E-4	62 237	310 896	81 180
	SC3	52.07	2.11E-4	62 232	334 414	83 695
OC4	SC1	51.16	2.10E-4	62 223	308 284	80 565
	SC2	46.02	2.10E-4	62 237	310 896	81 180
	SC3	49.23	2.11E-4	62 232	334 412	83 694

Table B.2 – Comparing search strategies : OSY problem

Ordering	Splitting	CPU time	hypervolume	# $\mathcal{S}_{out}$	# Sub-problems	# $\mathcal{Y}_U$
OC1	SC1	386.81	9.83E-5	445 716	2 047 986	45 439
	SC2	425.48	4.43E-5	416 730	2 171 042	61 865
	SC3	375.16	2.93E-5	503 776	1 881 900	42 318
OC2	SC1	135.11	3.41E-5	171 320	867 482	35 012
	SC2	187.91	6.69E-5	142 900	899 884	48 243
	SC3	184.74	3.55E-5	203 801	925 842	39 577
OC3	SC1	179.00	5.48E-5	201 892	965 008	38 421
	SC2	185.65	6.35E-5	156 828	965 554	57 300
	SC3	246.35	1.75E-5	250 748	1 139 760	52 699
OC4	SC1	214.79	5.99E-5	296 099	1 279 674	40 961
	SC2	255.04	5.05E-5	268 596	1 402 388	63 729
	SC3	323.64	1.86E-5	499 620	1 931 986	49 844

Table B.3 – Comparing search strategies : OSY* problem

Ordering	Splitting	CPU time	hypervolume	# $\mathcal{S}_{out}$	# Sub-problems	# $\mathcal{Y}_U$
OC1	SC1	0.60	6.39E-3	656	4 002	152
	SC2	0.94	1.95E-2	1 110	4 834	181
	SC3	0.63	5.70E-3	628	2 944	237
OC2	SC1	0.46	8.92E-3	582	2 770	116
	SC2	0.46	9.84E-3	605	2 656	98
	SC3	0.49	6.09E-3	585	3 030	123
OC3	SC1	0.53	1.01E-2	603	2 974	98
	SC2	0.65	1.57E-2	853	3 698	138
	SC3	0.47	5.68E-3	570	2 602	136
OC4	SC1	0.54	1.02E-2	593	3 326	120
	SC2	0.76	1.64E-2	1 037	4 366	164
	SC3	0.53	5.73E-3	618	2 870	152

Table B.4 – Comparing search strategies : NBI problem

Ordering	Splitting	CPU time	hypervolume	# $\mathcal{S}_{out}$	# Sub-problems	# $\mathcal{Y}_U$
OC1	SC1	266.46	3.00E-6	85 063	1 046 460	494 333
	SC2	170.47	2.46E-7	80 737	695 582	366 561
	SC3	264.79	7.36E-7	83 574	1 125 560	493 561
OC2	SC1	273.00	3.04E-6	84 876	1 043 972	493 313
	SC2	157.70	9.81E-7	79 799	681 888	357 303
	SC3	278.86	8.51E-7	83 634	1 126 296	493 794
OC3	SC1	240.81	5.40E-6	85 183	1 046 898	494 610
	SC2	153.64	2.64E-7	79 618	680 458	355 757
	SC3	247.73	7.95E-7	83 940	1 127 622	495 752
OC4	SC1	212.96	2.89E-6	85 118	1 045 874	494 199
	SC2	149.65	2.64E-7	79 596	680 462	355 970
	SC3	194.52	1.52E-6	83 950	1 128 006	495 155

Table B.5 – Comparing search strategies : SR problem

Ordering	Splitting	CPU time	hypervolume	# $\mathcal{S}_{out}$	# Sub-problems	# $\mathcal{Y}_U$
OC1	SC1	65.67	3.51E-4	39 721	217 442	6 098
	SC2	106.74	3.54E-4	40 087	390 318	3 274
	SC3	39.37	3.60E-4	31 793	98 506	3 418
OC2	SC1	68.59	4.08E-4	40 686	218 346	7 457
	SC2	154.62	4.41E-4	39 818	502 428	3 525
	SC3	31.25	3.55E-4	29 217	90 872	1 446
OC3	SC1	58.78	3.54E-4	38 339	168 274	6 407
	SC2	108.76	3.93E-4	38 674	385 430	3 775
	SC3	36.17	3.31E-4	25 160	89 022	3 098
OC4	SC1	52.65	3.56E-4	39 347	161 800	6 425
	SC2	99.81	3.57E-4	37 600	378 614	3 771
	SC3	32.46	3.20E-4	25 314	82 428	2 888

Table B.6 – Comparing search strategies : MOP problem

Ordering	Splitting	CPU time	hypervolume	# $\mathcal{S}_{out}$	# Sub-problems	# $\mathcal{Y}_U$
OC1	SC1	T.O	-	-	-	-
	SC2	1804.85	4.88E-6	71 125	2 806 456	70 490
	SC3	2419.35	4.93E-6	76 666	3 638 612	71 510
OC2	SC1	1658.44	4.95E-6	66 533	2 534 524	73 396
	SC2	T.O	-	-	-	-
	SC3	T.O	-	-	-	-
OC3	SC1	2103.29	4.92E-6	78 219	3 201 368	79 787
	SC2	2123.35	4.92E-6	77 936	3 229 520	75 727
	SC3	1466.34	4.94E-6	45 406	2 086 998	72 180
OC4	SC1	2006.14	4.88E-6	78 188	3 197 684	80 635
	SC2	2004.64	4.93E-6	78 030	3 236 806	75 995
	SC3	1431.35	4.97E-6	46 227	2 049 200	73 668

Bibliography

[1] E. L. Allgower and K. Georg. *Introduction to Numerical Continuation Methods*. Society for Industrial and Applied Mathematics, 2003.

[2] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. SIAM, 1999.

[3] S. Askar and A. Tiwari. Multi-objective optimisation problems : A symbolic algorithm for performance measurement of evolutionary computing techniques. In M. Ehrgott et al., editor, *Evolutionary Multi-Criterion Optimization*, volume 5467 of *Lecture Notes in Computer Science*, pages 169–182. Springer Berlin Heidelberg, 2009.

[4] V. Barichard and J. Hao. A population and interval constraint propagation algorithm. In *Lecture Notes in Computer Science*, pages 88–101. Springer, 2003.

[5] C. Beltrán and A. Leykin. Certified numerical homotopy tracking. *Experimental Mathematics*, 21(1) :69–83, 2012.

[6] F. Benhamou, F. Goualard, L. Granvilliers, and J-F. Puget. Revising hull and box consistency. In *INT. CONF. ON LOGIC PROGRAMMING*, pages 230–244. MIT press, 1999.

[7] F. Benhamou, D. McAllister, and P. Van Hentenryck. CLP(Intervals) Revisited. In *International Symposium on Logic Programming*, pages 124–138, 1994.

[8] D. P. Bertsekas. Nonlinear programming. 1999.

[9] W.-J. Beyn, C. Effenberger, and D. Kressner. Continuation of eigenvalues and invariant pairs for parameterized nonlinear eigenvalue problems. *Numerische Mathematik*, 119 :489–516, 2011.

[10] P. T. Boggs and J. W. Tolle. Sequential quadratic programming. *Acta numerica*, 4 :1–51, 1995.

[11] J.-D. Boissonnat and A. Ghosh. Triangulating smooth submanifolds with light scaffolding. *Math. in Computer Science*, 4 :431–461, 2010.

[12] M.L. Brodzik. The computation of simplicial approximations of implicitly defined p-dimensional manifolds. *Computers & Mathematics with Applications*, 36(6) :93 – 113, 1998.

[13] M. Brown and R. E. Smith. Directed multi-objective optimization. *Int. J. Comput. Syst. Signal*, 6(1) :3–17, 2005.

[14] G. Chabert and L. Jaulin. Contractor programming. *Artificial Intelligence*, 173(11) :1079 – 1100, 2009.

[15] D. Chablat and P. Wenger. Working Modes and Aspects in Fully Parallel Manipulators. In *International Conference on Robotics and Automation*, volume 3, pages 1964–1969, 1998.

[16] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag, 2006.

[17] Hélène Collavizza, François Delobel, and Michel Rueher. Comparing partial consistencies. *Reliable Computing*, 5(3) :213–228, 1999.

[18] T. Csendes and D. Ratz. Subdivision direction selection in interval methods for global optimization. *SIAM Journal on Numerical Analysis*, 34(3) :922–938, 1997.

[19] I. Das and J. Dennis. Normal-Boundary Intersection : A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems. *SIAM Journal on Optimization*, 8(3) :631–657, 1998.

[20] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm : NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2) :182–197, 2002.

[21] K. Deb, A. Pratap, and T. Meyarivan. Constrained test problems for multi-objective evolutionary optimization. In E. Zitzler, L. Thiele, K. Deb, C. A. Coello Coello, and D. Corne, editors, *Evolutionary Multi-Criterion Optimization*, volume 1993 of *Lecture Notes in Computer Science*, pages 284–298. Springer Berlin Heidelberg, 2001.

[22] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable test problems for evolutionary multiobjective optimization. In A. Abraham, L. Jain, and R. Goldberg, editors, *Evolutionary Multiobjective Optimization*, Advanced Information and Knowledge Processing, pages 105–145. Springer Berlin Heidelberg, 2005.

[23] K. I. Dickson, C. T. Kelley, I. C. F. Ipsen, and I. G. Kevrekidis. Condition estimates for pseudo-arclength continuation. *SIAM J. Numer. Anal.*, 45(1) :263–276, 2007.

[24] K. Du and R.B. Kearfott. The cluster problem in multivariate global optimization. *Journal of Global Optimization*, 5(3) :253–265, 1994.

[25] M. Ehrgott. *Multicriteria Optimization (2. ed.)*. Springer, 2005.

[26] M. Ehrgott and X. Gandibleux. Bound sets for biobjective combinatorial optimization problems. *Computers & Operations Research*, 34(9) :2674 – 2694, 2007.

[27] G. Eichfelder. Scalarizations for adaptively solving multi-objective optimization problems. *Computational Optimization and Applications*, 44(2) :249–273, 2009.

[28] T. Erfani and S. Utyuzhnikov. Directed search domain : a method for even generation of the pareto frontier in multiobjective optimization. *Engineering Optimization*, 43(5) :467–484, 2011.

[29] D. Faudot and D. Michelucci. A new robust algorithm to trace curves. *Reliable Computing*, 13(4) :309–324, 2007.

[30] H. Federer. Curvature Measures. *Transactions of the American Mathematical Society*, 93(3) :418–491, 1959.

[31] J. Fernández and B. Tóth. Obtaining an outer approximation of the efficient set of nonlinear biobjective problems. *Journal of Global Optimization*, 38(2) :315–331, 2007.

[32] J. Fernández and B. Tóth. Obtaining the efficient set of nonlinear biobjective optimization problems via interval branch-and-bound methods. *Computational Optimization and Applications*, 42(3) :393–419, 2009.

[33] J. Fliege, L. Drummond, and B. Svaiter. Newton Method for Multiobjective Optimization. *SIAM Journal on Optimization*, 20(2) :602–626, 2009.

[34] F. Glover and G. Kochenberger, editors. *Handbook of Metaheuristics*. Kluwer academic publishers, 2002.

[35] D. Goldberg. What every computer scientist should know about floating-point arithmetic. *Computing Surveys*, 23(1) :5–48, 1991.

[36] A. Goldsztejn. A Branch and Prune Algorithm for the Approximation of Non-Linear AE-Solution Sets. In *Proc. of ACM SAC 2006*, pages 1650–1654, 2006.

[37] A. Goldsztejn, F. Domes, and B. Chevalier. First order rejection tests for multiple-objective optimization. *Journal of Global Optimization*, 58(4) :653–672, 2014.

[38] A. Goldsztejn and F. Goualard. Box consistency through adaptive shaving. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 2049–2054, 2010.

[39] A. Goldsztejn and L. Granvilliers. A new framework for sharp and efficient resolution of NCSP with manifolds of solutions. *Constraints*, 15(2) :190–212, 2010.

[40] A. Goldsztejn and L. Jaulin. Inner Approximation of the Range of Vector-Valued Functions. *Reliable Computing (electronic edition)*, 14 :1–23, 2010.

[41] F. Goualard. *GAOL 3.1.1 : Not Just Another Interval Arithmetic Library*. LINA, 4.0 edition, 2006.

[42] F. Goualard. How do you compute the midpoint of an interval ? *ACM Trans. Math. Softw.*, 40(2) :11, 2014.

[43] L. Granvilliers. Adaptive Bisection of Numerical CSPs. In Michela Milano, editor, *Principles and Practice of Constraint Programming*, Lecture Notes in Computer Science, pages 290–298. Springer, 2012.

[44] L. Granvilliers and F. Benhamou. Algorithm 852 : RealPaver : an interval solver using constraint satisfaction techniques. *ACM Trans. Mathematical Software*, 32(1) :138–156, 2006.

[45] D. R. Grayson and M. E. Stillman. Macaulay2, a software system for research in algebraic geometry. Available at http ://www.math.uiuc.edu/Macaulay2/.

[46] J. Guddat, H. Th. Jongen, and D. Nowack. Parametric optimization : Pathfollowing with jumps. In J. A. Gómez-Fernandez et al., editor, *Approximation and Optimization*, volume 1354 of *Lecture Notes in Mathematics*, pages 43–53. Springer, 1988.

[47] J. Guddat, F. G. Vazquez, D. Nowack, and J. Ruckmann. A modified standard embedding with jumps in nonlinear optimization. *European Journal of Operational Research*, 169(3) :1185–1206, 2006.

[48] E. Hansen and G. W. Walster. *Global Optimization Using Interval Analysis - Revised And Expanded*. CRC Press, 2003.

[49] K. Harada, J. Sakuma, and S. Kobayashi. Local search for multiobjective function optimization : pareto descent method. In *GECCO*, pages 659–666. ACM, 2006.

[50] K. Harada, J. Sakuma, S. Kobayashi, and I. Ono. Uniform sampling of local pareto-optimal solution curves by pareto path following and its applications in multi-objective GA. In *GECCO*, pages 813–820. ACM, 2007.

[51] K. Harada, J. Sakuma, I. Ono, and S. Kobayashi. Constraint-handling method for multi-objective function optimization : Pareto descent repair operator. In *Evolutionary Multi-Criterion Optimization*, volume 4403 of *LNCS*, pages 156–170. Springer, 2007.

[52] M. E. Henderson. Multiple parameter continuation : Computing implicitly defined k-manifolds. *I. J. Bifurcation and Chaos*, 12(3) :451–476, 2002.

[53] C. Hillermeier. Generalized homotopy approach to multiobjective optimization. *J. Optimization Theory and Applications*, 110(3) :557–583, 2001.

[54] C. Hillermeier. *Nonlinear Multiobjective Optimization : a Generalized Homotopy Approach*, volume 135. Birkäuser Verlag, 2001.

[55] S. Huband, P. Hingston, L. Barone, and L. While. A review of multiobjective test problems and a scalable test problem toolkit. *Evolutionary Computation, IEEE Transactions on*, 10(5) :477–506, 2006.

[56] D. Ishii, A. Goldsztejn, and C. Jermann. Interval-Based Projection Method for Under-Constrained Numerical Systems . *Constraints*, 17(4) :432–460, 2012.

[57] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis with Examples in Parameter and State Estimation, Robust Control and Robotics*. Springer-Verlag, 2001.

[58] B. Kearfott and Z. Xing. An interval step control for continuation methods. *SIAM J. Numerical Analysis*, 31(3) :pp. 892–914, 1994.

[59] R. B. Kearfott. Interval Computations : Introduction, Uses, and Resources. *Euromath*, Bulletin 2(1) :95–112, 1996.

[60] R. B. Kearfott and M. Novoa. Algorithm 681 : Intbis, a portable interval newton/bisection package. *ACM Trans. Mathematical Software*, 16(2) :152–157, 1990.

[61] R. Baker Kearfott. *Rigorous Global Search : Continuous Problems*. Kluwer Academic Publishers, 1996.

[62] I. Y. Kim and O. L. de Weck. Adaptive weighted-sum method for bi-objective optimization : Pareto front generation. *Structural and Multidisciplinary Optimization*, 29(2) :149–158, February 2005.

[63] O. Knueppel. PROFIL/BIAS - A Fast Interval Library. *Computing*, 53(3-4) :277–287, 1994.

[64] B. J. Kubica and A. Woźniak. Using the second-order information in pareto-set computations of a multi-criteria problem. In Kristján Jónasson, editor, *Applied Parallel and Scientific Computing*, volume 7134 of *Lecture Notes in Computer Science*, pages 137–147. Springer, 2012.

[65] B. J. Kubica and A. Woźniak. Tuning the interval algorithm for seeking pareto sets of multi-criteria problems. In Pekka Manninen and Per Öster, editors, *Applied Parallel and Scientific Computing*, volume 7782 of *Lecture Notes in Computer Science*, pages 504–517. Springer, 2013.

[66] B.J. Kubica and A. Woźniak. Interval methods for computing the pareto-front of a multicriterial problem. In *Proceedings of the 7th international conference on Parallel processing and applied mathematics*, PPAM'07, pages 1382–1391, 2008.

[67] A. Lara, G. Sanchez, C. Coello Coello, and O. Schütze. HCS : A new local search strategy for memetic multiobjective evolutionary algorithms. *IEEE Trans. Evolutionary Computation*, 14(1) :112–132, 2010.

[68] Y. Lebbah, C. Michel, and M. Rueher. An efficient and safe framework for solving optimization problems. *Journal of Computational and Applied Mathematics*, 199(2) :372 –

377, 2007. Special Issue on Scientific Computing, Computer Arithmetic, and Validated Numerics (SCAN 2004) Special Issue on Scientific Computing, Computer Arithmetic, and Validated Numerics (SCAN 2004).

[69] S. Leyffer. A complementarity constraint formulation of convex multiobjective optimization problems. *INFORMS Journal on Computing*, 21(2) :257–267, April 2009.

[70] O. Lhomme. Consistency Techniques for Numeric CSPs. In *IJCAI*, pages 232–238, 1993.

[71] O. Lhomme. Quick Shaving. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 1*, AAAI'05, pages 411–415. AAAI Press, 2005.

[72] H. Li and Q. Zhang. Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. *Trans. Evol. Comp*, 13(2) :284–302, April 2009.

[73] F. Logist and J. Van Impe. Novel insights for multi-objective optimisation in engineering using Normal Boundary Intersection and (Enhanced) Normalised Normal Constraint. *Structural and Multidisciplinary Optimization*, 45(3) :417–431, 2012.

[74] A. Lovison. Singular continuation : Generating piecewise linear approximations to pareto sets via global analysis. *SIAM J. Optimization*, 21(2) :463–490, 2011.

[75] A. Lovison. Global search perspectives for multiobjective optimization. *J. Global Optimization*, 57(2) :385–398, 2013.

[76] B. Lundberg and A. Poore. Numerical continuation and singularity detection methods for parametric nonlinear programming. *SIAM J. Optimization*, 3(1) :134–154, 1993.

[77] A. K. Mackworth. Consistency in networks of relations. *Artificial intelligence*, 8(1) :99–118, 1977.

[78] R.T. Marler and J.S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6) :369–395, 2004.

[79] B. Martin, A. Goldsztejn, L. Granvilliers, and C. Jermann. Certified parallelotope continuation for one-manifolds. *SIAM Journal on Numerical Analysis*, 51(6) :3373–3401, 2013.

[80] B. Martin, A. Goldsztejn, L. Granvilliers, and C. Jermann. On continuation methods for non-linear bi-objective optimization : towards a certified interval-based approach. *Journal of Global Optimization*, pages 1–14, 2014. (Online first).

[81] J-P Merlet. *Parallel robots*. Kluwer, 2000.

[82] A. Messac, A. Ismail-Yahaya, and C. Mattson. The normalized normal constraint method for generating the pareto frontier. *Structural and Multidisciplinary Optimization*, 25(2) :86–98, 2003.

[83] A. Messac and C. Mattson. Generating well-distributed sets of pareto points for engineering design using Physical Programming. *Optimization and Engineering*, 3(4) :431–450, 2002.

[84] A. Messac and C. Mattson. Normal constraint method with guarantee of even representation of complete pareto frontier. *AIAA Journal*, 42 :2101–2111, 2004.

[85] K. Miettinen. *Nonlinear Multiobjective Optimization*, volume 12 of *Int. Series in Operations Research and Management Science*. Kluwer, 1999.

[86] R. Moore. *Interval Analysis*. Prentice-Hall, 1966.

[87] Sanaz Mostaghim and Jürgen Teich. Quad-trees : A data structure for storing pareto sets in multiobjective evolutionary algorithms with elitism. In *Evolutionary Multiobjective Optimization*, pages 81–104. Springer, 2005.

[88] R. Motta, S. Afonso, and P. Lyra. A modified NBI and NC method for the solution of N-multiobjective optimization problems. *Structural and Multidisciplinary Optimization*, 46(2) :239–259, 2012.

[89] A.J. Nebro, J.J. Durillo, J. García-Nieto, C.A. Coello Coello, F. Luna, and E. Alba. Smpso : A new pso-based metaheuristic for multi-objective optimization. In *2009 IEEE Symposium on Computational Intelligence in Multicriteria Decision-Making (MCDM 2009)*, pages 66–73. IEEE Press, 2009.

[90] A. Neumaier. *Interval Methods for Systems of Equations*. Cambridge University Press, 1991.

[91] A. Neumaier. Second-order sufficient optimality conditions for local and global nonlinear programming. *Journal of Global Optimization*, 9(2) :141–151, 1996.

[92] A. Neumaier. Complete search in continuous global optimization and constraint satisfaction. *Acta Numerica*, 13 :271–369, 5 2004.

[93] A. Osyczka and S. Kundu. A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural optimization*, 10(2) :94–99, 1995.

[94] V. Pereyra. Fast computation of equispaced pareto manifolds and pareto fronts for multiobjective optimization problems. *Mathematics and Computers in Simulation*, 79(6) :1935–1947, 2009.

[95] V. Pereyra, M. Saunders, and J. Castillo. Equispaced pareto front construction for constrained bi-objective optimization. *Mathematical and Computer Modelling*, 57(9-10) :2122–2131, 2013.

[96] A.B. Poore and C.A. Tiahrt. Bifurcation problems in nonlinear parametric programming. *Mathematical Programming*, 39(2) :189–205, 1987.

[97] J.-M. Porta, L. Jaillet, and O. Bohigas. Randomized path planning on manifolds based on higher-dimensional continuation. *I. J. Robotics Research*, 31(2) :201–215, 2012.

[98] A. Potschka, F. Logist, J. Van Impe, and H. Bock. Tracing the Pareto frontier in bi-objective optimization problems by ODE techniques. *Numerical Algorithms*, 57(2) :217–233, 2011.

[99] J. Rakowska, R. Haftka, and L. Watson. An active set algorithm for tracing parametrized optima. *Structural optimization*, 3(1) :29–44, 1991.

[100] J. Rakowska, R. Haftka, and L. Watson. Multi-objective control-structure optimization via homotopy methods. *SIAM J. Optimization*, 3(3) :654–667, 1993.

[101] J. Rao and P. Papalambros. A non-linear programming continuation strategy for one parameter design optimization problems. In *ASME Design Automation Conference*, pages 77–89, 1989.

[102] M. C. Recchioni. Modified newton method in circular interval arithmetic. *J. Optim. Theory Appl.*, 86(1) :223–244, July 1995.

[103] W. C. Rheinboldt. Solution fields of nonlinear equations and continuation methods. *SIAM J. Numer. Anal.*, 17(2) :221–237, 1980.

[104] W. C. Rheinboldt. Numerical analysis of continuation methods for nonlinear structural problems. *Computers and Structures*, 13 :103–113, 1981.

[105] W. C. Rheinboldt. On a theorem of S. Smale about Newton's method for analytic mappings. *Applied Mathematics Letters*, 1(1) :69–72, 1988.

[106] E. Rigoni and S. Poles. NBI and MOGA-II, two complementary algorithms for multi-objective optimizations. In J. Branke, K. Deb, K. Miettinen, and R. E. Steuer, editors, *Practical Approaches to Multi-Objective Optimization*, number 04461 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2005.

[107] M. Ringkamp, S. Ober-Blöbaum, M. Dellnitz, and O. Schütze. Handling high-dimensional problems with multi-objective continuation methods via successive approximation of the tangent space. *Engineering Optimization*, 44(9) :1117–1146, 2012.

[108] J. Roy and R. B. Kearfott. Global optimization and singular nonlinear programs : New techniques. *Reliable Computing*, 15(3) :242–250, 2011.

[109] G.R. Ruetsch. An interval algorithm for multi-objective optimization. *Structural and Multidisciplinary Optimization*, 30 :27–37, 2005.

[110] G.R. Ruetsch. Using interval techniques of direct comparison and differential formulation to solve a multi-objective optimization problem, june 2010. Patent US 7742902 B1.

[111] S.M. Rump. A note on epsilon-inflation. *Reliable Computing*, 4 :371–375, 1998.

[112] S.M. Rump. INTLAB - INTerval LABoratory. In Tibor Csendes, editor, *Developments in Reliable Computing*, pages 77–104. Kluwer Academic Publishers, Dordrecht, 1999.

[113] S. Ruzika and M.M. Wiecek. Approximation methods in multiobjective programming. *Journal of Optimization Theory and Applications*, 126(3) :473–501, 2005.

[114] J. Sanchis, M. Martínez, X. Blasco, and J. Salcedo. A new perspective on multiobjective optimization by enhanced normalized normal constraint method. *Structural and Multidisciplinary Optimization*, 36(5) :537–546, 2008.

[115] H. Schichl, M. C. Markót, and A. Neumaier. Exclusion regions for optimization problems. *Journal of Global Optimization*, 59(2-3) :569–595, 2014.

[116] H. Schichl and A. Neumaier. Interval analysis on directed acyclic graphs for global optimization. *Journal of Global Optimization*, 33(4) :541–562, 2005.

[117] D. Scholz. The multicriteria big cube small cube method. *TOP*, 18(1) :286–302, 2010.

[118] O. Schütze. *Set oriented methods for global optimization*. PhD thesis, University of Paderborn, Germany, 2004.

[119] O. Schütze, C. Coello Coello, S. Mostaghim, E. Talbi, and M. Dellnitz. Hybridizing evolutionary strategies with continuation methods for solving multi-objective problems. *Engineering Optimization*, 40 :383–402, 2008.

[120] O. Schütze, A. Dell'Aere, and M. Dellnitz. On continuation methods for the numerical treatment of multi-objective optimization problems. In J. Branke et al., editor, *Practical Approaches to Multi-Objective Optimization*, number 04461 in Dagstuhl Seminar, 2005.

[121] O. Schütze, A. Lara, and C. Coello Coello. Evolutionary continuation methods for optimization problems. In *GECCO*, pages 651–658. ACM, 2009.

[122] O. Schütze, S. Mostaghim, M. Dellnitz, and J. Teich. Covering pareto sets by multilevel evolutionary subdivision techniques. In C.M. Fonseca, P.J.. Fleming, E. Zitzler, L. Thiele, and K. Deb, editors, *Evolutionary Multi-Criterion Optimization*, volume 2632 of *Lecture Notes in Computer Science*, pages 118–132. Springer, 2003.

[123] P. K. Shukla and K. Deb. On finding multiple pareto-optimal solutions using classical and evolutionary generating methods. *European Journal of Operational Research*, 181(3) :1630 – 1652, September 2007.

[124] K. Sindhya, K. Deb, and K. Miettinen. A Local Search Based Evolutionary Multi-objective Optimization Approach for Fast and Accurate Convergence. In G. Rudolph, T. Jansen, S. Lucas, C. Poloni, and N. Beume, editors, *Parallel Problem Solving from Nature – PPSN X*, volume 5199 of *Lecture Notes in Computer Science*, pages 815–824. Springer, 2008.

[125] S. Smale. Newton's method estimates from data at one point. In *The Merging of Disciplines in Pure, Applied and Computational Mathematics*, pages 185–196. Springer, 1986.

[126] B. Tóth. *Interval Methods for Competitive Location Problems*. PhD thesis, University of Almería, 2008.

[127] G. Trombettoni, I. Araya, B. Neveu, and G. Chabert. Inner regions and interval linearizations for global optimization. In *AAAI*, 2011.

[128] G. Trombettoni and G. Chabert. Constructive interval disjunction. In Christian Bessière, editor, *Principles and Practice of Constraint Programming – CP 2007*, volume 4741 of *Lecture Notes in Computer Science*, pages 635–650. Springer, 2007.

[129] E.L. Ulungu and J. Teghem. The two phases method : An efficient procedure to solve bi-objective combinatorial optimization problems. *Foundations of Computing and Decision Sciences*, 20(2) :149–165, 1995.

[130] S. Utyuzhnikov, P. Fantini, and M. Guenov. A method for generating a well-distributed pareto set in nonlinear multiobjective optimization. *J. Computational and Applied Mathematics*, 223(2) :820 – 841, 2009.

[131] P. Van Hentenryck, L. Michel, and Y. Deville. *Numerica : A Modeling Language for Global Optimization*. MIT press, 1997.

[132] X-H. Vu, H. Schichl, and D. Sam-Haroud. Interval propagation and search on directed acyclic graphs for numerical constraint solving. *Journal of Global Optimization*, 45(4) :499–531, 2009.

[133] *Wolfram Research inc., Mathematica*. Champaign, Illinois, Version 7.0, 2008.

[134] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1) :25–57, 2006.

[135] Z. Zhang. Immune optimization algorithm for constrained nonlinear multiobjective optimization problems. *Applied Soft Computing*, 7(3) :840 – 857, 2007.

[136] E Zitzler, K Deb, and L Thiele. Comparison of multiobjective evolutionary algorithms : empirical results. *Evolutionary Computation*, 8(2) :173–195, 2000.

[137] E Zitzler, M. Laumanns, L. Thiele, C. M. Fonseca, and V. Grunert da Fonseca. Why quality assessment of multiobjective optimizers is difficult. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '02, pages 666–674, 2002.

[138] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. da Fonseca. Performance assessment of multiobjective optimizers : an analysis and review. *Evolutionary Computation, IEEE Transactions on*, 7(2) :117–132, 2003.

List of figures

2.1 Newton methods applied to a square and a rectangular system of equations 22
2.2 Solving $x_1^2 + x_2^2 + x_1x_2 - 3 = 0$ with PC continuation after the computation of an initial solution as in Figure 2.1(b). Dashed broken lines show the path taken by predicted and corrected points. 23
2.3 Newton contractor applied to a square system of equations 26
2.4 Newton contractor applied to an underconstrained system of equations 29
2.5 Application of the interval Krawczyk operator for parallelotopes in Example 2.3.3 30
2.6 Interval-based continuation on an underconstrained problem. The initial solution is depicted by a point. 33
2.7 Consistency of the NCSP $x_1 - x_2 = 0$ and $x_1 + x_2 = 0$. 34
2.8 Example of an application of HC4 contractor procedure 36
2.9 Narrowing operator based on box-consistency applied on Example 2.3.6 37
2.10 Illustration of B&B on a single variable unconstrained problem. 43

3.1 Decision space (left) and objective space (right) of a biobjective problem from Example 3.2.1. 55
3.2 Constructing ideal, anti-ideal and nadir points ; and the CHIM in dashed line. . . 57
3.3 Illustration of singular stationary solutions 61
3.4 Figure depicting different scalarization. 65
3.5 Direct vs inverse interval Branch & Bound 72
3.6 Pruning method from [32] on single variable biobjective problem. The portion $\boldsymbol{x}^1 \cap \boldsymbol{x}^2$ contains dominated solutions and can be removed from $\boldsymbol{x}$. 76
3.7 Hypervolumes computed from two sets of nondominated points with respect to a reference point (in black triangle). The set in red dots has a larger hypervolume than the one in blue square, but some areas covered by the latter are not covered by the former. 78

4.1 Construction of the parallelotope in Example 4.3.1. 88
4.2 Bounded manifold of infinite length 95
4.3 Solution manifold of Equation (4.39). 100
4.4 Comparison of ParCont (plain lines) with BoxCont (dashed line) on the manifold induced by Equation (4.39) (logarithmic scales). Square, triangle and circle markers represent different evaluation of the derivatives, respectively : hull of parallelotopes (default mode), mean value form and formal form. 101
4.5 Solution manifold of Problem (4.40). 102
4.6 Comparison of ParCont (plain line) with BoxCont (dashed line) on the manifold induced by Equation (4.40) (logarithmic scales). 102

4.7 Results of ParCont (plain lines) and BoxCont (dashed-lines) on the manifold induced by Equation (4.41) (axis y in logarithmic scale). Colored part represents the range (maximum and minimum) of the different measures among the 10 random transformation matrices M. Square and circle markers represent respectively the differentiation using the hull of parallelotopes (default mode) and a formal expression of $G'(u)$. 104
4.8 Comparison of the different methods on the manifold induced by Equation (4.44) (axis y in logarithmic scale). Plain, dashed and dotted lines corresponds respectively to ParCont, BoxCont and NAG4M2. Colored part represents the range (maximum and minimum) of the different measures among the μ values. Square and triangle markers represent respectively differentiation using the hull of parallelotopes (default mode) and mean value form. 105
4.9 Example of a two armed robot RRRRR. 107
4.10 Bi-objective problem of Example 4.5.1. 108
4.11 Illustration of problem 4.48 112
4.12 Captured Pareto-optimal curve in the objective space for problem (A.5) (p. 148). 113
4.13 Pareto optimal solution of problem (A.1) (p. 147) with $n = 10$ 115

5.1 Cutting $\boldsymbol{y}$ in the biobjective case from $\mathcal{Y}_U$. 121
5.2 Direct and inverse interval B&B with dominance peeler and dominance CID (bisection for inverse method) on KIM problem with $\epsilon_x = 0.05$ and $\epsilon_f = 1.1$. Black dots are points from the upper bound set. 129
5.3 Comparison of the search strategies. OC1, OC2, OC3 and OC4 are depicted in respectively blue, red, green and purple marks. Splitting criteria SC1, SC2 and SC3 are depicted respectively by circle, square and diamond marks. 135

6.1 Cutting $\boldsymbol{y}$ in the three-objective case from a vector $\hat{y}$. Dominated part are in gray. 142

Liste of tables

4.1 Change of $\mathcal{A}$: + means activated, − means disactivated 114

5.1 Problem characteristics 128
5.2 Result of direct methods with different use of dominance contractors. 132
5.3 Result of inverse methods with different use of dominance contractors. 133

B.1 Comparing search strategies : KIM problem 151
B.2 Comparing search strategies : OSY problem 152
B.3 Comparing search strategies : OSY* problem 152
B.4 Comparing search strategies : NBI problem 153
B.5 Comparing search strategies : SR problem 153
B.6 Comparing search strategies : MOP problem 154

List of definitions

2.2.1 Definition (Interval) . . . 15
2.2.2 Definition (Facets and ϵ-facets) . . . 15
2.2.3 Definition (Interval hull) . . . 16
2.2.4 Definition (Interval operations) . . . 16
2.2.5 Definition (Paving) . . . 16
2.2.6 Definition (Convergent interval extension) . . . 18
2.2.7 Definition (Inclusion monotonic interval extension) . . . 18
2.3.1 Definition (Constraint satisfaction over intervals) . . . 19
2.3.2 Definition (Singularity in system of equations) . . . 23
2.3.3 Definition (Parallelotope) . . . 29
2.3.4 Definition (Contractors) . . . 32
2.3.5 Definition (Hull-consistency) . . . 33
2.3.6 Definition (Box-consistency) . . . 34
2.3.7 Definition (3B-consistency) . . . 34
2.3.8 Definition (CID-consistency) . . . 35
2.3.9 Definition (Accumulation point for sequence of boxes) . . . 40
2.3.10 Definition (Convergent infinite sequence of boxes) . . . 40
2.4.1 Definition (Optimality) . . . 41
2.4.2 Definition (Local optimality) . . . 41
2.4.3 Definition (Stationary solutions) . . . 42
2.4.4 Definition (Singular stationary solution) . . . 42
2.4.5 Definition (Active constraints) . . . 42
2.4.6 Definition (Linear Independent Constraint qualification) . . . 42
2.4.7 Definition (Potential constraint activation) . . . 45

3.2.1 Definition (Vector comparison and dominance relation) . . . 54
3.2.2 Definition (Nondominance, weak nondominance, and optimality) . . . 54
3.2.3 Definition (Local optimality) . . . 55
3.2.4 Definition (Ideal, anti-ideal and nadir points) . . . 55
3.2.5 Definition (Bound sets) . . . 56
3.2.6 Definition (Convex Hull of Individual Minima) . . . 56
3.2.7 Definition (Stationary solutions in multiobjective problems) . . . 58

Liste of examples

2.2.1 Example (Natural interval extension) 18
2.3.1 Example (Newton contractor on square systems) 26
2.3.2 Example (Newton contractor on underconstrained systems) 28
2.3.3 Example (Interval Newton on parallelotopes) 30
2.3.4 Example (Relative interval operations) 35
2.3.5 Example (Application of HC4) 36
2.3.6 Example (Box-consistency contractor) 37
2.5.1 Example (Singular optimization problem) 48

3.2.1 Example (Dominance and Pareto optimality) 54
3.2.2 Example (Ideal, anti-ideal and nadir points) 56
3.2.3 Example (Computation of the CHIM) 57
3.2.4 Example (Singular stationary solutions) 61
3.3.1 Example (Direct and inverse B&B) 71

4.3.1 Example (Construction of a new parallelotope) 88
4.3.2 Example (Certification of a new parallelotope) 89
4.3.3 Example (Contracting output sides) 89
4.3.4 Example (Validation of a parallelotope) 90
4.5.1 Example (Tracking changes of constraint activity) 108

5.2.1 Example (Unbalanced splitting criterion) 125

List of Algorithms

1 Branch & Prune 20
2 Constraint Propagation 38
3 Interval Branch & Bound 44

4 SIVIA-like procedure 74

5 Contract 85
6 N–Inflate 86
7 ParCont 87
8 Check Active Set Change 110

9 Dominance contractor 122

Algorithmes rigoureux pour l'optimisation nonlinéaire biobjectif

Rigorous algorithms for nonlinear biobjective optimization

Benjamin MARTIN

Résumé

L'Optimisation de critères nonlinéaires contradictoires, sous contraintes nonlinéaires, apparaît dans de nombreux problèmes, par exemple en ingénierie ou dans des problèmes de localisation. La résolution d'un problème avec m objectifs nécessite de calculer son ensemble de solutions dites Pareto optimales, formant des variétés continues de dimensions $m-1$ potentiellement morcelées en plusieurs parties disjointes.

Dans cette thèse, nous nous intéressons aux algorithmes rigoureux, i.e. donnant des garanties de résultats, basés sur l'analyse par intervalle pour la résolution de problèmes biobjectifs. Nous proposons une méthode de continuation certifiée qui trace localement les variétés continues de solutions optimales. Cette méthode améliore d'autres techniques similaires de la littérature en proposant une meilleure adaptation à la forme de la variété tracée, ainsi que la prise en compte des contraintes d'inégalités du problème sources de singularités. De plus, nous proposons un algorithme de Branch & Bound (B&B) qui calcule globalement un encadrement vérifié des solutions optimales. Cette méthode intègre des techniques de propagation de contraintes, exploitant notamment les bornes sur les objectifs, afin d'accélérer la résolution. Elle généralise également d'autres approches similaires de la littérature. Enfin, nous discutons la perspective de coupler ces deux méthodes. Une telle approche est prometteuse dans la mesure où le BB converge globalement mais lentement. Ceci est dû aux efforts nécessaire pour couvrir totalement les variétés de solutions, tandis que la continuation est une méthode efficace, mais locale, pour effectuer ce travail.

Mots-clés : Optimisation nonlinéaire biobjectif, Analyse par intervalle, Satisfaction de contraintes numériques, Méthodes de continuation, Branch & Bound

Abstract

Many problems, such as in engineering design or in location problems, require the optimization of several conflicting nonlinear objectives subject to nonlinear constraints. Solving a multiobjective problem involving m objectives implies computing its set of Pareto-optimal solutions, that are in general $m-1$ dimensional manifolds possibly made of several disjoint connected components.

In this thesis, we are interested in interval-based rigorous algorithms, i.e. with guaranteed results, to solve biobjective problems. We propose a certified continuation method that tracks locally a connected manifold of optimal solutions. This method supplements other techniques from the literature as it adapts finely to the shape of manifolds and deals with singularities resulting from inequality constraints in biobjective problems. We also propose an interval Branch & Bound (B&B) algorithm that globally computes a verified enclosure of the optimal solutions. This method integrates constraint propagation techniques, noticeably exploiting bounds on the objectives, in order to enhance the solving process. It also generalizes other similar approaches from the literature. Eventually, we discuss the perspective of coupling the two techniques. Such an hybrid approach is promising as the B&B converges globally, but slowly. It indeed spends many efforts for covering the manifold of solutions, whereas the continuation is an efficient, but local, technique for building such covering.

Keywords: Biobjective nonlinear optimization, Interval analysis, Numerical constraint satisfaction, Continuation methods, Branch & Bound