RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MENTOURI CONSTANTINE 1

FACULTÉ DES SCIENCES EXACTES

DÉPARTEMENT DE MATHÉMATIQUES

N°d'ordre : 119\ Ds \2017
N° de série : 03\ Mat \2017

THÈSE

Présentée pour l'obtention du diplôme de

DOCTORAT EN SCIENCES MATHÉMATIQUES

Thème

Étude et mise en œuvre d'algorithmes d'optimisation pour résoudre certaines classes de problèmes pratiques

Option : Optimisation globale

Par

AAID DJAMEL

Soutenue le : 29 juin 2017 à l'université de Constantine 1

Devant le jury

DALAH Mohamed	**Pr**	**U. CONSTANTINE**	**Président**
AIB Nadia	**M.C**	**U. CONSTANTINE**	**Examinatrice**
ZIADI Abdelkader	**Pr**	**U. SETIF**	**Examinateur**
OUANES Mohand	**Pr**	**U. TIZI – OUZOU**	**Rapporteur**

Remercîment

Je tiens à exprimer mes plus vifs remerciements à ALLAH pour m'avoir facilité ce travail. Je tiens à exprimer tout d'abord ma profonde reconnaissance à mon directeur de recherche, le Professeur Mohand Ouanes , qui m'a toujours fait confiance et qui a su m'encourager à surmonter les diverses difficultés. Je tiens aussi à le remercier pour son appui scientifique et pour sa grande disponibilité qui a été essentielle pour la progression de ma recherche. Je remercie également les membres de jury qui ont accepté de juger ce travail, je suis très reconnaissant à leurs remarques et commentaires qui m'ont aidé beaucoup pour mieux présenter ce document.

Je remercie aussi mon ami, Madame Aib Nadia : maître de conférence à l'université de Constantine 1, pour son aide, ses conseils et ses encouragements.

Mes remerciements s'adressent aussi à toute l'équipe administrative du département de mathématique de Constantine, le comité et le conseil scientifique d'avoir entrepris avec une grande souplesse les démarches nécessaires pour la soutenance.

Enfin, je remercie tous ceux qui m'ont aidé de près ou de loin quel qu’ 'ils soient et d'où qu'ils soient.

A ma mère, à mon père et à ma femme, ils sauront pourquoi...

Résumé

Dans cette thèse, nous nous sommes intéressés à un sujet d'application très intéressant à savoir : l'étude et la mise en œuvre d'algorithmes d'optimisations pour résoudre certaines classes de problèmes pratiques tels que les problèmes de transport à quatres indices avec capacités. Initialement le problème de transport est étudié dans un contexte de programmation linéaire avec certaines particularités. On fait appel à des notions d'économie mathématiques, d'analyse convexe et de recherche opérationnelle.

Nous avons obtenu des résultats originaux en proposant une méthode efficace pour résoudre le problème difficile du cas dégénéré.

Par ailleurs, on est intéressé aux problèmes d'optimisation non convexes. Afin de résoudre ce type de problème nous avons proposé une approche qui consiste à construire plusieurs fonctions quadratiques en morceau au lieu d'une seule quadratique minorant la fonction objectif.

Les résultats obtenus sont encourageants et permettent d'affirmer dans certains cas que telle quadratique est préférable aux autres.

L'étude algorithmique de la méthode a donné lieu à un constat encourageant. L'étude menée est le cas d'une seule variable. Afin de généraliser cette étude au cas de plusieurs variables tout en préservant les avantages requis. Une combinaison de la méthode Aliénor avec la technique de séparation et évaluation a été mise au point avec succès.

Nous avons obtenu des résultats satisfaisants, ces derniers montrent que cette méthode est performante par rapport aux méthodes actuelles.

Mots clés : Optimisation Globale, Problème de Transport, Branch & Bound, Fonction Borne.

Abstract

In this thesis, we focused on a very interesting subject of application namely: the study and implementation of optimization algorithms to solve some classes of practical problems such as transport problems with capacity in four indices.

Initially the problem of transport is studied in a context of linear programming with certain particularities. We call on at the same time to mathematical notions of economy, convex analysis and operational research. We obtained original results by proposing an effective method to solve the degenerate problems.

Furthermore, we are interested in non-convex optimization problems. In order to solve this type of problem we proposed an approach which consists in building several quadratic functions into pieces instead of a single quadratic underestimating the objectives function.

The results are encouraging and support the assertion in some cases such as quadratic is preferable to another. The study algorithmic of the method gave rise to an encouraging report. The led study is the case of one variable. In order, to generalize this study in the case of several variables while preserving the advantages required, an effective combination of the Alienor method with the branch and bound technic was developed successfully.

We have obtained satisfactory results. They show that this method is efficient compared to other current methods.

Keywords: Global optimization, Transportation Problem, Branch & Bound, Underestimators.

ملخص

في هذه الأطروحة ، درسنا موضوع هام جدا في الجانب التطبيقي و هو:

دراسة وتنفيذ خوارزميات التجويد لحل فئات معينة من المسائل التطبيقية مثل مسألة النقل رباعية الأدلة ذات القدرات. في البداية ، تتم دراسة مسألة النقل في اطار البرمجة الخطية مع الأخذ بعين الاعتبار بعض الخصائص باستعمال مفاهيم الاقتصاد الرياضي ، التحليل المحدب وبحوث العمليات. ثم حصلنا على نتائج جيدة باقتراح طريقة فعالة لتجاوز مشكلة الدوران غير المجدي.

كذلك قمنا بدراسة مسائل التجويد غير المحدب ، و لحل هذا النمط من المسائل اقترحنا استعمال عدة دوال تربيعية حدية بدلا من دالة تربيعية حدية واحدة. النتائج المتحصل عليها مشجعة للغاية حيث أن احدى الدوال التربيعية الحدية دون غيرها تساعد في الوصول الى الغاية المرجوة.

قمنا أولا بحل مسألة ذات متغير واحد ثم عممنا ذات الطريقة لحل مسائل ذات عدة متغيرات مع الحفاظ على الأفضليات المكتسبة. و قد تم ذلك بمزج طريقتي Alienor و الفصل و التقييم (Branch & Bound).
و هكذا تحصلنا على نتائج جد مرضية و التي بدورها تبين مدى فعالية الخوارزمية مقارنة مع الخوارزميات الحالية.

الكلمات المفتاحية: التجويد العام ، مسألة النقل ، طريقة الفصل و التقييم ، الدالة الحدية .

Table des matières

Introduction générale

Nous faisons tous de l'optimisation. Dans notre vie quotidienne, nous cherchons à optimiser notre temps de travail, nos espaces de rangement, ou encore le trajet que nous aurons à parcourir pour nous rendre quelque part, etc. Nous recherchons tous une meilleure solution aux problèmes qui jalonnent notre existence. De manière générale, l'optimisation va donc consister à trouver cette meilleure solution.

Comme nous le rappelle l'adage populaire selon lequel : "les mathématiques permettent de mettre le monde en équation", il peut être tracé un parallèle entre l'optimisation quotidienne et celle, plus technique que l'on retrouve en science. En mathématiques, la meilleure solution se recherche au sein d'un domaine initial. Cette solution est souvent soumise à des contraintes qui correspondent à des obligations ou des souhaits à respecter. Le critère permettant de distinguer qu'une solution est la meilleure s'appelle la fonction objectif. L'optimisation mathématique va consister à rechercher dans le domaine initial une solution qui maximise ou minimise une fonction objectif tout en respectant des contraintes. Pour un domaine continu, on distingue classiquement deux types d'optimisation :

• L'optimisation locale recherche une solution qui est la meilleure localement, c'est-à-dire que dans son voisinage aucune solution n'est meilleure qu'elle. Cette solution est appelée un optimum local.

• L'optimisation globale recherche quant à elle la meilleure solution du do-

maine en entier, c'est-à-dire que dans tout le domaine il n'existe aucune solution qui lui soit meilleure tout en respectant les contraintes. Cette solution est appelée l'optimum global. Par définition, l'optimum global est aussi une solution locale. En revanche, il est bien plus épineux de trouver l'optimum global, car lorsque l'on pense avoir trouvé cet optimum, sa démonstration se révèle bien souvent particulièrement ardue. L'intérêt de l'optimisation globale par rapport à l'optimisation locale est patent. Elle garantit en effet que l'on ne peut pas avoir une solution meilleure que celle qui a été trouvée. Or, pour une entreprise, cette information a son importance, car la différence entre la solution globale et une solution locale est bien souvent significative. Mais l'intérêt n'est pas que compétitif. Dans de nombreux problèmes, l'optimum global est la seule solution mathématique correspondant à une réalité physique.

Dans notre étude, nous nous sommes concentrés sur la résolution de deux types de problèmes : les problèmes de transport à quatre indices avec capacités et les problèmes dans lesquels la fonction objectif est non convexe. Les algorithmes que nous allons étudier font partie des méthodes locales conçues spécialement pour résoudre une certaine classe de problèmes linéaires, ceux–ci sont des extensions de la méthode des potentiels, Les méthodes déterministes d'optimisation globale considérées permettent de résoudre des problèmes généraux d'optimisation. Par ailleurs, elles permettent de fournir le plus haut degré de certitude concernant l'optimalité globale des solutions trouvées. Il s'agit des algorithmes de Branch and Bound par sous-estimateurs de la fonction objectif . Ils peuvent théoriquement résoudre n'importe quel problème, pour peu que l'on dispose suffisamment de temps ainsi que d'une mémoire assez grande sur l'ordinateur.

Le premier chapitre de notre étude commence par un rappel des notions élémentaires de l'analyse convexe ainsi que des notations et des définitions qui en découlent, afin d'établir clairement les bases sur lesquelles reposent

nos approches.

Le deuxième chapitre est consacré quant à lui à l'amélioration de l'algorithme présenté par Zitouni et al [61] conçu spécialement pour résoudre les problèmes de transports a quatre indices avec capacités. Dans un premier temps, les conditions nécessaires et les conditions suffisantes d'optimalité sont présentées. Une nouvelle méthode [4] est décrite, basée sur le même principe que celui de Zitouni [60] [61] en modifiant le procédé qui traite la dégénérescence, et en réduisant les instructions dans l'algorithme de résolution. Celle-ci permet toujours de réduire le nombre de tests de vecteurs de base, et résoudre des problèmes de taille importante. Puis, une étude numérique de l'algorithme est effectuée. Deux méthodes sont comparées.

Dans le troisième chapitre, nous proposons une méthode explicite de construction de relaxation quadratique de problèmes d'optimisation globale à variables bornées. Cette construction est basée sur les travaux d'Ouanes et al [37], en utilisant les splines quadratiques. Les programmes quadratiques générés ont exactement des solutions optimales explicites. Dans chaque sous-intervalle, la fonction objectif est sous-estimée par plusieurs splines quadratiques fiables pour calculer des bornes inférieures. De nombreuses expériences numériques tirées de la littérature [23] sont effectuées, venant confirmer l'efficacité de cette nouvelle technique d'accélération.

Dans le quatrième chapitre, nous nous intéressons aux problèmes d'optimisation globales multivariables et la généralisation des techniques précédentes utilisées dans l'algorithme Branch and Bound (Séparation & Évaluation) par sous estimateurs de la fonction objectif. L'idée que nous proposons consiste à limiter les directions de recherche et de préserver l'énorme avantage des minorants qui est représenté par les solutions explicites des problèmes de bornes inférieures générés. De cette façon, nous introduisons un couplage entre la méthode Aliénor [35] et la méthode Branch and Bound par sous estimateurs de

la fonction objectif permettant de combiner les avantages des deux méthodes. Afin de valider notre approche, des expériences numériques sont réalisées sur la résolution de problèmes d'optimisation globale multidimensionnel sur un pavé. L'hybridation utilisée permet facilement d'encadrer la valeur du minimum global. L'objectif de cette combinaison est de préserver l'avantage de la solution explicite et de réduire les dimensions des problèmes en les rendant facile à résoudre dans le but de trouver de meilleures bornes. Le chapitre s'achève avec des expériences numériques sur une certaine classe de problèmes tests [19] en donnant une comparaison avec la méthode présentée par Ouanes et al [47].

Chapitre 1

Généralités

1.1 Introduction

Dans ce chapitre nous rappelons brièvement quelques notions importantes qui serviront à traiter notre sujet. Des développements sur les outils mathématiques évoqués peuvent être trouvés dans les monographies A. Keraghel [34], M. Minoux [43].

1.2 Convexité

Dans cette partie, nous rappelons les plus importants résultats d'analyse convexe nécessaires pour le développement de ce travail.

Un sous-ensemble C de $\mathbb{R}^n$ est dit convexe si ;

$$\forall \lambda \in [0,1]\,,\ \forall x, y \in C : (1-\lambda)x + \lambda y \in C. \tag{1.1}$$

Autrement dit, pour tout x, y appartenant à C, le segment $[x, y]$ est contenu dans C.

Soit C un sous-ensemble convexe de $\mathbb{R}^n$et $f : C \longrightarrow \mathbb{R}$ une fonction. On dit que f est **convexe** dans C si :

$$\forall \lambda \in [0,1]\,,\ \forall x, y \in C :\ f\left[(1-\lambda)x + \lambda y\right] \leq (1-\lambda)f(x) + \lambda f(y). \tag{1.2}$$

f est dite strictement convexe si l'inégalité ci-dessus est stricte

$$\forall x \neq y, \forall \lambda \in\,]0,1[\,.$$

Un sous-ensemble C de $\mathbb{R}^n$ est dit affine si

$$\forall \lambda \in \mathbb{R},\ \forall x, y \in C : (1-\lambda)x + \lambda y \in C. \tag{1.3}$$

Une fonction $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ est dite affine sur C si :

$$\forall \lambda \in \mathbb{R},\ \forall x, y \in C :\ f\left[(1-\lambda)x + \lambda y\right] \leq (1-\lambda)f(x) + \lambda f(y). \tag{1.4}$$

un ensemble convexe de la forme

$$P = \{x \in \mathbb{R}^n : Ax = b\ /\ A \in \mathbb{R}^{m\times n},\ b \in \mathbb{R}^m\} \tag{1.5}$$

est appelé polyèdre convexe.

un ensemble convexe de la forme

$$S_n = \left\{ x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i = 1 \right\} \tag{1.6}$$

est appelé n-simplexe.

1.3 Convexité et Optimisation

Un programme mathématique est un problème d'optimisation de la forme ;

$$(PM) \begin{cases} \min f(x) \\ g_i(x) \leq 0, \quad i = 1 : m \\ h_j(x) = 0, \quad j = 1 : p \\ \qquad x \in C \subseteq \mathbb{R}^n \end{cases}, \tag{1.7}$$

où $f; g_i; h_j$ sont des fonctions définies de $\mathbb{R}^n$ dans $\mathbb{R}$.

L'ensemble

$$S = \{x \in C : g_i(x) \leq 0,\ h_j(x) = 0,\ i = 1 : m,\ j = 1 : p\}$$

est appelé ensemble des solutions réalisables.

Une solution réalisable qui minimise la fonction objectif sur C est dite solution optimale globale de (PM). On note par $\underset{C}{global}\min f(x)$ l'ensemble des solutions optimales globales.

Un point $x^* \in C$ est une solution optimale locale de (PM) s'il existe V voisinage de x^* tel que $f(x^*) \leq f(x), \forall x \in V$ et on note $\underset{C}{loc}\min f(x)$ l'ensemble des solutions optimales locales de (PM).

Nous avons toujours $\underset{C}{global}\min f(x) \subseteq \underset{C}{loc}\min f(x)$ et on a l'égalité entre les deux ensembles si (PM) est convexe.

Remarque 1

Le problème d'optimisation (PM) *consiste :*

- Soit à chercher un point optimal (local ou global).

- Soit, si un tel point n'existe pas on cherche une borne inférieure à f*.*

- Soit à établir que f *est non bornée inférieurement sur* C *auquel cas on adopte la convention* $\inf_C f(x) = -\infty$*.*

- Lorsque C *est vide on pose par convention* $\inf_C f(x) = +\infty$*.*

1.3.1 Classification des problèmes d'optimisation

Les problèmes d'optimisation sont classés selon les caractéristiques des fonctions $f; g_i; h_j$ à savoir la convexité et la différentiabilité. À ce propos, (PM) est dit convexe si f et g_i sont convexes et h_j affines. Si ces dernières sont toutes différentiables on dit que (PM) est un programme différentiable. La classe des programmes mathématiques convexes différentiables est le modèle le mieux élaboré, en absence de la convexité ou de la différentiabilité le problème devient difficile à traiter.

1.3.2 Qualification des contraintes

Une contrainte $g_i(x) \leq 0$ est dite active ou saturée en $x^* \in C$ si elle satisfait $g_i(x^*) = 0$ à ce propos, une contrainte d'égalité est saturée par définition. Un point $x^* \in C$ est dit régulier si les gradients des contraintes saturées en x^* sont linéairement indépendants. On dit également que les contraintes sont qualifiées en x^*.

Remarque 2

La résolution complète de (PM) *traite dans l'ordre les points suivants :*

- L'existence (et éventuellement l'unicité) d'une solution optimale.

- La caractérisation de la solution (il s'agit des conditions d'optimalité)

- L'élaboration d'algorithmes pour calculer cette solution.

1.3.3 Existence et Unicité

Théorème 1 *[11]*

Soit $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ une fonction continue et C un sous-ensemble non vide de $\mathbb{R}^n$. Le problème (PM) admet au moins une solution optimale globale $x^* \in C$. si l'une des deux conditions suivantes est satisfaite :

1. C est compact ***(théorème de Weierstrass)***

2. la fonction f est coercive ($\lim_{\|x\|\to+\infty} f(x) = +\infty$) sur C fermé.

Remarque 3

L'unicité d'une solution éventuelle est en général une conséquence de la convexité de C et la stricte convexité de f.

1.3.4 Conditions d'optimalité

Théorème 2 ***(Karush-Kuhn-Tucher)****[46]*

Si x^* est une solution optimale locale de (PM) satisfaisant l'une des conditions de qualification précédentes, alors il existe des multiplicateurs $\lambda \in \mathbb{R}^m_+$ et $\mu \in \mathbb{R}^p$ tels que :

$$\begin{cases} \nabla f(x^*) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x^*) + \sum_{i=1}^{p} \mu_i \nabla h_i(x^*) = 0 \quad \textit{(optimalité)} \\ \\ \lambda_i g_i(x^*) = 0, \quad i = 1:m \quad \textit{(complémentarité)} \\ h_j(x^*) = 0, \qquad j = 1:p \end{cases}$$

Remarque 4

1. *Si la condition de régularité n'est pas satisfaite, les conditions de* ***K-K-T*** *ne s'appliquent pas (l'existence des multiplicateurs dans ce cas n'est pas assurée)*

2. *Si (PM) est convexe, les conditions de* ***K-K-T*** *sont à la fois nécessaires et suffisantes pour que x^* soit un minimum global.*

1.4 Classes de problèmes en programmation mathématique

1.4.1 Programmation Linéaire

La programmation linéaire est l'une des plus importantes techniques d'optimisation utilisées en recherche opérationnelle. Il faut bien sûr éviter de forcer tout modèle à être linéaire. Par contre, un très grand nombre de modèles constituent des extensions de programmes linéaires. Elle peut se définir comme une technique mathématique permettant de résoudre des problèmes de gestion et particulièrement ceux où le gestionnaire doit déterminer, face à différentes possibilités, l'utilisation optimale des ressources de l'entreprise pour atteindre un objectif spécifique comme la maximisation des bénéfices ou la minimisation des coûts.

Un problème de programmation linéaire mis sous la forme standard définie comme suit ;

$$(PL)\begin{cases} \min c^t x \\ Ax = b \\ x \geq 0 \end{cases}, \tag{1.8}$$

où $c \in \mathbb{R}^n,\ b \in \mathbb{R}^m,\ A$ une matrice réelle de type (m, n).

Au problème (PL), on associe son problème dual ;

$$(D)\begin{cases} \max b^t y \\ A^t y \leq c \\ y \in \mathbb{R}^m \end{cases}. \tag{1.9}$$

Proposition 1 *[11]*

Un programme linéaire réalisable et borné (objectif borné) possède au moins une solution optimale située sur la frontière du domaine réalisable.

Définitions 1

1. *Un point x est un sommet de S si et seulement s'il est une solution réalisable.*

2. *Une base de A est une sous-matrice B formée de m colonne linéairement indépendantes de la matrice A.*

3. *La solution de la base associée à B est le point $x = (x_B, x_N)$ de $\mathbb{R}^n$ tel que $x_B = B^{-1}b$, $x_N = 0$.*

4. *Une solution de base qui vérifie $x_B \geq 0$ est dite solution de base réalisable.*

5. *Un sommet est dit non dégénéré si $x_B > 0$. Il est dégénéré dans le cas contraire (au moins une composante de x_B est nulle).*

Proposition 2 *[11]*

Si un programme linéaire possède une solution optimale alors au moins un sommet du domaine réalisable est une solution optimale.

1.4.2 Programmation quadratique

La programmation quadratique est connue par ses applications multiples dans plusieurs domaines. Souvent, on fait intervenir des programmes quadratiques comme procédures intermédiaires pour des programmes non linéaires, c'est le cas entre autres des méthodes de programmation quadratiques successives (SQP).

Sans perte de généralité, on peut présenter un programme quadratique sous la forme suivante ;

$$(PQ) \begin{cases} \min \left[f(x) = \dfrac{1}{2} x^t Q x + c^t x \right] \\ Ax = b \\ x \geq 0 \end{cases}, \tag{1.10}$$

où Q est une matrice carré symétrique d'ordre n, $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ et $A \in \mathbb{R}^{m \times n}$ de rang plein ($rgA = m < n$). Rappelons que l'ensemble des contraintes est un polyèdre convexe et fermé, la fonction objectif est infiniment différentiable.

Remarque 5

Si la matrice Q est semi-définie positive, le problème (PQ) est convexe, et on montre, dans ce cas que la solution si elle existe, est unique.

1.4.3 Programmation non linéaire

La programmation non linéaire est la recherche de l'optimum d'une fonction non linéaire sur un sous-ensemble convexe d'un espace donné.

On présente les problèmes non linéaires systématiquement sous la forme ;

$$(PNL) \begin{cases} \min f(x) \\ g(x) = 0 \\ h(x) \leq 0 \\ x \in \mathbb{R}^n \end{cases}, \tag{1.11}$$

où f est une fonction de $\mathbb{R}^n$ dans $\mathbb{R}$ et les fonctions des contraintes g et h sont définies de $\mathbb{R}^n$ dans $\mathbb{R}^m$ et $\mathbb{R}^p$ respectivement.

Si une des fonctions f, g ou h est non convexe, alors on est dans le cadre d'optimisation globale.

1.5 Méthode d'optimisation globale

De nos jours, afin de résoudre des problèmes d'optimisation globale avec contraintes, de nombreuses stratégies algorithmiques s'avèrent disponibles. Pour guider le choix de la meilleure stratégie à utiliser, il est nécessaire de regarder :

• La taille du problème.

• Les propriétés de la fonction objectif et des contraintes (continuité, différentiabilité, linéarité, convexité,...).

• Ainsi que le temps disponible pour résoudre le problème. Voici une liste non exhaustive des différentes approches :

Les algorithmes évolutionnaires

les algorithmes les plus connus de cette classe de méthode sont : les algorithmes génétiques ou encore les colonies de fourmis, etc. Ils consistent à faire évoluer une population de solutions en effectuant des croisements entre elles ainsi que des mutations, et ce, tout en ne gardant que les meilleurs éléments d'une génération à l'autre. Malheureusement (ou heureusement), ce principe d'évolution est beaucoup plus efficace dans la nature qu'en mathématiques. Car, pour que ces techniques soient performantes, il est nécessaire d'avoir une bonne connaissance du problème à résoudre, afin d'ajuster les paramètres, les règles de croisements ainsi que les règles de mutations.

Les méthodes métaheuristiques

Tout comme dans les méthodes multistart, ces techniques sont basées sur un algorithme d'optimisation locale, mais dans cette stratégie, l'algorithme utilisé est combiné avec une stratégie pour explorer plus largement le domaine de recherche. En d'autres termes, l'exploration du domaine initial se réalise de façon moins aléatoire, en privilégiant ou interdisant certaines zones. Les deux métaheuristiques les plus connues sont la recherche Tabou et la recherche à voisinage variable (VNS).

Les méthodes déterministes globales

Dans ce type de méthode, l'aléatoire n'intervient pas, c'est-à-dire que pour résoudre un problème, l'algorithme se comportera toujours de la même façon et donnera toujours la même réponse. Ces algorithmes peuvent se classer en fonction du type de problème pouvant être résolu : les programmes linéaires, les problèmes convexes, quadratiques, polynomiaux ou plus généraux. Ces techniques ont généralement l'avantage de ne pas nécessiter de point de départ. Mais avant tout elles fournissent une réponse déterminante sur la qualité des solutions trouvées : l'optimum est-il local ou global ? Quel est le degré de certitude ? etc. Cette précision a une importance significative, car il est souvent beaucoup moins coûteux de trouver une solution que de prouver qu'il s'agit bien de l'optimum global.

Chapitre 2

Problème du transport à quatre indices avec capacités

2.1 Introduction

Le premier problème du transport a été élaboré en 1941 par Hitchcock. La première méthode de résolution est celle des potentiels présentée en 1949 par Kantorovich et Gavourin. Ensuite, G.B. Danzig propose une autre méthode de résolution pour le problème du transport classique, basée sur la méthode du simplex. En 1958, Gleyzal présente une méthode en utilisant l'algorithme du simplex dual et en 1963, Kuhn propose une méthode pour résoudre le problème d'affectation, un cas particulier du problème du transport, en développant l'idée d'un mathématicien hongrois en 1931. Bien que la méthode des potentiels soit proposée au milieu du $20^{ème}$ siècle jusqu'à maintenant, elle reste encore la plus utilisée dans la recherche et l'enseignement Ninh, [45], Zitouni,[61] et Gourgand et al, [26] [27]. On utilise un rectangle : une arrête représente les offres et l'autre représente les demandes. Chaque case représente un arc dans laquelle on note le coût et la quantité des marchandises. Après avoir calculé les potentiels, le critère d'optimalité est contrôlé. S'il n'est pas vérifié, on améliore cette solution pour obtenir une meilleure solution.

La taille la plus grande vérifiée est $m = 40,\ n = 40$ (Dubeau et Guèye, [22]).

Pour seulement chercher la « bonne » solution initiale, il y a plusieurs méthodes. En effet, à côté des méthodes classiques (coin du Nord-est, Vogel, ...) deux nouvelles méthodes sont proposées : la méthode DOR Dubeau et Guèye, [22] et une méthode approchée pour le problème Hitchcock Sharma et Prasard,[52]. Suite à ce problème, l'extension de la méthode des potentiels est proposée pour résoudre le problème du transport à deux indices à capacité ($PT2C$). Son extension étant une classe du $PT2C$ avec des bornes sur conditions, concernant la disponibilité aux origines et les exigences des destinations, trouve ses applications dans les réseaux de télécommunication, la production-distribution, le système de fret automatisé où la capacité des ressources est limitée (véhicules, quais, places de stationnement). Dahiya et Verma, [20].

Par l'augmentation du nombre d'indices, le problème du transport à trois indices $PT3$ est élaboré et résolu par une extension de la méthode des potentiels. Parmi les cas particuliers du problème multi-indices, le $PT4$ est un modèle bien adapté pour l'optimisation du système des navettes, qui a suscité l'intérêt de P.X. Ninh. il obtient la condition essentielle suffisante pour le $PT4$ avec la sommation sur 3 indices. Si le $PT4$ a une ou des solutions, ce problème a certainement une solution optimale. En se basant sur ce résultat, ce $PT4$ est résolu par une méthode exacte, une extension de la méthode des potentiels, dans le cas où le problème est non dégénéré et la condition essentielle suffisante est vérifiée Ninh, [44] [45]. Suite à ce succès, une condition de la capacité limitée sur le chemin a été ajoutée, le problème de transport à quatre indices à capacités $PT4C$ Zitouni, [58]. En 2010, basé sur une grande base des données (771×1500), Aaid [2] réalisait une étude numérique comparative entre trois méthodes pour $PT4C$: deux méthodes classiques simplex, points intérieurs [38] et méthode Zitouni [61], proposée sur extension de la méthode Ninh [45]. Basé sur le critère du nombre d'itérations et du temps d'exécution, le résultat obtenu démontre que la méthode Zitouni est la plus favorable Aaid, [2].

2.2 Position du problème et interprétation

Étant donné :

• m origines $A_1, .., A_m$ de disponibilités $\alpha_1, ..., \alpha_m$,

• n destinations $B_1, ..., B_n$ de demandes $\beta_1, ..., \beta_n$,

• p moyens de transport choisis convenablement $S_1, ..., S_p$ de charges réservées $\gamma_1, ..., \gamma_p$,

• q qualités de marchandises prises en même unités $H_1, ..., H_q$ des quantités $\delta_1, ..., \delta_q$.

On note par :

• d_{ijkl} les capacités des chemins de transport,

• c_{ijkl} l'unité du coût de transport d'un x_{ijkl},

• x_{ijkl} quantité de marchandises de qualité H_l transportées de l'origine A_i vers la destination B_j à travers les moyens de transport S_k.

Modèle

Le problème du transport à quatre indices avec capacités que nous désignons par $(PT4C)$ est formulé comme suit ;

$$\text{Minimiser } Z = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{k=1}^{q} c_{ijkl}x_{ijkl} \tag{2.1}$$

sous les contraintes

$$\sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q} x_{ijkl} = \alpha_i \quad \text{pour tout} \quad i = 1, ..., m$$

$$\sum_{i=1}^{m}\sum_{k=1}^{p}\sum_{l=1}^{q} x_{ijkl} = \beta_j \quad \text{pour tout} \quad j = 1, ..., n$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{l=1}^{q} x_{ijkl} = \gamma_k \text{ pour tout} \quad k = 1, ..., p$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} x_{ijkl} = \delta_l \text{ pour tout} \quad l = 1, ..., q$$

$$0 \leq x_{ijkl} \leq d_{ijkl} \quad \text{pour tout} \quad (i, j, k, l)$$

avec $\alpha_i, \beta_j, \gamma_k, \delta_l, d_{ijkl}$ et c_{ijkl} sont donnés tels que pour tout (i, j, k, l),
on ait $\alpha_i > 0$, $\beta_j > 0$, $\gamma_k > 0$, $\delta_l > 0$, $d_{ijkl} > 0$ et $c_{ijkl}0$
et $M = m + n + p + q$ et $N = mnpq$.

Le $PT4C$ est équivalent à un problème de programmation linéaire a varibles bornées :

$$(PL_{vb}) \quad \begin{cases} \min c^t x \\ Ax = b \quad , \\ 0 \leq x \leq d \end{cases} \tag{2.2}$$

où $x = (x_{1111}, ..., x_{mnpq}) = (x_{ijkl}) \in \mathbb{R}^N$, $c = (c_{1111}, ..., c_{mnpq}) = (c_{ijkl}) \in \mathbb{R}^N$
$b = (\alpha_1, ..., \alpha_m, \beta_1, ..., \beta_n, \gamma_1, ..., \gamma_p, \delta_1, ..., \delta_q) \in \mathbb{R}^M$, $d = (d_{1111}, ..., d_{mnpq}) = (d_{ijkl}) \in \mathbb{R}^N$,
A est une $M \times N$ matrice des M premières contraintes du $PT4C$.
avec $i = 1, ..., m$, $j = 1, ..., n, k = 1, ..., p, l = 1, ..., q$.
Le (PL_{vb}) est équivalent à un problème en forme standard doté d'une structure particulière :

$$(\widehat{PLT}) \quad \begin{cases} \min \widehat{c}^t \widehat{x} \\ \widehat{A}\widehat{x} = \widehat{b} \ , \\ \widehat{x} \geq 0 \end{cases} \tag{2.3}$$

où $y = (y_1, ..., y_N) = (y_\tau) \in \mathbb{R}^N$ (y_τ ; variables d'écarts $\tau = 1, ..., N.$), $\widehat{x} = (x, y)^t$,
$\widehat{c} = (c, 0)^t$, 0 est un N-vecteur nul, $\widehat{b} = (b, d)^t$, $\widehat{A}$ est une $(M + N) \times 2N$ matrice correspondant $Ax = b$ et $x_{ijkl} + y_\tau = d_{ijkl}$.

On remarque que : $\widehat{A} = \begin{bmatrix} A & 0 \\ I_N & I_N \end{bmatrix}$, où I_N est la matrice d'identité d'ordre N.

2.2.1 Conditions de réalisabilité

On suppose que la condition de réalisabilité suivante est vérifiée

$$\sum_{i=1}^{m} \alpha_i = \sum_{j=1}^{n} \beta_j = \sum_{k=1}^{p} \gamma_k = \sum_{l=1}^{q} \delta_l = H \tag{2.4}$$

• Une condition nécessaire pour que le problème $(PT4C)$ possède une solution réalisable est que les conditions suivantes

$$\begin{aligned}
\alpha_i &\leq \sum_{j=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q} d_{ijkl} \quad \text{pour tout} \quad i = 1, ..., m, \\
\beta_j &\leq \sum_{i=1}^{m}\sum_{k=1}^{p}\sum_{l=1}^{q} d_{ijkl} \quad \text{pour tout} \quad j = 1, ..., n, \\
\gamma_k &\leq \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{l=1}^{q} d_{ijkl} \text{ pour tout} \quad k = 1, ..., p, \\
\delta_l &\leq \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} d_{ijkl} \text{ pour tout} \quad l = 1, ..., q,
\end{aligned}$$

soient vérifiées.

• Une condition suffisante pour que le problème $PT4C$ possède une solution réalisable est que la condition suivante

$$\frac{\alpha_i \beta_j \gamma_k \delta_l}{d_{ijkl}} \leq H^3 \qquad \text{pour tout } (i, j, k, l)$$

soit vérifiée.

2.2.2 Conditions d'optimalité

Supposons que $PT4C$ est réalisable, alors une solution réalisable x de $PT4C$ est optimale si et seulement s'il existe un vecteur

$$(u_1\ , ...u_m, v_1, ..., v_n, ..., w_1, ..., w_p, ..., t_1, ..., t_q)^t \in \mathbb{R}^N, \text{ tel que :}$$

$$\begin{aligned} u_i + v_j + w_k + t_l &\leq c_{ijkl} \quad \text{si } x_{ijkl} = 0 \\ u_i + v_j + w_k + t_l &= c_{ijkl} \quad \text{si} \quad 0 < x_{ijkl} < d_{ijkl} \\ &\quad \text{et} \\ u_i + v_j + w_k + t_l &\geq c_{ijkl} \quad \text{si} \quad x_{ijkl} = d_{ijkl}. \end{aligned}$$

2.3 Définitions et propriétés

Définition 1

*Une solution réalisable x d'un $PT4C$ est dite **de base** si les colonnes de la matrice A_x obtenue de A en gardant seulement les colonnes correspondant aux variables x_{ijkl} vérifiant :* $0 < x_{ijkl} < d_{ijkl}$ *sont linéairement indépendantes, le 4-uplet (i, j, k, l) associé a une solution réalisable de base est appellé **case intéressante**.*

Définition 2

*Une solution réalisable de base est dite **non dégénérée** si :*

$$rg(A_x) = rg(A)$$

Définition 3

On appelle cycle, tout ensemble $\mathbf{U}$ contenant un nombre $\boldsymbol{\omega} > \mathbf{1}$ des cases (i, j, k, l) dont les vecteurs P_{ijkl} correspondants sont liés, mais toute sous famille de $(\boldsymbol{\omega} - \mathbf{1})$ éléments de cette famille de vecteurs est libre.

Tableau de transport

Les données du problème en question sont présentées sous forme d'un tableau appelé « tableau du $PT4C$ »

d_{1111} .	d_{1211} .	...	d_{mnpq} .	
c_{1111} .	c_{1211} .	...	c_{mnpq} .	
x_{1111} .	x_{1211} .	...	x_{mnpq} .	
1	1	...	0	α_1 .
:	:	...	:	:
0	0	...	1	α_m .
1	0	...	0	β_1 .
0	1	...	0	β_2 .
:	:	...	:	:
0	0	...	1	β_n .
1	1	...	0	γ_1 .
:	:	...	:	:
0	0	...	1	γ_p .
1	1	...	0	δ_1 .
:	:	...	:	:
0	0	...	1	δ_q .

Lemme 1

Le rang de la matrice des contraintes est $(M-3)$.

Preuve. Montrons par récurrence que le rang de la matrice A est égal à $(M-3)$.
1) Supposons que nous avons un problème de transport $PT4C$ avec $m=n=p=q=1$, alors $A=\begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}^t$ et le $rg(A)=4-3=1=M-3$.
2) Supposons que ce lemme est vrai pour un problème $PT4C$ tel que $M=\eta \geq 4$, et prouvons que c'est vrai aussi pour le problème avec $M=\eta+1$, pallier à l'esprit que le dernier élément diagonal est non nul, on le note $\rho \neq 0$ (voir le tableau de transport). Selon l'hypothèse de récurrence, nous pouvons extraire une sous-matrice A_η de taille $(\eta-3)\times(\eta-3)$ tels que $\det(A_\eta)\neq 0$, et permuter la dernière ligne (ou colonne) par le $(\eta-2)^{eme}$ ligne (ou colonne), on obtient la sous-matrice A_η augmenté par la nouvelle ligne et la nouvelle colonne, et on aura $\det(A_{(\eta+1)})=\rho\times\det(A_\eta)\neq 0$. Donc, nous pouvons extraire une sous-matrice de taille $(\eta-2)\times(\eta-2)$ dont le déterminant est non nul, alors $rg(A)\geq \eta-2=M-3$. D'autre part, de la condition de réalisabilité (2.4), nous avons $rg(A)\leq M-3$. C'est-à-dire $rg(A)=M-3$ et $rg(\widehat{A})=M+N-3$. ■

2.3.1 Méthodes de résolution d'un PT4C

2.3.2 Méthode du simplexe

Elle a été développée à la fin des années 40 par G.Dantzig. Elle évolue sur la frontière du domaine réalisable de sommet en sommet adjacent, en réduisant la valeur de l'objectif jusqu'à l'optimum. Un critère d'optimalité simple permet de reconnaitre le sommet optimale. Le nombre de sommet etant fini, l'algorithme ainsi défini converge en un nombre fini d'itérations sous l'hypothèse que tous les sommets visités sont non dégénérés.

Dans le cas dégénéré l'algorithme risque de cycler, cependant il existe des techniques convenables pour éviter ce phénomène. En général, la méthode du simplexe possède un comportement numérique très satisfaisant confirmé par ses applications multiples dans la résolution d'une large classe de problèmes pratiques. En théorie, la méthode

n'a pas autant de succès, elle est plutôt jugée inefficace par sa complexité arithmétique exponentielle de l'ordre de $O(2^n)$ opérations.

Soit le problème de programmation linéaire suivant :

$$(PL) \quad \begin{cases} \min c^t x \\ Ax = b \\ x \geq 0 \end{cases}, \tag{2.5}$$

où : c, x de $\mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$. supposée de rang plein $(rg(A) = m < n)$.

On note : $P = \{x; Ax = b, x \geq 0\}$

l'idée de l'algorithme :

Soit x_0 une solution de base admissible.

Pour $k = 0, ...$ faire trouver x_{k+1} solution de base admissible voisine telle que

$c^t x_{k+1} < c^t x_k$ jusqu'à ce qu'aucune solution de base admissible voisine n'améliore l'objectif. On trouve alors un minimum local, en programmation linéaire

minimum local=minimum globale.

Soit $x \in P$. On va se déplacer le long d'une direction $d \in \mathbb{R}^n$ qui doit nous maintenir dans P.

Définition 4

Soit x un élément d'un polyèdre P. Un vecteur $d \in \mathbb{R}^n$ est appelé direction admissible en x s'il existe un scalaire positif θ tel que $(x + \theta d) \in P$.

Soit x une solution de base admissible, soit B la matrice de base associée, alors on peut toujours mettre A sous la forme :

$A = [B \ \ N]$, où N est la sous matrice de A d'ordre $m \times (n - m)$ formée par les colonnes qui ne figurent pas dans la base.

De même on peut partitionner x en $\begin{bmatrix} x_B \\ x_N \end{bmatrix}$, c en $\begin{bmatrix} c_B \\ c_N \end{bmatrix}$, et d en $\begin{bmatrix} d_B \\ d_N \end{bmatrix}$.

Soient $B(1),\ .....,B(m)$ les indices des variables de base, alors $B = [A_{B(1)}, ..., A_{B(m)}]$;

($A_{B(i)}$ étant la colonne d'ordre $B(i)$ de la matrice A, $i = 1, ..., m$).

Le système $Ax = b$, s'écrit ainsi : $Bx_B + Nx_N = b$.

La solution x de $Ax = b$, est alors obtenue en faisant $x_N = 0$, elle est déterminée de manière unique par :

$$x_B = B^{-1}b = (x_{B(1)}, ..., x_{B(m)}).$$

Comment déterminer $(x + \theta d)$?

Choisir une variable x_j hors base (qui vaut 0), et augmenter sa valeur jusqu'à θ, tout en gardant les autres variables hors base à zéro.

Donc :

$d_j = 1$

$d_i = 0, i \neq j, i$ indice hors base, alors $d_N = e_j$; ($e_j \in R^{(n-m)}$ le vecteur canonique), et par conséquence $d = \begin{bmatrix} d_B \\ e_j \end{bmatrix}$.

Il faut rester admissible :

$$\begin{aligned} A(x + \theta d) &= b \\ Ax + \theta Ad &= b \end{aligned}$$

- x est admissible, et donc $Ax = b$.
- Pour que $(x + \theta d)$ soit admissible, il faut que $Ad = 0$.

$$\begin{aligned} Ad &= 0 \\ [B \quad N] \begin{bmatrix} d_B \\ e_j \end{bmatrix} &= 0 \\ Bd_B + Ne_j &= 0 \\ Bd_B + A_j &= 0\,; (A_j \text{ étant la colonne d'ordre } j \text{ de la matrice } A) \\ Bd_B + A_j &= 0. \end{aligned}$$

Nous obtenons : $d_B = -B^{-1}A_j$.

La direction d ainsi obtenue est appelée $j^{ième}$ direction de base, elle garantit que les contraintes d'égalité seront vérifiées lorsque l'on s'éloigne de x le long de d.

Qu'en est-il des contraintes de non négativité ?

Variables hors-base :

- x_j était nulle et devient positive
- $x_i, i \neq j$, restent à zéro

Variables de base :

Si x est une solution de base admissible non dégénérée, alors $x_B > 0$. Lorsque θ est suffisamment petit $(x_B + \theta d_B) \geq 0$.

Si x est une solution de base admissible dégénérée, alors d n'est pas toujours une direction admissible. C'est le cas lorsqu'une variable de base $x_i = 0$ et que la composante correspondante d_i de la direction est négative.

Si x est une solution de base admissible non dégénérée, la $j^{ième}$ direction de base en x est admissible, pour tout j indice de base.

Conditions d'optimalité

Définition 5

Soit x une solution de base, soit B la matrice de base associée, et c_B le vecteur de coût pour les variables de bases. Pour chaque j, le coût réduit est défini par $\bar{c}_j = c_j - c_B^t B^{-1} A_j$, $j = 1, ..., n$.

Que vaut le coût réduit pour les variables de base ?

Soit $B(i)$ indice d'une variable de base, le coût réduit est $c_{B(i)} - c_B^t B^{-1} A_{B(i)}$

- $B = \left[A_{B(1)}, ..., A_{B(m)}\right]$, $B^{-1}\left[A_{B(1)}, ..., A_{B(m)}\right] = I_m$,
  (I_m la matrice d'identité d'ordre m)
- $B^{-1}A_{B(i)} = e_i$; (e_i est la $i^{ième}$colonne de I_m).
- $\bar{c}_{B(i)} = c_{B(i)} - c_{B(i)}^t B^{-1} A_{B(i)} = c_{B(i)} - c_{B(i)}^t e_i$
  $= c_{B(i)} - c_{B(i)}^t = 0$, le coût réduit des variables de base est nul.

Théorème 3

Considérons une solution de base admissible x, associée à une matrice de base B. Soit $\bar{c}$ le vecteur de coûts réduits correspondant.

1. Si $\bar{c} \geq 0$, alors x est optimal

2. Si x est optimal et non dégénérée, alors $\bar{c} \geq 0$.

Une itération de la méthode du simplexe :

1. Soit une base $B = [A_{B(1)}, \ldots, A_{B(m)}]$ et x une solution de base admissible associée à B.

2. Calculer les coûts réduits pour chaque indice j hors base : $\bar{c}_j = c_j - c_B^t B^{-1} A_j$.

S'ils sont tous non négatifs, la solution courante est optimale. **STOP.**

3. Si non choisir j tel que $\bar{c}_j < 0$, et calculer $d_B = -B^{-1}A_j$.

Si aucune composante de d_B n'est négative, alors le coût optimal est infini. **STOP.**

4 . Calculer $\theta^* = \min_{\{i=1,\ldots,m; d_{B(i)}<0\}} \frac{-x_{B(i)}}{d_{B(i)}} = \frac{-x_{B(k)}}{d_{B(k)}}$

5 .Former une nouvelle base en remplaçant $A_{B(k)}$ par A_j.

Si y est la nouvelle solution de base admissible, les valeurs des nouvelles variables de base sont :

- $y_j = \theta^*$
- $y_{B(i)} = x_{B(i)} + \theta^* d_{B(i)}, i \neq k.$

Remarque 6

Etape 3 : Choisir j tel que $\bar{c}_j < 0$, l'algorithme ne spécifie pas quelle variable choisir pour rentrer dans la base, plusieurs règles existent.

Retenons la règle de Bland : Parmi les j tels que $\bar{c}_j < 0$, choisir l'indice le plus petit

Détermination d'une solution initiale de base

Pour initialiser l'algorithme du simplexe, il faut disposer d'une solution de base réalisable initiale, or une telle solution de base initiale n'est pas nécessairement disponible. De plus, en général, nous ne savons même pas si le domaine réalisable n'est pas vide.

Soit à résoudre le problème (2.5). Introduire les variables artificielles $t_1,\ldots,t_m$, et appliquer la méthode du simplexe au problème auxiliaire

$$(PA) \quad \begin{cases} \min \sum_{i=1}^{m} t_i = w \\ Ax + t = b \\ x, t \geq 0 \end{cases}, \tag{2.6}$$

si la valeur optimale w est strictement positive (i.e.,$w > 0$), alors le domaine réalisable du problème original est vide (i.e., le problème original n'est pas réalisable), si la valeur optimale w est nulle (i.e., $w = 0$), alors le domaine réalisable du problème original n'est pas vide (i.e., le problème original est réalisable). Dans ce cas, les valeurs que prennent les variables x_i constituent une solution pour le problème original où toutes les variables artificielles t_i sont égales à 0.

Maintenir en permanence : $B^{-1}[A,\, b] = [B^{-1}A_1 \ldots B^{-1}A_n,\, B^{-1}b]$, cette matrice s'appelle le tableau du simplexe.

- La dernière colonne contient les valeurs des variables en base.
- Les autres colonnes permettent de calculer $\bar{c}_j$ et d_B.

Définition 6

Lors d'une itération du simplexe : Si j est l'indice de la variable qui entre en base, la colonne $B^{-1}A_j$ est appelée colonne du pivot.
Si la variable de base $B(k)$ sort de la base, la ligne k du tableau est appelée ligne du pivot. L'élément qui se trouve sur la ligne du pivot et la colonne du pivot est appelé ***le pivot****.*

On augmente le tableau avec une ligne relative aux coûts : $[c^t - c_B^t B^{-1}A,\, -c_B^t B^{-1}b]$, et par conséquence on a ;

- $-c_B^t B^{-1}b = -c_B^t x_B = -c^t x =$ −fonction objectif,
- $c^t - c_B^t B^{-1}A$ coûts réduits.

Le tableau complet sera donc

$B^{-1}A$	$B^{-1}b$
$c^t - c_B^t B^{-1}A$	$-c_B^t B^{-1}b$

,

en posant : $B^{-1}A = (\bar{a}_{ij})$,

- $B^{-1}b = (\bar{b}_i)$ sont les solutions
- $\bar{c}_j = c^t - c_B^t B^{-1}A$ sont les coûts réduits
- $-c_B^t B^{-1}b$ est la valeur de l'objectif, avec $i = 1, ..., m$ et $j = 1, ..., n$.

Algorithme 1

1. *Soit une matrice de base B, une solution de base admissible x et le tableau du simplexe associés.*

2. *Examiner les coûts réduits dans la dernière ligne du tableau.*

S**'ils sont tous **non négatifs**, la solution de base admissible courante est optimale.* ***STOP.

Sinon, *choisir j tel que $\bar{c}_j < 0$.qui devient $\bar{c}_s$; (s indice de la colonne du pivot)*

3. Si *tous les $\bar{a}_{is} < 0$ le coût optimal est $-.\infty$* ***STOP.***

4. *Pour chaque i tel que $(\bar{a}_{is} > 0)$, calculer $\left(\frac{\bar{b}_i}{\bar{a}_{is}}\right)$.*

Soit r l'indice de la ligne correspondant au rapport le plus petit $\left(\frac{\bar{b}_r}{\bar{a}_{rs}}\right)$.

(r l'indice de la ligne du pivot), la variable x_r sort de la base. la variable x_s rentre en base.

5. *Pivoter autour du pivot $(\bar{a}_{rs})$.* ***Aller à l'étape (2).***

Fin Algorithme.

2.3.3 Méthode de points intérieurs de Karmarkar

Malgré sa complexité exponentielle, l'algorithme du Simplexe est resté longtemps l'algorithme de référence pour la programmation linéaire. La méthode est demeurée sans compétiteurs sérieux jusqu'en 1984 où un nouvel algorithme polynomial, l'algorithme des méthodes projectives a été proposé par Narendra Karmarkar aux laboratoires de AT&T Bell aux Etas Unis. Mais la mise en place et les transformations nécessaires pour résoudre un problème standard de programmation linéaire par la méthode de Karmarkar sont demeurées secrètes jusqu'en 1985. Cet algorithme a été essentiellement différent de la méthode du simplexe par le fait que, dans cette dernière, on se déplace en suivant la frontière du domaine réalisable, alors que pour la méthode de Karmarkar, on progresse tout en restant strictement à l'intérieur du domaine réalisable, d'où la connotation "intérieur", et c'était grâce à lui qu'une recherche intense dans les méthodes du points intérieurs s'est déclenchée et a donné comme résultat une grande variété d'algorithmes de ce type. L'algorithme de Karmarkar avait un grand intérêt théorique puisqu'il est polynomial ; en effet, il exige un $O(n^3)$ opérations arithmétiques, d'où une complexité totale de $O(n^4L)$ opérations. Karmarkar a même été capable de la réduire à $O(n^{3.5}L)$ en utilisant un processus de mise à jour partiel.

Ces méthodes projectives n'ont pas été présentées seulement par Karmarkar mais aussi par Todd et Burell, Anstreicher, Gay et Ye et Kojima [58]. Ces chercheurs ont suggéré des modifications de l'algorithme original de Karmarkar afin d'aboutir à un algorithme très efficace et plus utile dans la pratique. Ces méthodes ont été mise en ouvre hors des laboratoires Bell par Adler et al, Gill et al, Monma et Morton et Mcshane et al [59], ces différentes mise en ouvre ont donné de bons résultats prouvant leur supériorité par rapport aux autres méthodes (simplexe, ellipsoïde ...).

La méthode de Karmarkar est destinée à résoudre les problèmes de la forme :

$$(PK)\begin{cases} \min c^t x = 0 \\ Ax = 0 \\ x \in S_n = \{x \in \mathbb{R}^n, e_n^t x = 1, x \geq 0\} \end{cases}, \tag{2.7}$$

où $c \in \mathbb{R}^n$ (vecteur coût), A une matrice réelle de type $(m \times n)$ de rang plein ($rg(A) = m < n$) et $e_n = (1, 1, ..., 1)^t \in \mathbb{R}^n$.

Ayant comme solution strictement réalisable de (PK) le centre du simplexe S_n :

$$x^0 = \frac{e_n}{n}$$

Remarque 7

Si la valeur optimale est connue à priori mais non nulle l'égalité $e_{n+1}^t x = 1$ *permet de se ramener à un objectif nul. En effet, soit* x^* *une solution optimale du problème et* z^* *la valeur optimale de l'objectif. Alors :*

$$c^t x^* = z^* = z^* e_{n+1}^t x^* \Longrightarrow (c - z^* e_{n+1})^t x = (c^{'})^t x = 0.$$

On minimise alors l'objectif $(c^{'})^t x$ *au lieu de* $c^t x$, *où* $c_i' = c_i - z^* e_{n+1}, i = 1, ..., n$.
Pour le système de contraintes : $Ax = b, b \neq 0$, *on se ramène facilement à un système homogène, il suffit d'écrire :*

$$Ax = be_{n+1}^t x \Longrightarrow (A - be_{n+1}^t)x = 0.$$

Description de l'algorithme de base

L'algorithme de Karmarkar part d'un point x^0 et construit une suite de points intérieurs x^k qui converge vers une solution optimale x^* en un temps polynômial. A chaque

itération, on ramène l'itéré x^k au centre du simplexe S_n par la transformation projective T_k définie comme suit :

$$T_k : x \in S_n \to T_k(x) = y \in S_n$$

avec

$$T_k(x) = \frac{D_k^{-1}x}{e_n^t D_k^{-1}x} = y \quad \text{et} \quad T_k^{-1}(y) = \frac{D_k y}{e_n^t D_k y}$$

où $D_k = diag\{x^k\}$ et on teste à chaque fois la mesure d'optimalité $\left(c^t x^k \leq \varepsilon\right)$ où ε est la précision recherchée.

Algorithme de Karmarkar

Initialisation : $\varepsilon > 0$ est une précision donnée, $x^0 = \dfrac{e_n}{n}$, $k = 0$

Tant que : $c^t x^k > \varepsilon$ **faire**

Construire $D_k = diag\left\{x^k\right\}$, $A_k = AD_k$, $B_k = \begin{bmatrix} A_k \\ e_n^t \end{bmatrix}$

Calculer $p_k = (I - B_k^t(B_k B_k^t)^{-1}B_k)D_k c$

$$= (I - A_k^t(A_k A_k^t)^{-1}A_k - \frac{1}{n}e_n e_n^t)D_k c,$$

$$d_k = \frac{p_k}{\|p_k\|}$$

Calculer $y^k = \dfrac{e_n}{n} - \alpha r d_k$, $r = \dfrac{1}{\sqrt{n(n-1)}}$, $0 < \alpha < 1$

Prendre $x^{k+1} = T_k^{-1}(y^k) = \dfrac{D_k y^k}{e_n^t D_k y^k}$, $k = k + 1$

Fin tant que.

Fin algorithme.

Fonction potentiel et convergence

Pour établir la convergence de l'algorithme, Karmarkar introduit la fonction dite fonction potentiel définie par :

$$P(x) = n\ln(c^t x) - \sum_{i=1}^{n} \ln(x_i)$$

Lemme 2 *[33]*

Soit x^k le $k^{ième}$ itéré de l'algorithme, alors :

$$\frac{c^t x^k}{c^t x^0} \leq (\exp(P(x^k) - P(x^0)))^{\frac{1}{n}}$$

Théorème 4 *[33]*

Si $0 < \alpha \leq 0.25$, alors en partant de $x^0 = \frac{1}{n} e_n$, après $O(nq+n\ln n)$ itérations l'algorithme trouve un point réalisable x tel que :

$$\begin{cases} c^t x = 0 \\ ou \\ \dfrac{c^t x}{c^t x^0} \leq 2^{-q} \end{cases}$$

où q est une précision fixée.

Généralisation de l'algorithme de base

Soit le problème linéaire sous forme standard :

$$(P) \begin{cases} \min c^t x = z^* \\ Ax = b \\ x \geq 0 \end{cases} \tag{2.8}$$

A est une matrice de type (m, n), $c \in \mathbb{R}^n$ est le vecteur coût et $b \in \mathbb{R}^m$ constant.

On suppose que :

H1) Les lignes de la matrice A sont linéairement indépendantes.

H2) On dispose d'un point x^0 strictement réalisable ($Ax^0 = b, x^0 > 0$).

H3) La valeur optimale z^* de l'objectif est connue au départ.

L'hypothèse **1** est classique. L'hypothèse **2** n'est pas du tout restrictive, car on peut toujours obtenir une solution strictement réalisable ou prouver que le problème n'est pas réalisable moyennant une méthode simple de variable artificielle [32].

La transformation projective notée T_k est une fonction définie comme suit :

$$T_k : \mathbb{R}^n_+ \longrightarrow S_{n+1} \text{ telle que : } T_k(x) = y = \begin{cases} y_i = \dfrac{\dfrac{x_i}{x_i^k}}{1 + \sum\limits_{i=1}^{n} \dfrac{x_i}{x_i^k}}, i = 1 : n \\ y_{n+1} = 1 - \sum\limits_{i=1}^{n} y_i \end{cases}$$

On a :

$$y_i = \frac{x_i}{x_i^k} y_{n+1} \ , \ i = 1 : n$$

et on a :

$$x = T_k^{-1}(y) = \frac{D_k y[n]}{y_{n+1}}$$

On applique la transformation T_k au programme (2.8), on obtient :

$$\begin{cases} \min \dfrac{c^t D_k y[n]}{y_{n+1}} \\ \dfrac{A D_k y[n]}{y_{n+1}} = b \\ \sum\limits_{i=1}^{n} y_i = 1 \\ y[n] \geq 0, y_{n+1} > 0 \end{cases} \tag{2.9}$$

Or, si la valeur optimale z^* est connue, on peut écrire :

$$\begin{cases} \min \widehat{c}^t y \\ \widehat{A} y = 0 \\ e_{n+1}^t y = 1 \\ y \geq 0 \end{cases} \tag{2.10}$$

où : $\begin{cases} \widehat{c}_i = c_i x_i^k, \quad i = 1, ..., n \\ \widehat{c}_{n+1} = -z^* \end{cases}$, $y = \begin{pmatrix} y[n] \\ y_{n+1} \end{pmatrix}$ et $\widehat{A} = [AD_k \qquad -b]$.

Notons que toute solution réalisable de (2.8) est transformée par T_k en une solution réalisable de (2.9) et réciproquement, toute solution réalisable y de (2.9) avec $y_{n+1} > 0$ est transformée par T_k^{-1} en une solution réalisable x de (2.8).

Résultats de convergence

Pour contrôler la convergence de l'algorithme, on introduit la fonction potentiel associée au problème (P) définie par :

$$P(x) = (n+1)\ln(c^t x - z^*) - \sum_{i=1}^{n} \ln(x_i).$$

On a :

$$P(x^k) - P(x^0) = \left[(n+1)\ln(c^t x^k - z^*) - \sum_{i=1}^{n} \ln(x_i^k)\right] - \left[(n+1)\ln(c^t x^0 - z^*) - \sum_{i=1}^{n} \ln(x_i^0)\right]$$

$$= (n+1)\ln\left[\frac{c^t x^k - z^*}{c^t x^0 - z^*}\right] - \left[\sum_{i=1}^{n} \ln(x_i^k) - \sum_{i=1}^{n} \ln(x_i^0)\right]$$

$$\frac{P(x^k) - P(x^0)}{n+1} = \ln\left[\frac{c^t x^k - z^*}{c^t x^0 - z^*}\right] - \left[\frac{\sum_{i=1}^{n} \ln(x_i^k) - \sum_{i=1}^{n} \ln(x_i^0)}{n+1}\right]$$

$$\frac{c^t x^k - z^*}{c^t x^0 - z^*} = \upsilon(x^k) \exp\left(\frac{P(x^k) - P(x^0)}{n+1}\right)$$

$$\text{avec : } \upsilon(x^k) = \exp\left(\frac{\sum_{i=1}^{n} \ln(x_i^k) - \sum_{i=1}^{n} \ln(x_i^0)}{n+1}\right)$$

Théorème 5 *[32]*

A chaque itération de l'algorithme, la fonction potentiel se réduit d'une valeur constante δ telle que :

$$P(x^{k+1}) \leq P(x^k) - \delta \quad où \quad \delta = \alpha - \frac{\alpha^2}{2(1-\alpha)^2}.$$

Théorème 6 *[32]*

Sous les hypothèses :

1. Il existe une solution réalisable x^0 telle que : $x^0 \geq 2^{-L} e_n$.

2. La solution optimale x^* vérifie $x^* \leq 2^L e_n$.

De plus pour tout x réalisable on a : $-2^{3L} \leq z^* \leq c^t x \leq 2^{3L}$.

L'algorithme trouve une solution optimale après $O(nL)$ itérations. L est le nombre total des bits nécessaires

Preuve.

Puisque $\dfrac{c^t x^k - z^*}{c^t x^0 - z^*} = \upsilon(x^k) \exp\left(\dfrac{P(x^k) - P(x^0)}{n+1}\right)$

avec $\upsilon(x^k) = \exp\left(\dfrac{\sum_{i=1}^{n} \ln(x_i^k) - \sum_{i=1}^{n} \ln(x_i^0)}{n+1}\right)$

Par les hypothèses (1) et (2) on a : $\upsilon(x^k) \leq 2^{2L}$ dans la région de réalisabilité, alors :

$$f(x^k) - z^* \leq 2^{2L}(f(x^0) - z^*) \exp\left(\frac{P(x^k) - P(x^0)}{n+1}\right),$$

$$f(x^k) - z^* \leq 2^{3L} 2^{2L} \exp\left(\frac{-k\delta}{n+1}\right) \leq 2^{-4L}$$

(Généralement, on dit que (P) est réalisable si on trouve $x^k \in X$, sachant que $f(x^k) - z^* \leq 2^{-4L}$)

On a $k \geq h\,(n+1)\,L$ avec $h \in \mathbb{R}^*_{+.}$ ■

Nous signalons qu'il existe différentes méthodes pour transformer un P.L.G à un P.L.K et nous citons entre autres dans [33]

L'approche de Ye-Lustig

Soit le programme linéaire sous forme standard :

$$(PL) \quad \begin{cases} \min c^t x = z^* \\ Ax = b \\ x \geq 0 \end{cases}, \tag{2.11}$$

où : c, x de $\mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$.

avec l'hypotèse suivante :

H1) La matrice des contraintes A est de rang plein $(rg(A) = m < n)$

Dans cette méthode, à chaque itération k, on utilise une transformation projective qui ramène la région admissible polyédrique $\{Ax = b,\, x \geq 0\}$ à un simplexe

$$S_{n+1} = \left\{ x \in \mathbb{R}^{n+1}_+,\ \sum_{i=1}^{n+1} x_i = 1 \right\}.$$

Cette transformation est définie par :

$$T_a : \mathbb{R}^n_+ \longrightarrow S_{n+1}$$

$$T_a\left(x^k\right) = y = \begin{cases} y_i = \dfrac{\dfrac{x_i^k}{a_i}}{1 + \displaystyle\sum_{i=1}^{n} \dfrac{x_i^k}{a_i}} \quad , i = 1, ..., n \\ y_{n+1} = 1 - \displaystyle\sum_{i=1}^{n} y_i \quad , \quad a \in \mathbb{R}_+^n \end{cases} .$$

Le problème (PL) devient alors :

$$P_t \quad \begin{cases} \min(D_k c, -z^*)^t y \\ A_k y = 0 \\ y \in S_{n+1} = \left\{ y \in \mathbb{R}^{n+1},\ y \geq 0,\ \displaystyle\sum_{i=1}^{n+1} y_i = 1 \right\} \end{cases} ,$$

où $A_k = [AD_k \ -b] \in \mathbb{R}^{(n+1)\times m}$; $D_k = diag(x^k)$, matrice diagonale.

Comme le calcul d'une solution optimale d'un programme linéaire, sur une sphère est évident, on introduit à chaque itération la plus grande sphère inscrite dans le simplexe S_{n+1}.

On obtient le sous problème de P_t suivant

$$P_s \quad \begin{cases} \min(D_k c, -z^*)^t y \\ A_k y = 0 \\ \|y - e_{n+1}\|^2 \leq r^2 < 1 \end{cases} ,$$

où $r = \dfrac{1}{\sqrt{n(n+1)}}$, est le rayon de la sphère.

$\dfrac{e_{n+1}}{n+1} = (\dfrac{1}{n+1}, .., \dfrac{1}{n+1})^t \in \mathbb{R}^{n+1}$, est le centre de la sphère.

Comme la valeur de z^* est généralement inconnue, Ye-Lustig approxime z^* à chaque itération par des bornes supérieures $z^k = c^t x^k > z^*$.

Le problème P_s est remplacé par :

$$(P_k) \quad \begin{cases} \min(D_k c, -c^t x^k)^t y \\ A_k y = 0 \\ \|y - e_{n+1}\|^2 \leq r^2 < 1 \end{cases} .$$

En utilisant les conditions d'optimalités de KKT relatives au problème (P_k), la solution optimale est donnée par :

$$y_k^* = \frac{e_{n+1}}{n+1} - \alpha^k r d^k,$$

où α^k est le pas de déplacement ; $0 < \alpha_k < 1$.

$d^k = \dfrac{p^k}{\|p^k\|}$, p^k est la projection du vecteur coût $(D_k c, -c^t x^k)^t$ sur le noyau de la matrice A_k,

$$p^k = (I - A_k^t (A_k A_k^t)^{-1} A_k)(D_k c, -c^t x^k)^t.$$

La vitesse de convergence de cette méthode dépend du calcul de la projection p^k, et aussi du pas de déplacement.

Calcul de la projection

Nous avans

$$p^k = (I - A_k^t (A_k A_k^t)^{-1} A_k)(D_k c, -c^t x^k)^t.$$

Posons

$$u_k = (A_k A_k^t)^{-1} A_k (D_k c, -c^t x^k)^t,$$

donc

$$p^k = (D_k c, -c^t x^k) - A_k^t u_k,$$

le calcul de p^k dépend ainsi de la manière par laquelle, on résout le système linéaire :

$$A_k A_k^t u_k = A_k (D_k c, -c^t x^k)^t,$$

dont la matrice $A_k A_k^t$ est symétrique définie positive.

Algorithme 2

Initialisation

$x^0 > 0$, $k = 0$

Tan que $\dfrac{\left\|d^k\right\|}{|c^t x^0|} > \varepsilon$ **faire**

1.Construire $D_k = diag(x^k)$, $A_k = [AD_k, -b]$, $B_k = \begin{bmatrix} A_k \\ e_n^t \end{bmatrix}$.

2.Calculer la projection p_k

$$p_k = \left[I - B_k^t (B_k B_k^t)^{-1} B_k\right] (D_k c, -c^t x^k)^t$$

3.Normaliser la projection p_k

$$d^k = \frac{p_k}{\|p_k\|},\ r = \frac{1}{\sqrt{n(n+1)}}$$

4.Calculer l'itéré suivant

$$y^k = \frac{e_{n+1}}{n+1} - \alpha^k r d^k,\ \alpha^k \text{ le pas de déplacement}$$

5.Revenir au problème original par T_k^{-1}

$$x^{k+1} = T_k^{-1}(y^k) = \frac{D_k y^k\,[n]}{y_{n+1}^k}$$

Fin tant que

Fin algorithme.

Calcul d'une solution initiale strictement réalisable

Un point strictement réalisable de (PL) est une solution du problème :

$$(PF)\begin{cases} Ax = b \\ x > 0 \end{cases},$$

En utilisant la technique de la variable artificielle, le problème (PF) est équivalent au problème d'optimisation suivant :

$$(P_\lambda)\begin{cases} \min \lambda \\ \quad Ax + \lambda q = b \\ x > 0, \lambda \geq 0 \end{cases},$$

tel que : $q = b - Aa$ et $a \in \mathbb{R}^n_+$

(P_λ) est équivalent au programme linéaire :

$$(\tilde{P}\tilde{L})\quad \begin{cases} \min \tilde{c}^t \tilde{x} = w^* = 0 \\ \tilde{A}\tilde{x} = b \\ \tilde{x} \geq 0 \end{cases},$$

où $\tilde{x} = (x, \lambda)^t$, $\tilde{c} = (0, 0, 0, ..., 1)^t \in \mathbb{R}^{n+1}$, $\tilde{A} = [A,\ \ b - Aa] \in \mathbb{R}^{m\times(n+1)}$.

Etant donné que $(a, 1)$ est une solution strictement réalisable de $(\tilde{P}\tilde{L})$ pour tout $a > 0$. Alors le calcul de la solution optimale de $(\tilde{P}\tilde{L})$ est donné par la phase 2 de l'algorithme de Ye-Lustig. L'algorithme correspondant se présente comme suit :

Algorithme d'initialisation

Initialisation $x^0 = a$, $\lambda^0 = 1$, $\tilde{x} = (x^0, \lambda^0)$ et $k = 0$

1) **Si** $\|Ax^0 - b\| \leq \varepsilon$ **Stop ;** x^k est une solution approximative de (PF).

Sinon aller à l'étape (**2**)

2) **Si** $\lambda^k \leq \varepsilon$ **Stop ;** x^kest une solution approximative de (PF).

Sinon aller à l'étape (**3**).

3) Poser $\quad D_k = diag(\tilde{x}^k), \quad B = [A,\, b - Aa] \quad$ et $r = \dfrac{1}{\sqrt{(n+1)(n+2)}}$

Calculer $\quad p_k = [I - B_k^t(B_k B_k^t)^{-1} B_k]\,(D_k \hat{c}, -\tilde{c}^t \tilde{x}^k)^t$

Tel que $\quad B_k = [BD_k, -b] \qquad$ et $\qquad \tilde{c} = (0, 0, 0, ..., 1)^t \in \mathbb{R}^{n+1}$

Calculer $\quad y^{k+1} = \dfrac{e_{n+2}}{n+2} - \alpha^k r \dfrac{p_k}{\|p_k\|}$, $\tilde{x}^{k+1} = (y_{n+2}^{k+1})^{-1} D_k y^{k+1}[n+1]$.

4) **Faire** $k = k + 1$ et retourner à l'étape (2).

Fin

2.3.4 Méthode $\mathbf{AL}_{PT4Cm}$

Cet algorithme est une amélioration d'une variante de l'algorithme du simplexe, adapté pour le $PT4C$, qu'on le note AL_{PT4Cm}. Il est décrit comme suit :

Pour traiter le problème de dégénérescence on propose le procédé suivant :

Procédé détectif [4]

Soit :

• N_b le nombre de vecteurs P_{ijkl} tel que :

$$0 < x_{ijkl} < d_{ijkl},$$

si $N_b < M - 3$, alors le problème $PT4C$ est dégénéré.

• On considère la matrice D de M lignes et N colonnes construites des N_b premiers vecteurs P_{ijkl} tel que :

$$0 < x_{ijkl} < d_{ijkl}$$

et les vecteurs restants tels que :

$$x_{ijkl} = 0 \text{ ou } x_{ijkl} = d_{ijkl}$$

• On considère la sous-matrice D_B constituée par les N_b premières colonnes de D.

Si $rg(D_B) = M - 3$, alors D_B est une base, autrement, pour détecter les vecteurs de la base nous procédons comme suit :

• On augmente la matrice D_B par un vecteur si le rang de cette dernière augmente, alors ce vecteur est basique sinon il sera rejeté. On répète ce processus jusqu'à ce qu'une base soit déterminée.

$$I_b = \{(i,j,k,l) \text{ tels que } P_{ijkl} \text{ sont indépendants } \} \tag{2.12}$$

• On a les cases non intéressant

$$H_0 = \{(i,j,k,l) \text{ tels que } x_{ijkl} = 0\}$$

$$H_d = \{(i,j,k,l) \text{ tels que } x_{ijkl} = d_{ijkl}\}\,. \tag{2.13}$$

Pour déterminer les cases formant un cycle, on utilise la technique de résolution suivante.

Technique de résolution pour déterminer un cycle (TRC) [4]

• Les vecteurs P_{ijkl} correspondant à un cycle $\mathbf{U}$ vérifient la relation :

$$\sum_{(i,j,k,l)\in \mathbf{U}} \alpha_{\overline{ijkl}} P_{ijkl} = 0.$$

• Le vecteur P_{ijkl} entrant à la base prend comme coefficient

$$\alpha_{\overline{ijkl}} = \begin{cases} 1 & \text{si } \ \overline{i},\overline{j},\overline{k},\overline{l} \in H_0 \\ -1 & \text{si } \ \overline{i},\overline{j},\overline{k},\overline{l} \in H_d \end{cases}. \tag{2.14}$$

• Nous construisons le système linéaire suivant ;

$$BX = b\ , \tag{2.15}$$

où B est une $M \times (M-3)$ matrice constituée des vecteurs P_{ijkl} avec $(i,j,k,l) \in I_b$, i.e.

B est une base, $X = (\alpha_{ijkl})^t \in \mathbb{R}^{M-3}$ et $b \in \mathbb{R}^M$ avec :

$$b = \begin{cases} -P_{\overline{ijkl}} & \text{si } \overline{i}, \overline{j}, \overline{k}, \overline{l} \in H_0 \\ P_{\overline{ijkl}} & \text{si } \overline{i}, \overline{j}, \overline{k}, \overline{l} \in H_d \end{cases} . \tag{2.16}$$

Comme la matrice B possède M lignes et $M-3$ colonnes et le vecteur des inconnues X possède $M-3$ composantes, le système n'est pas compatible, on procède comme suit pour surmonter cette difficulté. On considère la matrice augmentée $[B, b]$, en utilisant l'élimination de Gauss avec seulement permutation de lignes, on obtient une matrice échelonnée $[\widetilde{B}; \widetilde{b}]$ de $M-3$ lignes et $M-2$ colonnes. Maintenant, nous pouvons résoudre le système linéaire $\widetilde{B}X = \widetilde{b}$. Les solutions $\alpha_{ijkl} \neq 0$ du système linéaire déterminent les cases formant le cycle.

Soit :

$$\begin{aligned} F &= \{(i,j,k,l) \text{ tels que } \alpha_{ijkl} = 1\} \\ B &= \{(i,j,k,l) \text{ tels que } \alpha_{ijkl} = -1\}, \end{aligned}$$

l'ensemble F contient des cases recevant et l'ensemble B contient les cases cédantes.

Pour calculer les variables duales on utilise la technique de résolution suivante ;

Technique de résolution pour déterminer les variables duales (TRVD) [4]

On considère le système linéaire suivant ;

$$c_{ijkl} = u_i + v_j + w_k + t_l, \quad \forall (i,j,k,l) \in I_b \tag{2.17}$$

Comme le système contient M inconnues, et on a que $M-3$ équations. Donc, il admet une infinité de solutions.

Nous construisons le système : $AX = b$ telle que pour tout $(\overline{i}, \overline{j}, \overline{k}, \overline{l}) \in I_b$ nous avons :

• Si $I \leq M-3$

$$a_{IJ} = \begin{cases} 1 \text{ si } i = \overline{i} & \text{pour tout} \quad J \leq m \\ 1 \text{ si } j = \overline{j} & \text{pour tout} \quad m < J \leq m+n \\ 1 \text{ si } k = \overline{k} & \text{pour tout} \quad m+n < J \leq m+n+p \\ 1 \text{ si } l = \overline{l} & \text{pour tout} \quad m+n+p < J \leq m+n+p+q \\ 0 & \text{ailleurs} \end{cases} \quad .. \qquad (2.18)$$

• Si $I > M-3$

$$a_{IJ} = \begin{cases} 1 & \text{si} \quad I = M-2 \quad \text{et} \quad J = 1 \\ 1 & \text{si} \quad I = M-1 \quad \text{et} \quad J = m+1 \\ 1 & \text{si} \quad I = M \quad \text{et} \quad J = m+n+1 \\ 0 & \text{ailleurs} \end{cases} .$$

Critère d'arrêt

Lemme 3 *[4]*

On définit la différence suivante :

$$D_{ijkl} = \begin{cases} u_i + v_j + w_k + t_l - c_{ijkl} & \textit{pour tout } (i,j,k,l) \in H_0 \\ c_{ijkl} - (u_i + v_j + w_k + t_l) & \textit{pour tout } (i,j,k,l) \in H_d \end{cases} , \qquad (2.19)$$

et soit l'ensemble $\Pi = \{D_{ijkl}$ *tel que* $D_{ijkl} > 0$, *pour tout* $(i,j,k,l) \in (H_0 \cup H_d)\}$. *Si* $\Pi \neq \emptyset$, *alors la solution actuelle* $x = (x_{ijkl})$ *n'est pas optimale.*

Preuve.

Supposons que $\Pi = \emptyset$, c'est-à-dire, pour tout $(i,j,k,l) \in (H_0 \cup H_d)$, nous avons $D_{ijkl} \leq 0$, ce qui indique que

$$\begin{cases} u_i + v_j + w_k + t_l - c_{ijkl} \leq 0 \quad \text{pour tout } (i,j,k,l) \in H_0 \\ \text{et} \\ c_{ijkl} - (u_i + v_j + w_k + t_l) \leq 0 \quad \text{pour tout } (i,j,k,l) \in H_d \end{cases}, \tag{2.20}$$

c'est-à-dire

$$\begin{cases} u_i + v_j + w_k + t_l - c_{ijkl} \leq 0 \quad \text{si } x_{ijkl} = 0 \\ \text{et} \\ c_{ijkl} - (u_i + v_j + w_k + t_l) \leq 0 \quad \text{si } x_{ijkl} = d_{ijkl} \end{cases}. \tag{2.21}$$

Selon le théorème d'optimalité, la solution x_{ijkl} est optimale.

Alors, si $\Pi \neq \emptyset$, la solution x_{ijkl} n'est pas optimale et peut-être améliorée. ■

Calcul d'une solution réalisable de base [4]

Étant donné un problème $PT4C$ réalisable, au départ toutes les variables sont nulles, c'est-à-dire aucune marchandise ne transite encore. Pour trouver une solution réalisable. Nous ordonnons les cases selon les coûts, et nous affectons une valeur à la variable x_{ijkl} correspondante à la première case.

On actualise nos paramètres, si $\alpha_i \neq 0$, $\beta_j \neq 0$, $\gamma_k \neq 0, \delta_l \neq 0$, il est nécessaire d'affecter à cette case une valeur, sinon on passe à la suivante qui vérifie la condition nécessaire d'affectation. On reprend ce processus jusqu'à ce que toutes les cases concernées soient affectées, et nous obtenons une solution réalisable de base pour notre problème. On résume les techniques précédentes dans l' algorithme suivant :

Algorithme 3 AL_{PT4Cm}

Phase1

Initialisation

- $x_{ijkl} = 0 \ \forall (i,j,k,l)$
- $N_b = 0$
- Ordonner les cases selon les coûts c_{ijkl}

Tant que $(\exists (i,j,k,l)$ tel que $(\alpha_i \beta_j \ \gamma_k \delta_l > 0$) **Faire**

- Prendre $(\overline{i}, \overline{j}, \overline{k}, \overline{l})$ la première case

• Prendre $x_{\overline{ijkl}} = \min(\alpha_{\bar{i}}, \beta_{\bar{j}}, \gamma_{\bar{k}}, \delta_{\bar{l}})$, si $x_{\overline{ijkl}} < d_{\overline{ijkl}}$ alors $N_b = N_b + 1$

• Merttre à jour $\alpha_{\bar{i}}, \beta_{\bar{j}}, \gamma_{\bar{k}}, \delta_{\bar{l}}$ comme suit :

$$\begin{cases} \alpha_{\bar{i}} = \alpha_{\bar{i}} - x_{\overline{ijkl}} \\ \beta_{\bar{j}} = \beta_{\bar{j}} - x_{\overline{ijkl}} \\ \gamma_{\bar{k}} = \gamma_{\bar{k}} - x_{\overline{ijkl}} \\ \delta_{\bar{l}} = \delta_{\bar{l}} - x_{\overline{ijkl}} \end{cases}$$

Fin tant que.

• Calculer les $u_i,\ v_j,\ w_k$ et t_l en utilisant la (**TRVD**)

• Déterminer l'ensemble Π en utilisant le lemme **2** (critère d'arrêt).

Phase 2

Tant que $\Pi \neq \emptyset$ **Faire**

• Trouver (i_0, j_0, k_0, l_0) tel que $D_{i_0 j_0 k_0 l_0} = \max\limits_{(i,j,k,l)} \Pi$

• Utiliser (**TRC**) pour déterminer les coefficients α_{ijkl} du cycle .

• Calculer $\theta^* = \min\left\{ \min\limits_{(i,j,k,l)\in B}(x_{ijkl}),\ \min\limits_{(i,j,k,l)\in F}(d_{ijkl} - x_{ijkl}) \right\} = \theta_{\overline{ijkl}}$

• Le nouvel itéré $x_{ijkl} = \begin{cases} x_{ijkl} + \alpha_{ijkl}\theta^* & \text{si} \quad (i,j,k,l) \in (F \cup B) \\ x_{ijkl} & \text{ailleurs} \end{cases}$

• Mettre à jour

$$I_B = \left\{ I_B \setminus \left\{ (\bar{i}, \bar{j}, \overline{k}, \bar{l}) \right\} \cup \left\{ (i_0, j_0, k_0, l_0) \right\} \right\}$$

• Calculer à nouveau les $u_i,\ v_j,\ w_k$ et t_l et déterminer l'ensemble Π.

Fin tant que

Fin algorithme.

Lemme 4 *[4]*

Soient $x^{(r)}$, $x^{(r+1)}$) deux solutions réalisables consécutives non dégénérées dont les valeurs de l'objectif correspondantes sont $Z^{(r)}$, $Z^{(r+1)}$ respectivement, alors

$$Z^{(r+1)} = Z^{(r)} - \theta^{(r)} D^{(r+1)}_{i_0 j_0 k_0 l_0} \tag{2.22}$$

d'où

$$Z^{(r+1)} < Z^{(r)}.$$

Preuve.

On considère deux cas :

Premier cas : $(i_0, j_0, k_0, l_0) \in H_0$. On pose $\sigma^{(r)} = \{(i,j,k,l) \text{ cases formant lecycle}\}$.

On a :

$$Z^{(r+1)} = \sum_{(i,j,k,l)\in\sigma^{(r)}} c_{ijkl} x^{(r+1)}_{ijkl} + \sum_{(i,j,k,l)\notin\sigma^{(r)}} c_{ijkl} x^{(r)}_{ijkl}. \tag{2.23}$$

Prendre

$$\sum_{(i,j,k,l)\notin\sigma^{(r)}} c_{ijkl} x^{(r)}_{ijkl} = K, \tag{2.24}$$

alors

$$Z^{(r+1)} = K + \sum_{(i,j,k,l)\in\sigma^{(r)}} c_{ijkl}(x^{(r+1)}_{ijkl} + \alpha_{ijkl}\theta^*). \tag{2.25}$$

Donc, si $\alpha_{i_0 j_0 k_0 l_0} = 1$ et $\widehat{\sigma}^{(r)} = \sigma^{(r)} - \{(i_0, j_0, k_0, l_0)\}$ alors

$$Z^{(r+1)} = Z^{(r)} - \theta^{(r)} \left(c_{i_0 j_0 k_0 l_0} + \sum_{(i,j,k,l)\notin\widehat{\sigma}^{(r)}} \alpha_{ijkl}(u^{(r)}_i + v^{(r)}_j + w^{(r)}_k + t^{(r)}_l) \right). \tag{2.26}$$

Posons

$$\begin{aligned} i(\widehat{\sigma}^{(r)}) &= \left\{(j,k,l) \text{ tel que } (i,j,k,l) \in \widehat{\sigma}^{(r)}\right\} \\ j(\widehat{\sigma}^{(r)}) &= \left\{(i,k,l) \text{ tel que } (i,j,k,l) \in \widehat{\sigma}^{(r)}\right\} \\ k(\widehat{\sigma}^{(r)}) &= \left\{(i,j,l) \text{ tel que } (i,j,k,l) \in \widehat{\sigma}^{(r)}\right\} \\ l(\widehat{\sigma}^{(r)}) &= \left\{(i,j,k) \text{ tel que } (i,j,k,l) \in \widehat{\sigma}^{(r)}\right\}. \end{aligned} \tag{2.27}$$

On vérifie facilement, que l'on a pour tout $i \neq i_0$,

$$\sum_{(i,j,k,l)\notin\widehat{\sigma}^{(r)}} \alpha_{ijkl} = 0 \quad . \tag{2.28}$$

Et pour $i = i_0$

$$u_{i_0}^{(r)} \left(\sum_{(i_0,j,k,l)\notin\widehat{\sigma}^{(r)}} \alpha_{i_0jkl} + 1 \right) = 0. \tag{2.29}$$

Par conséquent

$$\sum_{(i_0,j,k,l)\notin\widehat{\sigma}^{(r)}} \alpha_{i_0jkl} u_{i_0}^{(r)} = -u_{i_0}^{(r)}. \tag{2.30}$$

On obtient des résultats analogues aux (2.26) et (2.28) pour j, k, l.

Donc

$$\sum_{(i,j,k,l)\notin\widehat{\sigma}^{(r)}} \alpha_{ijkl}(u_i^{(r)} + v_j^{(r)} + w_k^{(r)} + t_l^{(r)}) = -(u_0^{(r)} + u_0^{(r)} + u_0^{(r)} + u_0^{(r)}) \tag{2.31}$$

En substituant cette valeur dans (2.26), on obtient

$$Z^{(r+1)} = Z^{(r)} - \theta^{(r)} D_{i_0j_0k_0l_0}^{(r+1)} \tag{2.32}$$

Ce qui montre que $Z^{(r+1)} < Z^{(r)}$ car $\theta^{(r)} > 0$ et $D_{i_0j_0k_0l_0}^{(r+1)} > 0$

Deuxième cas : $(i_0, j_0, k_0, l_0) \in H_d$

Par une démonstration analogue à celle utilisée dans le premier cas, on obtient

$$Z^{(r+1)} = Z^{(r)} - \theta^{(r)} D^{(r+1)}_{i_0 j_0 k_0 l_0} \tag{2.33}$$

Ce qui montre que $Z^{(r+1)} < Z^{(r)}$ car $\theta^{(r)} > 0$ et $D^{(r+1)}_{i_0 j_0 k_0 l_0} > 0$. ■

Théorème 7 *[4]*

Supposons que le problème est non dégénéré, alors l'algorithme AL_{PT4Cm} converge en un nombre fini d'itérations.

Preuve.

Le lemme précédent montre que l'algorithme AL_{PT4Cm} garantie que la même base ne peut jamais apparaître dans deux itérations distinctes, et comme le nombre de sommets visités est nécessairement fini, l'algorithme AL_{PT4Cm} converge et sa convergence est finie. ■

2.4 Tests numériques

Tout d'abord, considérons un problème du transport de la forme $PT4C$ avec ;

$m = n = p = 2,\ q = 1,\ \alpha_1 = 5,\ \alpha_2 = 10,\ \beta_1 = 9,\ \beta_2 = 6,\ \gamma_1 = 12,\ \gamma_2 = 3,\ \delta_1 = 15.$

Les quantités c_{ijkl} et d_{ijkl} sont données dans le tableau 1.

(i, j, k, l)	1111	1121	1211	1221	2111	2121	2211	2221
d_{ijkl}	15	10	8	9	7	11	11	9
c_{ijkl}	5	4	6	7	2	1	5	4

Tableau 1 : Quantités c_{ijkl} et d_{ijkl} du problème

Dans cet exemple illustratif, nous présentons seulement la solution par leurs composantes non nulles. La résolution de ce problème par AL_{PT4Cm} conduit aux résultats :

• Les éléments non nuls de la solution optimale $x^* = (x_{ijkl})$ sont : $x_{1211} = 5,\ x_{2111} = 6,\ x_{2121} = 3$ et $x_{2211} = 1$.

• Le nombre d'itérations est 2. Le temps d'exécution $= 0,000\ sec.$

• La valeur optimale est $z^* = 50$.

En second lieu, on compare notre méthode avec l'algorithme AL_{PT4C} décrit dans [61], sur un échantillon de 40 problèmes dont les dimensions sont données au tableau 2 et 3 ci-dessous. Tous ces problèmes sont générés aléatoirement. Nous noterons également la sensibilité de cette méthode AL_{PT4C} aux problèmes dégénérés de grande taille, à partir du $14^{ème}$ exemple de taille (23×1080) l'algorithme (AL_{PT4C}) échoue c'est-à-dire : incapable de traiter le phénomène de dégénérescence. C'est pourquoi, dans le tableau 2, correspondant aux problèmes 1 à 13, nous résumons les résultats numériques obtenus par les deux méthodes. Tendis que dans le tableau 3, nous exposons que les résultats obtenus par notre méthode pour les problèmes 14 à 40.

Nos tests sont effectués sur un Intel (R) Pentium (R) Dual Windows, les programmes sont écrits en C. Grâce aux tests numériques que nous avions réalisé, nous nous rendons compte de la stabilité et de la robustesse de notre algorithme par rapport à la méthode introduite dans [61].

- Nb_{It} : le nombre d'itération
- z^* : l'optimum
- T_{cpu} : le temps d'execution en seconde
- $ALg1 : AL_{PT4C}$ l'algorithme introduit dans [61]
- $ALg2 : AL_{PT4Cm}$ le nouvel algorithme introduit dans [4]

Ex	Taille	Nb_{It}		z^*		T_{cpu}	
N	$M \times N$	$ALg2$	$ALg1$	$ALg2$	$ALg1$	$ALg2$	$ALg1$
1	8×16	3	38	9	9	0	0
2	9×24	3	45	11	11	0	0.015
3	10×36	5	55	140	140	0	0.015
4	11×54	7	66	180	1.031	0	0.015
5	12×81	7	76	186	186	0	0.47
6	13×108	8	87	193	193	0	0.078
7	14×144	8	98	2950	2950	0	0141
8	18×360	12	152	253	253	0	0.687
9	19×600	13	185	1100	1100	0	6.125
10	20×648	14	184	269	269	0	1.406
11	21×720	14	202	235.93	235.93	0	8.578
12	22×750	14	203	237.75	237.75	0	2.734
13	23×900	14	220	242.18	242.18	0	4.312

Tableau 2 : Comparaison des deux méthodes AL_{PT4Cm} et AL_{PT4C}

Ex	$M \times N$	Nb_{It}	z^*	T_{cpu}
14	23×1080	6	189.063	0.062
15	24×1296	6	212.625	0.110
16	25×1512	7	7150	0.125
17	26×1764	7	3777.5	0.187
18	27×2058	7	8425	0.265
19	28×2401	7	2744	0.407
20	29×2744	8	3096	0.422
21	30×3136	8	492.5	0.610
22	31×3584	8	401	0.843
23	32×4096	8	714	1.187
24	33×4608	9	113.25	1.250
25	34×5184	9	691.5	1.719
26	35×5832	9	351.5	2.297
27	36×6561	9	477	3.078
28	37×7290	10	130.75	3.250
29	38×8100	10	266	4.312
30	39×9000	10	405.75	5.610
31	40×10000	10	550	7.282
32	41×11000	11	298.5	7.625
33	42×12100	11	303.5	9.828
34	43×13310	11	115.688	12.469
35	44×14641	11	117.536	15.782
36	45×15972	12	42.1875	16.469
37	46×17424	12	42.875	20.781
38	47×19008	12	30.688	25.871
39	48×20736	12	44.25	32.016
40	49×22446	13	47.3125	33.281

Tableau 3 : Résultat numérique pour la méthode proposée AL_{PT4Cm}

Notre algorithme traite le problème en question telle qu'il est sans lui augmenter la taille, et sans reformulation. Sachant que les problèmes de dégénérescences sont les plus fréquents en pratique, notre méthode utilise une technique qui génère aux pires des cas une base, ce qui nous permettent d'initialiser l'algorithme. On utilise des techniques de résolution en exploitant les particularités théoriques du problème. Nous pouvons toujours surmonter les diverses difficultés rencontrées lors de la résolution du problème. En récapitulatif, notre méthode s'est révélée très performante et compétitive en terme du temps d'exécution, le nombre d'itérations, et le pouvoir de la résolution notamment les problèmes dégénérés.

2.5 Conclusion

Un nouvel algorithme de résolution pour un problème du transport à quatre indices avec capacités est proposé dans ce chapitre. La solution optimale est souvent atteinte dès la phase d'initialisation, du fait qu'on évite les affectations inutiles et on traite le phénomène de la dégénérescence en utilisant un procédé détective, ce qui rend l'algorithme très robuste au pire des cas. Nous avons pu surmonter tous les difficultés rencontrées lors de l'implémentation avec succès, grâce à des techniques numériques utilisées pour résoudre divers systèmes linéaires qui servent à l'amélioration d'une solution initiale de base afin d'obtenir l'optimum.

Chapitre 3

Minimisation d'une fonction non convexe sur un intervalle

3.1 Introduction

Trouver un minimum global est en général un problème très difficile. En réalité il n'existe pas d'algorithmes parfaits. En fonction des problèmes on cherche la méthode la mieux adaptée. L'intérêt des algorithmes unidimensionnels ne vient pas seulement du fait que dans les applications on rencontre naturellement des problèmes unidimensionnels mais aussi du fait que ce dernier est un composant fondamental de bon nombre de méthodes d'optimisation multidimensionnelle.

Dans ce chapitre on s'intéresse à la minimisation d'une fonction d'une seule variable non convexe deux fois différentiable, en utilisant l'algorithme de Séparation et Évaluation (Branch and Bound) qui repose sur des techniques de sous estimateurs de la fonction objectif. À ce propos plusieurs travaux ont été proposés entre autres : Adjiman et al [7] et Hertz et al [29] ont travaillé sur la méthode αBB et ont proposé plusieurs nouvelles méthodes rigoureuses pour le calcul du paramètre α. Adjiman et al [6] ont présenté la mise en œuvre détaillée de l'approche αBB et des études de calcul des problèmes de conception de processus tels que les réseaux de l'échangeur de chaleur, réseaux réacteur-séparateur,

et la conception des lots dans l'incertitude. Ryoo et Sahinidis [51] ont étudié les bornes des fonctions multilinéaires par intervalles arithmétiques, intervalles arithmétiques récursifs, transformation logarithmique, et transformation exponentielle en fournissant des comparaisons des relaxations convexes résultantes. Tawarmalani et al [54] ont montré que les resserrements des relaxations linéaires peuvent être obtenus si le produit d'une variable continue et la somme de plusieurs variables continues est décomposable, Ils appliquent leur technique à certains problèmes concrets issus de la physique. Tawarmalani et Sahinidis [55] en introduisant des extensions des fonctions bornes convexes semi-continues, et ont étudié les conditions d'existences, et ont proposé une technique pour la construction d'enveloppes convexes de fonctions non linéaires, ils ont aussi étudié la distance de séparation maximale pour des fonctions telles que $\frac{x}{y}$. Akrotirianakis et Floudas [8] ont proposé une nouvelle classe de fonctions bornes convexes pour PNL deux fois continûment différentiables, ils ont étudié leurs propriétés théoriques et ont prouvé que la relaxation convexe résultante est améliorée par rapport à αBB. En outre, Akrotirianakis et Floudas [9] ont présenté les résultats de calcul de la nouvelle classe de fonctions bornes convexes utilisées dans Branch and Bound pour un PNL sur un pavé. Ils ont également proposé une méthode d'optimisation globale hybride qui comprend l'approche stochastique aléatoire dans le but d'améliorer les performances de calcul. Caratzoulas et Floudas [**?**] ont proposé de nouvelles fonctions bornes convexes pour les fonctions trigonométriques, qui sont elle mêmes des fonctions trigonométriques. Le procédé de sous-estimateur peut être appliqué à des problèmes unidimensionnels ainsi que multidimensionnels impliquant des polynômes trigonométriques, puisque le produit des fonctions trigonométriques peut toujours être décomposé en une somme de fonctions sinus et cosinus avec des arguments qui sont des combinaisons linéaires des variables du problème. Meyer et Floudas [41] ont proposé deux nouvelles classes de fonctions bornes convexes pour PNL de classe C^2 qui combinent les bornes αBB avec une perturbation quadratique par morceaux. Sherali et al [53] ont proposé une nouvelle méthode de plan de coupe qui est basé sur la construction d'une représentation de l'enveloppe convexe partielle pour un problème de programma-

tion mixte en nombres entiers $0, 1$ en utilisant la technique reformulation-linéarisation (TRL). Les coupes sont générées en projetant l'espace étendu de la formulation TRL dans l'espace d'origine et les auteurs ont étudié plusieurs règles de sélection de variables pour effectuer la convexification d'une manière efficace. Gounaris et Floudas [24] ont développé des fonctions bornes convexes serrées pour des fonctions d'une seule variable de classe C^2. L'idée est basée sur une application par morceaux des bornes αBB. Il est prouvé que théoriquement un nombre fini d'éléments sont suffisants pour obtenir une enveloppe convexe de la fonction. La méthodologie a été étendue pour gérer les fonctions à plusieurs variables (Gounaris et Floudas [25], par le biais de projections appropriées de l'épigraphe de la fonction dans certains espaces unidimensionnels. Des transformations orthonormées ont également été utilisées pour améliorer la qualité de la sous-estimation. Il est important de signaler les importants travaux d'Ouanes et al [47] qui ont proposé une fonction borne quadratique dont le minimum est calculé explicitement, ansi la résolution n'exige pas l'utilisation des solveurs locaux.

On va d'abord aborder les principales techniques fréquemment utilisées dans l'optimisation globale à savoir :

- Séparations et Évaluation.
- Les fonctions minorants l'objectif.

3.2 Méthode de Séparation et Évaluation

Pour plusieurs problèmes, en particulier les problèmes d'optimisation, l'ensemble de leurs solutions est fini (en tous les cas, il est dénombrable). Il est donc possible, en principe, d'énumérer toutes ces solutions, et prendre celle qui nous arrange. L'inconvénient majeur de cette approche est le nombre prohibitif des solutions : puisqu'il n'est pas facile d'effectuer cette énumération . La méthode de séparation et évaluation (procédure par évaluation et séparation progressive) consiste à énumérer ces solutions d'une manière intelligente en ce sens que, en utilisant certaines propriétés du problème en question, cette technique arrive à éliminer des solutions partielles qui ne mènent pas à la solution que l'on cherche. De ce fait, on arrive souvent à obtenir la solution recherchée en des temps raisonnables. Bien entendu, dans le pire des cas on arrive toujours sur l'élimination explicite de toutes les régions qui ne peuvent pas contenir la solution optimale.

Description de l'algorithme de base

Par convenance, on représente l'exécution de la méthode de séparation et évaluation à travers une arborescence. La racine de cette arborescence représente l'ensemble de toutes les solutions du problème considéré. Pour appliquer la méthode de Séparation et Évaluation, nous devons être en possession de :

1. Un moyen de calcul d'une borne inférieure d'une solution partielle.

2. Une stratégie de subdiviser l'espace de recherche pour créer des espaces de recherche de plus en plus petits.

3. Un moyen de calcul d'une borne supérieure pour au moins une solution.

La méthode commence par considérer le problème de départ avec son ensemble de solutions, appelées la racine. Des procédures de bornes inférieures et supérieures sont appliquées à la racine. Si ses deux bornes sont égales, alors une solution optimale est trouvée, et la procédure est arrêtée, l'ensemble des solutions est divisé en deux ou plusieurs sous-problèmes descendants de la racine. La méthode est ensuite appliquée récursivement à ses sous-problèmes, en engendrant ainsi une arborescence. Si une solution optimale est

trouvée pour un sous problème, elle est réalisable, mais pas nécessairement optimale, pour le problème de départ. Comme elle est réalisable, elle peut être utilisée pour éliminer toute la descendance : si la borne inférieure d'un nœud dépasse la valeur d'une solution déjà connue, alors on peut affirmer que la solution optimale globale ne peut être contenue dans le sous-ensemble de solution représenté par ce nœud. La recherche continue jusqu'à ce que tous les nœuds sont explorés ou bien éliminés.

Dans ce qui suit, nous résumons la méthode de Séparation et Évaluation sur un problème de minimisation :

$$\alpha = \min f(x) \ , x \in X = [a, b],$$

où $f : X \longrightarrow \mathbb{R}$ $(X \subset \mathbb{R})$ continue et non convexe.

L'algorithme de Séparation et Évaluation consiste à engendrer deux suites convergentes $\{UB^k\}$ et $\{LB^k\}$ représentent des bornes supérieures et inférieures de la valeur minimale de la fonction objectif. À chaque itération k les problèmes des bornes inf. et sup. seront résolus sur un nombre fini de sous ensembles de X.

On notera ces sous-ensembles $X_i^k \in M^k$ où M^k est la classe des sous-ensembles actifs à l'itération k. Sur chaque X_i^k les bornes LB^ket UB^k seront calculées par la relaxation de f sur X_i^k.

En effet, LB^k et UB^k obtenus à l'itération k sont données par :

$$\begin{cases} LB^k = \min LB_i^k \\ UB^k = \min UB_i^k \end{cases}.$$

et tout sous-ensemble sur lequel LB^k dépasse UB^k sera éliminé, car $\min f$ ne peut pas être atteint sur un tel sous-ensemble.

Au fait, cette méthode peut se représenter schématiquement par une arborescente qui a comme racine l'ensemble X, et comme sommets les sous-ensembles X_i^k qui s'obtiennent par subdivisions successives. Deux sommets seront reliés si et seulement si le deuxième

sous-ensemble est obtenu par la partition directe du premier, et à chaque niveau de l'arborescence les bornes LB^k et UB^k seraient obtenues par l'application d'une recherche locale.

On peut résumer la procédure précédente par les étapes suivantes ;

L'algorithme de base

- $\varepsilon > 0$ est une précision donnée
- Poser $k = 1$, $M^k = X$
- Construire les problèmes des bornes inf. et sup. de $\min f(x)$ sur X.

Soient LB^k et UB^k les solutions obtenues respectivement.

- Tant que $LB^k - UB^k > \varepsilon$ subdivisé X en deux sous-ensembles X_1^k et X_2^k tel que :

$$X_1^k \cup X_2^k = X, \quad \text{et} \quad \overset{\circ}{X}{}_1^k \cap \overset{\circ}{X}{}_2^k = \emptyset$$

où $\overset{\circ}{X}$ est l'intérieur de X.

- Construire les problèmes des bornes inf. et sup. de $\min f(x)$ sur $X_i^k, i = 1, 2$.

Soient LB_1^k, UB_1^k et LB_2^k, UB_2^k les solutions obtenues.

- Poser

$$\begin{cases} LB^k = \min\{LB_1^k, LB_2^k\} \\ UB^k = \min\{UB_1^k, UB_2^k\} \end{cases}.$$

- Poser $M^k = \{R_1^k, R_2^k\}$.
- Éliminer de M^k tout sous-ensemble $R_j^k, j = 1, 2$, tel que $LB_j^k > UB_j^k$ et poser : $M^k = R_i^k$.
- Posons $k = k + 1$ et réitérer les procédures.
- poser

$$\min f(x) = UB^k, \text{ et } x^* = x^k \in \{x : f(x) = UB^k, \ x \in X\}$$

Convergence

Pour établir la convergence de l'algorithme, on se sert du théorème suivant ;

Théorème 8

Si pour chaque suite infinie $\{R^k\}$, $R^{k+1} \subset R^k$, $k \in \mathbb{N}$ *des ensembles des partitions successives, les bornes inférieures et supérieures vérifient :*

$$\lim_{k\to\infty}(LB^k - UB^k) = \lim_{k\to\infty}(LB^k - UB(R^k)) = 0.$$

alors

$$UB = \lim_{k\to\infty} UB^k = \lim_{k\to\infty} f(x^k) = \lim_{k\to\infty} LB^k = LB.$$

3.3 Les fonctions minorantes

Soit le problème d'optimisation globale :

$$(P)\begin{cases} \alpha = \min f(x) \\ x \in X = [a,b] \end{cases}, \tag{3.1}$$

f est une fonction non convexe deux fois différentiable sur X .

Pour résoudre ce problème la méthode de séparation et évaluation se dote d'une fonction qui permet de mettre une borne a certaines solutions pour soit les exclure soit les maintenir comme des solutions potentielles. Bien entendu, la performance d'une méthode de branch and bound dépend, entre autres, de la qualité de cette fonction (de sa capacité d'exclure des solutions partielles plus tôt).

Dans ce qui suit on désigne par ;

- αBB l'algorithme de Séparation et Évaluation proposé par Floudas et al [7], [6].
- KBB l'algorithme de Séparation et Évaluation proposé par Ouanes et al [37].
- KBB_m l'algorithme que nous proposons dans ce chapitre [5].

Dans ce qui suit on donne les deux bornes développées dans les méthodes αBB et KBB.

3.3.1 La fonction borne inférieure αBB

Dans cette méthode développée par Floudas et al [19] on utilise la différence entre l'objectif et une fonction quadratique concave :

$$L(x) = f(x) - Q(x)$$

avec

$$Q(x) = \frac{1}{2}\alpha(x-a)(b-x)$$

où

- $f(x)$ la fonction objectif non convexe deux fois différentiable.
- $L(x)$ la fonction borne inférieure.
- $Q(x)$ la fonction quadratique qui rend $L(x)$ convexe.
- α constante positive vérifiant l'inégalité suivante :$\alpha \geq \max\left\{0, -\min_{[a,b]} f''(x)\right\}$

La borne $L(x)$ vérifie les propriétés suivantes :

1. Elle est convexe (i.e. $L''(x) = f''(x) + \alpha \geq f''(x) + \max\left\{0, -\min_{[a,b]} f''(x)\right\} \geq 0$, pour tout $x \in [a,b]$.
2. Elle est un sous estimateur de l'objectif. En effet, on a :

$$\phi(x) = f(x) - L(x).$$

Ce qui en suit ;

$$\phi''(x) = -2\alpha \leq 0 \text{ pour tout } x \text{ dans } [a,b],$$

Donc ϕ est une fonction concave, et alors on a ;

$$\phi(x) \geq \min_{x\in[a,b]} \phi(x) = \phi(a) = \phi(b) = 0.$$

3. La borne inférieure est obtenue par la résolution du problème convexe ;

$$\min_{[a,b]} L(x).$$

4. Elle coïncide avec $f(x)$ aux extrémités de l'intervalle $[a, b]$.

3.3.2 La fonction borne inférieure KBB

Dans cette méthode développée par Ouanes et al [37], ils utilisent la différence entre l'interpolant linéaire et une quadratique concave. D'une façon précise, soit K un réel tel que $|f''(x)| \leq K$ pour tout x dans $[a, b]$. Soit l_0 et $l_1$deux fonctions réelles tels que :

$$\begin{cases} l_0(x) = \dfrac{b-x}{b-a} \\ l_1(x) = \dfrac{x-a}{b-a} \end{cases} \quad \text{si } x \in [a, b],$$

pour tout x dans l'intervalle $[a, b]$, on a $l_0(x) + l_1(x) = 1$. On a aussi $l_i(x^j)$ est égal à 0 si $i \neq j$, et 1 autrement. Soit Lf l'interpolant linéaire de f aux points a, b ([18] ,[21])

$$Lf(x) = \sum_{i=0}^{1} l_i(x) f(x^i).$$

La quadratique borne inférieure $q(f)$ de f sur l' intervalle $[a, b]$ est la différence de l'interpolant linéaire et la quadratique concave. elle est donnée par :

$$\begin{aligned} q(x) &= Lf(x) - Q(x), \\ \text{avec } Q(x) &= \frac{1}{2} K(x-a)(b-x). \end{aligned}$$

Où K est une constante positive telle que $|f''(x)| \leq K$, pour tout x dans l'intervalle $[a, b]$.

En effet, la minorisation de q est démontrée dans le théorème suivant ;

Théorème 9

On a

$$Lf(x) - \frac{1}{8}K(b-a)^2 \leq q(x) \leq f(x),\ \forall x \in [a,b] \tag{3.2}$$

Preuve.

$$\begin{aligned} E(x) &= Lf(x) - \tfrac{1}{8}K(b-a)^2 - q(x) = -\frac{1}{8}(b-a)^2 + \frac{K}{2}(x-a)(b-x) \\ &= \frac{K}{2}\left[-x^2 + (a+b)x - ab - \frac{1}{4}(b-a)\right]. \end{aligned}$$

La fonction E est concave sur [a, b], et sa dérivée s'annule au point $x^* = \dfrac{1}{2}(a+b)$.

Par conséquent, pour tout $x \in [a,b]$ on a ;

$$E(x) \leq \max_{x\in[a,b]} E(x) = E(x^*) = 0,$$

la première inégalité de (3.2) est vérifiée.

Pour justifier la deuxième inégalité, on considère la fonction ϕ sur X définie par ;

$$\phi(x) = f(x) - q(x) = f(x) - Lf(x) + \frac{1}{2}K(x-a)(b-x).$$

C'est clair que $\phi''(x) = f''(x) - K \leq 0$ pour tout x dans $[a,b]$. Alors ϕ est une fonction concave, et on a :

$$\phi(x) \geq \min_{x\in[a,b]} \phi(x) = \phi(a) = \phi(b) = 0,$$

la seconde inégalité de (3.2) est prouvée. ■

La borne inférieure de $f(x)$ sur $[a,b]$, LB est calculé par la résolution du problème convexe suivant ;

$$LB = \min_{a\leq\ x\leq b} q(x). \tag{3.3}$$

Du fait que, q est une fonction quadratique de la forme ;

$$q(x) = \frac{K}{2}x^2 + \left[\frac{f(b)-f(a)}{(b-a)} - \frac{K}{2}(a+b)\right]x + \frac{K}{2}ab + \frac{f(a)b - f(b)a}{(b-a)}.$$

La solution optimale de (3.3) est donnée explicitement par ;

$$x^* = \begin{cases} \dfrac{1}{2}(a+b) - \dfrac{1}{K}\dfrac{f(b)-f(a)}{b-a} & \text{si} \quad x \in [a,b] \\ a & \text{si} \quad x \leq a \\ b & \text{si} \quad x \geq b \end{cases} . \tag{3.4}$$

Remarque 8

Si $x^* \notin [\ a, b\ ]$, *alors*

$$\min_{a \leq\ x \leq b} q(x) = \min_{a \leq\ x \leq b} f(x) = \min\{f(a), f(b)\}.$$

Bref, la borne KBB vérifie les propriétés suivantes :

1. Elle est convexe (i.e. $q''(x) = K \geq f''(x)$, pour tout $x \in [a,b]$.
2. Elle est un sous estimateur de l'objectif d'après le théorème précédent.
3. La valeur de la borne inférieure est donnée explicitement en utilisant (3.3).
4. Elle coïncide avec $f(x)$ aux extrémités de l'intervalle $[a,b]$.

Avantages et inconvénients

1. L'avantage principal de la borne $L(x)$ est qu'elle coïncide avec la fonction objectif dans les régions où l'objectif est convexe.
2. L'inconvénient principal de la méthode $\alpha\ BB$ est qu'elle utilise une méthode itérative locale pour déterminer la valeur de la borne inférieure.
3. L'avantage principal de la borne $q(x)$ est qu'elle possède des minima explicites.
4. L'inconvénient principal de la borne $q(x)$ est qu'elle s'éloigne de la fonction objectif pour les régions où l'objectif est convexe.

3.3.3 Une nouvelle fonction borne $\mathbf{KBB}_m$

Maintenant on va établir une nouvelle borne inférieure [5] qui rassemble les avantages des deux méthodes $\alpha\ BB$ et KBB. Autrement dit on se place le plus proche possible de

la fonction objectif autant que l'on veut.

Soit $f(x)$ une fonction a une variable qui doit être minorée sur l'intervalle $[a, b]$. Supposant que les points d'appui sont choisis équidistant dans $[a, b]$, c'est-à-dire : $x_i = a + ih,\ h = \dfrac{b-a}{n},\ i = 0, ..., n.$ avec $n \geq 2$. Soit l_i et l_{i+1}deux fonctions réelles telles que pour tout $i = 0, ..., n-1$ on a :

$$\begin{cases} l_i(x) = \dfrac{x - x_i}{x_{i+1} - x_i} \\ l_{i+1}(x) = \dfrac{x_{i+1} - x}{x_{i+1} - x_i} \end{cases} \quad \text{si } x \in [x_i, x_{i+1}], \tag{3.5}$$

et

on a

$$\sum_{i}^{i+1} l_i(x) = 1,\ \forall x \in [x_i, x_{i+1}],$$

$$l_i(x_j) = \begin{cases} 1, & i = j \\ 0\ , & i \neq j \end{cases}. \tag{3.6}$$

Soit $L_h f$ l'interpolant linéaire de f aux points x_i, x_{i+1} :

$$L_h f(x)\ = \sum_{i}^{i+1} f(x_i) l_i(x). \tag{3.7}$$

Pour chaque intervalle $[x_i, x_{+1}]$ on construit la minorisation locale :

$$p_i(x) = L_h f(x) - Q_i(x), \tag{3.8}$$

avec

$$Q_i(x) = \frac{1}{2} K_i (x - x_i)(x_{i+1} - x), \tag{3.9}$$

où K_i est un majorant de la dérivée seconde de la fonction objectif sur $[x_i, x_{+1}]$.

Le calcul des paramètres K_i

En pratique l'arithmétique d'intervalles est utilisée pour calculer des minorants et des majorants de fonction de $\mathbb{R}^n$ dans $\mathbb{R}$. Pour cela, introduisons d'abord quelques définitions.

Définitions 2 ***[40]***

1- *Soit $f : \mathbb{R}^n \to \mathbb{R}$, f possède une expression explicite si l'expression analytique de f est connue de façon explicite, c'est-à-dire qu'elle peut s'écrire en n'utilisant que des variables, des fonctions et des opérations élémentaires tels que $+$, $-$, $\times$, $/$, $\sqrt{}$, $\log$, $\exp$, abs, $\cos$, $\sin$, $\arccos$. On dit que f est une fonction explicite.*

2- *soit $f : X \subset \mathbb{R}^n \to \mathbb{R}$. Une fonction d'inclusion de f est une fonction retournant un minorant et un majorant de f sur un intervalle donné. On la note, de manière générale, $F : X \cap I^n \to I$. Ainsi : $\forall z \in Z \subseteq X,\ f(z) \in F(Z)$.*

3- *soit $f : \mathbb{R}^n \to \mathbb{R}$. Une extension naturelle aux intervalles d'une fonction est la réécriture de f en remplaçant toutes les occurrences d'une variable par l'intervalle correspondant et les opérateurs classiques par leur équivalent en arithmétique d'intervalles. Elle est notée $EN_f(X)$.*

Toutes les fonctions explicites possèdent une extension naturelle. Mais, l'évaluation de celle-ci n'est pas unique. En effet, certaines fonctions peuvent s'écrire sous forme développée ou factorisée, il existe donc une extension naturelle pour chaque façon d'écrire la fonction.

Proposition 3 ***[40]***

L'extension naturelle aux intervalles d'une fonction est une fonction d'inclusion.

Pratiquement les paramètres K_i sont calculés de la façon suivante : on cherche des majorants K_i de $|f''(x)|$ sur les intervalles $[x_i, x_{i+1}] = X_i$ pour tout $i = 0, ..., n-1$. Posons $|f''(x)| = h(x)$ et on auras par la suite

$$EN_h(X_i) = EN_h[x_i, x_{i+1}] = [A_i, B_i],\ \text{prenons } K_i = B_i.$$

Dans chaque intervalle $[x_i, x_{i+1}]$, on calcule la borne inférieure de la fonction quadratique minorant l'objectif f

$$x_i^* = \begin{cases} \frac{1}{2}(x_i + x_{i+1}) - \frac{1}{K_i}\frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} & \text{si} \quad x \in [x_i, x_{i+1}] \\ x_i & \text{si} \quad x \leq x_i \\ x_{i+1} & \text{si} \quad x \geq x_{i+1} \end{cases} . \tag{3.10}$$

Ensuite, la plus petite valeur de $p_i(x_i^*)$ est choisie comme une borne inférieure

$$LB_i = \min_i p_i(x_i^*) \tag{3.11}$$

La fonction objectif est évaluée en différents points dans le but de déterminer la borne supérieure ;

$$UB_i = \min\{f(x_i^*),\ f(x_i)\} \tag{3.12}$$

Théorème 10

On a

$$q(x) \leq p(x) \leq f(x), \quad \forall x \in [a, b], \tag{3.13}$$

où

$$p(x) = p_i(x),\ \forall x \in [\ x_i, x_{i+1}]\ ,\ i = 1, ..., n. \tag{3.14}$$

$p(x)$ est une fonction borne inferieure pour $f(x)$ pour tout x dans $[a, b]$, Elle est meilleur que la fonction borne $q(x)$ introduite dans (3.18).

Preuve.

Pour chaque intervalle $[x_i, x_{i+1}]$ on pose

$$E(x) = q(x) - p_i(x) = \frac{1}{2}(K_i - K)(x - x_i)(x_{i+1} - x).$$

C'est a dire on a $E''(x) = K - K_i \geq 0$ pour tout $x \in [x_i, x_{i+1}]$, alors E est une fonction

concave par morceaux, et en conséquence pour tout $x \in [x_i, x_{i+1}]$ on a :

$$E(x) \leq \max_{x \in [x_i, x_{i+1}]} E(x) = E(x_i) = E(x_{i+1}) = 0.$$

Et la première inégalité de (3.13) est vérifié. Pour prouver la seconde inégalité, considérons maintenant la fonction ϕ définie sur $[x_i, x_{i+1}]$ par :

$$\phi(x) = f(x) - p_i(x) = f(x) - L_h f(x) + \frac{1}{2} K_i (x - x_i)(x_{i+1} - x). \tag{3.15}$$

C'est clair, que $\phi''(x) = f''(x) - K_i \leq 0$ pour tout x dans $[x_i, x_{i+1}]$ donc ϕ est une fonction concave par morceaux, ce qui entraine :

$$\phi(x) \geq \min_{x \in [x_i, x_{i+1}]} \phi(x) = \phi(x_i) = \phi(x_{i+1}) = 0.$$

La seconde inégalité (3.13) est aussi prouvée. ■

Pratiquement nous résumons les étapes de la résolution dans l'algorithme de Branch and Bound suivant :

3.4 L'algorithme proposé

Algorithme 4

Entrées :

- f la fonction objectif
- $[a, b]$ intervalle des réelles
- n le nombre de quadratiques utilisées (sous-estimateurs)
- ε une précision donnée

Sorties :

- x^* minimum globale de f

1. **Initialisation** $k = 0$

   (a) **pour tout** $i = 0, ..., n$ Calculer $x_i = a + \dfrac{b-a}{n} i$, et soit $M = \underset{i=0}{\overset{n-1}{\cup}} \{[x_i, x_{i+1}]\}$

   (b) Calculer K_i tel que $|f''(x)| \leq K_i$ pour chaque $[x_i, x_{i+1}]$ **pour tout** $i = 0, ..., n-1$.

   (c) Calculer x_i^* En utilisant (**3.10**) **pour tout** $i = 0, ..., n-1$.

   (d) Calculer $UB^k = \min\{\min f(x_i^*), \min f(x_i)\}$

   (e) **Soit** $LB^k = \min\limits_i LB_i^k$ **avec** $LB_i^k = p_i(x_i^*)$

   $\bar{i}$: l'indice correspondant au $\min LB_i^k$

2. **Itération**

   Tant que $(UB^k - LB^k > \varepsilon$ et $M \neq \emptyset)$ **faire**

   (a) $a := x_{\bar{i}}$,$b := x_{\bar{i}+1}$ et appliquer les étapes a,b,c et d de **(1)**

   (b) Mettre à jour à jour UB^k

   (c) **Pour tout** $i = 1, ..., m$; $(m = card\ (M))$

   – **Élimination : si** $(UB^k - LB_i^k < \varepsilon)$ **alors** supprimer $[x_i, x_{i+1}]$ de M

   – **Sélection : si**$(UB^k - LB_i^k \geq \varepsilon)$ **alors** sélectionner

   $\bar{i}$: l'indice correspondant au $\min LB_i^k$

   (d) $k = k + 1$

   fin Tant que

3. $x^* = x^k$ la solution optimale correspondante au meilleur UB^k trouvé.

fin algorithme

Théorème 11 ***[Convergence de l'algorithme]***

Soit l'algorithme est fini ou il génère une suite bornée $\{x_k\}$*. Tout point d'accumulation de la suite est une solution optimale globale de* (P) *et on a* $UB^k \searrow \alpha$, $LB^k \nearrow \alpha$.

Preuve.

Supposons que l'algorithme est infini, il génère une suite infinie d'intervalles $\{T^k\}$ dont les longueurs h_i avec $i = 1, ..., n$ diminuent à zéro, alors la suite $\{T^k\}$ se rétrécit à un singleton. Étant donné que les valeurs d' UB^k obtenus en évaluant $f(x)$ en differents points de $[a, b]$, alors la suite $\{UB^k\}$ est minorée par $\alpha = \min_{[a,b]} f(x)$. D'autre part, les valeurs de LB^k sont précisément les minima de certaines quadratiques minorants l'objectif, donc ne doivent pas dépasser $\alpha = \min_{[a,b]} f(x)$, alors la suite $\{LB^k\}$ est majorée par $\alpha = \min_{[a,b]} f(x)$.

En conséquence, on a

$$LB^k \leq \alpha = \min_{[a,b]} f(x \leq UB^k,$$

il suffit de démontrer que les deux suite sont monotones.

Tout d'abord, à partir de la description de l'algorithme, nous voyons que, à chaque itération $k+1, k \geq 0$ la valeur d' UB^{k+1} sera choisi comme la moindre d' UB^k courant et la nouvelle valeur qui doit être déterminée.Ce qui entraine toujours $UB^{k+1} \leq UB^k \ \forall k \geq 0$, alors $\{UB^k\}$ est une suite décroissante.

Et de même à chaque itération $k+1, k \geq 0$ la valeur de LB^{k+1} sera choisi comme l'optimum d'une certaine quadratique située à l'intérieur d'une grande quadratique couvrant l'intervalle actuel $[a_{k+1}, b_{k+1}]$ et minorant l'objectif. Ce qui entraine toujours $LB^{k+1} \geq LB^k \ \forall k \geq 0$, alors $\{LB^k\}$ est une suite croissante. i.e . Le théorème est démontré. ■

3.5 Étude numérique

Dans ce qui suit, nous considérons les notations suivantes ;

- f^* est l'optimum obtenu
- LB_0 est la borne inférieure initiale
- T_{cpu} est le temps d'exécution en seconde
- m est le nombre total d'intervalle
- m_e est le nombre d'intervalles éliminé
- LM est le nombre de minima locaux

- GM est le nombre de minima global
- $*$ (astérisque) indique que la borne est égale à l'optimum global connu f^*

Pour mesurer les performances de notre méthode KBB_m, nous comparons les algorithmes KBB, αBB et KBB_m. Ces algorithmes sont implémentés en écrivant les programmes en langage C avec double précision, et exécutés sur un ordinateur avec un processeur Intel (R) Core (TM) i3-311MCP4 avec une vitesse de 2,40 GHz.

Nos tests numériques sont effectués en deux parties sur un ensemble de fonctions tests largement utilisés dans la littérature [24].

Au début, nous comparons les performances des algorithmes KBB, αBB et KBB_m sur un ensemble de 10 fonctions. Ici, nous incluons une méthode qui calcule les nombres positifs α et K. Le nombre n des fonctions quadratiques utilisés dans KBB_m à chaque itération est fixé à $n = 16$. La précision fixée à $\varepsilon = 10^{-6}$. Ensuite, nous testons l'algorithme KBB_m en doublant à chaque fois le nombre de sous-estimateurs sur un ensemble de 20 fonctions.

Exp	$f(x)$	$[x^L, x^U]$	LM	GM	opt
1	$e^{-3x} - \sin^3 x$	$[0, 20]$	4	1	-1
2	$\cos x - \sin(5x) + 1$	$[0.2, 7]$	6	1	-0.952897
3	$x + \sin(5x)$	$[0.2, 7]$	7	1	-0.077590
4	$e^{-x} - \sin(2\pi x)$	$[0.2, 7]$	7	1	-0.478362
5	$\ln(3x)\ln(2x) - 0.1$	$[0.2, 7]$	1	1	-0.141100
6	$\sqrt{x}\sin^2 x$	$[0.2, 7]$	3	2	0
7	$2\sin x \exp(-x)$	$[0.2, 7]$	2	1	-0.027864
8	$2\cos x + \cos(2x) + 5$	$[0.2, 7]$	3	2	3.5
9	$\sin x$	$[0, 20]$	4	3	-1
10	$\sin x \cos x - 1.5\sin^2 x + 1.2$	$[0.2, 7]$	3	2	-0.451388
11	$(x - x^2)^2 + (x - 1)^2$	$[-10, 10]$	1	1	0
12	$\frac{x^2}{20} - \cos x + 2$	$[-20, 20]$	7	1	1
13	$x^2 - \cos(18x)$	$[-5, 5]$	29	1	-1
14	e^{x^2}	$[-10, 10]$	1	1	1
15	$(x + \sin x)\exp(-x^2)$	$[-10, 10]$	1	1	-0.824239
16	$x^4 - 12x^3 + 47x^2 - 60x - 20\exp(-x)$	$[-1, 7]$	1	1	-32.78126
17	$x^6 - 15x^4 + 27x^2 + 250$	$[-4, 4]$	2	2	7
18	$x^4 - 10x^3 + 35x^2 - 50x + 24$	$[-10, 20]$	2	2	-1
19	$24x^4 - 142x^3 + 303x^2 - 276x + 3$	$[0, 3]$	2	1	-89
20	$\cos x + 2\cos(2x)\exp\ (-x)$	$[0.2, 7]$	2	1	-0.918397

Tableau 1 : Fonctions tests

Exp		αBB			
	T_{cpu}	m	m_e	LB_0	f^*
1	0	47	24	−273.76041	−0.99999
2	0	17	9	−65.9109	−0.95203
3	0	31	16	−118.96147	−0.07759
4	0	15	8	−121.60896	−0.47797
5	0	11	6	−527.67986	−0.14099
6	0	13	7	−2733.29510	0.00199
7	0	13	7	−18.57601	−0.02761
8	0	11	6	−28.84495	3.56245
9	0	13	7	−46.40909	−0.99997
10	0	17	9	−29.62761	−0.45138

Tableau 2 : Les résultats des tests obtenus pour les 10 fonctions par αBB.

Exp		KBB			
	T_{cpu}	m	m_e	LB_0	f^*
1	2.211	55	28	−564.01754	−1
2	10.645	25	13	−116.08120	−0.95289
3	4.604	25	13	−117.80163	−0.07758
4	3.576	57	29	−121.20354	−0.47834
5	3.312	19	10	−528.13263	−0.14110
6	2.854	29	15	−6664.14641	0
7	3.132	67	34	−5.50269	−0.02786
8	3.012	21	11	−14.03655	3.5
9	2.293	15	8	−45.19193	−1
10	3.460	27	14	−29.66644	−0.45139

Tableau 3 : Les résultats des tests obtenus pour les 10 fonctions par KBB

Exp		$KBBm$			
	T_{cpu}	m	m_e	LB_0	f^*
1	0	144	136	-2.26691	-1
2	0	32	31	-0.97784	-0.9529
3	0	48	46	-0.09467	-0.07759
4	0	176	166	-0.65053	-0.47820
5	0	48	46	-1.73375	-0.14110
6	0	48	46	-0.23383	0
7	0	112	106	-0.04618	-0.02786
8	0	48	46	3.49276	3.50001
9	0	64	61	-1.00563	-1
10	0	64	61	-0.45957	-0.45139

tableau 4 : Les résultats des tests obtenus pour les 10 fonctions par KBB_m avec $n = 16$

n	2	4	8	16	32	64	128
1	-152.72	-45.16	-7.15	-2.26	$*$	$*$	$*$
2	-28.43	-7.92	-2.16	-0.97	$*$	$*$	$*$
3	-28.45	-6.17	-1.34	-0.094	$*$	$*$	$*$
4	-30.018	-8.85	-2.207	-0.65	-0.49	$*$	$*$
5	-121.3	-29.66	-7.17	-1.73	-0.40	-0.149	$-0..$
6	-448.19	-41.56	-3.542	-0.23	-0.0019	$-0..$	$-0..$
7	-1.307	-0.340	-0.104	-0.04	-0.03	-0.02	$-0..$
8	0.33	2.54	3.394	3.49	$*$	$*$	$*$
9	-11.23	-3.751	-1.141	-1.005	$*$	$*$	$*$
10	-5.85	-1.98	-0.598	-0.459	-0.453	-0.452	$-0..$
11	-16118.1	-1297.05	-107.4	-9.34	-0.67	-0.09	$-0..$
12	1	$*$	$*$	$*$	$*$	$*$	$*$
13	-20.42	-2.2	$*$	$*$	$*$	$*$	$*$
14	$*$	$*$	$*$	$*$	$*$	$*$	$*$
15	-173493.6	-27703.7	-7855.9	-769.2	-575.3	-48.21	-46.4
16	-19351.54	-576.321	-45.152	-33.67	-32.84	-32.80	$-32..$
17	-14875.91	-2957.18	-362.63	-21.88	4.71	6.82	6.98
18	-93572.1	-9016.69	-1032.09	-142.7	-27.31	-7.07	-2.4
19	-578.6	-141.98	-95.79	-89.8	-89.1	-89.01	-89.001
20	-6.906	-2.59	$*$	$*$	$*$	$*$	$*$

Tableau 5 : les valeurs de LB_0 obtenue par KBB_m pour les 20 fonctions

Pour déterminer la borne inférieure, αBB utilise une méthode locale à chaque itération, ce qui rend le processus plus coûteux que dans la méthode KBB dans laquelle les bornes inférieures sont données explicitement. Donc, notre comparaison ne sera pas basée sur le nombre d'itérations nécessaires pour obtenir l'optimum.

Le temps d'exécution nécessaire pour atteindre la valeur optimale est un critère fiable et significatif pour mesurer les performances de l'algorithme en question. Selon les résultats numériques résumés, dans les tableaux (3) et (4), les performances de la méthode proposée sont nettement mieux que celles de la méthode KBB.

Les tableaux (2),(3) et (4) montrent que la méthode proposée KBB_m élimine environ 95% des intervalles tandis que KBB et αBB éliminernt 51%. En preuvant la capacité considérable d'exclure les intervalles qui ne peuvent pas contenir la solution optimale, ce qui accélère la convergence de la méthode en question.

La meilleure borne inférieure initiale obtenue est un critère important pour mesurer le resserrement d'un sous-estimateur. Dans ce contexte, le tableau (5) fait l'objet d'une étude qualitative de la borne inférieure initiale obtenue par notre algorithme KBB_m montre qu'il est compétitif notamment quand on retrouve directement l'optimum connu f^* en doublant le nombre de sous-estimateurs de l'objectif.

3.6 Conclusion

Dans ce chapitre, nous avons proposé une technique de sous estimation de l'objectif, les résultats numériques montrent que notre méthode possède une capacité importante d'exclure les solutions partielles qui ne mènent pas à la solution que l'on recherche. Ce qui accélère la convergence de la méthode en question. Le doublement du nombre de quadratiques minorants la fonction objectif nous permet de se rapprocher de cette dernière autant que l'on veut. Et aussi, améliore la qualité des bornes inférieures et supérieures. Brièvement, notre méthode rassemble et améliore les avantages des deux méthodes font l'objet de comparaison.

Chapitre 4

Couplage d'Aliénor avec Séparation et Évaluation

4.1 Introduction

Les progrès des sciences ont toujours été réalisés grâce à la modélisation qui consiste à représenter les phénomènes issus du réel par des équations mathématiques. L'objectif de la modélisation ne consiste pas uniquement à représenter un phénomène donné par un ensemble d'équations. La plupart du temps le modélisateur a aussi, en entière pensée, l'idée que l'on pourra ainsi agir sur le phénomène de façon à l'optimiser au sens de certains critères (coût, rendement, production, qualité, etc.). À partir du moment où l'on dispose d'un modèle fiable du système, étudier, optimiser son fonctionnement va conduire à ce que l'on appelle des problèmes de contrôle. Ces problèmes particulièrement difficiles entrent dans le cadre des problèmes d'optimisation. Autrement dit, on va devoir chercher des extrema (généralement des minima) de fonctions de plusieurs variables. Bien entendu, si l'on veut optimiser au mieux le système, ce sont les optima absolus qui nous intéresseront au premier chef d'où l'intérêt de l'optimisation globale ayant pour objectif l'obtention de minima ou de maxima absolus.

La littérature sur l'optimisation décrite pour l'essentiel des méthodes locales qui permettent l'obtention de minima locaux qui peuvent être certes intéressants dans une étude préalable du phénomène mais qu'il faudra de toute façon améliorer pour optimiser au mieux le phénomène modélisé.

Quant aux méthodes globales existantes elles sont la plupart du temps des améliorations des méthodes locales que l'on tente de débloquer d'un minimum local pour aller vers un meilleur minimum, espérant qu'in fine on obtiendra un minimum absolu. Contrairement à la transformation réductrice Aliénor [15]. Cette transformation permet de ramener une fonction à n variables à une fonction d'une seule variable qui conserverait les minimiseurs globaux au moins de façon approchée. La technique utilisée est basée sur la réduction de la dimension en utilisant des "courbes remplissant l'espace". Ces courbes ont l'avantage d'être de classe C^∞ et préservent les propriétés de la fonction objectif. La méthode Aliénor s'est révélée être d'une grande efficacité en s'associant à certaines méthodes unidimensionnelles telles que la méthode présentée au chapitre précédent. Le couplage d'Aliénor avec cet algorithme sera appliqué sur une classe de fonctions à plusieurs variables ayant un minimum global peut-être trouvé théoriquement [19]. Nous étudions dans ce chapitre le couplage de la méthode d'Aliénor avec la méthode de Séparation et Évaluation proposée dans le chapitre précédent.

4.2 La méthode Aliénor

Le principe fondamental de la méthode de la transformation réductrice [15] consiste à effectuer une transformation qui permet de ramener le problème multidimensionnel à un problème unidimensionnel afin d'appliquer les méthodes d'optimisation plus efficaces adaptées au cas d'une seule variable. L'idée de base consiste à densifier l'ensemble faisable par une courbe paramétrée ($\alpha-$dense) continue et assez régulière. La fonction multivariable est transformée en une fonction d'une seule variable. Ainsi notre problème est ramené à un problème plus facile à résoudre car il y a une seule direction à explorer.

La première transformation réductrice proposée par les inventeurs de cette méthode utilise la spirale d'Archimède. Nous en donnerons ci-après une brève description.

La spirale d'Archimède

Soit $(x, y) \in \mathbb{R}^2$. En coordonnées polaires ce point s'écrit :

$$x = \rho \cos\theta, \ y = \rho \sin\theta. \tag{4.1}$$

On peut alors relier ρ et θ grâce à la spirale d'Archimède d'équation : $\rho = a\theta$, $\theta \geq 0$ est un paramètre positif fixé (destiné à tendre vers zéro). Nous obtenons alors :

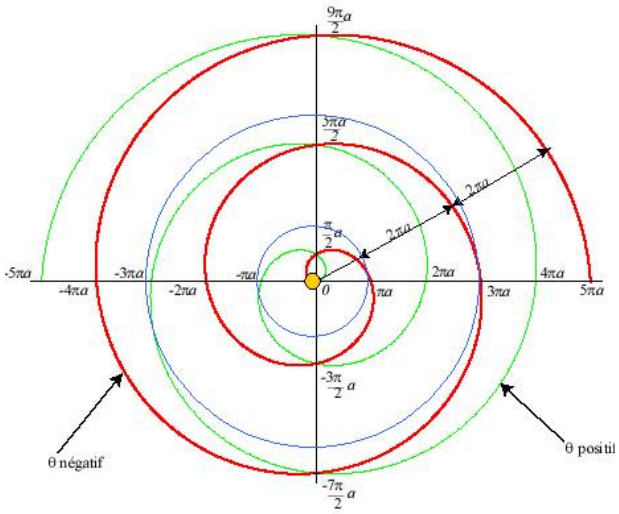
$$x = a\theta \cos\theta = h_1(\theta), \ y = a\theta \sin\theta = h_2(\theta).$$

Nous avons donc exprimé $(x, y) \in \mathbb{R}^2$ en fonction d'une seule variable $\theta \in \mathbb{R}$. Le paramètre a est destiné à tendre vers zéro afin de permettre d'approcher tout point du plan $\mathbb{R}^2$ par un point de la courbe

$$h(\theta) = (h_1(\theta), h_2(\theta))$$

Pour trois variables x_1, x_2, x_3, on relie d'abord x_1 et x_2 à l'aide d'une spirale d'angle θ_1 ce qui donne :

$$x_1 = a\theta_1 \cos\theta_1 \text{ et } x_2 = a\theta_1 \sin\theta_1$$



La spirale d'Archimède

Puis on relie θ_1 et x_3 à l'aide d'une spirale d'angle θ en posant :

$$\theta_1 = a\theta\cos\theta \text{ et } x_3 = a\theta\sin\theta.$$

Ainsi, on obtient la courbe paramétrée $h(\theta) = (h_1(\theta), h_2(\theta), h_3(\theta))$ définie par :

$$\begin{cases} h_1(\theta) = a_2\theta\cos\theta\cos(a\theta\cos\theta) \\ h_2(\theta) = a_2\theta\cos\theta\sin(a\theta\cos\theta) \\ h_3(\theta) = a\theta\sin\theta \end{cases}$$

Nous pourrions généraliser ce procédé à n variables x_1, x_2, ..., x_n en les reliant deux à deux par des spirales d'angle θ_i. À la fin du processus on obtient la variable θ qui permet d'exprimer tous les x_i :

$$x_i = h_i(\theta),\ i = 1, ..., n$$

Où les $h_i(\theta)$ sont des fonctions explicites de θ (de classe C^∞). Ainsi nous avons "approché" l'espace $\mathbb{R}^n$ par $\mathbb{R}$ grâce à une transformation réductrice utilisant la spirale

d'Archimède.

La précision de cette approximation dépend du coefficient a : plus a est petit et meilleur est la précision. Pour préciser tout cela, nous allons définir l'alpha-densité :

Définition 7

Un sous espace $S \subset \mathbb{R}^n$ est dit α-dense dans $\mathbb{R}^n$ si :

$$\forall M \in \mathbb{R}^n,\ \exists M' \in S : d(M, M\prime) \leq \alpha$$

Remarque 9

La spirale d'Archimède $\rho = a\theta$ est πa-dense dans $\mathbb{R}^2$.

Le théorème suivant montre un résultat très important :

Théorème 12 ***[15][14][57]***

Tout point de $\mathbb{R}^n$ peut être approché, avec la précision que l'on souhaite, par au moins un point de la transformation Aliénor.

Autres transformations réductrices

Dans cette méthode développé par B.Konfé et al [35], on utilise la transformation réductrice suivante :

$$h_i(\theta) = \cos(\omega_i\theta + \varphi_i), i = 1, ..., n.\ \ \theta \geq 0.$$

Où les (φ_i) et (ω_i) sont deux suites choisies lentement croissantes et on a le théorème suivant ;

Théorème 13 ***[15]***

La transformation $h(\theta) = (h_1(\theta), ..., h_n(\theta))$ est alpha dense :

• *avec des $\omega_i > 0$, croissante*

• *avec $\varphi_i > 0$ croissante et proche les uns des autres, est $\alpha-$dense et son coefficient de densification vaut :*

$$\alpha = 2k\sqrt{n-1}\frac{\omega_{n-1}}{\omega_n},\ k > 0. \tag{4.2}$$

4.3 La méthode mixte Aliénor avec Séparation et Évaluation

Soit f une fonction non convexe deux fois différentiable et définie sur un pavé $X = \prod_{i=1}^{n}[a_i, b_i]$ de $\mathbb{R}^n$ à valeurs dans $\mathbb{R}$.
Considérons le problème de minimisation suivant :

$$(P)\begin{cases} \min f(x_1, ..., x_n) \\ (x_1, ..., x_n) \in X \end{cases}, \tag{4.3}$$

que l'on veut résoudre avec une précision exigée $\varepsilon > 0$.

Nous allons maintenant appliquer la méthode Aliénor à l'optimisation globale.

On construit alors une courbe paramétrée

$$h(\theta) = (h_1(\theta), h_2(\theta), ..., h_n(\theta)), \tag{4.4}$$

α-dense dans $\prod_{i=1}^{n}[a_i, b_i]$ telle que :

$$x_i = h_i(\theta), i = 1, ..., n, \tag{4.5}$$

le problème (4.3) devient alors :

$$(P_t)\begin{cases} \min f^*(\theta) \\ \theta \in [0, \theta_{\max}] = \top \end{cases}, \text{ où} \tag{4.6}$$

$$f^*(\theta) = f(h_1(\theta), ..., h_n(\theta)) \tag{4.7}$$

θ_{max} est la plus grande valeur que peut prendre θ quand $(x_1, x_2, ..., x_n)$ décrivent l'ensemble $\prod_{i=1}^{n}[a_i, b_i]$. Le problème (4.6) est un problème de minimisation d'une fonction à une seule variable.

Voici à présent un résultat fondamental dont la démonstration est donnée en détail dans [15],[14]

Théorème 14

Tout minimiseur global du problème (4.3) peut-être approché par un minimiseur de (4.6).

Il est important de remarquer que tout minimum local de f^* n'est pas forcément une approximation d'un minimum de f car la fonction f^* est la restriction de f à la spirale généralisée $h(\theta)$. Aussi, les minima locaux de $\top$ ne sont pas obligatoirement une approximation des minima locaux de X car $\top$ est strictement contenu dans X. Par contre tout minimiseur absolu de f^* est bien une approximation d'un minimiseur absolu de f.

On applique, maintenant l'algorithme de Branch and Bound proposé dans le chapitre précédent.

Algorithme 5 *(Multivariable)*

Entrée

- n la dimension
- f fonction objectif
- $[a_i, b_i]$ intervalles des réelles
- N nombre de quadratiques utilisées (sous-estmateurs)
- ε une précision donnée.

Sorties :

- x^* minimum global de f

Transformation :

- poser $x_i = h_i(\theta)$, où $h_i(\theta) = \frac{1}{2}\left[(b_i - a_i)\cos(\omega_i\theta + \phi_i) + b_i + a_i\right],\ i = 0, ..., n-1.$

avec $\theta \geq 0,\ \theta \in [0, 2\pi]$

1. **Initialisation** $k = 0$

   (a) **pour tout** $i = 0, ..., n$ Calculer $\theta_i = a + \frac{b-a}{n}i,$ et soit $M = \cup_{i=0}^{n-1}\{[\theta_i, \theta_{i+1}]\}$

   (b) Calcule K_i tel que $|f^{*\prime\prime}(\theta)| \leq K_i$ pour chaque $[\theta_i, \theta_{i+1}]$ **pour tout** $i = 0, ..., n-1.$

   (c) Calculer θ_i^* en utilisant **(3.10) pour tout** $i = 0, ..., n-1.$

   (d) Calculer $UB^k = \min\{\min f(\theta_i^*),\ \min f(\theta_i)$

   (e) Soit $LB^k = \min LB_i^k$ avec $LB_i^k = p_i(\theta_i^*)$

   $\bar{i}$: l'indice correspondant au $\min LB_i^k$

2. **Itération**

   Tant que $(UB^k - LB^k > \varepsilon$ et $M \neq \emptyset)$ **faire**

   (a) $a := \theta_{\bar{i}}$,$b := \theta_{\bar{i}+1}$ et **appliquer les étapes** a,b,c et d de (1)

   (b) Mettre à jour UB^k

   (c) **Pour tout** $i = 1, ..., m;\ (m = card\ (M))$

   - **Elimination : si** $(UB^k - LB_i^k < \varepsilon)$ **alors** supprimer $[\theta_i, \theta_{i+1}]$ de M
   - **Sélection : si**$(UB^k - LB_i^k \geq \varepsilon)$ **alors** sélectionner

   $\bar{i}$:l'indice correspondant au $\min_i LB_i^k$

1. d. $k = k + 1$

   fin Tant que

3. $\theta^* = \theta^k$ la solution optimale correspondante au meilleur UB^k trouvé.

4. **Retour :**

Trouver $x_i^* = \frac{1}{2}\left[(b_i - a_i)\cos(\omega_i\theta^* + \varphi_i) + (b_i + a_i)\right], i = 0, ..., n-1.$

fin algorithme

Théorème 15

La méthode mixte Aliénor-Séparation et Évaluation appliquée au problème (P) *converge en un nombre fini d'itérations vers le minimum global avec une précision inférieure ou égale à* ε.

Pour mesurer l'efficacité de ce couplage nous allons le comparer avec la méthode proposée par Ouanes et al [47].

4.4 Étude numérique

Caractéristiques des fonctions test : dans ces fonctions tests d'optimisation globale [19] les dérivées partielles premières par rapport à toutes les variables ont les mêmes expressions.

Par exemple, considérons la fonction Rastrigin donnée par :

$$F : [-5.12, 5.12]^n \to \mathbb{R}$$

$$F(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i)). \tag{4.8}$$

Pour lesquels la solution optimale globale est $(0, 0, ...0)$.

Dans le cas bidimensionnel, la fonction Rastrigin est représentée comme suit :

$$F(x) = 20 + x^2 - 10\cos(2\pi x) + y^2 - 10\cos(2\pi y).$$

Les premières dérivées partielles sont :

$$\begin{aligned} f(x) &= \frac{\partial}{\partial x}F(x,y) = 2x + 20\pi\sin(2\pi x) \\ f(y) &= \frac{\partial}{\partial y}F(x,y) = 2y + 20\pi\sin(2\pi y) \end{aligned} \tag{4.9}$$

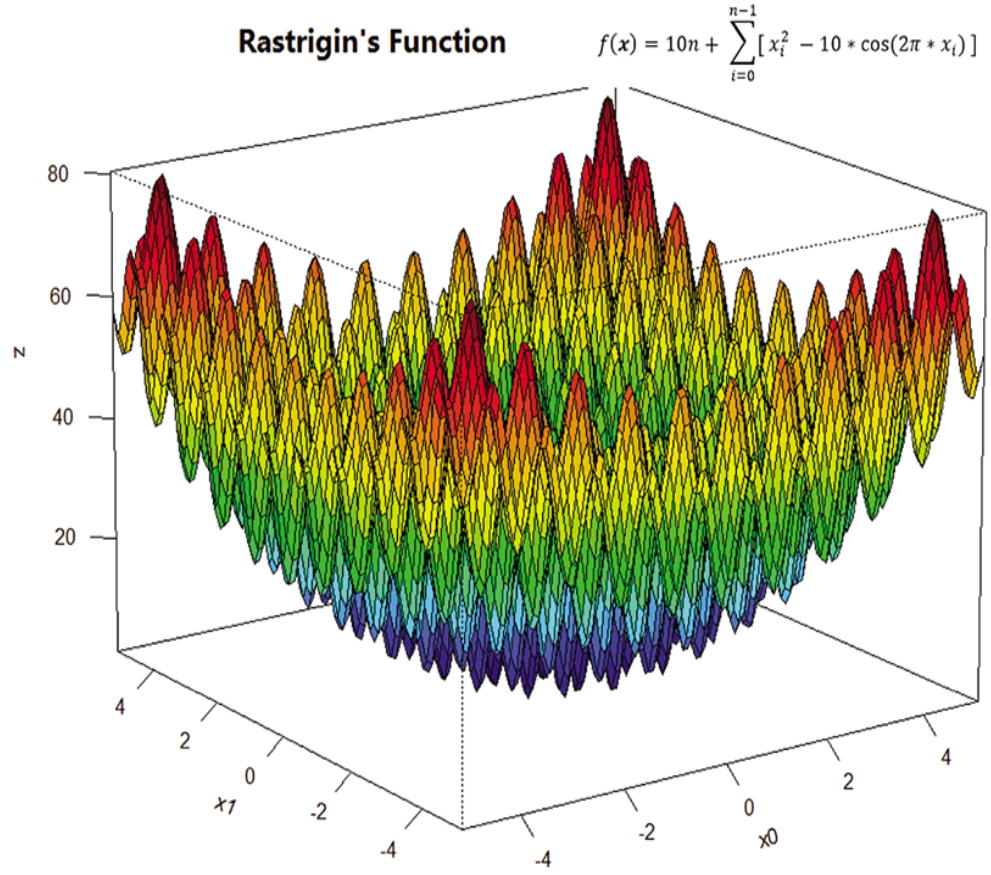
À partir des équations ci-dessus, il est évident que les deux dérivés ont la même expression qui est donnée par ;

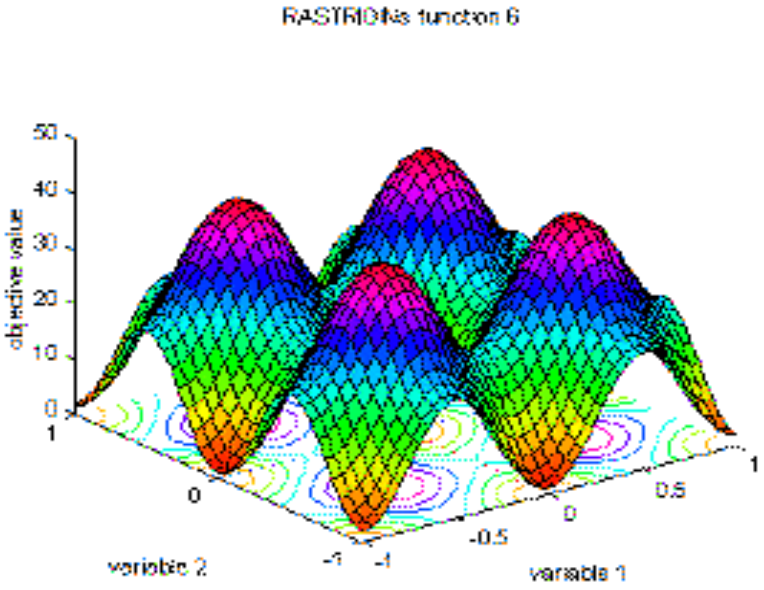
$$f(t) = 2t + 20\pi\sin(2\pi t)$$

Et on a le résultat suivant :

Théorème 16 ***[17]***

Soit $F : Q \to \mathbb{R}$ une fonction continue deux fois différentiable, où Q est une partie bornée et fermée de $\mathbb{R}^n$. Si les dérivés partielles $f : Q \to \mathbb{R}^n$ ont la même équation par rapport à l'ensemble des variables, F prend alors son optimum global (maximum ou minimum) en un point où tous les symboles ont la même valeur et où $g(t) = \int f(t)dt$ prend également son optimum global (maximum ou minimum).

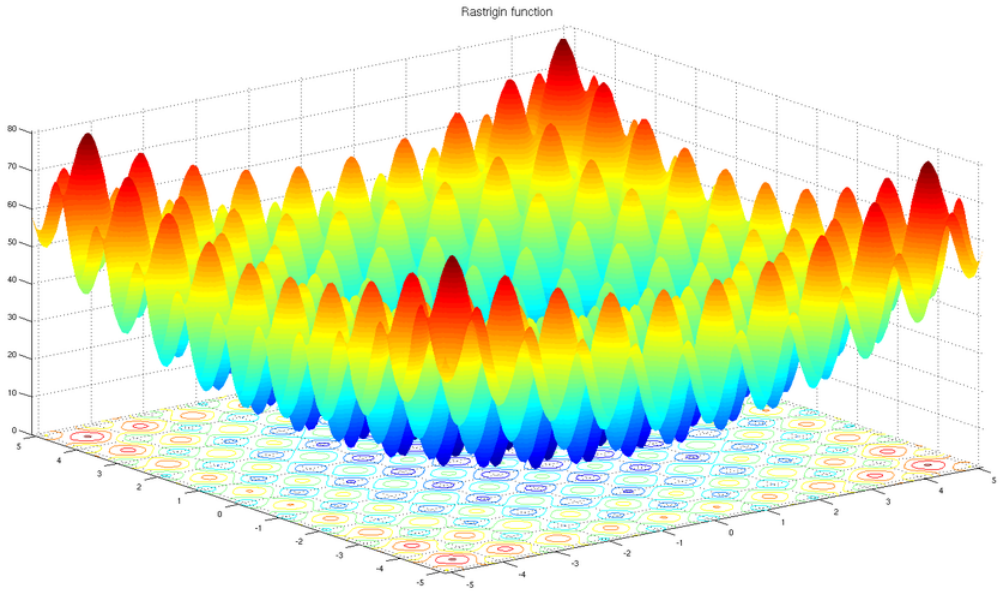

Rastrigin's Function
$f(x) = 10n + \sum_{i=0}^{n-1} [\, x_i^2 - 10 * \cos(2\pi * x_i) \,]$
80
60
40
20
z
4
2
0
-2
-4
x1
-4
-2
0
2
4
x0



Exp	Alg 1		
	$[a_i, b_i]$	f^*	T_{cpu}
1	$[-5.12, 5.12]$	0.000000	2.348
2	$[-5.12, 6.12]$	0.000000	2.749
3	$[-3.14, 2]$	0.000000	1.521
4	$[-3.14, 2.5]$	0.000000	1.221
5	$[-10, 10]$	0.000000	3.049
6	$[-20, 20]$	0.000000	1.815
7	$[-0.5, 1]$	0.000000	1.399
8	$[-1, 1]$	0.000000	1.509
9	$[-3, 9]$	0.000000	1.493
10	$[-0.02, 7]$	0.000000	1.191

(4.10)

Tableau 1 : Les résultats des tests obtenus par la méthode présenté dans [47] désigné par Alg1



Exp	Alg 2		
	$[a_i, b_i]$	f^*	T_{cpu}
1	$[-5.12, 5.12]$	0.000000	0
2	$[-5.12, 6.12]$	0.000000	0
3	$[-3.14, 2]$	0.000000	0
4	$[-3.14, 2.5]$	0.000000	0
5	$[-10, 10]$	0.000000	0
6	$[-20, 20]$	0.000000	0
7	$[-0.5, 1]$	0.000000	0
8	$[-1, 1]$	0.000000	0
9	$[-3, 9]$	0.000000	0
10	$[-0.02, 7]$	0.000000	0

Tableau 2 : Les résultats des tests obtenus par la méthode proposée désigné par Alg 2.

– Les deux tableaux montrent clairement, que notre approche est meilleure en terme du temps d'exécution. Les résultats sont encourageants et prometteurs.

4.5 Conclusion

Le couplage de la méthode Aliénor et de la méthode Séparation et Évaluation s'est avéré très performant pour résoudre les problèmes d'optimisation globale à variables bornées. L'utilisation de la méthode Aliénor permet de remplacer la fonction objectif à plusieurs variables par une fonction à une seule variable. On obtient alors un problème d'optimisation unidimensionnelle facile à résoudre. Les résultats numériques confirment l'intérêt de ce couplage.

Conclusion générale

Notre travail a abouti à la résolution de tous les cas du $PT4C$ non traités auparavant du point de vue mathématique. Ainsi, nous avons proposé un nouvel outil pour l'optimisation des problèmes de transport des marchandises avec capacités des chemins. nous avons réalisé un programme en langage C, le temps d'exécution est assez rapide pour les moyennes tailles.

Dans le deuxième chapitre, nous avons présenté un nouveau type de technique de génération de bases fiable et robuste afin de surmonter les difficultés de la dégénérescence. Cette méthode constitue de manière certaine une base. Elle est très peu consommatrice en temps de calcul et permet d'améliorer les solutions courantes. L'efficacité de cette technique a été validée sur 40 problèmes tests générés aléatoirement.

Dans le troisième chapitre, nous avons présentés une méthode de sous-estimateurs de la fopnction objectif non convexe basé sur des fonctions quadratiques en morceaux qui possèdent des minima explicites. Une comparaison est effectuée sur les bornes inférieures favorise une telle quadratique par rapport aux autres en garantissant la sous-estimation de l'objectif.

Cette approche est validée en considérant une une méthode déterministe de Séparation et Évaluation qui est entièrement détaillée permet d'encadrer la valeur du minimum global à la fin de l'exécution.

Toutefois, l'extension de cette technique au cas multidimensionnel nécessite encore des préservations d'avantages déjà requis. Notre travail est achevé par

un couplage de la méthode d'Aliénor et la méthode Branch and Bound, qui permet de réduire la dimension du problème et simplifie la résolution. En prouvant les performances de ce couplage notamment pour les problèmes à deux dimensions. On prévoit tester ce dernier sur des problèmes de tailles encore plus élevées.

Bibliographie

[1] Aaid, D., Zitouni, R., Étude numérique comparative d'algorithmes pour un problème de transport a quatre indices avec capacités, Cima'09, Annaba, Algérie, (2009).

[2] Aaid, D., Étude numérique comparative entre des méthodes de résolution d'un problème de transport à quatre indices avec capacités. Mémoire de Magistère : Mathématique d'application, Université de Constantine, Algérie (2010).

[3] Aaid, D. , Noui, A., Le Thi, H. A. , Zidna, A., Une Synthèse sur le Problème de Transport a Quatre Indices avec Capacités, COSI'11 Guelma, Algérie, (2011).

[4] Aaid , D.,Noui ,A., Le Thi. H. A., Zidna ,A., A modified classical algorithm AL_{PT4C} for solving a capacitated four-index transportation problem, Acta Mathematica Vietnamica, Vol 37, No 3, pp. 379-390, (2012).

[5] Aaid, D., Noui , A., Ouans, M., New technique for solving univariate global optimization. Archivum Mathematicum, Vol. 53 No. 1, 19–33(2017).

[6] Adjiman, C.S.,Androulakis I.P., Floudas C.A. , A global optimization method ,α BB, for general twice di erentiable NLPs .II. Implementation and computational results. Comput. Chem. Eng. 22(9), 1159-1179 (1998).

[7] Adjiman, C.S., Dallwig, S., Floudas, C.A., Neumaier, A., A global optimization method, α BB, for general twice-differentiable NLPs – I. Theoretical advances. Comput. Chem. Eng. 22(9), 1137–1158 (1998a).

[8] Akrotirianakis, I.G., Floudas, C.A., A new class of improved convex underestimators for twice continuously differentiable constrained NLPs. J. Glob. Optim. 30(4), 367–

390 (2004a).

[9] Akrotirianakis, I.G., Floudas, C.A., Computational experience with a new class of convex underestimators : box-constrained NLP problems. J.Glob. Optim. 29(3), 249–264 (2004b).

[10] Anu ,A., Arora, S. R., Multi index charge bicriterion transportation problem, Indian J. Pure Appl. Math. 32(5) 739-746 (2001).

[11] Bazarra, S., Sherali, H. D., Shetty, C. M., Nonlinear Programming,Theory and algorithms, Second edition (1993).

[12] Caratzoulas, S., Floudas, C.A., A trigonometric convex underestimator for the base functions in Fourier space. J. Optim. Theory Appl. 124(2), 339–362 (2005).

[13] Cenk ,C., A specialized network simplex algorithm for the constrained maximum flow problem, European J. Oper. Res. 210(2) 137-147 (2011).

[14] Cherrault,Y., Modèles et méthodes mathématiques pour les sciences du vivant, Presses Universitaires de France (P.U.F).(1998a).

[15] Cherrault,Y. , Optimisation : méthodes locales et globales, Presses Universitaires de France (P.U.F). (1999).

[16] Cherrault,Y. , Mora, G., Optimisation globale : Théorie des courbes des courbes α-denses, Economica (2005).

[17] Claire ,S. , Adjiman , Christodoulos, A. , Floudas, Rigorous Convex Underestimators for General Twice-Di erentiable Problems. Journal of Global Optimization 9 : 23-40, (1996).

[18] Ciarlet.P.G.,The Finite Element Method for Elliptic Problems Studies in Math.and its Appl., (1979).

[19] Crina, G., Ajith, A., On a Class of Global Optimization Test Functions, www.softcomputing.net/nnw (2009).pdf

[20] Dahiya, K., Verma, V., Capacitated transportation problem with bounds on RIM conditions. European Journal of Operational Research, Vol 178(3), p. 718-737, (2007).

[21] De Boor, C., A Practical Guide to Splines, Applied Mathematical Sciences, Springer Verlag,(1978).

[22] Dubeau, F. , Guèye, O. M., Une nouvelle méthode d'initialisation pour le problème de transport, RAIRO-Oper. Res. (42) 389-400,(2008).

[23] Floudas ,C. A., Gounaris ,C. E., A review of recent advances in global optimization. J Glob Optim DOI 10.1007/s10898-008-9332-8, (2008).

[24] Gounaris, C.E. : Floudas, C.A., Tight convex underestimators for C^2-continuous problems : I. Univariate functions.J. Glob. Optim. 42(1), 51-67 (2008).

[25] Gounaris, C.E., Floudas, C.A., Tight convex underestimators for C^2-continuous problems : II. Multivariate functions. J. Glob. Optim.42(1), 69–89 (2008b).

[26] Gourgand, M., Pham, T.H. , Tanguy, A., Four index transportation problem : principle, resolution and application in industry, 3rd IEEE International Symposium Logistics and Industrial Informatics (LINDI'2011), Budapest, Hungary (2011).

[27] Gourgand, M., Pham ,T.H., Tanguy, A., Mathematical model and industrial application of four index transportation problem. 25th European Simulation and Modelling Conference (ESM'2011), Guimaraes, Portugal (2011).

[28] Hanachi, A. , Étude théorique et numérique d'une méthode de linéarisation de type Karmarkar pour la programmation quadratique convexe, Thèse de Magister, institut de Mathématique, Université Ferhat Abbas, Sétif (1997).

[29] Hertz, D., Adjiman, C.S., Floudas, C.A., Two results on bounding the roots of interval polynomials. Comput. Chem. Eng. 23, 1333 (1999).

[30] Kalpana ,D., Vanita, V., Capacitated transportation problem with bounds on RIM conditions, European J. Oper. Res. (178) 718-737 (2007).

[31] Kebbiche, Z., Étude et extensions d'algorithmes de points intérieurs pour la programmation non linèaire. Thèse de Doctorat de Mathématiques appliquées, Université Ferhat Abbas, Sétif (2007).

[32] Kebbiche, Z. , Keraghel, A. , Yassine, A., Extension of a projective interior point method for linearly constrained convex programming, Applied Mathematics and Computation Vol. 193, Issue 2, 1 553-559 November (2007).

[33] Keraghel, A. , Étude adaptative et comparative des principales variantes dans l'algorithme de Karmarkar}. Thèse de Doctorat de Mathématiques appliquées, Université Joseph Fourier, Grenoble, France (1989).

[34] Keraghel, A., Analyse convexe : Théorie fondamentale et exercices, Éditions Dar El'Houda, Ain Mlila, Algérie (2001).

[35] Konfe, B. O., Cherrault, Y. , Bennouala, T., A global Optimization method for large number of variables (variant of Alienor method), Kybernetes, Vol 34, pp1070-1083 (2005).

[36] Kumar, A. , Kaur ,A., Gupta, A., Fuzzy linear programming approach for solving fuzzy transportation problems with Transshipment, J.Math. Model. Algorithms 10(2) 163-180 (2011).

[37] Le Thi,H.A., Ouanes.M., Convex quadratic underestimation and Branch and Bound for univariate global optimization with one nonconvex constraint, RAIRO Operations Research 40, 285-302 (2006).

[38] Leulmi , A., Une procédure améliorante d'une méthode projective en programmation linèaire. Thèse de Magistère, Université de Skikda (2008).

[39] McShane, K.A., Monma, C.L. , Shanno, D.F., An implementation of a primal dual interior point method for linear programming. ORSA journal on computing.1 :70-83 (1989).

[40] Messine, F., Méthodes d'Optimisation Globale basées sur l'Analyse d'Intervalle pour la R esolution des Probl emes avec Contraintes , LIMA-IRIT-ENSEEIHT-INPT, Domain : Computer Science, Toulouse,(1997).

[41] Meyer, C.A., Floudas, C.A., Convex underestimation of twice continuously differentiable functions by piecewise quadratic perturbation : spline α BB underestimators. J. Glob. Optim. 32, 221–258 (2005b).

[42] Min, H. C.Y., Cheol,P. T.Y.L. , A new global optimization method for univariate constrained twice-differentiable NLP problems. J Glob Optim 39 :79 100 (2007).

[43] Minoux, M., Programmation mathématique : Théorie et algorithmes, Tome I, Dunod , Paris (1983).

[44] Ninh, P.X., n index transportation problem. Mathematical Journal, Vol 7(1), 18-25, Vietnam (1979).

[45] Ninh, P.X., Les problèmes de transport à plusieurs indices. Thèse de Doctorat, Institut Polytechnique de Hanoi, Vietnam (1980).

[46] Nocedal, J., Wright, S. J., Numerical optimization. Springer series in operations research (2006).

[47] Ouanes, M. , Le Thi, H.A. , Zidna, A., New quadratic lower bound for Multivariate functions in global optimization. Mathematics and Computers in Simulation, Vol 109, pp197-211 (2015).

[48] Prilutskii, M. Kh. , Multicriterial Multi-Index Resource Scheduling Problems, J. Comput. Syst. Sci. Int. 46(1) 78-82 (2007).

[49] Puri , S., Puri ,M. C., Max-min sum minimization transportation problem, Ann. Oper. Res. 143(1) 265-275 (2006).

[50] Rafael ,A. , Melo , and Laurence, A., Wolsey, Optimizing production and transportation in a commit-to-delivery business mode, European J. Oper.Res. 203(3) 614-618 (2010).

[51] Ryoo, H.S., Sahinidis, N.V., Analysis of bounds for multilinear functions. J. Glob. Optim. 19, 403–424 (2001).

[52] Sharma, R.R.K., Prasad, S. , Obtaining a good primal solution to the incapacitated transportation problem. European Journal of Operational Research, Vol 144(3), p. 560-564, (2003).

[53] Sherali, H.D., Lee, Y., Kim, Y., Partial convexification cuts for 0-1mixed-integer programs. Eur. J. Oper. Res. 165(3), 625–648 (2005).

[54] Tawarmalani, M., Ahmed, S., Sahinidis, N.V., Product disaggregation in global optimization and relaxations of rational programs. J. Glob. Optim. 3, 281–303 (2002a).

[55] Tawarmalani, M., Ahmed, S., Sahinidis, N.V., Global optimization of 0-1 hyperbolic programs. J. Glob. Optim. 24, 385–416(2002b).

[56] Ye ,Y, Kojima, M., Recovering optimal bais is Karmarkar's polynomial algorithm for linear programming. Mathematical Programming, 39 :3O5-3 17 (1987).

[57] ZIADI, A., Optimisation globale : Contribution à l'étude de la méthode de la transformation réductrice ALIENOR, Thèse de doctorat de l'Université Ferhat Abbas SETIF (2000).

[58] Zitouni, R., Le problème de transport à quatre indices à capacités. Thèse de Magistère : Mathématique d'application, Université d'Oran-Es-Senia, Algérie (1994).

[59] Zitouni ,R. ,Keraghel, A., Resolution of a capacitated transportation problem with four subscripts, Kybernetes 32(9/10) 1450-1463 (2003).

[60] Zitouni, R ., Étude qualitative de modèles de transport et localisation, Thèse de doctorat d'état Soutenue a l'université de Sétif, Algérie (2007).

[61] Zitouni, R., Keraghel ,A., Benterki, D., Elaboration and implantation of an algorithm solving a capacitated four-index transportation problem, Appl. Math. Sci. 1(53) 2643-2657 (2007).