Conception optimale de systèmes mécaniques : Optimisation en variables mixtes.

Pascal Lafon

N° d'ordre : 273

THESE

présentée à

L'Institut National des Sciences Appliquées de Toulouse

pour obtenir le

DOCTORAT de L'I.N.S.A.T.

*Spécialité : **GENIE MECANIQUE***

(arrêté en date du 30 mars 1992)

par

Pascal LAFON

Conception Optimale de Systèmes Mécaniques : Optimisation en variables mixtes.

Soutenue le 15 février 1994 devant le jury composé de :

BOHATIER Claude, Professeur à l'IFMA de Clermont Ferrand	Rapporteur
BOUDET René, Professeur à l'UPS Toulouse	Président du jury
GAY Daniel, Professeur à l'IUT A Toulouse	Examinateur
GUILLOT Jean, Professeur à l'INSA Toulouse	Examinateur
LABORDE Patrick, Professeur à l'UPS Toulouse	Rapporteur
SARTOR Marc, Maître de Conférences à l'INSA Toulouse	Examinateur

Laboratoire de Génie Mécanique de TOULOUSE :
INSA Département de Génie Mécanique Complexe Scientifique de Rangueil 31077 TOULOUSE CEDEX

Remerciements

Monsieur Claude BOHATIER, Professeur à l'IFMA de Clermont Ferrand, a accepté d'examiner mon travail et d'en être rapporteur, qu'il trouve ici l'expression de toute ma reconnaissance.

Monsieur Patrick LABORDE, Professeur à l'UPS de Toulouse, me fait l'honneur d'être rapporteur de mon travail, je lui adresse tous mes sincères remerciements.

Monsieur René BOUDET, Professeur à l'UPS de Toulouse, et Monsieur Daniel GAY, Professeur à l'IUT A de Toulouse et responsable du Laboratoire de Génie Mécanique, ont bien voulu participer à ce jury, je les remercie de l'intérêt qu'ils portent à mon travail.

Je remercie également Monsieur Marc SARTOR, Maître de Conférences à l'INSA de Toulouse, de sa présence dans ce jury.

Monsieur Jean GUILLOT, Professeur à l'INSA de Toulouse, m'a chaleureusement accueilli au sein de son équipe. Ses conseils éclairés, sa gentillesse et sa disponibilité m'ont permis d'achever ce travail dans les meilleures conditions, qu'il veuille bien accepter de sincères et respectueux remerciements.

Les membres du personnel du département de Génie Mécanique de l'INSA ont fait preuve à mon égard de beaucoup de gentillesse et de disponibilité, qu'ils trouvent ici l'expression de toute ma gratitude.

Je remercie tous mes collègues du Laboratoire de Génie Mécanique de Toulouse, qui m'ont encouragé tout au long de ce travail. Je leur souhaite à mon tour beaucoup de courage et les assure de toute mon amitié.

Mes amis m'ont beaucoup aidé dans les derniers instants, et ont accepté de relire avec attention et compétence ce mémoire. Je leur suis très reconnaissant, et je les remercie chaleureusement.

TABLE DES MATIERES

Seconde partie :
Méthode de résolution des problèmes d'optimisation

Chapitre 3 : Méthodes pour variables continues

Chapitre 4 : Analyse monotone

Chapitre 5 : Méthodes pour variables mixtes

Troisième partie : Résolution des problèmes de conception optimale : Implémentation d'une méthode

Chapitre 6 : Implémentation d'une méthode

ANNEXES

Annexe 1 : Notations

Annexe 2 : Méthodes de directions conjuguées

Annexe 3 : Méthodes quasi newtoniennes

Annexe 4 : Formulation des problèmes d'optimisation

Introduction

Les performances des logiciels de Conception Assistée par Ordinateur (CAO) ont atteint aujourd'hui un niveau tout à fait remarquable. Ces progrès spectaculaires sont sans aucun doute dus à la disponibilité d'une puissance de calcul sans cesse croissante pour des coûts toujours plus faibles permettant d'exploiter efficacement des langages de programmation avancés (langages de programmation orientés objets et de type "experts").

Les logiciels actuellement disponibles sur le marché intègrent, sous une même interface utilisateur très conviviale, plusieurs modules. Ils sont généralement composés d'un modeleur géométrique permettant de concevoir les volumes les plus complexes et de les représenter à l'écran de façon très réaliste (représentation 3D volumique avec effets d'éclairages). La puissance graphique des stations de travail actuelles permet en outre des manipulations dynamiques très performantes de ces représentations. La plupart de ces logiciels comporte aussi un module de calcul par éléments finis associé à un mailleur automatique et également un module de simulation d'usinage sur machines à commande numérique. La communication des informations relatives au travail en cours entre ces différents modules est assurée de manière totalement transparente pour l'utilisateur.

Dans ces logiciels de CAO, un système mécanique est représenté comme l'ensemble des volumes des différentes pièces qui le composent. Ce mode de représentation ne permet pas à l'utilisateur de concevoir facilement un système mécanique. En effet un certain nombre de notions fondamentales sont encore absentes des logiciels de CAO actuels. Un système mécanique n'est pas uniquement un assemblage de pièces décrites par un volume mais aussi l'assemblage des solutions technologiques des différentes liaisons qui composent ce mécanisme.

Les concepts de solution technologique d'une liaison et également de décomposition fonctionnelle d'un système mécanique, permettent d'exprimer le problème de conception d'un système mécanique sous forme d'un problème d'optimisation.

L'objet de ce travail est de caractériser ce problème d'optimisation et de mettre au point une méthode de résolution adaptée.

Dans la première partie de ce mémoire, nous définirons le concept de décomposition fonctionnelle d'un mécanisme et de solution technologique d'une liaison pour ensuite montrer comment un problème de conception, peut s'exprimer sous la forme d'un problème d'optimisation. La démarche permettant d'obtenir une expression mathématique de ce problème sera ensuite détaillée et illustrée par un exemple concret. Nous conclurons cette première partie en précisant les particularités de ce type de problèmes d'optimisation.

La seconde partie est consacrée à une étude bibliographique des méthodes de résolution disponibles. Cette présentation détaillée nous permettra d'exposer des résultats fondamentaux en optimisation non linéaire comme les conditions d'optimalité de *Kuhn et Tucker* et la notion de dualité. Une partie de cette étude bibliographique concernera également une classe particulière de problèmes d'optimisation : les problèmes monotones. Les difficultés introduites par l'existence de variables discrètes feront l'objet du dernier chapitre de cette seconde partie.

Cette étude bibliographique nous permettra de justifier les choix réalisés dans la mise au point de la méthode de résolution que nous proposons. La dernière partie, concerne la description de cette méthode et de son implémentation informatique. Enfin un exemple nous permettra de valider le processus de conception et le logiciel d'optimisation que nous avons réalisé.

Optimisation
des
systèmes mécaniques

Chapitre 1

Conception des systèmes mécaniques

1 Introduction

La création d'un produit ou d'un système mécanique nécessite un ensemble d'étapes allant de la mise en place du cahier des charges jusqu'à la destruction de ce produit. Un grand nombre de personnes interviennent dans cette chaîne : Les hommes du marketing qui définissent ce cahier des charges, l'ingénieur d'étude qui va bâtir une solution technique, le dessinateur qui va la représenter, le designer qui va définir des formes agréables à l'œil, l'ingénieur de calcul qui va dimensionner les éléments garantissant un certain comportement en service ou une durée de vie du produit, le bureau des méthodes qui va choisir un procédé d'obtention et étudier les gammes de fabrication, les hommes de l'atelier qui vont réaliser le prototype, l'équipe des essais qui acceptera ou refusera le produit après vérification de son adéquation au cahier des charges et enfin l'équipe de maintenance qui suivra le produit en service.

L'intégration de l'outil informatique dans ce processus de création nécessite une analyse précise de la démarche de création. De nombreuses études couvrent le sujet et il n'est pas dans notre propos de faire ici la synthèse de ces analyses. Nous nous intéresserons aux étapes de l'avant projet et du projet, c'est à dire la phase de conception proprement dite. Si nous voulions définir en quelques mots cette phase de conception nous citerions la phrase suivante extraite de [28]:

> *"Concevoir un objet technique, c'est chercher à définir la meilleure solution technologique du moment compte tenu des différentes contraintes. Pour le concepteur, pratiquement toujours soumis à des objectifs contradictoires, c'est la recherche (jamais atteinte) du meilleur compromis."*

Ce processus de conception est, comme le souligne *J. GUILLOT* [28], d'une grande complexité, dont les principales causes sont :

- Le très grand nombre de facteurs à prendre en compte.
- L'interdépendance très forte des choix à réaliser.
- Le caractère flou de certaines conditions de définition.

Cette complexité ne permet pas d'aborder globalement la conception d'un objet technique complexe. Une démarche naturelle consiste alors à le décomposer en systèmes élémentaires simples, parfaitement définis, qui font l'objet de la connaissance technologique. On notera que cette décomposition n'est accessible que lorsque le type de l'objet à concevoir est connu.

2 Processus de conception d'un objet technique de type connu

2.1 Structuration par niveaux

S'appuyant sur cette décomposition, le processus de conception peut s'imaginer comme une définition fonctionnelle globale descendante, allant des caractéristiques les plus générales, aux caractéristiques particulières des éléments constitutifs, puis par une définition technologique remontante, en partant des solutions technologiques de base à la machine.

La figure 1.1 donne une représentation de ce processus de conception dans le cas d'un exemple particulier.

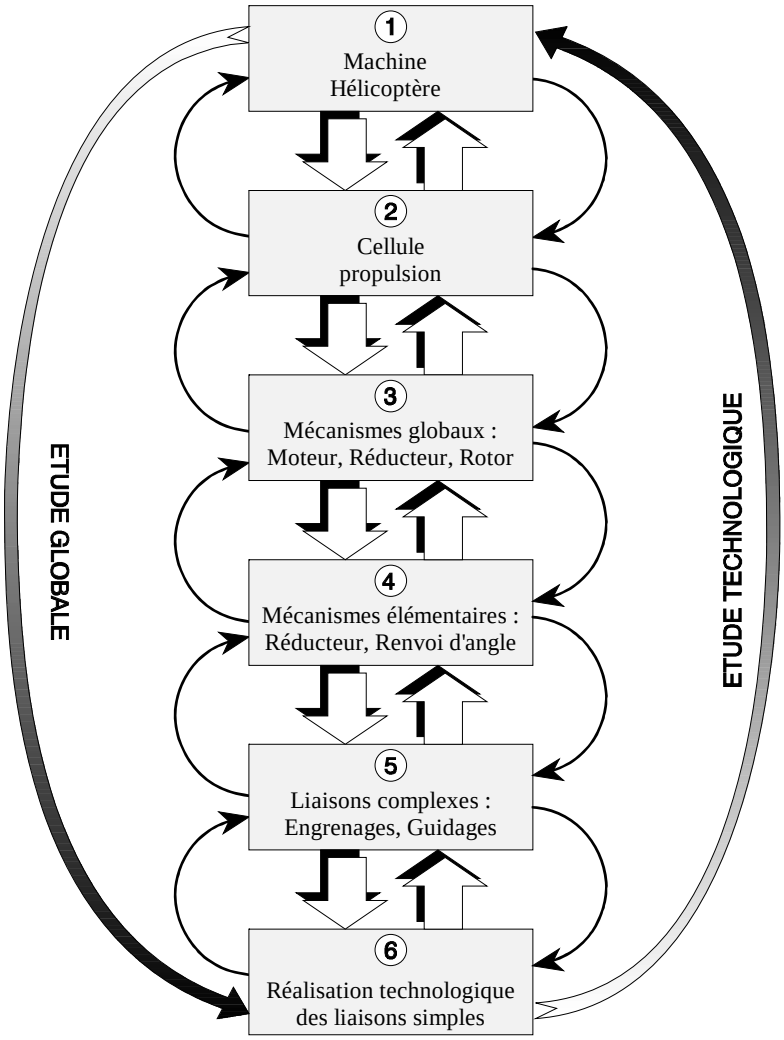


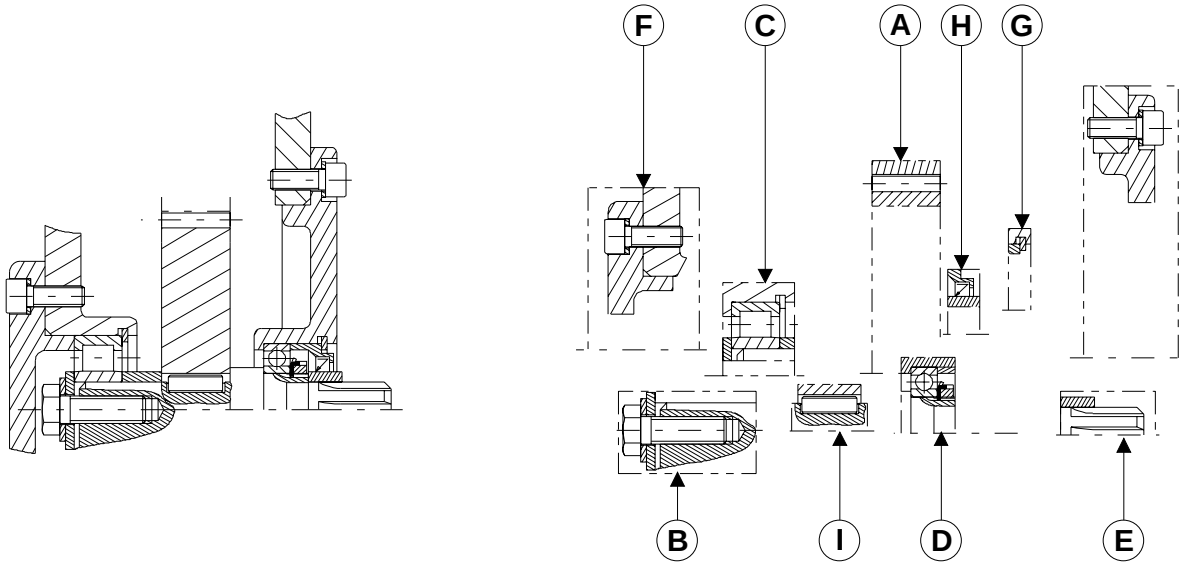
Figure 1.1 : Processus de conception : niveaux de définition d'un hélicoptère d'après [28].

A chaque niveau correspond un processus de définition avec des règles spécifiques, permettant de faire le meilleur choix parmi l'ensemble des solutions connues. Les choix effectués à un niveau vont se répercuter sur le niveau inférieur. En cas d'incompatibilité avec les règles de définition de ce niveau, il faudra effectuer un "retour en arrière" pour remettre en question les choix du niveau supérieur. Lors de cette étude globale descendante on va atteindre le niveau de définition le plus fin constitué par les réalisations technologiques des liaisons simples. A partir de cette étape on peut réaliser en remontant l'étude technologique pour aboutir à la machine. Bien entendu, le processus de conception ne se résume pas à une seule descente et une seule remontée de cette décomposition fonctionnelle, de nombreux bouclages à un même niveau et retour au niveau précédent sont souvent nécessaires pour parvenir à une bonne solution.

2.2 Réalisation technologique d'une liaison

Considérons le mécanisme présenté sur la figure 1.2a, qui pourrait se situer au niveau 4 de la décomposition fonctionnelle de la figure 1.1 Ce mécanisme peut être vu comme l'assemblage des solutions technologiques des différentes liaisons (figure 1.2b), dimensionnées pour satisfaire aux données issues d'un niveau supérieur.

Figure 1.2 : Principe de conception d'un mécanisme.



(a) : Mécanisme complet. ***(b) : Solutions technologiques des liaisons.***

J. GUILLOT précise la définition d'une réalisation technologique d'une liaison de la manière suivante [28]:

> *"La réalisation technologique d'une liaison, est constituée de l'ensemble des parties fonctionnelles (surfaces de contact, volumes de matière) appartenant aux différentes pièces et nécessaires pour assurer la transmission des efforts entre deux pièces principales (définies dans la procédure de niveau supérieur)."*

Ces liaisons peuvent se classer en trois catégories :

1) Les liaisons mécaniques de base (liaison ponctuelle, linéaire annulaire, rotule, pivot) satisfaisant certaines conditions cinématiques.
2) Les liaisons complètes (ou encastrement) satisfaisant des conditions de montage.
3) Les liaisons de type étanchéité, jouant un rôle particulier, surabondantes du point de vue de l'étude strictement mécanique.

Des liaisons de même nature (appartenant à la même catégorie) n'interviennent pas nécessairement au même niveau dans la décomposition fonctionnelle du mécanisme. En effet les liaisons **C** et **D** sur la figure 1.2b font partie du niveau de définition le plus fin, tandis que **A**, assurant la liaison entre deux éléments d'un mécanisme, intervient au niveau immédiatement supérieur. Le choix de **A** va imposer des conditions de définition sur **C** et **D**.

2.3 Conception d'un mécanisme : Recherche de la meilleure solution

Le mécanisme représenté sur la figure 1.2a comporte des liaisons par denture d'engrenage **A**, par cannelures **E**, et par bride vissée **F**. Ces liaisons sont définies par des conditions extérieures au mécanisme considéré. En supposant que $\mathbf{B}_i$ $(i = 1,2,3,4)$ et $\mathbf{C}_i, \mathbf{D}_i$ $(i = 1,2,3)$ sont toutes les solutions possibles des liaisons composant le mécanisme, l'ensemble des solutions globales peut être décrit par le graphe de la figure 1.3.

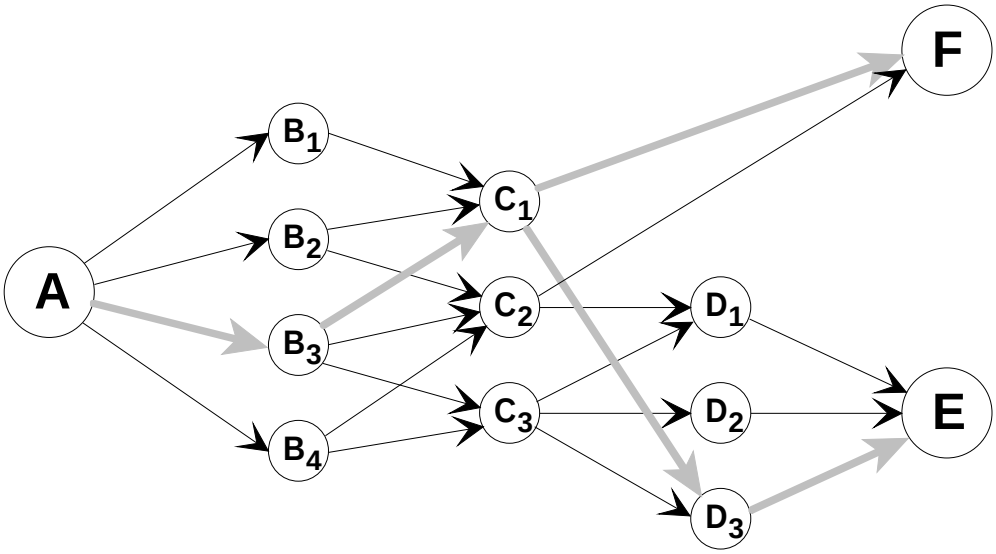


Figure 1.3 : Graphe de l'ensemble des solutions réalisables.

L'existence d'un arc entre deux sommets successifs traduit la compatibilité entre deux solutions technologiques partielles. Cette compatibilité se limite fréquemment aux conditions géométriques de montage ou d'usinage. Le nombre de chemins de parcours de ce graphe est également le nombre de solutions globales, ou de conception, du mécanisme considéré. Donc la solution à retenir, considérée comme la meilleure, sera celle qui satisfait aux critères d'optimalité retenus par le concepteur.

2.4 Le système SICAM

On comprend facilement l'intérêt d'un outil informatique permettant au concepteur d'explorer rapidement l'ensemble des chemins possibles du graphe de la figure 1.3, pour la recherche de la meilleure solution.

Afin d'expérimenter la démarche exposée ci-dessus, *J. GUILLOT* et son équipe ont conçu le logiciel *SICAM* (Système Interactif de Conception Assistée de Mécanisme). Ce logiciel permet de concevoir tous les mécanismes composés d'engrenages montés sur des paliers à roulements et comprenant des liaisons arbre moyeux par clavettes ou par cannelures, des liaisons par vis ou boulons, des liaisons par circlips ou écrous spéciaux et des étanchéités de tous types pour arbre tournant.

Ce logiciel est architecturé autour d'une base de données contenant l'ensemble des solutions technologiques des différentes liaisons, d'un ensemble de modules de calcul spécifiques à chaque liaison et d'un module de dessin 2D. Chaque module de calcul de liaisons est conçu de manière à déterminer la meilleure solution en fonction d'un ensemble de critères d'optimisation pré-définis.

A travers un ensemble de questions présentées sous forme de menus graphiques récapitulant tous les choix possibles, ce système permet à l'opérateur de choisir la disposition et la solution technologique des différentes liaisons du mécanisme, chaque liaison étant automatiquement dimensionnée. A tout instant *SICAM* offre la possibilité de remettre en question les choix effectués et de revenir "en arrière" pour en effectuer de nouveaux. L'utilisateur peut ainsi explorer rapidement de nombreuses solutions de conception pour ce type de mécanisme et obtient pour chaque conception un dessin d'ensemble coté.

Les résultats obtenus avec *SICAM* valident le principe d'un processus de conception organisé autour de la décomposition d'un système mécanique complexe en sous-systèmes élémentaires que sont les "réalisations technologiques des liaisons".

3 Conclusions

Le graphe de la figure 1.3 montre que la meilleure solution (en quelque sorte le meilleur chemin) est celle qui satisfait le mieux aux critères d'optimisation retenus par le concepteur, et donc **qu'un problème de conception de mécanisme peut se formuler comme un problème d'optimisation**.

Toutefois pour modéliser complètement le problème de conception de ce mécanisme, il faudrait exprimer la totalité du graphe sous la forme d'un problème d'optimisation. Comme chaque sommet du graphe représente la réalisation technologique d'une liaison et possède sa propre méthode de dimensionnement, ce problème d'optimisation ne peut être formulé que pour un chemin donné. Donc il ne peut être décrit mathématiquement que pour un ensemble fixé de choix technologiques.

Dans le logiciel *SICAM*, le problème d'optimisation est formulé pour chaque liaison du mécanisme, et c'est l'opérateur qui agence lui-même l'ensemble de ces solutions optimales pour créer un mécanisme. Cette approche permet de traiter des problèmes de petites tailles que l'on peut résoudre par une méthode graphique.

La mise au point d'une méthode de résolution générale permettrait de résoudre des problèmes d'optimisation de plus grande taille modélisant la conception d'un mécanisme comportant plusieurs liaisons.

Avant d'aborder la mise au point d'une telle méthode, il est nécessaire de préciser la démarche à suivre pour passer d'un problème de conception à un problème d'optimisation. Cette étape nous permettra également de caractériser les difficultés inhérentes à ce type de problème d'optimisation que nous nommerons : "Problème de conception optimale"

Chapitre 2

Expression du problème d'optimisation

Nous avons montré au chapitre précédent qu'un problème de conception peut finalement se ramener, sous certaines conditions, à un problème d'optimisation. On se propose maintenant, d'étudier plus en détail les différents aspects de la modélisation d'un problème de conception sous la forme d'un problème d'optimisation.

1 Enoncé mathématique d'un problème d'optimisation

Un problème d'optimisation, en programmation mathématique, s'énonce généralement de la manière suivante :

Déterminer les valeurs x_i des *n* variables qui sont les composantes du vecteur :

$$x = \{x_1 \dots x_i \dots x_n\}^T \ , \ x \in S \text{ avec } S \subset R^n$$

qui minimisent (ou maximisent) la fonction réelle, appelée fonction objectif, telle que :

$$f : x \in S \longrightarrow f(x) \in R$$

et qui satisfont un ensemble de conditions restrictives, également exprimées sous la forme de fonctions réelles, appelées fonctions contraintes telles que :

$$c_j : x \in S \longrightarrow c_j(x) \in R \text{ pour : } j = 1..m+l$$

Avec:

$$c_j(x) \le 0 \quad j = 1..m$$
$$c_j(x) = 0 \quad j = m+1..m+l$$

Notons que dans l'énoncé ci-dessus on distingue deux types de conditions restrictives : Les *m* fonctions contraintes inégalités et les *l* fonctions contraintes égalités.

L'ensemble de ces fonctions contraintes limite un domaine de *S* appelé domaine des solutions réalisables. Nous étudierons plus précisément les conditions d'existence d'une solution pour ce type de problème dans le chapitre suivant. Nous admettrons pour l'instant que les fonctions objectif et contraintes sont continues et différentiables sur R^n.

Considérons par exemple le problème suivant, de deux variables comportant trois fonctions contraintes inégalités :

$$\begin{cases} \text{Minimiser} \quad f(x) = 10(x_1^{\,2} - x_2)^2 + (x_1 - 1)^2 + 4 \\ \text{Sous les fonctions contraintes :} \\ c_1(x) = 2.5x_1^{\,2} + 3x_1 - 8x_2 + 8 \le 0 \\ c_2(x) = 1.5x_1^{\,2} - 5x_1 - 8x_2 + 12 \le 0 \\ c_3(x) = x_1 - 1.25 \le 0 \end{cases}$$

$$\text{Avec } x = \{x_1, x_2\}^T$$

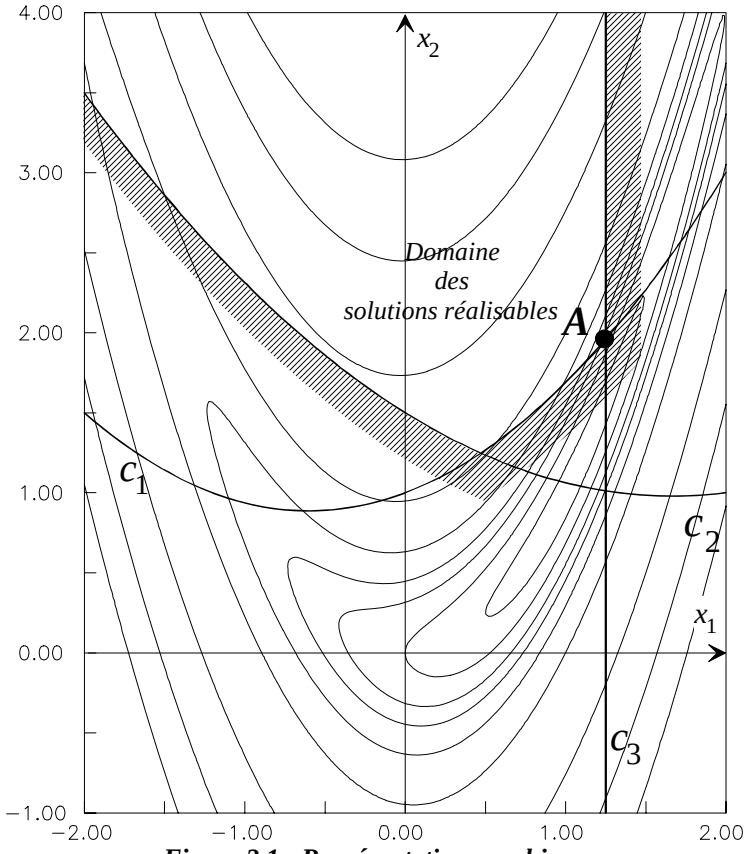


Figure 2.1 : Représentation graphique d'un problème bidimensionnel.

La figure 2.1 représente les contours (les courbes *f*(*x*) = constante) de la fonction objectif, ainsi que les courbes $c_j(x) = 0$, pour $j = 1,2,3$, celles-ci sont hachurées côté "positif" de sorte que le domaine *D* est représenté par la région du plan située du côté "négatif" des fonctions contraintes et limité par les courbes $c_j(x) = 0$, pour $j = 1,2,3$.La solution de ce problème est représentée par le point *A* de coordonnées $x_1 = 1.25$ et $x_2 = 1.957$.

2 Modélisation du problème de conception : Démarche générale

Cette étape, fondamentale dans le processus de recherche de la solution optimale, aura pour objet l'identification des différents éléments de l'énoncé mathématique du problème d'optimisation : les variables, la fonction objectif et les fonctions contraintes.

Lorsque l'ensemble des choix technologiques du mécanisme, ainsi que la configuration de base de celui-ci sont définis, on peut décrire ce mécanisme à l'aide d'un certain nombre de paramètres, par exemple :

- Des paramètres géométriques (nombre, position, dimensions des divers éléments).
- Des paramètres liés aux matériaux (module d'élasticité, coefficient de poisson, limite de résistance, masse volumique, etc. ...)
- Des paramètres fonctionnels (charges, vitesses, etc. ..)

L'analyse fonctionnelle globale, effectuée au niveau immédiatement supérieur au système étudié, permet de fixer la valeur d'une partie de ces paramètres. Ceux ci ne pourront pas être choisis comme variables du problème d'optimisation, nous les considérerons alors comme des données du problème. Toutefois leurs valeurs pourront être remises en causes, dans le cas où par exemple le domaine des solutions du problème d'optimisation serait vide.

Tous ces paramètres descriptifs sont liés par un ensemble de relations fonctionnelles spécifiques du mécanisme étudié, dont il faudra établir une liste.

On obtient généralement un nombre conséquent de relations et de paramètres. Il sera indispensable, dans un deuxième temps, de les réduire au minimum afin d'obtenir l'expression la plus simple possible pour le problème d'optimisation.

2.1 Expression initiale du problème

Le recensement de toutes les relations permettant de décrire l'ensemble du mécanisme tant au niveau fonctionnel que géométrique est une opération qui nécessite de bonnes connaissances technologiques. Il est important que l'expression du problème d'optimisation soit la plus complète possible, c'est à dire qu'elle permette de traiter l'ensemble des mécanismes du même type.

Il n'est pas possible d'établir ici une liste exhaustive de l'ensemble des relations fonctionnelles permettant de décrire un mécanisme quelconque; de plus cette étape ne relève à l'heure actuelle d'aucune démarche systématique. On retrouve cependant un certain nombre de points communs dans de nombreux mécanismes.

Par exemple, les modèles de calcul permettant de dimensionner les divers éléments de machines qui composent bon nombre de mécanismes (engrenages, roulements,..). Ceux ci fournissent une grande partie des relations fonctionnelles, à laquelle on ajoutera souvent des conditions de résistance statique et dynamique liant des paramètres caractéristiques d'un matériau (limite élastique, limite de fatigue,..) aux paramètres géométriques des divers éléments.

Les conditions d'encombrement géométrique, ainsi que les conditions liées au fonctionnement (positionnement des roues dentées d'un engrenage, tension d'une courroie, efforts exercés par un ressort de rappel, etc...), permettent d'écrire d'autres relations.

On peut aussi inclure des relations définissant le coût des divers éléments (lorsque celui-ci est facilement accessible en fonction des dimensions caractéristiques par exemple) ou des relations provenant de conditions de fabrication particulières (épaisseur de paroi minimale pour des pièces moulées, ..).

La liste donnée ici n'est, bien sur, pas limitative et comporte deux types de relations fonctionnelles :

- Les relations fonctionnelles liant certains paramètres entre eux, ce sont alors des équations fonctionnelles, elles permettront soit d'éliminer par substitution quelques paramètres soit d'écrire des conditions fonctionnelles égalités (fonctions contraintes égalités).
- Les relations fonctionnelles imposant des valeurs limites sur les paramètres du mécanisme, elles permettront d'écrire des conditions fonctionnelles limites (fonctions contraintes inégalités).

Il faudra enfin exprimer en fonction des paramètres du mécanisme, le ou les critères d'optimisation ou autrement dit, le ou les objectifs à atteindre, soit par exemple :

- Minimiser la masse, l'encombrement, ou le coût du mécanisme.
- Maximiser une fréquence propre, une raideur, ou un coefficient de sécurité statique ou dynamique.

- Obtenir avec une précision maximale une valeur fixée, par exemple un entraxe donné, un rapport de transmission. Dans ce cas on sera amené à minimiser l'écart entre la valeur calculée et la valeur souhaitée.

L'objectif à atteindre n'est, en pratique, jamais unique. On souhaite souvent satisfaire simultanément plusieurs critères d'optimisation contradictoires, et donc obtenir une formulation multicritères du problème. L'expression de la fonction objectif doit alors tenir compte de l'ensemble des critères retenus. De toute façon le problème peut toujours se ramener à la formulation énoncée au § 1. On trouvera dans [28] une étude détaillée sur les différentes formulations d'une fonction objectif multicritères.

2.2 Expression finale : réduction de la taille du problème

Soit n_p le nombre total de paramètres de description du mécanisme étudié (non compris les paramètres fixés et considérés comme données du problème),et m_e le nombre d'équations fonctionnelles liant ces n_p paramètres. Le degré de liberté du problème est :

$$DL = n_p - m_e$$

On notera que ce degré de liberté est nécessairement positif ou nul, le cas contraire signifierait que l'on a oublié des paramètres descriptifs , ou que certaines équations fonctionnelles sont redondantes.

Théoriquement, ces m_e équations fonctionnelles doivent permettre d'éliminer par substitution m_e paramètres descriptifs. Les conditions de cette élimination sont fixées par les hypothèses du théorème des fonctions implicites [43]. En pratique ces éliminations ne seront possibles que pour des expressions analytiques suffisamment simples des équations fonctionnelles.

Si n_{pe} est le nombre de paramètres éliminables par substitution, avec :

$$n_{pe} \leq m_e$$

il reste $n_p - n_{pe}$ paramètres "libres", et le problème d'optimisation s'exprimera en fonction de $n_p - n_{pe}$ variables. Donc, théoriquement le nombre d'expressions possibles N_{ep}, est donné par :

$$N_{ep} = \left(\frac{n_p!}{n_{pe}!\left(n_p - n_{pe}\right)!} \right)$$

Sur ces N_{ep} expressions possibles, certaines sont irréalisables, parce qu'elles utiliseront comme variables des paramètres facilement éliminables, ou parce que parmi les $n_p - n_{pe}$ paramètres choisis aucun n'interviendra dans l'expression de la quantité à optimiser. De façon

pratique, on choisira d'abord d'éliminer les paramètres conduisant aux expressions analytiques les plus simples. Une fois l'ensemble des paramètres éliminables fixé, les paramètres restants seront automatiquement choisis comme variable dans l'expression du problème d'optimisation.

Finalement le problème d'optimisation s'exprimera avec les $n_p - n_{pe}$ paramètres choisis comme variables et les $m_e - n_{pe}$ équations fonctionnelles restantes qui ne permettront pas de substitution simple et qui donneront autant de conditions fonctionnelles égalités, ou fonctions contraintes égalités. Il faudra également inclure dans cette expression l'ensemble des m_i relations fonctionnelles s'exprimant par des inégalités, ce sont elles qui donneront les conditions fonctionnelles limites, ou fonctions contraintes inégalités.

Parmi ces m_i conditions fonctionnelles limites on trouve celles qui imposent directement des limites sur les variables du problème (encombrement géométrique par exemple) et celles qui limitent indirectement les variables du problème (condition de résistance limitant des paramètres géométriques par exemple).

Nous proposons d'illustrer la démarche exposée ci-dessus, en l'appliquant sur un exemple de conception de mécanisme.

3 Exemple de modélisation : "accouplement à plateaux"

Considérons le mécanisme élémentaire représenté sur la figure 2.2, constitué de quelques liaisons élémentaires, et destiné à transmettre par adhérence un couple entre deux arbres coaxiaux.

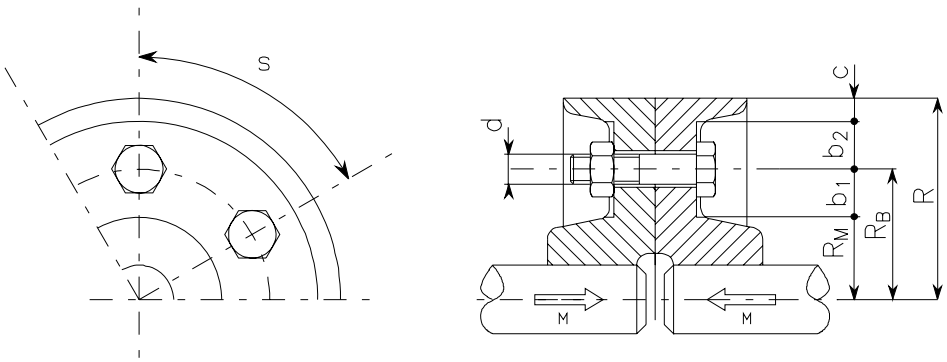


Figure 2.2 : Accouplement à plateaux.

Le problème de conception que l'on cherchera à exprimer comme un problème d'optimisation est le suivant :

"Déterminer le nombre de boulons N, de diamètre d et de qualité donnée, disposés sur le rayon R_B, transmettant par adhérence un couple M et tel que l'encombrement diamétral du mécanisme soit le plus faible possible"

3.1 Expression initiale

L'étude que nous faisons ici, suppose que la liaison encastrement arbre - plateau a déjà été choisie et dimensionnée. Nous admettrons également que la forme des plateaux est imposée par des conditions de fabrication.

La quantité à minimiser est clairement identifiable et s'écrit, avec les notations de la figure 2.2 :

$$R = R_M + b_1 + b_2 + c$$

Avec : R_M Plus grand rayon d'encombrement des liaisons arbre - moyeu.

c Epaisseur de la jante extérieure de protection, fonction du procédé d'obtention des plateaux.

Les relations fonctionnelles que l'on peut écrire sur ce mécanisme sont les suivantes :

L'équation fonctionnelle définissant le couple minimal transmissible par cet accouplement donne :

$$M = Q_m . f_m . N . R_B \qquad (1)$$

Avec : Q_m Précontrainte minimale dans le boulon.

f_m Coefficient de frottement minimal entre les plateaux.

En utilisant les modèles de calcul des assemblages boulonnés issus de [27] on peut écrire les relations suivantes :

Contrainte équivalente de Von Mises maximale dans le boulon :

$$\sigma_B = \sqrt{\left(\frac{Q_M}{S_e}\right)^2 + 3\left(\frac{16C}{\pi d_e^{\,3}}\right)^2} \qquad (2)$$

Avec : Q_M Précontrainte maximale dans le boulon.

S_e Section résistante équivalente de la vis.

d_e Diamètre de la section résistante.

Couple maximal de torsion appliqué à la vis lors du serrage :

$$C = (0.16\,p + 0.583 d_2 f_1) Q_M \tag{3}$$

Avec : p Le pas de la vis.
d_2 Diamètre moyen à flanc de filets de la vis.
f_1 Coefficient de frottement vis - écrou.

Précontrainte installée dans le boulon lors du serrage :

$$Q_M = \alpha Q_m \tag{4}$$

Avec : α Coefficient d'incertitude, lié à l'outil de serrage utilisé.

Les équations fonctionnelles issues de conditions géométriques sont les suivantes :

Section résistance S_e du boulon : $S_e = \frac{\pi d_e^{\,2}}{4}$ (5)

Intervalle entre les boulons : $s = \frac{2\pi R_B}{N}$ (6)

Sur les rayons : $R_B = R_M + b_1$ (7)

Sur les diamètres : $d_e = \phi_1(d)$ (8)

$d_2 = \phi_2(d)$ (9)

Sur le pas : $p = \phi_3(d)$ (10)

Encombrement géométrique de l'outil de serrage : $b_m = \phi_4(d)$ (11)

$s_m = \phi_5(d)$ (12)

Il convient enfin de rajouter les conditions fonctionnelles limites suivantes :

Sur la résistance statique du boulon : $\sigma_B \le 0.9 R_e$ (13)

Avec : R_e Limite élastique de la classe de qualité des vis considérées.

Sur le couple transmis : $M \ge M_T$ (14)

Sur le nombre de boulons : $N \ge N_m$ (15)

Sur le diamètre des boulons : $d \ge d_m$ (16)

Sur l'encombrement de l'outil de serrage : $b_2 \geq b_m$ (17)

$b_1 \geq b_m$ (18)

$s \geq s_m$ (19)

Les relations ci-dessus font intervenir les 26 paramètres suivants :

Paramètres fonctionnels : $M, M_T, Q_m, Q_M, C, \alpha, \sigma_B$

Paramètres géométriques : $d_m, d, d_e, d_2, p, S_e, s, b_1, b_2, b_m, s_m, N, N_m, R_B, R_M, c$

Paramètres liés aux matériaux des boulons et des plateaux : R_e, f_m, f_1

Les relations ϕ_i liant les divers paramètres géométriques concernant le boulon et l'outil de serrage ne sont pas explicitées ici. Elles expriment le fait que les paramètres comme le pas et les diamètre moyen et à flanc de filets dépendent du diamètre des boulons *d*. Ce sont des valeurs normalisées accessibles uniquement par la valeur de *d*. Dans les calculs ultérieurs les valeurs de ces paramètres seront extraites d'un tableau à partir de la valeur de *d*. Dans l'expression du problème les fonctions ϕ_i seront conservées pour simplifier les écritures des conditions fonctionnelles.

Nous supposerons que les valeurs des 9 paramètres suivants :

$$M_T, f_m, f_1, \alpha, R_e, N_m, R_M, c, d_m$$

ont été fixées par une analyse fonctionnelle globale, ils seront considérés comme données du problème, il reste donc 17 paramètres variables dans ce problème.

3.2 Expression finale : réduction du nombre de variables

L'ensemble des 19 relations fonctionnelles comporte 12 équations fonctionnelles qui vont permettre d'éliminer les 12 paramètres suivants :

$$C, Q_m, Q_M, S_e, s, b_1, \sigma_B, d_e, d_2, p, b_m, s_m$$

Par conséquent nous exprimerons ce problème en fonction des 5 variables suivantes :

$$d, N, R_B, M, b_2$$

En combinant les équations fonctionnelles (1),(2),(3) et (4) avec la condition fonctionnelle limite (13) on obtient la relation :

$$\frac{\alpha.M}{N.R_B} \le K(d) \text{ Avec : } K(d) = \frac{0.9\, f_m R_e \pi(\phi_1(d))^2}{4\sqrt{1+3\left(\frac{4(0.16\phi_3(d)+0.583\phi_2(d)\, f_1)}{\phi_1(d)}\right)^2}}$$

De même les combinaisons des équations fonctionnelles (6) et (7) avec les inégalités respectives (19) et (18) permettent d'écrire :

$$\frac{2\pi R_B}{N} \ge \phi_5(d)$$
$$R_B \ge R_M + \phi_4(d)$$

On cherche à minimiser l'expression :

$$R = R_M + b_1 + b_2 + c = R_B + b_2 + c$$

avec : $b_2 \ge b_m$. Donc on aura systématiquement R minimal pour $b_2 = b_m = \phi_4(d)$. On se ramène alors au problème suivant, comportant 4 variables de description et 6 fonctions contraintes inégalités :

Minimiser l'expression : $R = R_B + \phi_4(d) + c$

Sous les fonctions contraintes :
$M \ge M_T$
$\frac{\alpha.M}{N.R_B} \le K(d)$
$\frac{2\pi R_B}{N} \ge \phi_5(d)$
$R_B \ge R_M + \phi_4(d)$
$N \ge N_m$
$d \ge d_m$

Pour présenter tous les problèmes de conception optimale traités dans cet ouvrage nous adopterons l'écriture suivante qui présente l'avantage d'identifier clairement les variables choisies

Minimiser la fonction objectif :	$F(d, N, R_B, M) = R = R_B + \phi_4(d) + c$
Sous les fonctions contraintes :	$c_1(d, N, R_B, M) = M_T - M \leq 0$
	$c_2(d, N, R_B, M) = \frac{\alpha . M}{N . R_B} - K(d) \leq 0$
	$c_3(d, N, R_B, M) = \phi_5(d) - \frac{2\pi R_B}{N} \leq 0$
	$c_4(d, N, R_B, M) = R_M + \phi_4(d) - R_B \leq 0$
	$c_5(d, N, R_B, M) = N_m - N \leq 0$
	$c_6(d, N, R_B, M) = d_m - d \leq 0$
Vecteur variable :	$x = \{d, N, R_B, M\}^T$
Données :	$(M_T, f_m, f_1, \alpha, R_e, N_m, R_M, c, d_m)$

A partir de cet exemple très représentatif du type de problèmes d'optimisation que nous aurons à traiter en conception optimale, on peut tirer les conclusions suivantes :

On constate que plusieurs types de variables interviennent dans cette formulation :

- Des variables pouvant varier continûment entre les limites définies par le domaine des solutions, comme le moment M et le rayon R_B. Nous nommerons ce type de variables : **variables continues**.
- Des variables astreintes à prendre des valeurs **entières** comme le nombre de boulons N ou encore des variables **discrètes** comme le diamètre des boulons issues de la normalisation.

Le problème de conception optimale est donc très souvent un problème d'optimisation en variables **mixtes**. Soulignons également la particularité de certaines variables discrètes comme ici le diamètre des boulons. En effet certains paramètres sont directement dépendants de ce type de variables, ils agissent en quelque sorte comme des variables discrètes "secondaires" dont les valeurs ne sont accessibles qu'au travers de tableaux. Cette situation se rencontre pour beaucoup d'éléments de construction mécanique normalisés (les roulements par exemple) ou standards (les joints).

Ce type de problèmes est généralement très contraint, autrement dit, le nombre de fonctions contraintes est très souvent supérieur au nombre de variables. Cependant les expressions analytiques des différentes relations fonctionnelles permettent souvent d'éliminer par substitution les équations fonctionnelles de sorte que on a très rarement des fonctions contraintes égalités. Remarquons enfin que ce sont généralement des problèmes **non linéaires**, comportant fréquemment des **fonctions monotones**.

3.3 Résolution graphique

Nous allons voir qu'il est tout à fait possible de réduire encore, le nombre de variables de ce problème d'optimisation, pour se ramener au cas idéal d'un problème de deux variables. Nous pourrons ensuite appliquer la méthode préconisée par *Johnson* [34] et construire ainsi le "diagramme de variation" du problème, qui nous permettra d'obtenir rapidement la solution optimale.

Le problème de l'accouplement à plateaux fait intervenir une seule variable discrète *d*, et de plus l'ensemble des valeurs normalisées d'utilisation courante limite le nombre de diamètres à quelques dizaines. On peut raisonnablement envisager de résoudre le problème pour une valeur fixée de diamètre et de parcourir ensuite l'ensemble des valeurs normalisées pour trouver la valeur optimale du diamètre des boulons. Donc dans la suite du raisonnement *d* ainsi que les fonctions $\phi_i(d)$ seront considérées comme des données du problème.

En remarquant, également que pour *N* fixé le problème s'écrit :

Minimiser l'expression : $F(R_B, M) = R = R_B + K_1$

Sous les fonctions contraintes : $c_1(R_B, M) = M_T - M \leq 0$

$c_2(R_B, M) = K_2 . R_B - M \leq 0$

$c_3(R_B, M) = K_3 - R_B \leq 0$

Avec K_1, K_2, K_3 constants et

$$K_1 = \phi_4(d) + c, K_2 = \frac{N.K(d)}{\alpha}, K_3 = Max\left\{\frac{\phi_5(d).N}{2\pi}, R_M + \phi_4(d)\right\}$$

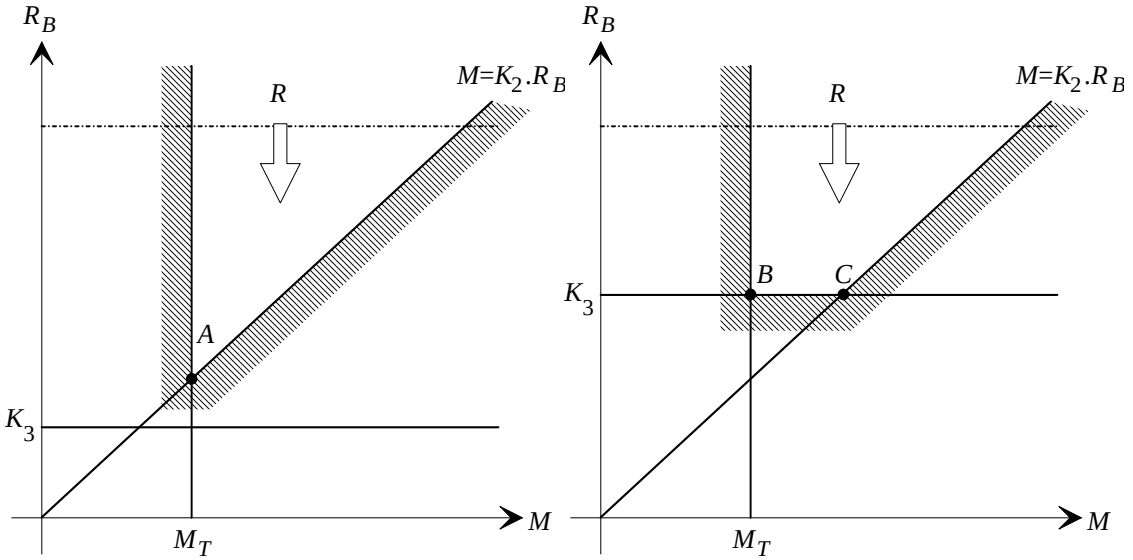


Figure 2.3 : Accouplement à plateaux réduction du nombre de variables.

On obtient la situation représentée sur la figure 2.3. Le rayon d'encombrement du plateau est minimal soit pour le point *A*, soit pour l'ensemble des points du segment [*B*,*C*].

Dans la première situation la solution du problème est unique, et dans la seconde le problème admet une infinité de solutions. Il sera donc nécessaire d'ajouter un autre critère d'optimisation pour obtenir un choix unique. La valeur du couple transmissible par cet accouplement est minimale au point B et maximale en C. On montre facilement que le point B correspond à l'utilisation du nombre minimum de boulons et le point C au nombre maximum de boulons que l'on peut disposer sur un cercle de rayon R_B.

Donc nous retiendrons comme solution les points *A* et *B*, situés sur la frontière définie par $M = M_T$.

Cette analyse nous permet finalement d'exprimer ce problème d'optimisation à l'aide de seulement deux variables : N et R_B.

Pour *d* donné et $M = M_T$, on obtient le problème :

Minimiser l'expression : $F(N, R_B) = R = R_B + \phi_4(d) + c$

Sous les fonctions contraintes :

$$c_1(N, R_B) = \frac{\alpha . M_T}{N . R_B} - K(d) \leq 0$$

$$c_2(N, R_B) = \phi_5(d) - \frac{2\pi R_B}{N} \leq 0$$

$$c_3(N, R_B) = R_M + \phi_4(d) - R_B \leq 0$$

$$c_4(N, R_B) = N_m - N \leq 0$$

La figure 2.4 représente les frontières du domaine des solutions définies par les 6 fonctions contraintes du problème. On obtient, par exemple, pour la contrainte $c_1(N, R_B)$:

$$c_1(N, R_B) = 0 \Rightarrow R_B = \left(\frac{\alpha M_T}{K(d)} \right) . \frac{1}{N}$$

soit une hyperbole dans le plan (N, R_B). Les flèches situées sur chaque frontière du domaine des solutions précisent les variations de position de ces frontières en fonction de la variation des données du problème (et en particulier du diamètre des boulons *d*).

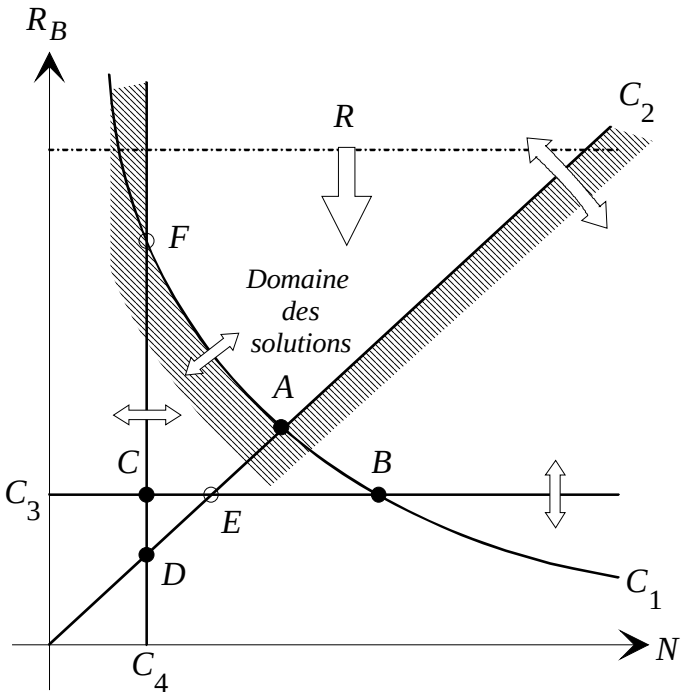


Figure 2.4 : Accouplement à plateaux : diagramme de variation.

On a également représenté l'ensemble de tous les points solutions dans les différentes configurations du domaine des solutions. A priori, il y a 6 points solutions, mais les configurations représentées sur la figure 2.5 montrent clairement que seuls les 4 premiers points sont à considérer.

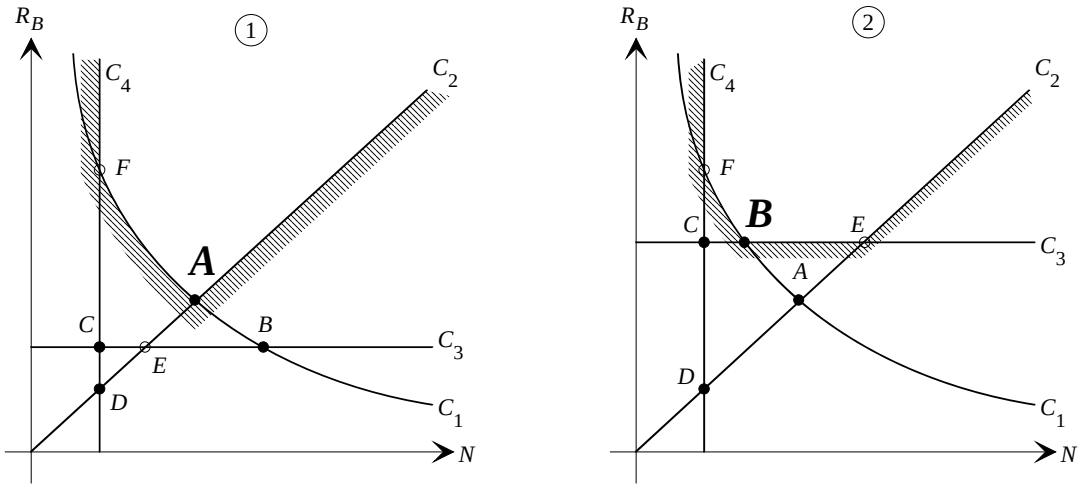


Figure 2.5 : Solutions optimales du problème de l'accouplement à plateaux.

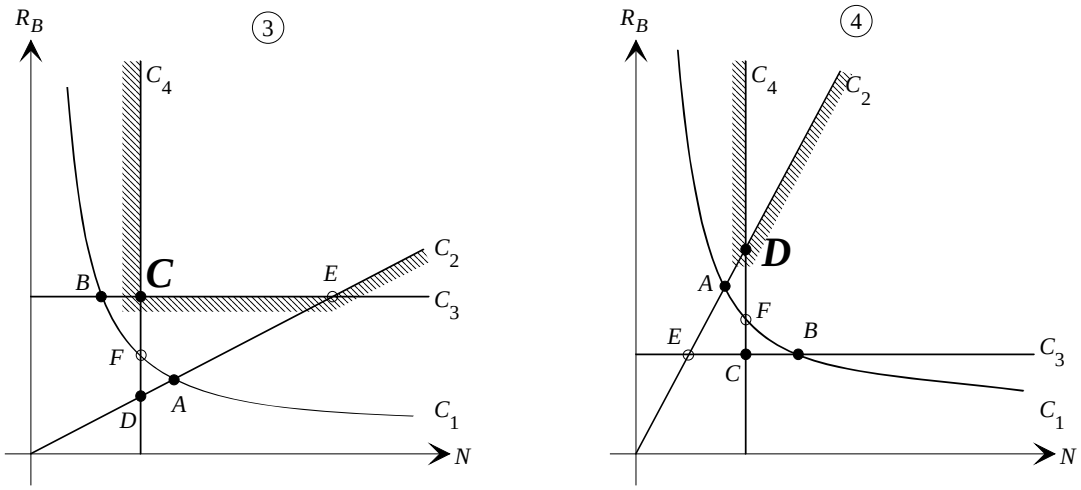


Figure 2.5 : Solutions optimales du problème de l'accouplement à plateaux (suite).

On peut à partir de la figure 2.5, facilement calculer chaque solution pour ces 4 configurations. Dans le cas des points *A* et *B*, la valeur calculée de *N* peut ne pas être entière et il sera nécessaire de l'arrondir. Pour le point *B* on choisira l'entier immédiatement supérieur, tandis que pour le point *A* on choisira parmi les entiers les plus proches celui qui donne la plus faible valeur pour R_B (figure 2.6). On notera que dans les cas 2 et 3, on obtient une valeur identique du rayon d'encombrement pour tous les points des segments [*B*,*C*] et [*C*,*E*]. Le choix des points *B* et *C* comme solution du problème correspond là encore au choix implicite d'un critère d'optimisation supplémentaire : le nombre minimal de boulons.

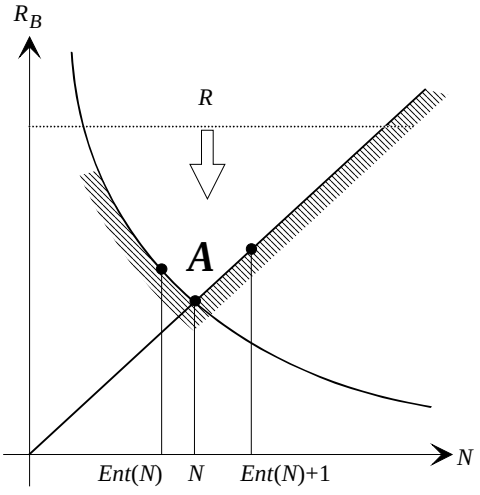


Figure 2.6 : Procédure d'arrondi[1] pour le nombre de boulons.

[1]La fonction Ent(*x*) désigne ici la partie entière du réel *x*.

Finalement on obtient les solutions suivantes :

Point *A* :

$$\begin{cases} N = \sqrt{\dfrac{2\pi.\alpha.M_T}{\phi_5(d).K(d)}}; N = Ent(N) \text{ pour } \mathrm{R_B} = R_{B1};\ N = Ent(N)+1 \text{ pour } \mathrm{R_B} = R_{B2} \\ R_{B1} = \dfrac{\alpha.M_T}{K(d).Ent(N)},\ R_{B2} = \dfrac{\phi_5(d)\left(Ent(N)+1\right)}{2\pi}; R_B = Min\left\{R_{B1}, R_{B2}\right\} \end{cases}$$

Point *B* :

$$\begin{cases} N = Ent\left(\dfrac{\alpha.M_T}{K(d).\left(R_M + \phi_4(d)\right)}\right)+1 \\ R_B = R_M + \phi_4(d) \end{cases}$$

Point *C* :

$$\begin{cases} N = N_m \\ R_B = R_M + \phi_4(d) \end{cases}$$

Point *D* :

$$\begin{cases} N = N_m \\ R_B = \dfrac{\phi_5(d).N_m}{2\pi} \end{cases}$$

Le choix du point solution convenable, pour un diamètre de boulons fixé, parmi ces 4 possibilités se fera de la manière suivante :

On retiendra d'abord le ou les points qui ont la plus grande valeur de R_B puis on choisira celui pour lequel la valeur de *N* est maximale.

Finalement on obtiendra la solution optimale sur l'ensemble des diamètres normalisés avec l'algorithme suivant :

1) Lecture des données:
2) $j \leftarrow 1, R_{opt} = 0$
3) Lecture des valeurs normalisées : d_j et $\phi_i(d_j)$ $\quad i = 1..4$
4) Calcul du coefficient $K(d_j)$

   Calcul de tous les points solutions *A,B,C,D*.
   Retenir les points tels que R_B maxi puis *N* maxi
5) Calcul de R_j
6) Si $R_j > R_{opt}$ alors conserver les valeurs de $R_B, N, R_j\, d_j$.
7) $j \leftarrow j+1$
8) Si FIN des valeurs normalisées

   alors afficher les valeurs conservées, STOP.

   sinon aller en 3)

Avec les données normalisées concernant les assemblages boulonnés [1] regroupées dans le tableau ci-dessous :

d	$d_e = \phi_1(d)$	$d_2 = \phi_2(d)$	$p = \phi_3(d)$	$b_m = \phi_4(d)$	$s_m = \phi_5(d)$
6	5.062	5.350	1.00	7.50	14.50
8	6.827	7.188	1.25	9.50	18.50
10	8.593	9.026	1.50	12.50	23.50
12	10.358	10.863	1.75	13.50	26.50
14	12.124	12.701	2.00	15.50	29.50
16	14.124	14.701	2.00	17.00	32.00
20	17.655	18.376	2.50	21.00	40.00
24	21.185	22.051	3.00	25.00	48.00

et les données suivantes du problème :

M_T	= 4000 m.N
f_m	= 0.15
f_1	= 0.15
α	= 1.5 (Serrage par clé dynamométrique)
R_e	= 627 Mpa (Classe de qualité des boulons : 8.8)
N_m	= 8
R_M	= 50 mm
c	= 5 mm

on obtient le tableau de résultats suivants :

d		***A***	***B***	***C***	***D***
6	N	45.00	81.00	8.00	8.00
	R_B	103.85	57.50	57.50	18.46
	R	116.35	70.00	70.00	30.96
8	N	29.00	43.00	8.00	8.00
	R_B	87.24	59.50	59.50	23.55
	R	101.74	74.00	74.00	38.05
10	N	21.00	26.00	8.00	8.00
	R_B	78.54	62.50	62.50	29.92
	R	96.04	80.00	80.00	47.42
12	N	16.00	18.00	8.00	8.00
	R_B	68.08	63.50	63.50	33.74
	R	86.58	82.00	82.00	52.24
14	N	13.00	13.00	8.00	8.00
	R_B	61.04	65.50	65.50	37.56
	R	81.54	86.00	86.00	58.06
16	N	11.00	9.00	8.00	8.00
	R_B	56.02	67.00	67.00	40.74
	R	78.02	89.00	89.00	62.74
20	N	8.00	6.00	8.00	8.00
	R_B	50.93	71.00	71.00	50.93
	R	76.93	97.00	97.00	76.93

Les deux solutions obtenues avec ces données et encadrées dans ce tableau sont très proches, et quasi équivalentes d'un point de vue technologique. Le choix entre ces deux

alternatives pourra se faire sur le coût de chaque solution. Dans ce cas on privilégiera la solution comportant 13 boulons de diamètre 14.

4 Conclusions

Nous venons de voir comment l'utilisation d'un diagramme de variation permettait de trouver la solution d'un problème d'optimisation. C'est une approche intéressante pour des problèmes comportant très peu de variables (3 ou 4 maximum). Pour ce type de problème cette méthode permet d'obtenir une image dynamique du problème de conception optimale en mettant en évidence l'importance des variations sur les données et les cas de spécifications incompatibles (domaine des solutions vide). Elle permet également d'obtenir l'ensemble des points solutions ainsi que les conditions fonctionnelles limites actives pour chacun d'eux. On peut à partir de là facilement obtenir l'expression analytique des solutions, ou lorsque l'ensemble d'un segment frontière est solution introduire un critère de choix supplémentaire et affiner ainsi la modélisation du problème.

Par contre cette technique d'analyse et de résolution d'un problème d'optimisation ne pourra pas s'appliquer à des problèmes de plus grandes tailles comportant plusieurs variables discrètes. De plus même appliquée sur de petits problèmes, elle ne permet pas de bâtir des algorithmes de résolution généraux. On remarque cependant que l'algorithme mis au point avec cette méthode traite l'ensemble des liaisons de type "accouplements à plateaux" indépendamment de la valeur des données du problème.

On se propose de mettre au point un algorithme de résolution général utilisant les méthodes de la programmation mathématique, et applicable aux problèmes de conception optimale en variables mixtes.

Méthodes de résolution des problèmes d'optimisation

Introduction

La programmation mathématique a pour objet l'étude théorique des problèmes d'optimisation ainsi que la conception et la mise en œuvre des algorithmes de résolution. La présence du terme "programmation" s'explique par le fait qu'historiquement les premières recherches et les premières applications se sont développées dans le contexte de l'économie et de la recherche opérationnelle; la terminologie employée reflète alors l'étroite relation entre l'activité d'analyse mathématique d'un problème et son interprétation économique (la recherche d'un programme économique optimal).

Il est convenu d'associer la naissance de la programmation mathématique à la découverte de la méthode du simplexe en 1947, méthode destinée à la résolution de problèmes d'optimisation linéaires. Ensuite les travaux sur la programmation non linéaire (*Kuhn* et *Tucker* 1951, *Zangwill* 1969, *Luenberger* 1973), la programmation dynamique (*Bellman* 1957), la programmation en nombre entier (*Gomory* 1958), la théorie des graphes conduisant à l'optimisation combinatoire, furent intégrés dans cette nouvelle discipline, clarifiant et unifiant ainsi l'apparente diversité des thèmes abordés pendant ces années.

Le nombre, la diversité et l'importance des applications de la programmation mathématique dans les sciences de l'ingénieur sont certainement l'une des causes de l'intense activité de recherche déployée depuis une trentaine d'années sur le sujet, et du nombre important de publications qui lui sont consacrées chaque année.

Cette partie est divisée en trois chapitres. Le premier concerne les méthodes de résolution destinées aux variables continues, et présente les principaux algorithmes de recherche de la solution optimale. Nous reprenons dans cet exposé la démarche qui permet d'aboutir aux deux résultats fondamentaux d'optimisation non linéaire : les conditions d'optimalité de *Kuhn et Tucker* et la théorie de la dualité. Cela nous permet, sous une présentation homogène au niveau des notations de bien définir les hypothèses d'application de ces résultats, et en particulier le rôle de la convexité en programmation non linéaire.

Le second concerne un exposé plus original sur les méthodes de résolution non itératives comme l'analyse monotone, destinée aux problèmes ne comportant que des fonctions monotones.

Le dernier chapitre de cette partie est consacré à une présentation des méthodes de résolution pour les problèmes comportant des variables discrètes et entières. Nous y ferons la synthèse des dernières techniques disponibles dans la littérature.

Chapitre 3

Méthodes pour variables continues

1 Généralités

1.1 Le problème d'optimisation en programmation mathématique

On peut distinguer deux types de problèmes en programmation mathématique :
Le problème de minimisation d'une fonction de plusieurs variables, que l'on peut écrire :

$$(P_l) \quad \left\langle \begin{array}{l} Trouver \ f(x^*) = \underset{x \in R^n}{Min} \ f(x) \\ \text{Où : } f : x \in R^n \longrightarrow f(x) \in R \end{array} \right.$$

La fonction à minimiser ou fonction objectif, f, sera supposée **continue** et **différentiable**. On appelle également (P_l) : problème d'optimisation sans contrainte car les composantes du vecteur variable x de R^n peuvent prendre toutes les valeurs de $-\infty$ à $+\infty$.

On distinguera également le problème d'optimisation avec contraintes suivant :

$$(P_c) \quad \left\langle \begin{array}{l} Trouver \ f(x^*) = \underset{x \in D}{Min} \ f(x) \\ \text{Où : } f : x \in R^n \longrightarrow f(x) \in R \end{array} \right.$$

L'ensemble D ,inclus dans R^n, est le domaine des solutions possibles de (P_c).

Tout point de *D* minimisant la fonction *f* est solution de (P_c). Ici nous définirons *D* par un ensemble de *m* fonctions non linéaires, notées $c_j(x)$ avec $c_j : x \in R^n \longrightarrow c_j(x) \in R$ pour $j = 1 \cdots m$, appelées fonctions contraintes qui ,comme la fonction objectif seront supposées continues et différentiables.

Il est possible de formuler deux types de fonctions contraintes : les fonctions contraintes inégalités ,$c_j(x) \le 0$, et les fonctions contraintes égalités ,$c_j(x) = 0$. Nous considérerons que *D* n'est défini que par des fonctions contraintes inégalités. Toutefois pour conserver une certaine généralité à l'exposé nous indiquerons les particularités liées aux fonctions contraintes égalités dans les différentes définitions et propriétés.

On supposera que *D* est **non vide**, c'est à dire qu'il existe au moins un point de R^n tel que :

$$c_j(x) \le 0 \quad \forall j = 1 \cdots m$$

Le problème *(P_c)* peut s'écrire de manière équivalente :

$$(P_c) \begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1 \ldots m \end{cases}$$

On n'envisage ici que le cas de la minimisation, mais cela n'est pas restrictif dans la mesure où un problème de maximisation d'une fonction *g* se traduit par la minimisation de $f(x) = -g(x)$. La solution optimale de (P_c), x^*, se trouve généralement sur une frontière de l'ensemble des solutions *D* . En ce point une ou plusieurs fonctions contraintes sont dites "actives" ou "saturées". On a dans ce cas :

$$c_j(x^*) = 0 \quad j \in J(x^*)$$

où $J(x^*)$ désigne l'ensemble des indices des contraintes actives en x^*.

La ou les solutions de ces problèmes sont caractérisées par un ensemble de conditions mathématiques appelées : **conditions d'optimalité**; selon que ces conditions seront vérifiées localement ou sur l'ensemble des valeurs admissibles par les variables, la solution x^* sera qualifiée d'optimum local ou d'optimum global. Nous verrons qu'il est généralement très difficile de statuer sur l'optimalité globale d'une solution, sauf avec des hypothèses assez restrictives, comme la convexité.

Dans le cas de l'optimisation sans contraintes, nous verrons dans la partie suivante que ces conditions d'optimalité sont tout simplement les conditions d'extremum d'une fonction réelle de plusieurs variables, alors que dans le cas de l'optimisation avec fonctions contraintes, ces

conditions sont un petit peu plus complexes et sont dénommées : **les conditions d'optimalité de *Kuhn et Tucker***

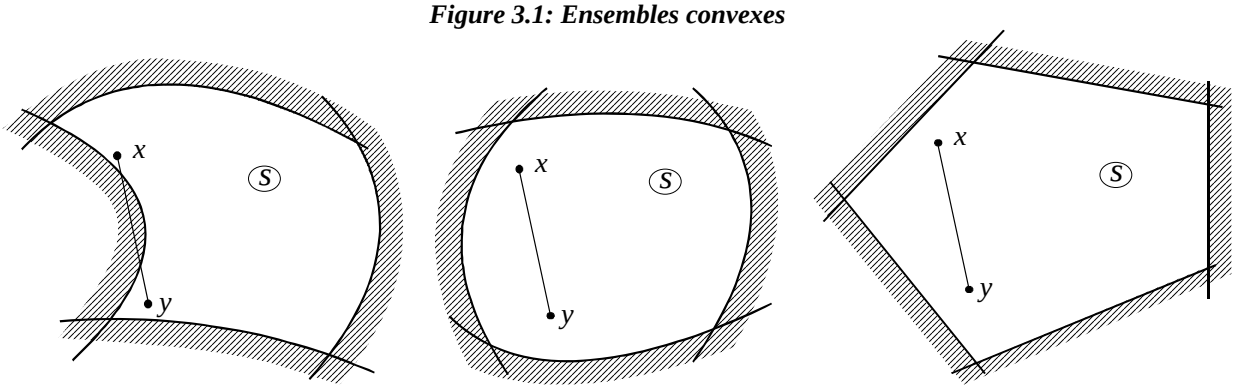
1.2 Fonctions convexes, problèmes convexes [45]

Un ensemble *S* inclus dans R^n est convexe si et seulement si :

$$\left.\begin{array}{l}\forall x \in S \\ \forall y \in S \\ \forall \theta \in [0,1]\end{array}\right\} \Rightarrow \theta x + (1-\theta) y \in S$$

Ou encore, de façon équivalente, on peut dire que *S* est convexe si pour deux points quelconques *x* et *y* de *S*, le segment $[x, y]$est, tout entier, contenu dans *S*. (Voir figure 3.1)

Figure 3.1: Ensembles convexes



Ensemble non convexe. ***Ensemble convexe.*** ***Ensemble convexe (Polyèdre).***

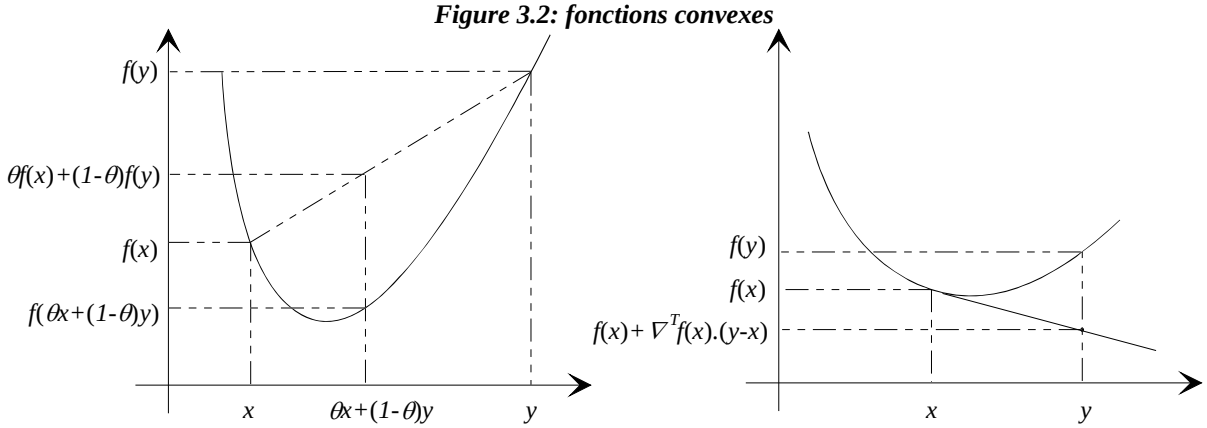
Une fonction *f*, continue et différentiable, de R^n dans R est convexe si au moins une des trois propriétés suivantes est vérifiée :

a)	$\forall \theta \in [0,1] f(\theta.x + (1-\theta).y) \leq \theta f(x) + (1-\theta) f(y)$
b)	$f(y) \geq f(x) + \nabla^T f(x).(y-x)$
c)	$y^T.\nabla^2 f(x).y \geq 0$: Le hessien de *f* en *x*, $\nabla^2 f(x)$, est une matrice semi-définie positive.

Si les inégalités strictes sont vérifiées pour $x \neq y$ et $\theta \in]0,1[$, *f* est alors strictement convexe.

La figure 3.2 illustre les propriétés (a) et (b) dans le cas d'une fonction de *R* dans *R* .

Figure 3.2: fonctions convexes



(a): Le segment [x,y] est toujours situé au "dessus" de la courbe représentant f.

(b): La tangente à la courbe représentant f est toujours située au "dessous" de la courbe.

Dans les problèmes d'optimisation, avec ou sans fonctions contraintes, la convexité joue un rôle fondamental. En effet pour la plupart des algorithmes de recherche de la solution optimale, la convergence vers un optimum global ne peut être démontrée qu'avec des hypothèses de convexité.

Un problème d'optimisation est convexe, s'il **consiste à minimiser une fonction convexe sur un domaine convexe**, dans ce cas une propriété fondamentale apparaît dans le résultat suivant [45] :

Pour un problème convexe, tout optimum local est un optimum global.

Si de plus ce problème est strictement convexe alors il admet une solution optimale globale unique.

1.3 Convergence, vitesse de convergence

A partir d'un point de départ x^0 , la majorité des méthodes de résolution génèrent une suite de points $\{x^k\}$ convergeant vers une solution optimale du problème d'optimisation.

Un algorithme de résolution est un procédé qui permet à partir de la donnée du point initial x^0, et des informations sur la fonction à minimiser et éventuellement sur les fonctions contraintes d'engendrer la suite $x^1, x^2, \cdots, x^k, \cdots, x^*$.

Cet algorithme est globalement convergeant, si quel que soit le point de départ choisi la suite $\{x^k\}$ qu'il engendre converge vers <u>un point satisfaisant une condition nécessaire d'optimalité.</u>

Cette propriété de convergence globale garantit la sûreté de fonctionnement de l'algorithme. Elle constitue une exigence minimale pour toute méthode de résolution. La programmation mathématique fournit un cadre d'analyse pour établir cette propriété de convergence globale pour de nombreux algorithmes. Elle ne signifie en aucun cas que l'algorithme converge vers un optimum global du problème d'optimisation, seule les hypothèses de convexité permettent de s'en assurer.

Il est également intéressant de caractériser la convergence asymptotique d'un algorithme c'est à dire le comportement de la suite $\{x^k\}$ au voisinage de x^*. Cela permet de définir pour chaque algorithme un indice d'efficacité appelé : **vitesse de convergence**.

Une suite $\{x^k\}$ converge vers x^* à l'ordre *r*, si *r* est le plus grand réel positif tel que :

$$0 \leq \lim_{k \to +\infty} \frac{\left\| x^{k+1} - x^* \right\|}{\left\| x^k - x^* \right\|^r} < +\infty$$

On définit l'erreur asymptotique ,γ par :

$$\gamma = \lim_{k \to +\infty} \frac{\left\| x^{k+1} - x^* \right\|}{\left\| x^k - x^* \right\|^r}$$

Si r=1 on doit avoir $\gamma < 1$ pour qu'il y ait convergence. Pour r=1 et $\gamma < 1$ on parlera de vitesse de convergence linéaire. Quand r=1 et $\gamma = 0$ on définira la vitesse de convergence comme superlinéaire, et superlinéaire d'ordre r lorsque $1 < r < 2$. Enfin, lorsque $r = 2$ la vitesse de convergence sera quadratique.

2 Méthodes pour les problèmes d'optimisation sans fonction contrainte

On cherche à résoudre le problème :

$$(P_l) \left\langle \begin{array}{l} Trouver\ f(x^*) = \underset{x \in R^n}{Min}\ f(x) \\ \text{Où : } f : x \in R^n \longrightarrow f(x) \in R \end{array} \right.$$

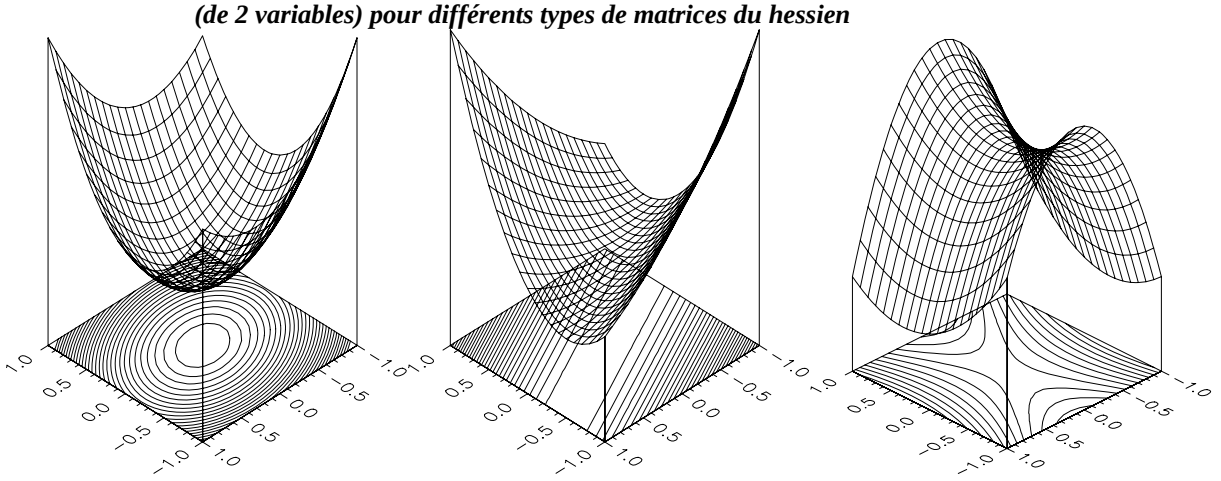
Commençons par énoncer les conditions d'optimalité, c'est à dire les conditions qui caractérisent un minimum global ou local de *f*, sachant que la continuité de *f* et la compacité de R^n nous garantissent ici l'existence d'une solution.

2.1 Conditions suffisantes d'optimalité locale

x^* est un optimum **local** (minimum local) de la fonction *f*, continue et différentiable si et seulement si :	
a)	$\nabla f(x^*) = 0$: x^* est un point stationnaire de f.
b)	Le hessien $\nabla^2 f(x^*)$ est une matrice définie positive : $\forall y \in R^n, \quad y^T . \nabla^2 f(x^*) . y > 0$

La condition b) revient à supposer *f* localement convexe dans un voisinage de x^*, ou encore que toutes les valeurs propres de la matrice du hessien $\nabla^2 f(x^*)$ sont positives.

Figure 3.3: Représentation graphique de plusieurs fonctions quadratiques (de 2 variables) pour différents types de matrices du hessien



(a): Hessien défini positif 2 valeurs propres positives

(c): Hessien semi-défini positif 1 valeur propre positive, 1 nulle

(c): Hessien non défini positif 1 valeur propre positive, 1 négative

Une condition d'optimalité globale ne peut être obtenue qu'avec des hypothèses de convexité sur la fonction *f* on aura donc :

x^* est un optimum global de f sur R^n (minimum global) si :
a) f est convexe
b) $\nabla f(x^*) = 0$: x^* est un point stationnaire

2.2 Structure d'une méthode de minimisation

L'idée de base est qu'à partir d'un point de départ fixé, le point suivant est obtenu par un "déplacement" dans une direction fixée de R^n. Le calcul de cette direction de déplacement d^k et de la valeur du déplacement dans cette direction α^k dépend de la méthode utilisée.
De sorte qu'au cours du processus de recherche, à l'itération *k*, le point suivant x^{k+1} est calculé à partir du point courant x^k par une relation du type :

$$x^{k+1} = x^k + \alpha^k d^k$$

La valeur de déplacement α^k, à l'itération *k*, est telle que la décroissance de la fonction objectif *f* dans la direction d^k soit maximale, on a alors :

$$g(\alpha^k) = Min\{g(\alpha)\}$$
$$\alpha \geq 0$$

Avec :

$$g(\alpha) = f(x^k + \alpha d^k)$$

Il s'agit donc d'une minimisation d'une fonction d'une seule variable, également dénommée optimisation unidimensionnelle ou recherche unidimensionnelle.(Cf figure 3.4)

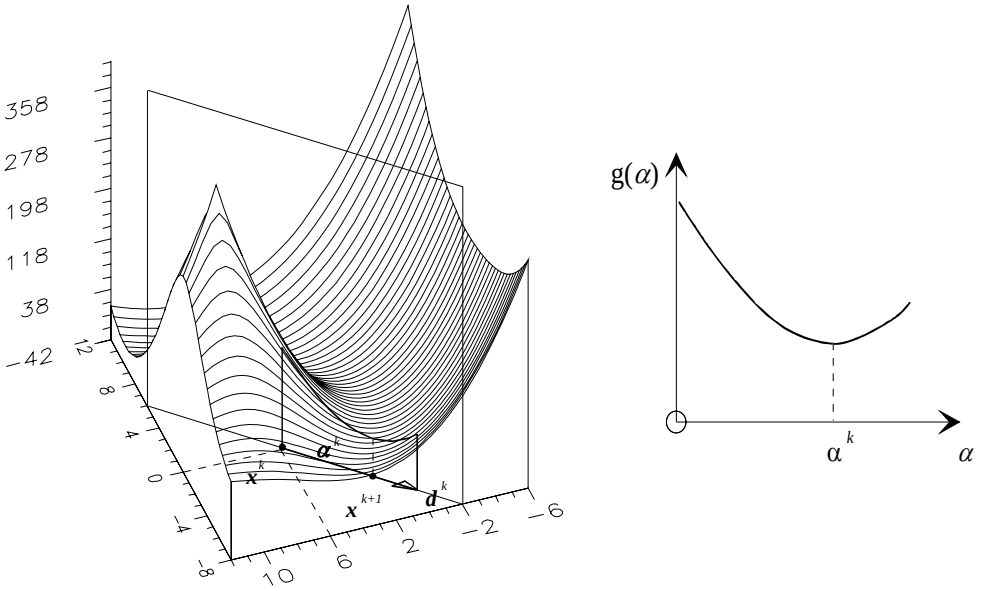


Figure 3.4: Minimisation sans contrainte : schéma de principe.

On obtient alors l'algorithme de principe suivant **commun** à toutes les méthodes de minimisation de fonctions continues et différentiables :

0) Point de départ : $x^0, k \leftarrow 0$
1) Calcul d'une "direction de déplacement" d^k vecteur **non nul** de R^n
2) Recherche unidimensionnelle :
   calcul de $g(\alpha^k) = \underset{\alpha \geq 0}{Min}\{g(\alpha)\}$ avec $g(\alpha) = f(x^k + \alpha d^k)$
3) $x^{k+1} = x^k + \alpha^k d^k$
4) Critère d'arrêt satisfait ?
   OUI : x^{k+1} **optimum**
   NON : faire $k \leftarrow k+1$ et retourne en 1)

Les critères d'arrêt seront détaillés ultérieurement, leur rôle est d'assurer l'optimalité de x^{k+1} en effectuant un test sur la stationnarité de x^{k+1}. L'algorithme ci-dessus converge donc vers un point **stationnaire** de la fonction à minimiser, donc dans certains cas (très rares cependant) il est possible ne de pas obtenir un minimum local.

Le calcul de la direction de déplacement d^k utilise fréquemment le gradient de la fonction objectif en x^k et pour quelques méthodes les dérivées secondes. Cependant on impose systématiquement la condition suivante :

d^k : Direction de "descente" pour la fonction f en x^k, d'où la condition :

$$\nabla^T f(x^k).d^k \leq 0$$

Cette condition est importante, elle doit être remplie à chaque itération de l'algorithme. Elle permet de s'assurer de la décroissance monotone de la fonction objectif, hypothèse nécessaire pour établir la propriété de convergence globale de nombreux algorithmes.

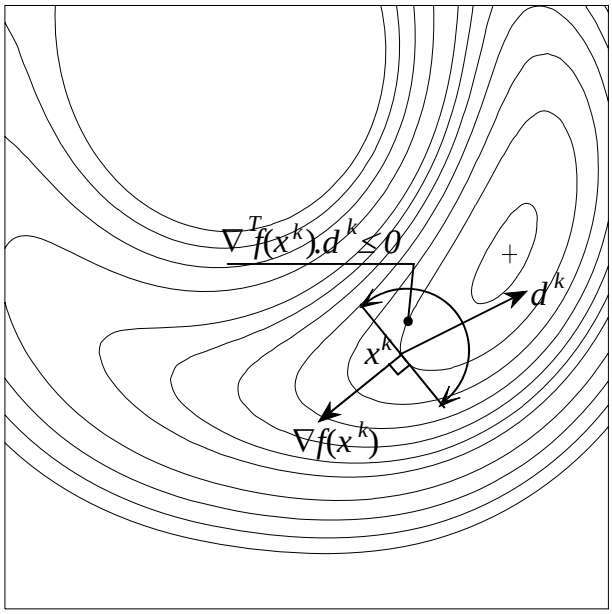


Figure 3.5 : Contour de la surface représentée figure 3.4. Direction de descente : le produit scalaire entre le gradient et la direction de déplacement est négatif.

2.3 Minimum unidimensionnel : interprétation géométrique

On a:

$$g(\alpha) = f(x^k + \alpha d^k)$$

La fonction f est différentiable , donc la fonction $g(\alpha)$ est dérivable et :

$$g'(\alpha) = \nabla^T f(x^k + \alpha d^k).d^k = d^{kT}.\nabla f(x^k + \alpha d^k)$$
$$g''(\alpha) = d^{kT}.\nabla^2 f(x^k + \alpha d^k).d^k$$

Sachant que :

$$g(\alpha^k) = \underset{\alpha \ge 0}{Min}\{g(\alpha)\}$$

On a:

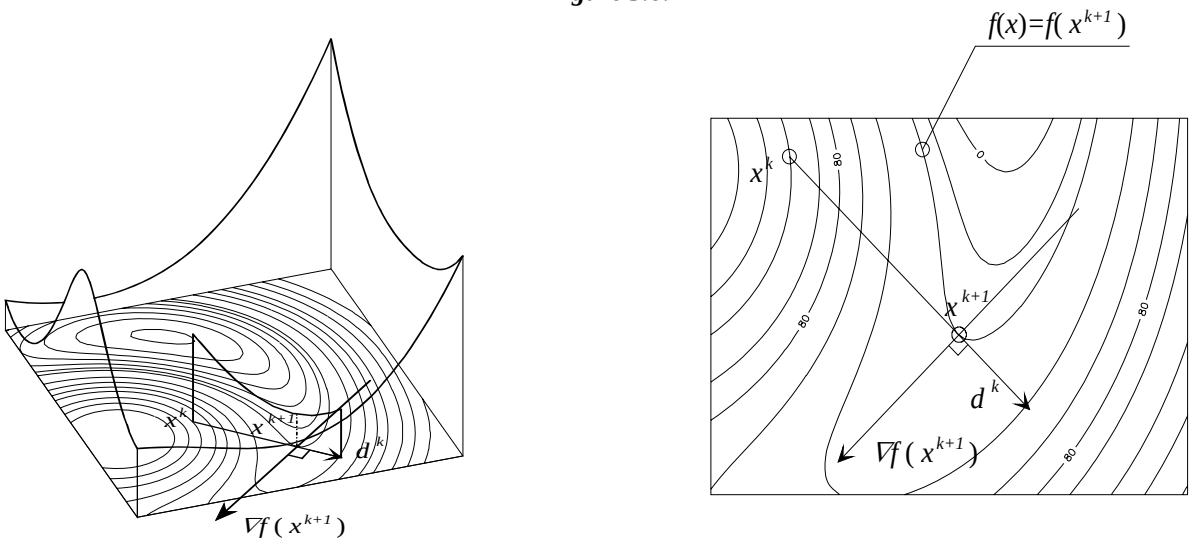
$$g'(\alpha^k) = 0 \Rightarrow d^{kT}.\nabla f(x^k + \alpha^k d^k) = 0 \Rightarrow d^{kT}.\nabla f(x^{k+1}) = 0$$

Deux cas peuvent se présenter :

- Soit x^{k+1} est un point stationnaire et on a : $\nabla f(x^{k+1}) = 0$
- Soit le gradient de f en x^{k+1}, $\nabla f(x^{k+1})$, et la direction de déplacement d^k sont orthogonaux.

La dernière situation est illustrée sur la figure 3.6.

Figure 3.6:



(a): Orthogonalité entre le gradient et la direction de déplacement.

(b): Tangence de la direction de déplacement et du contour de la fonction en x^{k+1}.

On notera que sur la figure 3.6b, x^{k+1} est tel que la direction de déplacement est tangente au contour de la fonction à minimiser en x^{k+1}. En ce point, minimum de la fonction dans la direction d^k, le gradient est orthogonal à d^k.

On remarquera enfin, qu'ayant à chaque itération :

$$\nabla^T f(x^k).d^k \leq 0$$

cela conduit à :

$$g'(0) = \nabla^T f(x^k).d^k \leq 0 .$$

La pente à l'origine de la fonction *g* est toujours négative, comme on le constate sur la figure 3.6a.

2.4 Recherche du minimum d'une fonction d'une variable

Les algorithmes de calcul de α^k sont eux-mêmes des procédures itératives, et sont inclus dans l'algorithme décrit au paragraphe 2.2 . Ce calcul est effectué à chaque itération, d'où la nécessité de techniques performantes capables de déterminer α^k avec le minimum d'itérations donc avec le plus petit nombre possible d'évaluations de la fonction à minimiser. On peut schématiquement diviser ces méthodes de calcul en deux catégories :

- Celle qui utilise la fonction à minimiser $g(\alpha)$ et sa dérivée $g'(\alpha)$.
- Celle qui nécessite seulement le calcul de $g(\alpha)$.

Le choix de l'une de ces catégories de méthodes se fera fonction de la facilité de calcul de $g'(\alpha)$ et de son coût d'évaluation. En effet, fréquemment le gradient de la fonction à minimiser n'est pas fournit explicitement à l'algorithme, mais doit être calculé par une méthode de différences finies. Pour certaines fonctions (beaucoup de variables, forme analytique complexe) ce calcul peut être long et coûteux (en terme d'évaluations de fonction). Il paraît alors judicieux d'utiliser une méthode ne nécessitant pas le calcul de $g'(\alpha)$.

2.4.1 Méthodes utilisant la dérivée

La recherche du minimum de $g(\alpha) = f(x^k + \alpha d^k)$ se ramène à la recherche d'un point stationnaire de $g'(\alpha)$, soit à résoudre l'équation non linéaire : $g'(\alpha) = 0$.

On peut appliquer la méthode classique de **Newton-Raphson** :

Dans ce cas $g'(\alpha)$ est approximée par sa tangente au point courant du processus α^j. Le point suivant α^{j+1} est défini comme l'intersection entre l'axe $[0,\alpha[$ et cette tangente.

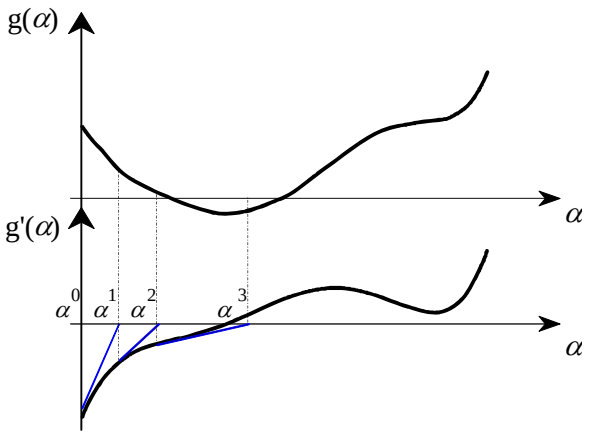


Figure 3.7: Principe de la méthode de Newton-Raphson

On obtient alors :

$$\alpha^{j+1} = \alpha^j - \frac{g'(\alpha^j)}{g''(\alpha^j)}$$

Il est intéressant de remarquer que cette méthode converge en une seule itération lorsqu'elle est appliquée sur une fonction quadratique de la forme :

$$g(\alpha) = u\alpha^2 + v\alpha + w \text{ avec } (u>0)$$

Dans ce cas on a:

$$\alpha^1 = \alpha^0 - \frac{2u\alpha^0 + v}{2u} = -\frac{v}{2u}$$

La nécessité de calculer $g''(\alpha)$ et l'absence de propriété de convergence globale (si $|g''(\alpha)| \to 0, \alpha^{j+1} \to -\infty$) rendent l'utilisation de cette méthode délicate sur des fonctions quelconques. On peut tout de même établir que la vitesse de convergence est **quadratique** au voisinage du minimum α^k si la fonction g (donc la fonction f) satisfait une condition de Lipschitz du genre (la dérivée seconde est "bornée") [45]:

$$|g''(\alpha) - g''(\beta)| \le c|\alpha - \beta|, \quad \alpha, \beta, c \in R \; c > 0$$

Lorsque la dérivée seconde est trop complexe à calculer, on peut éviter ce calcul en remplaçant $g''(\alpha)$ par l'approximation suivante :

$$g''(\alpha^j) = \frac{g'(\alpha^j) - g'(\alpha^{j-1})}{\alpha^j - \alpha^{j-1}}$$

La formule de Newton-Raphson devient alors :

$$\alpha^{j+1} = \alpha^j - g'(\alpha^j)\frac{(\alpha^j - \alpha^{j-1})}{g'(\alpha^j) - g'(\alpha^{j-1})}$$

On obtient ainsi la méthode de la **sécante**, dans laquelle $g'(\alpha)$ n'est plus approximée par la tangente mais par la droite passant par les points $(\alpha^{j-1}, g'(\alpha^{j-1}))$ et $(\alpha^j, g'(\alpha^j))$.

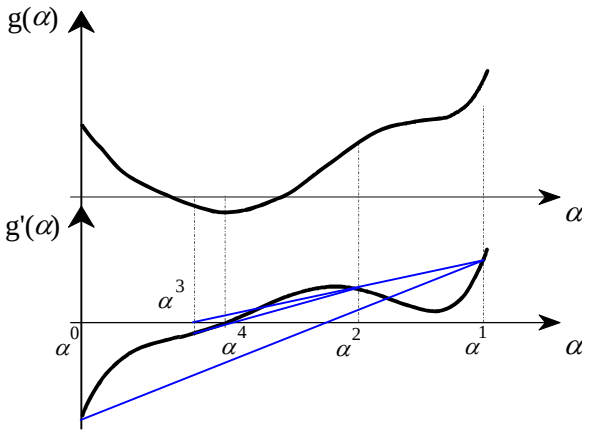


Figure 3.8 : Principe de la méthode de la sécante

Comme la méthode de Newton-Raphson, elle ne possède pas de propriété de convergence globale lorsqu'elle est appliquée à des fonctions quelconques. On choisira donc un point de départ proche du minimum α^k. Cette méthode a une vitesse de convergence **superlinéaire** d'ordre $\gamma = 1.618$ si on suppose $g''(\alpha) > 0$ et si g est trois fois continûment différentiable.

2.4.2 Méthodes n'utilisant pas la dérivée

On suppose que l'on connaît un intervalle de départ $[\alpha^1, \alpha^2]$ encadrant le minimum α^k. Le principe de ces méthodes est la réduction de cet intervalle jusqu'à une certaine longueur, définissant ainsi la précision de la solution obtenue. La seule hypothèse nécessaire sur la fonction à minimiser est celle d'unimodalité sur l'intervalle de départ.

> La fonction g est unimodale sur l'intervalle $[\alpha^1, \alpha^2]$ si elle admet un minimum $\alpha^k \in [\alpha^1, \alpha^2]$ et si $\forall (\beta, \delta) \in [\alpha^1, \alpha^2]$ avec $\beta < \delta$ on a:
>
> $$\delta \leq \alpha^k \Rightarrow g(\delta) > g(\beta)$$
> $$\beta \geq \alpha^k \Rightarrow g(\beta) > g(\delta)$$

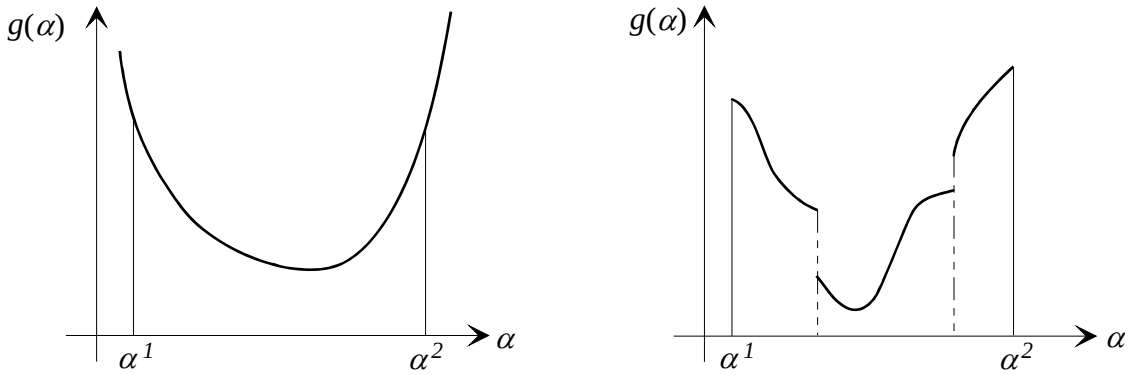


Figure 3.9 : Exemple de fonctions unimodales

Le principe de base de ces méthodes se trouve résumé par les trois schémas de la figure 3.10 :

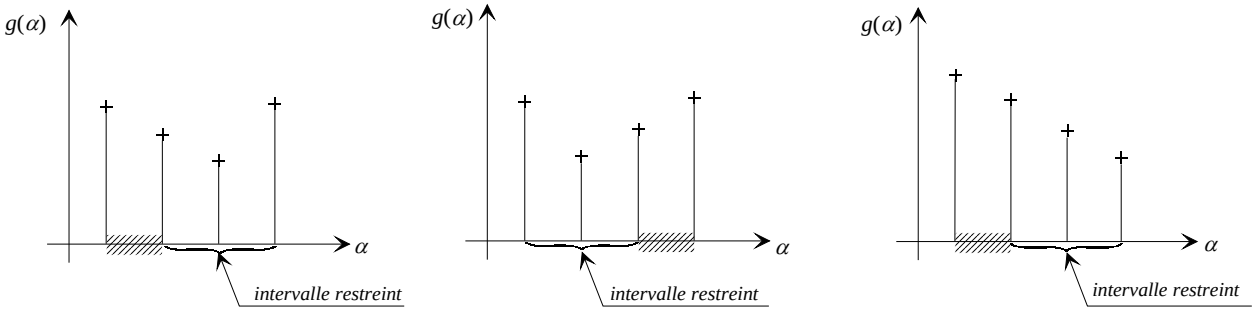


Figure 3.10: Principe de réduction de l'intervalle de départ

En divisant l'intervalle initial en plusieurs sous intervalles (minimum 3) on peut, en utilisant la propriété d'unimodalité, toujours éliminer un ou plusieurs sous intervalles parce que l'optimum ne peut y être situé.

- **Méthode de dichotomie:**

Au départ la fonction *g* est évaluée en α_1, α_2 et au point milieu de l'intervalle $[\alpha_1, \alpha_2]$, $\alpha_3 = (\alpha_1 + \alpha_2)/2$. En divisant ensuite ces 2 intervalles, on obtient 4 sous intervalles de longueur $(\alpha_2 - \alpha_1)/4$. En utilisant la propriété d'unimodalité, on constate qu'il est toujours possible d'éliminer 2 de ces 4 sous intervalles pour conserver 2 intervalles contigus contenant la solution. Le processus est alors réitéré sur l'intervalle restant de longueur, $2(\alpha_2 - \alpha_1)/4$, jusqu'à l'obtention d'un intervalle suffisamment petit.

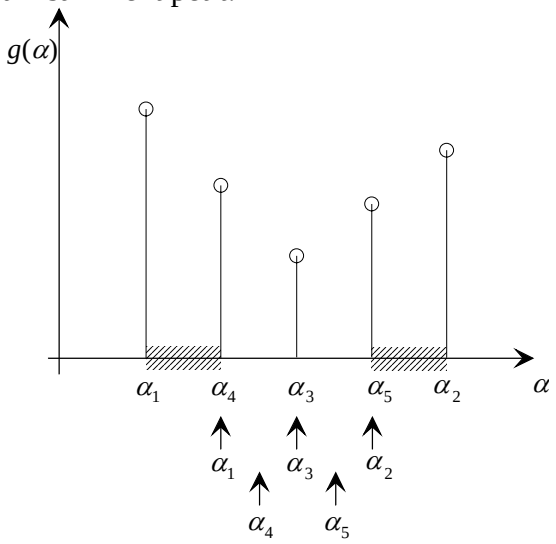


Figure 3.11 : Dichotomie schéma de principe

Au bout de *k* évaluations de fonctions, l'intervalle de départ est réduit d'un rapport de : $2^{(k-3)/2}$. La convergence globale de la méthode pour les fonctions unimodales découle de la définition d'unimodalité, et on constate une vitesse de convergence linéaire avec un taux de :

$$\gamma = 1/\sqrt{2} \approx 0.707$$

La méthode de dichotomie n'est pas optimale dans la mesure où pour un nombre de calculs fixés elle ne donne pas un intervalle final de longueur minimale.

- **Méthode utilisant la suite de Fibonacci:**

Considérons un intervalle de départ $[a_1, d_1]$ de longueur Δ_1, encadrant le minimum α^k. On choisit 2 points b_1 et c_1 dans cet intervalle, répartis symétriquement par rapport au milieu de $[a_1, d_1]$.

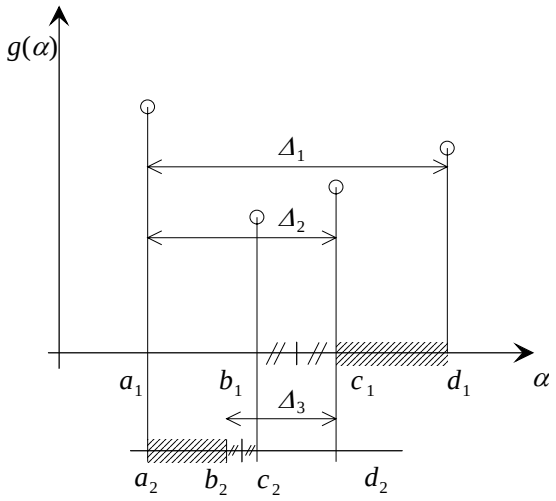


Figure 3.12 : Méthode de la suite de Fibonacci

On pose : $c_1 - a_1 = d_1 - b_1 = \Delta_2$. En évaluant la fonction g en b_1 et c_1, et en utilisant l'unimodalité on peut toujours éliminer un de ces 3 sous intervalles, de sorte qu'il reste toujours un intervalle de longueur Δ_2. Pour fixer les idées, supposons que $[c_1, d_1]$ soit éliminé (cf. figure 3.12). Si on recommence l'opération sur $[a_1, c_1]$ on devra calculer la fonction g au point b_2 qui ne peut être que le symétrique de b_1 par rapport au milieu de $[a_1, c_1]$ (renommé $[a_2, d_2]$ pour pouvoir réitérer le processus). De la même façon on obtient un intervalle réduit $[a_2, c_2]$ ou $[b_2, d_2]$ de longueur : $c_2 - a_2 = d_2 - b_2 = \Delta_3$.

On constate que l'on a :

$$\Delta_1 = (c_1 - a_1) + (d_1 - c_1) = (c_1 - a_1) + (b_1 - a_1)$$

et donc que :

$$\Delta_1 = \Delta_2 + \Delta_3.$$

A l'itération j du processus on déduit que :

$$\Delta_j = \Delta_{j+1} + \Delta_{j+2}.$$

Soit Δ_{k-1} la longueur de l'intervalle obtenu au bout de k calculs de la fonction g, et F_{k-1} le rapport de réduction entre l'intervalle initial Δ_1 et l'intervalle final Δ_{k-1}. On obtient les relations suivantes :

$$\Delta_1 = F_{k-1}\Delta_{k-1} \ , \ \Delta_2 = F_{k-2}\Delta_{k-1} \ , \ \cdots \ , \ \Delta_j = F_{k-j}\Delta_{k-1} \ , \ \Delta_{k-1} = F_1\Delta_{k-1}$$

$$\Rightarrow F_1 = 1$$

Comme :

$$\Delta_j = \Delta_{j+1} + \Delta_{j+2}$$

on a :

$$F_{k-j} = F_{k-j-1} + F_{k-j-2}.$$

Pour avoir Δ_{k-1} minimal, il faut F_{k-1} maximal donc F_2 maximal.
Sachant que :

$$\Delta_{k-2} = F_2 \Delta_{k-1} \Rightarrow F_2 = {\Delta_{k-2}}/{\Delta_{k-1}} \leq 2$$

On prendra : $F_2 = 2$.

La suite des nombres F_j, suite de *Fibonacci* (1202) est alors entièrement définie :

$$\begin{cases} F_1 = 1, F_2 = 2 \\ F_j = F_{j+1} + F_{j+2} \text{ pour } j \geq 3 \end{cases}$$

Par exemple si on désire obtenir un intervalle final 10^4 plus petit que l'intervalle initial, on choisira *k* tel que :

$$\frac{\Delta_1}{10^4 \Delta_1} = \frac{1}{F_{k-1}} \Rightarrow F_{k-1} \geq 10^4 \Rightarrow k = 21$$

Le rapport de réduction de chaque intervalle est alors déterminé par la suite de Fibonacci et :

$$\frac{\Delta_1}{\Delta_2} = \frac{10946}{6725} \;\; ; \;\; \frac{\Delta_2}{\Delta_3} = \frac{6765}{4181} \;\; ; \cdots ; \frac{\Delta_{k-2}}{\Delta_{k-1}} = \frac{2}{1}$$

La nécessité de connaître la précision recherchée au départ de la recherche, et de fixer cette précision non pas sur l'écart relatif entre 2 valeurs consécutives de la fonction *g* mais sur la longueur de l'intervalle final peut être gênant pour des fonctions mal conditionnées et très "irrégulières". La méthode suivante est alors préférable.

k	F_k	k	F_k
1	1	11	144
2	2	12	233
3	3	13	377
4	5	14	610
5	8	15	987
6	13	16	1597
7	21	17	2584
8	34	18	4181
9	55	19	6765
10	89	20	10946

Tableau des 20 premiers termes de la suite de Fibonacci

- **Méthode du nombre d'or:**

La technique de découpage de l'intervalle reste identique à la précédente, mais le rapport entre 2 intervalles est fixé.

D'où :

$$\frac{\Delta_1}{\Delta_2} = \frac{\Delta_3}{\Delta_4} = \cdots = \frac{\Delta_j}{\Delta_{j+1}} = \chi$$

On conserve la relation :

$$\Delta_j = \Delta_{j+1} + \Delta_{j+2}$$

qui devient :

$$\frac{\Delta_j}{\Delta_{j+1}} = 1 + \frac{\Delta_{j+2}}{\Delta_{j+1}} \Rightarrow \chi = 1 + \frac{1}{\chi} \Rightarrow \chi^2 - \chi - 1 = 0$$

La racine positive de cette équation est :

$$\chi = \frac{\sqrt{5}+1}{2} \approx 1.61803 \text{ : Le nombre d'or.}$$

Donc la réduction d'intervalle obtenue après *k* évaluations de fonctions est :$(1/\chi)^{k-3}$, il en résulte une vitesse de convergence linéaire avec un taux de :$1/\chi \approx 0.618$. Bien qu'elle ne soit pas optimale, dans la mesure où pour un nombre d'évaluations de fonction donné, elle ne donne pas le plus petit intervalle final, la méthode du nombre d'or est néanmoins très proche de la méthode utilisant la suite de Fibonacci. De plus on a:

$$\lim_{k\to\infty} \frac{F_k}{F_{k-1}} = \chi$$

De sorte que l'on préférera quasi systématiquement la méthode du nombre d'or aux autres méthodes pour la recherche du minimum de $g(\alpha)$. L'hypothèse d'unimodalité est très peu restrictive, elle ne nécessite ni la continuité, ni la dérivabilité de la fonction à minimiser, d'où la grande fiabilité de ces méthodes que l'on pourra appliquer sur des fonctions très mal conditionnées.

2.4.3 Interpolation, approximations polynomiales

Connaissant les valeurs de la fonction $g(\alpha)$ en plusieurs points, on peut approximer la fonction $g(\alpha)$ par un polynôme ayant les mêmes valeurs que $g(\alpha)$ aux points d'évaluations. Les coefficients du polynôme peuvent être calculés grâce aux valeurs de $g(\alpha)$, mais également grâce à la dérivée $g'(\alpha)$. Le tableau ci après regroupe les possibilités les plus fréquemment employées :

Polynômes de degré 2 : *Interpolation quadratique.*	Polynômes de degré 3 : *Interpolation cubique.*
2 points d'évaluation: $\left[\alpha_j, g(\alpha_j)\right] \quad j = 1,2$ + dérivée : $g'(\alpha)$	3 points d'évaluation: $\left[\alpha_j, g(\alpha_j)\right] \quad j = 1,3$ + dérivée : $g'(\alpha)$
3 points d'évaluation	4 points d'évaluation

On peut alors approximer le minimum de la fonction $g(\alpha)$ par le minimum du polynôme. Cette approximation sera d'autant meilleure que l'écart entre la valeur du polynôme et celle de la fonction pour ce minimum approché sera faible.

Ces techniques peuvent s'utiliser de deux manières :

1) Pour améliorer la précision des méthodes comme celle du nombre d'or ou de dichotomie. Dans ce cas on applique en premier quelques itérations de la méthode du nombre d'or pour terminer par une approximation polynomiale en utilisant les informations sur le dernier intervalle calculé. Pour des fonctions suffisamment régulières (comportement quadratique au voisinage du minimum) on obtient de très bons résultats.

2) En tant que méthode de recherche du minimum à part entière : a partir d'un intervalle de départ sur lequel la fonction est unimodale, et de trois évaluations de fonctions en $(\alpha_1, \alpha_2, \alpha_3)$ on peut calculer une interpolation quadratique $q(\alpha)$. Si α_4, le minimum de $q(\alpha)$ convient comme approximation de l'optimum de $g(\alpha)$ le processus est stoppé, sinon l'interpolation suivante est recalculée avec les 3 nouveaux points, $(\alpha_1', \alpha_2', \alpha_3')$:

$$
\begin{aligned}
&= (\alpha_2, \alpha_4, \alpha_3) \text{ si } \alpha_2 \le \alpha_4 \le \alpha_3 \text{ et } g(\alpha_4) \le g(\alpha_2) \\
&= (\alpha_1, \alpha_2, \alpha_4) \text{ si } \alpha_2 \le \alpha_4 \le \alpha_3 \text{ et } g(\alpha_4) > g(\alpha_2) \\
&= (\alpha_1, \alpha_4, \alpha_2) \text{ si } \alpha_1 \le \alpha_4 \le \alpha_2 \text{ et } g(\alpha_4) \le g(\alpha_2) \\
&= (\alpha_4, \alpha_2, \alpha_3) \text{ si } \alpha_1 \le \alpha_4 \le \alpha_2 \text{ et } g(\alpha_4) > g(\alpha_2)
\end{aligned}
$$

On peut établir la convergence globale de cette méthode si la fonction g est continue et unimodale sur l'intervalle de départ. En supposant g trois fois continûment différentiable, on montre que la vitesse est superlinéaire d'ordre $\gamma = 1.3$ [45].

2.4.4 Choix d'un intervalle de départ

Des méthodes comme celles du nombre d'or ou de la sécante nécessitent la connaissance d'un intervalle de départ dans lequel se situe α^k. La borne inférieure est facile à déterminer puisqu'il s'agit généralement de $\alpha = 0$.

Lorsque la dérivée de g , $g'(\alpha)$ est disponible on peut, par exemple, envisager le choix suivant pour la borne supérieure $\alpha_{\max}$:

$$\alpha_{\max} = \frac{|0.1g(0)|}{|g'(0)|}$$

La figure ci-dessous donne une interprétation graphique de ce choix :

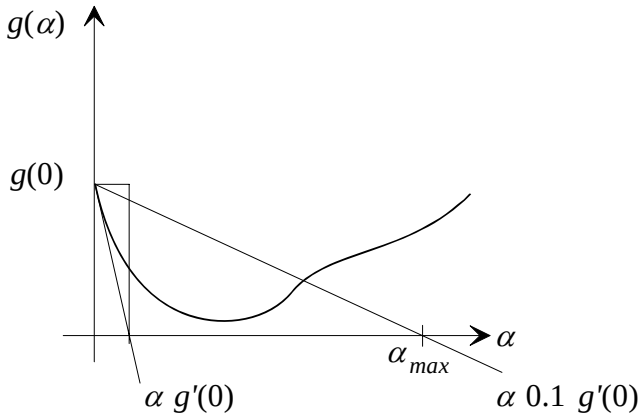


Figure 3.13 : Calcul d'un intervalle de départ

Dans le cas contraire, lorsque $g'(\alpha)$ n'est pas disponible, on peut choisir pour $\alpha_{\max}$ la distance estimée entre le point de départ x^0 de la minimisation sans contrainte et l'optimum x^*. L'application de quelques itérations de la méthode du nombre d'or conduit rapidement à un intervalle encadrant α^k.

2.4.5 Critère d'arrêt pour la recherche unidimensionnelle

Les tests qui permettent d'arrêter le processus itératif de recherche de α^k, donc de statuer sur la validité du pas de déplacement α^j couramment calculé sont étroitement liés à la méthode dans laquelle ils sont intégrés.

Pour les méthodes n'utilisant pas la dérivée, un test basé sur la longueur du dernier intervalle obtenu sera effectué :

$$\frac{\left|\alpha^{j}-\alpha^{j-1}\right|}{\left|\alpha_2-\alpha_1\right|}<\varepsilon_1$$

Avec $[\alpha_1,\alpha_2]$ intervalle de départ et $[\alpha^{j-1},\alpha^{j}]$ intervalle courant. On notera que ε_1 déterminant la précision de la solution, est fixé par avance au départ des calculs, par conséquent avec la méthode du nombre d'or on pourra déterminer le nombre d'évaluations de fonctions nécessaire pour atteindre cette précision. Ce test se ramène alors à un simple comptage des évaluations de fonctions effectuées.

En définissant l'écart relatif entre les valeurs de $g(\alpha)$ aux bornes de l'intervalle courant par :

$$\delta g=\frac{\left|g(\alpha^{j})-g(\alpha^{j-1})\right|}{1+\left|g(\alpha^{j})\right|}$$

on effectuera également le test :

$$\delta g<\varepsilon_2$$

Les quantités ε_1 et ε_2 petites et positives seront choisies en fonction de l'arithmétique du calculateur et de la précision souhaitée au niveau du minimum..

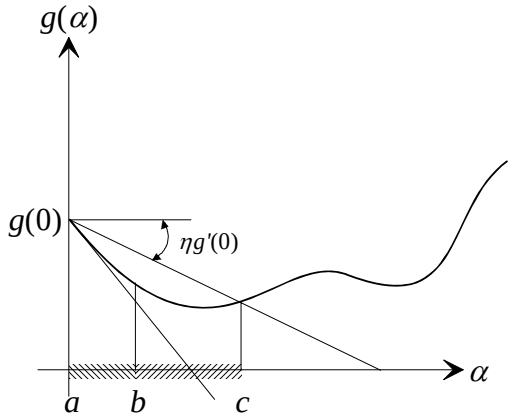


Figure 3.14 : Critères d'arrêt utilisant la dérivée

Lorsque la dérivée de $g(\alpha)$ est disponible, ou facile à évaluer, le test d'arrêt s'effectuera de préférence sur $g'(\alpha)$, donc sur la pente de la courbe représentant $g(\alpha)$, en vérifiant la condition (segment [b,c] sur la figure 3.14) :

$$\left|g'(\alpha^{j})\right|<-\eta g'(0)$$

De plus pour s'assurer d'une décroissance suffisante de la fonction objectif on vérifiera également la condition (segment [a,c] sur la figure 3.14):

$$g(0) - g(\alpha^j) \geq -\mu\alpha^j g'(0)$$

Avec :

$\mu \in]0,1/2]$, $\eta \in [0,1[$ et $\mu \leq \eta$.

Le choix des valeurs de μ et η permettant d'ajuster la précision désirée sur la recherche unidimensionnelle (notion importante pour les méthodes de minimisation sans contrainte de type quasi newtonienne).

2.4.6 Choix d'une méthode : quelques éléments de réponse

L'efficacité d'une méthode de recherche unidimensionnelle est étroitement liée à la fonction que l'on minimise, son conditionnement, la disponibilité et le coût d'évaluation de son gradient, mais également à la méthode de minimisation sans contrainte dans laquelle elle est intégrée.

La précision requise de l'optimum est un bon indicateur de choix : si elle est importante il faudra utiliser en fin de convergence des techniques d'interpolation polynomiale ou appliquer quelques itérations d'une méthode de Newton-Raphson. L'obtention d'une bonne précision avec des méthodes comme celle du nombre d'or conduit à un nombre important d'évaluation de fonctions.

Lorsque le gradient de la fonction à minimiser est disponible, on aura intérêt à utiliser une méthode de type sécante par exemple, car les méthodes utilisant la dérivée offrent une bonne vitesse de convergence : quadratique pour Newton-Raphson, superlinéaire pour la méthode de la sécante. Les méthodes sans dérivées ne permettent que des vitesses de convergence linéaire.

Le conditionnement numérique de la fonction à minimiser est aussi un critère de choix. Si la fonction est très "irrégulière", voire discontinue et donc non dérivable, seule des méthodes basées sur la propriété d'unimodalité seront applicables. Ces méthodes sont très robustes mais procurent une précision moyenne que l'on peut facilement améliorer par une interpolation polynomiale avec les points obtenus en fin de convergence.

2.5 Recherche du minimum d'une fonction de plusieurs variables

Les méthodes que nous allons présenter maintenant s'appliquent aux fonctions continues et différentiables. La structure de l'algorithme est celle définie au paragraphe 2.2. Nous allons détailler la spécificité de chaque méthode c'est à dire la manière de déterminer la direction de recherche d^k permettant de définir la recherche unidimensionnelle.

On a vu que d^k doit être une direction de descente pour la fonction objectif *f*, donc elle vérifie la relation :

$$\nabla^T f(x^k).d^k \leq 0$$

2.5.1 Méthode de la plus forte pente

L'idée la plus naturelle pour déterminer cette direction de descente est de définir d^k comme l'opposé du gradient de *f* en x^k puisque la direction du gradient est celle de la plus forte augmentation de $f(x)$.

D'où l'algorithme :

0) Point de départ : $x^0, k \leftarrow 0$
1) $d^k = -\nabla f(x^k)$
2) Recherche unidimensionnelle :
   calcul de $g(\alpha^k) = \underset{\alpha \geq 0}{Min}\{g(\alpha)\}$ avec $g(\alpha) = f(x^k + \alpha d^k)$
3) $x^{k+1} = x^k + \alpha^k d^k$
4) Critère d'arrêt satisfait ?
   oui : x^{k+1} optimum
   non : faire $k \leftarrow k+1$ et retourne en 1)

Cette méthode possède une propriété de convergence globale que l'on peut établir sous des hypothèses très peu restrictives puisque :

Si *f* est continûment différentiable avec la propriété suivante :

$$\lim_{\|x\| \to \infty} f(x) = +\infty$$

Alors pour tout point de départ x^0 la méthode de la plus forte pente (avec recherche unidimensionnelle exacte ou approchée) converge vers un point stationnaire de *f*. (x^* point stationnaire $\Rightarrow \nabla f(x^*) = 0$)

Comme le précise le résultat ci-dessus, la méthode ne nécessite pas à chaque itération que α^k soit le minimum de $g(\alpha) = f(x^k + \alpha d^k)$, la seule condition est d'avoir :

$$f(x^{k+1}) < f(x^k)$$

On concevra cependant que les performances seront d'autant meilleures que la diminution de la fonction objectif, à chaque itération, sera plus forte. Pour établir la vitesse de convergence de la méthode, on doit supposer que le Hessien de *f* à l'optimum est défini positif, on a alors :

$$\lim_{k \to \infty} \frac{f(x^{k+1}) - f(x^*)}{f(x^k) - f(x^*)} = \gamma \leq \frac{(A-a)^2}{(A+a)^2}$$

où *A* et *a* sont respectivement la plus grande et la plus petite valeur propre de $\nabla^2 f(x^*)$. La vitesse de convergence est donc linéaire et le taux de convergence γ est directement lié au

conditionnement (rapport *A*/*a*) de la matrice $\nabla^2 f(x^*)$. La convergence peut être excessivement lente pour les fonctions mal conditionnées.

2.5.2 Méthodes de directions conjuguées

En écrivant un développement de Taylor au second ordre de la fonction *f* au voisinage d'un minimum local x^* (dans ce cas $\nabla f(x^*) = 0$) on a la relation :

$$f(x) \approx f(x^*) + \frac{1}{2}(x - x^*)^T . \nabla^2 f(x^*) . (x - x^*) + \left\| x - x^* \right\| \theta(x - x^*) \left\| x - x^* \right\|$$
$$\text{Avec : } \lim_{x \to 0} \theta(x) = 0$$

Si x^* satisfait les conditions d'optimalité définies au paragraphe 2.1 le hessien est défini positif. On constate que, en négligeant les termes d'ordre 3, $f(x)$ se comporte comme une fonction strictement quadratique. D'où le postulat suivant : **pour qu'une méthode de minimisation soit efficace sur une fonction quelconque, elle doit l'être au moins sur une fonction quadratique**. Nous allons voir que les méthodes de directions conjuguées ont la particularité de converger vers le minimum d'une fonction quadratique en un nombre fini d'itération.

Soit :

$$q(x) = \frac{1}{2} x^T . A . x + b . x + c$$

où : *A* est une matrice définie positive et symétrique de R^n

$b \in R^n$

$c \in R$.

Les *n* directions $d^0, d^1, \cdots, d^{n-1}$ sont dites conjuguées par rapport à la forme quadratique $q(x)$, ou *A* conjuguées, si :

$$d^{i^T} . A . d^j = 0 \quad \begin{vmatrix} \forall i \in [0, n-1] \\ \forall j \in [0, n-1] \\ i \neq j \end{vmatrix}$$

Remarque :

Si *A*=*I* (matrice identité), la propriété de conjugaison devient une propriété d'orthogonalité.

On montre alors la propriété suivante (démonstration rapportée dans l'annexe 2) :

En supposant que $\forall\ k = 0,1,\ldots,n-2$, x^{k+1} soit déterminé par la relation :

$$x^{k+1} = x^k + \alpha^k d^k$$

Où : $q(\alpha^k) = \underset{\alpha \geq 0}{Min}\, q(x^k + \alpha d^k)$, les directions d^k étant *A* conjuguées.

Le point obtenu après *n* itérations, c'est à dire :

$$x^n = x^0 + \sum_{k=1}^{n-1} \alpha^k d^k$$

est nécessairement le minimum de $q(x)$, donc vérifie :

$$\nabla q(x^n) = 0 \Rightarrow A.x^n + b = 0$$

Appliquée sur une fonction quadratique, une méthode de directions conjuguées, c'est à dire une méthode basée sur l'algorithme du paragraphe 2.2 générant des directions conjuguées par rapport à la forme quadratique de $q(x)$ converge vers l'optimum de $q(x)$ en au plus *n* itérations. (*n* nombre de composantes du vecteur variable *x*).

Le calcul de α^k peut être fait explicitement puisque :

$$q(x^k + \alpha^k d^k) = \underset{\alpha \geq 0}{Min}\, q(x^k + \alpha d^k)$$
$$\Rightarrow {d^k}^T.\nabla q(x^k + \alpha d^k) = 0 \Rightarrow {d^k}^T.\left[A.(x^k + \alpha^k d^k)\right] = 0$$

On obtient

$$\alpha^k = \frac{d^k.(A.x^k + b)}{{d^k}^T.A.d^k}$$

De nombreuses possibilités existent pour la détermination des directions conjuguées $(d^0, d^1, \ldots, d^{n-1})$, permettant d'imaginer beaucoup d'algorithmes basés sur ce principe. Parmi ces possibilités nous présenterons deux méthodes très fréquemment utilisées : la méthode du gradient conjugué pour les fonctions quadratiques et son extension au cas des fonctions quelconques : la méthode de *Fletcher* et *Reeves*.

2.5.2.1 Méthode du gradient conjugué pour les fonctions quadratiques.

L'idée est de définir les *n* directions conjuguées $(d^0, d^1, \ldots, d^{n-1})$ nécessaires à la minimisation de $q(x)$ comme une combinaison linéaire du gradient de $q(x)$ et des directions précédentes :

A l'étape *k* de l'algorithme on calcule la direction de déplacement d^k avec la relation :

$$d^k = -\nabla q(x^k) + \beta^{k-1} d^{k-1} \qquad (1)$$

Pour initialiser le processus on prendra : $\beta^0 = 0$ et $d^0 = -\nabla q(x^0)$.

En utilisant la propriété de conjugaison des directions d^k on détermine immédiatement le scalaire β^k et :

$$\beta^k = \frac{\nabla^T q(x^{k+1}).A.d^k}{d^{kT}.A.d^k} \tag{2}$$

On montre également que l'on peut établir les formules suivantes, toutes équivalentes dans le cas d'une fonction quadratique.

$$\beta^k = \frac{\nabla^T q(x^{k+1}).\left[\nabla q(x^{k+1}) - \nabla q(x^k)\right]}{\nabla^T q(x^k).\nabla q(x^k)} \tag{3}$$

$$= \frac{\nabla^T q(x^{k+1}).\nabla q(x^{k+1})}{\nabla^T q(x^k).\nabla q(x^k)} \tag{4}$$

Le détail des calculs est présenté dans l'annexe 2.

On obtient alors l'algorithme du gradient conjugué pour les fonctions quadratiques :

0) Point de départ : x^0 , $\beta^0 = 0$, $d^0 = -\nabla q(x^0)$ et $k \leftarrow 0$

1) A l'itération k :

Calcul de :

$$d^k = -\nabla q(x^k) + \beta^{k-1} d^{k-1}$$

$$\alpha^k = -\frac{d^k.(A.x^k + b)}{d^{kT}.A.d^k}$$

$$x^{k+1} = x^k + \alpha^k d^k$$

$$\beta^k = \frac{\nabla^T q(x^{k+1}).A.d^k}{d^{kT}.A.d^k}$$

2) Si $\beta^k \neq 0$ faire $k \leftarrow k+1$ et retourner en 1) sinon **FIN**

L'équivalence entre les relations (3) et (4) suppose :

$$\nabla^T q(x^{k+1}).\nabla q(x^k) = 0$$

On peut montrer que $\forall (i, j) \in [0, \ldots, n-1]$ on a :

$$\nabla^T q(x^j).\nabla q(x^i) = 0$$

Les points x^{k+1} engendrés par la méthode sont tels, que les gradients de la fonction quadratique en ces points sont mutuellement orthogonaux.

Appliquée sur des fonctions quadratiques, cette méthode hérite de la propriété des méthodes de directions conjuguées : convergence vers **l'optimum en au plus *n* itérations**.

Ce résultat est théorique, il suppose qu'au cours des itérations successives les directions générées par la relation (1) sont mutuellement conjuguées par rapport à *A*. En fait les erreurs d'arrondi au cours des calculs altèrent les relations des *A* conjugaisons . On établit que le nombre d'itérations nécessaires est proportionnel à $\sqrt{k(A)}$ ou $k(A)$ et le conditionnement[2] de la matrice *A*. Plus ce nombre est proche de 1 (matrice *A* bien "conditionnée") plus l'algorithme convergera vite ([37], [21], [23]).

L'algorithme du gradient conjugué est également utilisé pour résoudre des systèmes linéaires à matrice symétrique et définie positive.

En effet, chercher le minimum de la quadratique $q(x)$ conduit à la résolution du système linéaire :

$$A.x + b = 0 \tag{5}$$

Pour conserver de bonnes performances lorsque cette méthode est appliquée sur des systèmes linéaires dont la matrice *A* est mal conditionnée ($k(A)$ grand) des techniques de préconditionnement sont mises en oeuvres :

Le principe général en est le suivant :
On remplace la résolution de (5) par :

$$C.A.x + C.b = 0$$

La matrice de préconditionnement *C* est telle que : $k(C.A) << k(A)$. Le choix idéal est $C = A^{-1}$ dans ce cas : $k(C.A) = 1$. En pratique il faut trouver la matrice *C* la plus proche possible de l'inverse de *A* sans que les calculs soient trop coûteux. On pourra consulter la référence [37] pour un exposé détaillé de quelques techniques de préconditionnement.

En ce qui concerne les algorithmes d'optimisation, on trouvera dans les références [23] et [24] un algorithme de gradient conjugué préconditionné.

2.5.2.2 Cas des fonctions quelconques.

Cette méthode est une extension directe de la précédente au cas des fonctions quelconques. On définit, de la même façon, la direction d^k par la relation (1). On obtient alors :

$$d^k = -\nabla f(x^k) + \beta^{k-1} d^{k-1}$$

[2]Rapport entre la plus grande et la plus petite valeur propre de la matrice *A*.

D'où l'algorithme du gradient conjugué pour les fonctions quelconques :

0) Point de départ : x^0 , $\beta^0 = 0$, $d^0 = -\nabla q(x^0)$ et $k \leftarrow 0$

1) A l'itération k :
Calcul de :

$$d^k = -\nabla q(x^k) + \beta^{k-1} d^{k-1}$$

$$g(\alpha^k) = \underset{\alpha \geq 0}{Min}\{g(\alpha)\} \text{ avec } g(\alpha) = f(x^k + \alpha d^k)$$

$$x^{k+1} = x^k + \alpha^k d^k$$

$$\beta^k = \frac{\nabla^T f(x^{k+1}).\nabla f(x^{k+1})}{\nabla^T f(x^k).\nabla f(x^k)} = \frac{\left\|\nabla f(x^{k+1})\right\|^2}{\left\|\nabla f(x^k)\right\|^2}$$

2) Si test d'arrêt vérifié **FIN**
sinon faire $k \leftarrow k+1$ et retourner en 1) .

Le calcul de β^k ne se fait plus grâce à la relation (2), la matrice *A*, le hessien de $q(x)$, donc ici le hessien de $f(x)$ n'est plus disponible. Une variante de cet algorithme, connue sous le nom de variante de *Polak Ribière* consiste à utiliser la relation (4) pour le calcul de β^k. En effet dans le cas d'une fonction quelconque les relations (3) et (4) ne sont plus équivalentes car on n'a plus :

$$\nabla^T f(x^{k+1}).\nabla f(x^k) = 0$$

Pour chaque formulation de β^k on obtient, dans le cas général, des résultats différents

La méthode ne converge plus de façon finie, en fait on ne peut établir la convergence globale de cette méthode que si périodiquement on "réinitialise" la direction de déplacement avec l'opposé du gradient de la fonction objectif en x^k. Alors la convergence globale peut s'établir avec les mêmes hypothèses que celles de la méthode de la plus forte pente.

Plusieurs stratégies de réinitialisation sont possibles: *Fletcher* [20] propose de réinitialiser la direction d^k toutes les *n* étapes, *Vanderplaats* [67] suggère de réinitialiser d^k lorsque d^k n'est plus une direction de descente, donc quand :

$$\nabla^T f(x^k).d^k \geq 0$$

On trouve dans [20] plusieurs expérimentations numériques de cette méthode sur diverses fonctions, les résultats de cette étude montrent que l'utilisation de la relation (4) procure globalement les meilleurs résultats. Ils montrent également que la précision de la recherche unidimensionnelle est importante pour la rapidité de convergence.

En effet si le hessien $\nabla^2 f(x^*)$ est défini positif, la méthode de *Flécher Revers* et sa variante de *Polak Ribière* ont une vitesse de convergence superlinéaire sur *n* étapes. Lorsqu'on procède à une réinitialisation toutes les *n* étapes. On a alors :

$$\frac{\left\|x^{k+1} - x^*\right\|}{\left\|x^k - x^*\right\|} \to 0 \text{ lorsque } k \to +\infty$$

Cette vitesse de convergence ne peut être établie que si :

$$\frac{\left\|\nabla^T f(x^{k+1}).(x^{k+1} - x^k)\right\|}{\left\|\nabla^T f(x^{k+1})\right\|.\left\|(x^{k+1} - x^k)\right\|} \to 0 \text{ lorsque } k \to +\infty$$

Donc si le gradient en x^{k+1} tend à être orthogonal à la direction de déplacement $:\alpha^k d^k = (x^{k+1} - x^k)$. La figure 6b § 2.3 montre que cette nécessité conduit à une bonne précision dans le calcul de α^k.

Il est intéressant de noter que cette méthode nécessite pour chaque itération le stockage de peu d'information. Seulement 3 vecteurs de dimension *n*: les gradients de $f(x)$ en x^k et x^{k+1} et la direction de déplacement courante d^k

2.5.3 Méthode de Newton

En généralisant la méthode de Newton-Raphson au cas d'une fonction de plusieurs variables de R^n dans *R* on obtient la relation suivante :

$$x^{k+1} = x^k - \left[\nabla^2 f(x^k)\right]^{-1}.\nabla f(x^k) \tag{6}$$

On constate que x^{k+1} est le minimum d'une fonction quadratique $q(x)$ qui s'écrit :

$$q(x) = f(x^k) + \nabla^T f(x^k).(x - x^k) + \frac{1}{2}(x - x^k)^T.\nabla^2 f(x^k).(x - x^k) \tag{7}$$

En effet $q(x)$ est minimum pour :

$$\nabla^2 f(x^k).(x - x^k) + \nabla f(x^k) = 0 \tag{8}$$

La relation (6) définit donc bien x^{k+1} comme solution de (8).

L'expression (7) n'est autre que le développement de Taylor de $f(x)$ au second ordre au voisinage de x^{k+1} dans lequel les termes d'ordre supérieur ont été négligés. On remarque alors que l'expression (6) définit également x^{k+1} comme le minimum de la fonction quadratique

approximant $f(x)$ en x^k. Il apparaît clairement que le pas de déplacement α^k et la direction de déplacement d^k sont fixés puisque (6) peut également s'écrire :

$$\alpha^k d^k = (x^{k+1} - x^k) = -\left[\nabla^2 f(x^k)\right]^{-1}.\nabla f(x^k)$$

On sait que la condition de descente sur d^k impose :

$$\nabla^T f(x^k).d^k \leq 0$$

relation qui doit être vérifiée pour toute itération k le l'algorithme. On en déduit alors que l'on doit avoir :

$$\forall y \in R^n; \forall x^k \quad : \quad y^T.\nabla^2 f(x^k).y > 0 \quad \Rightarrow \quad \nabla^2 f(x^k) \text{ défini positif}^3.$$

La convergence globale de la méthode de Newton n'est donc assurée que pour des fonctions strictement convexes. Dans le cas des fonctions quelconques on appliquera cette méthode localement, dans le voisinage du minimum x^* en s'assurant que le hessien à l'optimum est défini positif. Appliquée sur une fonction quadratique la méthode newton converge en une seule itération, cependant cela nécessite l'inversion de la matrice du hessien, donc la résolution d'un système linéaire. L'utilisation d'une méthode de directions conjuguées évite cette opération d'inversion.

L'approximation de $f(x)$ par une quadratique n'étant valable qu'un voisinage de x^k on peut modifier la relation (1) en limitant le déplacement dans la direction : $-\left[\nabla^2 f(x^k)\right]^{-1}.\nabla f(x^k)$ en introduisant α^k tel que :

$$x^{k+1} = x^k - \alpha^k\left[\nabla^2 f(x^k)\right]^{-1}.\nabla f(x^k) \tag{9}$$

$$\text{Avec } g(\alpha^k) = \underset{\alpha \geq 0}{Min}\{g(\alpha)\} \text{ et } g(\alpha) = f\left(x^k - \alpha\left[\nabla^2 f(x^k)\right]^{-1}.\nabla f(x^k)\right)$$

Lorsque le hessien n'est plus défini positif on peut imaginer de perturber la matrice du hessien en la remplaçant par une matrice de la forme [45] :

$$G^k = \mu^k I + \nabla^2 f(x^k) \text{ avec } I \text{ matrice identité}$$

Perturbation dans laquelle le réel μ^k est tel que, toutes les valeurs propres de G^k soient positives.

Pour cette méthode de Newton modifiée on peut établir une propriété de convergence globale lorsque μ^k est suffisamment grand. En effet pour μ^k très grand les termes de $\nabla^2 f(x^k)$ sont alors négligeables et on retrouve la méthode de la plus forte pente. Même implantée sous cette forme la méthode de Newton n'est guère praticable. L'inversion à chaque itération du hessien de la fonction à minimiser, la nécessité de disposer des dérivées secondes, leur coût

[3]L'inverse d'une matrice définie positive est aussi une matrice définie positive.

d'évaluation conduisent à des temps de calcul prohibitifs pour des problèmes avec beaucoup de variables.

2.5.4 Méthodes quasi newtoniennes : principe général [45]

L'idée de base est de généraliser la relation (9) en remplaçant l'inverse du hessien de $f(x)$ par une matrice définie positive H^k. D'où une formule du type :

$$x^{k+1} = x^k - \alpha^k H^k . \nabla f(x^k) \tag{10}$$

$$\text{Avec } g(\alpha^k) = \underset{\alpha \geq 0}{Min}\{g(\alpha)\} \text{ avec } g(\alpha) = f\left(x^k - \alpha H^k . \nabla f(x^k)\right)$$

La matrice H^k étant par définition définie positive la direction $d^k = -H^k . \nabla f(x^k)$ est nécessairement une direction de descente pour la fonction $f(x)$. On pourrait donc, a priori, choisir n'importe quelle forme pour la matrice H^k, à condition qu'elle soit définie positive. On remarquera que pour $H^k = I$, la relation (10) donne le même résultat que la méthode de la plus forte pente.

Pour obtenir de bonnes performances on devra choisir une matrice H^k aussi proche que possible de l'inverse du hessien de $f(x)$, **donc il sera nécessaire de modifier H^k à chaque itération.** Considérons à nouveau le développement de Taylor au second ordre de $f(x)$ en x autour de x^k :

$$f(x) = f(x^k) + \nabla^T f(x^k).(x - x^k) + \frac{1}{2}(x - x^k)^T . \nabla^2 f(x^k).(x - x^k) + \ldots$$

En dérivant, et en négligeant les termes d'ordre trois (nul pour une quadratique) on obtient :

$$\nabla f(x) = \nabla f(x^k) + \nabla^2 f(x^k).(x - x^k)$$

En particulier, au point x^{k+1} défini par (7) cette relation donne :

$$\left[\nabla^2 f(x^k)\right]^{-1} . \left[\nabla f(x^{k+1}) - \nabla f(x^k)\right] = (x^{k+1} - x^k)$$

Donc si H^k constitue une approximation de l'inverse du hessien , elle doit vérifier :

$$H^k . \left[\nabla f(x^{k+1}) - \nabla f(x^k)\right] \approx (x^{k+1} - x^k)$$

Plus précisément, compte tenu des erreurs d'approximation on écrira :

$$\left[H^k + D^k\right] . y^k = s^k \tag{11}$$

En posant:

$$\begin{cases} y^k = \nabla f(x^{k+1}) - \nabla f(x^k) \\ s^k = x^{k+1} - x^k = \alpha^k d^k \end{cases}$$

Avec D^k matrice de correction définie positive [66].

On obtient alors l'algorithme général d'une méthode quasi newtonienne :

0) Point de départ : x^0, H^0 définie positive et $k \leftarrow 0$
1) A l'itération k :
   Calcul de :
   $$d^k = -H^k \nabla f(x^k)$$
   $$g(\alpha^k) = \underset{\alpha \geq 0}{Min}\{g(\alpha)\} \text{ avec } g(\alpha) = f(x^k + \alpha d^k)$$
   $$x^{k+1} = x^k + \alpha^k d^k$$
2) Si test d'arrêt vérifié **FIN**
3) sinon calcul de D^k
   Faire $H^{k+1} = H^k + D^k$ et $k \leftarrow k+1$ puis retourner en 1) .

La matrice de départ H^0 peut être choisie quelconque, cependant le choix le plus simple impose $H^0 = I$.
Le calcul de D^k doit être tel que :

- Les matrices H^k restent définies positives pour toute itération *k*. Ce qui permet de conserver d^k comme direction de descente.
- La relation (11) soit satisfaite, c'est elle qui assure que H^k est une approximation de l'inverse du hessien de la quadratique approximant la fonction à minimiser. Cette relation est souvent appelée : condition quasi newtonienne.

2.5.4.1 Correction de rang 1.

Le moyen le plus simple consiste à définir la matrice D^k comme une matrice de rang 1. Le choix d'une matrice de correction de rang 1 est justifié dans [24] de la manière suivante :

Etant donné que l'on minimise $f(x)$ dans une direction de R^n, les informations sur la courbure de $f(x)$ ne peuvent être obtenues que dans cette direction. Par conséquent, l'ajout d'une matrice de rang 1, déterminée grâce à un seul vecteur devrait suffire pour corriger les erreurs d'approximation sur H^k.

On choisit une matrice D^k de la forme :

$$D^k = \lambda^k u^k . u^{k^T} \text{ où } \lambda^k \in R \text{ et } u^k \in R^n$$

Dont le terme général *i,j* est : $\lambda^k u_i^k . u_j^k$. Par construction D^k est symétrique et définie positive pour tout $u^k \neq 0$.

On peut déterminer λ^k et u^k, sachant que la relation (11) doit être satisfaite, on a:

$$\left[H^k + \lambda^k u^k . u^{k^T}\right] . y^k = s^k$$

Tous calculs faits (voir annexe 3) on obtient la formule de correction suivante :

$$H^{k+1} = H^k + \frac{(s^k - H^k . y^k).(s^k - H^k . y^k)^T}{y^{k^T}(s^k - H^k . y^k)} \qquad (12)$$

Cette formule de mise à jour permet de construire des algorithmes, qui appliqués sur une fonction quadratique, convergent en au plus *n*+1 itérations.

Il est intéressant de noter que cette démonstration ne nécessite pas : $s^{k^T} . y^k > 0$ c'est à dire que x^{k+1} soit le minimum de $f(x)$ dans la direction d^k. On se contentera alors de choisir α^k de façon à ce que : $f(x^k + \alpha^k d^k) < f(x^k)$. Cette particularité permet de concevoir des algorithmes particulièrement simples ne nécessitant pas de recherche unidimensionnelle. Par contre, quelques précautions sont à prendre lors de l'utilisation de la formule de correction (12). En effet, même appliquée sur des fonctions quadratiques avec une matrice de départ H^0, définie positive, les matrices générées par cette formule peuvent ne plus être définies positives. *Cullum* et *Brayton* [14] expliquent que ce phénomène est lié a une perte d'indépendance des directions d^k, engendrée par les erreurs d'arrondi dans les calculs, et propose un moyen pour palier ces inconvénients. Cependant même corrigée, cette formule de mise à jour offre des performances inférieures aux méthodes s'appuyant sur des matrices de correction de rang 2.

2.5.4.2 Correction de rang 2.

On choisit pour D^k la forme suivante :

$$D^k = \lambda^k u^k u^k + \beta^k v^k v^k \quad u^k, v^k \in R^n \quad \lambda^k, \beta^k \in R$$

La relation (11) donne alors :

$$\left[H^k + \lambda^k u^k . u^{k^T} + \beta^k v^k v^k\right] . y^k = s^k$$
$$\Rightarrow \lambda^k (u^{k^T} . y^k) u^k + \beta^k (v^{k^T} . y^k) v^k = (s^k - H^k y^k)$$

On dispose d'une seule équation et de 4 inconnues $\lambda^k, \beta^k, u^k, v^k$, donc a priori d'une infinité de choix, autorisant autant de formules de correction que de choix. *Fletcher* [20] propose le choix trivial suivant :

$$\lambda^k (u^{k^T}.y^k) = 1 \; ; \; u^k = s^k$$
$$\beta^k (v^{k^T}.y^k) = 1 \; ; \; v^k = -H^k y^k$$

On obtient alors la formule de correction suivante : appelée formule de correction DFP, des initiales de leurs auteurs : *Davidon* (1959), *Fletcher* (1963), *Powel* (1963).

$$H^{k+1} = H^k + \frac{s^k.s^{k^T}}{s^{k^T}.y^k} - \frac{H^k.y^k.y^{k^T}.H^k}{y^{k^T}.H^k.y^k}$$

Cette formule de correction engendre des matrices H^k définies positives si, pour chaque itération $s^{k^T}.y^k > 0$ [20]. Cette condition est normalement remplie dans le cadre de l'algorithme du paragraphe 2.5.3. En effet :

$$s^{k^T}.y^k = \alpha^k d^{k^T}.\left[\nabla f(x^{k+1}) - \nabla f(x^k)\right]$$
$$s^{k^T}.y^k > 0 \Rightarrow d^{k^T}.\nabla f(x^{k+1}) > d^{k^T}.\nabla f(x^k)$$

Avec :

$d^{k^T}.\nabla f(x^k) \le 0 : d^k$ direction de descente pour $f(x)$.

$d^{k^T}.\nabla f(x^{k+1}) = 0 : x^{k+1}$ est le minimum de $f(x)$ dans la direction d^k.

Cette condition est également remplie si on utilise le critère d'arrêt pour la recherche undimensionnelle présenté dans le paragraphe 2.4.4:
En supposant que :

$$g'(\alpha^k) < -\eta g'(0), \eta \in [0,1[$$

soit vérifié, on a:

$$d^{k^T}.\nabla f(x^{k+1}) < -\eta d^{k^T}.\nabla f(x^k)$$

Comme $\eta d^{k^T}.\nabla f(x^k) > d^{k^T}.\nabla f(x^k)$, puisque $\eta < 1$ et $d^{k^T}.\nabla f(x^k) \le 0$. La condition $d^{k^T}.\nabla f(x^{k+1}) > d^{k^T}.\nabla f(x^k)$ est bien remplie donc $s^{k^T}.y^k > 0$. **On peut alors se contenter d'une recherche unidimensionnelle approchée** pour le calcul de α^k.

Cette particularité théorique , a priori bien intéressante, n'est en pratique absolument pas vérifiée. *Fletcher* [20] rapporte des résultats de calcul qui montrent clairement la nette dégradation de performance de la formule de correction DFP (le nombre d'itérations et le nombre d'évaluations de fonction augmentent) lorsque l'imprécision sur la recherche unidimensionnelle augmente.

La formule de correction suivante **présente l'avantage d'être bien moins sensible aux imprécisions dans la procédure de recherche unidimensionnelle**. Développée indépendamment par *Broyden* [7], *Fletcher* [19], *Goldfarb* [26] et *Shanno* [66], la formule de correction BFGS utilise également une formule de correction de rang 2. Pour introduire cette nouvelle formule de correction nous adopterons la démarche de *Fletcher* dans [19].

On sait que : $H^{k+1}.y^k = s^k$. Donc $G^{k+1} = \left[H^{k+1}\right]$ vérifie la relation : $G^{k+1}.s^k = y^k$. *Fletcher* observe alors qu'en reprenant la formule DFP et en remplaçant H^{k+1} par G^{k+1}, H^k par G^k et en inversant les rôles de y^k et s^k, on obtient une formule de correction, non plus de l'inverse du hessien mais du hessien lui-même, d'où :

$$G^{k+1} = G^k + \frac{y^k.y^{k^T}}{y^{k^T}.s^k} - \frac{G^k.s^k.s^{k^T}.G^k}{s^{k^T}.G^k.s^k} \tag{13}$$

En calculant l'inverse des deux membres de la relation (13) on obtient une formule de correction pour l'inverse du hessien :(calcul détaillé dans l'annexe 3). D'où la formule de correction BFGS :

$$H^{k+1} = H^k + \left(1 + \frac{y^{k^T}.H^k.y^k}{s^{k^T}.y^k}\right)\frac{s^k.s^{k^T}}{s^{k^T}.y^k} - \frac{H^k.y^k.s^{k^T} + s^k.y^{k^T}.H^k}{y^{k^T}s^k}$$

Les algorithmes utilisant les formules de correction DFP et BFGS possèdent les propriétés suivantes :

Avec une recherche unidimensionnelle exacte et appliquée sur une fonction quadratique :

- Convergence finie en au plus *n* itérations
- Si $H^0 = I$, et si le point de départ est identique, équivalence avec la méthode du gradient conjugué de *Fletcher Reeves* et donc génération de directions et de gradients conjugués.
- Lorsque ces méthodes terminent en *n* itérations, H^{k+1} est égal à l'inverse du hessien de la fonction quadratique.

Avec une recherche undimensionnelle exacte et appliquée sur une fonction quelconque :

- Convergence globale vers un point stationnaire si le hessien en ce point est défini positif, si les matrices H^k sont toutes définies positives avec un conditionnement borné.
- Vitesse de convergence superlinéaire.

La formule de correction BFGS, appliquée sur une fonction quelconque, présente en outre l'avantage de posséder ces propriétés même avec une recherche unidimensionnelle approchée.

D'autres formules de correction sont possibles. En incluant les formules de correction DFP et BFGS dans une classe plus vaste, dite famille de *Broyden* [7], grâce à une combinaison linéaire des formules DFP et BFGS on obtient :

$$H_{\theta}^{k+1} = (1-\theta) H_{DFP}^{k+1} + \theta H_{BFGS}^{k+1} \quad ; \quad \theta \in R \tag{14}$$

On retrouve évidemment pour $\theta = 0$ la formule DFP et pour $\theta = 1$ la formule BFGS, mais également la formule (12) de correction de rang 1 pour :

$$\theta = \frac{{s^k}^T . y^k}{{y^k}^T s^k - {y^k}^T . H^k . y^k}$$

On trouve dans [20] une étude des propriétés des formules de la famille *Broyden*, en fait elles possèdent les mêmes propriétés que les formules DFP et BFGS. L'intérêt de la relation (14) est de pouvoir expérimenter numériquement d'autres formules de correction en faisant varier la valeur de θ, de manière à déterminer la meilleure formule de correction pour un ensemble de fonctions tests. Ces études montrent que globalement la formule BFGS reste la plus performante [24], [66]. Cependant l'argument le plus efficace en faveur de la formule BFGS reste que **c'est la seule formule de correction de la famille *Broyden*, pour laquelle on puisse établir une propriété de convergence globale avec une recherche unidimensionnelle approchée [24].**

L'implémentation numérique des méthodes quasi newtoniennes est délicate car les formules de correction nécessitent un nombre conséquent de calculs matriciels. On retrouve ici les problèmes liés au conditionnement de l'inverse du hessien à l'optimum, rencontré dans les méthodes de directions conjuguées.

Au cours des itérations successives, à cause des erreurs d'arrondi, les matrices H^k peuvent ne plus être définies positives, mettant en péril la convergence de l'algorithme.

On trouve dans [24] un exemple frappant qui montre bien ce phénomène. Considérons la relation (13) à partir de la laquelle on établit la formule BFGS.

Avec $G^k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $s^k = \begin{bmatrix} 1 \\ \varepsilon \end{bmatrix}$, $y^k = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ et ε petit et positif

on a: ${y^k}^T . s^k = 1 + \varepsilon^2$ donc théoriquement G^{k+1} est définie positive.
Le calcul de G^{k+1} donne :

$$\begin{bmatrix} 1-\dfrac{1}{1+\varepsilon^2} & -\dfrac{\varepsilon^2}{1+\varepsilon^2} \\ -\dfrac{\varepsilon^2}{1+\varepsilon^2} & 1-\dfrac{1}{\varepsilon(1+\varepsilon^2)} \end{bmatrix}$$

On constate alors que pour $\varepsilon = 10^{-4}$ l'élément (1,1) de G^{k+1} vaut : $\approx 9.9999\,10^{-9}$, si l'arithmétique du calculateur travaille avec 7 chiffres significatifs après la virgule (cas de la majorité des FORTRAN en simple précision) cet élément sera arrondi à 0. La matrice G^{k+1} n'est alors plus définie positive et par conséquent son inverse H^{k+1} ne l'est plus également, d'où un risque de non convergence de l'algorithme.

Pour éviter ce phénomène, un processus de réinitialisation peut être mis en place en remplaçant toutes les *n* itérations la matrice H^k par la matrice identité. L'inconvénient de ce procédé est que les informations accumulées dans les termes non diagonaux de H^k sont perdues entraînant une perte d'efficacité. On peut adopter une autre technique, en utilisant la relation (13) définissant une mise à jour du hessien lui-même, et en intégrant dans cette relation une factorisation de Cholesky [23]. On obtient alors une matrice G^{k+1} factorisée sous la forme :

$$G^{k+1} = L^{k+1}.D^{k+1}.L^{k+1}$$

Avec :

D^{k+1} : matrice diagonale ($D_{ij}^{k+1} = 0$ pour $i \neq j$ et $D_{ij}^{k+1} > 0$ pour $i=j$)

L^{k+1} : matrice triangulaire inférieure ($L_{ij}^{k+1} \neq 0$ pour $i \neq j$ et $L_{ij}^{k+1} = 1$ pour $i=j$).

L'inversion de G^{k+1}, donc la résolution du système linéaire :

$$G^{k+1}.d^k = -\nabla f(x^k)$$

est alors facilité de part la forme de G^{k+1}. L'avantage de cette méthode est de disposer indépendamment des termes diagonaux de G^{k+1} et donc de mieux contrôler les erreurs d'arrondi de G^{k+1} lors du calcul de d^k. On peut ainsi mettre en place une procédure de réinitialisation évitant la perte des informations accumulées dans les termes non diagonaux de G^{k+1}.

2.5.5 Critères d'arrêt pour fonctions différentiables

Un critère d'arrêt est un ensemble de conditions qui permettent de décider de l'arrêt ou de la poursuite des calculs au point x^k couramment généré par un algorithme. Trois types de tests sont généralement effectués :

- Un test sur l'épuisement d'une certaine ressource allouée pour la recherche de la solution optimale du problème. Cette ressource est évaluée en nombre d'itérations disponibles et ,ou en nombre d'évaluations de la fonction objectif.
- Un test sur la convergence de la suite $\{x^k\}$ générée par l'algorithme.
- Un test sur la satisfaction d'une condition d'optimalité : ici il s'agit de la stationnarité du point x^k.

Le premier test est facile à mettre en place. Il s'agit de compter les itérations et le nombre d'évaluations de la fonction objectif, puis quand ces valeurs sont supérieures à une limite fixée d'interrompre les calculs. Cette limite peut être déterminée par l'utilisateur, ou automatiquement en fonction du nombre de variables. La valeur Max[10*n*,100] est généralement admise. La limite sur le nombre d'évaluations de fonction est plus difficile à fixer automatiquement car elle dépend du conditionnement de la fonction à minimiser, une grande partie des évaluations de fonction effectuée par un algorithme de minimisation est "consommée" par la recherche unidimensionnelle. La présence de ce test dans tout algorithme d'optimisation est indispensable, elle évite le bouclage des calculs.

Le test de convergence de la suite $\{x^k\}$ est effectué de la manière suivante :

$$\left| f(x^{k+1}) - f(x^k) \right| < \varepsilon_1 \tag{15}$$

$$\left\| x^{k+1} - x^k \right\| < \varepsilon_2 \tag{16}$$

Normalement pour les fonctions objectifs bien conditionnées la satisfaction de la condition (15) entraîne celle de la condition (16), cependant dans le cas d'un mauvais conditionnement la condition (16) "force" l'algorithme à continuer vers une meilleure approximation de la solution.

Enfin le test sur la stationnarité de x^k est de la forme :

$$\left\| \nabla f(x^k) \right\| < \varepsilon_3$$

Les paramètres $\varepsilon_1, \varepsilon_2, \varepsilon_3$ peuvent être fixés indépendamment par l'utilisateur ou liés entre eux par un seul paramètre, [24] propose :

$$\varepsilon_1 = \tau_F \left(1 + \left| f(x^k) \right|\right)$$

$$\varepsilon_2 = \sqrt{\tau_F} \left(1 + \left\| x^k \right\|\right)$$

$$\varepsilon_2 = \sqrt[3]{\tau_F} \left(1 + \left| f(x^k) \right|\right)$$

Dans ce cas l'utilisateur ne précise que la valeur de τ_F, déterminant ainsi la précision obtenue. Toutefois il faudra vérifier que la précision requise ne soit pas inférieure à celle de l'arithmétique du calculateur.

2.5.6 Choix d'une méthode

Le choix d'une méthode pour minimiser une fonction de plusieurs variables, continue et différentiable est assez simple, il peut se résumer à ces trois alternatives :

- Si la fonction à minimiser comporte beaucoup de variables (plus d'une centaine) l'utilisation d'une méthode de gradient conjugué est préférable. En effet ces méthodes nécessitent le stockage de très peu d'information, essentiellement 3 vecteurs de dimensions *n*, et un nombre d'opérations de calcul (additions, multiplications, divisions) proportionnel à *n*. Il s'agit là du principal atout de ces méthodes par rapport aux méthodes quasi newtoniennes. En effet ces dernières nécessitent le stockage d'une matrice de dimension *n* et utilisent un nombre de calculs proportionnel à n^2 Les performances inférieures en terme de vitesse de convergence des méthodes de gradient conjugué sont alors compensée par le plus petit nombre de calculs qu'elles nécessitent. La précision de la recherche unidimensionnelle est importante et conditionne les performances des méthodes de gradient conjugué, de sorte qu'une interpolation polynomiale sera souvent effectuée dans la recherche undimensionnelle pour améliorer la précision des résultats.

- Par contre lorsqu'on désire minimiser une fonction avec peu de variables, les méthodes quasi newtoniennes sont les plus indiquées et plus particulièrement la formule de correction BFGS. Son insensibilité aux imprécisions de la recherche unidimensionnelle permet l'utilisation d'une technique de minimisation unidimensionnelle approchée, économique en terme d'évaluation de fonction. Sa vitesse de convergence superlinéaire et la robustesse de sa propriété de convergence globale autorisent son utilisation sur un vaste champ de fonctions.

- Enfin, pour quelques problèmes particuliers, lorsque le hessien de la fonction est disponible explicitement, une méthode de Newton appliquée dans le voisinage de la solution (on peut alors supposer une convexité locale), ou sur des fonctions strictement convexes peut se révéler intéressante grâce à sa très bonne vitesse de convergence (quadratique).

3 Optimisation avec fonctions contraintes

On cherche à résoudre le problème :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \end{cases}$$

problème que l'on peut écrire de manière équivalente :

$$(P_c)\begin{cases} \text{Trouver x}^* \text{ tel que : } f(x^*) = \underset{x \in D}{Min}\, f(x) \\ \text{où} \\ D = \left\{ x \in R^n / c_j(x) \le 0 \; j = 1...m \right\} \end{cases}$$

$$\text{Avec } f, c_j \; : \; x \in R^n \longrightarrow f(x), c_j(x) \in R \quad j = 1..m$$

On suppose $f(x)$et $c_j(x)$ continues et différentiables et on rappelle que nous notons $J(x)$ l'ensemble des indices des contraintes actives au point *x* de *D*.

La condition d'existence d'une solution du problème (P_c) s'énonce de la façon suivante :

> Si l'ensemble *D* est compact (fermé et borné) et non vide, la fonction $f(x)$ étant continue le problème (P_c) admet nécessairement une solution optimale $x^* \in D$. Cette solution optimale est unique si l'ensemble *D* est convexe et si la fonction $f(x)$est strictement convexe. Dans ce dernier cas il s'agit évidemment d'une solution optimale globale.

3.1 Conditions nécessaires d'optimalité

On cherche à établir ici les conditions mathématiques qui vont caractériser un minimum local ou global du problème d'optimisation. Pour établir ces conditions d'optimalité, commençons par énoncer une condition nécessaire de minimum relatif sur un ensemble convexe, que nous généraliserons ensuite au cas d'un ensemble quelconque.

Soit $f : x \in R^n \longrightarrow f(x) \in R$ et U une partie convexe de R^n. Si la fonction $f(x)$ admet un minimum relatif en x_0 par rapport à U alors :[13]

$$\nabla^T f(x_0).(y - x_0) \geq 0 \quad \forall y \in U$$

En effet, soit $y = x_0 + w$ un point quelconque de U, l'ensemble U étant convexe les points de la forme $(x_0 + \theta w)$ avec $\theta \in [0,1]$, sont toujours dans U. Comme la fonction $f(x)$ est dérivable en x_0, on peut écrire :

$$f(x_0 + \theta w) = f(x_0) + \theta \nabla^T f(x_0).w + \varepsilon(\theta)$$

$$\text{Avec } \lim_{\theta \to 0} \varepsilon(\theta) = 0$$

Si $f(x_0)$ est un minimum relatif de f en x_0 on a : $f(x_0 + \theta w) - f(x_0) \geq 0 \quad \forall \theta \in [0,1]$, donc nécessairement il faut $\nabla^T f(x_0).w \geq 0$.

Dans la démarche ci-dessus, comme U est convexe on est certain que les points $(x_0 + \theta w)$ pour $\theta \in [0,1]$ appartiennent à U. Dans le cas d'un ensemble U quelconque, c'est à dire non nécessairement convexe, cela n'est plus vrai.

On introduit alors la notion de cône des directions admissibles en $x_0 \in U$ pour généraliser la notion la condition de minimum relatif aux ensembles non convexes.

Ce cône ,noté $C_{ad}(x_0)$, est formé du singleton $\{0\}$(direction "nulle") et de l'ensemble des directions issues de x_0 telles que tout déplacement dans l'une de ces directions ne fasse pas sortir de l'ensemble U. On peut également définir le cône des directions admissibles comme l'ensemble des tangentes en x_0 à tous les arcs de courbe issus de x_0 et entièrement contenus dans l'ensemble U.

La condition nécessaire de minimum relatif sur un ensemble non convexe s'exprime maintenant sous la forme : [13]

Soit U une partie quelconque non vide de R^n :

(1) En tout point de U le cône des directions admissibles C_{ad} est fermé.

(2) Soit $f : R^n \longrightarrow R$, définie sur R^n contenant U, si la fonction $f(x)$ admet en $x_0 \in U$ un minimum relatif par rapport à l'ensemble U et si elle est dérivable en x_0 alors :

$$\nabla^T f(x_0).(y - x_0) \geq 0 \forall y \in \{x_0 + C_{ad}(x_0)\}$$

Examinons, maintenant, le cas où l'ensemble U correspond au domaine des solutions possibles D et sous quelles hypothèses la définition précédente est applicable :
On a:

$$D = \{x \in R^n / c_j(x) \leq 0 \, j = 1...m\}$$

Soit $C^*(x)$ l'ensemble des vecteurs $y \in R^n$ tel que :

$$C^*(x) = \{y \in R^n / \nabla^T c_j(x).y \leq 0; j \in J(x)\}$$

Généralement, comme le montre la figure 3.15, $C^*(x)$ et le cône des directions admissibles en x ,$C_{ad}(x)$, sont égaux .

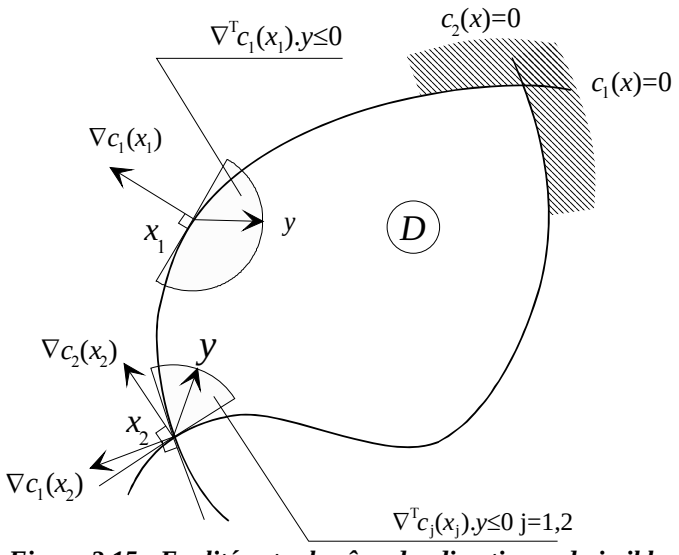


Figure 3.15 : Egalité entre le cône des directions admissibles et celui défini par les gradients des fonctions contraintes.

Cependant on ne peut pas espérer avoir l'égalité dans tous les cas, car le cône $C^*(x)$ est toujours un ensemble convexe alors que $C_{ad}(x)$ ne l'est pas nécessairement comme le montre l'exemple ci-dessous:

Soit l'ensemble D définit par :

$$c_1(x) = -x_1 - x_2 \le 0$$

$$c_2(x) = x_1(x_1^{\,2} + x_2^{\,2}) - 2(x_1^{\,2} - x_2^{\,2}) \le 0$$

$$\nabla c_1(x) = \begin{vmatrix} -1 \\ -1 \end{vmatrix} \qquad \nabla c_2(x) = \begin{vmatrix} 3x_1^{\,2} + x_2^{\,2} - 4x_1 \\ 2x_1x_2 + 4x_2 \end{vmatrix}$$

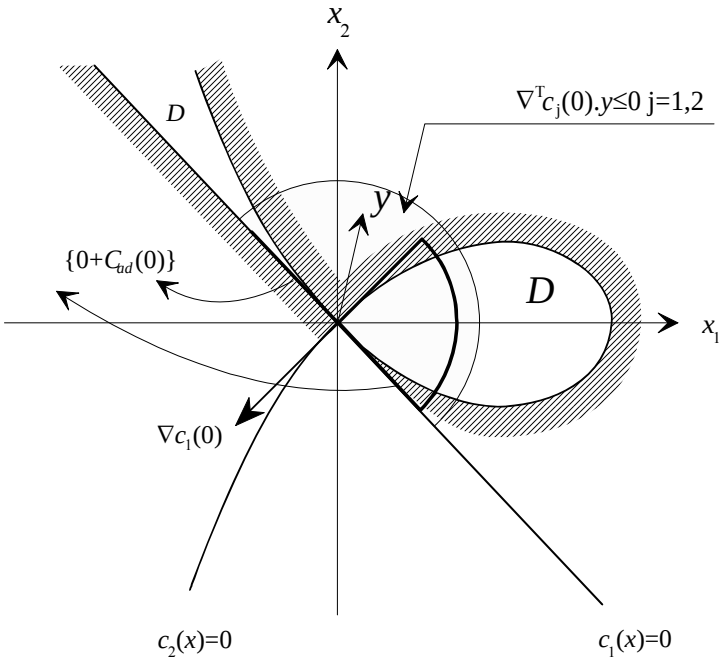


Figure 3.16 : Exemple de problème où l'égalité n'est pas vérifiée.

D'où la nécessité d'introduire une définition supplémentaire :

Les fonctions contraintes définissant l'ensemble des solutions seront dites "qualifiées" en $x \in D$ si on a l'égalité :[2]

$$C_{ad}(x) = C^*(x)$$

ou encore si il existe un vecteur y de R^n tel que, pour tout $j \in J(x)$:

(a) $\nabla^T c_j(x).y \le 0$

(b) $\nabla^T c_j(x).y < 0$ si $c_j(x)$ n'est pas linéaire

Dans cette définition de qualification des contraintes, il y a une distinction entre les fonctions contraintes non linéaires et linéaires à travers les inégalités (a) et (b). La figure ci-dessous explicite bien cette nécessité :

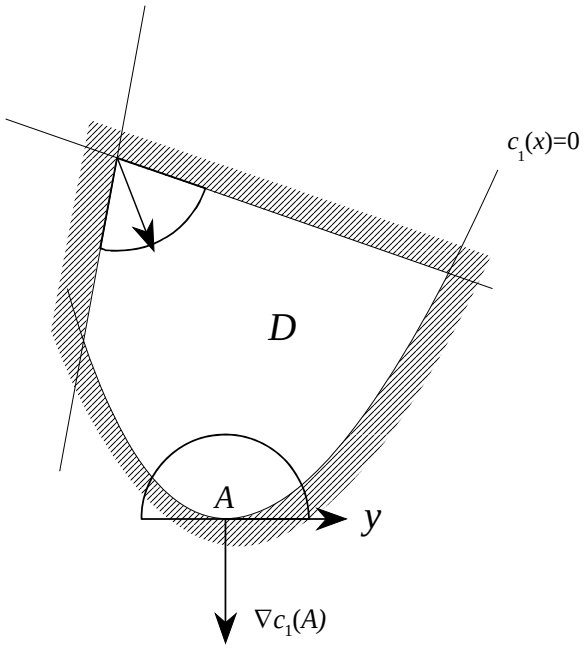


Figure 3.17 : Nécessite de la distinction entre les fonctions contraintes linéaires et non linéaires

Au point *A*, on voit clairement que le cas $\nabla^T c_1(A).y = 0$ est à exclure, car tous les déplacements dans la direction *y* provoqueront une "sortie" du domaine des solutions.

La condition nécessaire de minimum relatif sur l'ensemble *D* donne alors :

> Soit x_0 un point de *D*, les fonctions contraintes étant qualifiées en x_0.
> Si la fonction $f : R^n \longrightarrow R$, admet un minimum relatif en x_0 par rapport à *D* et si elle est dérivable en x_0 alors :
>
> $$\nabla^T f(x_0).y \geq 0 \forall y \in \{x_0 + C_{ad}(x_0)\}$$
>
> Les contraintes sont qualifiées en x_0 donc $C_{ad}(x) = C^*(x)$, pour $y \in C^*(x)$ on a alors :
>
> $$\nabla^T c_j(x_0).y \leq 0$$

Rappelons alors le théorème de Farkas-Minkowski qui précise l'existence d'une solution non négative pour un système linéaire :

> Soit *A* une matrice de dimensions $p \times q$, et *b* un vecteur de R^p.
> Pour qu'il existe $x \geq 0$, $x \in R^q$ tel $A.x = b$
> Il faut et il suffit que $\forall u \in R^p$ tel que $u^T.A \geq 0$ on ait : $u^T.b \geq 0$.

En identifiant la matrice A^T avec la matrice dont les colonnes sont les vecteurs de R^n, $-\nabla c_j(x_0) \quad j \in J(x_0)$, le vecteur *y* de R^n au vecteur *u* et enfin le vecteur *b* au vecteur de R^n, $\nabla f(x_0)$ on a:

$$\begin{cases} \nabla^T f(x_0).y \geq 0 \longrightarrow u^T.b \geq 0 \\ \nabla^T c_j(x_0).y \leq 0 \quad j \in J(x_0) \longrightarrow A^T.u \geq 0 \longrightarrow u^T.A \geq 0 \end{cases}$$

Les conditions d'application du théorème précédent étant remplies, il existe donc un vecteur *x* de R^k,($k = |J(x_0)|$: nombre de contraintes actives) **positif ou nul** tel que :

$$A.x = b \text{ soit } \sum_{j=1}^{k} \left(-\nabla c_j(x_0)\right) x_j = \nabla f(x_0)$$

En appelant les x_j , λ_j et en choisissant pour les fonctions contraintes inactives $j \in \{J - J(x_0)\}$ $\lambda_j = 0$ ce qui s'exprime par les relations $\lambda_j c_j(x_0) = 0$, on obtient un résultat fondamental d'optimisation non linéaire : Les conditions d'optimalités de *Kuhn et Tucker* qui s'énonce :

> x^* est un optimum local (minimum relatif) du problème (P_C), si l'hypothèse de qualification des contraintes est satisfaite en x^* et s'il existe des nombres réels $\lambda_j \quad j = 1..m$, ***positifs ou nuls*** appelés ***multiplicateurs de Kuhn et Tucker*** tels que :
>
> $$\begin{cases} \nabla f(x^*) + \sum_{j=1}^{m} \lambda_j \nabla c_j(x^*) = 0 \\ \lambda_j c_j(x^*) = 0 \; j = 1..m \end{cases}$$

La notion de qualification des contraintes exprimée jusqu'à présent est peu maniable et dépend du point d'application. Les conditions suffisantes pour qu'elles soient satisfaites sont regroupées ci-dessous :

1) Toutes les fonctions $c_j(x)$ sont linéaires et l'ensemble *D* est non vide, alors les contraintes sont qualifiées pour tout $x \in D$.
2) Toutes les fonctions $c_j(x)$ sont convexes et l'intérieur de *D* est non vide : Il existe un point $\overline{x} \in R^n$ tel que $c_j(x) < 0$ si $c_j(x)$ est non linéaire ou tel que $c_j(x) \leq 0$ si $c_j(x)$ est linéaire pour tout $j = 1..m$, alors les contraintes sont qualifiées pour tout $x \in D$.
3) Les gradients des contraintes actives en x^* sont linéairement indépendants, c'est à dire qu'il n'existe pas de β_j, $j \in J(x^*)$ non nuls tels que :

$$\sum_{j \in J(x^*)} \beta_j \nabla c_j(x) = 0$$

alors les contraintes sont qualifiées en x^*. Le point x^* est **un point régulier** de *D*.

3.1.1 Cas des fonctions contraintes égalités

Les conditions de *Kuhn et Tucker* s'étendent aisément à des problèmes comportant des fonctions contraintes égalités. Dans ce cas les multiplicateurs associés aux contraintes égalités sont non restreints en signe, on retrouve là les multiplicateurs de Lagrange.

La notion de qualification des contraintes s'exprime de la même manière, mis à part la définition de $C^*(x)$ qui est dans ce cas :

$$C^*(x) = \left\{ y \in R^n / \nabla^T c_j(x).y = 0; j \in J_e \right\}$$

où J_e est l'ensemble des indices des contraintes égalités.

Les conditions suffisantes pour que la qualification des contraintes soit réalisée sont :

1) Les fonctions contraintes $c_j(x) = 0$, $j \in J_e$ sont linéaires et il existe $\overline{x}$ tel que $c_j(\overline{x}) = 0$, $j \in J_e$.
2) En x^* les gradients des contraintes égalités sont linéairement indépendants.

Dans ce cas les conditions de *Kuhn et Tucker* s'expriment sous la forme :

x^* est un optimum local (minimum relatif) du problème (P_C), si l'hypothèse de qualification des contraintes est satisfaite en x^* et s'il existe des nombres réels $\lambda_j \quad j = 1..m$, ***non restreints en signe*** appelés ***multiplicateurs de Kuhn et Tucker*** tels que :

$$\begin{cases} \nabla f(x^*) + \sum_{j=1}^{m} \lambda_j \nabla c_j(x^*) = 0 \\ c_j(x^*) = 0 \, j = 1..m \end{cases}$$

3.1.2 Interprétation géométrique des conditions de *Kuhn et Tucker*

Considérons, par exemple, le problème de deux variables et trois fonctions contraintes suivant :

$$\begin{cases} \text{Minimiser} \quad f(x) = 10(x_1{}^2 - x_2)^2 + (x_1 - 1)^2 + 4 \\ \text{Sous les fonctions contraintes :} \\ c_1(x) = 2.5\mathrm{x}_1{}^2 + 3\mathrm{x}_1 - 8x_2 + 8 \le 0 \\ c_2(x) = 1.5\mathrm{x}_1{}^2 - 5\mathrm{x}_1 - 8x_2 - 12 \le 0 \\ c_3(x) = \mathrm{x}_1 - 1.25 \le 0 \end{cases}$$

$$\text{Avec } x = \{x_1, x_2\}^T$$

Au point A de coordonnées (1.25,1.957), solution optimale globale du problème, il y a deux contraintes actives et $J(A) = \{1,3\}$ (figure 3.18).

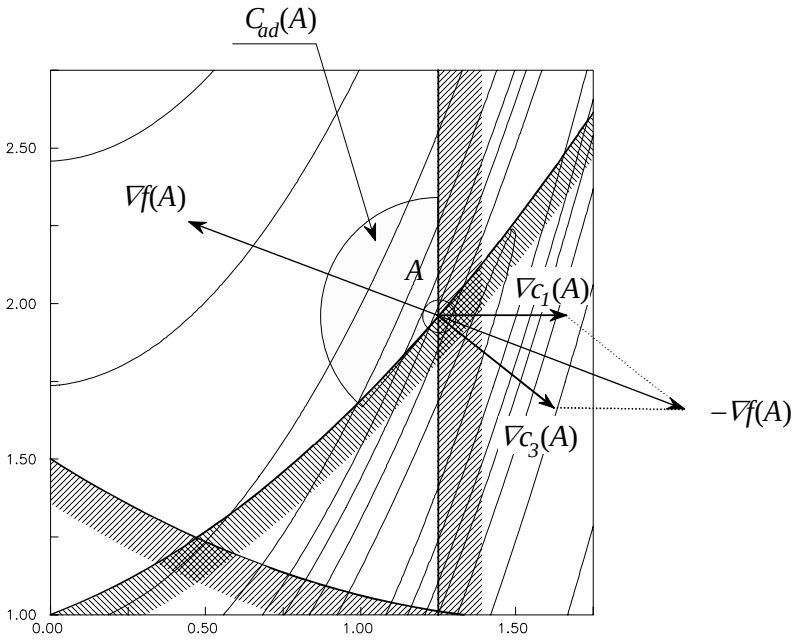


Figure 3.18 : Interprétation graphique des multiplicateurs de Kuhn et Tucker

Au point *A* l'ensemble des directions admissibles *y* forme un cône intersection des 2 demi-espaces d'équation :

$$\nabla^T c_1(x).y < 0$$
$$\nabla^T c_3(x).y \le 0$$

En *A* le vecteur $-\nabla f(A)$ fait un angle obtus ($> \pi/2$) avec chaque direction admissible. On vérifie également que $-\nabla f(A)$ s'exprime comme une combinaison linéaire à coefficients positifs (les nombres λ_j) des gradients des contraintes actives.

3.1.3 Unicité des multiplicateurs de *Kuhn et Tucker* à l'optimum

Les conditions de *Kuhn et Tucker* s'écrivent comme un système de (*n*+*m*) équations non linéaires à (*n*+*m*) inconnues, $x_1 \ldots x_n, \lambda_1 \ldots \lambda_m$, où *n* et *m* sont respectivement le nombre de variables et de contraintes inégalités du problème.

$$\begin{cases} \dfrac{\partial f}{\partial x_1}(x^*) + \lambda_1 \dfrac{\partial c_1}{\partial x_1}(x^*) + \cdots + \lambda_m \dfrac{\partial c_m}{\partial x_1}(x^*) = 0 \\ \vdots \\ \dfrac{\partial f}{\partial x_n}(x^*) + \lambda_1 \dfrac{\partial c_1}{\partial x_n}(x^*) + \cdots + \lambda_m \dfrac{\partial c_m}{\partial x_n}(x^*) = 0 \\ \lambda_1 c_1(x^*) = 0 \\ \vdots \\ \lambda_m c_m(x^*) = 0 \end{cases}$$

En supposant x^* connu, on constate que les multiplicateurs de *Kuhn et Tucker* sont solutions du système linéaire :

$$\sum_{j \in J(x^*)} \nabla c_j(x^*) \lambda_j = 0 \qquad (17)$$

Il seront donc déterminés de manière unique si les gradients des contraintes actives, $\nabla c_j(x^*) \;\; j \in J(x^*)$ sont linéairement indépendants.

Bien que cette propriété puisse s'établir indépendamment de l'hypothèse de qualification des contraintes, il faut néanmoins que celle-ci soit satisfaite pour pouvoir appliquer les conditions de *Kuhn et Tucker*. On remarquera également que, lorsque les contraintes sont qualifiées parce que les gradients sont linéairement indépendants cela conduit à l'unicité des multiplicateurs à l'optimum.

On peut écrire la relation (17) de la manière suivante :

$$A.x = b$$

En identifiant les colonnes de la matrice A^T avec les gradients des contraintes actives en x^*, $\nabla c_j(x^*) \;\; j \in J(x^*)$, le vecteur b avec $-\nabla f(x^*)$ et enfin le vecteur x avec les multiplicateurs de *Kuhn et Tucker* associés aux contraintes actives.

$$\underbrace{\begin{bmatrix} \nabla c_1(x^*) \\ \vdots \\ \nabla c_m(x^*) \end{bmatrix}}_{A} \otimes \underbrace{\begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} \vdots \\ -\nabla f(x^*) \\ \vdots \end{bmatrix}}_{b}$$

La matrice *A* comporte donc *n* colonnes, le vecteur *x* possède alors au plus *n* composantes non nulles. **On en déduit qu'il y a au plus *n* contraintes actives non redondantes dans un problème d'optimisation à *n* variables.**

On peut alors remarquer que, dans le cas d'un problème convexe, dans lequel les contraintes sont qualifiées en tout point du domaine des solutions , si il existe un point x^* pour lequel il y a *n* contraintes actives, et si les gradients ne sont pas linéairement indépendants,

nécessairement il y a au moins une contrainte active redondante, dont la suppression n'entraînera aucun changement dans la solution du problème. Dans le cas général, il est impossible de prédire le nombre de contraintes actives à l'optimum, on peut tout au plus affirmer qu'il y a au maximum *n* contraintes actives non redondantes.

3.2 Méthodes primales

Ces méthodes sont dénommées ainsi, parce qu'elles traitent directement le problème d'optimisation sous sa forme initiale, elles opèrent dans l'espace des variables primales $x \in R^n$. Ces méthodes sont, comme leurs homologues pour les problèmes sans fonction contrainte, itératives et nécessitent **un point de départ appartenant au domaine des solutions**, donc délicat à déterminer. Le principe de base est similaire, c'est à dire un déplacement dans l'espace des variables à chaque itérations, en tenant compte des limitations imposées par les fonctions contraintes.

3.2.1 Méthode de directions réalisables

Directement inspirée des méthodes pour les problèmes sans fonction contrainte, elle utilise le principe d'une direction de déplacement, qui doit être une direction de descente pour la fonction objectif. A partir d'un point de départ à l'intérieur du domaine des solutions, on calcule la direction de déplacement d^k telle que :

1) Tout déplacement positif, $\alpha \geq 0$, dans cette direction ne fasse pas sortir de l'ensemble des solutions *D*.
2) La fonction objectif diminue strictement.

On montre que la direction d^k est solution du problème d'optimisation linéaire suivant :

$$(P_L) \begin{cases} \textit{Minimiser } \nabla^T f(x^k).d^k \\ \textit{Sous les contraintes :} \\ c_j(x^k) + \nabla^T c_j(x^k).d^k \leq 0 \\ \sum_{i=1}^{n} \left| d_i^k \right| = 1 \end{cases}$$

On calcule ensuite la valeur maximale de α, $\overline{\alpha}^k$ telle que le point x^{k+1} défini par :

$$x^{k+1} = x^k + \overline{\alpha}^k d^k$$

appartienne au domaine des solutions, puis on détermine $\alpha^k \in \left[0, \overline{\alpha}^k\right]$ minimisant la fonction objectif dans la direction d^k.

On peut imager le fonctionnement de la méthode en précisant que l'on obtient en quelque sorte un cheminement le long de la frontière du domaine des solutions dans la direction de plus forte pente pour la fonction objectif.

Le calcul de la direction de déplacement devra prendre en compte toutes les fonctions contraintes du problème et pas seulement les fonctions contraintes actives, c'est à dire toutes celles qui définissent la frontière du domaine des solutions au point courant. Cela permet d'éviter les discontinuités dues au changement de l'ensemble des fonctions contraintes actives. Ce calcul devra également prendre en compte la non linéarité des fonctions contraintes en évitant les directions de déplacement tangentes aux fonctions contraintes non linéaires. Pour cela on imposera à d^k de "s'écarter" localement de la frontière du domaine en introduisant des coefficients d'écart pour chaque fonction contrainte.

La propriété de convergence globale pour cette méthode ne peut être établie que sous des hypothèses assez restrictives :

Il faudra supposer que le domaine des solutions est convexe et que les fonctions $f(x)$ et $c_j(x) \quad j = 1..m$ soient continûment différentiables. De plus tous les points de l'ensemble D devront être des points réguliers[4]. On peut alors montrer que la méthode converge vers un point satisfaisant les conditions de *Kuhn et Tucker*, optimum local du problème si $f(x)$ n'est pas convexe et optimum global lorsque $f(x)$ est convexe.

On établit également que la vitesse de convergence est linéaire avec un taux de convergence asymptotique γ égal à :

$$\gamma = \left[\frac{(A-a)}{(A+a)} \right]^2$$

où A et a sont respectivement la plus grande et la plus petite des valeurs propres du Hessien de la fonction objectif à l'optimum du problème. On retrouve ici les problèmes liés au conditionnement de la fonction objectif rencontrés avec la méthode de la plus forte pente. Pour une description plus détaillée de cette méthode on pourra consulter les références [21], [45] et [67].

3.2.2 Méthode de linéarisation

Les performances de la méthode du simplexe, utilisée pour la résolution des problèmes linéaires, ont incité de nombreux auteurs à développer des méthodes de linéarisation.

Le principe de base est le suivant : on remplace la résolution d'un problème non linéaire par la résolution d'une suite de problèmes linéaires approximant le problème donné.

Le problème initial :

[4]Les gradients des contraintes actives pour tous les points de la frontière de D devront être linéairement indépendants.

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \end{cases}$$

est remplacé par le problème "linéarisé" en x^k suivant :

$$(P_L{}^k)\begin{cases} \textit{Minimiser } f(x^k) + \nabla^T f(x^k).(x - x^k) \\ \textit{Sous les fonctions contraintes} \\ c_j(x^k) + \nabla^T c_j(x^k).(x - x^k) \le 0 \quad j = 1...m \end{cases}$$

Il s'agit bien d'un problème linéaire en "x".

Cette méthode ne converge que pour des problèmes **convexes**. En effet le domaine des solutions de (P_c) doit être inclus dans le domaine linéarisé, ou polytope, défini par les contraintes linéarisées. On doit donc avoir les relations suivantes :

$$c_j(x^k) \ge c_j(x^k) + \nabla^T c_j(x^k).(x - x^k) \quad j = 1..m$$

Pour que les points appartenant au domaine des solutions initial défini par :

$$c_j(x) \le 0 \quad j = 1..m$$

appartiennent également au domaine des solutions linéarisé donné par :

$$c_j(x^k) + \nabla^T c_j(x^k).(x - x^k) \le 0 \quad j = 1..m$$

D'où l'inévitable hypothèse de convexité sur les fonctions contraintes.

En ce qui concerne la vitesse de convergence, aucun résultat n'a pu être établi. On trouvera dans les références [2], [21] et [45] une description de cette méthode de linéarisation ainsi qu'une description de la méthode de linéarisation par "génération de colonne" qui utilise une approximation barycentrique des fonctions du problème plutôt qu'une linéarisation tangentielle comme celle que nous venons de décrire. On notera que toutes les méthodes de linéarisation nécessitent un domaine des solutions convexe.

3.2.3 Méthode du gradient réduit généralisé

La méthode du gradient réduit généralisé est plutôt destinée aux problèmes d'optimisation comportant des fonctions contraintes égalités. On peut cependant l'appliquer au cas des fonctions contraintes inégalités en introduisant des "variables d'écart" [67]. De sorte que le problème :

$$(P_c)\begin{cases} Minimiser\ \ f(x) \\ Sous\ les\ fonctions\ contraintes \\ c_j(x) \le 0 \quad j = 1...m \end{cases}$$

devient :

$$(P_c')\begin{cases} Minimiser\ \ f(x) \\ Sous\ les\ fonctions\ contraintes \\ c_j(x) + s_j = 0 \quad j = 1...m \\ s_j \ge 0 \quad ; s_j \in R \end{cases}$$

on passe alors d'un problème à *n* variables à un problème comportant *m*+*n* variables.

En considérant le vecteur "variable étendu" de R^{m+n} défini par :

$$\tilde{x} = \begin{bmatrix} \tilde{x}_1 \\ \vdots \\ \tilde{x}_{m+n} \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \\ s_1 \\ \vdots \\ s_m \end{bmatrix}$$

Le problème (P_c') peut également s'écrire :

$$(P_c')\begin{cases} Minimiser\ \ f(\tilde{x}) \\ Sous\ les\ fonctions\ contraintes \\ h(\tilde{x}) = 0 \quad j = 1...m \\ \tilde{x} \ge 0 \end{cases}$$

Avec $h_j(\tilde{x}) = c_j(\tilde{x}) + s_j \quad j = 1..m.$

Théoriquement, les *m* fonctions contraintes égalités, $h_j(\tilde{x}) = 0$, peuvent servir à exprimer *m* variables en fonction des *n* autres. On peut alors partitionner le vecteur variable $\tilde{x}$ en 2 parties :

$$\tilde{x} = \begin{bmatrix} \tilde{x}_I \\ \tilde{x}_D \end{bmatrix}$$

Avec :

$\tilde{x}_I$: vecteur de *n* variables "indépendantes"

$\tilde{x}_D$: vecteur des *m* variables "dépendantes"

On notera que le choix des variables indépendantes peut être à priori quelconque. Nous verrons un peu plus tard suivant quel critère ce choix peut être effectué.

Le principe de base de la méthode du gradient réduit généralisé est de résoudre le problème dans l'espace des variables indépendantes. Plaçons nous en $\tilde{x}^k$, un point courant du processus de résolution. La variation de f pour un déplacement infinitésimal $d\tilde{x} = \left[d\tilde{x}_I, d\tilde{x}_D\right]$ s'exprime :

$$df(\tilde{x}^k) = \nabla_I^T f(\tilde{x}^k).d\tilde{x}_I + \nabla_D^T f(\tilde{x}^k).d\tilde{x}_D$$

Où ∇_I, ∇_D sont les gradients par rapport aux variables indépendantes et dépendantes.

Pour un même déplacement, la variation des fonctions contraintes s'écrit :

$$dh_j(\tilde{x}^k) = \nabla_I^T h_j(\tilde{x}^k).d\tilde{x}_I + \nabla_D^T h_j(\tilde{x}^k).d\tilde{x}_D \quad j = 1..m$$

Relation que l'on peut formuler matriciellement :

$$\begin{bmatrix} dh_1(\tilde{x}^k) \\ \vdots \\ dh_m(\tilde{x}^k) \end{bmatrix} = \begin{bmatrix} \nabla_I^T h_1(\tilde{x}^k) \\ \vdots \\ \nabla_I^T h_m(\tilde{x}^k) \end{bmatrix}.d\tilde{x}_I + \begin{bmatrix} \nabla_D^T h_1(\tilde{x}^k) \\ \vdots \\ \nabla_D^T h_m(\tilde{x}^k) \end{bmatrix}.d\tilde{x}_D$$

$$dh(\tilde{x}^k) = A.d\tilde{x}_I + B.d\tilde{x}_D$$

Avec :

A : matrice à n colonnes et m lignes

B : matrice à m colonnes et m lignes.

$dh(\tilde{x}^k) = \left[dh_1(\tilde{x}^k) \cdots dh_m(\tilde{x}^k)\right]^T$

Ce déplacement, $d\tilde{x} = \left[d\tilde{x}_I, d\tilde{x}_D\right]$, est compatible avec les fonctions contraintes si :

$$dh(\tilde{x}^k) = 0$$

On en déduit, en supposant la matrice B régulière, donc inversible que (nous verrons plus loin que cela est un critère de choix pour les variables indépendantes et dépendantes) :

$$d\tilde{x}_D = -B^{-1}.A.d\tilde{x}_I$$

De sorte que :

$$df(\tilde{x}^k) = \left[\nabla_I^T f(\tilde{x}^k) - \nabla_D^T f(\tilde{x}^k).B^{-1}.A\right].d\tilde{x}_I$$

On appellera le vecteur $u \in R^n$ tel que :

$$u^T = \left[\nabla_I^T f(\tilde{x}^k) - \nabla_D^T f(\tilde{x}^k).B^{-1}.A\right]$$

Le gradient réduit généralisé.

A partir de ce gradient réduit généralisé, on peut alors définir une direction de descente $\tilde{d}_I^k$, pour la fonction objectif dans l'espace des variables indépendantes, dont chaque composante, $\tilde{d}_{I\,i}^k$, est définie par :

$$\tilde{d}_{I\,i}^k = \begin{cases} 0 \text{ si } u_i > 0 \text{ et } \tilde{x}_{Ii} = 0 \\ -u_i \text{ sinon} \end{cases} \text{ pour } i = 1..n$$

La distinction de ces deux cas permet de respecter les contraintes de positivité sur le vecteur variables $\tilde{x}$.

Les variables indépendantes sont mises à jour grâce à :

$$\tilde{x}_I^{k+1} = \tilde{x}_I^k + \alpha^k \tilde{d}_I^k$$
$$\text{où : } g(\alpha^k) = \underset{\alpha \geq 0}{Min}\ f(\tilde{x}_I^k + \alpha \tilde{d}_I^k)$$

Cette itération modifie évidemment la valeur des fonctions contraintes égalités. Il faut donc calculer la mise à jour correspondante des variables dépendantes qui vérifient les contraintes égalités. Cette opération conduit alors à la résolution du système non linéaire en $\tilde{x}_D^{k+1}$ suivant :

$$h_j(\tilde{x}_I^{k+1}, \tilde{x}_D^{k+1}) = 0 \text{ pour } j = 1..m$$

Si on applique une méthode de Newton, on obtient la relation itérative suivante pour la mise à jour des variables dépendantes :

$$\tilde{x}_D^{k+1} = \tilde{x}_D^k - \left[B^{-1}.A\right] h(\tilde{x}_I^{k+1}, \tilde{x}_D^{k+1})$$
$$\text{Avec } h(\tilde{x}_I^{k+1}, \tilde{x}_D^{k+1}) = \left[h_1(\tilde{x}_I^{k+1}, \tilde{x}_D^{k+1}) \cdots h_m(\tilde{x}_I^{k+1}, \tilde{x}_D^{k+1})\right]^T.$$

On montre que la méthode converge vers un point vérifiant les conditions de *Kuhn et Tucker* lorsque $\tilde{d}_I = 0$. [45]

On trouve dans [67] une stratégie de choix pour les variables indépendantes. Le principe est le suivant : la matrice carrée *B* de dimension *m* doit être inversée, il faut donc qu'elle soit

régulière. Les variables indépendantes sont choisies de sorte que, le fractionnement de la matrice contenant tous les gradients des fonctions contraintes égalités (*m* lignes, *n+m* colonnes) produise une matrice régulière, donc qu'ils soient linéairement indépendants.

Minoux [45] rapporte que la convergence globale de la méthode du gradient réduit généralisé ne peut être établie que sous des hypothèses généralement excessivement difficiles à vérifier en pratique et qu'il n'existe pas de résultats concernant la vitesse de convergence. Cependant de nombreux auteurs [67],[53],[63],[71], considèrent la méthode comme efficace sur de nombreux problèmes pratiques. *Vanderplaats* [67] indique toutefois que cette méthode rencontre quelques difficultés lorsqu'elle est appliquée sur des problèmes fortement non linéaires, ou sur des problèmes comportant beaucoup de fonctions contraintes inégalités.

3.3 Méthodes de pénalité

3.3.1 Principe général

Le principe de ces méthodes est le suivant : remplacer la résolution d'un problème d'optimisation avec fonctions contraintes par la résolution d'une suite de problèmes sans fonction contrainte en introduisant la notion de pénalisation. Nous verrons ultérieurement que l'on peut également obtenir cette transformation avec les méthodes dites "duales" et que les méthodes de pénalité peuvent également s'interpréter comme des méthodes duales.

Considérons le problème :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \end{cases}$$

Soit la fonction "pénalisée" :

$$P : x \in R^n \longrightarrow P(x) \in R$$
$$\text{telle que :}$$
$$P(x) = f(x) + H(x)$$

La fonction $H : x \in R^n \longrightarrow H(x) \in R$, dénommée fonction de pénalisation, sera définie par :

$$\forall x \in R^n : H(x) = \sum_{i=1}^{m} h\left(c_j(x)\right) \text{ avec } h(y) = \begin{cases} 0 \text{ si } y < 0 \\ +\infty \text{ si } y \ge 0 \end{cases}$$

Si le problème (P_C) admet une solution, il est clair que le minimum de $P(x)$ ne peut être atteint en un point n'appartenant pas à D car dans ce cas $P(x) = +\infty$. On constate que pour tout point $x \in D$ on a $H(x) = 0$, et par conséquent que $P(x) = f(x)$. Donc résoudre le problème (P_C) est équivalent à la recherche du minimum de $P(x)$.

Théoriquement séduisantes, ces méthodes ne sont pas ,en pratique, applicables sous cette forme. En effet les discontinuités de $H(x)$ rendent très difficiles la minimisation de $P(x)$.

3.3.2 Méthodes de pénalité extérieure : fonction de pénalisation quadratique

On peut éviter facilement la discontinuité de $H(x)$ en adoptant pour $h(x)$ l'expression suivante :

$$h(x) = \begin{cases} h(x) = 0 & \text{si } x < 0 \\ h(x) = x^2 & \text{si x} \geq 0 \end{cases}$$

On a donc :

$$H(x) = \sum_{j=1}^{m} h\left(c_j(x)\right) = \sum_{j=1}^{m} \left(Max\left\{0, c_j(x)\right\}\right)^2$$

Ce qui permet de définir la fonction $P : (x,r) \in R^n \times R \longrightarrow P(x) \in R$, comme :

$$P(x,r) = f(x) + r\,H(x) \text{ avec } r \text{ réel positif.}$$

La fonction $H(x)$ est appelée : fonction de pénalisation extérieure, et r le coefficient de pénalité. La fonction $P(x,r)$ ainsi obtenue est deux fois continûment différentiable mais possède une dérivée seconde discontinue aux points où les fonctions contraintes changent de signe.

Pour $r > 0$ notons $x^*(r)$ le minimum de $P(x,r)$. On remarque que la valeur de r doit être choisie grande pour que le point obtenu en minimisant $P(x,r)$ soit proche de la frontière du domaine des solutions, c'est à dire que $H\left(x^*(r)\right)$ soit suffisamment petit. Il faut cependant éviter de choisir r trop élevé sinon la fonction $P(x,r)$ devient mal conditionnée et des difficultés numériques apparaissent.

En effet le Hessien de $P(x,r)$ s'écrit :

$$\nabla_x^2 P(x,r) = \nabla^2 f(x) + 2r \sum_{j \in J_+} \left[c_j(x) \nabla^2 c_j(x) + \nabla c_j(x) . \nabla^T c_j(x) \right]$$

$$\text{où : } J_+ = \left\{ j \in J \,/\, c_j(x) > 0 \right\}$$

Si p est le nombre d'éléments de J_+, on démontre que lorsque $r \longrightarrow +\infty$, la matrice $\nabla_x^2 P(x,r)$ possède p valeurs propres qui tendent vers l'infini, les (n-p) autres valeurs restant bornées. Le conditionnement du hessien de $P(x,r)$ tend alors lui aussi vers l'infini [45].

Une méthode de pénalité sera mise en œuvre grâce à l'algorithme :

1) Choix d'un coefficient de pénalité de valeur modérée : r_k, $k \longleftarrow 0$.
   Calcul de :
   $$P\left(x^*(r_k), r_k\right) = \underset{x \in R^n}{Min}\ P(x, r_k)$$
2) Si la quantité $H\left(x^*(r)\right)$ est suffisamment faible alors $x^*(r)$ est une bonne approximation de l'optimum, sinon choisir $r_{k+1} > r_k$, faire $k \longleftarrow k+1$ et recommencer le processus à l'étape 1)

La méthode de pénalité extérieure converge vers une solution optimale de (P_C) sous des conditions très peu restrictives. En effet on peut établir que : [45]

Si $f(x)$ est continue, si l'ensemble *D* est fermé et si l'une des deux conditions suivantes est vérifiée :

a) $f(x) \longrightarrow +\infty$ quand $\|x\| \longrightarrow +\infty$

b) L'ensemble *D* est borné et $H(x) \longrightarrow +\infty$, quand $\|x\| \longrightarrow +\infty$

Alors, lorsque le coefficient de pénalité *r* tend vers $+\infty$, la suite $x^*(r)$ admet au moins un point d'accumulation solution optimale de (P_C).

La vitesse de convergence de la méthode dépend essentiellement de la vitesse de convergence de la méthode utilisée pour minimiser $P(x,r)$. On obtiendra une vitesse de convergence superlinéaire, par exemple, si on utilise une méthode quasi newtonienne. On notera que le **point de départ** pour la première minimisation sans contrainte peut être choisi **quelconque**, c'est là un très gros avantage par rapport aux méthodes primales. Lors des itérations suivantes, il est astucieux de choisir comme point de départ le point minimum obtenu à l'étape précédente: $x^*(r_{k-1})$.

Considérons les deux exemples suivants qui illustrent bien le principe d'une méthode de pénalité extérieure :

Exemple 1:

$$\begin{cases} \textit{Minimiser} \quad f(x) = (x-2)^2 \\ \textit{Sous la contrainte :} \\ c(x) = x - 1 \le 0 \end{cases}$$

La fonction de pénalisation a pour expression :

$$P(x,r) = (x-2)^2 + r.\left(Max\{0,(x-1)\}\right)^2$$

Soit $P(x,r) = (x-2)^2 + r.(x-1)^2$ pour $x \ge 1$, et $P(x,r) = (x-2)^2$ lorsque $x < 1$.

Le minimum de $P(x,r)$ est atteint pour $x \geq 1$ lorsque :

$$\frac{\partial P}{\partial x}(x,r) = 2(x-2) + 2r.(x-1) = 0 \Rightarrow x(r) = \frac{2+r}{1+r}$$

On voit dans le tableau de valeurs ci-dessous que : $\lim_{r \to +\infty} x(r) = 1 = x^*$ solution optimale du problème.

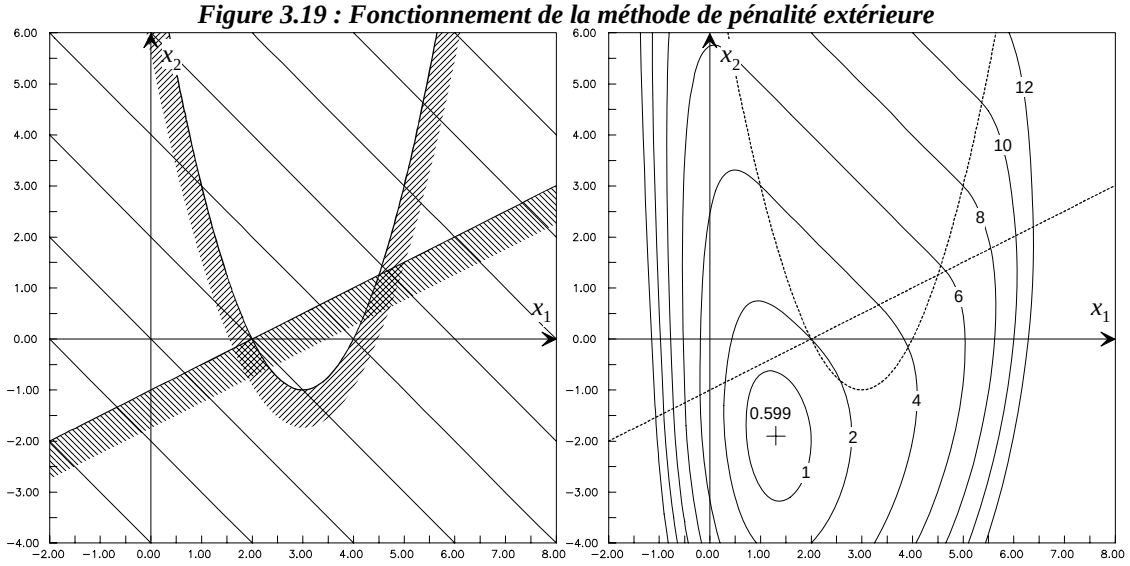
r	$x(r)$
1	1.5
2	1.33
5	1.166
10	1.0909
100	1.0099
1000	1.0009999

Exemple 2 :

$$\begin{cases} \textit{Minimiser } f(x_1, x_2) = x_1 + x_2 \\ \textit{Sous les contraintes} \\ c_1(x_1, x_2) = -2 + x_1 - 2x_2 \leq 0 \\ c_1(x_1, x_2) = 8 - 6x_1 + x_1^2 - x_2 \leq 0 \end{cases}$$

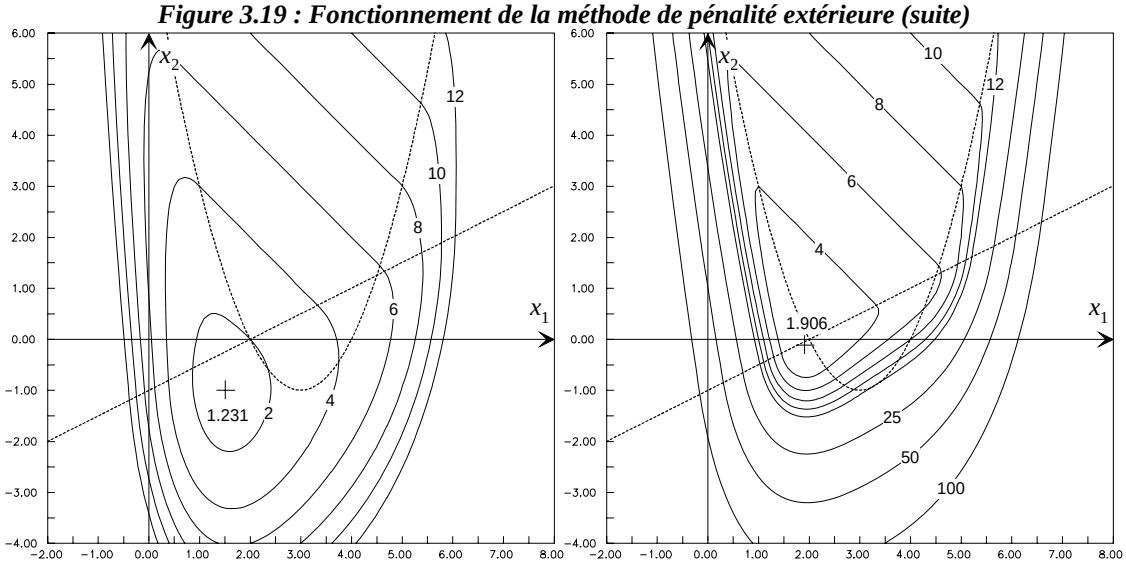
La figure 3.19 représente la fonction pénalisée $P(x_1, x_2, r)$ pour différentes valeurs du coefficient de pénalité *r*. Comme on peut le constater sur la figure, la méthode de pénalité extérieure approche la solution du problème par " l'extérieur" du domaine des solutions.

Figure 3.19 : Fonctionnement de la méthode de pénalité extérieure



(a) : Problème initial. ***(b) : Fonction pénalisée : r=0.05***

Figure 3.19 : Fonctionnement de la méthode de pénalité extérieure (suite)



(c) : Fonction pénalisée : r=0.1 ***(d) : Fonction pénalisée : r=1***

3.3.3 Méthodes de pénalité intérieure

Le fait que la solution soit approchée par l'extérieur du domaine des solutions peut parfois être gênant, car en fin de convergence la solution obtenue ne satisfait jamais vraiment toutes les fonctions contraintes du problème. Une modification de l'expression de la fonction de pénalisation permet de palier cet inconvénient.

Comme précédemment on aura :

$$P\ :\ (x,r) \in R^n \times R \longrightarrow P(x) \in R$$

avec :

$$P(x,r) = f(x) + r\,H(x)\ ,\ r \text{ réel positif.}$$

mais ici on utilisera :

$$H(x) = \sum_{j=1}^{m} -\frac{1}{c_j(x)}$$

On constate que pour *x* situé à l'intérieur de *D* on a:

$$c_j(x) < 0 \quad j = 1..m \quad \Rightarrow H(x) > 0$$

et que lorsque *x* tend vers la frontière de *D* :

$$H(x) \longrightarrow 0$$

Le principal inconvénient de la méthode est de nécessiter un point de départ, pour la première minimisation de $P(x,r)$, situé à l'intérieur du domaine des solutions. En ce point, la fonction de

pénalité n'ayant pas une valeur très élevée, il faudra choisir un coefficient de pénalité assez grand. Au cours des prochaines itérations ce coefficient de pénalité sera diminué, car $x^*(r)$ s'approchant de la frontière du domaine, la fonction $H(x)$ croît fortement.

L'algorithme de résolution est le suivant :

1) Choix d'un point de départ : $x_0 \in D, k \longleftarrow 0$, prendre r_k grand.
2) Calcul de :
$$P\left(x^*(r_k), r_k\right) = \underset{x \in R^n}{Min}\ P(x, r_k)$$
3) La quantité $H\left(x^*(r)\right)$ est suffisamment faible, alors $x^*(r)$ est une bonne approximation de l'optimum.
Sinon choisir $r_{k+1} < r_k$, faire $k \longleftarrow k+1$ et aller en 1)

Les conditions de convergence globale sont identiques à celles des méthodes de pénalité extérieure à ceci prés qu'on montre que la suite $x^*(r)$ admet un point d'accumulation solution optimale de (P_C) lorsque $r \longrightarrow 0$.

3.3.4 Approximation des multiplicateurs de *Kuhn et Tucker* à l'optimum

Les méthodes de pénalité permettent d'obtenir une approximation des multiplicateurs de *Kuhn et Tucker* à l'optimum. La précision de cette approximation, comme celle de la solution, dépend de *r* le coefficient de pénalité.

Si x^* est une solution optimale de (P_C) et si x^* est un point régulier de *D* alors les conditions de *Kuhn et Tucker* sont vérifiées et donnent :

$$\begin{cases} \nabla f(x^*) + \sum_{j=1}^{m} \lambda_j^* \nabla c_j(x^*) = 0 \\ \lambda_j^* c_j(x^*) = 0 \; j = 1..m \end{cases}$$

En reprenant le formulation de $P(x,r)$ pour une méthode de pénalité extérieure on a:

$$P(x,r) = f(x) + r\sum_{j=1}^{m} \left(Max\left\{0, c_j(x)\right\}\right)^2$$

Le gradient de $P(x,r)$ par rapport à *x* s'écrit :

$$\nabla_x P(x,r) = \nabla_x f(x) + 2r\sum_{j=1}^{m} Max\left\{0, c_j(x)\right\}.\nabla c_j(x)$$

Lorsque $r \longrightarrow +\infty$, $x^*(r) \longrightarrow x^*$, par continuité des fonctions $f(x)$ et $c_j(x)$ on déduit que :

$$2r.Max\left\{0, c_j\left(x^*(r)\right)\right\} \longrightarrow \lambda_j^* \quad j \in J(x^*)$$

De la même façon pour une méthode de pénalité intérieure :

$$P(x,r) = f(x) + r\sum_{j=1}^{m} -\frac{1}{c_j(x)}$$

$$\nabla_x P(x,r) = \nabla_x f(x) + r\sum_{j=1}^{m}\left(\frac{1}{c_j(x)}\right)^2 \nabla c_j(x)$$

Sachant que : $r \longrightarrow 0,\ x^*(r) \longrightarrow x^*$ on a :

$$\frac{r}{c_j{}^2(x)} \longrightarrow \lambda_j^*,\ j \in J(x^*)$$

3.3.5 Méthodes de pénalité : discussion

Les méthodes de pénalité ont fait l'objet de nombreux travaux, comme en témoigne la vaste bibliographie sur le sujet [5],[45],[67],[63], pour plus de références consulter [24]. Elles sont simples et efficaces pour résoudre les problèmes d'optimisation avec fonctions contraintes non linéaires. Les méthodes de pénalité souffrent cependant de sérieux inconvénients.

Utilisées conjointement avec une méthode de minimisation sans contraintes de type quasi newtonienne, elles permettent d'obtenir rapidement une approximation de la solution optimale et des multiplicateurs de *Kuhn et Tucker*. Mais pour parvenir à une bonne approximation de la solution , la nécessité d'utiliser des valeurs proches des valeurs extrêmes pour le coefficient de pénalité entraîne quelques difficultés d'ordre numérique, rendant délicate la minimisation de $P(x,r)$, difficultés auxquelles viennent s'ajouter les problèmes de définition des fonctions de pénalisation intérieure pour les points situés sur la frontière du domaine.

La difficulté du choix d'une valeur initiale pour le coefficient de pénalité et de son évolution au cours des itérations, empêche la réalisation de codes de calcul généraux, car la stratégie de pénalisation est spécifique à chaque problème et résulte d'un savoir faire "expérimental".

Malgré ces inconvénients les méthodes de pénalités ne sont pas à négliger car une fois ces difficultés maîtrisées elles constituent un outil efficace pour résoudre les problèmes d'optimisation fortement non linéaires. Nous verrons ultérieurement que le principe de pénalisation est utilisé avec succès dans d'autres méthodes comme la méthode utilisant les lagrangiens augmentés.

3.4 Conditions nécessaires et suffisantes d'optimalité : existence d'un point col

Commençons par définir la notion de point col.

> Le point $(x^*, y^*) \in R^n \times R^m$ est un point col de la fonction $\varphi : R^n \times R^m \longrightarrow R$ si :
>
> $$\underset{y \in R^m}{Sup}\ \varphi(x^*, y) \quad = \quad \varphi(x^*, y^*) \quad = \quad \underset{x \in R^n}{Inf}\ \varphi(x, y^*)$$

La figure 3.20 représente la surface d'équation : $\varphi(x, y) = x^2 - y^2$. On voit que la fonction $\varphi(x, y)$ admet un point col en (0,0)

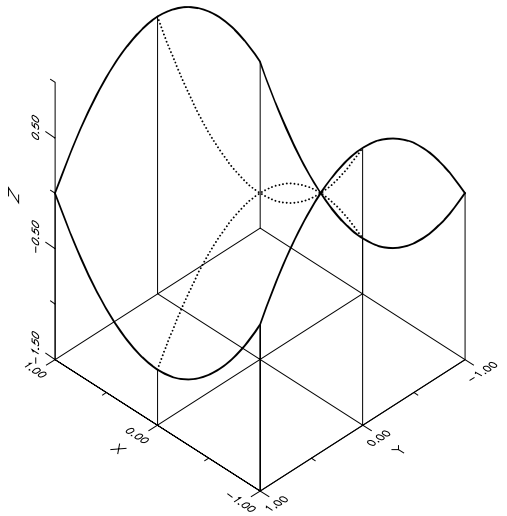


Figure 3.20 : Exemple de point col.

Cette notion de point col est fondamentale en programmation mathématique. En effet nous allons voir que si (x^*, y^*) est le point col d'une fonction $L(x, y)$ correctement choisie et associée au problème d'optimisation (P_C) alors x^* est un optimum global de (P_C).

3.4.1 Lagrangien associé au problème d'optimisation

On rappelle que (P_C) s'écrit :

$$(P_c) \begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \end{cases}$$

$$\text{Avec } f, c_j(x) \quad : \quad x \in R^n \longrightarrow f(x), c_j(x) \in R \quad j = 1..m$$

Soit la fonction de Lagrange associée à (P_C), ou lagrangien, notée $L(x, \lambda)$ définit par :

$$L\ :\ (x,\lambda) \in R^n \times R^{m+} \longrightarrow L(x,\lambda) \in R$$
$$\text{avec}: L(x,\lambda) = f(x) + \sum_{j=1}^{m} \lambda_j c_j(x)$$

On remarque que $L(x,\lambda)$ est uniquement définie pour $\lambda_j \geq 0$ puisque $\lambda \in R^{m+}$. Dans le cas où (P_C) comporterait des fonctions contraintes égalités, les variables λ_j associées à ces fonctions contraintes ne seraient pas restreintes en signe.

3.4.2 Condition suffisante d'optimalité : point col et fonction de Lagrange

En appliquant la définition d'un point col pour la fonction $L(x,\lambda)$ on peut écrire :

(x^*,λ^*) point col de $L(x,\lambda)$ si :

$$L(x^*,\lambda^*) \leq L(x,\lambda^*)\ \forall x \in R^n$$
$$L(x^*,\lambda) \geq L(x^*,\lambda^*)\ \forall \lambda \in R^{m+}$$

On peut alors établir la propriété suivante, en utilisant la définition du lagrangien et la notion de point col :

$$(x^*,\lambda^*) \text{ point col de } L(x,\lambda) \Leftrightarrow \begin{cases} a) L(x^*,\lambda^*) = \underset{x \in R^n}{Min}\ L(x,\lambda^*) \\ b) c_j(x^*) \leq 0 \forall j \in J \\ c) \lambda_j^* c_j(x^*) \leq 0 \forall j \in J \end{cases}$$

Conséquence directe de cette propriété :

$$\text{a)} \Rightarrow f(x^*) + \sum_{j=1}^{m} \lambda_j^* c_j(x^*) \quad \leq \quad f(x) + \sum_{j=1}^{m} \lambda_j^* c_j(x) \quad \forall x \in R^n$$

En utilisant b) et c) et le fait que $\lambda_j \geq 0\ \ \forall j \in [1..m]$ on a :

$$f(x^*) \leq f(x)\ \ \forall j \in [1..m] \text{ et } \forall x \in R^n \text{ tel que } c_j(x^*) \leq 0$$

Donc finalement :

Si (x^*,λ^*) point col de $L(x,\lambda)$, alors x^* **optimum global** de (P_C)

On obtient alors une nouvelle condition d'optimalité pour les problèmes avec contraintes plus générale encore que les conditions de *Kuhn et Tucker*, s'appliquant à tous les types de problèmes. En effet le résultat ci-dessus ne nécessite aucune hypothèse de convexité, continuité

ou différentiabilité pour être établi, seule la définition du lagrangien et la notion de point col sont nécessaires

L'intérêt majeur de la notion de point col est de permettre de caractériser l'existence d'un **optimum global** du problème d'optimisation, sans hypothèse de convexité, ce que les conditions d'optimalité de *Kuhn et Tucker* sont incapables de faire.

Il est important de noter que si (x^*,λ^*) est un point col de $L(x,\lambda)$, seul x^*, est solution du problème avec contraintes dans le cas général. Dans cette situation λ^* n'est pas nécessairement un multiplicateur de *Kuhn et Tucker*. Il faudra donc distinguer les multiplicateurs de point col et les multiplicateurs de *Kuhn et Tucker*. En effet dans les hypothèses du résultat précédent on ne suppose rien sur la différentiabilité des fonctions objectif et contraintes ni sur l'hypothèse de qualification des contraintes en x^*. De ce fait rien n'indique que les multiplicateurs de point col vérifient la relation :

$$\nabla f(x^*) + \sum_{j=1}^{m} \lambda_j^* \nabla c_j(x^*) = 0$$

Dans le cas particulier d'un problème convexe, c'est à dire avec une fonction objectif convexe et un domaine des solutions non vide et convexe, toute solution optimale est une solution optimale globale. On démontre que ce type de problème possède alors un point col (x^*,λ^*) et que, si les fonctions objectif et contraintes sont différentiables, le multiplicateur de point col λ^* est également un multiplicateur de *Kuhn et Tucker* [45].

Nous allons voir qu'il est possible de caractériser l'existence d'un point col de la fonction de Lagrange en introduisant la notion de "perturbation" dans un problème d'optimisation.

3.4.3 Condition d'existence d'un point col : fonction de perturbation

Soit *y* le vecteur de R^m noté :

$$y = \left[y_1 \cdots y_j \cdots y_m \right]^T$$

dont on associera chacune des *m* composantes à une fonction contrainte du problème :

$$(P_c) \begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1 \ldots m \end{cases}$$

On définit le problème perturbé (P_Y) de la manière suivante :

$$(P_Y)\begin{cases} Minimiser\ \ f(x) \\ Sous\ les\ fonctions\ contraintes \\ c_j(x) \le y_j \quad j = 1..m \end{cases}$$

Soit la fonction ϕ telle que :

$$\phi : y \in R^m \longrightarrow \phi(y) \in R.$$

Cette fonction, appelée **fonction de perturbation**, sera définie comme étant la valeur optimale globale du problème perturbé (P_Y) pour *y* parcourant R^m. On notera :

$$\phi(y) = \underset{x \in R^n}{Min} \left\{ f(x) / c_j(x) \le y_j \; j = 1..m \right\}$$

Donc pour *y*=0 on trouve $\phi(0) = f(x^*)$, où x^* est la solution optimale globale de (P_C).

Notons que, à cause de sa définition, la fonction de perturbation $\phi(y)$ est monotone décroissante, c'est à dire que :

$$\forall (y', y) \in R^m \times R^m \text{ tel que } y' \le y$$
$$\text{on a :}$$
$$\phi(y') \ge \phi(y)$$

En effet le domaine des solutions "perturbé " par *y'* noté :

$$D_{y'} = \left\{ x \in R^n / c_j(x) \le y' \; j = 1..m \right\}$$

est inclus dans le domaine des solutions "perturbé" par *y* tel que :

$$D_y = \left\{ x \in R^n / c_j(x) \le y \; j = 1..m \right\}$$

puisque $y' \le y$. Donc nécessairement le minimum global de $f(x)$ sur D_y est inférieur ou égal au minimum global de $f(x)$ sur $D_{y'}$ et par conséquent :

$$\phi(y') \ge \phi(y)$$

En utilisant les propriétés d'un point col du lagrangien établies plus haut et la définition de la fonction de perturbation on montre que [45] :

En supposant que (P_C) ait un optimum global x^* de valeur finie, λ^* est un multiplicateur de point col si et seulement si :

$$\forall y \in R^m : \phi(y) \geq \phi(0) - \lambda^{*T}.y$$

Ou autrement dit, si l'hyperplan de R^m d'équation $z(y) = \phi(0) - \lambda^{*T}.y$ est un hyperplan d'appui en y=0 du graphe de la fonction de perturbation.

Sachant que pour un problème convexe, la fonction de perturbation est convexe [45], on retrouve bien l'existence d'un point col dans le cas convexe, puisqu'une fonction convexe admet nécessairement un hyperplan d'appui en $(0, \phi(0))$.

Pour illustrer cette définition considérons l'exemple ci-dessus :
Soit le problème :

$$\begin{cases} \textit{Minimiser } f(x_1, x_2) = x_1^2 + x_2^2 \\ \textit{Sous la fonction contrainte :} \\ c_1(x_1, x_2) = 2x_1 + x_2 + 4 \leq 0 \end{cases}$$

dont on donne une représentation graphique sur la figure 3.21a

Le problème perturbé s'écrit :

$$\begin{cases} \phi(y) = Min\{x_1^2 + x_2^2\} \\ \text{Sous la fonction contrainte :} \\ c_1(x_1, x_2) = 2x_1 + x_2 + 4 - y \leq 0 \end{cases}$$

L'application des conditions de *Kuhn et Tucker* donne les équations :

$$\begin{cases} 2x_1 + 2\lambda = 0 \\ 2x_2 + \lambda = 0 \\ \lambda(2x_1 + x_2 + 4 - y) = 0 \end{cases} \Rightarrow \begin{cases} x_1 = -\lambda \\ x_2 = -\lambda/2 \\ \lambda(-2\lambda - \lambda/2 + 4 - y) = 0 \end{cases}$$

On obtient les solutions suivantes :

$$\begin{cases} x_1 = 0; x_2 = 0 si \lambda = 0 \\ x_1 = \dfrac{2y-8}{5}; x_2 = \dfrac{y-4}{5} si \lambda \neq 0 \end{cases}$$

La fonction de perturbation a pour expression :

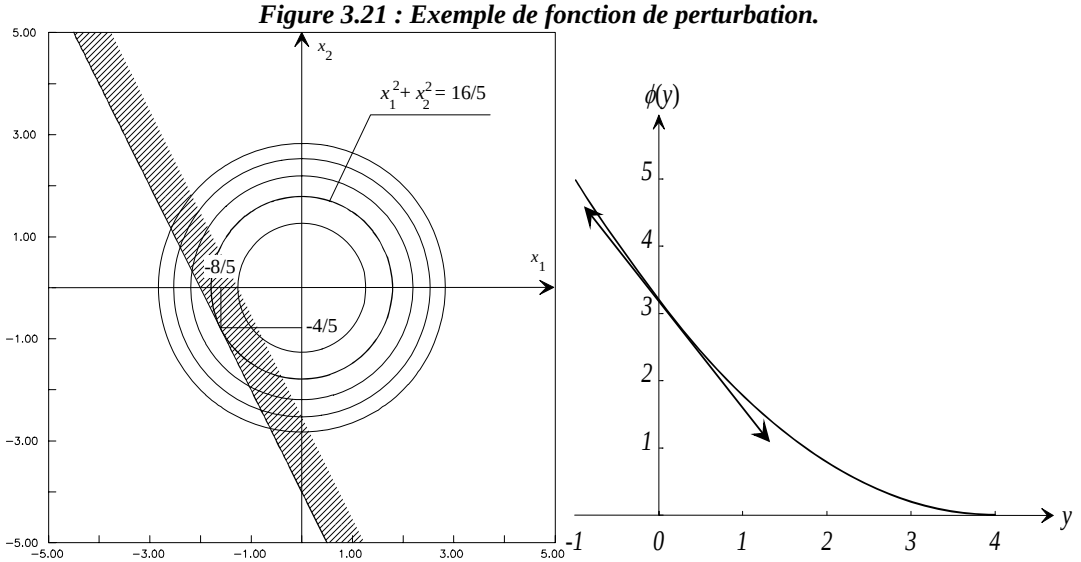
$$\begin{cases} \phi(y) = 0 si \lambda = 0 \\ \phi(y) = \left(\frac{2y-8}{5}\right)^2 + \left(\frac{y-4}{5}\right)^2 = \frac{1}{5}y^2 - \frac{8}{5}y + \frac{16}{5} si \lambda \neq 0 \end{cases}$$

Remarque :

Le cas $\lambda \neq 0$ correspond à : $y \leq 4$.

Le cas $\lambda = 0$ correspond à : $y \leq 4$, la contrainte "passe derrière" le point (0,0)

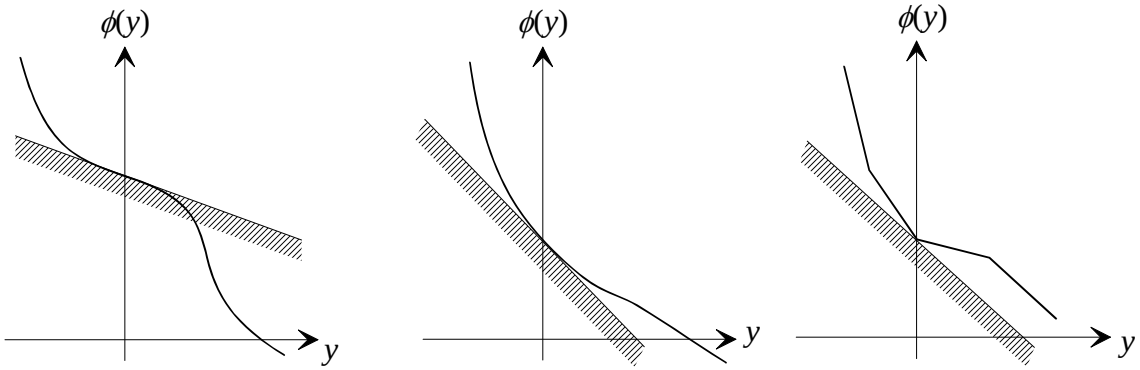
Figure 3.21 : Exemple de fonction de perturbation.



(a) : Problème convexe. ***(b) : Fonction de perturbation associée.***

La fonction de perturbation est représentée sur la figure 3.21b. Il s'agit d'une fonction convexe admettant une tangente de pente $-8/5 = -\lambda$ en y=0. D'où l'existence d'un point col dont le multiplicateur est $8/5$. Dans cet exemple, le problème étant convexe la fonction de perturbation est elle aussi convexe et admet nécessairement un hyperplan d'appui en $(0, \phi(0))$.

Figure 3.22 : Exemple de fonctions de perturbation.



(a) : Pas de point col ***(b) : ϕ n'est pas convexe mais il existe un point col*** ***(c) : ϕ n'est ni convexe, ni différentiable mais il existe un point col.***

La figure 3.22 représente un certain nombre de cas de fonctions de perturbation pour un problème d'optimisation à une seule fonction contrainte. Dans ce cas *y* et λ sont des scalaires. On constate que l'existence d'un hyperplan d'appui pour le graphe de la fonction de perturbation, donc d'un point col, n'est pas limitée aux problèmes convexes et différentiables. (figure 3.22b et 3.22c). Il existe cependant de nombreux cas de problèmes non convexes pour lesquels il n'y a pas de point col (figure 3.22a). Malheureusement ces cas sont extrêmement difficiles à caractériser

3.5 Dualité lagrangienne

Grâce à la fonction de Lagrange associée au problème d'optimisation nous avons pu introduire la notion de point col et faire apparaître une nouvelle condition d'optimalité. Nous allons voir comment la fonction de Lagrange permet également de définir l'image du problème dans un autre espace de variables et transformer un problème d'optimisation avec fonctions contraintes en un problème d'optimisation sans fonction contrainte.

3.5.1 Définition de la fonction duale et du problème dual

Nous venons d'établir que si (x^*,λ^*) point col de $L(x,\lambda)$ alors x^* est solution optimale globale de (P_C). Donc si on sait déterminer un point col de la fonction de Lagrange, le problème est résolu.
D'après la définition d'un point col on a (cf. § 3.4.2)

$$(x^*,\lambda^*) \text{ point col de } L(x,\lambda) \Leftrightarrow \begin{cases} L(x^*,\lambda^*) \le L(x,\lambda^*) \forall x \in R^n \\ L(x^*,\lambda^*) \ge L(x^*,\lambda) \forall \lambda \in R^{m+} \end{cases}$$

La recherche d'un point col de $L(x,\lambda)$ se ramène alors au problème :

$$L(x^*,\lambda^*) = \underset{\lambda \in R^{m+}}{Max} \left\{ \underset{x \in R^n}{Min}\, L(x,\lambda) \right\}$$

En définissant la fonction duale de la manière suivante :

$$w : \lambda \in R^{m+} \longrightarrow w(\lambda) \in R$$
telle que :
$$w(\lambda) = L(x^*,\lambda) = \underset{x \in R^n}{Min}\, L(x,\lambda)$$

On obtient alors le problème dual :

$$(D_C) \begin{cases} L(x^*,\lambda^*) = \underset{\lambda}{Max}\, w(\lambda) \\ \lambda \ge 0 \end{cases}$$

Bien que nous ayons utilisé la notion de point col pour introduire la définition de la fonction duale, celle-ci reste parfaitement définie même lorsqu'il n'existe pas de point col, il en est de même pour le problème dual.

Le problème dual (D_C) associé au problème "primal" (P_C) est bien un problème de maximisation sans fonction contrainte d'une fonction de *m* variables, la fonction duale, dans R^{m+} l'espace des variables duales λ par opposition à R^n l'espace des variables primales *x*.

3.5.2 Propriétés de la fonction duale

La fonction duale possède un certain nombre de propriétés, que l'on peut directement établir à partir de sa définition [45]:

Propriété 1:

La fonction duale est une fonction **concave** de λ. Cette propriété s'établit sans **aucune hypothèse sur la convexité** des fonctions objectif et contraintes de (P_C). Conséquence directe de cette concavité : tout optimum local de $w(\lambda)$ est un optimum global (non nécessairement unique). C'est une raison pour laquelle le problème dual est souvent plus facile à résoudre que le problème primal.

Propriété 2:

La fonction duale $w(\lambda)$ est un minorant de l'optimum global, x^* de (P_C). Si $w(\lambda^*)$ est la valeur optimale de la fonction duale on a la relation :

$$\forall \lambda \geq 0 : w(\lambda) \leq w(\lambda^*) \leq f(x^*)$$

Cette propriété est intéressante car sans connaître λ^*, l'optimum de la fonction duale, $w(\lambda)$ constitue une borne inférieure pour la valeur optimale globale du problème primal.

L'importance de l'existence d'un point col pour un problème d'optimisation, et sa conséquence sur le problème dual associé est mise en évidence grâce au résultat suivant :

Théorème de dualité :

Si le problème (P_C) admet un point col (x^*, λ^*) alors on a:

$$Max\{(D_C)\} = w(\lambda^*) = f(x^*) = Min\{(P_C)\}$$

et réciproquement, s'il existe x^* solution de (P_C) et $\lambda^* \geq 0$ tel que :

$$w(\lambda^*) = f(x^*)$$

alors (P_C) admet (x^*, λ^*) comme point col.

Dans le cas de l'exemple du paragraphe 3.4.3, l'application de ce théorème donne rapidement la solution de ce problème convexe :.

$$\begin{cases} \textit{Minimiser } f(x_1,x_2) = x_1^2 + x_2^2 \\ \textit{Sous la fonction contrainte :} \\ c_1(x_1,x_2) = 2x_1 + x_2 + 4 \le 0 \end{cases}$$

Comme il existe un point col on a: $w(\lambda^*) = f(x^*)$. Le lagrangien s'écrit :

$$L(x,\lambda) = \left(x_1^{\,2} + x_2^{\,2}\right) + \lambda(2x_1 + x_2 + 4)$$

Le minimum en x de $L(x,\lambda)$ est défini par :

$$\begin{cases} \dfrac{\partial L}{\partial x_1} = 2x_1 + 2\lambda = 0 \Rightarrow x_1 = -\lambda \\ \dfrac{\partial L}{\partial x_2} = 2x_2 + \lambda = 0 \Rightarrow x_2 = -\dfrac{\lambda}{2} \end{cases}$$

L'expression de la fonction duale est :

$$w(\lambda) = \underset{x \in R^n}{Min}\, L(x,\lambda) = -\frac{5\lambda^2}{4} + 4\lambda$$

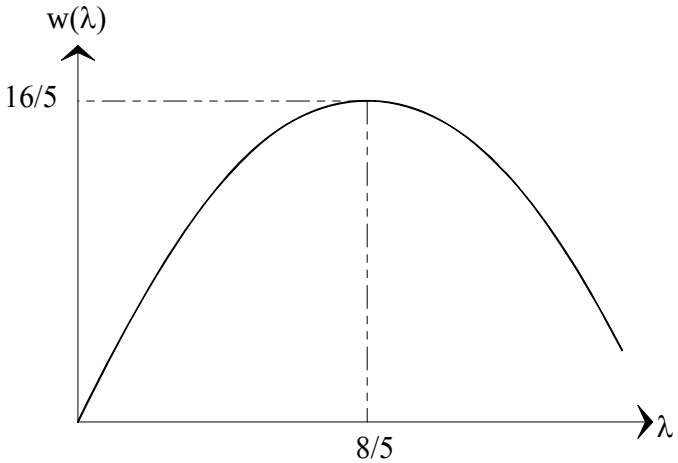


Figure 3.23 : Concavité de la fonction duale.

On peut vérifier sur la figure 3.23 la propriété de concavité de $w(\lambda)$. Le maximum de $w(\lambda)$ est atteint pour :

$$\frac{\partial w}{\partial \lambda} = \frac{-5\lambda}{2} + 4 = 0 \Rightarrow \lambda^* = \frac{8}{5}$$

On a alors :

$$w(\lambda^*) = f(x^*) = \frac{16}{5} \text{ et } x^* = (-8/5, -4/5)$$

Le calcul de la fonction duale nécessite, pour chaque valeur de λ, la détermination du minimum global du lagrangien $L(x,\lambda)$ en x. Cette opération n'est possible que si le lagrangien possède un minimum unique en x pour λ fixé. Cette condition peut être remplie dans le cas convexe, mais également dans le cas de fonctions quelconques. On concevra cependant qu'il est très difficile de caractériser ce dernier cas.

Si $\bar{x}$ est l'unique minimum de $L(x,\lambda)$ en x pour λ fixé, on montre que la fonction duale est différentiable et que le gradient de $w(\lambda)$ au point λ est :

$$\nabla_\lambda w(\lambda) = \left[c_1(\bar{x}) \ldots c_j(\bar{x}) \ldots c_m(\bar{x})\right]^T \in R^m$$

3.5.3 Propriétés du problème dual

Soit λ^* un optimum du dual (D_C). On notera que λ^* est nécessairement un optimum global puisque la fonction duale est convexe. On a les possibilités suivantes :

a) Si (P_C) admet un point col, alors il existe x^* solution de (P_C) tel que (x^*, λ^*) soit un point col.
b) Si (P_C) admet un point col, et si $L(x,\lambda)$ a un minimum unique en x, x^*, alors (x^*, λ^*) est un point col de (P_C).
c) Si on suppose $w(\lambda)$ différentiable en λ^* et si $L(x,\lambda^*)$ a un minimum unique en x, x^*, alors (x^*, λ^*) est un point col de (P_C).

3.6 Méthodes duales

En supposant que les hypothèses b) et c) du §3.5.3 soient vérifiées, et que l'on connaisse la valeur de λ^*, l'optimum de la fonction duale, la solution du problème peut être obtenue en une seule minimisation en x du lagrangien $L(x,\lambda)$. Dans ce cas (x^*, λ^*) est un point col de $L(x,\lambda)$ et x^* l'optimum global du problème. De cette manière la résolution d'un problème avec fonctions contraintes se ramène bien à celle d'un problème sans contrainte.

Généralement on ne connaît pas la valeur de λ^*. On peut cependant la déterminer par une méthode itérative engendrant une suite de point λ^k maximisant la fonction duale $w(\lambda)$. Dans ce cas si la suite des points λ^k converge vers λ^*, la suite des points x^k obtenus en minimisant $L(x,\lambda^k)$ converge vers x^*, solution optimale globale du problème.

La méthode d'*Uzawa* par exemple utilise une procédure classique de gradient pour maximiser la fonction duale. On obtient alors une méthode de type "plus forte pente" (Cf §2.5.1)

appliquée à la fonction duale. L'utilisation d'une méthode de gradient, bien que motivée par la facilité de calcul puisque :

$$\nabla_\lambda w(\lambda) = \left[c_1\left(x(\lambda^k)\right)\ldots c_j\left(x(\lambda^k)\right)\ldots c_m\left(x(\lambda^k)\right)\right]^T \in R^m$$

où $x(\lambda^k)$ est l'unique minimum de $L(x,\lambda^k)$

n'est en fait pas très judicieuse.

En effet connaissant les limitations à propos de la vitesse de convergence des méthodes de gradient, la méthode d'*Uzawa* convergera linéairement avec une vitesse de convergence d'autant plus lente que le conditionnement de la matrice du hessien de la fonction duale :

$$\nabla_\lambda{}^2 w(\lambda) = -A.\left[\nabla_x{}^2 L(x^*,\lambda^*)\right]^{-1}.A^T$$

$$\text{avec } A = \left[\frac{\partial c_j}{\partial x_i}(x^*)\right]_{\substack{i=1..n\\ j=1..m}}$$

sera mauvais.

Pour améliorer la vitesse de convergence on peut profiter du fait que les méthodes quasi newtoniennes utilisées pour minimiser $L(x,\lambda)$ procurent en fin de convergence une approximation de l'inverse du hessien du lagrangien : $H \approx \left[\nabla_x{}^2 L(x,\lambda)\right]^{-1}$. Le calcul du hessien de la fonction duale est plus simple puisque :

$$\nabla_\lambda{}^2 w(\lambda) = -A.H.A^T$$

les gradients des contraintes dans la matrice *A* étant évalués en $x^*(\lambda)$ le minimum en *x* de $L(x,\lambda)$. A partir de là, on peut appliquer la formule itérative d'une méthode de type "Newton" pour le calcul de λ^{k+1} :

$$\lambda^{k+1} = \lambda^k - \left[\nabla_\lambda{}^2 w(\lambda)\right]^{-1}.\left[\nabla_\lambda{}^2 w(\lambda)\right]$$

Minoux [45] rapporte que cette formule itérative conduit à une vitesse de convergence superlinéaire.

L'utilisation de la dualité lagrangienne classique pour résoudre des problèmes non linéaires n'est pas fondamentalement intéressante, en effet les restrictions imposées par la présence d'un point col de la fonction de Lagrange, et surtout l'unicité du minimum en *x* du lagrangien font que pratiquement un algorithme utilisant ce type de méthodes ne sera utilisable que sur des problèmes convexes.

Il existe cependant un certain nombre de cas où la résolution du problème dual est plus avantageuse que celle du primal. Notamment dans les problèmes linéaires comportant beaucoup de fonctions contraintes par rapport au nombre de variables.

En effet considérons le problème linéaire :

$$\begin{cases} \textit{Minimiser} \;\; f(x) = c^T.x \\ \textit{Sous les fonctions contraintes :} \\ A.x + b \le 0 \end{cases}$$

Avec $c \in R^n$, $b \in R^m$ et A matrice $m \times n$ de rang n.

La fonction de Lagrange s'écrit :

$$L(x,\lambda): R^n \times R^{m+} \longrightarrow R$$
$$L(x,\lambda) = c^T.x + \lambda^T.\left[A.x + b\right]$$

La fonction duale peut être calculée explicitement et :

$$w(\lambda) = \underset{x \in R^n}{Min}\left\{L(x,\lambda)\right\} = \underset{x \in R^n}{Min}\left\{\left(c - \lambda^T.A\right).x + \lambda^T.b\right\}$$
$$\Rightarrow w(\lambda) = \begin{cases} \lambda^T.b \forall \lambda \in R^m \text{ tel que} : c - \lambda^T.A = 0 \\ -\infty \text{ dans les autres cas} \end{cases}$$

D'où le problème dual :

$$\begin{cases} \textit{Minimiser} \;\; w(\lambda) = \lambda^T.b \\ \textit{Sous les fonctions contraintes :} \\ c - \lambda^T.A = 0 \\ \lambda \ge 0 \end{cases}$$

Le problème étant par définition convexe, il existe un point col pour la fonction de Lagrange et le maximum du problème dual est égal au minimum du problème primal. Donc lorsque le primal comporte beaucoup de contraintes $(m >> n)$, le dual est plus simple à résoudre puisqu'il ne comporte que n contraintes pour m variables.

Lorsque la fonction objectif et les fonctions contraintes sont non linéaires mais séparables, c'est à dire lorsque :

$$f(x) = f_1(x_1) + f_2(x_2) + \cdots + f_n(x_n)$$
et
$$c_j(x) = c_{j1}(x_1) + c_{j2}(x_2) + \cdots + c_{jn}(x_n) \;\; j = 1..m$$

avec :

$$x = \left[x_1, x_2, \ldots, x_n\right]^T \in R^n$$

et

$$f_i, c_{ji} \ : \ x_i \in R \longrightarrow f(x_i), c_{ji}(x_i) \in R \ \ pour \ \ i = 1..n, j = 1..m$$

les méthodes utilisant la dualité, présentent dans ce cas, un intérêt certain. En effet, le lagrangien associé au problème s'écrit :

$$\begin{aligned} L(x,\lambda) &= \sum_{i=1}^{n} f_i(x) + \sum_{j=1}^{m} \left(\lambda_j \sum_{i=1}^{n} c_{ji}(x) \right) \\ &= \sum_{i=1}^{n} \left(f_i(x) + \lambda_j \sum_{j=1}^{m} c_{ji}(x) \right) \end{aligned}$$

En utilisant le fait que le minimum d'une fonction séparable est obtenu lorsque chaque terme est minimal, la fonction duale a pour expression :

$$w(\lambda) = \underset{x \in R^n}{Min} \{ L(x,\lambda) \} = \sum_{i=1}^{n} \left(\underset{x_i \in R}{Min} \left\{ f_i(x) + \lambda_j \sum_{j=1}^{m} c_{ji}(x) \right\} \right)$$

Le calcul de $w(\lambda)$ est plus simple, puisqu'il peut être effectué grâce à une minimisation unidimensionnelle des termes $\left\{ f_i(x) + \lambda_j \sum_{j=1}^{m} c_{ji}(x) \right\}$. Dans certains cas il peut même être fait explicitement, notamment lorsque les fonctions $f_i(x)$ et $c_{ji}(x)$ sont issues d'une linéarisation. Cette particularité permet la mise au point de méthodes très performantes pour les problèmes d'optimisation de structures, dans lesquels le nombre de variables et le coût d'évaluation des différentes fonctions sont très élevés.

3.7 Lagrangiens généralisés : le lagrangien augmenté

Nous venons de voir que les méthodes duales "classiques", ne présentent un intérêt que pour les problèmes dont le lagrangien ordinaire admet un point col, donc dans une majorité de cas pratiques, pour des problèmes convexes. On se propose de montrer comment la combinaison d'une méthode de pénalité et d'une méthode duale permet d'élargir l'existence d'un point col d'une fonction de lagrange dite "généralisée" dans le cas de problèmes non convexes.

3.7.1 Interprétation graphique des méthodes duales classiques et des méthodes de pénalité

La base de cette interprétation graphique repose sur le lien que l'on va établir entre la fonction de perturbation et la fonction duale. Afin de simplifier l'écriture des différentes relations et surtout pour permettre des graphes de fonction d'une seule variable nous supposerons que le problème comporte une fonction contrainte, cela n'altérant pas le caractère général des explications qui vont suivre.

Soit le problème :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous la fonction contrainte} \\ c(x) \le 0 \end{cases}$$

$$\text{Avec } f, c : x \in R^n \longrightarrow f(x), c(x) \in R$$

La fonction duale $w : \lambda \in R^+ \longrightarrow w(\lambda) \in R$ est telle que :

$$w(\lambda) = \underset{x \in R^n}{Min} \{ f(x) + \lambda c(x) \} = f(\overline{x}) + \lambda c(\overline{x}) \ , \ \forall \lambda \in R^+$$

On peut également, $\forall y \in R$ définir le problème perturbé par :

$$(P_Y)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous la fonctions contrainte} \\ c(x) \le y \end{cases}$$

La fonction de perturbation $\phi : y \in R \longrightarrow \phi(y) \in R$ s'écrit :

$$\phi(y) = \underset{x \in R^n}{Min} \{ f(x) / c(x) \le y \}$$

On remarque que $\forall y \in R$ tel que (P_y) ait une solution x on a :

$$\phi(y) = f(x) \text{ et } c(x) \le y$$

Donc :

$$\begin{array}{c} \forall \lambda \ge 0 \ , \ \lambda c(x) \le \lambda y \\ \Rightarrow \quad f(x) + \lambda c(x) \quad \le \quad \phi(y) + \lambda y \end{array}$$

Comme par définition, $w(\lambda) \le f(x) + \lambda c(x), \forall x \in R^n$

On déduit immédiatement la relation liant la fonction duale et la fonction de perturbation :

$$\boxed{\begin{array}{c}\forall \lambda \geq 0 \text{ et } \forall y \in R \text{ tel que } (P_y) \text{ ait une solution } x \\ w(\lambda) \leq \phi(y) + \lambda y\end{array}}$$

En réutilisant le fait que :

$$w(\lambda) = f(\overline{x}) + \lambda c(\overline{x}) \ , \ \forall \lambda \in R^+$$

et en posant : $\overline{y} = c(\overline{x})$ on a : $\phi\big(c(\overline{x})\big) = \phi(\overline{y}) = f(\overline{x})$.

Finalement on obtient :

$$\begin{cases} \phi(y) \geq w(\lambda) - \lambda y \\ \phi(\overline{y}) = w(\lambda) - \lambda \overline{y} \text{ pour } \overline{y} = c(\overline{x}) \end{cases}$$

On constate alors que le graphe de $\phi(y)$ est entièrement situé au-dessus d'une droite de pente $-\lambda$ et d'ordonnée à l'origine $w(\lambda)$, et qu'il est tangent à cette droite au point de coordonnées $\big(\overline{y}, \phi(\overline{y})\big)$.

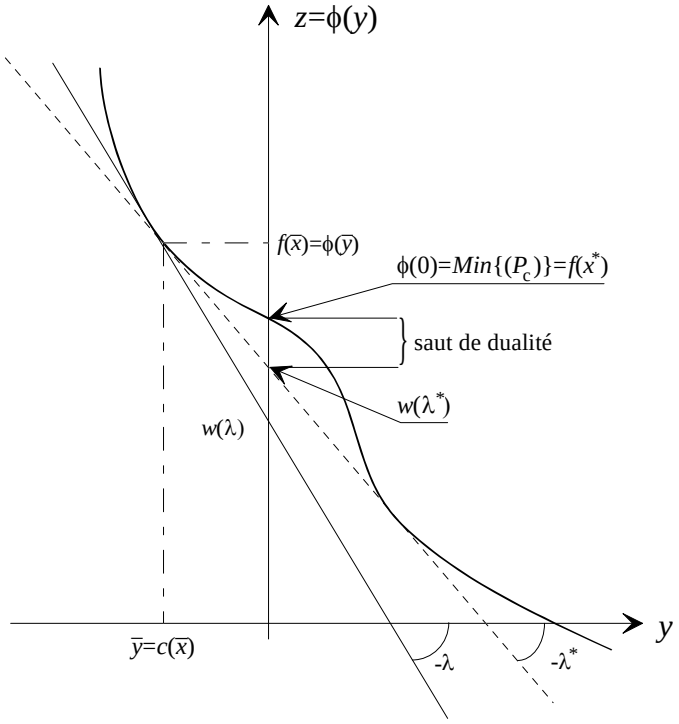


Figure 3.24 : Interprétation graphique d'une méthode duale classique.

La fonction de perturbation représentée sur la figure 3.24 n'admet pas au point de coordonnées $\big(0, \phi(0)\big)$ de tangente entièrement située sous le graphe de $\phi(y)$, donc il n'existe pas de point col pour la fonction de Lagrange associée au problème. Dans ce cas de figure une méthode basée sur la dualité lagrangienne ne peut pas converger vers la solution optimale

globale, x^*, du problème (P_c). On obtient cependant un minorant de la solution optimale, l'écart entre $w(\lambda^*)$ et $f(x^*)$ étant appelé : **saut de dualité**. La valeur optimale de la fonction duale, $w(\lambda^*)$, représente l'ordonnée à l'origine maximale d'une droite de pente $-\lambda^*$ tangente au graphe de $\phi(y)$.

Considérons maintenant la fonction de pénalisation extérieure pour le problème (P_c) :

$$P \ : \ R^n \times R^+ \longrightarrow P(x,r) \in R$$
telle que :
$$P(x,r) = f(x) + r.\left(Max\{0, c(x)\}\right)^2 \text{ avec } r > 0$$

Comme la fonction duale $w(\lambda)$ définie à partir de la fonction de Lagrange, on peut définir une fonction similaire $w(r)$ telle que :

$$w \ : \ r \in R^+ \longrightarrow w(r) \in R$$
avec :
$$w(r) = \underset{x \in R^n}{Min} \left\{ f(x) + r.\left(Max\{0, c(x)\}\right)^2 \right\}$$

On montre très simplement que cette fonction possède les mêmes propriétés que la fonction duale classique; en fait on montre que la fonction de pénalisation $P(x,r)$ est une représentation lagrangienne du problème (P_c) au même titre que $L(x,\lambda)$ [45].

En utilisant de nouveau la fonction de perturbation et notamment le fait que :

$$\forall y \in R \text{ tel que } (P_y) \text{ ait une solution } x,\ \phi(y) = f(x) \text{ et } c(x) \le y$$

on a la relation :

$$w(r) \le f(x) + r.\left(Max\{0, c(x)\}\right)^2 \ , \ \forall x \in R^n$$
$$\Rightarrow w(r) \le \phi(y) + r.\left(Max\{0, c(x)\}\right)^2 \le \phi(y) + r.\left(Max\{0, y\}\right)^2$$

En écrivant que :

$$w(r) = f(\bar{x}) + r.\left(Max\{0, c(\bar{x})\}\right)^2$$

et en posant $\bar{y} = c(\bar{x})$, on a toujours $\phi(\bar{y}) = f(\bar{x})$. En effet s'il existait x' tel que : $f(x') < f(\bar{x})$ et $c(x') < \bar{y} = c(\bar{x})$ on aurait :

$$f(x') + r.\left(Max\{0, c(x')\}\right)^2 \le f(\bar{x}) + r.\left(Max\{0, c(\bar{x})\}\right)^2$$

d'où une contradiction avec la définition de $w(r)$.

Finalement on obtient le résultat suivant :

$$\begin{cases} \phi(y) \geq w(\lambda) - r.(Max\{0, y\})^2 \\ \phi(\bar{y}) = w(\lambda) - r.(Max\{0, \bar{y}\})^2 \text{ pour } \bar{y} = c(\bar{x}) \end{cases}$$

D'où l'on déduit que le graphe de $\phi(y)$ est toujours situé au-dessus d'une courbe d'équation :

$$\begin{cases} z = -ry^2 + w(r) & \text{pour y} \geq 0 \\ z = w(r) & \text{pour y} < 0 \end{cases} \qquad (18)$$

et tangent à celle-ci au point de coordonnées : $(\bar{y}, \phi(\bar{y}))$.

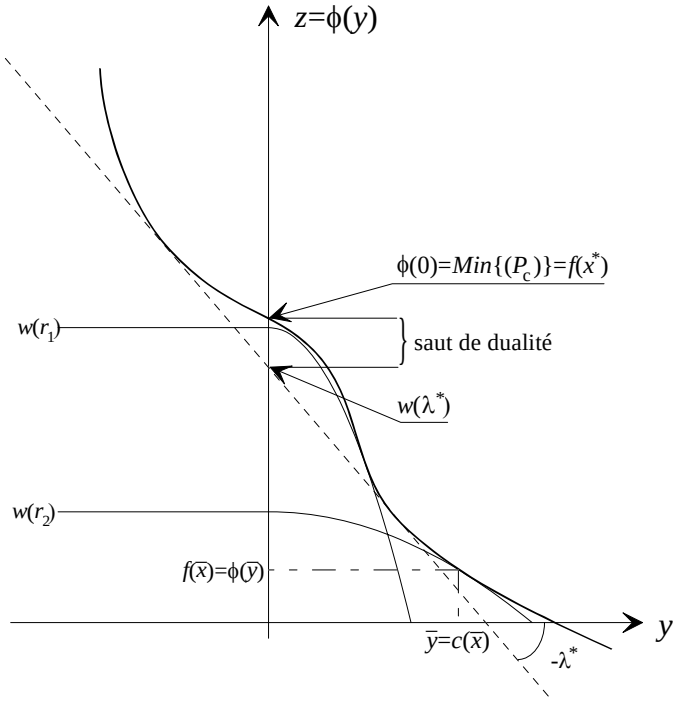


Figure 3.25 : Interprétation graphique d'une méthode de pénalité.

La figure 3.25 représente la courbe d'équation (18) pour deux valeurs différentes du coefficient de pénalité, r_1 et r_2 avec $r_1 > r_2$. On voit que $w(r)$ augmente lorsque le coefficient de pénalité *r* augmente. Ce qui est confirmé par les relations :

$\phi(y) \leq \phi(0)$, $\forall y \geq 0$ (Propriété de monotonie de ϕ)
et de plus :
$w(r) = \phi(\bar{y}) + r\bar{y}^2 \ \leq \ \phi(0)$ pour $\bar{y} \geq 0$

Donc lorsque $r \longrightarrow +\infty$, $w(r)$ tend vers $\phi(0)$ même si le lagrangien classique présente un saut de dualité. Dans ce cas le sommet de la parabole, situé sur l'axe $z = \phi(y)$, tend à être tangent au graphe de $\phi(y)$. Comme, d'autre part $\bar{y} \longrightarrow 0$ lorsque $r \longrightarrow +\infty$, la suite des points x^k obtenue en minimisant $P(x,r)$ converge vers une solution optimale (globale) du problème non perturbé, sous réserve que x^k soit bien le minimum absolu de $P(x,r)$.

D'un point de vue strictement théorique une méthode de pénalité pourra s'affranchir de l'absence de point col du lagrangien classique, et convergera donc vers la solution optimale ce qui est impossible à obtenir avec une méthode duale classique. Cependant le fait qu'il faille faire tendre le coefficient de pénalité vers des valeurs extrêmes pour obtenir un point col de la fonction de pénalisation pose des problèmes numériques dans la minimisation de $P(x,r)$.

3.7.2 Le lagrangien augmenté

C'est à *Hestenes* (1969) et indépendamment *Powel* (1969) que l'on doit l'idée d'une combinaison d'une expression de type pénalité extérieure et du lagrangien classique. Parallèlement les travaux de *Rockafellar* sur l'analyse convexe [57] ont permis d'interpréter et de justifier l'efficacité de cette combinaison. Initialement prévu pour les problèmes d'optimisation avec des fonctions contraintes égalités, ce principe sera également généralisé par *Rockafellar* [58] aux fonctions contraintes inégalités.

Pour résoudre le problème :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) = 0 \quad j = 1...m \end{cases}$$

Hestenes et *Powel* ont proposé de le remplacer par une séquence de minimisation de la fonction :

$$\hat{L} \quad : \quad (x,\lambda,r) \in R^n \times R^{m+} \times R^+ \longrightarrow \hat{L}(x,\lambda,r) \in R$$

telle que :

$$\hat{L}(x,\lambda,r) = L(x,\lambda) + r.\sum_{j=1}^{m} c_j{}^2(x) = f(x) + \sum_{j=1}^{m}\left(\lambda_j c_j(x) + r.c_j{}^2(x)\right) \qquad (19)$$

En introduisant des variables d'écarts $s_j \geq 0$, $j = 1..m$ la généralisation à un problème avec des fonctions contraintes inégalités est immédiate [58], puisque :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \leq 0 \quad j = 1...m \end{cases}$$

se ramène à :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) + s_j = 0 \quad j = 1...m \end{cases}$$

On doit alors résoudre une séquence de problème sans contrainte de la forme :

$$\underset{\substack{x\in R^n \\ s\geq 0}}{Min}\left\{\hat{L}(x,\lambda,s,r)\right\} = \underset{\substack{x\in R^n \\ s\geq 0}}{Min}\left\{ f(x) + \sum_{j=1}^{m}\left(\lambda_j\left(c_j(x)+s_j\right) + r.\left(c_j(x)+s_j\right)^2\right)\right\} \qquad (20)$$

avec :

$$\hat{L} \quad : \quad (x,\lambda,s,r) \in R^n \times R^{m+} \times R^{m+} \times R^{+} \longrightarrow \hat{L}(x,\lambda,s,r) \in R$$

Cette expression se simplifie en remarquant que pour *x* fixé, la minimisation suivant les variables s_j peut être faite explicitement.

En effet :

$$\frac{\partial \hat{L}}{\partial s_j}(x,\lambda,s,r) = \lambda_j + 2r.\left(c_j(x)+s_j\right) = 0$$

$$\Rightarrow s_j = -\frac{\lambda_j}{2r} - c_j(x)$$

La condition $s_j \geq 0$ entraîne :

$$c_j(x) \leq -\frac{\lambda_j}{2r}$$

Dans ce cas l'expression (20) vaut :

$$f(x) + \sum_{j=1}^{m}\left(\lambda_j\left(-\frac{\lambda_j}{2r}\right) + r.\left(-\frac{\lambda_j}{2r}\right)^2\right)$$

Par contre dans la situation où $c_j(x) \leq -\frac{\lambda_j}{2r}$ l'expression :

$$\lambda_j s_j + r\left(c_j(x)+s_j\right)^2$$

est minimale pour $s_j = 0$ et donc (20) est équivalent à :

$$f(x) + \sum_{j=1}^{m}\left(\lambda_j c_j(x) + r.c_j^{\,2}(x)\right)$$

Finalement on obtient l'expression du lagrangien augmenté de *Rockafellar* [58] :

$$\hat{L}(x,\lambda,r) = f(x) + \sum_{j=1}^{m}\left(\lambda_j \Psi_j(x) + r.\Psi_j^{\,2}(x)\right) \qquad (21)$$

$$\text{avec } \Psi_j(x) = Max\left\{c_j(x), -\frac{\lambda_j}{2r}\right\} \text{ et } r > 0$$

Dans le cas d'un problème comportant à la fois des fonctions contraintes inégalités et égalités on peut combiner les expressions (19) et (21).

En appliquant la démarche du § 3.7.1, et donc en considérant un problème avec une seule fonction contrainte inégalité, l'expression du lagrangien augmenté devient :

$$\hat{L}(x,\lambda,r) = f(x) + \lambda\Psi(x) + r.\Psi^2(x)$$
$$\text{avec } \Psi(x) = Max\left\{c(x), -\frac{\lambda}{2r}\right\} \text{ et } r > 0$$

En s'appuyant sur le fait que $\hat{L}(x,\lambda,r)$ est aussi une représentation lagrangienne du problème (P_c) du § 3.7.1 on peut définir la fonction duale augmentée par :

$$\hat{w}(\lambda,r) = \underset{x\in R^n}{Min}\left\{f(x) + \lambda\Psi(x) + r.\Psi^2(x)\right\} = f(\bar{x}) + \lambda\Psi(\bar{x}) + r.\Psi^2(\bar{x})$$

En gardant la même définition pour le problème perturbé et la fonction de perturbation associée, on a :

$$\forall y \in R \text{ tel que } (P_y) \text{ ait une solution } x,\ \phi(y) = f(x) \text{ et } c(x) \le y$$

Donc :

$$\hat{w}(\lambda,r) \le f(x) + \lambda\Psi(x) + r.\Psi^2(x)$$
$$\Rightarrow \hat{w}(\lambda,r) \le \phi(y) + \lambda Max\left\{c(x), -\frac{\lambda}{2r}\right\} + r.\left(Max\left\{c(x), -\frac{\lambda}{2r}\right\}\right)^2$$
$$\Rightarrow \hat{w}(\lambda,r) \le \phi(y) + \lambda Max\left\{y, -\frac{\lambda}{2r}\right\} + r.\left(Max\left\{y, -\frac{\lambda}{2r}\right\}\right)^2$$

Comme $\phi(\bar{y}) = f(\bar{x})$ pour $\bar{y} = c(\bar{x})$ l'égalité :

$$\hat{w}(\lambda,r) = \phi(\bar{y}) + \lambda Max\left\{\bar{y}, -\frac{\lambda}{2r}\right\} + r.\left(Max\left\{\bar{y}, -\frac{\lambda}{2r}\right\}\right)^2$$

est également vérifiée.

On obtient donc les relations suivantes :

$$\begin{cases} \phi(y) \ge \hat{w}(\lambda,r) - \lambda y - ry^2 & y \ge -\dfrac{\lambda}{2r} \\ \phi(\bar{y}) = \hat{w}(\lambda,r) + \dfrac{\lambda^2}{4r} & y < -\dfrac{\lambda}{2r} \end{cases}$$

Ce résultat montre que la valeur de la fonction duale augmentée est l'ordonnée de l'intersection avec l'axe $y = 0$ et d'une parabole d'équation :

$$z(y) = \hat{w}(\lambda, r) - \lambda y - r y^2$$

La figure 3.26 montre clairement que cette parabole est située partout au-dessous du graphe de $\phi(y)$ et qu'elle est tangente à ce graphe au point de coordonnées $(\bar{y}, \phi(\bar{y}))$, la pente de la tangente en ce point étant de $-\lambda$.

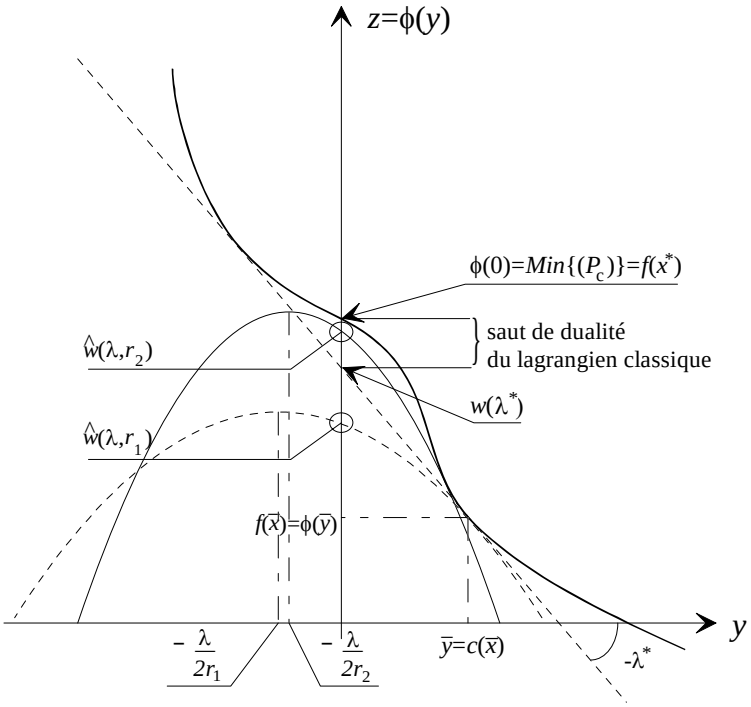


Figure 3.26 : Interprétation graphique du lagrangien augmenté.

On retrouve ici le fait que quand r augmente, la fonction duale augmentée $\hat{w}(\lambda, r)$ augmente et tend vers $\phi(0)$. La différence essentielle par rapport à une méthode de pénalité se situe dans la position du sommet de la parabole. En effet dans le cas du lagrangien augmenté le sommet de cette parabole est décalé de $-\dfrac{\lambda}{2r}$ par rapport à l'axe $z = \phi(y)$ par conséquent le point de tangence au graphe de $\phi(y)$ se situera nécessairement non plus au sommet, mais sur le coté de cette parabole. On conçoit alors que l'égalité :

$$\hat{w}(\lambda, r) = \phi(0) = Min\{(P_c)\} \qquad (22)$$

traduisant l'existence d'un point col pour le lagrangien augmenté, s'obtiendra pour les valeurs de r bien moins élevées que dans le cas d'une méthode de pénalité.

La relation (22) se traduit alors par la condition :

"La fonction de perturbation admet au point $(0, \phi(0))$ une fonction de support quadratique concave" [45]

On conçoit que cette condition, à mettre en parallèle avec l'existence d'un hyperplan d'appui dans le cas du lagrangien ordinaire, puisse être remplie par une classe de fonctions beaucoup plus vaste que celle des fonctions convexes.

3.7.3 Méthode utilisant le lagrangien augmenté

L'algorithme de résolution utilisant le lagrangien augmenté est proche dans son principe de la méthode d'*Uzawa* décrite au § 3.6. Il s'agit en fait d'une séquence de minimisation en *x* du lagrangien augmenté, alternée avec une mise à jour des variables duales λ, le coefficient de pénalité étant augmenté à chaque itération. L'algorithme ci dessous décrit la méthode :

0) Point de départ : $x^0, \lambda^0, r^0, k \leftarrow 0$
1) Calcul de : $\hat{L}(x^k, \lambda^k, r^k) = \underset{x \in R^n}{Min} \left\{ \hat{L}(x, \lambda^k, r^k) \right\}$
2) Mise à jour des variables duales : $\lambda^{k+1} \longrightarrow \lambda^k$
3) Si test d'arrêt vérifié : **FIN**
4) Sinon $r^{k+1} \longrightarrow r^k$ avec $r^{k+1} \geq r^k$, $k \leftarrow k+1$

   retour en 0)

La mise à jour des variables duales peut être effectuée de la manière suivante [58]:
Après l'étape de minimisation en *x* de $\hat{L}(x, \lambda^k, r^k)$ on a:

$$\nabla_x \hat{L}(x^k, \lambda^k, r^k) = \nabla f(x^k) + \sum_{j=1}^{m} \left[\left(\lambda_j{}^k + 2r.\Psi_j(x^k) \right) \nabla c_j(x^k) \right] \approx 0$$

L'idée la plus naturelle consiste à définir λ^{k+1} comme :

$$\lambda_j{}^{k+1} = \lambda_j{}^k + 2r.\Psi_j(x^k) \text{ pour } j = 1..m \qquad (23)$$

En fait cette mise à jour correspond à un pas de type "gradient" sur les variables duales car :

$$\nabla_\lambda \hat{L}(x^k, \lambda^k, r^k) = \left[\Psi_1(x^k), \ldots, \Psi_j(x^k), \ldots, \Psi_m(x^k) \right]^T$$
$$\Rightarrow \lambda^{k+1} = \lambda^k + 2r.\nabla_\lambda \hat{L}(x^k, \lambda^k, r^k)$$

La convergence de cette méthode vers un optimum global du problème d'optimisation est liée à l'existence d'un point col du lagrangien augmenté. Dans le cas convexe il existe bien un point col pour le lagrangien augmenté pour toute valeur positive de *r*. Dans le cas non convexe, l'existence d'un point col du lagrangien augmenté n'est assurée que lorsqu'il existe deux réels $r > 0$ et *q* quelconque, tel que quelque soit $y \in R^m$ on ait [60] :

$$\phi(y) \geq q - r.\|y\|^2 \tag{24}$$

Cette condition de "croissance quadratique" sur la fonction de perturbation $\phi(y)$ ne peut malheureusement pas être reliée à une caractérisation précise des fonctions objectifs et contraintes permettant d'obtenir la condition (24).

Rockafellar [60] montre que dans le cas convexe ou lorsque la condition (24) (dans le cas non convexe) est satisfaite, alors il existe une valeur suffisamment grande de *r* telle que le lagrangien augmenté admette un point col. Dans ces conditions la méthode converge vers un point satisfaisant les conditions de *Kuhn et Tucker* si :

1) Les fonctions objectifs et contraintes sont continues et différentiables.
2) Le problème admet un optimum de valeur finie.
3) Les hypothèses de qualification des fonctions contraintes sont satisfaites, ce qui revient à supposer que les gradients des fonctions contraintes actives à l'optimum sont linéairement indépendants.

En ce qui concerne la vitesse de convergence, *Minoux* [45] rapporte que la relation de mise à jour des variables duales (23) permet d'obtenir une vitesse de convergence linéaire des points λ^k vers λ^* avec un taux de convergence d'autant meilleur que *r* est grand. Cela provient du fait que la fonction duale augmentée est d'autant mieux conditionnée que *r* est grand. Par contre, c'est le lagrangien augmenté qui devient mal conditionné pour de grande valeur de *r*. L'utilisation d'une méthode quasi newtonienne pour minimiser le lagrangien augmenté est alors fortement recommandée.

4 Conclusions

Finalement en ce qui concerne les méthodes de résolution des problèmes sans fonction contrainte nous rappellerons les points fondamentaux suivants :

L'hypothèse de continuité de la fonction à minimiser est fondamentale, c'est elle qui assure l'existence d'une solution.

Dans le cadre des méthodes que nous avons présentées, la fonction à minimiser doit être au moins une fois dérivable.

La condition $f(x) \to +\infty$ lorsque $\|x\| \to +\infty$ doit être également remplie, c'est sous cette hypothèse minimale que l'on démontre la propriété de convergence globale de toutes les méthodes présentées.

Toutes ces méthodes de minimisation convergent vers un point stationnaire de la fonction à minimiser. On est donc certain d'obtenir un optimum global que sur des fonctions convexes. On peut cependant, dans certains cas particuliers, obtenir un minimum global sur quelques fonctions non convexes. Par exemple une méthode de gradients conjugués, ou quasi newtonienne donnera bien le minimum global (1,1) de la fonction :

$$f(x_1, x_2) = 10(x_1 - x_2)^2 + (x_1 - 1)^2 + 1$$

qui pourtant n'est pas convexe.

L'implémentation d'une méthode de minimisation performante nécessite sans aucun doute l'utilisation d'une formule de correction de type BFGS. La robustesse de sa propriété de convergence globale et son insensibilité aux imprécisions dans la recherche unidimensionnelle en font un outil privilégié.

A propos des méthodes destinées aux problèmes d'optimisation avec contraintes, nous préciserons que :

L'existence d'une solution est liée à la continuité de la fonction objectif à optimiser, et au fait que le domaine des solutions doit être compact et non vide.

L'hypothèse de qualification des fonctions contraintes est fondamentale et plus particulièrement l'indépendance linéaire des gradients des fonctions contraintes à l'optimum. C'est sous cette hypothèse que les conditions d'optimalité de *Kuhn et Tucker* permettent de statuer sur l'optimalité locale ou globale (dans le cas convexe) d'un point solution.

L'existence d'un point col pour le lagrangien augmenté permet d'obtenir un optimum global grâce à une méthode duale, dans certains cas de problèmes non convexes.

Les méthodes de résolution pour les problèmes d'optimisation avec fonctions contraintes, se divisent en deux catégories :

1) Les méthodes primales, engendrant une suite de points situés à l'intérieur du domaine des solutions. Elles ont l'avantage de fournir dans toutes les circonstances (arrêt des calculs, par exemple) un point appartenant au domaine des solutions, et l'inconvénient d'être difficiles à mettre au point et de nécessiter un point de départ situé dans le domaine des solutions. Leur propriété de convergence globale ne peut être établie sans hypothèse de convexité sur le domaine des solutions.

2) Les méthodes duales permettant de ramener le problème à une suite de problèmes sans contrainte. Elles ont l'avantage de permettre un point de départ quelconque et possèdent une propriété de convergence globale plus robuste que celle des méthodes primales. Par contre en cas d'interruption dans les calculs le point obtenu peut être situé à l'extérieur du domaine des solutions.

Les méthodes duales semblent particulièrement bien adaptées à la résolution des problèmes de conception optimale. En effet dans ce type de problème il sera souvent très difficile de déterminer un point de départ situé à l'intérieur du domaine des solutions. D'autre part la possibilité d'obtenir un optimum global dans certains cas de problèmes non convexes avec une méthode utilisant le lagrangien augmenté constitue également un gros avantage.

Chapitre 4

Analyse Monotone

1 Introduction

L'ensemble des méthodes de résolution décrites précédemment sont des méthodes itératives générant une suite de points convergeant vers une solution optimale du problème d'optimisation, ou plus exactement vers un point satisfaisant certaines conditions d'optimalité. A partir d'un point courant du processus, le point suivant est déterminé grâce aux "informations" localement disponibles : valeurs, gradients, et quelque fois hessien des fonctions objectif et contraintes. Souvent ces informations s'accumulent tout au long du calcul sous la forme de matrices mises à jour à chaque itération (cas des méthodes quasi newtoniennes). Ces méthodes de résolution étant par nature, les plus générales possibles, elles n'utilisent pas les spécificités de certain problème d'optimisation.

Nous avons vu au chapitre 1 que pour des problèmes d'optimisation de quelques variables la méthode *MOD* (Method for Optimal Design) de *Johnson* [34], s'appuyant sur un raisonnement graphique permettait d'obtenir l'ensemble des solutions optimales d'un problème.

Lorsque le problème d'optimisation comporte des fonctions monotones, l'analyse monotone permet également d'obtenir l'ensemble des solutions optimales. Ces deux méthodes de résolution reposent sur un principe commun : **la recherche des fonctions contraintes actives à l'optimum**.

2 Méthodes de résolution non itératives

Les problèmes d'optimisation issus de la modélisation d'un problème de conception admettent de façon quasi systématique une solution optimale située sur une frontière du domaine des solutions. En effet dans la plupart des cas, le choix du critère d'optimisation (masse, encombrement, coût, ...) et des variables de conception est tel, que ce sont les conditions fonctionnelles limites (résistance, encombrement géométrique, ...) qui bornent les variables de conception.

Les frontières du domaine des solutions, sont constituées de l'ensemble des points de l'espace des variables de conception pour lesquels les conditions fonctionnelles limites, ou fonctions contraintes, ont atteint leur valeur limite. Pour cet ensemble de points les fonctions contraintes sont dites "actives" ou "saturées".

La connaissance des fonctions contraintes actives à l'optimum du problème simplifie considérablement le calcul de la solution. En effet on se retrouve alors, dans l'un des deux cas de figure suivants.

2.1 Restriction totale par les fonctions contraintes

Dans cette situation, le nombre de contraintes actives est égal au nombre de variables. On peut espérer trouver la solution grâce à la résolution du système non linéaire formé par les fonctions contraintes actives. Dans le cas d'un problème d'optimisation de deux variables, on voit sur la figure 4.27 que le point *A* est solution du système :

$$\begin{cases} c_1(x_1, x_2) = 0 \\ c_2(x_1, x_2) = 0 \end{cases}$$

Dans de nombreux cas de problèmes de conception ce calcul est possible sans avoir recours à des méthodes numériques, cela dépend de la complexité analytique des fonctions contraintes. Lorsque ce point solution peut être déterminé, la valeur des multiplicateurs de *Kuhn et Tucker* peut être calculée très simplement. Il suffira de résoudre le système linéaire formé par les gradients des contraintes actives pour statuer sur l'optimalité du point calculé. (Cf. chap. 3 §3.1.3).

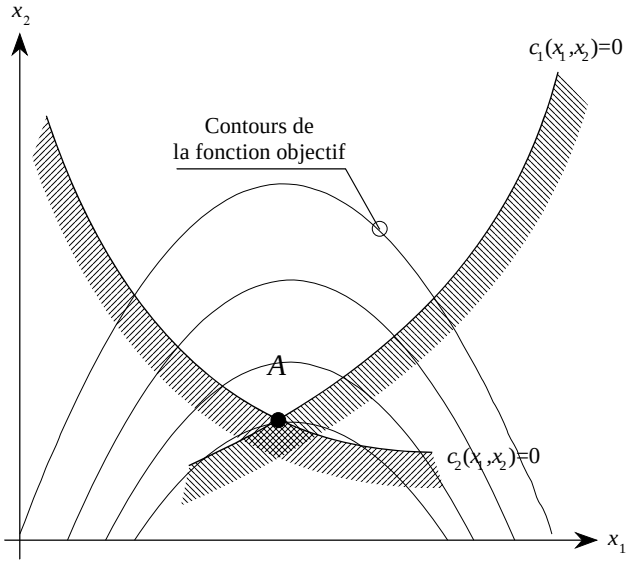


Figure 4.27 : Restriction totale par les contraintes.

2.2 Restriction partielle par les fonctions contraintes

Lorsque le nombre de fonctions contraintes actives est inférieur au nombre de variables, il subsiste un certain "degré de liberté" égal à la différence entre le nombre de variables et le nombre fonctions contraintes actives.

On s'aperçoit sur la figure 4.1 que le point solution est déterminé par le point de tangence du contour de la fonction objectif et de la fonction contrainte active. En ce point on a la relation :

$$\lambda \nabla c_1 = \nabla f$$

Plus généralement le point solution est déterminé grâce à la résolution du système :

$$\begin{cases} \nabla f(x^*) + \sum_{j=1}^{m} \lambda_j \nabla c_j(x^*) = 0 \\ c_j(x^*) = 0 \, j \in J(A) \end{cases}$$

où $J(A)$ est l'ensemble des indices des fonctions contraintes actives au point *A*.

On remarquera que cela conduit à la résolution des équations données par les conditions d'optimalité de *Kuhn et Tucker* dans le cas des fonctions contraintes égalités. Les réels λ_j, $j \in J(A)$, inconnues supplémentaires dans ce système d'équations sont ici les multiplicateurs de *Kuhn et Tucker*.

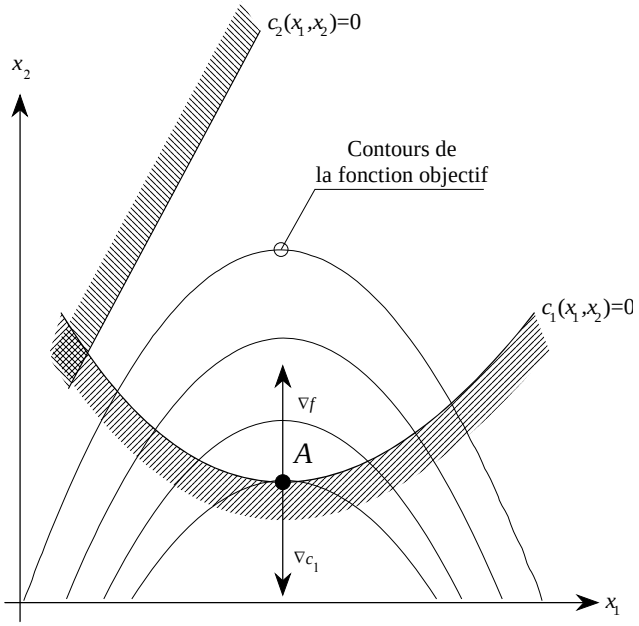


Figure 4.1 : Restriction partielle par les fonctions contraintes.

2.3 Identification des fonctions contraintes actives à l'optimum

Les problèmes de conception optimale comportent généralement un grand nombre de fonctions contraintes, souvent très supérieur au nombre de variables du problème.

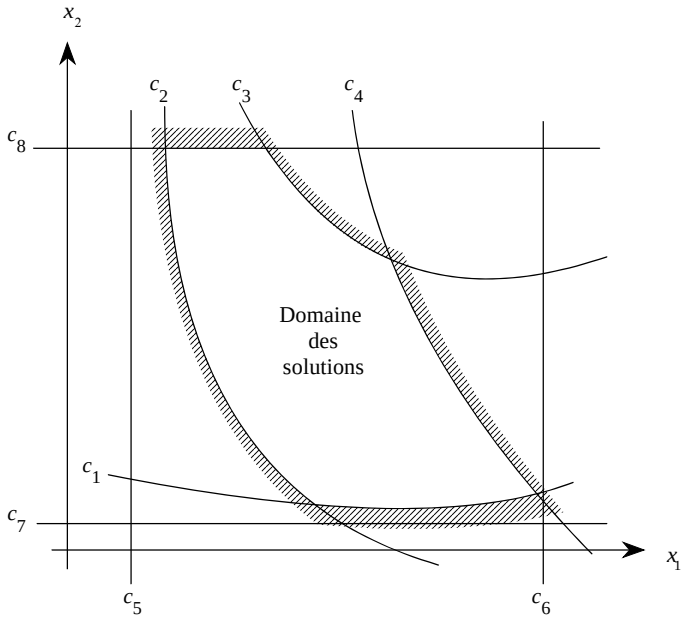


Figure 4.2 : Frontières du domaine des solutions.

La figure 4.2 présente le graphe d'un problème de deux variables où l'on a représenté l'ensemble des limites définies par les fonctions contraintes. On constate que toutes les fonctions contraintes ne participent pas à la définition du domaine des solutions, et que les frontières du domaine dépendent des données du problème. En effet, pour un ensemble de données différentes

les limites définies par les fonctions contraintes sont modifiées, leur position relative varie, et cela entraîne un changement de "morphologie" du domaine des solutions.

L'étude de ces variations de position permet d'établir le diagramme de variation de la méthode *MOD*. Lorsque le problème comporte trop de variables on ne peut plus s'appuyer sur ce genre de méthode graphique pour identifier les contraintes actives à l'optimum.

En fait la détermination de ces contraintes actives à l'optimum consiste à chercher la combinaison de *p* contraintes actives vérifiant les conditions d'optimalité de *Kuhn et Tucker*. Une démarche exhaustive serait d'examiner l'ensemble des combinaisons de *p*, pour *p* variant de 1 à *n*, fonctions contraintes actives parmi les fonctions contraintes définissant la frontière du domaine. Comme dans le cas général il est impossible de connaître à l'avance les frontières du domaine des solutions, on sera obligé de tenir compte de toutes les contraintes du problème. Dans ces conditions le nombre de combinaisons de contraintes devient considérable. En effet, si *n* est le nombre de variables et *m* le nombre de fonctions contraintes, le nombre total de combinaisons s'écrit :

$$N_c = \sum_{p=1}^{n} C_m^p = \sum_{p=1}^{n} \left(\frac{m!}{p!(m-p)!} \right)$$

La résolution du système non linéaire formé par les conditions de *Kuhn et Tucker* pour l'ensemble de ces N_c combinaisons conduirait à un nombre de calcul croissant exponentiellement en fonction de *n* et de *m*.

Nous allons voir comment grâce à l'étude des sens de variation des fonctions objectif et contraintes, l'analyse monotone permet d'obtenir rapidement la solution optimale du problème, en éliminant sans calcul, parmi ces N_c combinaisons possibles celles qui ne permettront pas d'obtenir la solution optimale.

3 Analyse monotone : principe, concepts généraux

3.1 Fonctions monotones, strictement monotones

La fonction $f(x)$, avec $f : x \in R^n \longrightarrow f(x) \in R$, continue et différentiable est monotone sur R^n par rapport aux n variables $x_1 \ldots x_i \ldots x_n$ du vecteur variable x si toutes ses dérivées partielles, $\frac{\partial f}{\partial x_i}(x)$, sont de signe constant pour tout $x \in R^n$. Elle est strictement monotone si toutes ses dérivées partielles sont en plus, non nulles pour tout $x \in R^n$.

3.2 Problèmes d'optimisation avec fonctions strictement monotones

Reprenons la formulation du problème introduite précédemment, soit :

$$(P_c) \begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1 \ldots m \end{cases}$$

Avec les hypothèses suivantes :

1) Les fonctions contraintes sont continues et différentiables
2) Le domaine des solutions défini par : $D = \left\{ x \in R^n / c_j(x) \le 0 \, j = 1 \ldots m \right\}$ est fermé, borné et non vide.
3) La fonction objectif est strictement monotone.

On montre très facilement que le problème (P_C) admet une solution optimale globale, nécessairement située sur la frontière de *D*. L'exemple ci-dessous montre que la solution de ce type de problème n'est pas toujours unique.

$$\text{Minimiser } f(x_1, x_2) = x_1 + x_2$$
$$\text{Sous les fonctions contraintes :}$$
$$c_1(x_1, x_2) = 2 - x_1 - x_2$$
$$c_2(x_1, x_2) = x_2 - 2x_1^{\,2}$$
$$c_3(x_1, x_2) = 2 - \frac{1}{x_1} - x_2$$

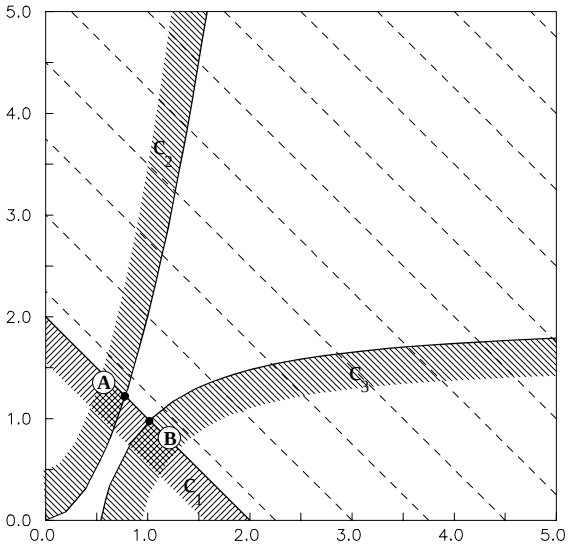


Figure 4.3 : Exemple de problème strictement monotone n'admettant pas de solution unique.

Tous les points du segment [*A*,*B*] sont solution du problème ci-dessus, on vérifiera simplement qu'en tout point du segment [*A*,*B*] les conditions d'optimalité de *Kuhn et Tucker* sont satisfaites. Ce cas se produit lorsque la frontière contenant les solutions optimales est tangente au contour de la fonction objectif.

L'ajout d'une hypothèse de stricte monotonie sur les fonctions contraintes du problème, et les conditions d'optimalité de *Kuhn et Tucker* permettent de mettre en place les deux théorèmes fondamentaux de l'analyse monotone.

3.3 Principe de l'analyse monotone

Soit $\overline{x}$ un point situé sur la frontière du domaine des solutions, donc un point potentiellement solution optimale du problème. Supposons que le domaine des contraintes satisfasse l'hypothèse de qualification des contraintes, ce qui revient à supposer ici que les gradients des contraintes sont linéairement indépendants.

En $\overline{x}$ les conditions de *Kuhn et Tucker* s'écrivent :

$$\begin{cases} \nabla f(\overline{x}) + \sum_{j=1}^{m} \lambda_j \nabla c_j(\overline{x}) = 0 \\ \lambda_j c_j(\overline{x}) = 0, \lambda_j \geq 0 \; j = 1..m \end{cases} \quad (1)$$

et plus particulièrement par rapport à $\overline{x_i}$ une composante de $\overline{x}$ on obtient :

$$\begin{cases} \frac{\partial f}{\partial x_i}(\overline{x}) + \sum_{j=1}^{m} \lambda_j \frac{\partial c_j}{\partial x_i}(\overline{x}) = 0 \\ \lambda_j c_j(\overline{x}) = 0, \lambda_j \geq 0 \; j = 1..m \end{cases}$$

Si on énonce les conditions nécessaires conduisant à la vérification des relations (1), donc permettant de statuer sur l'optimalité de $\overline{x}$ on obtient :

(1) Si $\frac{\partial f}{\partial x_i}(\overline{x}) \neq 0$, puisque $\lambda_j \geq 0$ pour $j = 1..m$, il doit nécessairement exister au moins un λ_j non nul, donc une fonction contrainte active telle que $\frac{\partial c_j}{\partial x_i}(\overline{x})$ et $\frac{\partial f}{\partial x_i}(\overline{x})$ soient de signe opposé.

(2) Si $\frac{\partial f}{\partial x_i}(\overline{x}) = 0$, puisque $\lambda_j \geq 0$ pour $j = 1..m$, il faut qu'il existe au moins λ_{j1} et λ_{j2} non nuls, donc deux fonctions contraintes actives telles que $\frac{\partial c_{j1}}{\partial x_i}(\overline{x})$ et $\frac{\partial c_{j2}}{\partial x_i}(\overline{x})$ soient de signe opposé, ou alors il faut que toutes les fonctions contraintes telles que $\frac{\partial c_j}{\partial x_i}(\overline{x}) \neq 0$ soient inactives ($\lambda_j = 0$).

Ces deux conditions sont résumées dans les théorèmes fondamentaux de l'analyse monotone, énoncés par *Wilde* [69].

Théorème n°1:

Si la variable x_i est explicitement représentée dans la fonction objectif à minimiser, alors il existe au moins une contrainte active avec un sens de variation opposé par rapport à x_i.

Théorème n°2:

Une variable x_i qui n'est pas explicitement représentée dans la fonction objectif doit être seulement contenue dans les contraintes inactives, ou alors il doit exister au moins deux contraintes actives ayant un sens de variation opposé par rapport à x_i.

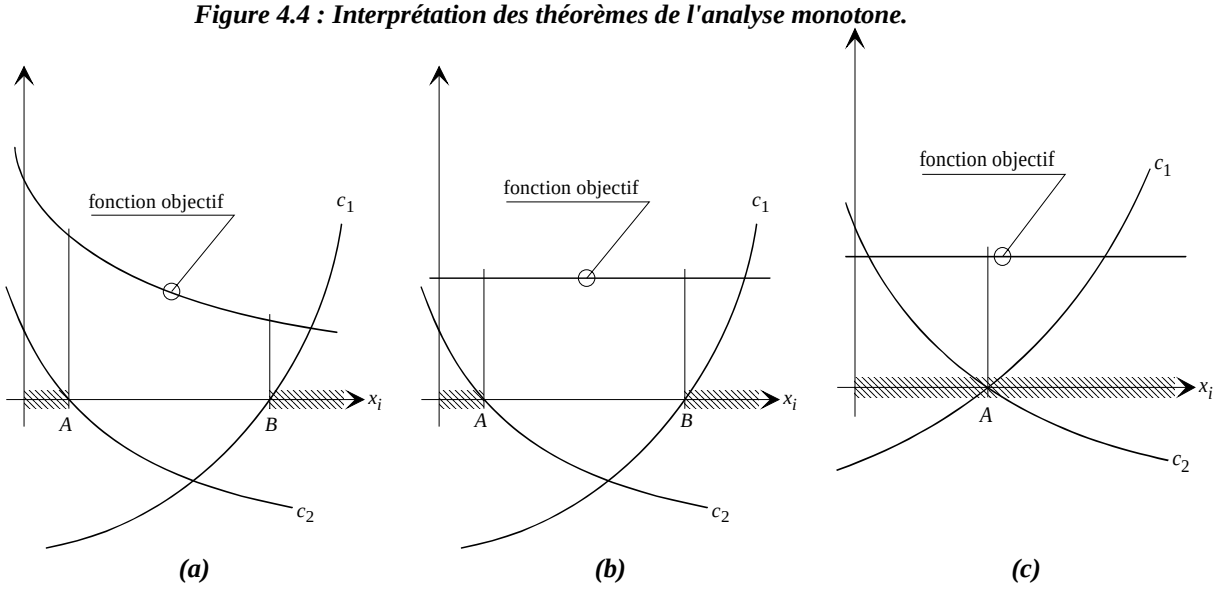
On ajoutera également le résultat suivant, issu du chapitre précédent (§3.1.3) :

Théorème n°3:

Le nombre de contraintes actives non redondantes ne peut pas être supérieur au nombre de variables.

Les deux premiers théorèmes de l'analyse monotone expriment un concept très simple : Le fait qu'à l'optimum d'un problème monotone les variables sont bornées par une fonction (objectif ou contrainte) croissante et une autre décroissante par rapport aux variables du problème.

Figure 4.4 : Interprétation des théorèmes de l'analyse monotone.



(a) ***(b)*** ***(c)***

On a représenté sur la figure 4.4 les fonctions objectif et contraintes d'un problème monotone. Les graphes sont établis par rapport à une variable x_i du problème les autres étant supposées fixes. La figure 4.4a montre que la valeur optimale de x_i est en *B* et que la contrainte c_1 est active (théorème n°1). Dans le cas des figures 4.4b et 4.4c la fonction objectif est indépendante de la variable x_i. Sur la figure 4.4b toutes les valeurs du segment [*A*,*B*] conviennent, la valeur optimale de x_i sera déterminée en fonction des autres variables du problème et les contraintes c_1 et c_2 sont inactives (théorème n°2). Sur la figure 4.4c les fonctions contraintes sont telles, que la valeur optimale de x_i est fixée en *A* et dans ce cas les contraintes c_1 et c_2 sont actives (théorème n°2).

4 Utilisation de l'analyse monotone

En regroupant dans un tableau les sens de variation de la fonctions objectif et des fonctions contraintes du problème on obtient la "table de monotonie" du problème [69]. Cette représentation permet de détecter rapidement les contraintes actives à l'optimum.

Considérons la table de monotonie suivante :

	x_1	x_2	x_3	x_4	...	x_n
$f(x)$	+		+	-		
$c_1(x)$	-	+				
$c_2(x)$	+	-				
$c_3(x)$			-	+		
$c_4(x)$	-			+		
⋮						
$c_m(x)$						

Les signes "+" et "-" représentent respectivement les fonctions croissantes et décroissantes par rapport aux variables du problème. Dans le cas des fonctions indépendantes de certaines variables les cases sont vides.

4.1 Résolution du problème par élimination de variables

Si on applique le théorème n°1 par rapport à la variable x_3 de la table de monotonie ci-dessus, on en déduit que la contrainte c_3 doit être active à l'optimum du problème. Cette contrainte active doit alors permettre l'élimination d'une variable du problème (du moins théoriquement). On notera que la variable éliminée n'est pas nécessairement x_3. En appliquant le théorème n°2 par rapport à x_2 on en déduit que les contraintes c_1 et c_2 sont soit simultanément actives soit simultanément inactives. On peut alors formuler deux "sous problèmes". L'un dans lequel on a éliminé trois variables et l'autre dans lequel on a éliminé une seule variable. Avec les tables de monotonie associées à ces deux problèmes on peut recommencer l'analyse précédente jusqu'à élimination de toutes les variables du problème. Cette démarche permet de trouver rapidement la solution dans de nombreux cas de problèmes monotones, comme en témoignent les exemples traités dans les références [47], [48], [49], [50], [68], [70] et [44].

4.2 Démarche générale de résolution

La méthode de résolution s'appuyant sur les théorèmes de l'analyse monotone que nous proposons généralise celle utilisée ci-dessus. Plutôt que de procéder par élimination successive des variables du problème on se propose d'utiliser l'approche combinatoire décrite dans le paragraphe 2.3 de ce chapitre.

En effet pour une combinaison de contraintes actives donnée, si les variables ne sont pas bornées par des fonctions croissantes et décroissantes, les théorèmes 1 et 2 ne sont pas vérifiés. Cette combinaison de contraintes actives ne pourra pas vérifier les conditions d'optimalité de *Kuhn et Tucker*, plus précisément les λ_j associés aux contraintes actives ne seront pas tous positifs. Elle sera donc éliminée de l'ensemble des combinaisons susceptibles de donner une solution optimale. Pour les combinaisons restantes, le signe des multiplicateurs de *Kuhn et Tucker* ne pourra pas être déduit simplement. Il faudra alors résoudre le système d'équations formé par les conditions d'optimalité de *Kuhn et Tucker*. On retrouve ici les deux situations décrites dans le § 2.3 : les cas de restriction partielle ou totale par les fonctions contraintes actives.

La recherche de la solution optimale d'un problème strictement monotone se déroulera donc de la manière suivante :

1) Identification des sens de variation des fonctions objectif et contraintes
2) Calcul du nombre de combinaisons de fonctions contraintes actives.
$$N_c = \sum_{p=1}^{n} C_m^p = \sum_{p=1}^{n} \left(\frac{m!}{p!(m-p)!} \right)$$
3) Elimination combinaisons ne remplissant pas les conditions des théorèmes 1 ou 2.
4) Pour les combinaisons restantes : résolution des systèmes non linéaires formé par les fonctions contraintes actives.
5) Pour chaque solution calculée vérifier :
   la positivité des multiplicateurs de *Kuhn et Tucker*
   l'appartenance au domaine des solutions
6) Si ces deux conditions sont réunies alors la solution calculée est solution optimale.

L'avantage de cette démarche est de permettre d'évaluer rapidement les performances de l'analyse monotone sur un problème donné. En effet on obtient sans aucun calcul le nombre de combinaisons qui pourront être éliminées. Les calculs ultérieurs seront d'autant plus faciles que ce nombre sera élevé.

Cette étape d'élimination des combinaisons non optimales est longue et source d'erreurs lorsqu'elle est effectuée manuellement. Nous avons conçu un programme qui effectue automatiquement cette tâche (étapes 1,2 et 3 de la démarche).

A partir des équations du problème et d'un point d'évaluation fourni par l'utilisateur le programme établi la table de monotonie du problème. Les sens de variation des fonctions objectif et contraintes sont donnés par les dérivées partielles calculées par le programme grâce à une méthode de différences finies. A partir de cette table de monotonie le programme dresse la liste lexicographique de toutes les combinaisons de contraintes actives et élimine toutes celles qui ne satisfont pas les théorèmes 1 ou 2. En sortie le programme fourni le nombre et la liste des combinaisons restantes.

Les deux exemples qui vont suivre montrent que sur certains types de problèmes l'analyse monotone permet de trouver rapidement la solution optimale.

4.3 Exemples d'application

4.3.1 Ressort de pompe hydraulique

Considérons la pompe hydraulique à pistons axiaux représentée sur la figure 4.5. Au cours du fonctionnement, les ressorts de rappel des pistons sont soumis à un effort de compression variable dans le temps. La longévité du ressort est alors caractérisée par la limite d'endurance du matériau du ressort. Le problème du dimensionnement optimal de ce ressort de rappel se pose en ces termes :

"Rechercher le nombre de spires N, leur diamètre d'enroulement moyen D, et le diamètre d du fil constituant le ressort qui maximisent le coefficient de sécurité en fatigue".

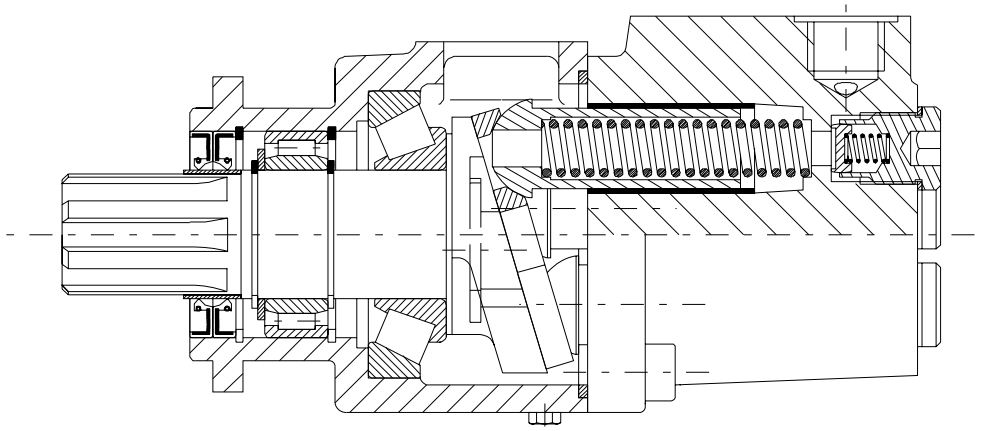


Figure 4.5 : Pompe hydraulique à pistons axiaux.

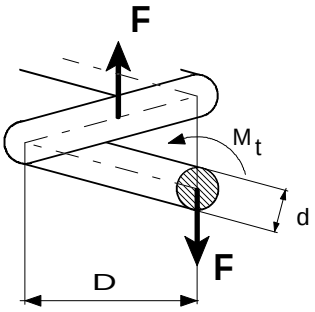
4.3.1.1 Expression du problème d'optimisation

En assimilant le fil du ressort à une poutre de section circulaire soumise à un moment de torsion $M_t = D.F/2$, et en supposant que le travail fourni par ce ressort est égal à l'énergie de déformation de cette poutre, on détermine la raideur *k* du ressort (figure 4.7a).

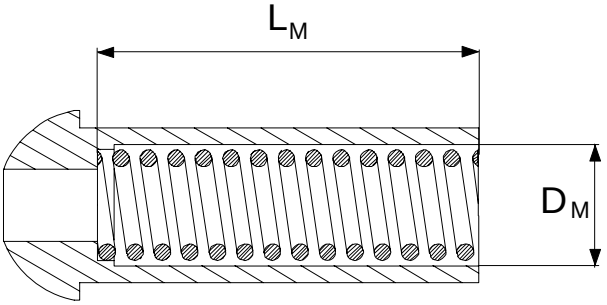
$$k = \frac{G.d^4}{8D^3.N}$$

Avec *G*: Module d'élasticité de cisaillement du matériau du ressort.

Figure 4.7 : Expression du problème d'optimisation.



(a) : Raideur du ressort.



(b) : Encombrement géométrique.

Si on note F_{MAXI} et F_{MINI} les valeurs extrêmes de l'effort de compression au cours du fonctionnement de la pompe, on obtient la contrainte de cisaillement maximale (τ_{MAXI}) et alternée (τ_A) dans le ressort.

$$\tau_{MAXI} = K_t \frac{8D}{\pi d^3} F_{MAXI}$$

$$\tau_A = K_t \frac{4D}{\pi d^3}\left(F_{MAXI} - F_{MINI}\right) = K_t \frac{4D}{\pi d^3} kC$$

$$K_t = 1.6\left(\frac{D}{d}\right)^{-0.14}$$

Avec K_t: Coefficient de concentration de contrainte, fonction de la forme du ressort.

C: Course du ressort pour $\Delta F = F_{MAXI} - F_{MINI}$

On ajoutera aux équations fonctionnelles précédentes les équations définissant :

La longueur du ressort pour $F = F_{MAXI}$: $L = Nd\left(1 + \varepsilon\right)$

Avec ε: Jeu minimal entre les spires du ressort.

La masse du ressort : $m = \frac{\pi^2}{4} \rho d^2 DN$

Avec ρ: Masse volumique du matériau du ressort

La première fréquence propre de vibration : $f_1 = \frac{1}{2}\sqrt{\frac{k}{m}}$

Le diamètre d'encombrement : $D_e = D + d$

Les conditions fonctionnelles limites pour ce problème donnent les relations suivantes :

Résistance statique limite : $\tau_{MAXI} \leq \tau_L$

Avec τ_L : Résistance statique limite, fonction du diamètre du fil du ressort *d*.

Contrainte alternée inférieure à la limite de fatigue : $\tau_A \leq \tau_D$

Avec τ_D : Limite de fatigue du matériau du ressort.

Condition de fabrication : $4 \leq \frac{D}{d} \leq 16$

Sur la première fréquence propre : $f_1 \geq f_{MAXI}$

Avec f_{MAXI} : = deux fois la fréquence de rotation de la pompe.

Sur le diamètre d'encombrement maximal : $D_e \leq D_M$

Sur l'encombrement en longueur du ressort : $L \leq L_M$

Avec D_M, L_M : Dimension maximale du piston (Cf figure 4.7b)

Sur la raideur du ressort : $k_m \leq k \leq k_M$

Avec k_m, k_M : Borne inférieure et supérieure du domaine des raideurs

Enfin le coefficient de sécurité en fatigue, donc la fonction à maximiser, s'écrit :

$$\alpha_F = \frac{\tau_D}{\tau_A}$$

Les 12 paramètres suivants seront considérés comme des données du problème d'optimisation.

$$C, F_{MAXI}, k_m, k_M, D_M, L_M, f_{MAXI}, \tau_D, \tau_L, \rho, G, \varepsilon$$

Il reste alors 10 paramètres pouvant être choisis comme variables du problème. Les 7 équations fonctionnelles de définition permettent d'éliminer simplement les 7 paramètres suivants :

$$L, k, m, f_1, D_e, \tau_{MAXI}, \tau_A$$

Donc ce problème s'exprimera en fonction des variables suivantes :

$$d, N, D$$

Le diamètre du fil du ressort est une variable discrète dont dépend la valeur de τ_L. Pour traiter ce problème par l'analyse monotone nous serons obligés d'adopter une démarche similaire à celle de l'accouplement à plateaux. Le diamètre du fil sera donc considéré comme une donnée du problème et l'on envisagera successivement tous les diamètres normalisés de fil jusqu'à l'obtention de l'optimum.

Finalement on obtient l'expression suivante pour ce problème d'optimisation :

Minimiser la fonction objectif : $$F(D,N) = -\alpha_F = -\left(\frac{\pi\tau_D}{0.8GCd^{1.14}}\right)D^{2.14}N$$

Sous les fonctions contraintes :

$$c_1(D,N) = \left(\frac{0.8GCd^{1.14}}{\pi\tau_D}\right) - ND^{2.14} \le 0$$

$$c_2(D,N) = D^3N - \frac{Gd^4}{8k_m} \le 0$$

$$c_3(D,N) = \frac{Gd^4}{8k_M} - D^3N \le 0$$

$$c_4(D,N) = \frac{d}{2\pi f_{MAXI}}\sqrt{\frac{G}{2\rho}} - D^2N \le 0$$

$$c_5(D,N) = N - \frac{L_M}{d(1+\varepsilon)} \le 0$$

$$c_6(D,N) = D - D_{maxi} \le 0$$

$$c_7(D,N) = D_{mini} - D \le 0$$

Vecteur variable : $$x = \{D, N\}^T$$

Données : $$C, F_{MAXI}, k_m, k_M, D_M, L_M, f_{MAXI}, \tau_D, \tau_L, \rho, G, \varepsilon, d$$

Avec :

$$D_{mini} = 4d \text{ et } D_{\max i} = Max\left\{Min\left\{16d, D_M - d, \left(\frac{\pi\tau_L d^{2.86}}{12.8F_{MAXI}}\right)^{1/0.86}\right\}; D_{\min i}\right\}$$

4.3.1.2 Résolution par l'analyse monotone

Les dérivées partielles des fonctions objectif et contraintes de ce problèmes sont toutes de signe constant pour $D > 0$ et $N > 0$. Notre programme d'analyse d'un problème d'optimisation monotone donne la table de monotonie suivante :

	F	c_1	c_2	c_3	c_4	c_5	c_6	c_7
D	-	-	+	-	-		+	-
N	-	-	+	-	-	+		

Table de monotonie pour le problème du ressort de pompe hydraulique.

On obtient également le tableau ci-dessous regroupant la liste des 28 combinaisons de contraintes actives possibles. Les cases ombrées contiennent les combinaisons retenues par le programme d'analyse.

c_1	c_1,c_2	c_2,c_3	c_3,c_4	c_4,c_5	c_5,c_6	c_6,c_7
c_2	c_1,c_3	c_2,c_4	c_3,c_5	c_4,c_6	c_5,c_7	
c_3	c_1,c_4	c_2,c_5	c_3,c_6	c_4,c_7		
c_4	c_1,c_5	c_2,c_6	c_3,c_7			
c_5	c_1,c_6	c_2,c_7				
c_6	c_1,c_7					
c_7						

Liste des combinaisons de contraintes valides.

Conformément à la démarche du §3.3 il faut maintenant résoudre chaque système formé par les différentes combinaisons de contraintes.

Par exemple dans le cas où la contrainte c_2 est active les équations de *Kuhn et Tucker* donnent :

$$\begin{cases} \left(\frac{\pi\tau_D}{0.8GCd^{1.14}}\right)2.14D^{2.14}N - 3D^2N\lambda_2 = 0 \\ \left(\frac{\pi\tau_D}{0.8GCd^{1.14}}\right)D^{2.14} - D^3\lambda_2 = 0 \\ D^3N - \frac{Gd^4}{8k_m} = 0 \end{cases} \Rightarrow \begin{cases} N = \frac{Gd^4}{8k_m(3/2.14)^3} \\ D = 3/2.14 \approx 1.40186 \\ \lambda_2 = \left(\frac{\pi\tau_D}{0.8GCd^{1.14}}\right)(3/2.14)^{-0.86} \end{cases}$$

Le multiplicateur λ_2 associé à la contrainte active de cette combinaison est positif, si le point calculé appartient au domaine des solutions il sera solution optimale du problème.

Lorsqu'on résout le cas où les fonctions contraintes c_1, c_2 sont actives on obtient :

$$\begin{cases} \left(\frac{\pi\tau_D}{0.8GCd^{1.14}}\right)2.14D^{2.14}N - 2.14ND^{1.14}\lambda_1 + 3D^2N\lambda_2 = 0 \\ \left(\frac{\pi\tau_D}{0.8GCd^{1.14}}\right)D^{2.14} - D^{2.14}\lambda_1 + D^3\lambda_2 = 0 \\ \left(\frac{0.8GCd^{1.14}}{\pi\tau_D}\right) - ND^{2.14} = 0 \\ D^3N - \frac{Gd^4}{8k_m} = 0 \end{cases} \Rightarrow \begin{cases} D = \left[\frac{\pi\tau_D d^{2.86}}{6.4Ck_m}\right]^{1/0.86} \\ N = \frac{Gd^4}{8k_mD^3} \\ \lambda_1 = \left(\frac{-\pi\tau_D}{0.8GCd^{1.14}}\right) \\ \lambda_2 = 0 \end{cases}$$

Le multiplicateur λ_1 est toujours négatif (quelque soit les données du problème), donc le point calculé avec cette combinaison de contrainte ne sera **jamais solution optimale** de ce problème.

En répétant cette démarche pour les 6 autres combinaisons on obtient au total seulement 4 solutions dont les multiplicateurs sont toujours positifs. Finalement l'ensemble des solutions optimales de ce problème est :

Contraintes actives	D	N
c_2	$3/2.14$	$\frac{Gd^4}{8k_m(3/2.14)^3}$
c_2, c_5	$\left[\frac{Gd^5(1+\varepsilon)}{8k_mL_M}\right]^{1/3}$	$\frac{L_M}{d(1+\varepsilon)}$
c_2, c_7	D_{mini}	$\frac{Gd^4}{8k_mD_{mini}^3}$
c_5, c_6	D_{maxi}	$\frac{L_M}{d(1+\varepsilon)}$

Tableau des solutions pour le problème du ressort de pompe hydraulique

Les multiplicateurs calculés pour ces trois solutions étant toujours positifs, le choix de la solution optimale se fera simplement sur l'appartenance au domaine des solutions du problème. On vérifiera alors que toutes les fonctions contraintes sont satisfaites.

Remarque :

La condition minimale pour que le point solution obtenu lorsque la contrainte c_2 est active appartienne au domaine des solutions sera d'avoir : $D_{mini} = 4d \leq 3/2.14$ (respect de la contrainte c_7). Avec les données que nous utiliserons cette condition n'est pas remplie et donc ce point ne sera jamais solution du problème.

La recherche de la solution optimale sur un ensemble de diamètres de fils *d* se résumera à la détermination de la solution optimale pour une valeur fixée de *d* puis au choix de *d* offrant le meilleur coefficient de sécurité en fatigue.

- **Application Numérique :**

Nous utiliserons les données suivantes :

Sur le problème :

C = 15 mm	D_M = 15 mm	L_M = 62 mm	ε = 0.1 mm
F_{MAXI} = 300 N	k_m = 4 N/mm	k_M = 10 N/mm	f_{MAXI} = 100 Hz
τ_D = 300 Mpa	G = 80 000 Mpa	$\rho = 7.8\ 10^{-6}$ Kg/mm^3	

Sur le fil du ressort (XC85) :

d (mm)	1	1.5	2	2.5	3	3.5	4	4.5	5
τ_L (N/mm^2)	1220	1150	1080	1030	980	920	890	850	810

Les calculs donnent le graphe suivant :

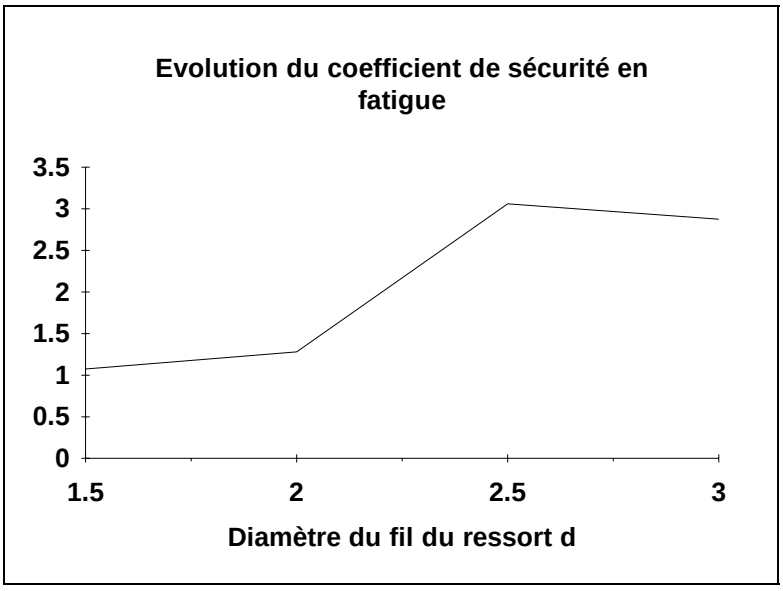


Figure 4.8 : Résultat de l'optimisation du coefficient de sécurité en fatigue.

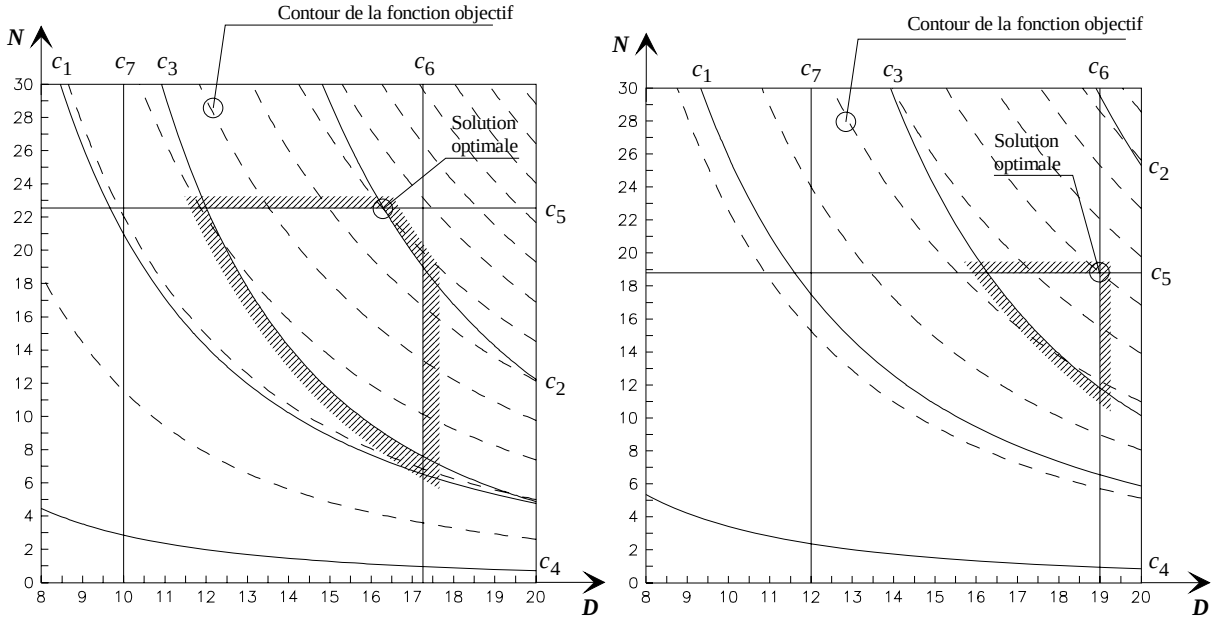
Ce coefficient atteint sa valeur maximale pour un diamètre de fil de 2.5 mm. Pour les valeurs de diamètre inférieures à 1 mm et supérieures à 3 mm, le problème d'optimisation n'a pas

de solution avec ce jeu de données. En effet pour ces diamètres le domaine des solutions est vide, et aucune des solutions calculées ne respecte l'ensemble des fonctions contraintes du problème. On peut vérifier qu'en augmentant la valeur de τ_D (de 300 à 450 Mpa) et le domaine des raideurs de ressort admissibles (k_M passe de 10 à 70 N/mm) on obtient une solution pour l'ensemble des diamètres. Avec ces nouvelles données le coefficient de sécurité en fatigue reste toujours maximal pour d = 2.5 mm.

Le tableau ci-dessous donne la valeur optimale des variables du problème pour chaque diamètre de fil du ressort. Ces résultats montrent que l'optimum pour un diamètre de fil donné n'est pas toujours atteint pour les mêmes contraintes actives.

d	D	N	α_F	Solution
1.50	6.00	37.58	1.07	c_5,c_6
2.00	8.68	28.18	1.28	c_5,c_6
2.50	16.30	22.55	3.06	c_2,c_5
3.00	19.00	18.79	2.87	c_5,c_6

Figure 4.9 : Domaine des solutions du problème du ressort.



(a) : d=2.5 mm ***(b) : d=3 mm***

La figure 4.9 représente le domaine des solutions du problème pour deux valeurs de *d*. Cette figure montre clairement que le domaine des solutions d'un problème de conception optimale peut être modifié de façon importante en fonction des données du problème. On constate également, et cette conclusion est tout à fait importante, que le domaine passe d'un domaine **convexe** à un domaine **non convexe**. Une méthode de programmation mathématique

nécessitant la convexité du domaine des solutions devra donc toujours être appliquée avec prudence sur un problème de conception optimale.

4.3.2 Accouplement à plateaux

Une méthode de recherche de la solution optimale s'appuyant sur les principes de l'analyse monotone permet de résoudre rapidement certain type de problème comme nous l'avons vu sur l'exemple du ressort de pompe hydraulique. L'analyse monotone permet également de déceler les modélisations imprécises d'un problème de conception.

Reprenons le problème de l'accouplement à plateaux énoncé au chapitre 2. Pour pouvoir résoudre ce problème par l'analyse monotone nous supposerons que le diamètre des boulons *d* est pris comme une donnée du problème. On obtient alors un problème de 3 variables et de 5 fonctions contraintes qui s'exprime :

Minimiser la fonction objectif : $F(N, R_B, M) = R = R_B + \phi_4(d) + c$

Sous les fonctions contraintes :

$$c_1(N, R_B, M) = M_T - M \le 0$$

$$c_2(N, R_B, M) = \frac{\alpha . M}{N . R_B} - K(d) \le 0$$

$$c_3(N, R_B, M) = \phi_5(d) - \frac{2\pi R_B}{N} \le 0$$

$$c_4(N, R_B, M) = R_M + \phi_4(d) - R_B \le 0$$

$$c_5(N, R_B, M) = N_m - N \le 0$$

Vecteur variable : $x = \{N, R_B, M\}^T$

Données : $(M_T, f_m, f_1, \alpha, R_e, N_m, R_M, c, d_m, d)$

Avec

$$K(d) = \frac{0.9 f_m R_e \pi (\phi_1(d))^2}{4\sqrt{1 + 3\left(\frac{4(0.16\phi_3(d) + 0.583\phi_2(d) f_1)}{\phi_1(d)}\right)^2}}$$

Les fonctions objectif et contraintes de ce problème sont strictement monotones par rapport aux variables N, R_B, M. La table de monotonie s'écrit :

	F	c_1	c_2	c_3	c_4	c_5
N			-	+		-
R_B	+		-	-	-	
M		-	+			

Sur ces 25 combinaisons seulement 4 satisfont les théorèmes 1 et 2 :

c_1	c_1,c_2	c_2,c_3	c_3,c_4	c_4,c_5	c_1,c_2,c_3	c_2,c_3,c_4	c_3,c_4,c_5
c_2	c_1,c_3	c_2,c_4	c_3,c_5		c_1,c_2,c_4	c_2,c_3,c_5	
c_3	c_1,c_4	c_2,c_5			c_1,c_2,c_5	c_2,c_4,c_5	
c_4	c_1,c_5				c_1,c_3,c_4		
c_5					c_1,c_3,c_5		
					c_1,c_4,c_5		

Liste des combinaisons de contraintes valides.

Dans le cas où la contrainte c_4 est active les équations de *Kuhn et Tucker* donnent le système d'équations suivant :

$$\begin{cases} 0+0\lambda_4=0 \\ 1-\lambda_4=0 \\ 0+0\lambda_4=0 \\ R_M+\phi_4(d)-R_B=0 \end{cases} \Rightarrow \begin{cases} \lambda_4=1 \\ R_B=R_M+\phi_4(d) \end{cases}$$

Ces équations ne permettent pas de calculer la valeur de *N* et *M*. Sachant que le point solution correspondant à cette combinaison doit appartenir au domaine des solutions on peut écrire les relations suivantes :

$$\left.\begin{matrix} c_3(N,R_B,M)\le 0 \\ c_5(N,R_B,M)\le 0 \end{matrix}\right\} \Rightarrow N\in\left[N_m,\frac{2\pi\left(R_M+\phi_4(d)\right)}{\phi_5(d)}\right]$$

$$\left.\begin{matrix} c_1(N,R_B,M)\le 0 \\ c_2(N,R_B,M)\le 0 \end{matrix}\right\} \Rightarrow M\in\left[M_{T,}\frac{K(d)N_m\left(R_M+\phi_4(d)\right)}{\alpha}\right]$$

Sauf pour des cas particuliers de données pour lesquels les bornes de ces intervalles sont confondues, ces relations donnent une infinité de solutions. On peut comme dans la démarche exposée au chapitre 2 faire un choix et garder comme solution les valeurs minimales de *N* et *M*, ce choix correspondant à l'ajout de deux critères supplémentaires d'optimisation.

Une autre approche consiste à modifier l'expression de la fonction objectif de ce problème, et d'intégrer dans une formulation multicritères ces deux critères de choix supplémentaires. Dans le cadre de l'analyse monotone l'expression de la fonction objectif sous la forme des "fonctions d'utilités" semble être la mieux adaptée [28].

On adopte alors la fonction objectif multicritères suivante :

$$F(N,R_B,M)=\beta_1\left(\frac{N}{N_m}\right)+\beta_2\left(\frac{R_B+\phi_4(d)+c}{R_M}\right)+\beta_3\left(\frac{M}{M_T}\right)$$

La valeur des coefficients β_i, supposés non nuls, définit l'importance relative des trois critères d'optimisation. Plus cette valeur est grande, plus le critère correspondant est privilégié. Cette formulation conservant la monotonie de la fonction objectif par rapport aux variables du problème, la table de monotonie devient :

	F	c_1	c_2	c_3	c_4	c_5
N	+		-	+		-
R_B	+		-	-	-	
M	+	-	+			

Il y a toujours 25 combinaisons de contraintes actives possibles, mais dans ce cas on trouve 6 combinaisons vérifiant les théorèmes 1 et 2 de l'analyse monotone.

Les multiplicateurs de *Kuhn et Tucker* calculés pour ces 6 combinaisons de contraintes sont toujours positifs. On obtient donc le tableau de solutions suivant :

Contraintes actives	N	R_B	M	
c_1, c_2	$\sqrt{\frac{\beta_2 \alpha M_T N_m}{\beta_1 K(d) R_M}}$	$\sqrt{\frac{\beta_1 \alpha M_T R_M}{\beta_2 K(d) N_m}}$	M_T	G
c_1, c_2, c_3	$\sqrt{\frac{2\pi\alpha M_T}{K(d)\phi_5(d)}}$	$\sqrt{\frac{\alpha\phi_5(d) M_T}{2\pi K(d)}}$	M_T	A
c_1, c_2, c_4	$\frac{\alpha M_T}{K(d)\left(R_M + \phi_5(d)\right)}$	$R_M + \phi_5(d)$	M_T	B
c_1, c_2, c_5	N_m	$\frac{\alpha M_T}{K(d) N_m}$	M_T	F
c_1, c_3, c_5	N_m	$\frac{\phi_5(d) N_m}{2\pi}$	M_T	C
c_1, c_4, c_5	N_m	$R_M + \phi_5(d)$	M_T	D

Tableau des solutions optimales de l'accouplement à plateaux.

Pour un ensemble de données déterminé, la solution optimale sera celle qui satisfait l'ensemble des contraintes du problème. Les valeurs de *N* pour les trois premières solutions peuvent ne pas être entières. La procédure d'arrondi sera identique pour ces trois solutions.

On choisira parmi les valeurs entières les plus proches (Ent(N) et Ent(N)+1) celle qui donne la plus petite valeur de la fonction objectif et qui satisfait toutes les contraintes du problème.

Les 6 solutions optimales de ce problème sont placées sur le graphe de la figure 4.10. Les flèches indiquent la variation de la fonction objectif en fonction des coefficients β_i. L'ensemble de ces solutions contient les 4 solutions trouvées par les calculs du chapitre 2.

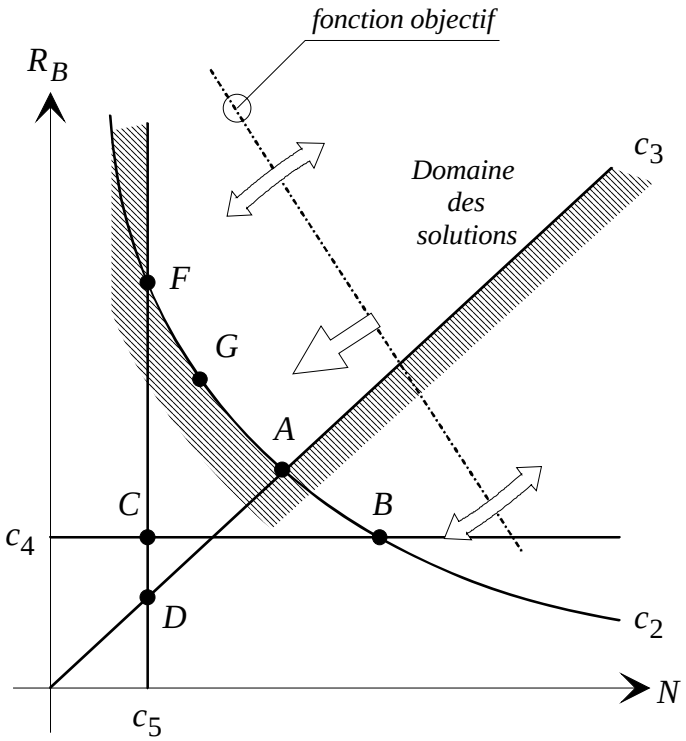


Figure 4.10 : Localisation des solutions optimales de l'accouplement à plateaux.

Avec les données définies dans le chapitre 2, et un algorithme de calcul similaire on obtient les solutions récapitulées ci-dessous, pour plusieurs valeurs des coefficients β_i.

β_1	10	1	1
β_2	1	1	10
β_3	1	1	1
R (mm)	93.98	89.00	86.00
d (mm)	16	16	14
N	8	9	13
R_B (mm)	71.98	67.00	65.50
M (m.N)	$4\ 10^3$	$4\ 10^3$	$4\ 10^3$
solution	*F*	*B*	*B*

On retrouve bien la solution retenue dans le chapitre 2 pour une grande valeur du coefficient associé au critère du rayon minimal.

On constate que même sur un exemple relativement simple il est difficile d'obtenir directement une modélisation précise du problème de conception. Cet exemple d'application montre bien que l'analyse développée lors de la résolution du problème par l'analyse monotone permet de dégager des critères de choix supplémentaires, et de les intégrer dans une formulation multicritères. Le choix des coefficients "d'influences" sur les trois critères de minimisation donne un ensemble de solutions optimales plus important, laissant au concepteur plus de latitude dans le choix d'une solution adéquate.

4.4 Intégration de l'analyse monotone dans une méthode itérative

Dans le cadre de la démarche du § 4.2, on pourrait envisager l'utilisation d'une méthode de programmation mathématique "classique" pour résoudre les systèmes non linéaires formés par les fonctions contraintes actives (étape 4 de la démarche). On doit alors résoudre un "sous problème" d'optimisation avec contraintes égalités. Bien que ce problème soit a priori plus facile à résoudre que le problème complet, puisqu'il comporte au maximum *n* fonctions contraintes actives, la recherche de la solution optimale du problème complet nécessiterait néanmoins la résolution de nombreux sous problèmes. Le nombre de calculs nécessaires serait alors très certainement supérieur au nombre de calculs consommés par la résolution du problème complet. L'intégration d'une méthode itérative dans la démarche du § 4.2 semble donc irréaliste. Par contre on peut utiliser le principe des théorèmes 1 et 2 lorsqu'on applique une méthode itérative sur un problème monotone.

En effet la plupart des algorithmes de résolution pour les problèmes avec fonctions contraintes utilisent l'itération type :

$$x^{k+1} = x^k + \alpha^k d^k$$

C'est le cas notamment des méthodes primales, mais également des méthodes duales qui nécessitent une minimisation sans contrainte.

Si la fonction objectif $f(x)$ est monotone, la fonction $g(\alpha) = f(x^k + \alpha d^k)$ l'est également par rapport à la variable α,et il en est de même pour les fonctions contraintes. Dans le cas d'un problème monotone, on peut alors obtenir la situation de la figure 4.11.

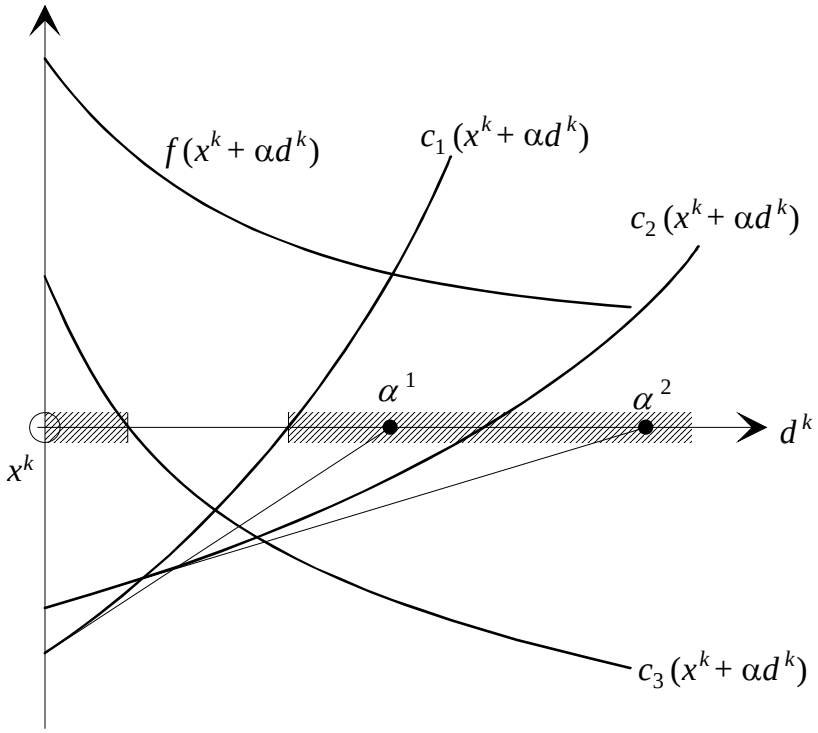


Figure 4.11 : Méthode itérative et analyse monotone.

En considérant les dérivées directionnelles :

$$\frac{df}{d\alpha}(0) = \nabla^T f(x^k).d^k$$
$$\frac{dc_j}{d\alpha}(0) = \nabla^T c_j(x^k).d^k$$

on peut appliquer les théorèmes 1 et 2 en x^k. Dans le cas de la figure, la contrainte c_3 est décroissante comme la fonction objectif par rapport à α, elle pourra donc être considérée inactive et être retirée provisoirement des calculs (théorème 1).

Considérons α_j tel que $c_j(x^k + \alpha_j d^k) = 0$ et le développement de Taylor à l'ordre 1 de $c_j(x^k + \alpha d^k)$, on a alors :

$$c_j(x^k + \alpha_j d^k) = c_j(x^k) + \alpha_j \nabla^T c_j(x^k).d^k = 0$$

Soit J_D l'ensemble des indices des fonctions contraintes telles que $\frac{df}{d\alpha}(0)$ et $\frac{dc_j}{d\alpha}(0)$ soient de signe opposé, la valeur α_D donnée par :

$$\alpha_D = \underset{j \in J_D}{Min} \left\{ \frac{-c_j(x^k)}{\nabla^T c_j(x^k).d^k} \right\}$$

définit la fonction contrainte "dominante", celle qui limite le pas de déplacement α dans la direction d^k. En vertu du théorème 1, cette contrainte peut temporairement être considérée

comme active, α_D est alors une borne supérieure au pas de déplacement admissible dans la direction d^k.

L'analyse monotone permet donc d'augmenter l'efficacité d'une méthode de recherche unidimensionnelle en fournissant à chaque itération une bonne estimation du pas de déplacement optimal. Intégrée dans une méthode de gradient réduit généralisé [71] et également dans une méthode d'approximation quadratique récursive [72], cette technique a permis d'augmenter les performances de ces méthodes entraînant une diminution significative du nombre d'itérations et d'évaluations de fonctions nécessaires.

5 Conclusions

La méthode *MOD* de *Johnson* et l'analyse monotone apparaissent finalement comme des outils privilégiés d'analyse d'un problème de conception optimale. Bien que limitée aux problèmes strictement monotones, l'analyse monotone est cependant plus générale que la méthode *MOD*, car ne nécessitant pas de graphique elle peut s'appliquer sur des problèmes de plus grande taille. En fait l'analyse monotone est très efficace sur des problèmes dans lesquels les variables n'interviennent pas toutes simultanément dans les fonctions contraintes. La table de monotonie est alors relativement "creuse", peu de combinaisons de contraintes satisfont les théorèmes 1 et 2, et finalement il reste peu de systèmes d'équations à résoudre.

Nous soulignerons le fait que ces méthodes non itératives permettent l'obtention de **l'ensemble des solutions optimales d'un problème d'optimisation, et dans beaucoup de cas sous forme analytique**. Cela permet de construire des algorithmes de recherche de la solution extrêmement simple, lorsque le problème comporte une ou deux variables discrètes. On peut alors "balayer" l'ensemble des valeurs discrètes pour trouver la valeur optimale de ces variables.

Nous citerons également les travaux de *Hansen, Jaumard* et *Lu* [31], [32], [33], qui ont mis au point un programme de résolution des problèmes monotones. Ce programme est capable d'effectuer automatiquement la démarche de résolution, de la détection des sens de variations jusqu'au calcul des solutions. Il s'appuie sur un langage de programmation permettant le calcul formel et utilise la technique d'élimination des variables décrites dans le § 4.1. Cependant, les limitations intrinsèques au calcul formel n'autorisent pas l'utilisation de ce programme sur une

"classe" de problèmes plus large que la "classe" des problèmes que l'on peut résoudre manuellement.

Les derniers travaux de recherche concernant l'analyse monotone s'orientent vers la mise au point d'outils d'analyse du problème d'optimisation, permettant de détecter des problèmes mal formulés, dans lesquels certaines variables ne seraient pas bornées par la fonction objectif ou par les fonctions contraintes [55].

Ces travaux montrent que les problèmes de conception optimale sont des problèmes d'optimisation difficiles à formuler, et que pour les traiter les méthodes numériques de résolution devront être utilisées avec beaucoup de précautions.

Chapitre 5

Méthodes pour variables mixtes

1 Généralités

Nous désignons par vecteur en variables mixtes un vecteur regroupant plusieurs types de variables :

- Des variables continues dont les valeurs peuvent varier continûment entre deux bornes réelles distinctes.
- Des variables entières astreintes à prendre des valeurs entières entre deux bornes entières distinctes.
- Des variables discrètes acceptant des valeurs discrètes **réparties de façon quelconque** entre deux bornes réelles distinctes. Nous supposerons que ces différentes valeurs sont ordonnées de façon croissante.

Les composantes du vecteur de variables mixtes seront séparées en deux parties : les composantes continues et les composantes discrètes et entières. On notera ce vecteur de la manière suivante :

$$x = \{xc, xd\}^T \in R^n$$

Où $xc \in R^{(n-d)}$ désigne les $(n-d)$ variables continues
et $xd \in E^d \subset R^d$ les d variables discrètes et entières.

Pour simplifier les notations nous regroupons l'ensemble des variables entières et discrètes sous l'appellation commune : xd. La seule différence entre ces deux types de variables se situe au niveau de la répartition de l'ensemble des valeurs acceptables. Pour les variables entières il suffit de spécifier les bornes inférieures et supérieures alors que dans le cas des variables discrètes il faudra énumérer l'ensemble des valeurs possibles.

Le problème d'optimisation en variables mixtes s'écrira :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1..m \\ x \in R^{(n-d)} \times E^d \end{cases}$$

$$\text{Avec : } f, c_j \ : \ x \in R^{(n-d)} \times E^d \longrightarrow f(x), c_j(x) \in R \quad j = 1..m$$

On suppose que les fonctions $f(x)$ et $c_j(x)$ sont **continues et différentiables sur** R^n, c'est à dire dans l'espace des variables continues étendu à tout le vecteur variable. De cette manière lorsque nous "relaxerons" les restrictions dues aux variables discrètes le problème (P_c) pourra être considéré comme un problème d'optimisation classique de *n* variables continues. Le domaine des solutions de (P_c) est toujours défini par :

$$D = \left\{ x \in R^{(n-d)} \times E^d \ / \ c_j(x) \le 0 \ j = 1...m \right\}$$

1.1 Conditions d'optimalité en variables mixtes

Le partage du vecteur variable mixte en variables continues et discrètes permet de considérer deux types de sous problèmes : l'un en variables continues, les variables discrètes étant supposées fixes, l'autre en variables discrètes les variables continues restant fixes. Ces deux sous problèmes sont généralement étroitement liés, les valeurs des fonctions objectif et contraintes du sous problème en variables continues dépendront de celles des variables discrètes et vice-versa.

L'optimum en variables mixtes, $x^* = \left\{ xc^*, xd^* \right\}^T$, sera atteint lorsque les composantes continues et discrètes du vecteur variable auront simultanément atteint leur valeur optimale.

Pour $xd = xd^*$, les composantes continues optimales xc^* satisferont les conditions d'optimalité de *Kuhn et Tucker*, sous réserve que le domaine des solutions soit qualifié en $x^* = \left\{ xc^*, xd^* \right\}^T$.
On a alors :

> Pour $xd = xd^*$, xc^* est un optimum local continu du problème en variables mixtes (P_c), s'il existe des réels $\lambda_j \ \ j = 1..m$ positifs ou nuls tels que :

$$\begin{cases} \nabla_{xc} f(xc^*, xd^*) + \sum_{j=1}^{m} \lambda_j \nabla_{xc} c_j(xc^*, xd^*) = 0 \\ \lambda_j c_j(xc^*, xd^*) = 0 \; j = 1..m \end{cases}$$

Le voisinage discret de $x = \{xc, xd\}^T$ sera défini comme l'ensemble des points de $R^{(n-d)} \times E^d$ dont les composantes discrètes prennent les valeurs discrètes "adjacentes" à celle de xd, excepté lui-même, les composantes continues restant fixes (figure 5.1).

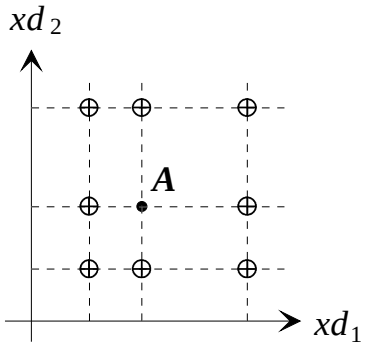


Figure 5.1 : Le voisinage discret dans E^2 est composé de 8 éléments

Si d est le nombre de variables discrètes, le voisinage discret noté $V_d(xd) \subset E^d$ est composé de $3^d - 1$ éléments.

Pour $xc = xc^*$, xd^* sera un optimum local discret pour le problème en variables mixtes (P_c) si :

$$\forall\ xd \in V_d(xd) \text{ tel que : } c_j(xc^*, xd) \le 0 \quad j = 1..m$$

on a :

$$f(xc^*, xd^*) \le f(xc^*, xd) \text{ et } c_j(xc^*, xd^*) \le 0 \quad j = 1..m$$

Cette définition ne permet pas d'établir des conditions mathématiques permettant de caractériser un optimum local discret. La seule façon envisageable pour statuer sur l'optimalité locale discrète est **d'évaluer la fonction objectif pour l'ensemble des éléments du voisinage discret appartenant au domaine des solutions du problème** (P_c).

Cette méthode n'est envisageable que lorsque le problème comporte quelques variables discrètes. En effet on voit dans le tableau ci-après, qu'à partir de 5 variables discrètes, l'évaluation de la fonction objectif pour tous les éléments du voisinage discret conduit à un nombre de calculs considérable (on suppose pour simplifier que tous les éléments du voisinage discret appartiennent au domaine des solutions).

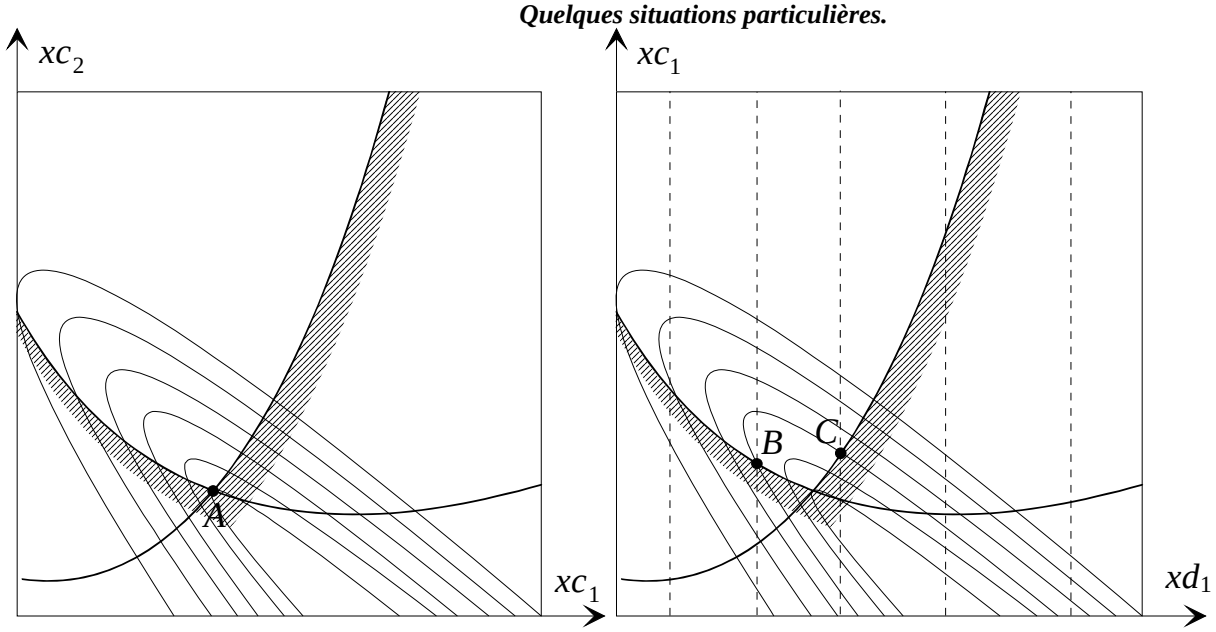
d	1	2	5	10	15
$3^d - 1$	2	8	243	59048	$1.43\,10^7$

Cette opération est d'autant plus lourde, qu'elle devra être recommencée pour chaque nouvelle valeur des composantes continues du vecteur en variables mixtes.

1.2 Influence des variables discrètes

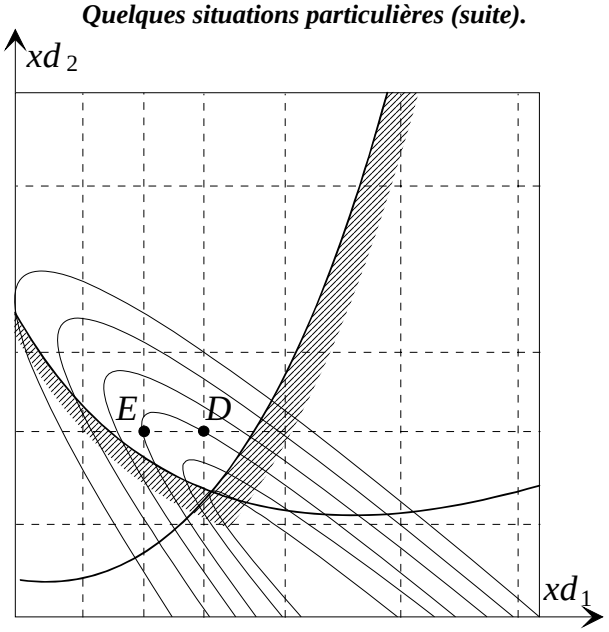
Nous avons vu au chapitre 3 qu'un problème d'optimisation strictement convexe admettait une solution optimale globale unique. Cette particularité cesse d'être vraie dans le cas des problèmes en variables mixtes.

Figure 5.2 : Problèmes convexes en variables mixtes : Quelques situations particulières.



(a) : 2 variables continues. ***(b) : 1 variable continue, 1 discrète.***

Figure 5.2 : Problèmes convexes en variables mixtes :
Quelques situations particulières (suite).



(c) : 2 variables discrètes.

La figure 5.2 représente différents cas particuliers de solutions possibles pour plusieurs combinaisons de variables continues et discrètes. Etant donné que la répartition des valeurs discrètes peut être a priori quelconque, toutes les configurations sont possibles. Dans le cas de la figure 5.2c la répartition des valeurs discrètes des variables est telle que ce problème strictement convexe admet deux solutions optimales globales (point *D* et *E*).

Nous verrons que dans certains processus itératifs de recherche de l'optimum en variables mixtes, le point de départ est donné par la solution du problème d'optimisation dans lequel toutes les restrictions dues aux variables discrètes ont été "relaxées". On obtient ainsi un point de départ dont les *n* composantes ont des valeurs quelconques. L'optimum en variables mixtes est alors choisi dans le voisinage discret de ce point en adoptant pour les variables discrètes les valeurs discrètes les plus proches.

Notons au passage que le voisinage discret d'un point en variables continues comporte 2^d éléments contrairement au voisinage discret d'un point en variables discrètes ($3^d - 1$ éléments).

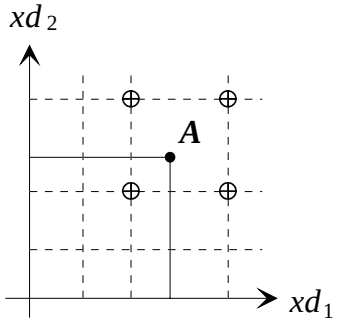


Figure 5.3 : Les 2^d éléments du voisinage discret d'un point continu.

Pour simplifier les représentations graphiques nous considérons un problème de deux variables discrètes. Dans le cas du problème de la figure 5.4, on constate que l'optimum est situé dans le voisinage discret de la solution du problème en variables continues.

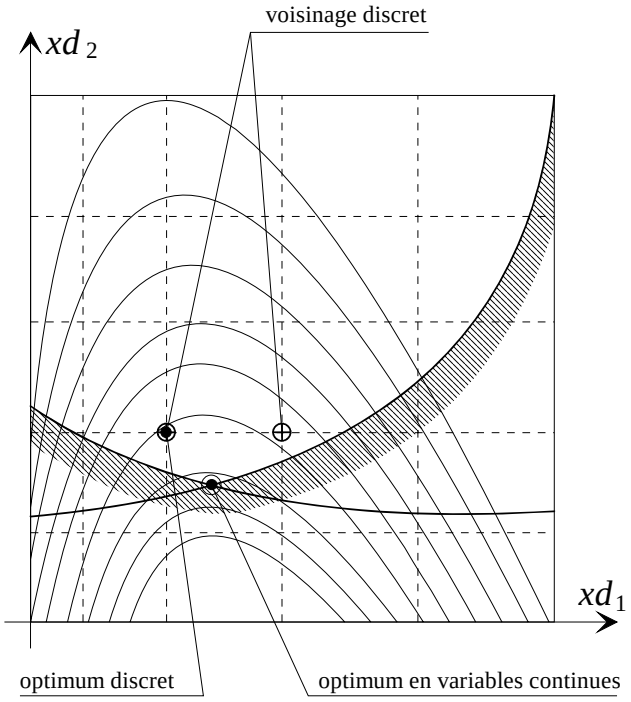


Figure 5.4 : Proximité des solutions optimales en variables discrètes et continues.

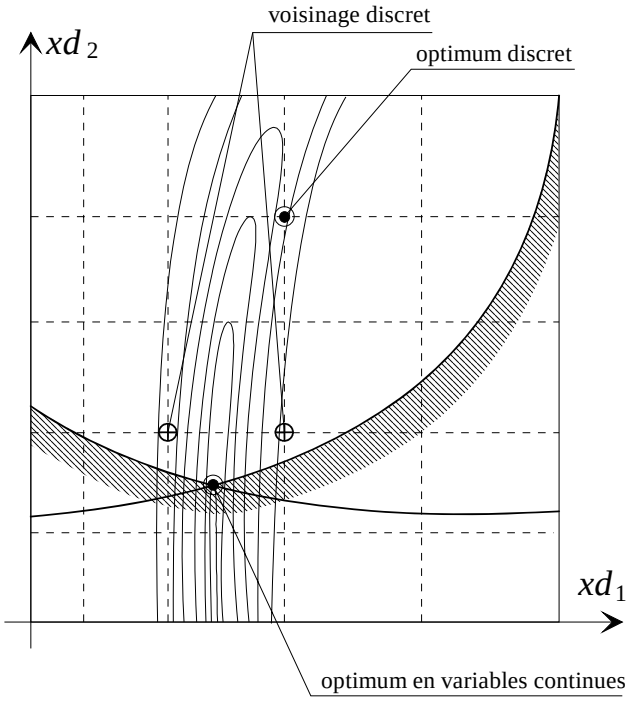


Figure 5.5 : L'optimum discret n'est pas dans le voisinage discret de la solution en variables continues.

Cependant l'optimum discret n'est pas toujours situé à proximité de l'optimum en variables continues. En conservant le même ensemble de valeurs discrètes et le même domaine

des solutions, on s'aperçoit cette fois que le choix d'un point du voisinage discret comme solution optimale discrète ne donne pas l'optimum discret (figure 5.5).

La distance entre la solution obtenue en relaxant les variables discrètes et l'optimum discret dépend de nombreux paramètres, dont entre autres : la "densité" de la répartition des valeurs discrètes, la forme du domaine des solutions et celle de la fonction objectif. Il est toujours possible d'imaginer des problèmes en variables mixtes possédant de nombreux optimum globaux et dans lesquels cette distance est aussi élevée que l'on veut.

2 Méthodes de résolution

Pour des valeurs fixées des variables discrètes, les valeurs optimales correspondantes des variables continues d'un problème en variables mixtes peuvent être obtenues avec les méthodes de résolution décrites dans le chapitre 3. Les méthodes de résolution en variables mixtes exploitent cette propriété, le problème étant alors de déterminer les valeurs optimales des variables discrètes.

Cette opération peut s'effectuer de plusieurs façons :

- Il est possible d'assimiler les restrictions dues aux variables discrètes à des fonctions contraintes et d'utiliser une méthode de pénalité pour ramener le problème en variables mixtes à un problème en variables continues.
- On peut, comme dans le cas des variables continues, imaginer une méthode de "déplacement" dans l'espace des variables discrètes.
- L'une des dernières possibilités est l'énumération explicite ou implicite de l'ensemble des points discrets appartenant au domaine des solutions.

2.1 Méthode utilisant un principe de pénalisation

Davydov et *Sigal* [15] proposent d'intégrer les restrictions dues aux variables entières grâce à une fonction contrainte égalité associée à chaque variable entière xd_i. L'expression retenue pour cette fonction contrainte est :

$$\Psi(xd_i) = \begin{cases} (xd_i - q_i)^2 & si\, xd_i < q_i \\ (xd_i - Q_i)^2 & si\, xd_i > Q_i \\ \sin^2(\pi.xd_i) & si\, xd_i \in [q_i, Q_i] \end{cases}$$

où q_i et Q_i sont les bornes entières inférieures et supérieures de xd_i

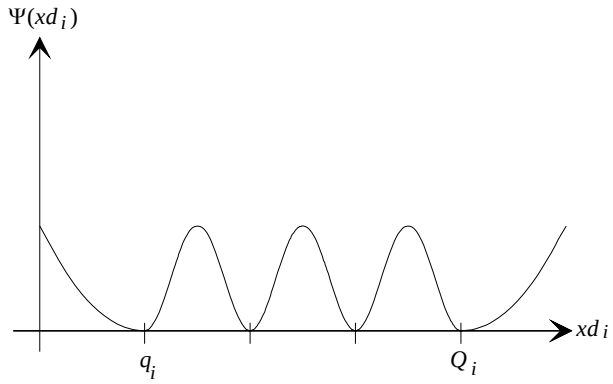


Figure 5.6 : Allure de la fonction contrainte associée aux variables entières

Ces fonctions contraintes, associées à un coefficient de pénalisation spécifique, sont ensuite ajoutées aux fonctions contraintes du problème dans une méthode de pénalité extérieure utilisant une fonction de pénalisation quadratique.

Gisvold et *Moe* [25] ont suggéré une approche similaire s'appliquant aux cas des variables discrètes avec une répartition quelconque des valeurs discrètes. Dans ce cas les restrictions dues aux variables discrètes sont intégrées dans une seule fonction contrainte dont l'expression est :

$$Q{:}xd \in E^d \longrightarrow Q(xd) \in R$$

telle que :

$$Q(xd) = \sum_{i=1}^{d} 4q_i(1-q_i)^{\beta^k} \;\; ; \;\; q_i = \frac{(xd_i - z_i^l)}{(z_i^u - z_i^l)}$$

Avec :

z_i^l, z_i^u: Bornes inférieures et supérieures des valeurs discrètes encadrant la valeur courante de xd_i.

$\beta^k \geq 1$: Exposant de pénalisation.

Les auteurs de cette méthode utilisent une méthode de pénalité intérieure pour résoudre un petit problème d'optimisation de structure (10 variables dont 8 discrètes, 27 contraintes).

A priori séduisantes, ces approches posent cependant quelques problèmes. L'introduction de ces fonctions contraintes dans une fonction de pénalisation produit un certain nombre de minimums locaux au niveau des valeurs entières et discrètes. L'obtention d'un optimum est alors délicate compte tenu des risques de convergence de la méthode de minimisation sans contrainte vers l'un de ces nombreux minimums locaux. De plus l'ajout des coefficients de pénalité associés à ces fonctions contraintes complique le choix des valeurs initiales et des stratégies d'évolution et augmente également les risques d'instabilité numérique.

2.2 Déplacement dans l'espace des variables discrètes

Pappas et *Allentuch* [52] proposent une méthode de recherche optimale discrète utilisant comme point de départ l'optimum en variables continues déterminé par une méthode de pénalité extérieure en relaxant les restrictions sur les variables discrètes. A partir de cet optimum la méthode consiste à se déplacer vers le point discret le plus proche et à procéder à une séquence de minimisation par rapport à une variable discrète les autres restant fixes. **L'ordre des séquences de minimisation est celui des "sensibilités" décroissantes de chaque variable discrète**.

La sensibilité s_i de la variable discrète xd_i est définie par :

$$s_i = \frac{\left[f^*(xd_i) - f^*(xd_i + \Delta_i) \right]}{\Delta_i}$$

Avec :

Δ_i différence entre la valeur courante xd_i et les valeurs discrètes adjacentes.

$f^*(xd_i) = P(xc^*, xd_1, \ldots, xd_i, \ldots, xd_d, r)$

$f^*(xd_i + \Delta_i) = P(xc^*, xd_1, \ldots, xd_i + \Delta_i, \ldots, xd_d, r)$

La fonction $P : (x,r) \in R^n \times R \longrightarrow P(x,r) \in R$ est une fonction de pénalisation quadratique et xc^* la valeur optimale des variables continues pour $xd_1, \ldots, xd_i, \ldots, xd_d$ fixé.

Cette minimisation par rapport à la variable discrète xd_i, utilisant une méthode de dichotomie modifiée pour tenir compte des valeurs discrètes, est répétée autant de fois qu'il y a de variables discrètes. Pour chaque nouvelle valeur discrète de xd_i, l'optimum xc^* est recalculé par la méthode de pénalité. Si l'ordre des sensibilités croissantes est modifié un nouveau cycle de minimisation est entamé. Lorsque cet ordre n'est plus modifié, ce processus est arrêté. On obtient alors généralement un optimum discret lorsque la fonction de pénalisation est unimodale.

Les auteurs précisent que cette méthode n'est applicable que sur des problèmes de petites dimensions (4,5 variables) dans lesquels les effets de la discrètisation ne sont pas trop prononcés, car dans ce cas on peut espérer avoir un optimum discret dans le voisinage discret de l'optimum en variables continues.

Cha et *Mayne* [8],[9] ont modifié une méthode d'approximation quadratique récursive pour pouvoir l'appliquer sur des problèmes en variables mixtes. A partir d'un point de départ en variables mixtes situé à l'intérieur du domaine des solutions, la direction de recherche pour une minimisation unidimensionnelle est calculée en résolvant un sous-problème d'optimisation quadratique construit au point courant du processus.

La recherche unidimensionnelle effectuée dans cette direction tient compte des limites imposées par les fonctions contraintes, des éventuelles propriétés de monotonie locale (cf. chap. 4 §4.4) et des restrictions dues aux variables discrètes. Le point obtenu est alors éventuellement arrondi au point discret le plus proche. Si cette étape ne parvient pas à améliorer la solution courante, une procédure de recherche locale discrète est appliquée. Les points discrets du voisinage discret situés du coté opposé au gradient de la fonction objectif sont considérés un à un.

Cette méthode permet d'obtenir une solution optimale en variable mixte pour 25 problèmes tests avec un nombre d'évaluations de fonction inférieur par rapport aux autres méthodes utilisées dans ce test, et notamment celles basées sur un principe de séparation et d'évaluation. Il semble toutefois que cette méthode soit réservée aux problèmes présentant un "pas de discrétisation" relativement faible et régulier. Dans le cas contraire, la procédure d'arrondi utilisée risquerait de compromettre la convergence de la méthode.

2.3 Méthodes d'énumération

Le moyen le plus sur pour localiser un optimum discret est sans doute d'énumérer l'ensemble des points discrets du domaine des solutions et de résoudre pour chacun d'eux le problème d'optimisation en variables continues, les variables discrètes étant fixées par cette énumération. Il suffira alors de retenir comme solution le point discret donnant la plus petite valeur de la fonction objectif. Cependant nous avons vu dans le §1.2 de ce chapitre que cette approche n'est valable que pour certains cas particuliers, car généralement le nombre de calculs nécessaires est prohibitif.

Grâce à un principe algorithmique s'appuyant sur une représentation particulière de l'ensemble des valeurs discrètes on peut éviter l'énumération de tous les points discrets du domaine des solutions.

2.3.1 Principe d'une méthode de séparation et évaluation (Branch and Bound)

Considérons le problème d'optimisation énoncé sous la forme très générale suivante :

$$\text{Trouver : } f(s^*) = \underset{s \in S}{Min}\{f(s)\}$$

$$\text{Avec :}$$

$$f \ : \ s \in S \longrightarrow f(s) \in R$$

Soit un sous-ensemble S_i inclus dans S. Le principe d'évaluation sera défini de la manière suivante [62]:

> On dit que l'on sait "évaluer" le sous-ensemble S_i si on peut déterminer un réel, dépendant de S_i, $g(S_i)$ tel que :
>
> $$g(S_i) \leq f(s) \quad \forall s \in S_i$$
>
> Cette évaluation est dite "exacte" si on connaît un élément de S_i, s_i^* tel que :
>
> $$g(S_i) = f(s_i^*)$$
>
> L'évaluation $g'(S_i)$ est meilleure que l'évaluation $g(S_i)$ si :
>
> $$g'(S_i) > g(S_i)$$

L'évaluation associée à un sous-ensemble vide ($S_i = \varnothing$) sera : $g(S_i) = +\infty$. On notera que la meilleure évaluation que l'on puisse obtenir est une évaluation exacte.

> L'ensemble S est dit "séparé" en k sous-ensembles S_i, $i = 1..k$ lorsque :
>
> 1) Chacun des sous-ensembles S_i est inclus dans S
> 2) L'union de tous ces sous-ensembles est égale à S

On déduit immédiatement de ces deux définitions le résultat suivant :

Lorsque pour chacun des sous-ensembles S_i tel que :

$$S_i \subset S \text{ pour } i = 1..k$$
$$S_1 \cup S_2 \cup \ldots \cup S_k = S$$

on obtient une évaluation exacte, la solution optimale s^* du problème d'optimisation est donnée par le sous-ensemble S_i^* ayant l'évaluation $g(S_i^*)$ minimale, et on a:

$$g(S_i^*) = f(s^*) = \underset{i=1..k}{Min}\{g(S_i)\}$$

Supposons que l'on connaisse une solution $\hat{s}$ approchant la solution optimale, s^* du problème d'optimisation. Par définition $\hat{s}$ est telle que :

$$f(\hat{s}) \geq f(s^*)$$

Si l'évaluation du sous-ensemble S_i est telle que :

$$g(S_i) \geq f(\hat{s}) \geq f(s^*)$$

Le sous-ensemble S_i ne peut pas contenir de solution optimale, sinon cela contredirait le principe d'évaluation définit plus haut. Dans ce cas il sera inutile de décomposer (séparer) ce sous-ensemble en sous-ensembles plus petits. Ce sous-ensemble sera qualifié de "stérile". Lorsque l'évaluation pour un sous-ensemble donné est exacte, ce sous-ensemble peut être également "stérilisé".

L'algorithme général d'une méthode par séparation et évaluation est le suivant [62]:

1) Au départ la valeur de $\hat{s}$ est indéterminée et $\hat{f} = f(\hat{s}) = +\infty$
2) Tant qu'il reste des sous-ensembles à séparer (non stérile) effectuer les opérations suivantes :
3) Choix d'un sous-ensemble à séparer (au départ on sépare l'ensemble *S* tout entier).
4) Calcul de l'évaluation :
   si l'évaluation est exacte alors $\hat{s} = s_i^*$ et $\hat{f} = f(\hat{s}) = f(S_i^*) = g(S_i)$
5) Séparation du sous-ensemble.
6) Stérilisation des sous-ensembles S_i tel que :
   $$g(S_i) \geq f(\hat{s}) \text{ ou } f(s_i^*) = g(S_i)$$

La méthode sera entièrement définie lorsqu'on aura précisé :

- Le choix du sous-ensemble à séparer à chaque étape.
- Le principe de calcul de l'évaluation.
- La méthode de séparation des sous-ensembles.

L'association d'une arborescence à la décomposition progressive de l'ensemble permet de visualiser la mise en œuvre d'une procédure de recherche par séparation et évaluation. Le sommet de ce graphe est représenté par l'ensemble *S* tout entier, à chaque étape les sommets

pendants correspondent aux sous-ensembles S_i. Les arcs reliant chaque sommet représentant les relations d'inclusion (figure 5.7).

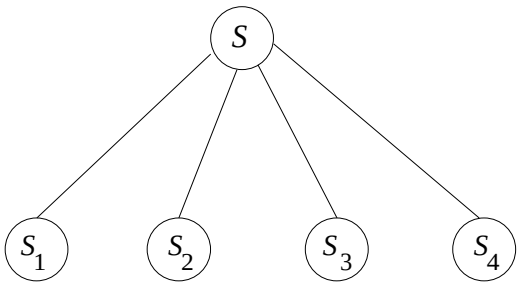


Figure 5.7 : Arborescence associée à la décomposition de S

Les performances d'une méthode de recherche de la solution optimale par séparation et évaluation dépendent pour une très large part de la manière dont sont choisis les sous-ensembles à séparer et de la méthode de séparation de ces sous-ensembles, mais également de la qualité de l'évaluation [45]. En pratique il faudra trouver un bon compromis entre un principe d'évaluation donnant une approximation assez grossière de la valeur minimale de la fonction objectif sur un sous-ensemble donné, mais rapide à calculer et une évaluation plus précise et donc souvent plus longue à calculer.

Les stratégies pour le choix du sous-ensemble à séparer sont généralement les suivantes :

1) Sélectionner le sous-ensemble dont l'évaluation est la plus faible possible, avec l'idée que ce sous-ensemble a le plus de chances de contenir une solution optimale. Ce choix correspond à une stratégie d'exploration de l'arborescence dite de "largeur d'abord". Le risque ici est d'avoir à explorer une fraction importante de l'arborescence avant d'obtenir une évaluation exacte, donc une solution approchée $\hat{s}$.

2) Sélectionner le sous-ensemble le plus récemment séparé. Il s'agit alors d'une exploration dite de "profondeur d'abord". Cette méthode permet d'obtenir le plus tôt possible une solution approchée $\hat{s}$, par contre cette solution est généralement moins bonne que celle obtenue par une exploration de "largeur d'abord". L'inconvénient de ce type de choix est de ne pas tenir compte de la fonction d'évaluation, on est alors amené à traiter des sous-ensembles dont l'évaluation est très médiocre et qui ont peu de chance de contenir une solution optimale.

On a toujours intérêt à obtenir rapidement une solution approchée $\hat{s}$, car celle-ci permet de stériliser très tôt les sous-ensembles ayant une évaluation supérieure à la valeur de la fonction objectif pour cette solution. On pourra stériliser d'autant plus de sous-ensembles que cette solution approchée est meilleure.

Il n'existe pas de stratégies donnant systématiquement de meilleurs résultats, cela dépend du problème à traiter. L'approche la plus couramment utilisée consiste à débuter par une exploration de "profondeur d'abord", permettant d'obtenir rapidement une solution approchée $\hat{s}$, et ensuite de poursuivre par une exploration de "largeur d'abord" pour améliorer cette solution [62].

On remarque que la méthode présente une certaine indépendance vis à vis de la structure du problème d'optimisation. Plus précisément les liens entre une méthode procédant par séparation et évaluation et le type du problème à traiter (linéaire, non linéaire, avec ou sans fonction contrainte) sont définis par le principe d'évaluation et la technique de séparation de l'ensemble *S*.

2.3.2 Mise en œuvre d'un principe de séparation et évaluation : résolution des problèmes linéaires en nombres entiers

C'est pour résoudre ce type de problème que le principe de séparation et évaluation fut mis au point. Ce principe a ensuite été appliqué sur de nombreux cas de problèmes de graphes et d'optimisation combinatoire [46]. Nous présenterons ici les particularités de cette mise en œuvre en précisant seulement le principe de séparation et celui d'évaluation spécifique aux problèmes linéaires en nombre entier.

Un problème d'optimisation linéaire en nombres entiers peut s'écrire sous la forme :

$$\begin{cases} \textit{Minimiser } f(x) = c^T.x \\ \textit{Sous les fonctions contraintes :} \\ A.x + b \le 0 \\ x_i \in N \quad i = 1 \ldots n \end{cases}$$

Avec $c \in R^n$, $b \in R^m$ et A matrice $m \times n$ de rang n.

Nous supposerons que l'ensemble des solutions *S* tel que :

$$S = \left\{ x \in R^n \,/\, A.x + b \le 0; x_i \in N; i = 1 \ldots n \right\}$$

est borné.

Soit deux vecteurs de *n* composantes entières *u* et *v* tel que : $u < v$. Le sous-ensemble $S_{u,v}$ inclus dans *S* et défini grâce à ces deux vecteurs est :

$$S_{u,v} = \left\{ x \in S \,/\, u \le x \le v \right\}$$

Si on associe à $S_{u,v}$ le problème linéaire en nombres entiers suivant :

$$(P_{u,v})\begin{cases}\text{Minimiser} \quad f(x)=\mathrm{c}^{\mathrm{T}}.\mathrm{x}\\ \text{Sous les fonctions contraintes :}\\ A.x+b\le 0\\ u\le x\le v; x_i\in N\, i=1\ldots n\end{cases}$$

et le problème linéaire en variables continues et bornées :

$$(P'_{u,v})\begin{cases}\text{Minimiser} \quad f(x)=\mathrm{c}^{\mathrm{T}}.\mathrm{x}\\ \text{Sous les fonctions contraintes :}\\ A.x+b\le 0\\ u\le x\le v\end{cases}$$

on peut facilement définir le principe d'évaluation pour ce sous-ensemble $S_{u,v}$. En effet il suffit de voir que l'ensemble des solutions de $(P_{u,v})$ est inclus dans celui de $(P'_{u,v})$ et que par conséquent la valeur optimale de la fonction objectif de $(P'_{u,v})$ est nécessairement inférieure ou égale à celle de $(P_{u,v})$. D'autre part, lorsque la solution optimale de $(P'_{u,v})$ est "entière", la valeur de la fonction objectif de $(P'_{u,v})$ est égale à celle du problème $(P_{u,v})$. Donc la valeur optimale de la fonction objectif de $(P'_{u,v})$ constitue bien une évaluation (au sens de la définition du §2.3.1) du sous-ensemble $S_{u,v}$. Lorsque la solution optimale correspondante est entière cette évaluation est "exacte". On notera qu'il n'existe pas ici de meilleure évaluation.

Le principe de séparation est le suivant [5],[62]:

Soit $\bar{x}$ la solution optimale de $(P'_{u,v})$. Pour séparer $S_{u,v}$ on choisira un indice r tel que la $r^{\text{ième}}$ composante de $\bar{x}$ **ne soit pas entière**. L'ensemble $S_{u,v}$ est alors séparé en **deux sous ensembles disjoints** $S_{u',v}$ et $S_{u,v'}$ où u' et v' sont deux vecteurs de n composantes entières tels que :

$$u'_i=\begin{cases}Ent(\bar{x}_r)+1 & sii=r\\ u_i & sii\neq r\end{cases} \text{ et } v'_i=\begin{cases}Ent(\bar{x}_r) & sii=r\\ v_i & sii\neq r\end{cases}$$

Ent(x) : Partie entière du réel x

On a alors :

$$\begin{array}{|c|}\hline S_{u,v'}\cap S_{u',v}=\varnothing\\ S_{u,v'}\cup S_{u',v}=S_{u,v}\\ \bar{x}\notin S_{u',v}\ ;\ \bar{x}\notin S_{u,v'}\\ \hline\end{array}$$

Il existe de nombreuses règles pour le choix de la variable de séparation, la plus couramment appliquée correspond au choix de $\bar{x}_r$ tel que la valeur de $\bar{x}_r$ soit le plus "**fractionnaire possible**". C'est à dire lorsque $|\bar{x}_r - Ent(\bar{x}_r)|$ est le plus proche possible de *0.5*.

L'algorithme de résolution est identique à celui du §2.3.1. Au départ de la méthode on affectera à *u* et *v* de grandes valeurs, et on choisira comme point de départ la solution optimale de $(P'_{u,v})$.

2.3.3 Application aux problèmes non linéaires en variable mixtes

Gupta et *Ravindran* [29] ont réutilisé le principe ci-dessus pour l'appliquer au problème non linéaire en variables entières suivant :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \\ x_i \in N \; i = 1..n \end{cases}$$

La méthode de séparation des sous ensembles est identique, seul le type du problème associé à chaque sous-ensemble est modifié. Dans ce cas pour évaluer le sous-ensemble défini par :

$$S_{u,v} = \left\{ x_i \in N i = 1..n / c_j(x) \le 0 \; j = 1...m; u \le x \le v \right\}$$

il faut résoudre le problème d'optimisation en variables continues suivant :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \\ u \le x \le v \end{cases}$$

Dans l'algorithme proposé ce calcul est effectué avec une méthode de gradient réduit généralisé. L'étude statistique menée par les auteurs sur une vingtaine de problèmes tests a montré que pour **un problème non linéaire en nombres entiers, la séparation des sous ensembles suivant la variable dont la valeur est la plus fractionnaire possible donne les meilleurs résultats.**

Sandgren [65] propose une démarche similaire, applicable sur des problèmes en variables mixtes. Le principe de séparation pour les variables entières est identique. Dans le cas des variables discrètes, les bornes *u* et *v* sont données par les valeurs discrètes immédiatement supérieures et inférieures à la valeur "non discrète" de la variable de séparation. L'auteur suggère d'effectuer la séparation suivant la variable discrète provoquant la plus forte diminution de la

fonction objectif. On retrouve ici l'idée de *Pappas* et *Allentuch*, d'effectuer les minimisations unidimensionnelles discrètes dans l'ordre des "sensibilités" décroissantes (Cf. §2.2).

Les principaux inconvénients des méthodes procédant par séparation et évaluation de l'ensemble des solutions sont [41], [42] :

- Elles nécessitent généralement de nombreux calculs pour parvenir à une solution optimale. En effet le nombre de problèmes non linéaires à résoudre est d'autant plus important que l'arborescence comporte de sommets (ou sous ensembles) non stérilisés.
- Il est généralement impossible de connaître la taille de l'arborescence qui sera développée, donc la quantité de mémoire qui sera utilisée, pour la résolution d'un problème.

Elles possèdent cependant les avantages fondamentaux suivants :

- Ces méthodes permettent d'obtenir :
  Une solution optimale en variables mixtes (la première solution approchée pour laquelle on obtient une évaluation exacte, par exemple).
  Toutes les solutions optimales en variables mixtes en explorant l'ensemble de l'arborescence.
- Elles sont extrêmement souples au niveau de leur mise en œuvre, et peuvent prendre en compte les structures discrètes les plus variées.

3 Conclusions

Les problèmes d'optimisation non linéaires en variables mixtes sont des problèmes d'optimisation très difficiles à résoudre. Ces difficultés sont intrinsèquement liées à la présence des variables discrètes. En effet elles introduisent de nombreux optimums globaux même dans le cas de problèmes convexes. De plus ces optimums discrets ne sont pas nécessairement situés à

proximité d'une solution optimale en variables continues. Dans ce type de situation les méthodes procédant par approximation ne donnent pas de bons résultats. L'absence de conditions d'optimalité permettant de statuer efficacement sur l'optimalité d'une solution couramment calculée constitue un sérieux écueil dans la mise au point de méthodes de résolution efficaces.

En ce qui concerne les problèmes de conception optimale, dans lesquels la présence des variables discrètes est inévitable, les méthodes procédant par séparation et évaluation sont certainement les mieux adaptées à la recherche des solutions optimales pour ce type de problèmes. L'argument de poids en faveur de ces méthodes est sans aucun doute la souplesse de leur mise en œuvre et leur capacité à prendre en compte des variables discrètes avec de très larges pas de discrétisation, et réparties de façon quelconque. Comme généralement les problèmes de conception optimale sont de taille moyenne (une dizaine de variables), on peut espérer obtenir une arborescence associée à l'ensemble des solutions de taille raisonnable.

Résolution des problèmes de conception optimale :

Implémentation d'une méthode

Chapitre 6

Implémentation d'une méthode

1 Choix d'une méthode de résolution

Les exemples de problèmes de conception optimale présentés précédemment nous ont permis de définir le type de problèmes d'optimisation que nous aurons à traiter. Nous avons vu :

- Que la présence de variables mixtes dans un problème de conception optimale était inévitable.
- Qu'il s'agit de problèmes non linéaires comportant beaucoup de fonctions contraintes par rapport au nombre de variables (ex : le ressort de pompe hydraulique, 2 variables et 7 fonctions contraintes).
- Que les variations des valeurs des données du problème peuvent modifier le domaine des solutions. En effet celui-ci peut être convexe pour certaines valeurs des données et devenir non convexe pour de nouvelles valeurs.

En considérant le choix d'une méthode destinée à traiter les problèmes en variables mixtes, utilisant un principe de séparation et d'évaluation, nous sommes tout d'abord amenés à préciser celui d'une méthode de résolution pour les problèmes en variables continues.

1.1 Tests comparatifs d'algorithmes d'optimisation

Un certain nombre d'études concernant l'évaluation et la comparaison des performances de divers algorithmes d'optimisation ont été publiées [16],[63]. Le but de ces tests est d'identifier la méthode d'optimisation donnant les meilleurs résultats : c'est à dire celle qui donne le code de calcul capable de résoudre le plus de problèmes possibles avec les meilleures performances. Dans ces études comparatives les indices de performances, permettant un classement, sont

généralement établis à partir du temps de calcul, du nombre d'itération et d'évaluation nécessaire et de la précision de la solution obtenue. Les codes de calculs considérés, regroupés suivants la méthode d'optimisation utilisée, sont appliqués à un ensemble de problèmes "tests".

Par exemple, l'étude de *Sandgren* et *Ragsdell* [63] porte sur 35 codes de calculs utilisant les méthodes suivantes :

Méthode de linéarisation.

Gradient réduit généralisé.

Méthode de pénalité extérieure et intérieure.

Les auteurs de cette étude ont évalué les performances de ces codes de calcul sur un ensemble de 14 problèmes d'optimisation non linéaires comportant de 2 à 48 variables et comportant jusqu'à 19 fonctions contraintes inégalités et 11 fonctions contraintes égalités. Cette étude montre que globalement les codes de calculs utilisant la méthode du gradient réduit généralisé sont capables de résoudre plus de problèmes et de façon plus efficace que les méthodes de pénalité. En ce qui concerne les méthodes de pénalités les résultats de ce test mettent en évidence la supériorité d'un principe de pénalisation extérieure. Les auteurs ont également constaté que des codes de calculs utilisant la même méthode d'optimisation peuvent avoir des performances très différentes.

Notre objectif, n'est pas de mettre au point un algorithme susceptible de résoudre le maximum de problèmes d'optimisation. Dans cette étude, nous cherchons à définir la méthode la mieux adaptée à un type particulier de problèmes d'optimisation.

1.2 Choix d'une méthode adaptée

Même si les algorithmes utilisant une méthode de gradient réduit généralisé semblent présenter des performances intéressantes, la nécessité d'un point de départ situé à l'intérieur du domaine des solutions constitue pour nous un sérieux inconvénient.

La conclusion du chapitre 3 nous a permis d'établir que les méthodes duales étaient les mieux adaptées à la résolution des problèmes de conception optimale. En effet, avec ce type de méthodes le point de départ peut être quelconque, et il sera possible d'obtenir un optimum global sur certains cas de problèmes non convexes. Nous avons également montré l'avantage du lagrangien augmenté qui permet d'obtenir un point col avec des valeurs du coefficient de pénalité moins élevées que dans le cas des méthodes de pénalité classiques.

A propos des algorithmes basés sur l'utilisation du lagrangien augmenté, *Minoux* [45] rapporte qu'ils font partie des méthodes les plus efficaces pour résoudre les problèmes d'optimisation fortement non linéaires. *Vanderplaats* [67] fournit également quelques résultats de calculs effectués à l'aide d'une méthode de lagrangien augmenté. Appliquée sur un type de problème différent des nôtres, (minimisation de la masse d'une poutre "console" de section variable), l'algorithme montre de bonnes performances par rapport aux méthodes de pénalités.

Notre choix se portera donc sur une méthode utilisant le lagrangien augmenté.

2 Algorithme utilisant le lagrangien augmenté

2.1 Définition de l'algorithme

L'algorithme que nous proposons s'applique aux problèmes d'optimisation posés sous la forme :

$$(P_c)\begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \\ c_j(x) = 0 \quad j = m+1...m+l \end{cases}$$

$$\text{Avec } f, c_j : x \in R^n \longrightarrow f(x), c_j(x) \in R \quad j = 1..m+l$$

Bien que les problèmes de conception optimale ne comportent généralement que des fonctions contraintes inégalités, pour plus de généralité nous prendrons en compte l'éventualité de fonctions contraintes égalités.

Pour ce type problème, l'expression du lagrangien augmenté est la suivante :

$$\hat{L}(x,\lambda,r) = f(x) + \sum_{j=1}^{m}\left(\lambda_j \Psi_j(x) + r.\Psi_j{}^2(x)\right) + \sum_{j=m+1}^{m+l}\left(\lambda_j c_j(x) + r.c_j{}^2(x)\right)$$

$$\text{avec } \Psi_j(x) = Max\left\{c_j(x), -\frac{\lambda_j}{2r}\right\} \text{ et } r > 0$$

dont le gradient par rapport à *x* est :

$$\nabla_x \hat{L}(x,\lambda,r) = \nabla f(x) + \sum_{j=1}^{m} \left[\left(\lambda_j + 2r.\Psi_j(x) \right) . \nabla c_j(x) \right]$$
$$+ \sum_{j=m+1}^{m+l} \left[\left(\lambda_j + 2r.c_j(x) \right) . \nabla c_j(x) \right]$$

L'algorithme utilisant le lagrangien augmenté est celui défini dans le chapitre 3, que nous rappelons ici :

0) Point de départ : $x^0, \lambda^0, r^0, k \leftarrow 0$
1) Calcul de : $\hat{L}(x^k,\lambda^k,r^k) = \underset{x \in R^n}{Min} \left\{ \hat{L}(x,\lambda^k,r^k) \right\}$
2) Mise à jour des variables duales : $\lambda^{k+1} \longrightarrow \lambda^k$
3) Si test d'arrêt vérifié : **FIN**
4) Sinon $r^{k+1} \longrightarrow r^k$ avec $r^{k+1} \geq r^k$, $k \leftarrow k+1$

retour en 0)

Pour la mise à jour des variables duales nous adoptons la relation :

$$\lambda_j{}^{k+1} = \lambda_j{}^k + 2r.\Psi_j(x) \text{ pour } j = 1..m$$
et
$$\lambda_j{}^{k+1} = \lambda_j{}^k + 2r.c_j(x^k) \text{ pour } j = m+1..m+l \qquad (1)$$

correspondant à une itération de type "plus forte pente" sur la fonction duale augmentée. Ce qui à priori ne donne pas de bons résultats sur une fonction mal conditionnée numériquement (cf. chap. 3 § 2.5.1). Il faut cependant remarquer que la fonction duale augmentée est d'autant mieux conditionnée que la valeur de *r* est élevée. On aura donc intérêt à utiliser des valeurs élevées pour *r* afin d'obtenir une bonne vitesse de convergence pour cette formule de mise à jour des variables duales. La difficulté est alors de calculer le minimum en *x* de $\hat{L}(x,\lambda^k,r^k)$ à chaque itération car le conditionnement numérique de cette fonction se dégrade lorsque *r* augmente.

Nous avons choisi d'effectuer ce calcul avec une méthode de minimisation quasi newtonienne utilisant une formule de correction de type BFGS. Les raisons qui motivent ce choix sont les suivantes :

La robustesse de la propriété de convergence globale de ce type de méthode va nous permettre d'utiliser des valeurs élevées pour *r*, le coefficient de pénalité, sans trop affecter la vitesse de convergence de la méthode BFGS.

La précision de la recherche unidimensionnelle pour ce type de méthode n'est pas fondamentale, on pourra donc se contenter d'un minimum unidimensionnel approché. Cette particularité permettra d'obtenir le minimum en *x* de $\hat{L}(x,\lambda^k,r^k)$ avec peu d'évaluations de fonctions.

2.1.1 Test de convergence

Dans les méthodes de minimisation sans contrainte, le test de convergence consiste à vérifier les conditions d'optimalité au point couramment calculé : c'est à dire vérifier que la norme du gradient au point courant est suffisamment faible. Dans le cas des problèmes avec fonctions contraintes, il faudrait normalement vérifier que les conditions de *Kuhn et Tucker* sont satisfaites pour chaque point.

On constate qu'en première approximation on a, après l'étape de minimisation :

$$\nabla_x \hat{L}(x^k,\lambda^k,r^k) = \nabla f(x^k) + \sum_{j=1}^{m}\left[\left(\lambda_j{}^k + 2r^k.\Psi_j(x^k)\right).\nabla c_j(x^k)\right] + \sum_{j=m+1}^{m+l}\left[\left(\lambda_j{}^k + 2r^k.c_j(x^k)\right).\nabla c_j(x^k)\right] \approx 0 \qquad (2)$$

Lorsque les termes $2r^k.\Psi_j(x^k)$, $j = 1..m$ et $2r^k.c_j(x^k)$, $j = m+1..m+l$ sont nuls l'expression (2) devient :

$$\nabla_x \hat{L}(x^k,\lambda^k,r^k) = \nabla f(x^k) + \sum_{j=1}^{m+l}\left[\lambda_j{}^k \nabla c_j(x^k)\right] \approx 0$$

on voit alors que $\left(x^k,\lambda^k\right)$ vérifie les conditions de *Kuhn et Tucker*, en supposant que le domaine des contraintes est qualifié en x^k.

On pourra donc considérer que l'algorithme a convergé vers une solution optimale lorsque :

$$\begin{gathered}\Psi_j(x^k) < \varepsilon_1 \text{ pour } j = 1..m \\ \left|c_j(x^k)\right| < \varepsilon_2 \text{ pour } j = m+1..m+l \\ \text{avec :} \\ \varepsilon_1 \text{ et } \varepsilon_2 \text{ réels petits et positifs}\end{gathered} \qquad (3)$$

Nous donnons ici une justification approximative de ce test de convergence, on trouvera dans [30] plus de précisions sur cette démonstration.

L'existence d'un point col pour le lagrangien augmenté est fondamentale, c'est l'une des conditions qui permettent d'établir la propriété de convergence globale de ce type d'algorithme. Dans ce cas la valeur optimale du problème dual associé et celle du problème sont égales.

A la fin de l'étape de chaque minimisation du lagrangien on obtient la valeur de la fonction duale augmentée, puisque :

$$\hat{w}(\lambda^k,r^k) = \underset{x\in R^n}{Min}\left\{\hat{L}(x,\lambda^k,r^k)\right\} = \hat{L}(x^k,\lambda^k,r^k)$$

Un bon moyen de tester la convergence est alors de calculer l'écart entre la valeur du lagrangien augmenté en (x^k, λ^k, r^k)et celle de la fonction objectif en x^k. Ce qui donne le test d'arrêt suivant :

$$\frac{\left|\hat{L}(x^k, \lambda^k, r^k) - f(x^k)\right|}{1 + \left|f(x^k)\right|} \leq \varepsilon_3$$

Cette expression, donnant l'écart relatif lorsque $\left|f(x^k)\right| >> 1$ et absolu pour $\left|f(x^k)\right| << 1$ a l'avantage d'être définie quelle que soit la valeur de $f(x^k)$.

En pratique ces deux tests de convergence seront effectués simultanément, et nous considérons que (x^k, λ^k) est une solution optimale du problème lorsque l'un ou l'autre de ces deux tests sera satisfait.

2.1.2 Choix des valeurs initiales des multiplicateurs

Pour commencer les calculs, la méthode nécessite le choix des valeurs initiales des multiplicateurs, λ^0, associés aux fonctions contraintes du problème. Lorsque les valeurs optimales des multiplicateurs sont connues nous avons vu que l'optimum du problème était atteint en une seule étape de minimisation. Dans la pratique ces valeurs sont inconnues, plusieurs minimisations et mises à jour des variables duales seront alors nécessaires.

Le meilleur choix pour des valeurs initiales est sans aucun doute :

$$\lambda_j^{\,0} = 0 \text{ pour } j = 1..m + l$$

La première itération correspond alors à celle d'une méthode de pénalité extérieure avec une fonction de pénalisation quadratique. Ce choix à l'avantage de ne pas démarrer le calcul avec des "fausses" valeurs des multiplicateurs. Il correspond au cas où toutes les fonctions contraintes sont inactives. En fin de convergence les multiplicateurs positifs permettront d'identifier rapidement les contraintes actives à l'optimum.

2.1.3 Evolution du coefficient de pénalité

C'est certainement le point le plus délicat dans ce type d'algorithme. En effet il n'existe pas de règle systématique concernant l'évolution de ce coefficient. Comme généralement le point de départ, x^0, n'appartient pas au domaine des solutions, plusieurs fonctions contraintes ont des valeurs positives et les termes $\Psi_j^{\,2}(x)$ ont alors des valeurs non négligeables. Il convient donc de choisir une valeur initiale de *r* modérée.

Le coefficient de pénalité sera augmenté à chaque itération avec la relation :

$$r^{k+1} = \gamma r^k \text{ avec } \gamma \geq 1 \qquad (4)$$

Afin d'éviter les problèmes numériques liés à des valeurs excessives de *r*, la croissante de ce coefficient de pénalité sera limitée par une valeur maximale r_{max}.

2.1.4 Minimisation du lagrangien

Pour l'étape de minimisation sans contrainte nous avons utilisé la méthode BFGS de la librairie de calculs scientifiques NAG. Notre choix s'est porté sur cette librairie, car lorsque nous avons commencé à développer ce code de calcul, c'était la seule librairie de calculs dont nous disposions qui offrait ce genre de méthodes. Ce choix s'est avéré amplement justifié par la suite, compte tenu des performances obtenues.

La méthode BFGS utilisée s'appuie sur l'algorithme défini dans le chapitre 3 § 2.5.4. Elle utilise un principe de factorisation de Cholesky, ce qui permet de mettre à jour indépendamment les termes diagonaux et non diagonaux de l'inverse du hessien (Chap. 3 § 2.5.4.2). L'algorithme intègre également le préconditionnement de l'inverse du hessien et une stratégie de réinitialisation de la diagonale de la matrice mise à jour, toutes les *n* itérations (*n* : nombre de variables) [23].

La méthode de recherche unidimensionnelle utilise une méthode de type "sécante" couplée avec une technique d'interpolation polynomiale cubique.

Les paramètres de calculs (η, τ_F) de cet algorithme permettent de définir la précision souhaitée pour chaque étape de recherche unidimensionnelle dont le critère d'arrêt est du type :

$$\left|g'(\alpha^j)\right| < -\eta g'(0)$$

et la précision finale de l'optimum est donnée par les tests d'arrêt suivant :

$$\begin{aligned}
&f(x^{k+1}) - f(x^k) < \tau_F \left(1 + \left|f(x^k)\right|\right) \\
&\left\|x^{k+1} - x^k\right\| < \sqrt{\tau_F}\left(1 + \left\|x^k\right\|\right) \\
&\left\|\nabla^T f(x^k).d^k\right\| < \sqrt[3]{\tau_F}\left(1 + \left|f(x^k)\right|\right)
\end{aligned}$$

Etant donné que pour des valeurs élevées du coefficient de pénalité, le lagrangien augmenté est mal conditionné numériquement, ce calcul de minimisation ne se termine pas systématiquement avec succès. Les principales causes d'échec sont les suivantes :

1) Le nombre d'itérations alloué à ce calcul est insuffisant.
2) La direction de recherche unidimensionnelle calculée à partir du gradient du lagrangien augmenté et l'inverse de son hessien n'est plus une direction de descente, on a alors :

$$g'(0) = \nabla_x^T \hat{L}(x^k, \lambda^k, r^k).d^k > 0$$

Dans cette situation la pente à l'origine de la fonction $g(\alpha) = \hat{L}(x^k + \alpha d^k, \lambda^k, r^k)$ n'est plus négative et le critère d'arrêt de la recherche unidimensionnelle ne peut plus s'appliquer.

Le premier cas de figure est assez simple à gérer, il suffit d'augmenter le nombre d'itérations disponibles. Dans le second cas il est beaucoup plus difficile de déterminer la cause précise de l'échec. En effet l'erreur peut provenir d'un calcul de gradient erroné (erreur dans la codification des gradients des fonctions objectif et contraintes) ou du fait que l'inverse du hessien de l'étape précédente n'était plus défini positif. Cela se produit généralement lorsque le coefficient de pénalité est trop élevé, il faudra dans ce cas relancer les calculs en modifiant l'évolution du coefficient de pénalité.

Cette opération pourrait être effectuée automatiquement, en choisissant de multiplier le coefficient de pénalité courant par un coefficient correcteur (<1) fixé au départ des calculs. On pourrait alors renouveler l'étape de minimisation avec un coefficient de pénalité moins élevé. Nous avons essayé cette alternative, et cela n'a pas donné de bons résultats. Dans la majorité des cas on a pu, tout au plus, effectuer quelques minimisations supplémentaires, l'erreur se renouvelant quelques itérations plus loin.

De façon générale, dans tout calcul d'optimisation, il n'est pas toujours souhaitable de gérer les erreurs de façon automatique. Cette gestion d'erreurs est souvent délicate, car celles ci sont rarement dues à un phénomène se produisant à l'itération courante, mais plus généralement à l'accumulation d'erreurs d'approximation dans les itérations précédentes. Il est alors préférable d'arrêter des calculs mal engagés plutôt que de les poursuivre à tout prix, au risque d'obtenir une solution finale erronée.

2.2 Organigramme de calcul

L'ensemble des choix définissant l'algorithme ayant été fait, on peut maintenant établir avec plus de précision l'organigramme du code de calcul utilisant le lagrangien augmenté (figure 6.1) que nous avons réalisé.

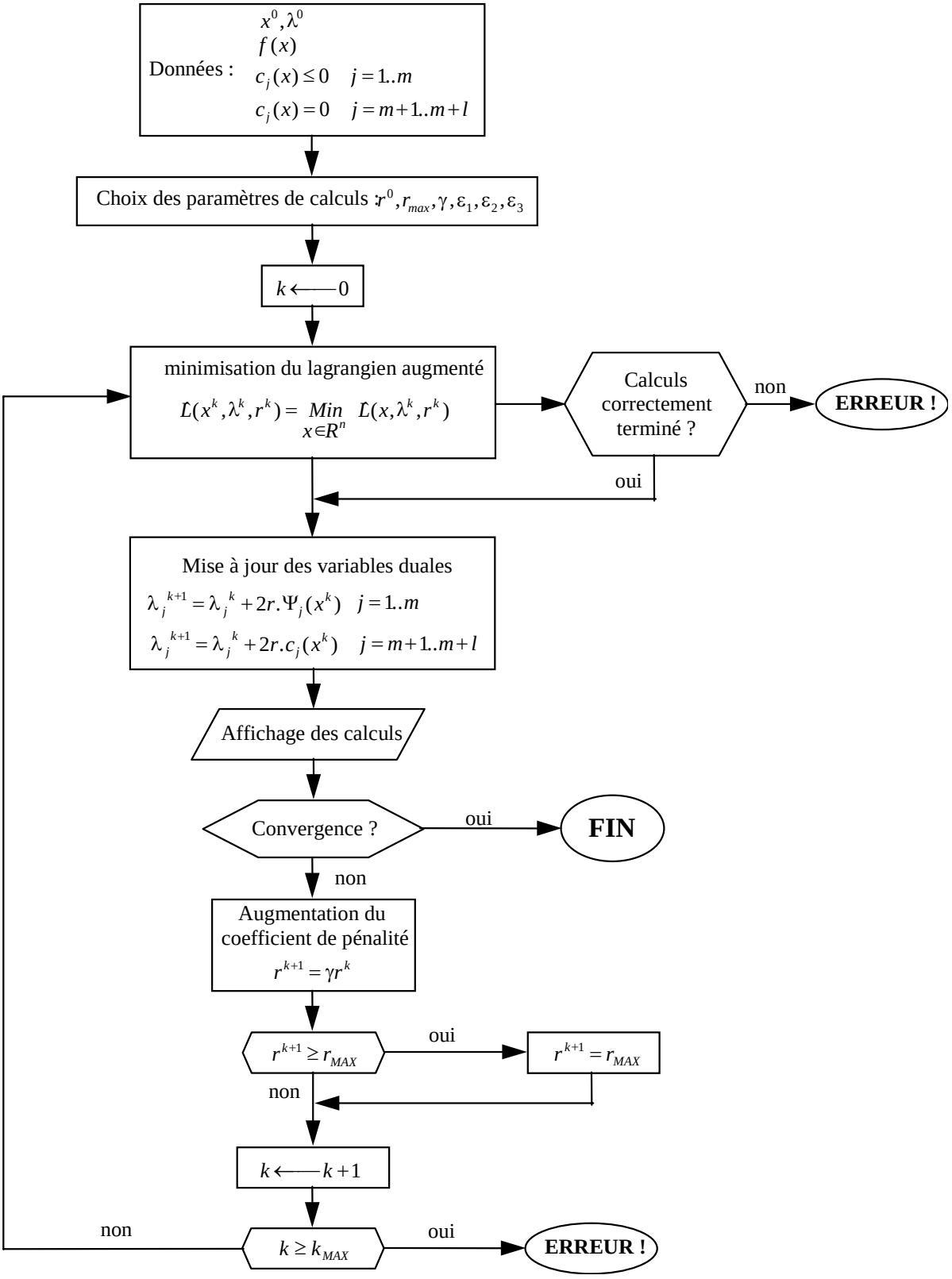


Figure 6.1 : Organigramme du code de calcul basé sur le lagrangien augmenté

2.3 Calcul des gradients par différences finies

Dans le code de calcul que nous avons réalisé, les gradients des fonctions objectif et contraintes peuvent être déterminés analytiquement et codifiés avec les équations du problème d'optimisation. Pour des problèmes simples, avec peu de variables, cette opération est facilement réalisable à la main. Lorsque le nombre de variables et de fonctions contraintes augmente, ce calcul analytique devient rapidement lourd et source d'erreurs. Une erreur dans la codification ou dans le calcul d'une dérivée partielle perturbe gravement l'algorithme et provoque généralement un arrêt des calculs.

Les dérivées partielles d'une fonction réelle de plusieurs variables peuvent être approximées en x^k avec une formule de différences finies arrière, centrale ou avant dont les expressions sont respectivement :

$$\frac{\partial f}{\partial x_i}(x^k) \approx \frac{f(x^k) - f(x^k - he_i)}{h}$$
$$\frac{\partial f}{\partial x_i}(x^k) \approx \frac{f(x^k + he_i) - f(x^k - he_i)}{2h}$$
$$\frac{\partial f}{\partial x_i}(x^k) \approx \frac{f(x^k + he_i) - f(x^k)}{h}$$

où e_i est un vecteur de R^n dont toutes les composantes sont nulles sauf la i$^{\text{ième}}$ qui est égale à 1.

La formule de différence avant est préférable car elle évite une évaluation supplémentaire (en $x^k - he_i$), et sur des fonctions suffisamment régulières les approximations par différence avant et arrière bornent l'erreur maximale commise par la même valeur [24]. Dans les techniques usuelles de calcul, la valeur "du pas" *h* est choisie par avance et fixée pour les calculs ultérieurs. Dans le cadre d'un algorithme d'optimisation cette méthode ne convient pas car les gradients doivent être évalués en des points très différents et les fonctions objectif et contraintes peuvent avoir des comportements numériques très différents. *Gill* et *Murray* [24] proposent un algorithme de calcul d'un pas "optimal" pour chaque variable donnant une bonne approximation des dérivées partielles d'une fonction suffisamment régulière quel que soit le point d'évaluation.

C'est la technique que nous avons utilisée pour calculer les gradients par différences finies. Nous verrons qu'elle donne de bons résultats (en termes de précision) mais qu'elle est excessivement "gourmande" en évaluations de fonction. Puisque le calcul d'une dérivée partielle peut nécessiter jusqu'à 3 évaluations de fonctions; et donc en tout 3*n* évaluations pour déterminer le gradient complet d'une fonction.

2.4 Codification des équations du problème d'optimisation

Dans les problèmes de conception optimale, les variables du problème sont toujours soumises à des contraintes "bornes", soit explicitement par des conditions fonctionnelles limites soit implicitement par des limites physiques de la modélisation du problème de conception (par exemple les variables définissant la géométrie d'un élément sont toujours positives). Donc on peut toujours exprimer deux vecteurs u et v de R^n tels que :

$$u \leq x \leq v$$

Ces bornes sur les variables peuvent s'écrire comme $2n$ fonctions contraintes, de la façon suivante :

$$\begin{cases} c_i(x) = u_i - x_i \leq 0 \\ c_{i+n}(x) = x_i - v_i \leq 0 \end{cases} \text{ pour } i = 1..n$$

et être introduites dans la formulation générale du problème d'optimisation. Pour des raisons pratiques, et aussi pour permettre éventuellement de faire varier la valeur de ces bornes indépendamment des données du problème celles ci sont explicitement fournies au code de calcul. L'intégration de ces bornes, sous la forme de contraintes inégalités, dans la formulation du lagrangien augmenté est faite automatiquement par le programme.

Les fonctions contraintes d'un problème d'optimisation peuvent se formuler de différentes façons. Par exemple, dans le cas du problème du ressort de pompe hydraulique, les trois formulations (5), (6) et (7) de la fonction contrainte c_2 expriment la même condition fonctionnelle limite sur les variables du problème et donc la même frontière du domaine des solutions (les variables D et N sont supposées positives et non nulles). Par conséquent ces trois formulations équivalentes "au sens mathématique" donnent la même solution optimale.

$$c_2(D,N) = D^3 N - \frac{Gd^4}{8k_m} \leq 0 \tag{5}$$

$$c_2(D,N) = \left(\frac{8k_m}{Gd^4}\right) D^3 N - 1 \leq 0 \tag{6}$$

$$c_2(D,N) = 1 - \left(\frac{Gd^4}{8k_m D^3 N}\right) \leq 0 \tag{7}$$

Le calcul des gradients respectifs correspondant à ces formulations donne :

$$\nabla c_2(D,N) = \left\{3D^2N, D^3\right\}^T$$

$$\nabla c_2(D,N) = \left\{\frac{24k_m}{Gd^4}D^2N, \frac{8k_m}{Gd^4}D^3\right\}^T$$

$$\nabla c_2(D,N) = \left\{\frac{3Gd^4}{8k_mD^4N}, \frac{Gd^4}{8k_mD^3N^2}\right\}^T$$

On constate la différence importante qu'il existe entre ces 3 expressions, en effet pour une même valeur des variables, les normes de ces gradients sont très différentes.

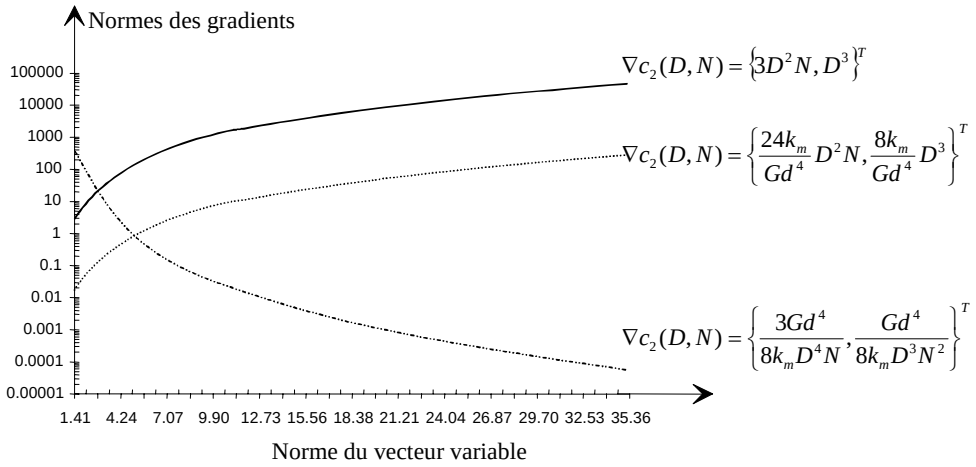


Figure 6.2 : Evolution des normes des gradients (courbes établies avec les données du chapitre 4).

On remarque sur la figure 6.2 que l'expression (6) de cette fonction contrainte donne un gradient de norme "moyenne" par rapport aux autres formulations. Numériquement c'est cette formulation qui donne les meilleurs résultats. Ce point est très important, car nous allons montrer sur des exemples que pour une même formulation initiale, certaines codifications des équations du problème donnent de très mauvais résultats numériques, ce qui met en évidence la nécessité d'un bon conditionnement numérique.

2.5 Exemples d'application, résultats de calculs

Le code de calcul s'appuyant sur cette méthode de résolution permet de déterminer efficacement la solution d'un problème de conception optimale. Afin de tester ses performances nous l'avons appliqué aux problèmes présentés précédemment : le ressort de pompe hydraulique, et l'accouplement à plateaux boulonnés.

Ces problèmes d'optimisation comportant une seule variable discrète, la méthode utilisée pour les résoudre fut d'envisager toutes les valeurs discrètes et de choisir celle qui donnait la meilleure solution. Le calcul de la solution optimale pour chacune d'elles, nécessitait la résolution d'un problème d'optimisation en variables continues. Dans un premier temps nous

allons appliquer la même stratégie sur ces deux exemples. Cela va nous permettre d'évaluer la stabilité des performances de l'algorithme lorsque les données du problème varient. Nous étudierons également l'influence du point de départ choisi pour le calcul, ainsi que celle de l'évolution du coefficient de pénalité.

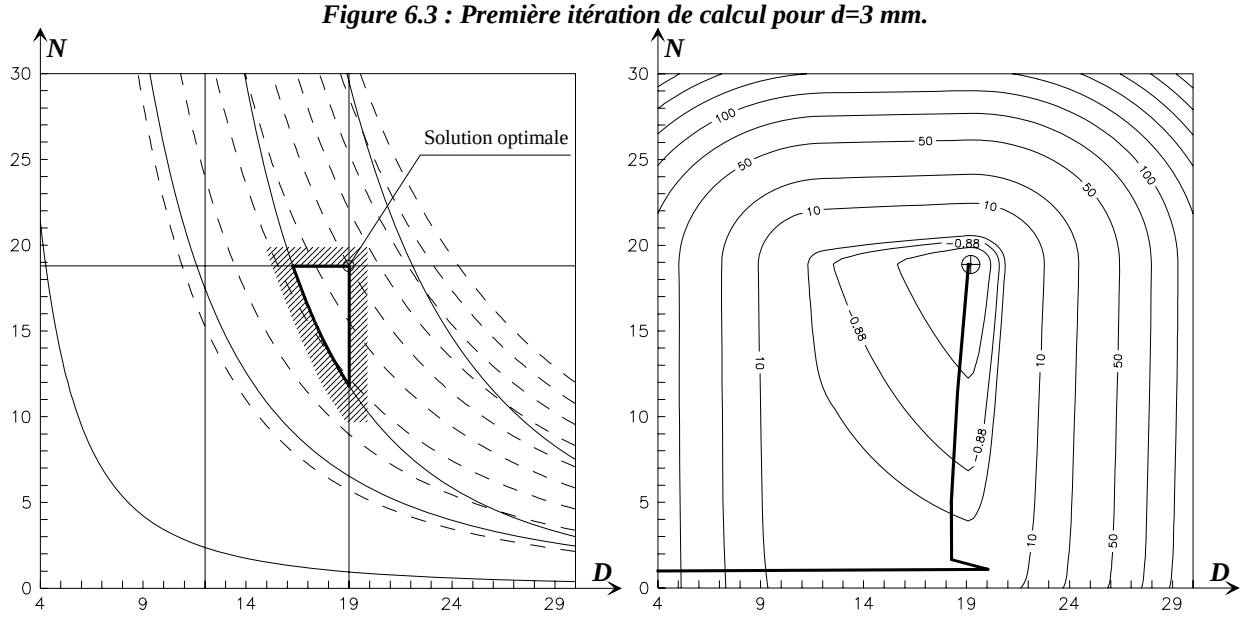
Les expressions de ces deux problèmes d'optimisation ont été définies au chapitre 4. Elles ont été reprises et modifiées en tenant compte des remarques faites précédemment (on trouvera le détail des expressions dans l'annexe 4).

Pour chaque cas de calcul nous préciserons le nombre d'itérations nécessaires (colonnes "Iter"), c'est à dire le nombre de minimisations du lagrangien augmenté, et le nombre d'évaluations de fonction (colonnes "Eval") nécessaires pour obtenir la solution. Lorsque les gradients sont fournis analytiquement au code de calcul, le nombre d'évaluations est identique pour les fonctions objectif et contraintes. Dans le cas de gradients calculés par différences finies, ce nombre peut varier d'une fonction à l'autre. En effet le "pas optimal" pour chaque variable, dans le calcul des gradients, dépend de la fonction. (non linéarité, comportement numérique).

2.5.1 Ressort de pompe hydraulique

Ce problème de 2 variables va nous permettre d'illustrer graphiquement la méthode. Les figures 6.3 et 6.4 représentent le domaine des solutions et les contours du lagrangien augmenté pour deux valeurs de diamètre de fil du ressort.

Figure 6.3 : Première itération de calcul pour d=3 mm.

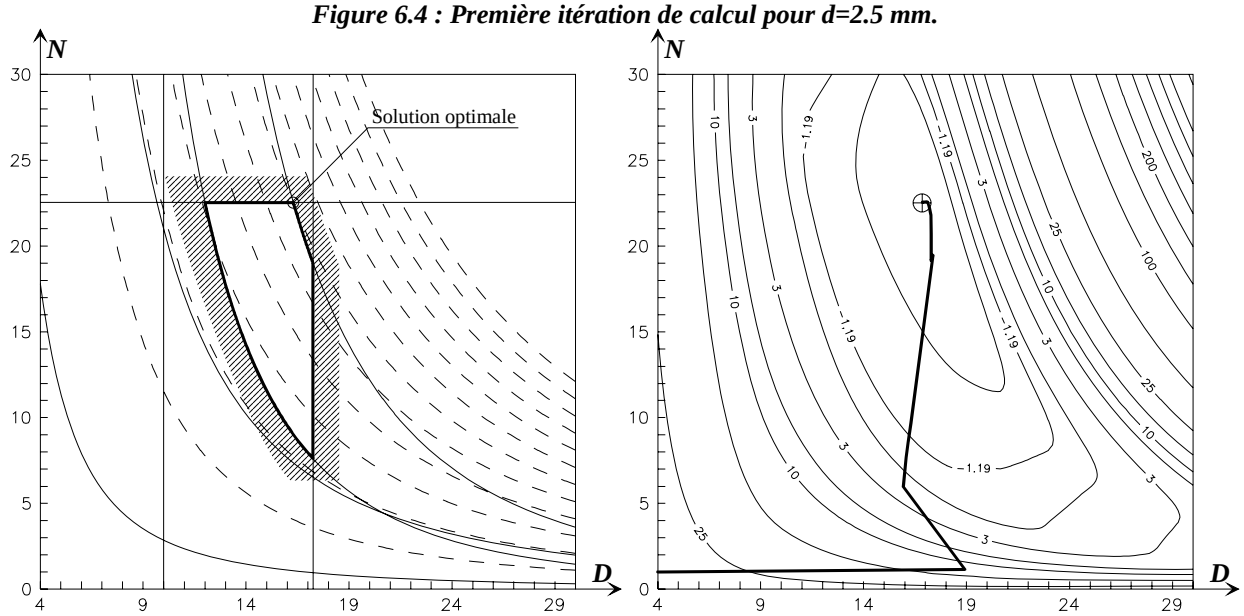


(a) : Domaine des solutions . ***(b) : Contours du lagrangien augmenté pour r=1.***

Les itérations de calculs intermédiaires, pendant l'étape de minimisation du lagrangien augmenté, ont été visualisées sur les deux figures. On constate que dès la première itération le minimum du lagrangien augmenté est proche de la solution du problème. On remarque également que les contours "suivent" la forme du domaine des solutions, même dans le cas d'un

domaine non convexe (figure 6.4). C'est pendant cette première itération que s'effectue la majorité du travail de l'algorithme.

Figure 6.4 : Première itération de calcul pour d=2.5 mm.



(a) : Domaine des solutions. ***(b) : Contours du lagrangien pour r=10.***

Le tableau 6.5 regroupe les solutions obtenues et la valeur du coefficient de sécurité en fatigue pour l'ensemble des diamètres de fils du ressort défini dans le chapitre 4. Les calculs ont été effectués avec deux ensembles de données différentes.

- Les données "standard" sont :

C = 15 mm	D_M = 15 mm	L_M = 62 mm	ε = 0.1 mm
F_{MAXI} = 300 N	k_m = 4 N/mm	k_M = 10 N/mm	f_{MAXI} = 100 Hz
τ_D = 300 MPa	G = 80 000 MPa	ρ = 7.8 10^{-6} Kg/mm^3	

- Les données "modifiées" permettant d'obtenir une solution pour tous les diamètres de fils sont identiques aux précédentes, sauf pour :

$$k_M = 70 \text{ N/mm et } \tau_D = 450 \text{ Mpa}$$

Pour tous les calculs présentés dans le tableau 6.5, le point de départ des itérations (n'appartenant pas au domaine des solutions) et la valeur du coefficient de pénalité r et celle de γsont identiques. Cela permet d'observer l'influence d'une variation des données sur le nombre d'itérations et d'évaluation de fonctions.

Avec les données standard et pour les diamètres 1, 3.5, 4, 4.5, 5 le domaine des solutions est vide. Dans ce cas les calculs se terminent par une erreur au bout d'un nombre non négligeable d'itérations. Dans le cas général d'un problème de conception, il est impossible de savoir si le domaine des solutions est vide avant de commencer un calcul.

Dans le cas des données modifiées, on remarque que le nombre d'itérations et d'évaluations varie très peu. Seul le cas d=2.5 mm, demande plus de calculs à cause de la non convexité du domaine des solutions.

	Point de départ : D = 4 , N = 1 r = 1 ; γ = 5									
	Données standards					Données modifiées				
d	D	N	α_F	Iter	Eval	D	N	α_F	Iter	Eval
1	*	*	*	10	192	4.000	39.063	1.117	4	120
1.5	6.000	37.576	1.074	3	74	6.000	37.576	1.612	3	78
2	8.681	28.182	1.280	3	79	8.681	28.182	1.920	3	79
2.5	16.301	22.545	3.058	4	113	16.301	22.545	4.588	5	135
3	19.000	18.788	2.874	3	67	19.000	18.788	4.311	3	65
3.5	*	*	*	10	151	18.500	16.104	2.927	3	69
4	*	*	*	10	142	18.000	14.091	2.074	3	65
4.5	*	*	*	10	151	18.000	12.525	1.612	3	69
5	*	*	*	10	145	20.000	11.273	1.612	3	74

Tableau 6.5 : Influence des données.

En reprenant les données standards, et pour les cas de calculs le plus court (d=3 mm) et le plus long (d=2.5 mm) nous avons observé l'influence d'une modification du point de départ des itérations. La stratégie de pénalisation (choix initial de *r* et de γ) est identique à la précédente. Les points choisis sont situés aux "quatre coins" du domaine des valeurs admissibles défini par les bornes sur les variables et n'appartiennent pas au domaine des solutions défini par les fonctions contraintes du problème.

	d=2.5		d=3	
Point de départ (D,N)	Iter	Eval	Iter	Eval
(4,1)	4	113	3	67
(4,56.36)	4	108	3	67
(80,1)	4	119	3	73
(80,56.36))	4	103	3	60

Tableau 6.6 : Influence du point de départ.

Le tableau 6.6 montre que le point de départ des itérations n'a pas d'influence sur la longueur des calculs, on constate juste quelques faibles variations dans le nombre d'évaluation de fonction.

	d=2.5	d=3

	$\gamma = 1.5$		$\gamma = 5$		$\gamma = 10$		$\gamma = 1.5$		$\gamma = 5$		$\gamma = 10$	
	Iter	Eval	Iter	Eval	Iter	Eval	Iter	Eval	Iter	Eval	Iter	Eval
r=1	7	147	4	113	4	128	3	62	3	67	3	65
r=5	4	99	3	94	3	113	3	46	2	47	2	47
r=10	4	114	3	98	3	112	2	52	2	53	2	58
r=50	3	97	3	110	2	95	2	68	2	76	2	69
r=100	3	114	2	111	2	108	2	86	2	94	2	87

Tableau 6.7 : Influence de l'évolution du coefficient de pénalité.

Les résultats regroupés dans le tableau 6.7 montrent l'influence de la stratégie de pénalisation. Ces chiffres confirment bien le fait que la vitesse de convergence dépend de la valeur du coefficient de pénalité (cases ombrées dans le tableau). On constate également qu'il existe une stratégie optimale : celle qui réalise le meilleur compromis entre le petit nombre d'itérations et le coût de calcul (nombre d'évaluation) le plus faible (cases entourées).

2.5.2 Accouplement à plateaux boulonnés

La même démarche a été appliquée sur le problème d'accouplement à plateaux. La modélisation utilisée et les données sont celles définies dans le chapitre 4. La fonction objectif est ici une fonction objectif multicritères dans laquelle on minimise le nombre de boulons, le rayon d'encombrement et le moment transmis. L'importance relative de chaque critère est donnée par la valeur des coefficients β_i.

Le tableau 6.8 regroupe les résultats obtenus pour plusieurs diamètres de boulons et pour trois combinaisons de valeur des coefficients associés à chaque critère. Dans la première, le nombre minimal de boulons est privilégié, dans la seconde tous les critères ont la même importance et la dernière favorise le rayon d'encombrement minimal.

On notera que la valeur optimale de *M* ne correspond pas aux données initiales du problème. Cela vient du fait que les données ont été normées pour éviter les trop grandes dispersions sur la valeur des variables. En effet avec les données initiales la valeur optimale de *M* est de 4.10^6 mm.N alors que celle de R_B est de l'ordre de 100 mm environ.

Les résultats du tableau 6.8 ont été obtenus avec le même point de départ (hors du domaine des solutions) et la même stratégie de pénalisation, sauf pour un calcul (cases ombrées dans le tableau) pour lequel la valeur initiale de *r* a du être modifiée pour obtenir une solution. La difficulté pour ce problème fut de trouver la bonne stratégie de pénalisation permettant d'obtenir une solution pour chaque valeur de diamètre de vis. Les résultats de ces calculs sont le reflet de cette difficulté, on constate des variations importantes dans le nombre d'itérations et d'évaluations de fonction. La seule explication que l'on puisse fournir, est celle d'une variation importante du conditionnement numérique du problème en fonction des données.

	Point de départ N = 1; Rb=10; M=10 ; r = 10 ; γ = 1.5								
	d	6	8	10	12	14	16	20	24
$\beta_1 = 10$ $\beta_2 = 1$ $\beta_3 = 1$	N	8.620	8.000	8.000	8.000	8.000	8.000	8.000	8.000
	Rb	538.75	316.25	198.54	136.16	99.133	71.986	71.000	75..000
	M	40	40	40	40	40	40	40	40
	Iter	7	6	5	5	4	4	2	2
	Eval	303	196	177	136	136	182	135	108
$\beta_1 = 1$ $\beta_2 = 1$ $\beta_3 = 1$	N	27.259	20.120	15.942	13.202	11.265	8.594	8.000	8.000
	Rb	170.37	125.75	99.635	82.512	70.404	67.000	71.000	75.000
	M	40	40	40	40	40	40	40	40
	Iter	5	5	4	4	4	4	2	2
	Eval	176	205	136	131	175	197	143	111
$\beta_1 = 1$ $\beta_2 = 10$ $\beta_3 = 1$	N	44.859	29.313	20.608	16.071	12.109	8.595	8.000	8.000
	Rb	103.52	86.309	77.076	67.781	65.499	67.000	71.000	75.000
	M	40	40	40	40	40	40	40	40
	Iter	7	7	7	7	8	6	4	4
	Eval	224	220	157	157	222	194	149	228

Tableau 6.8 : Influence des données

Le tableau 6.9 regroupe les résultats relatifs au test sur la variation du point de départ des itérations.

	d = 14 $\beta_1 = 1; \beta_2 = 10; \beta_3 = 1$		d = 6 $\beta_1 = 10; \beta_2 = 1; \beta_3 = 1$	
Point de départ (N,Rb,M)	Iter	Eval	Iter	Eval
(1,10,10)	8	222	7	303
(100,10,10)	8	234	8	263
(1,1000,10)	8	253	7	297
(100,1000,10)	8	246	7	292
(1,10,100)	8	217	7	322
(100,10,100)	8	218	7	248
(1,1000,100)	8	222	7	252
(100,1000,100)	8	200	7	261

Tableau 6.9 : Influence du point de départ.

Nous avons choisi pour ce test les cas de calculs précédents qui semblaient les plus "difficiles". Les chiffres montrent que même sur un problème délicat, une fois la stratégie de pénalisation correctement établie, la méthode est insensible au point de départ des itérations.

Le dernier test effectué sur ce problème concerne l'influence du coefficient de pénalité et de son coefficient d'augmentation. Les calculs ont été réalisés pour les cas de données du test précédent. Les résultats présentés dans le tableau 6.10 confirment les tendances observées sur le problème du ressort. On notera "une curiosité" dans les calculs, que nous n'expliquons pas : le fait que nous n'ayons pas obtenu de solutions pour d = 14 et r = 10.

	d = 14 $\beta_1 = 1; \beta_2 = 10; \beta_3 = 1$						d = 6 $\beta_1 = 10; \beta_2 = 1; \beta_3 = 1$					
	$\gamma = 1.5$		$\gamma = 5$		$\gamma = 10$		$\gamma = 1.5$		$\gamma = 5$		$\gamma = 10$	
	Iter	Eval	Iter	Eval	Iter	Eval	Iter	Eval	Iter	Eval	Iter	Eval
r=1	12	226	6	170	5	176	12	368	6	261	5	236
r=5	8	222	5	182	4	200	8	260	5	191	4	199
r=10	*	*	*	*	*	*	7	303	4	237	4	242
r=50	4	146	3	164	3	171	5	189	3	161	3	166
r=100	3	152	3	162	3	162	4	244	3	247	3	241

Tableau 6.10 : Influence de l'évolution du coefficient de pénalité.

2.5.4 Optimisation de forme : poutre encastrée

Afin de comparer les résultats obtenus avec ce code de calcul avec ceux donnés par d'autres codes, nous l'avons appliqué sur un exemple d'optimisation de structure : la minimisation de la masse (donc du volume) d'une poutre encastrée homogène constituée de tronçons de section variable (figure 6.11).

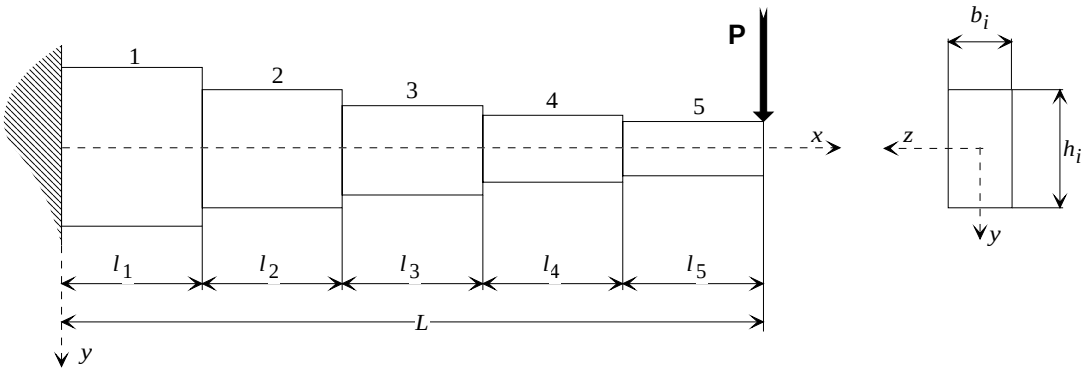


Figure 6.11 : Géométrie et chargement de la poutre à tronçons de section variable.

Ce problème, présenté par *Vanderplaats* [67], comporte $2N$ variables et $2N+1$ fonctions contraintes inégalités, N désignant le nombre de tronçons constituant la poutre (détail des expressions dans l'annexe 4). Les variables sont la hauteur et la largeur de chaque tronçon, les fonctions contraintes sont données par des conditions de résistance sur chaque tronçon, ainsi que par une condition de flèche maximale admissible en bout de poutre.

Les équations de ce problème ont été écrites de manière à permettre un nombre quelconque de tronçons, ce qui va nous permettre d'observer le comportement de notre code de calcul lorsque la taille du problème à résoudre augmente.

Le tableau 6.12 regroupe les résultats de calculs obtenus pour des poutres composées de 2 à 25 tronçons, soit des problèmes d'optimisation de 5 à 50 variables et 6 à 51 fonctions contraintes. On remarque que le nombre d'itérations augmente très peu lorsque la taille du problème augmente, par contre le coût des calculs (nombre d'évaluations) croît fortement avec la taille du problème. Les courbes de la figure 6.13 montrent que cette croissance est d'allure linéaire en fonction du nombre de variables.

	Point de départ A b_i=1; h_i=5 (cm)		Point de départ B b_i=5 ; h_i= 40 (cm)		Point de départ C b_i=100 ; h_i=500 (cm)				
Nbre de tronçons/ Nbre de variables	Iter	Eval	Iter	Eval	Iter	Eval	Volume (cm^3)	Coefficient de pénalité	γ
2/4	3	923	3	920	3	780	72958	50000	10
5/10	3	1267	3	860	3	923	61909	10000	10
10/20	4	2351	4	1565	4	1495	57939	1000	10
15/30	4	2925	4	1698	4	1747	56567	750	10
20/40	4	3244	4	2123	4	1808	55870	500	10
25/50	4	3204	4	2417	4	2085	55445	250	10

Tableau 6.12 : Résultats poutre à tronçons.

La valeur initiale du coefficient de pénalité a été ajustée en fonction du nombre de variables et de fonctions contraintes. En fait cette valeur n'est liée qu'au nombre de contraintes du problème. Elle diminue quand la taille du problème augmente pour compenser la forte augmentation du terme $\sum_{j=1}^{m}\left(\Psi_j{}^2(x)\right)$ dans l'expression du lagrangien augmenté.

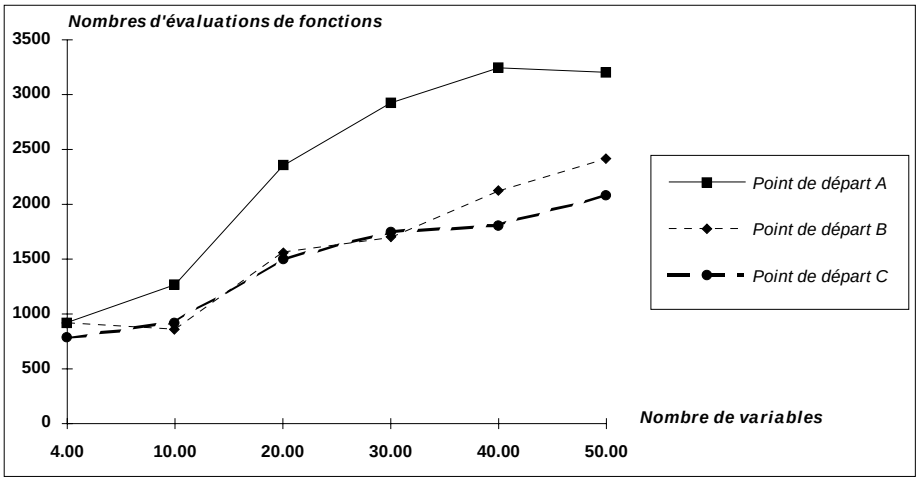


Figure 6.13 : Croissance du coût de calcul en fonction de la taille du problème.

Le tableau 6.14 détaille les résultats obtenus pour le cas d'une poutre à 5 tronçons, et ceux donnés par *Vanderplaats*. Notre calcul a nécessité moins d'itérations mais plus d'évaluation de fonctions que celui de *Vanderplaats*. La différence est sans aucun doute liée à la valeur des coefficients de pénalité que nous avons utilisée.

La solution que nous avons obtenue est meilleure mais certaines contraintes sont légèrement "violées", comme en témoigne la valeur de $\Sigma = \sum_j \text{Max}\left\{c_j(x^*), 0\right\}$.

N° tronçon	b (cm)	h (cm)	b (cm)	h (cm)
1	3.00	59.77	2.99	59.83
2	2.76	55.42	2.77	55.54
3	2.52	50.47	2.52	50.47
4	2.27	44.99	2.20	44.09
5	2.17	42.94	1.75	34.99
	Iter 8	Eval 533	Iter 3	Eval 860
Volume (cm^3)	65678		61909	
Σ	0		$5.11\ 10^{-4}$	

Tableau 6.14 : Poutre à tronçons : comparaison des résultats.

2.5.5 Influence des calculs de gradients par différences finies

Les tableaux 6.15 et 6.16 reprennent des cas de calculs précédents pour les problèmes du ressort et de l'accouplement, tous les gradients ayant été calculés par différences finies. La

colonne "Grad" précise le nombre d'évaluations de gradients et les colonnes "Eval obj", "Eval C1", le nombre d'évaluations pour la fonction objectif et les fonctions contraintes.

	Point de départ : D = 4 , N = 1 r = 1 ; γ = 5									
	Données standards									
d	D	N	α_F	Iter	Grad	Eval Obj	Eval C1	Eval C2	Eval C3	Eval C4
2.5	16.301	22.545	3.058	4	113	1445	1379	1393	1373	1365
3	19.000	18.788	2.874	3	67	859	809	775	836	809

Tableau 6.15 : Ressort de pompe hydraulique : calcul des gradients par différences finies.

On constate que le coût du calcul, différent pour chaque fonction, est fortement augmenté par les différences finies mais que la solution obtenue est identique. On notera l'avantage procuré par l'intégration automatique des bornes sur les variables comme fonctions contraintes par le code de calcul. En effet les gradients de ces contraintes sont explicitement codés par le programme puisque le gradient de :

$$c_i(x) = u_i - x_i \leq 0$$

s'écrit simplement :

$$\nabla c_i(x) = \{0,...,-1,.....0\}^T \text{ (le -1 est en i}^{\text{ième}} \text{ position).}$$

	Point de départ N = 1; Rb=10; M=10	
	r=10 / γ = 1.5 $\beta_1 = 10 ; \beta_2 = 1 ; \beta_3 = 1$	r=5 / γ = 1.5 $\beta_1 = 1 ; \beta_2 = 10 ; \beta_3 = 1$
d	6	14
N	8.620	12.109
Rb	538.75	65.499
M	40	40
Iter	7	8
Grad	302	226
Eval Obj	5402	3624
Eval C1	5470	4182
Eval C2	6028	4018
Eval C3	6020	4140

Tableau 6.16 : accouplement à plateaux : calcul des gradients par différences finies.

2.6 Conditionnement numérique

2.6.1 Principe

L'idée d'un conditionnement numérique, est qu'à partir d'un problème réel exprimé sans précaution particulière, on puisse déterminer certains "coefficients de conditionnement" pour ramener l'expression du problème à un modèle numériquement acceptable, c'est à dire un problème dans lequel les différentes les variables et les fonctions contraintes ont toutes le même ordre de grandeur [61].

Un choix simple pour ces coefficients de conditionnement est de multiplier chaque fonction contrainte par un coefficient non nul q_j, et d'effectuer "le changement variable" suivant :

$$\tilde{x} = Z.x \qquad (8)$$

où Z est une matrice carrée régulière de R^n [67].

De sorte que le problème initial :

$$(P_c) \begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \\ c_j(x) = 0 \quad j = m+1...m+l \end{cases}$$

est remplacé par :

$$(\tilde{P}_c) \begin{cases} \textit{Minimiser } f(\tilde{x}) \\ \textit{Sous les fonctions contraintes} \\ q_j c_j(\tilde{x}) \le 0 \quad j = 1...m \\ q_j c_j(\tilde{x}) = 0 \quad j = m+1...m+l \end{cases}$$

L'algorithme d'optimisation est alors appliqué sur le problème "conditionné" $(\tilde{P}_c)$. Les gradients des différentes fonctions sont déterminés à partir de ceux des fonctions "originales" de la manière suivante :

$$\begin{aligned} \tilde{\nabla} f(\tilde{x}) &= Z^{-1} \nabla f(x) \\ \tilde{\nabla} c_j(\tilde{x}) &= q_j Z^{-1} \nabla c_j(x) \end{aligned} \qquad (9)$$

Pour faciliter les calculs, le plus souvent on choisit pour Z une matrice diagonale. Dans notre algorithme le problème est "conditionné" avant l'opération de minimisation avec la transformation (8), pendant le processus de minimisation les gradients "conditionnés" sont calculés avec les relations (9), puis à la fin de l'étape de minimisation les variables sont ramenées à leurs valeurs "non conditionnées" grâce à la transformation :

$$x = Z^{-1}.\tilde{x}$$

Il n'existe pas de règle systématiquement meilleure concernant le calcul de ces coefficients de conditionnement. Il doit être assez simple pour éviter que l'augmentation du coût de calcul dû au conditionnement ne soit pas supérieure au coût de calcul de la solution du problème non conditionné.

Pour notre part, nous avons effectué des tests avec les trois types de calcul suivants :

1) Au point de départ x^0, on calcule q_j avec :

$$q_j = \frac{1}{\left(1+\left|c_j(x^0)\right|\right)} \quad j = 1..m+l, \text{ et } Z_{ii}=1 \text{ pour } i = 1..n$$

2) Au point de départ x^0, on calcule q_j avec :

$$q_j = \frac{1+\left\|\nabla f(x^0)\right\|}{Max\left\{\left\|\nabla c_j(x^0)\right\|, \varepsilon\right\}} \text{ , } \varepsilon \text{ petit et positif, } j = 1..m+l, \text{ et } Z_{ii}=1 \text{ pour } i = 1..n$$

3) Au point de départ x^0, on calcule q_j et Z_{ii} grâce à la méthode préconisée par *Root* et *Ragsdell* [61], dans laquelle les *n* coefficients Z_{ii} sont la moyenne des dérivées partielles non nulles des fonctions contraintes, les coefficients q_j étant calculés comme le rapport entre la moyenne des composantes non nulles du gradient de la fonction objectif et la moyenne des composantes non nulles du gradient de la fonction contrainte $c_j(x)$.

2.6.2 Apport d'un conditionnement numérique

Pour vérifier l'efficacité de ces trois types de conditionnement nous les avons testés sur les problèmes du ressort de pompe hydraulique et de l'accouplement codifié comme ils ont été définis dans le chapitre 4.

	Point de départ : D = 4 , N = 1 r = 1 ; γ= 5 / d=2.5 mm							
	Sans conditionnement		Conditionnement (1)		Conditionnement (2)		Conditionnement (3)	
r	Iter	Eval	Iter	Eval	Iter	Eval	Iter	Eval
0.01	*	*	*	*	4	193	*	*
0.1	*	*	*	*	2	170	*	*
1	*	*	*	*	*	*	*	*
5	*	*	*	*	*	*	*	*
10	*	*	10	334	*	*	*	*
100		*	6	225	*	*	*	*

Tableau 6.17 : Problème du ressort : apport d'un conditionnement numérique.

Dans le cas de l'accouplement, aucune solution n'a pu être obtenue quel que soit le conditionnement numérique appliqué et quel que soit la stratégie de pénalisation essayée.

En ce qui concerne le problème du ressort, les résultats obtenus sont présentés dans le tableau 6.17.

On constate l'efficacité très moyenne des trois conditionnements testés sur ce problème. Seul le conditionnement 2) semble apporter des améliorations intéressantes pour des valeurs de coefficients de pénalité assez faibles.

Cette conclusion ne doit pas être systématique, car dans le cas du problème de la poutre encastrée, une amélioration notable des performances est possible (tableau 6.18).

Point de départ B : b_i =5 ; h_i = 40 (cm) / r=1000 / γ=10 Nbre de tronçons/Nbre de variables : 5/10					
Conditionnement (1)		Conditionnement (2)		Conditionnement (3)	
Iter	Eval	Iter	Eval	Iter	Eval
1	1293	1	629	2	2469

Tableau 6.18 : Poutre encastrée : apport du conditionnement numérique.

Ces résultats montrent que le conditionnement numérique d'un problème d'optimisation ne permet pas systématiquement de résoudre des problèmes mal formulés et qu'il est bien plus efficace de modifier la formulation, que de chercher une formule miracle de conditionnement. Toutefois l'expérience acquise lors de la résolution des problèmes présentés dans ce travail nous permet de tirer quelques conclusions concernant la modélisation des problèmes de conception optimale.

C'est de l'écriture des fonctions contraintes que dépend en grande partie le "bon conditionnement" du problème. Il est important d'obtenir un ensemble de fonctions contraintes dont les normes des gradients soient du même ordre de grandeur. On peut souvent y parvenir en divisant les fonctions contraintes par les termes ne dépendant pas des variables. On obtient ainsi des valeurs proches de 1 pour des points situés près de la frontière du domaine des solutions.

Les problèmes de conception optimale font intervenir des grandeurs d'ordre variées : des dimensions géométriques d'éléments (quelques dizaines voire quelques centaines de millimètres), des efforts (quelques milliers de Newton), des déplacements (quelques dixièmes de millimètres), etc. ... Il faudra alors éviter les trop grandes dispersions dans les ordres de grandeur des variables par exemple en choisissant convenablement les unités.

2.7 Algorithme de lagrangien augmenté : conclusion

La solution d'un problème d'optimisation est rarement obtenue dès le premier calcul. Il est indispensable de mettre au point une stratégie de pénalisation spécifique au problème. Cette mise au point ne nécessite généralement que quelques essais. Les tendances extrêmes sont :

- Une faible valeur de départ pour le coefficient de pénalité (entre 1 et 5) et un coefficient d'augmentation assez fort (entre 2 et 10).
- Une valeur de départ importante (entre 10 et 100) et un coefficient d'augmentation modéré (entre 1.1 et 1.5).

Ces valeurs sont données à titre indicatif et dépendent du problème à résoudre. Une fois la stratégie de pénalisation établie, la méthode se révèle être extrêmement fiable, le point de départ des itérations n'ayant pas beaucoup d'incidence sur les calculs.

3 Algorithme de séparation et évaluation

La méthode que nous venons de décrire pourra être utilisée comme principe d'évaluation dans le cadre d'une méthode de séparation et évaluation. L'algorithme que nous envisageons d'utiliser est basé sur le principe défini dans le chapitre 5 § 2.3.1 et s'applique à la résolution du problème non linéaire en variables mixtes posé de la façon suivante :

$$(P_{u,v}) \begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \\ c_j(x) = 0 \quad j = m+1...m+l \\ u \le x \le v \\ u, v, x \in R^{(n-d)} \times E^d \end{cases}$$

Avec : $f, c_j \ : \ x \in R^{(n-d)} \times E^d \longrightarrow f(x), c_j(x) \in R \quad j = m+1..m+l$

$D_{u,v} = \left\{ x \in R^{(n-d)} \times E^d \ / \ c_j(x) \le 0 \ j = 1...m; c_j(x) = 0 \ j = m+1...m+l; u \le x \le v \right\}$

Considérons également le problème suivant en **variables continues** :

$$(P'_{u,v}) \begin{cases} \textit{Minimiser } f(x) \\ \textit{Sous les fonctions contraintes} \\ c_j(x) \le 0 \quad j = 1...m \\ c_j(x) = 0 \quad j = m+1...m+l \\ u \le x \le v \\ u, v \in R^{(n-d)} \times E^d \quad x \in R^n \end{cases}$$

Avec : $f, c_j \ : \ x \in R^n \longrightarrow f(x), c_j(x) \in R \quad j = m+1..m+l$

$D'_{u,v} = \left\{ x \in R^n \ / \ c_j(x) \le 0 \ j = 1...m; c_j(x) = 0 \ j = m+1...m+l; u \le x \le v \right\}$

Le principe d'évaluation défini dans le chapitre 5 § 2.3.2 pour les problèmes linéaires en nombres entiers est toujours valable pour un problème non linéaire en variables mixtes. En effet puisque :

$$D_{u,v} \subset D'_{u,v}$$

on a toujours :

$$Min\left\{(P'_{u,v})\right\} \le Min\left\{(P_{u,v})\right\}$$

Donc la valeur optimale de la fonction objectif de $(P'_{u,v})$ constitue bien une évaluation pour l'ensemble $D_{u,v}$. Et lorsque la solution correspondante respecte les restrictions dues aux variables mixtes, cette évaluation est exacte ($(P'_{u,v})$et $(P_{u,v})$ ont la même valeur optimale).

Rappelons le principe de séparation de l'ensemble $D_{u,v}$:

Soit $\bar{x} = \left\{\bar{x}c, \bar{x}d\right\}^T$ la solution optimale de $(P'_{u,v})$ obtenue pour certaines valeurs de *u* et *v*. La séparation de $D_{u,v}$ s'effectuera suivant la composante $\bar{x}d_r$ de $\bar{x} = \left\{\bar{x}c, \bar{x}d\right\}^T$ qui **ne sera ni entière, ni discrète**.

On définira alors z_r^l et z_r^u comme étant respectivement :

- Les entiers inférieurs et supérieurs les plus proches de $\bar{x}d_r$ si $\bar{x}d_r$ est une variable entière.
- Les valeurs discrètes inférieures et supérieures les plus proches de $\bar{x}d_r$ si $\bar{x}d_r$ est une variable discrète.

Ce qui permet de "construire" deux nouveaux vecteurs bornes u' et v' tel que :

$$u'_i = \begin{cases} z_r^u & sii = r \\ u_i & sii \neq r \end{cases} \text{ et } v'_i = \begin{cases} z_r^l & sii = r \\ v_i & sii \neq r \end{cases}$$

et donc de définir les deux sous ensembles de $D_{u,v}$: $D_{u',v}$ et $D_{u,v'}$.

Pour illustrer la méthode considérons l'exemple simple ci-dessous, d'un problème non linéaire de deux variables entières.

$$\begin{cases} Minimiser \quad f(xd_1, xd_2) = -xd_1 - 1.8xd_2 \\ Sous\ les\ fonctions\ contraintes : \\ c_1(xd_1, xd_2) = xd_1{}^2 + (xd_2 + 6)^2 - 85 \leq 0 \\ c_2(xd_1, xd_2) = 1 - xd_1{}^2 \leq 0 \\ c_3(xd_1, xd_2) = -xd_2{}^2 \leq 0 \end{cases}$$

dont on donne la représentation graphique sur la figure 6.19.

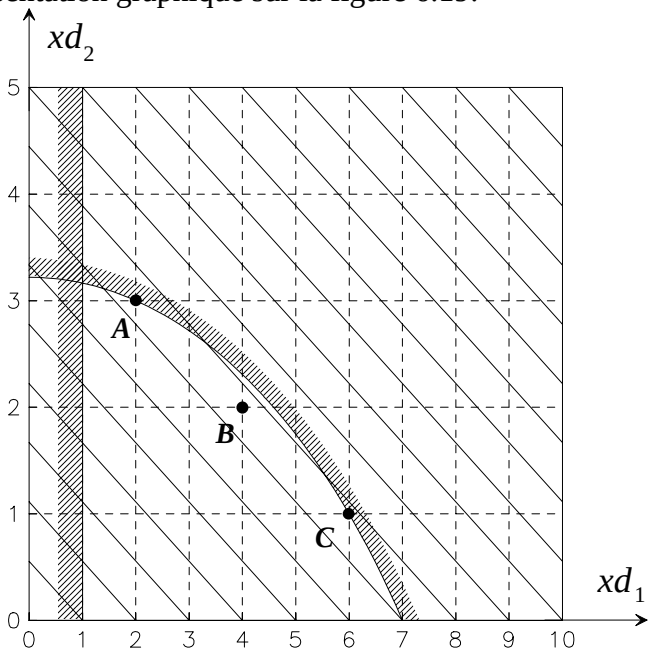


Figure 6.19 : Exemple de problème en variable entière.

La figure 6.20 explicite le concept d'arborescence associée à la décomposition progressive de l'ensemble des solutions de cet exemple. On trouve au sommet de l'arborescence la solution en variable continue du problème. La première séparation s'effectue suivant la variable xd_1 (la plus fractionnaire), ce qui permet de créer deux sous ensembles associés chacun à un sommet. La résolution des deux problèmes non linéaires associés à ces sous ensembles (sommets) donne les évaluations -8.1519 et -8.1427. En choisissant de séparer le sommet

d'évaluation -8.1519 (règle de largeur d'abord) on obtient deux sommets avec des solutions entières correspondant aux points ***A*** et ***B*** de la figure 6.19. La séparation du sommet restant donne finalement la solution optimale entière correspondant au point ***C***. L'ensemble des sommets pendants ayant une solution entière, on peut arrêter le processus et choisir la meilleure solution (le point ***C*** correspond au sommet d'évaluation -7.8).

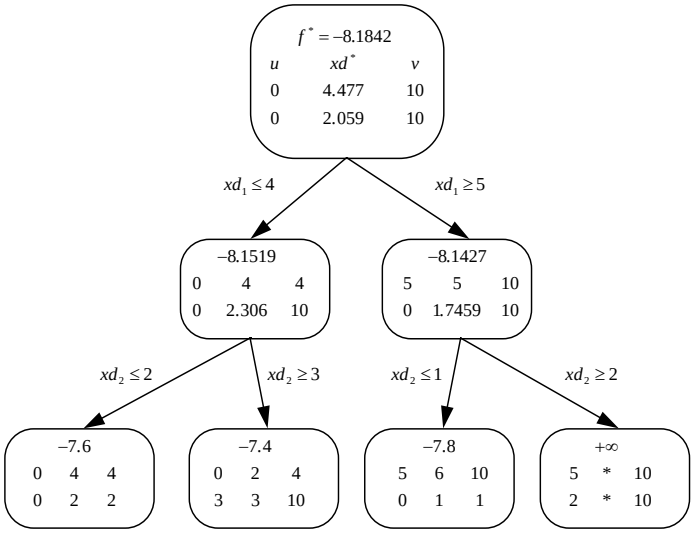


Figure 6.20 : Arborescence associée au problème précédent.

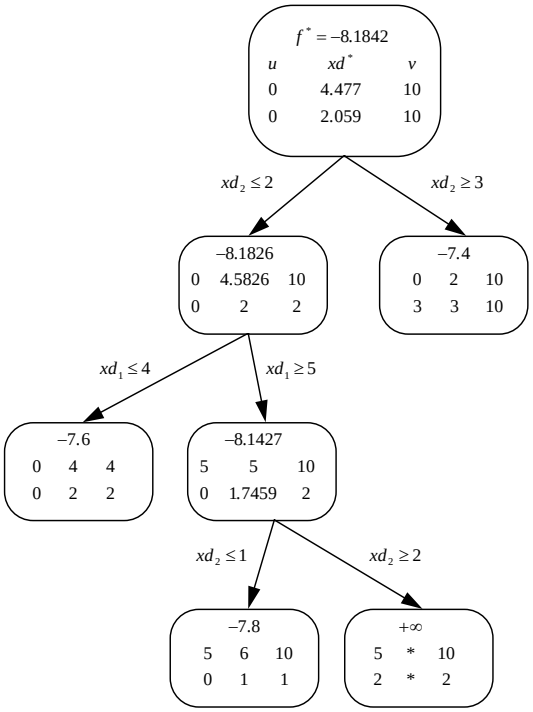


Figure 6.21 : Arborescence de l'exemple avec un nouvel ordre de séparation.

Lorsque les bornes *u* et *v* sont telles que le sous ensemble est vide $D_{u,v}$, on lui associera une évaluation infinie, de sorte que ce sommet ne sera jamais séparé. Dans ces deux cas

d'arborescence la première solution entière trouvée (de valeur -7.4) n'a permis aucune "stérilisation" de sommets.

3.1 Principe : stockage de l'arborescence

Dans l'exemple que nous venons de présenter, l'arborescence à été entièrement développée pour deux cas de séparation. Dans le cadre d'un algorithme, la progression des calculs "suivra" un certain cheminement dans cette arborescence. De sorte que, en fonction de l'ordre dans le choix des sommets et des variables à séparer, certains sommets resteront "provisoirement" non séparés. Il est alors absolument nécessaire de converser ces sommets en mémoire, mais il est impossible de connaître le nombre de sommets à conserver. On peut tout au plus connaître la borne supérieure de cette quantité : le nombre de points discrets du domaine des solutions.

On comprend alors l'intérêt de disposer très tôt dans les calculs d'une solution approchée en variables mixtes. En effet lorsque dans le processus de calcul, l'évaluation du sommet courant sera supérieure à cette solution approchée, on pourra éviter le stockage des informations nécessaires à une séparation ultérieure de ce sommet. Lorsque le sommet courant donne une meilleure solution approchée (évaluation inférieure et solution en variables mixtes), on procédera à un cycle de "stérilisation", c'est à dire au retrait de la mémoire de tous les sommets dont l'évaluation est supérieure à cette nouvelle solution.

Les informations minimales à conserver pour chaque sommet sont :

1) La valeur de l'évaluation : c'est à dire la valeur optimale de la fonction objectif.
2) La valeur optimale des variables du problème d'optimisation associées à ce sommet.
3) Les bornes qui définissent le sous ensemble associé à ce sommet.

Soit 3 vecteurs de dimension *n* (nombre total de variables) et un réel pour chaque sommet.

3.2 Algorithme de séparation et évaluation

L'algorithme de notre méthode de séparation et évaluation est présenté sur la figure 6.22. Lors de chaque opération de séparation, deux problèmes non linéaires doivent être résolus. Ces 2 calculs sont effectués avec la méthode utilisant le lagrangien décrite précédemment. On constate ici, l'utilité de traiter de manière indépendante les bornes sur les variables dans le code de calcul.

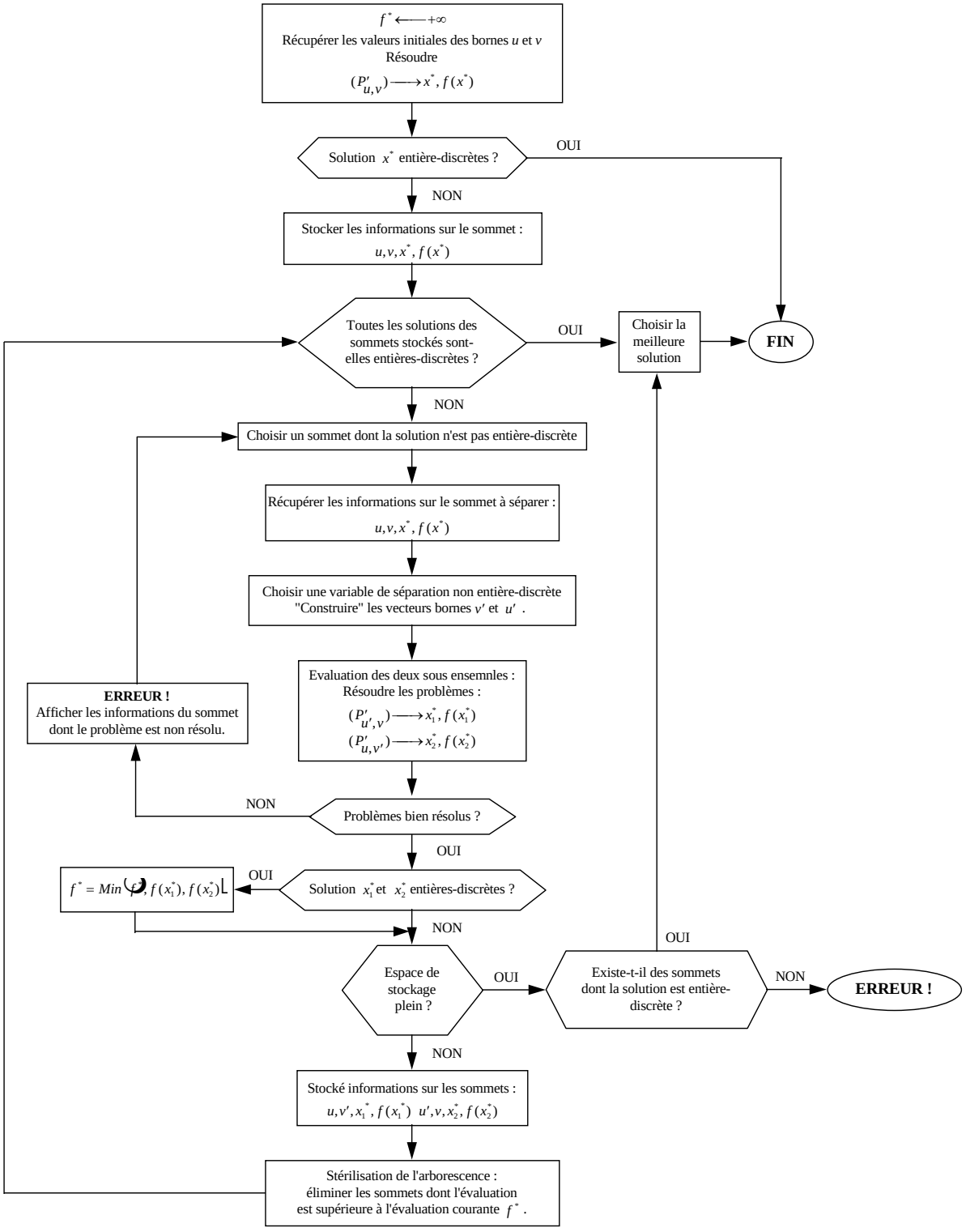


Figure 6.22 : Algorithme de séparation et évaluation.

Lors de chaque nouvelle séparation, il est nécessaire de stocker les informations relatives aux deux sommets couramment créés, on remarque alors que pour l'un des deux sommets, on peut stocker ces informations à la place de celles du sommet que l'on vient de séparer. Donc dans

le pire des cas l'encombrement mémoire augmente d'une allocation de stockage à chaque étape de séparation. De sorte qu'en réservant N_{MAX} allocation de stockage en mémoire, on pourra résoudre jusqu'à $2N_{MAX}$ problèmes non linéaires pour obtenir la solution.

Dans cet organigramme de calcul on ne teste pas la valeur de l'évaluation des sommets avant de les stocker. En fait ces sommets seront éventuellement retirés de l'arborescence lors de l'étape de stérilisation. Cette particularité permet d'obtenir facilement, (en "sautant" cette phase de stérilisation) toutes les solutions en variables mixtes du problème.

Le point délicat de cette méthode est la gestion des éventuelles erreurs lors de la résolution des problèmes en variables continues. En effet il est difficile de savoir si l'erreur est produite par un ensemble des solutions vide ou par un échec des calculs. La solution adoptée dans notre démarche est d'admettre que c'est systématiquement la première éventualité qui produit l'erreur (domaine vide), dans ce cas on affectera à ce sommet une évaluation de très grande valeur. Il sera éliminé lors du prochain cycle de stérilisation.

3.3 Règles de séparation, de progression

Le choix de la variable à séparer à chaque étape du processus est établi en fonction des composantes du gradient de la fonction objectif. L'idée ici est de séparer prioritairement suivant la variable provoquant la plus forte diminution de la fonction objectif. Cet ordre est établi au début de la procédure, avec le gradient de la fonction objectif en x^* qui constitue la solution en variable continue du problème initial. On fixe ainsi l'ordre de séparation pour un cycle complet sur les variables discrètes et entières.

Le choix du sommet à séparer s'effectue suivant une règle de "largeur d'abord" : on sélectionne le sommet dont la valeur optimale de la fonction objectif est la plus faible. Nous avons vu que cette stratégie pouvait conduire à l'exploration d'une partie importante de l'arborescence, cependant dans le cas des problèmes de conception optimale, le nombre de variables étant modéré, il est raisonnable de penser que cette stratégie permet d'obtenir rapidement une bonne solution.

3.4 Résolution des problèmes de conception optimale en variables mixtes : exemples

3.4.1 Prise en compte des paramètres discrets

L'algorithme de séparation et évaluation que nous avons développé, s'appuie sur des principes généraux et s'applique sur tout type de problèmes non linéaires en variables mixtes. Le problème doit cependant pouvoir être résolu en ignorant les restrictions dues aux variables discrètes. Dans un certain nombre de problèmes de conception optimale cette hypothèse est applicable sur les variables discrètes ou entières qui ne sont liées à aucun autre paramètre discret. Par contre dans le cas des exemples présenté dans ce travail, cette hypothèse n'est pas directement applicable. En effet dans le problème du ressort, la valeur de la limite de résistance statique du fil du ressort dépend du diamètre de ce fil. On trouve une situation identique dans le cas de l'accouplement à plateaux où les paramètres géométriques des boulons dépendent de leur diamètre.

Le moyen le plus simple pour traiter ces variables discrètes (diamètre de fil et des boulons) comme des variables continues tout en conservant "le lien" entre ces variables et les paramètres qui en dépendent consiste à approximer ce lien par une expression polynomiale du type :

$$\phi(d) = \sum_{n=0}^{p} a_n d^n$$

Ce qui permet d'obtenir de bonnes approximations avec une valeur de p assez élevé. Dans le cas des boulons par exemple, la relation entre le pas et le diamètre nominal peut être approchée avec une expression de degré 6 (p=6), dont on donne une représentation graphique sur la figure 6.23.

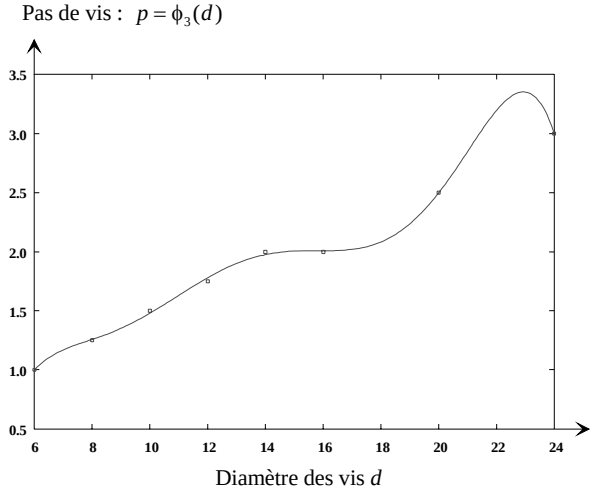


Figure 6.23 : Approximation du pas de vis en fonction du diamètre nominal des vis.

3.4.2 Exemples de calculs

Pour appliquer l'algorithme de séparation et évaluation sur les exemples du ressort et de l'accouplement, les expressions de ces problèmes ont été modifiées en intégrant ces diverses expressions polynomiales (détail des expressions de ces problèmes dans l'annexe 4).

Avant de pouvoir utiliser la méthode de séparation et évaluation, il est nécessaire d'établir une stratégie de pénalisation correcte pour ces deux problèmes, elle sera ensuite réutilisée par l'algorithme de séparation et évaluation pour la résolution de tous les problèmes associés aux sous ensembles séparés.

Dans le cas du ressort on obtient les résultats résumés sur la figure 6.24, et présentés sous la forme d'une arborescence pour plus de clarté. Pour chaque sommet la valeur de la fonction objectif, la solution obtenue et les bornes sur les variables sont précisées. On constate l'efficacité de la méthode, le calcul de la solution optimale en variables mixtes nécessitant seulement une séparation et la résolution de trois problèmes en variables continues. (calculs effectués avec les données standard).

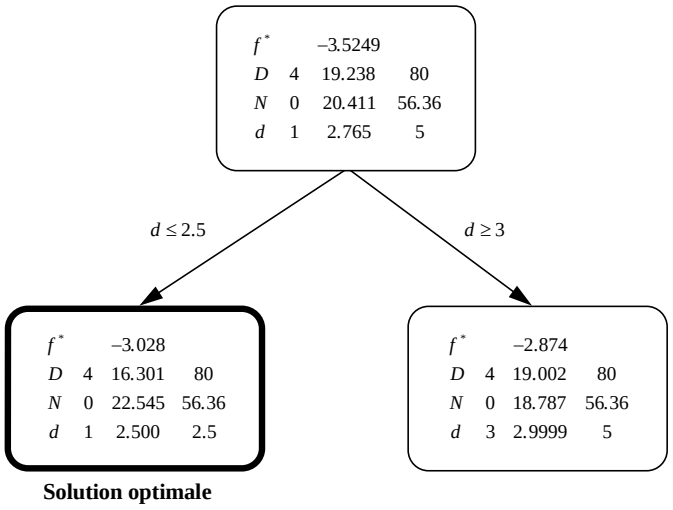


Figure 6.24 : Résultats sur le ressort de pompe hydraulique .

La résolution du problème de l'accouplement à plateaux est représentée par l'arborescence de la figure 6.25. Les chiffres situés à coté de chaque sommet indique le nombre d'itérations et le nombre d'évaluations de fonction nécessaire au calcul de la solution pour chaque sommet. En deux séparations nécessitant la résolution de 5 problèmes la méthode parvient à localiser la solution optimale du problème. Si on désactive la procédure de stérilisation dans l'algorithme on

obtient le reste de l'arborescence (partie entourée sur la figure) et d'autres solutions approchées (calculs effectués pour $\beta_1 = 1; \beta_2 = 1; \beta_3 = 1$).

On constate que la solution optimale est rapidement obtenue, et que seulement une petite partie des diamètres de boulons a été utilisée par l'algorithme.

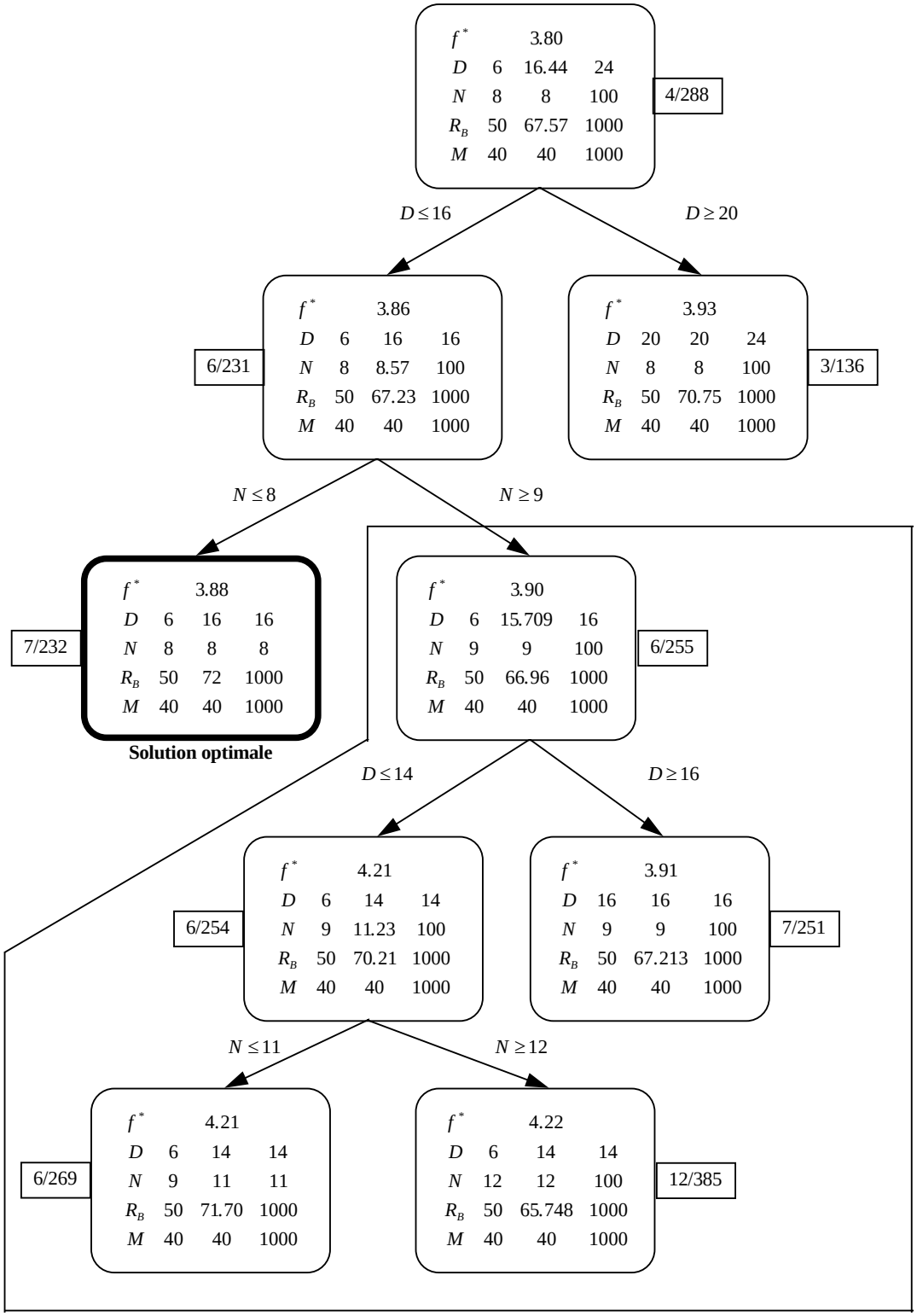


Figure 6.25 : Arborescence du problème de l'accouplement à plateaux.

3.5 Conclusions

L'utilisation d'une méthode de séparation et évaluation pour résoudre les problèmes de conception optimale donne des résultats intéressants. Dans les deux cas d'exemples présentés, la solution en variables mixtes a pu être obtenue sans parcourir tout l'ensemble des valeurs discrètes possibles. La méthode s'applique bien sur des problèmes comportant des paramètres discrets liés répartis de façon croissante monotone.

Il existe cependant certains cas de problèmes qui ne peuvent pas être résolus de cette manière. Notamment les problèmes de conception optimale comportant des roulements à billes. En effet dans ce type de problème la variable discrète attachée au choix d'un roulement est généralement la capacité de charge dynamique, dont dépendent les dimensions géométriques et la masse.

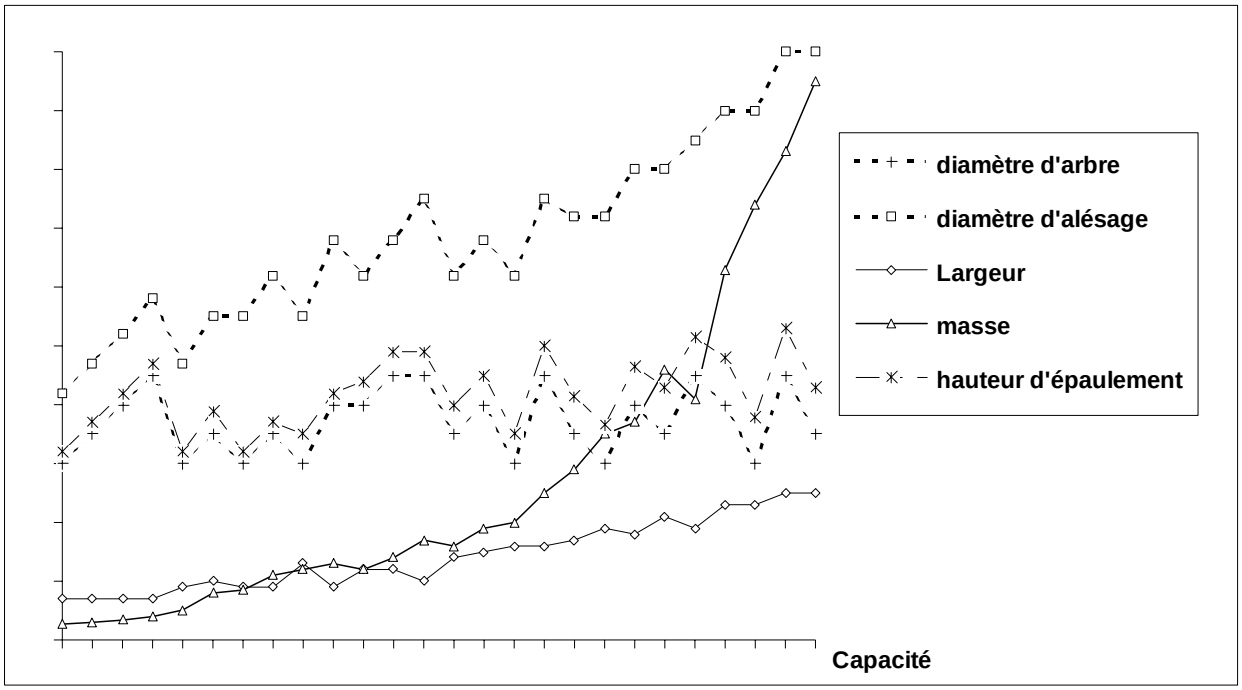


Figure 6.26 : Répartition non monotone croissante des paramètres discrets propres aux roulements à billes.

La figure 6.26 montre une répartition typique des dimensions géométriques et de la masse pour des roulements à une seule rangée de billes en fonction des capacités de charges dynamique croissante. On constate alors qu'il sera difficile d'établir une expression analytique simple d'une relation liant la capacité à ces divers paramètres. Et même si cela était possible, la non convexité de ces expressions risquerait de perturber la résolution du problème d'optimisation en variables continues ainsi obtenu.

Pour traiter ce type de problème, on peut imaginer d'autres règles de séparation de l'ensemble des valeurs discrètes. En effet plutôt que de séparer cet ensemble en deux sous ensembles définis par des bornes sur les variables, on pourrait créer autant de sous ensembles

que de valeurs discrètes. Le principe d'évaluation serait similaire, il s'agirait alors de résoudre le problème d'optimisation en variables continues, les composantes discrètes du vecteur des variables étant fixes.

La principale difficulté de cette approche se situe dans le choix de valeurs de départ correctes, puisqu'il sera impossible de calculer une valeur de départ en variables continues obtenue en relaxant les restrictions dues aux variables discrètes.

4 Mise en œuvre informatique

Sur tous les systèmes informatiques, la création d'un programme (code) exécutable par la machine passe par deux phases. Ces deux opérations nécessitent des outils qui sont eux-mêmes des programmes exécutables. Le premier d'entre eux, le compilateur, dépend du système et du langage de programmation, le second, l'éditeur de liens (linker), n'est généralement fonction que du système.

Le rôle du compilateur est de traduire un programme écrit dans un langage accessible à l'homme en code informatique. Toutes les opérations simples contenues dans le programme (addition, multiplication, boucles, etc..) sont directement traduites en code exécutable et chaque fois que le compilateur rencontre une instruction nécessitant une opération plus complexe (opération d'entrées sorties, calcul de fonctions mathématiques, etc ..), il introduit dans ce code informatique une référence à un autre code exécutable censé effectuer cette opération.

La tâche de l'éditeur de liens est d'assurer la présence de tous ces "morceaux" de codes exécutables dans le programme final.

Tous les systèmes informatiques sont livrés avec leurs propres compilateurs et éditeurs de liens, mais également avec l'ensemble des codes exécutables permettant ces opérations complexes, regroupés sous la forme de "bibliothèques ou librairies systèmes".

Pour augmenter les fonctionnalités d'un système informatique le programmeur peut acquérir des bibliothèques informatiques du commerce, ou créer lui même ses propres librairies. En ce qui concerne le calcul scientifique par exemple, les bibliothèques actuellement disponibles sur le marché contiennent un grand nombre d'opérations mathématiques couramment utilisées : résolution de systèmes linéaires, calcul de valeurs propres, etc. .. .

4.1 Structure du logiciel d'optimisation

Un logiciel d'optimisation nécessite deux types de données :

- Des données numériques spécifiques au problème à résoudre.
- La description du problème sous forme d'équations codifiées dans un certain langage de programmation.

Les résultats du calcul sont les valeurs des différentes variables du problème, celles des fonctions objectif et contraintes, mais également toutes les informations relatives au processus itératif : évolution de ces valeurs au cours des itérations, nombre d'itérations et d'évaluations de fonctions nécessaires, etc.. .

L'utilisation d'un logiciel d'optimisation nécessite donc une étape de programmation préliminaire dans laquelle on codifie les fonctions objectif et contraintes du problème et éventuellement les différents gradients du problème. La structure la plus adéquate pour ce type de logiciel est certainement celle d'une bibliothèque de calcul. L'utilisateur peut alors utiliser les fonctionnalités du logiciel comme de simples "sous-programmes".

Le logiciel d'optimisation que nous avons mis en place se présente donc comme un ensemble de sous programmes regroupés dans une structure de bibliothèque. L'utilisation de ce logiciel est schématiquement présentée sur la figure 6.27. Ce schéma est également valable pour toute bibliothèque informatique.

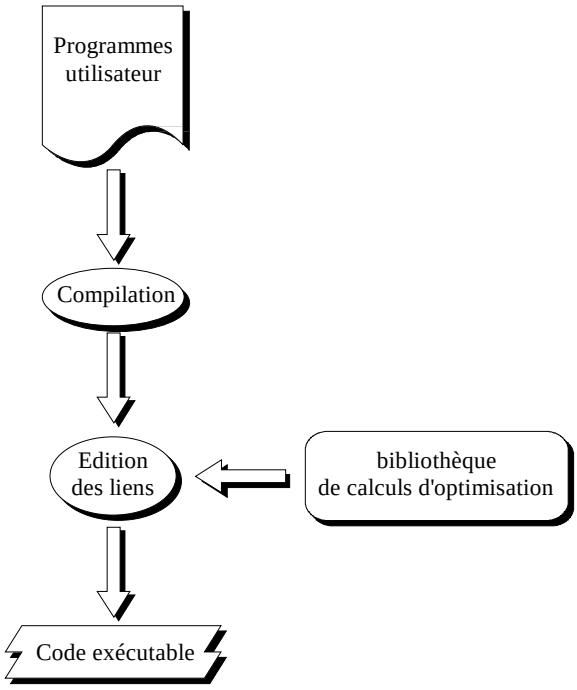


Figure 6.27 : Utilisation du logiciel d'optimisation.

La structure interne du logiciel s'articule autour de différents modules (figure 6.28). Chacun de ces modules est chargé d'une tâche précise. Par exemple le module MLA (Méthode du Lagrangien Augmenté) contient les différents sous programmes de calcul spécifiques à cette méthode. Ces modules utilisent un ensemble de sous programmes chargés des fonctions de "bas niveau" de la bibliothèque comme par exemple les entrées sorties sur fichiers, les calculs élémentaires (produits scalaires, normes, etc..) et également la gestion de la mémoire. Cette structure modulaire présente l'avantage d'une grande souplesse de modification permettant, par exemple, l'adjonction de nouvelles fonctionnalités sans gros efforts de programmation.

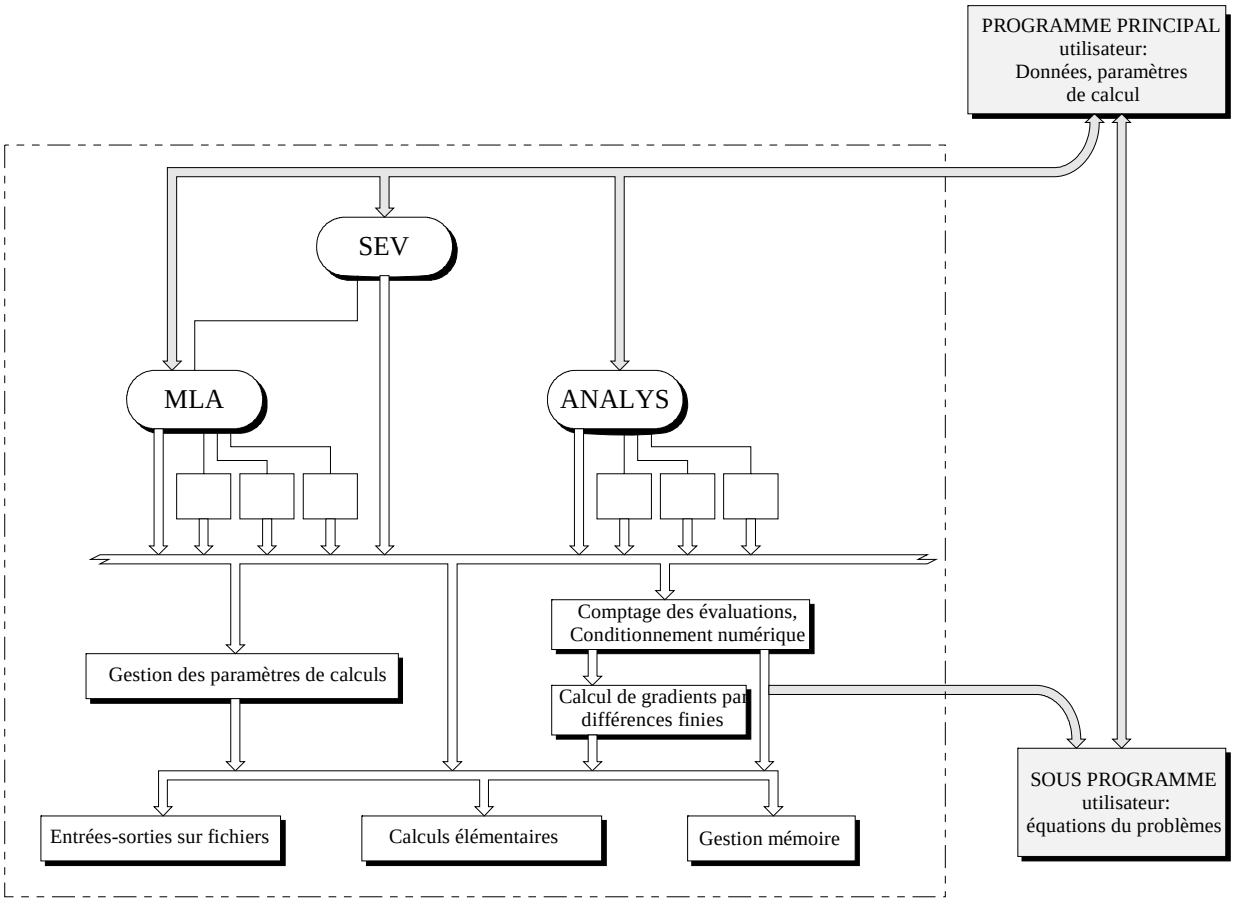


Figure 6.28 : Structure générale de la bibliothèque de calculs d'optimisation.

L'ensemble de ce logiciel est écrit en FORTRAN 77 standard, langage de prédilection pour le calcul scientifique. Ce langage ne permet pas l'utilisation de "mémoire temporaire" pour stocker des vecteurs ou des matrices de données en cours de calcul. La totalité de la mémoire nécessaire doit donc être "réservée" avant tout calcul. Dans certaines bibliothèques de calcul, c'est l'utilisateur qui réserve cette mémoire de stockage temporaire dans son programme principal. Afin d'éviter cette opération fastidieuse, notre logiciel comporte un module de gestion de mémoire temporaire réservant une quantité fixée de mémoire. Lorsque cette quantité n'est pas suffisante, pour des problèmes de grandes dimensions (beaucoup de variables, et de fonctions

contraintes), l'utilisateur est informé de la quantité de mémoire nécessaire et il peut alors facilement l'augmenter par l'appel d'une fonction spécialisée.

4.2 Fonctionnalités

Cette bibliothèque comporte les méthodes d'optimisation suivantes:

- Méthode du lagrangien augmenté, avec possibilité de plusieurs types de conditionnement numérique et calcul des gradients par différences finies.
- Analyse des combinaisons de contraintes actives par l'analyse monotone.
- Méthode de séparation et évaluation.

La conception modulaire de tous les éléments de cette bibliothèque, permet d'ajouter facilement d'autres méthodes d'optimisation.

Certains paramètres de contrôle sont fournis au sous programme utilisateur. Ils permettent de choisir à tous moments dans le processus de calcul l'évaluation des gradients par différences finies d'une ou de plusieurs fonctions contraintes. Cette particularité permet de codifier les gradients faciles à calculer analytiquement et d'utiliser les différences finies pour les autres. On peut également, grâce à ces paramètres de contrôle, arrêter les calculs à tout instant, dans le cas par exemple où l'algorithme nécessite l'évaluation d'une fonction en un point pour lequel elle n'est pas définie.

Conclusions, Perspectives

Partant des concepts fondamentaux que sont la réalisation technologique d'une liaison et la décomposition fonctionnelle d'un système mécanique complexe en sous ensembles élémentaires, nous avons pu montrer qu'un problème de conception pouvait se formuler comme un problème d'optimisation.

Nous avons ensuite défini ce type particulier de problème d'optimisation en précisant la démarche de sa formulation. Le problème de conception optimale ainsi obtenu est un problème d'optimisation non linéaire, fortement contraint, comportant des variables mixtes.

L'étude des méthodes de programmation mathématique nous a alors permis de justifier les choix réalisés dans la mise au point d'une méthode de résolution adaptée. Cette méthode s'appuyant sur un algorithme de lagrangien augmenté couplé avec un principe de séparation et évaluation est intégrée dans un environnement logiciel souple et modulaire. Les solutions optimales obtenues, sur les exemples présentés dans ce travail, ont montré la validité et les performances de ce code de calcul.

L'utilisation d'une méthode de programmation mathématique pour la résolution des problèmes de conception optimale appelle quelques commentaires :

- La manipulation de ce code de calcul nécessite de bonnes connaissances en programmation mathématique. Il serait utopique de vouloir obtenir une solution dès le premier calcul. L'étude préalable du problème à résoudre est alors indispensable.

- L'application systématique de ce code de calcul pour trouver la solution d'un problème ne fait certainement pas partie de la meilleure stratégie. Il existe en effet des petits problèmes d'optimisation très fortement non convexes qui ne pourront pas être résolus avec cette méthode. Lorsque ces problèmes comportent des fonctions monotones, l'analyse monotone apporte une aide efficace à la fois dans la modélisation et dans la résolution de ce type de problèmes.

- L'analyse monotone constitue une alternative efficace par rapport aux méthodes traditionnelles. Le couplage d'un algorithme de calcul de la solution optimale basé sur les principes de l'analyse monotone avec une méthode de séparation et évaluation est extrêmement intéressant.

Ce travail ouvre plusieurs perspectives :

L'expérience acquise lors du développement de ce code de calcul a permis d'entamer la mise au point d'une autre méthode résolution utilisant également le lagrangien augmenté. Elle utilise le principe d'une direction de déplacement des méthodes primales. Cette direction de recherche est calculée à partir d'une approximation quadratique du problème d'optimisation et la procédure de recherche undimensionnelle consiste alors à minimiser le lagrangien augmenté dans cette direction [11]. Les résultats obtenus sur quelques problèmes tests sont très prometteurs.

L'utilisation d'un principe de séparation et d'évaluation doit faire l'objet d'études supplémentaires. En effet, la prise en compte d'une classe de paramètres discrets liés plus générale que celle que nous avons envisagée semble tout à fait possible. Cela permettrait de traiter les problèmes comportant des ensembles standards complexes tels que les roulements.

L'intégration de ce code de calcul dans un logiciel de CAO est tout à fait envisageable. Les modeleurs géométriques récents intègrent des fonctionnalités permettant d'avoir accès à certaines informations concernant la géométrie des pièces contenues dans le système. On peut alors imaginer une base de données contenant des problèmes d'optimisation préformulés pour un type particulier de mécanisme dont les variables seraient "en liaison" avec le modeleur géométrique du logiciel de CAO. L'application de notre code de calcul permettrait alors de déterminer la solution optimale des problèmes préformulés dont les données seraient issues du modeleur.

Références bibliographiques

[1] AFNOR
"Recueil de normes françaises : boulonnerie et visserie" 1980

[2] ABADIE J.
"Non linear programming"
Edition North Holland Company Amsterdam 1967

[3] ALMGREN A.S., AGOGINO A.M.
"A Generalization and correction of welded beam optimal design problem using symbolic computation"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Technical Briefs, Vol. 111, Mars 1989, p 137-140

[4] AZARM S., LI W.-C.,
"Multi-level design optimization using global monotonicity analysis"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, Juin 1989, p259-251

[5] BEIGHTLER C.S., PHILLIPS D.T., WILDE D.J.
"Fondation of Optimization "
Edition Prentice-Hall 1979

[6] BEVERIDGE , SCHECHTER
"Optimization theory and pratice "
Edition Mac Graw Hill Serie chemical engineering 1970

[7] BROYDEN C.G.
"The convergence of a class of double-rank minimization algorithms parts I and II "
J. Inst. Maths Applics, 1970, p 76-90 et 222-231

[8] CHA J.Z., MAYNE R.W.
"Optimisation with discrete variables via recursive quadratic programming: Part 1 - Concepts and definitions"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, mars 1989, p 124-129

[9] CHA J.Z., MAYNE R.W.
"Optimisation with discrete variables via recursive quadratic programming: Part 2 - Algorithm and results"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, mars 1989, p 130-136

[10] CHA J.Z., MAYNE R.W.
"The symmetric rank one formula and its application in discrete nonlinear optimization"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 113, septembre 1991, p 312-317

[11] CHEN C., KONG W.C., CHA J.Z.
"An equality constrained RQP algorithm based on the augmented lagrangian penalty function"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, septembre 1989, p 368-374

[12] CHICUREL E.
"Global optimum search in design"
Proceedings of the Fifth World Congress on Theory of Machines and Mechanisms, Published by the ASME 1979

[13] CIARLET P.G.
"Introduction à l'analyse numérique matricielle et l'optimisation"
Edition MASSON 1982

[14] CULLUM J., BRAYTON R.K.
"Some remarks on the symmetric rank-one update"
Journal of Optimization Theory and Applications, Vol. 29, n° 4, decembre 1979, p 493-519

[15] DAVYDOV E.G., SIGAL I.KH.
"Application of the penalty function method in integer programming problems "
Engineering Cybernetics, Vol. 10., n° 10, 1972, p 21-24

[16] EASON E.D., FENTON R.G.
"A comparison of numerical optimization methods for engineering design"
ASME Journal of Engineering for Industry, Vol. 96, n° 96,Février 1974, p 196-200

[17] FLETCHER R., POWEL M.J.D.
" A rapidly convergent descent method for minimization"
Computer Journal, Vol. 6, n°2, 1963, p 163-168

[18] FLETCHER R.
"Function minimization without evaluating derivatives - a review"
Computer Journal, Vol. 8, 1965, p 33-41

[19] FLETCHER R.
"A new approach to variable metric algorithms "
Computer Journal, Vol. 13, n°3, 1970, p 317-322

[20] FLETCHER R.
"Practical Methods of Optimization " Volume 1 Unconstrained Optimization
Edition John Wiley & Sons 1980

[21] FOX .L.
"Optimization Method for engineering design"
Edition Addison-Wesley 1971

[22] FREE J.W., PARKINSON A.R., BRYCE G.R., BALLING R.J.
"Approximation of computationally expensive and noisy functions for constrained nonlinear optimization"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 110, décembre 1987, p 528-531

[23] GILL P.E., MURRAY W.
"Conjugate-gradient Methods for Large-scale Nonlinear Optimization"
Department of Operations Research, Stanford University, Technical Report SOL 79-15, 1979

[24] GILL P.E., MURRAY W.
"Practical Optimization"
Edition Academic-Press, New York, 1981

[25] GISVOLD K.M., MOE J.
"A method for nonlinear mixed-integer programming and its application to design problems"
ASME Journal of Engineering for Industry, Vol. 94, n° 2, mai 1972, p 353-364

[26] GOLDFARB D.
"A family of Variable-Metric Methods Derived by Variational Means"
Mathematics of Computation, Vol. 24, 1970, p 23-26

[27] GUILLOT J.
"Assemblages par éléments filetés"
Technique de l'ingénieur n° B 5560 - 1987

[28] GUILLOT J.
"Méthodologie de définition des ensembles mécaniques en conception assistée par ordinateur , recherche des solutions optimales."
Thèse d'état, n° d'ordre 1343, Université Paul Sabatier, TOULOUSE (1987)

[29] GUPTA O.K., RAVINDRAN A.
"Nonlinear integer programming and discrete optimization"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, juin 1983, p 160-164

[30] HAN S.P.
"Penalty lagrangian methods via a quasi-newton approach"
Journal of Mathematics of Operations Research, Vol. 4, n° 3, aout 1979, p 291-302

[31] HANSEN P., JAUMARD B., LU S.H.
"Some further results on monotonicity in globally optimal Design"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, septembre 1989, p 345-352

[32] HANSEN P., JAUMARD B., LU S.H.
"A framework for algorithms in globally optimal design "
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, septembre 1989, p 353-360

[33] HANSEN P., JAUMARD B., LU S.H.
"An automated procedure for globally optimal design. "
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, septembre 1989, p 361-367

[34] JOHNSON R. C.
"Optimum design of mechanical elements "
Edition Wiley-Interscience 1980

[35] KELAHAN R.C., GADDY J.L.
"Application of the adaptive random search to discrete and mixed integer optimization "
International Journal for Numerical Methods in Engineering, Vol. 12., 1978, p 289-298

[36] LAFON P., GUILLOT J.
"Application de l'analyse monotone à la conception optimale d'éléments de machine"
I.I.T.T - Colloque STRUCENG & FEMCAD - GRENOBLE 17-18, Octobre 1990, p 111-116

[37] LASCAUX P., THEODOR R.
"Analyse numérique matricielle appliquée à l'art de l'ingénieur" Tome 1 et 2
Edition MASON 1986

[38] LEE W.-J., WOO T.C.
"Optimum selection of discrete tolerances"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 111, juin 1989, p 243-251

[39] LI H.L., PAPALAMBROS P.Y.
"A interior linear programming algorithm using local and global knowledge"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 110, mars 1988, p 58-64

[40] LI H.L., PAPALAMBROS P.Y.
"A combined local-global active set strategy for nonlinear design optimization"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 110, décembre 1988, p 464-471

[41] LOH H.T., PAPALAMBROS P.Y.
"A sequential linearization approach for solving mixed-discrete nonlinear design optimization problems"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 113, septembre 1991, p 325-334

[42] LOH H.T., PAPALAMBROS P.Y.
"Computional implementation and tests of a sequential linearization algorithm for mixed-discrete nonlinear design optimization problems "
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 113, septembre 1991, p 335-345

[43] MANGASARIAN O. L.
"Nonlinear Programming"
Edition Mac Graw Hill 1969

[44] MICHELENA N.F., AGOGINO A. M.
"Multiobjective hydraulic cylinder design"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 110, mars 1988, p 81-87

[45] MINOUX M.
"Programmation Mathématique : Théorie et Algorithmes " Tome 1 et 2
Edition DUNOD 1983

[46] MINOUX M., GONDRAN M.
"Graphes et algorithmes"
Edition Eyrolles 1979

[47] PAPALAMBROS P.Y., WILDE D.J.
"Global non-iterative design optimization using monotonicity analysis"
ASME Journal of Mechanical Design, Vol. 101, 1979, p 645-649.

[48] PAPALAMBROS P.Y., WILDE D.J.
"Regional monotonicity in optimum design"
ASME Journal of Mechanical Design, Vol. 102, Juin 1980, p 497-500

[49] PAPALAMBROS P.Y.
"Monotonicity goal and geometric programming"
ASME Journal of Mechanical Design, Vol. 104, Janvier 1982, p 108-113

[50] PAPALAMBROS P.Y., WILDE D.J.
"Monotonicity analysis in optimum design of marine risers"
ASME Journal of Mechanical Design, Vol. 104, Octobre 1982, p 819-854

[51] PAPPAS M., AMBA-RAO C.L.
"A discrete search procedure for the minimization of stiffened cylindrical shell stability equations"
AIAA Journal, Vol. 8, n° 11, Novembre 1970.

[52] PAPPAS M., ALLENTUCH A.
"Mathematical programming procedures for mixed discrete-continuous design problems"
ASME Journal of Engineering for Industry, février 1974, p 201-209

[53] PARKINSON A., WILSON M.
"Development of a hybrid SQP-GRG algorithm for contrained nonlinear programming"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 110, septembre 1988, p 308-315

[54] POWEL M.J.D.
"An efficient method for finding the minimum of a function of several variables without calculating derivatives"
Computer Journal, 1964, Vol. 7, p 155-162

[55] RAO J.R., PAPALAMBROS P.Y.
"PRIMA: A production based implicit elimination system for monotonicity analysis of optimal design models".
ASME Journal of Mechanical Design, Vol. 113, Décembre 1991, p 408-415

[56] REITER S., RICE D.B.
"Discrete optimizing solution procedures for linear and nonlinear integer programming problems"
Management Science, Vol. 12, n° 11, Juin 1965, p 829-850

[57] ROCKAFELLAR R.T.
"Convex Analysis"
Edition Princeton University Press 1970

[58] ROCKAFELLAR R.T.
"The multiplier method of Hestenes and Powel applied to convex programming"
Journal of optimization theory and applications, Vol. 12, n° 6, 1973, p 555-562

[59] ROCKAFELLAR R.T.
"A dual approach to solving nonlinear programming problems by unconstrained optimization"
Mathematical Programming, n° 5, 1973, p 354-373

[60] ROCKAFELLAR R.T.
"Augmented lagrange multiplier functions and duality in nonconvex programmings"
SIAM Journal Control, Vol. 12, 1974, p 268-285

[61] ROOT R.R, RAGSDELL K.M.
"Computational enhancements to the method of Multipliers"
ASME Journal of Mechanical Design, Vol. 102, Juin 1980 p 517-523

[62] SAKAROVITCH M.
"Optimisation combinatoire, programmation discrète"
Edition Hermann 1984

[63] SANDGREN E., RAGSDELL K.M.
"The utility of nonlinear programming algorithms: A comparative study - Part I and Part II"
ASME Journal of Mechanical Design, Vol. 112, Juin 1980, p 540-551

[64] SANDGREN E.
"Nonlinear integer and discrete programming for topological decision making in engineering design"
ASME Journal of Mechanical Design, Vol. 112, mars 1990, p 118-122

[65] SANDGREN E.
"Nonlinear integer and discrete programming in mechanical design optimization"
ASME Journal of Mechanical Design, Vol. 112, juin 1990, p 223-229

[66] SHANNO D.F.
"Conditioning of Quasi-Newton Methods for Function Minimization"
Mathematics of Computation, Vol. 24, n°11, 1970, p 647-656

[67] VANDERPLAATS G.N.
"Numerical optimization techniques for engineering design"
Edition Mac Graw Hill 1984

[68] WILDE D.J.
"Monotonicity and dominance in optimal hydraulic cylinder design"
ASME Journal of Engineering Optimization ,Vol. 97, novembre 1975, p 1390-1394.

[69] WILDE D.J.
"The monotonicity table in optimal engineering design"
Engineering Optimization, 1976, Vol 2, p 29-34.

[70] WILDE D.J.
"Case identification in optimal design of a weldement"
CAD-CAM Robotics and Automation conference, University of ARIZONA, TUCSON, 11-15 Février 1985, p 171-178

[71] ZHOU J., MAYNE R.W.
"Monotonicity analysis and the reduced gradient method in constrained optimization"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 113, mars 1984, p 90-94

[72] ZHOU J., MAYNE R.W.
"Monotonicity analysis and recursive quadratic programming in contrained optimization"
ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 107, décembre 1985, p 459-462

ANNEXES

Annexe 1

Notations

1 Ensembles

R	Ensemble des nombres réels
$\{x, y, z\}$	Ensemble constitué de trois éléments
$x \in D$	x est élément de D.
$x \notin D$	x n'est pas élément de D.
$\{x \,/\, x \text{ tel que } \cdots\}$	ensemble des éléments x tel que ....
$\forall x \in D$	Quelque soit x élément de D.
$(P) \Rightarrow (Q)$	La propriété *(P)* entraîne la propriété *(Q)*.

2 Vecteurs et matrices

Notation générales :

Lettres minuscules grecques pour les réels $(\alpha, \beta, \gamma, \delta \cdots)$
Lettres minuscules latines pour les vecteurs $(u, v, x, d \cdots)$
Lettres majuscules pour les matrices $(H, A, G \cdots)$

R^n	ensemble des vecteurs à n composantes.
$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$	Vecteur de R^n de composantes : $x_1, x_2, \ldots, x_n$ identifé à une matrice colonne.
x^T	Tranposé du vecteur x : matrice ligne.
$x^T.y$	Produit scalaire du vecteur x par le vecteur y, vecteurs de R^n, $x^T.y = y^T.x = \sum_{i=1}^{n} x_i y_j$
$\|x\|$	Norme euclidienne du vecteur x.

$x \geq y$	Chaque composante du vecteur *x* est supérieure à la composante correspondante du vecteur *y*.
$x = 0$	Toutes les composantes du vecteur *x* sont nulles.
$A = \left[a_{ij} \right]_{\substack{i=1\cdots n \\ j=1\cdots n}}$	Matrice à *n* lignes et *n* colonnes de terme général a_{ij}.
$A.B$	Produit matriciel.
A^T	Matrice transposée de la matrice *A*.

3 Fonctions, gradients, hessiens

$\nabla f(x)$	Si $f(R^n \to R)$ est une fonction des variables $x_1, x_2, \ldots, x_n$, $\nabla f(x)$ est le gradient de *f* au point *x*, soit le vecteur de *n* composantes : $\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x)$ où $\frac{\partial f}{\partial x_i}(x)$ sont les dérivées partielles de *f* par rapport à x_i evaluées en *x*.
$\nabla^2 f(x)$	Désigne le hessien de *f* au point *x*, c'est à dire la matrice réelle $n \times n$ symétrique dont le terme général (i, j) est : $\frac{\partial^2 f}{\partial x_i \partial x_j}(x)$: les dérivées partielles étant évaluées au point *x*.
$\nabla_x f(x, y)$	Si $f(R^n \times R^m \to R)$ est une fonction des variables $x_1, x_2, \ldots, x_n$ et $y_1, y_2, \ldots, y_m$, $\nabla_x f(x, y)$ est le gradient au point (*x*,*y*), c'est à dire le vecteur de *n* composantes : $\frac{\partial f}{\partial x_1}(x, y), \ldots, \frac{\partial f}{\partial x_n}(x, y)$, les dérivées partielles étant évaluées au point (*x*,*y*).
$\nabla_x^2 f(x, y)$	Désigne le hessien en *x* de *f* au point (*x*,*y*) , c'est à dire la matrice réelle $n \times n$ symétrique dont le terme général (i, j) est : $\frac{\partial^2 f}{\partial x_i \partial x_j}(x, y)$: les dérivées partielles étant évaluées au point (*x*,*y*).
$f(x^*) = \underset{x \in D}{Min}\ f(x)$	*x* éléments de *D* est le plus grand minorant de *f* sur *D*.

Annexe 2

Méthodes de directions conjuguées

1 Convergence finie d'une méthode de direction conjuguée

Soit :

$$q(x) = \frac{1}{2} x^T . A . x + b . x + c$$

Avec :
A : matrice carrée symétrique et définie positive
b : Vecteur de R^n.
c : réel.

On a :

$$x^{k+1} = x^k + \alpha^k d^k \text{ et } \alpha^k \text{ tel que : } q(x^k + \alpha^k d^k) = \underset{\alpha \geq 0}{Min}\ q(x^k + \alpha d^k)$$

Aprés n itérations de la méthode on a:

$$x^n = x^0 + \sum_{j=1}^{n-1} \alpha^j d^j \qquad \text{(A2.1)}$$

En multipliant par : $d^{k^T} . A$ on a $\forall k \in [0, \ldots, n-1]$:

$$d^{k^T} . A . x^n = d^{k^T} . A . x^0 + \sum_{j=1}^{n-1} \alpha^j d^{k^T} . A . d^j \qquad \text{(A2.2)}$$

Or les directions $(d^0, d^1, \ldots, d^{n-1})$ sont A conjuguées donc :

$$\sum_{j=1}^{n-1} \alpha^j d^{k^T} . A . d^j = \alpha^k d^{k^T} . A . d^k$$

(il ne reste que le terme pour $j=k$).

On obtient alors la relation :

$$d^{k^T} . A . x^n = d^{k^T} . A . x^0 + \alpha^k d^{k^T} . A . d^k, \ \forall k \in [0, \ldots, n-1] \qquad \text{(A2.3)}$$

En particulier pour $n=k$, (A2.2) devient :

$$d^{k^T} . A . x^k = d^{k^T} . A . x^0 \qquad \text{(A2.4)}$$

puisque : $d^{k\,T}.A.d^{j} = 0 \;\; \forall j \in [0,\ldots,k-1]$

Comme α^k minimise $q(x^k + \alpha d^k)$ on a:

$$d^{k\,T}.\nabla q(x^k + \alpha^k d^k) = 0$$
$$\Rightarrow d^{k\,T}.\left[A.(x^k + \alpha^k d^k) + b\right] = 0$$
$$\Rightarrow \alpha^k d^{k\,T}.A.d^k = -d^{k\,T}.\left[A.x^k + b\right] = -d^{k\,T}.\left[A.x^0 + b\right]$$

En reprenant (A2.4) il vient :

$$d^{k\,T}.A.x^n = d^{k\,T}.A.x^0 - d^{k\,T}.\left[A.x^0 + b\right] \qquad \forall k \in [0,\ldots,n-1]$$
$$\Rightarrow d^{k\,T}.\left[A.x^n + b\right] = 0 \Rightarrow d^{k\,T}.\nabla q(x^n) = 0, \qquad \forall k \in [0,\ldots,n-1]$$

D'où l'on déduit que nécessairement $\nabla q(x^n) = 0$,puisque les directions $(d^0, d^1, \ldots, d^{n-1})$ sont mutuellement conjuguées, donc linéairement indépendantes. Donc x^n minimum de $q(x)$.

2 Méthode du gradient conjugué pour fonction quadratique

Dans cette méthode la direction de déplacement courante est déterminée grâce à :

$$d^k = -\nabla q(x^k) + \beta^{k-1} d^{k-1} \qquad \text{(A2.5)}$$

Le résultat précédent permet d'écrire :

$$d^{k\,T}.\nabla q(x^{k+1}) = 0 \qquad \forall k \in [0,\ldots,n-1]$$
$$d^{k-1\,T}.\nabla q(x^{k+1}) = 0 \qquad \forall k \in [0,\ldots,n-1]$$

D'où :

$$\nabla^T q(x^{k+1}).\nabla^T q(x^k) = \nabla^T q(x^{k+1}).\left[-d^k + \beta^{k-1} d^{k-1}\right] = 0 \qquad \text{(A2.6)}$$

Donc les points générés par cette méthode sont tels que, les gradients en ces points sont mutuellement orthogonaux.

2.1 Calcul du pas de déplacement α^k

Comme α^k est tel que :

$$q(x^k + \alpha^k d^k) = \underset{\alpha \geq 0}{Min}\; q(x^k + \alpha d^k)$$

On obtient :

$$d^{k\,T}.\nabla q(x^k + \alpha^k d^k) = d^{k\,T}.\left[A.(x^k + \alpha^k d^k) + b\right] = 0$$

$$\alpha^k = \frac{-d^k.\left[A.x^k + b\right]}{d^{k\,T}.A.d^k} = \frac{-d^k.\nabla q(x^k)}{d^{k\,T}.A.d^k} \qquad \text{(A2.7)}$$

2.2 Détermination de β^k

Les directions $(d^0, d^1, \ldots, d^{n-1})$ sont mutuellement conjuguées par rapport à A.

Donc :

$$d^{k\,T}.A.d^{k+1} = 0 \qquad \forall k \in [0, \ldots, n-1]$$
$$\Rightarrow d^{k\,T}.A.\left[-\nabla q(x^{k+1}) + \beta^k d^k)\right] = 0$$

D'où :

$$\beta^k = \frac{\nabla^T q(x^{k+1}).A.d^k}{d^{k\,T}.A.d^k} \qquad \text{(A2.8)}$$

En remarquant que :

$$\nabla q(x^{k+1}) - \nabla q(x^k) = A.\left[x^{k+1} - x^k\right] = \alpha^k.A.d^k$$

et en multipliant cette relation par $\nabla q(x^{k+1})$ on en tire :

$$\nabla q(x^{k+1}).A.d^k = \frac{1}{\alpha^k}\nabla^T q(x^{k+1}).\left[\nabla q(x^{k+1}) - \nabla q(x^k)\right]$$

En utilisant (A2.8) il vient :

$$\beta^k = \frac{\nabla^T q(x^{k+1}).A.d^k}{d^{k\,T}.A.d^k}$$
$$= \frac{\nabla^T q(x^{k+1}).\left[\nabla q(x^{k+1}) - \nabla q(x^k)\right]}{\nabla^T q(x^k).\nabla q(x^k)} \qquad \text{(A2.9)}$$

La relation (A2.6) permet d'écrire également :

$$\beta^k = \frac{\nabla^T q(x^{k+1}).\nabla q(x^{k+1})}{\nabla^T q(x^k).\nabla q(x^k)} \qquad \text{(A2.10)}$$

Annexe 3

Méthodes quasi newtoniennes

1 Correction de rang 1

Calcul des termes de la matrice de correction de rang 1:

On a:

$$H^{k+1} = H^k + \lambda^k u^k . u^{k\,T}$$

Sachant que la relation $H^{k+1} y^k = s^k$ doit être satisfaite on a:

$$\left[H^k + \lambda^k u^k . u^{k\,T}\right] . y^k = s^k \qquad \text{(A3.1)}$$

En supprimant l'indice *k* pour plus de clarté :

$$\lambda u . u^T = s - H . y \qquad \text{(A3.2)}$$

En prenant le produit scalaire des deux membres avec *y* on obtient :

$$\lambda (y^T . u)(u^T . y) = y^T . [s - H . y]$$

$$\Rightarrow \lambda (y^T . u)^2 = y^T . [s - H . y] \qquad \text{(A3.3)}$$

Si on utilise l'identité :

$$\lambda (u . u^T) = \frac{(\lambda u . u^T . y)(\lambda u . u^T . y)}{\lambda (u^T . y)^2}$$

et les relations (A3.2) et (A3.3) on obtient :

$$H^{k+1} - H^k = \lambda u . u^T = \frac{[s - H . y] . [s - H . y]^T}{y^T . [s - H . y]}$$

D'où la formule de correction de rang 1 :

$$H^{k+1} = H^k + \frac{\left[s^k - H^k . y^k\right] . \left[s^k - H^k . y^k\right]^T}{y^{k\,T} . \left[s^k - H^k . y^k\right]} \qquad \text{(A3.4)}$$

2 Formules de correction de rang 2

2.1 Méthode DFP

Considérons la relation :

$$H^{k+1} = H^k + D^k$$

En utlisant (A3.1) il vient :

$$D.y = s - H.y \tag{A3.5}$$

$$\Rightarrow D.y(D.y)^T = D.y.y^T.D = \left[s - H.y\right].\left[s - H.y\right]^T$$

$$\Rightarrow D.y.y^T.D = s.s^T - (H.y.s^T + s.y^T.H) + H.y.y^T.H \tag{A3.6}$$

Si on pose $D = \lambda u.u^T + \beta v.v^T$, en développant le produit $D.y.y^T.D$ on obtient :

$$\begin{aligned} D.y.y^T.D = {} & \lambda^2 (u^T.y)^2 u.u^T + \lambda\beta (v^T.y)(y^T.u) v.u^T \\ & + \lambda\beta (u^T.y)(y^T.v) u.v^T + \beta^2 (v^T.y)^2 v.v^T \end{aligned} \tag{A3.7}$$

En identifiant (A3.6) et (A3.7) on a:

$$\left.\begin{array}{l} \lambda^2 (u^T.y)^2 = 1 \\ \lambda\beta (v^T.y)(y^T.u) = -1 \\ \beta^2 (v^T.y)^2 = 1 \end{array}\right| \begin{array}{l} u = s \\ v = H.y \end{array}$$

Donc $\lambda = \pm \dfrac{1}{s^T.y}$; $\beta = \pm \dfrac{1}{y^T.H.y}$

Or $\lambda (u^T.y)u + \beta (v^T.y)v = s - H.y$ d'aprés (A3.5).

L'une des possibilités est :

$$\begin{cases} \lambda (u^T.y).u = s \\ \beta (v^T.y).v = -H.y \end{cases}$$

D'où l'on déduit :$\lambda = \dfrac{1}{s^T.y}$; $\beta = -\dfrac{1}{y^T.H.y}$

On obtient alors la formule de correction DFP :

$$H^{k+1} = H^k + \frac{s^k.s^{k^T}}{s^{k^T}.y^k} - \frac{H^k.y^k.y^{k^T}.H^k}{y^{k^T}.H^k.y^k} \tag{A3.8}$$

L'identification des formules (A3.6) et (A3.7) laisse beaucoup de degrés de liberté dans le choix des coefficients, puisque à partir d'une équation ((A3.5)) il est impossible de déterminer les 4 inconnues (λ, β, u, v). Il existe donc une infinité de possibilités pour définir une formule de correction de rang 2. La formule DFP correspond à un choix trivial au niveau des différents coefficients qui la définissent.

2.2 Correction BFGS

Elle s'établit à partir de la formule de correction DFP en remplaçant H^k par $G^k = \left[H^k\right]^{-1}$ et en permuttant s^k et y^k, puis en calculant l'inverse des deux membres de la relation obtenue.

A partir de (A3.11) on a:

$$G^{k+1} = G^k + \frac{y^k . y^{k^T}}{y^{k^T} . s^k} - \frac{G^k . s^k . s^{k^T} . G^k}{s^{k^T} . G^k . s^k} \tag{A3.9}$$

Commençons par calculer l'inverse de :$(I + u.v^T)$, avec $u, v \in R^n$ où $u.v^T$ est une matrice carrée de dimension *n* de rang 1.

Soit $x, y \in R^n$, tel que :

$$(I + u.v^T).x = y \tag{A3.10}$$

$$\Rightarrow v^T.(I + u.v^T).x = v^T.y$$

$$\Rightarrow v^T.x + v^T.u.v^T.x = v^T.y$$

$$\Rightarrow v^T.x = \frac{v^T.y}{1 + v^T.u} \tag{A3.11}$$

Or (A3.10) s'écrit aussi :$x = y - u.v^T.x$, en incorporant (A3.11) on obtient :

$$x = y - \frac{u.v^T.y}{1 + v^T.u} = \left(I - \frac{u.v^T}{1 + v^T.u}\right).y$$

Donc :

$$\left[I + u.v^T\right]^{-1} = \left[I - \frac{u.v^T}{1 + v^T.u}\right] \tag{A3.12}$$

On peut écrire l'inverse des deux membres de (A3.9) sous la forme :

$$\begin{aligned} \left[G^{k+1}\right]^{-1} = \left[A - B\right]^{-1} &= \left[A.(I - A^{-1}.B)\right]^{-1} \\ &= (I - A^{-1}.B)^{-1}.A^{-1} \end{aligned} \tag{A3.13}$$

Avec :

$$A = G + \frac{y.y^T}{y^T.s}$$

$$\Rightarrow A^{-1} = \left[G.\left(I + \frac{G^{-1}.y.y^T}{y^T.s}\right)\right]^{-1} = \left(I + \frac{G^{-1}.y.y^T}{y^T.s}\right)^{-1}.G^{-1}$$

et :

$$B = \frac{G.s.s^T.G}{s^T.G.s} = \frac{(G.s).(G.s)^T}{s^T.G.s}$$

Le calcul de l'inverse de *A* est immédiat grâce à (A3.12) :

Avec :$u = \frac{G^{-1}.y}{y^T.s}$ et $v = y$ on a:

$$A^{-1} = \left[I - \frac{G^{-1}.y.y^T}{y^T.s + y^T.G^{-1}.y} \right].G^{-1} \qquad \text{(A3.14)}$$

Comme *B* est de rang 1 (de la forme $x.x^T$ avec $x = G.s$), le produit $A^{-1}.B$ est de rang 1. On peut donc à nouveau appliquer (A3.12) pour le calcul de $\left(I - A^{-1}.B\right)^{-1}$.

Avec :

$$u = -\frac{A^{-1}.G.s}{s^T.G.s} \text{ et } v = G.s$$

On obtient :

$$\left(I - A^{-1}.B\right)^{-1} = \left(I + \frac{A^{-1}.(G.s).(G.s)^T}{s^T.G.s - (G.s)^T.A^{-1}.(G.s)} \right) \qquad \text{(A3.15)}$$

Sachant que $G^{-1} = H \Rightarrow H.G = I$, à l'aide (A3.14) et avec :

$$\alpha = (s^T.y) = (y^T.s)$$
$$\beta = y^T.H.y$$

on explicite les relations suivantes :

$$s^T.G.s - (G.s)^T.A^{-1}.(G.s) = \frac{\alpha^2}{\alpha + \beta}$$

$$A^{-1}.(G.s).(G.s)^T.A^{-1} = s.s^T - \frac{\alpha}{\alpha + \beta}\left[H.y.s^T + s.y^T.H\right] + \frac{\alpha^2}{(\alpha + \beta)^2} H.y.y^T.H$$

Donc , en reprenant les expressions (A3.13) et (A3.15) on a :

$$\left[G^{k+1}\right]^{-1} = H^{k+1} = A^{-1} + \frac{A^{-1}.(G.s).(G.s)^T.A^{-1}}{s^T.G.s - (G.s)^T.A^{-1}.(G.s)}$$

$$H^{k+1} = \left(H - \frac{H.y.y^T.H}{(\alpha + \beta)} \right) + \left(\frac{\alpha + \beta}{\alpha^2} \right) s.s^T - \frac{1}{\alpha}\left(H.y.s^T + s.y^T.H\right) + \frac{H.y.y^T.H}{(\alpha + \beta)}$$

d'où la formule de correction BFGS :

$$H^{k+1} = H^k + \left(1 + \frac{y^{kT}.H^k.y^k}{s^{kT}.y^k} \right) \frac{s^k.s^{kT}}{s^{kT}.y^k} - \frac{H^k.y^k.s^{kT} + s^k.y^{kT}.H^k}{y^{kT}s^k}$$

Annexe 4

Formulation des problèmes d'optimisation

1 Ressort de pompe hydraulique

La formulation du problème à 2 variables utilisée dans les calculs d'optimisation est la suivante :

Minimiser la fonction objectif : $F(D,N) = -\alpha_F = -\left(\frac{\pi\tau_D}{0.8GCd^{1.14}}\right)D^{2.14}N$

Sous les fonctions contraintes :

$$c_1(D,N) = 1-\left(\frac{\pi\tau_D}{0.8GCd^{1.14}}\right)ND^{2.14} \le 0$$

$$c_2(D,N) = D^3N\left(\frac{8k_m}{Gd^4}\right)-1 \le 0$$

$$c_3(D,N) = 1-\left(\frac{8k_M}{Gd^4}\right)D^3N \le 0$$

$$c_4(D,N) = 1-\left(\frac{2\pi f_{MAXI}}{d}\sqrt{\frac{2\rho}{G}}\right)D^2N \le 0$$

Bornes sur les variables :

$$D_{maxi} \le D \le D_{maxi}$$

$$0 \le N \le \left(\frac{L_M}{d(1+\varepsilon)}\right)$$

Avec :

$$D_{mini} = 4d$$

$$D_{\max i} = Max\left\{Min\left\{16d, D_M - d, \left(\frac{\pi\tau_L d^{2.86}}{12.8F_{MAXI}}\right)^{1/0.86}\right\}; D_{\min i}\right\}$$

Données : $C, F_{MAXI}, k_m, k_M, D_M, L_M, f_{MAXI}, \tau_D, \tau_L, \rho, G, \varepsilon, d$

La formulation du problème à 3 variables utilisée dans les calculs avec la méthode de séparation et évaluation est :

Minimiser la fonction objectif :

$$F(D,N,d) = -\alpha_F = -\left(\frac{\pi\tau_D}{0.8GC}\right)\frac{D^{2.14}N}{d^{1.14}}$$

Sous les fonctions contraintes :

$$c_1(D,N,d) = \left(\frac{12.8F_{MAXI}}{\pi}\right)\frac{D^{0.86}}{\phi(d).d^{2.86}} - 1 \le 0$$

$$c_2(D,N,d) = 1 - \left(\frac{\pi\tau_D}{0.8GC}\right)\frac{ND^{2.14}}{d^{1.14}} \le 0$$

$$c_3(D,N,d) = \frac{D^3N}{d^4}\left(\frac{8k_m}{G}\right) - 1 \le 0$$

$$c_4(D,N,d) = 1 - \left(\frac{8k_M}{G}\right)\frac{D^3N}{d^4} \le 0$$

$$c_5(D,N,d) = 1 - \left(2\pi f_{MAXI}\sqrt{\frac{2\rho}{G}}\right)\frac{D^2N}{d} \le 0$$

$$c_6(D,N,d) = dN\left(\frac{1+\varepsilon}{L_M}\right) - 1 \le 0$$

$$c_7(D,N,d) = \frac{D+d}{D_M} - 1 \le 0$$

$$c_8(D,N,d) = 4d - D \le 0$$

$$c_9(D,N,d) = D - 16d \le 0$$

Bornes sur les variables :

$4 \le D \le 80$ ($4d$ pour d=1, et $16d$ pour d=5)

$0 \le N \le \left(\frac{L_M}{1+\varepsilon}\right)$ (Borne maxi pour d=1)

$1 \le d \le 5$

Données : $C, F_{MAXI}, k_m, k_M, D_M, L_M, f_{MAXI}, \tau_D, \rho, G, \varepsilon$

Les données numériques sont :

C = 15 mm	D_M = 15 mm	L_M = 62 mm	ε = 0.1 mm
F_{MAXI} = 300 N	k_m = 4 N/mm	k_M = 10 N/mm	f_{MAXI} = 100 Hz
τ_D = 300 Mpa	G = 80 000 Mpa	ρ = 7.8 10^{-6}Kg/mm^3	

Sur le fil du ressort (XC85) :

d (mm)	1	1.5	2	2.5	3	3.5	4	4.5	5

τ_L (N/mm^2)	1220	1150	1080	1030	980	920	890	850	810

Elles donnent les bornes sur les variables suivantes:

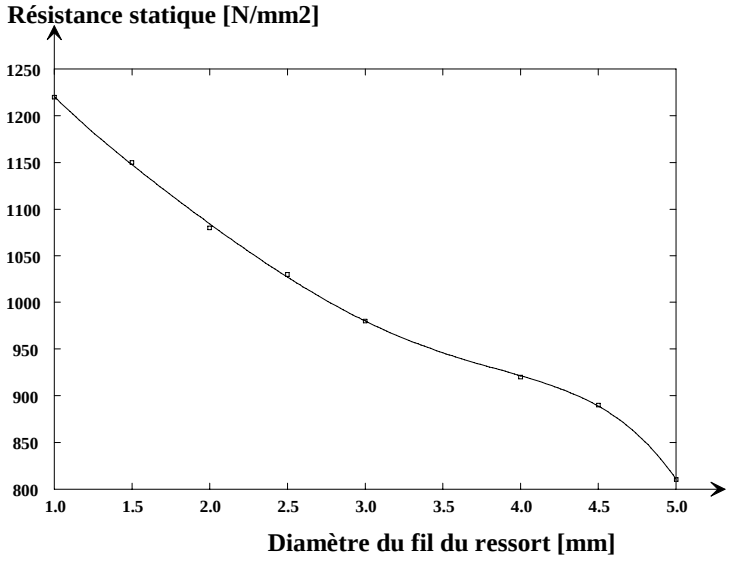
$$D_{mini} = 4d = 4 \leq D \leq D_{maxi} = 16d = 80$$

$$0 \leq N \leq \left(\frac{L_M}{d(1+\varepsilon)} \right) = 56.36$$

La fonction $\tau_L = \phi(d)$ a été approximée par l'expression polynomiale de degré 5 suivante :

$$\phi(d) = \sum_{n=0}^{5} a_n d^n$$

a_0	1480.834
a_1	-422.7076
a_2	246.6273
a_3	-104.908
a_4	22.46006
a_5	-1.806949



2 Accouplement à plateaux boulonnés

La formulation du problème à 3 variables utilisée pour les calculs est :

Minimiser la fonction objectif :

$$F(N, R_B, M) = \beta_1\left(\frac{N}{N_m}\right) + \beta_2\left(\frac{R_B + \phi_4(d) + c}{R_M}\right) + \beta_3\left(\frac{M}{M_T}\right)$$

Sous les fonctions contraintes :

$$c_1(N, R_B, M) = \frac{\alpha . M}{N . R_B . K(d)} - 1 \le 0$$

$$c_2(N, R_B, M) = 1 - \frac{2\pi R_B}{\phi_5(d) . N} \le 0$$

$$c_3(N, R_B, M) = 1 - \frac{R_B}{R_M + \phi_4(d)} \le 0$$

Bornes sur les variables :

$$N_m \le N \le N_{MAXI}$$

$$R_M \le R_B \le R_{MAXI}$$

$$M_T \le M \le M_{MAXI}$$

Données : $M_T, f_m, f_1, \alpha, R_e, N_m, R_M, c, d_m, d, N_{MAXI}, R_{MAXI}, M_{MAXI}$

Avec

$$K(d) = \frac{0.9 f_m R_e \pi (\phi_1(d))^2}{4\sqrt{1 + 3\left(\frac{4(0.16\phi_3(d) + 0.583\phi_2(d) f_1)}{\phi_1(d)}\right)^2}}$$

Les données numériques pour ce problème sont :

M_T = 4000 m.N	f_m = 0.15	α = 1.5	R_M = 50 mm
R_e = 627 Mpa	f_1 = 0.15	N_m = 8	c = 5 mm

Les bornes maximales $N_{MAXI}, R_{MAXI}, M_{MAXI}$ ont pour valeurs respectives : 100,1000,1000.

Remarque :

La variable *M* doit normalement être exprimée en mm.N. Cette unité ne convient pas car elle donne une valeur trop importante de *M* (4.10^6 mm.N) par rapport aux autres valeurs des variables du problème. On choisira alors de diviser cette valeur, ainsi que celle de R_e

par 10^5, afin de ramener les valeurs des variables du problème dans des proportions correctes.

Les valeurs normalisées utilisées pour les vis sont :

d	$d_e = \phi_1(d)$	$d_2 = \phi_2(d)$	$p = \phi_3(d)$	$b_m = \phi_4(d)$	$s_m = \phi_5(d)$
6	5.062	5.350	1.00	7.50	14.50
8	6.827	7.188	1.25	9.50	18.50
10	8.593	9.026	1.50	12.50	23.50
12	10.358	10.863	1.75	13.50	26.50
14	12.124	12.701	2.00	15.50	29.50
16	14.124	14.701	2.00	17.00	32.00
20	17.655	18.376	2.50	21.00	40.00
24	21.185	22.051	3.00	25.00	48.00

La formulation du problème à 4 variables utilisée pour les calculs avec la méthode de séparation et évaluation est :

Minimiser la fonction objectif :

$$F(d, N, R_B, M) = \beta_1\left(\frac{N}{N_m}\right) + \beta_2\left(\frac{R_B + \phi_4(d) + c}{R_M}\right) + \beta_3\left(\frac{M}{M_T}\right)$$

Sous les fonctions contraintes :

$$c_1(d, N, R_B, M) = \frac{\alpha . M}{N . R_B . K(d)} - 1 \le 0$$

$$c_2(d, N, R_B, M) = 1 - \frac{2\pi R_B}{\phi_5(d) . N} \le 0$$

$$c_3(d, N, R_B, M) = 1 - \frac{R_B}{R_M + \phi_4(d)} \le 0$$

Bornes sur les variables :

$$N_m \le N \le N_{MAXI}$$

$$R_M \le R_B \le R_{MAXI}$$

$$M_T \le M \le M_{MAXI}$$

$$6 \le d \le 24$$

Données : $M_T, f_m, f_1, \alpha, R_e, N_m, R_M, c, d_m, N_{MAXI}, R_{MAXI}, M_{MAXI}$

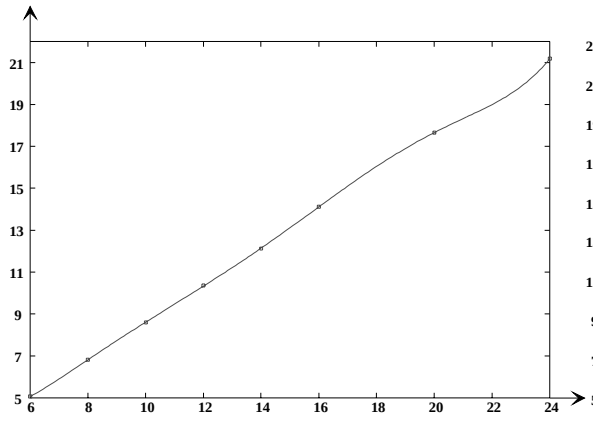
Les fonctions $\phi_i(d)$ ont été approximées par des fonctions polynomiales de degré p=3,5 et 6 du type :

$$\phi(d) = \sum_{n=0}^{p} a_n d^n$$

Les valeur des coefficients a_n, calculées pour minimiser le carré de l'écart entre les valeurs normalisées et la courbe polynomiale sont données dans le tableau ci-dessous :

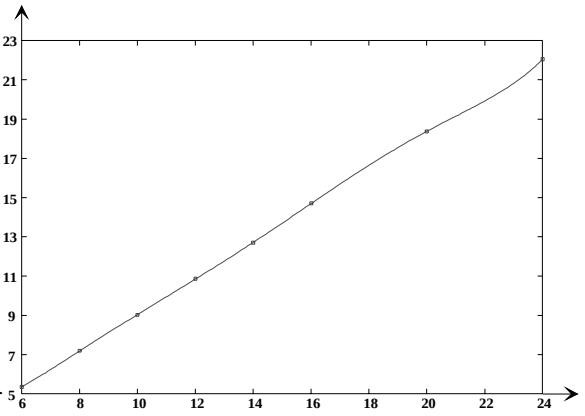
	$d_e = \phi_1(d)$	$d_2 = \phi_2(d)$	$p = \phi_3(d)$	$b_m = \phi_4(d)$	$s_m = \phi_5(d)$
a_0	21.60334	14.94477	-22.94488	-1.947764	52.33168
a_1	-10.99842	-7.304091	12.74743	1.94395	-21.30006
a_2	2.570593	1.77972	-2.731572	-7.173304E-02	4.052386
a_3	-0.283038	-0.1960102	0.3008197	1.567824E-03	-0.3256121
a_4	1.672547E-02	1.158507E-02	-1.777905E-02		1.211329E-02
a_5	-5.031615E-04	-3.485732E-04	5.349259E-04		-1.689069E-04
a_6	6.033268E-06	4.180142E-06	-6.414825E-06		

Diamètre section résistance équivalente : $d_e = \phi_1(d)$



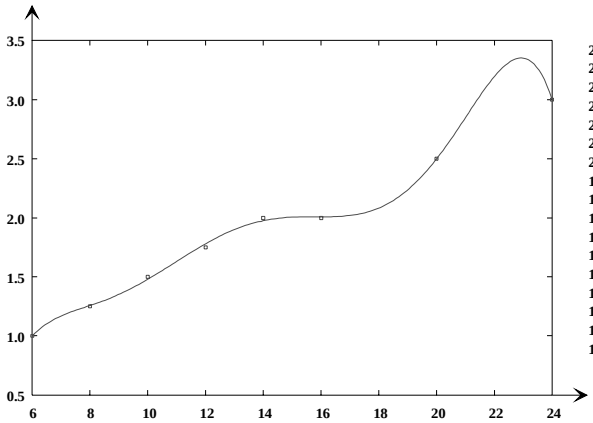
Diamètre des vis *d*

Diamètre moyen à flanc de filets : $d_2 = \phi_2(d)$



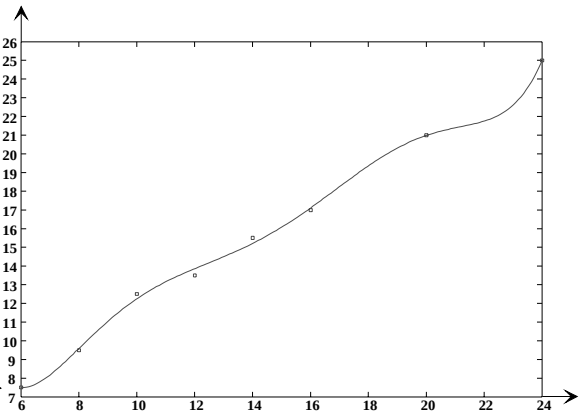
Diamètre des vis *d*

Pas de vis : $p = \phi_3(d)$



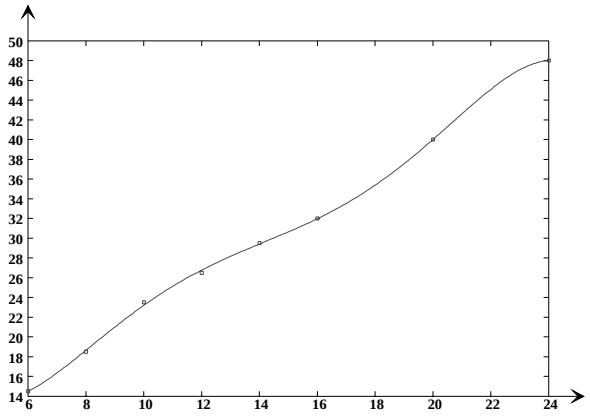
Diamètre des vis *d*

Encombrement outil de serrage : $b_m = \phi_4(d)$



Diamètre des vis *d*

Encombrement outil de serrage : $s_m = \phi_5(d)$



Diamètre des vis *d*

3 Poutre à tronçons de sections variables.

Expression du problème d'optimisation

Minimiser la masse de la poutre, donc le volume de matière : $\sum_{i=1}^{N} b_i h_i l_i$

La flèche en bout de poutre doit être inférieure à une valeur limite fixée : y_{MAX} :

La flèche est définie par les relations de récurrences suivantes :

$$\begin{cases} y_0 = y'_0 = 0 \\ y'_i = \dfrac{P l_i^{\,2}}{2EI_i}\left(L + \dfrac{2l_i}{3} - \sum_{j=1}^{i} l_j\right) + y'_{i-1} \\ y_i = \dfrac{P l_i}{EI_i}\left(L + \dfrac{l_i}{2} - \sum_{j=1}^{i} l_j\right) + y_{i-1} + y'_{i-1} l_i \end{cases}$$

La contrainte maximale de flexion à l'extrémité gauche de chaque tronçon σ_i doit rester inférieure à la limite élastique du matériaux : σ_{MAX}

$$\sigma_i = \frac{6P}{b_i h_i^2}\left(L + l_i - \sum_{j=1}^{i} l_j\right)$$

On impose également les conditions géométriques suivantes sur chaque tronçon :

$$\begin{cases} h_i \le k b_i \\ b_{\mathrm{MIN}} \le b_i \le b_{\mathrm{MAX}} \\ h_{MIN} \le h_i \le h_{MAX} \end{cases}$$

Finalement ce problème d'optimisation comportant $2N$ variables et $2N+1$ fonctions contraintes se formule de la manière suivante :

Minimiser la fonction objectif : $F(b_i, h_i) = \sum_{i=1}^{N} b_i h_i l_i$

Sous les fonctions contraintes : $c_1(b_i, h_i) = \frac{y_N}{y_{MAX}} - 1 \leq 0$

Pour $j = 2 \ldots N+1$ et $i = j-1$

$$c_j(b_i, h_i) = \frac{\sigma_i}{\sigma_{MAX}} - 1 \leq 0$$

Pour $j = N+2 \ldots 2N+1$ et $i = j-(N+1)$

$$c_j(b_i, h_i) = b_i - kh_i \leq 0$$

Bornes sur les variables :

$$b_{MIN} \leq b_i \leq b_{MAX}$$

$$h_{MIN} \leq h_i \leq h_{MAX}$$

Les données utilisées pour le calcul sont :

P= 50000 N E = 200Gpa L = 500 cm σ_{MAX} = 140 Mpa y_{MAX} = 0.5 cm