Éléments d'optimisation

Pierre Bernhard

1er septembre 2001

Table des matières

Chapitre 1

Prologue : rappels d'analyse

Les rappels qui suivent sont sans doute inutiles pour la plupart des élèves. Nous les donnons à fin de référence. Ils se limitent aux concepts d'analyse qui seront utiles à ce cours. Nous conseillons au lecteur de commencer sa lecture là où commence ce cours, soit au dernier paragraphe "Dérivation" de ce chapitre, (car il contient des conventions de notations utilisées ensuite) et de ne se reporter aux paragraphes antérieurs qu'en cas de besoin au cours de l'étude du cours.

1.1 La droite réelle

1.1.1 Propriétés fondamentales

On note par $\mathbb{N}$ l'ensemble des entiers. On appelle "rationnel" le rapport de deux entiers. On note $\mathbb{Q}$ l'ensemble des rationnels.

L'ensemble des réels, appelé aussi "droite réelle", noté $\mathbb{R}$, contient les rationnels, lesquels sont "denses dans $\mathbb{R}$" (c'est à dire que tout réel peut être approché arbitrairement près par des rationnels). Sa propriété fondamentale, que ce soit *par construction* (par "complétion" des rationnels) ou par axiome, est que les *suites de Cauchy* convergent. (On dit que $\mathbb{R}$ est *complet*.)

On suppose connu le concept de limite : $a_n \to a$ si quelque soit $\varepsilon > 0$, il existe un entier N tel que pour tout n supérieur à N, $|a - a_n| < \varepsilon$. Toutefois, il est plus adroit d'écrire cette définition de façon à peine différente :

Définition 1.1 (Limite) *On dit que la suite* $\{a_n\}_{n\in\mathbb{N}}$ tends vers a*, et on note* $a_n \to a$*, si*

$$\forall \varepsilon > 0,\ \exists N \in \mathbb{N} :\ \forall n > N,\ a_n \in (a - \varepsilon,\ a + \varepsilon)\,.$$

(Voir ci-dessous notre notation pour les intervalles, mais ici le fait de prendre un intervalle ouvert ou fermé n'a aucune importance. L'élégance pousse à choisir un ouvert.)

On rappelle à ce propos qu'une suite de Cauchy est une suite $\{a_n\}_{n\in\mathbb{N}}$ telle que "à condition d'attendre assez longtemps, les éléments en sont tous arbitrairement proches les uns des autres". Mathématiquement :

Définition 1.2 (Suites de Cauchy) *Une suite réelle* $\{a_n\}_{n\in\mathbb{N}}$ *est appelée* suite de Cauchy *si*

$$\forall \varepsilon > 0,\ \exists N : \forall m, n > N,\ |a_n - a_m| < \varepsilon\,.$$

Une conséquence fondamentale de la convergence des suites de Cauchy est la célèbre propriété suivante :

Proposition 1.1 (Suites monotones) *Les suites monotones bornées convergent.*

Démonstration : exercice.

Rappelons enfin une définition :

Définition 1.3 (Point d'accumulation) *Le point c est appelé* point d'accumulation *de la suite $\{a_n\}$ si $\forall \varepsilon > 0$, l'intervalle $(c-\varepsilon,\, c+\varepsilon)$ contient un nombre infini de points de la suite.*

Bien sûr, une limite est un point d'accumulation, mais une suite peut avoir aucun ou plusieurs points d'accumulation et (donc) ne pas converger. Au demeurant, on a le résultat facile suivant :

Proposition 1.2 *Une suite converge si et seulement si elle a un unique point d'accumulation.*

Un concept important est celui de sous-suite. Si $\{n_k\}_{k\in\mathbb{N}}$ est une suite strictement croissante d'entiers tendant vers l'infini, (i.e. croissant au-delà de tout nombre donné), la suite $\{a_{n_k}\}_{k\in\mathbb{N}}$ est appelée *sous-suite* de la suite $\{a_n\}_{n\in\mathbb{N}}$.

Proposition 1.3 *Les sous-suites d'une suite convergente convergent vers la même limite que la suite.*

Les sous-suites sont un outil utile surtout quand on ne sait pas si la suite converge. Elles servent à démontrer des propriétés des points d'accumulation grâce au fait simple suivant :

Proposition 1.4 *Si a est un point d'accumulation de la suite $\{a_n\}_{n\in\mathbb{N}}$, il existe une sous-suite qui converge vers a.*

En effet, il suffit de choisir un point x_{n_k} de la suite dans chaque intervalle (voir ci-dessous) de longueur $2/k$ entourant le point d'accumulation. (Dans $\mathbb{R}^n$, on prendra les *boules* de rayon $1/k$ centrées sur ce point.)

1.1.2 Intervalles

Définition 1.4 (Intervalles fermés et ouverts) *On appelle* intervalle fermé *ou* segment $[a,\, b]$ *l'ensemble des nombres compris entre a et b bornes comprises :*

$$[a,\, b] = \{t \in \mathbb{R} \mid a \leq t \leq b\}\,.$$

On appelle intervalle ouvert $(a,\, b)$ [1] *l'ensemble des nombres compris entre a et b bornes* non *comprises :*

$$(a,\, b) = \{t \in \mathbb{R} \mid a < t < b\}$$

Nous faisons remarquer les propriétés essentielles suivantes.

Propriété ($\mathcal{F}$) Soit I un intervalle fermé et t_n une suite d'éléments de I. Si cette suite converge, soit t sa limite, alors t appartient à I.

1. Nous avons choisi de privilégier la notation anglosaxonne $(a,\, b)$ à la notation française $]a,\, b[$ pour deux raisons : d'une part elle est beaucoup plus répandue dans la littérature, d'autre part, la notation française est sujette à des expressions peu claires. Le principal inconvénient de notre notation est qu'elle coïncide avec celle du produit scalaire. La confusion est cependant impossible dans la mesure où un intervalle a des bornes qui sont des *nombres réels*, un produit scalaire est entre *vecteurs*.

(Les intervalles fermés contiennent les limites de leurs suites convergentes.) Remarquons bien sûr qu'une suite contenue dans un fermé peut ne pas converger, par exemple la suite définie par $t_{2k} = a$, $t_{2k+1} = b$. Cette affirmation ne porte que sur les suites convergentes.

Propriété ($\mathcal{O}$) Soit I un intervalle ouvert, et t un élément de I. Il existe un réel ε positif (*non nul*) tel que $(t - \varepsilon,\, t + \varepsilon) \subset I$.

(Les intervalles ouverts contiennent un intervalle de longueur 2ε non nulle centré en chacun de leurs points.) Naturellement, ε dépend de t.

Remarque Les *demi-droites* $(-\infty,\, a]$ et $[b,\, +\infty)$ (c'est à dire respectivement l'ensemble des réels inférieurs ou égaux à a et l'ensemble des réels supérieurs ou égaux à b) satisfont encore la propriété ($\mathcal{F}$). De même, les *demi-droites ouvertes* $(-\infty,\, a)$ et $(b,\, +\infty)$ satisfont la propriété ($\mathcal{O}$). Nous ne les appellerons pas "intervalles", réservant ce nom à des intervalles *bornés*. On peut aussi faire remarquer que $\mathbb{R}$ tout entier satisfait trivialement à la fois les propriétés ($\mathcal{F}$) et ($\mathcal{O}$). Cela n'en fait pas un intervalle non plus !

Exercice 1.1 (Intervalles fermés emboîtés) *Soit $\{I_n\}_{n\in\mathbb{N}}$ une suite d'intervalles fermés emboîtés, i.e. des intervalles $[a_n,\, b_n]$ tels que si $m > n$, alors $[a_m,\, b_m] \subset [a_n,\, b_n]$. Montrer que les I_n "tendent", dans un sens que l'on précisera, vers un intervalle* non vide $[a,\, b]$. *(La propriété des intervalles fermés emboîtés*

$$\bigcap_{n\in\mathbb{N}} I_n \neq \emptyset$$

est parfois prise comme définition *axiomatique de $\mathbb{R}$ à la place de la convergence des suites de Cauchy. Elle lui est évidemment équivalente.)*

Montrer en outre que si $|b_n - a_n| \to 0$, alors il existe un réel c tel que $I_n \to c$ au sens où pour toute suite $\{t_n \in I_n\}$ d'éléments des I_n, $t_n \to c$.

Nous énonçons maintenant un théorème fondamental, bien que très facile :

Théorème 1.5 (Bolzano-Weierstrass) *Toute suite infinie d'un intervalle fermé (borné, donc) admet au moins un point d'accumulation. (On dit que les intervalles fermés (bornés) sont* compacts.*)*

Démonstration On procédera par dichotomie et en utilisant le résultat de l'exercice 1.1 ci-dessus. En effet, si on coupe l'intervalle fermé en deux, l'un au moins des deux demi-intervalles (pris fermé) contient un nombre infini de points de la suite, et on recommence.

Exercice 1.2 *Montrer que si une suite $\{I_n\}_{n\in\mathbb{N}}$ d'intervalles fermés est telle que toute intersection* finie *est non vide, c'est à dire que*

$$\forall N \in \mathbb{N},\ \bigcap_{n\leq N} I_n \neq \emptyset,$$

alors l'intersection infinie *en est non vide (il existe des points de $\mathbb{R}$ qui appartiennent à* tous *les I_n) :*

$$\bigcap_{n\in\mathbb{N}} I_n \neq \emptyset\,.$$

Faire un contre-exemple avec des intervalles ouverts, et un avec des demi-droites fermées.

1.1.3 Ouverts et fermés de $\mathbb{R}$, ou "topologie" de $\mathbb{R}$

Définition 1.5 (Sous-ensembles fermés et ouverts) *Tout sous-ensemble de $\mathbb{R}$ qui possède la propriété $(\mathcal{F})$ est dit* fermé. *Tout sous-ensemble qui possède la propriété $(\mathcal{O})$ est dit* ouvert.

Proposition 1.6 *Le complémentaire d'un fermé est un ouvert, et le complémentaire d'un ouvert est un fermé.*

Proposition 1.7 *Les intersections de fermés sont des fermés, les unions finies de fermés sont des fermés.*

Les unions d'ouverts sont des ouverts, les intersections finies d'ouverts sont des ouverts.

Démonstration : exercice.

On donne sans démonstration une caractérisation des ouverts de $\mathbb{R}$:

Proposition 1.8 *Les ensembles ouverts de $\mathbb{R}$ sont des unions, finies ou dénombrables, d'intervalles ouverts.*

On se méfiera de ce que si les ouverts sont comparativement des objets "simples", si on en croit la caractérisation ci-dessus, leurs complémentaires les fermés peuvent être épouvantablement compliqués. En particulier, il serait faux de prétendre que les fermés de $\mathbb{R}$ sont des unions finies ou dénombrables d'intervalles fermés. Un contre-exemple célèbre est donné par l'ensemble de Cantor, obtenu comme suit.

Exemple : le fermé de Cantor On fabrique une suite de fermés de la façon suivante : $C_0 = [0,\, 1]$, $C_1 = [0,\, 1/3] \cup [2/3,\, 1]$, puis on continue ainsi, chaque C_n étant une union d'intervalles fermés de longueur $1/3^n$ on enlève le tiers central ouvert de chacun d'entre-eux pour passer à C_{n+1}. L'ensemble C est la limite infinie, ou, si on veut, l'intersection de tous les C_n puisque ces ensembles sont emboîtés.

On se convainc facilement que la somme des longueurs des segments retirés vaut 1 (exercice). On pourrait croire que ne reste que l'ensemble dénombrable des points de division. Qu'il n'en est rien est montré par le fait qu'on vérifie aussi que 1/4 est intérieur à tous les C_n. En fait, si on adopte une numération triadique des nombres (i.e. à base 3), on a gardé tous les nombres qui s'écrivent sans chiffre 1. (On note $0,111\ldots$ le nombre 0,2 et donc on le retire, et de même pour tous les nombres a développement triadique fini se finissant par un 2.)

Les propriétés de cet ensemble sont assez curieuses pour qu'il ait intrigué les mathématiciens. Nous ne nous y étendrons pas d'avantage, qu'il nous suffise pour nous souvenir que si les ouverts de $\mathbb{R}$ sont "simples", les fermés peuvent en être compliqués.

Faisons remarquer la propriété essentielle (pour nous) des fermés :

Théorème 1.9 *Les fermés bornés de $\mathbb{R}$ sont compacts.*

Démonstration Soit F un fermé borné de $\mathbb{R}$. Puisqu'il est borné, on peut l'inclure dans un intervalle fermé borné. Donc toute suite infinie de F comporte au moins un point d'accumulation. Soit donc une sous-suite convergeant vers ce point d'accumulation. Comme elle est dans F qui est fermé, sa limite —le point d'accumulation choisi—, est dans F.

Nous avons enfin besoin de deux ou trois éléments de vocabulaire :

Définition 1.6 (Intérieur, adhérence, frontière) *On appelle* intérieur *d'un ensemble E, et on note $\overset{\circ}{E}$, l'union de tous les ouverts qu'il contient.*

On appelle adhérence *ou* fermeture *de E, et on note $\bar{E}$, l'intersection de tous les fermés qui le contiennent.*

On appelle frontière *de E, et on note ∂E, l'adhérence de E privée de l'intérieur de E.*

Sans autre commentaire, ces définitions peuvent sembler arbitraires. En fait, l'intérieur est *le plus grand ouvert contenu dans E*. On appelle *point intérieur* de E tout point de $\overset{\circ}{E}$, ou, de manière équivalente tout point qui est le centre d'un petit intervalle ouvert contenu dans E. Ainsi, $\overset{\circ}{E}$ est l'ensemble des points intérieurs de E.

De même, $\bar{E}$ est le plus petit fermé contenant E. On appelle *point adhérant* de E, ou valeur d'adhérence, tout point qui peut être approché arbitrairement prés par une suite de points de E. L'adhérence de E est l'ensemble de ses points adhérants.

Un point frontière de E enfin est caractérisé par le fait que tout intervalle centré en ce point contient des points de E et des points de son complémentaire.

On parle parfois d'extérieur de E pour désigner l'intérieur de son complémentaire, ou de manière équivalente le complémentaire de son adhérence.

1.1.4 Inf, min, sup, max

Ce cours concerne la minimisation d'une fonction, ou la maximisation, c'est la même chose en remplaçant la fonction par son opposée. Le minimum sera souvent approché par une suite d'approximations successives. Il faut donc avoir un peu de vocabulaire concernant cette question.

Définition 1.7 (Inf) *Étant donné un ensemble E de réels, on appelle son* inf*, et on note* $\inf_{t\in E}\{t\}$ *son* plus grand minorant *s'il existe, et on pose* $\inf_{t\in E}\{t\} = -\infty$ *sinon.*

On aura plus souvent la situation où les nombres dont on veut chercher l'inf sont des fonctions d'un entier (cas d'une suite) ou d'un réel (cas d'une fonction à minimiser sur un sous-ensemble de $\mathbb{R}$) ce qui mènera à considérer des expressions comme $\inf_{n\in\mathbb{N}}\{a_n\}$, ou $\inf_{t\in F}\{u(t)\}$. Ici, $\{a_n\}_{n\in\mathbb{N}}$ est une suite réelle, et $t \in F \subset \mathbb{R} \mapsto u(t)$ une fonction réelle, parfois notée $u(\cdot)$.

Notons que, suivant l'usage des bonnes imprimeries, Latex écrit pour nous l'inf (ou le min, cf ci-dessous) en caractères droits, (pourvu qu'on utilise la commande $\backslash\inf$) même dans les formules mathématiques, dont les lettres sont plutôt en italique, et dispose l'indice différemment dans les formules détachées du texte [2] :

$$\inf_{n\in\mathbb{N}}\{a_n\}\,, \qquad \inf_{t\in F}\{u(t)\}\,.$$

Proposition 1.10 *Il existe toujours une (des)* suite(s) minimisante(s)*, c'est à dire une sous-suite $\{a_{n_k}\}_{k\in\mathbb{N}}$ telle que $a_{n_k} \to \inf_{n\in\mathbb{N}}\{a_n\}$ dans le premier cas, et une suite de réels $\{t_k\}$ de F telle que $u(t_k) \to \inf_{t\in F}\{u(t)\}$ dans le second.*

Démonstration Soit $\bar{u} = \inf_{t\in F}\{u(t)\}$. Supposons qu'il ne puisse pas être approché arbitrairement près par des éléments de $\{u(t) \mid t \in F\}$. Il existerait $\varepsilon > 0$ tel qu'aucun élément de cet ensemble ne

2. On remarquera à cette occasion que, quand il ne l'oublie pas, l'auteur de ce texte ponctue même les équations détachées, ce qui est aussi conforme au bon usage typographique, avec l'accentuation des majuscules, et quelques petits autres traits telles les ligatures qui distinguent les belles éditions.

soit dans $(\bar{u},\, \bar{u}+\varepsilon)$. Dans ces conditions, $\bar{u}+\varepsilon$ est lui-même un minorant des $\{u(t)\}$, contredisant la définition de $\bar{u}$ comme le plus grand minorant. Ainsi, pour tout $n \in \mathbb{N}$, il existe t_n tel que $u(t_n) \in (\bar{u},\, \bar{u}+1/n)$, et les $u(t_n)$ forment la suite recherchée.

La démonstration pour le cas d'une suite est identique.

Donc, l'inf peut être approché arbitrairement près par des éléments de l'ensemble considéré. Par contre, il peut, suivant les cas appartenir ou ne pas appartenir à cet ensemble. Ainsi, par exemple, $\inf_{t\in\mathbb{R}}\{t^2\} = 0$, et cet inf est atteint en $t = 0$. On dit alors que c'est un *min*, et on écrit $\min_{t\in\mathbb{R}}\{t^2\} = 0$. Par contre, $\inf_{t>0}\{1/t\} = 0$, mais ici il n'y a pas de min. (C'est l'occasion de rappeler que $+\infty$ n'est pas un nombre réel.) Les suites minimisantes sont toutes le suites tendant vers $+\infty$.

Définition 1.8 (Minimum, minimum strict) *S'il existe $t^* \in F$ tel que $\forall t \in F$, $u(t) \geq u(t^*)$, on dit que c'est un minimum. Si $u(t) > u(t^*)$ pour tout $t \neq t^*$, le minimum est dit* strict.

On a les mêmes définitions et mêmes remarques mutatis mutandis pour le sup, qui est un max s'il est atteint.

Exercice 1.3 *Trouver les infs suivants et dire s'il y a un min (l'un d'entre-eux est difficile) :*

$$\inf_{n\in\mathbb{N}_*}\{\ln(n)\},\ \inf_{n\in\mathbb{N}_*}\{\frac{\ln(n)}{n}\},\ \sup_{n\in\mathbb{N}}\{\sin(n)\},\ \sup_{r\in\mathbb{Q}}\{\sin(r)\},\ \sup_{t\in\mathbb{R}}\{\sin(t)\}\,.$$

Donnons enfin encore une définition utile

Définition 1.9 (Minimum local) *Soit $u(\cdot)$ une fonction de $\mathbb{R}$ dans $\mathbb{R}$. On dit que t^* est un* minimum local *sur $F \subset \mathbb{R}^n$ s'il existe un nombre $\varepsilon > 0$ tel que $\forall t \in F \cap (t^*-\varepsilon,\, t^*+\varepsilon)$, on a $u(t) \geq u(t^*)$. On dit que ce minimum local est* strict *si c'est un minimum strict sur $F \cap (t^*-\varepsilon,\, t^*+\varepsilon)$*

1.2 Espace $\mathbb{R}^n$

On appelle $\mathbb{R}^n$ l'ensemble des n-uplets de nombres (qu'on notera d'habitude en colonne) qu'on appellera *vecteurs* :

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}.$$

On notera x^t le transposé de x :

$$x^t = (\ x_1 \quad x_2 \quad \cdots \quad x_n\ )\,.$$

et de même, la transposée d'une matrice M sera notée M^t.

1.2.1 Propriétés algébriques

Algèbre linéaire

$\mathbb{R}^n$ est de manière naturelle un espace vectoriel (addition et produit par un scalaire définis élément à élément). L'étude des propriétés algébriques de la structure d'espace vectoriel nous emmènerait trop

loin. Nous y renonçons. Toutefois, la maîtrise des rudiments de l'algèbre linéaire et du calcul matriciel est nécessaire pour suivre ce cours.

Les concepts nécessaires comportent la notion de combinaison linéaire, de dépendance et d'indépendance linéaire, d'opérateur linéaire, de matrice, de produit de matrices, de rang d'une matrice et ses liens avec l'espace image et le noyau de l'application linéaire, de valeur propre et de vecteur propre.

Le cours évitera la notion de valeur singulière d'une matrice et de décomposition associée. Mais la connaissance des problèmes de conditionnement est indispensable pour faire du calcul numérique.

On rappelle ici une seule définition :

Définition 1.10 (Rayon spectral) *On appelle* rayon spectral *d'une matrice le module de la valeur propre de plus grand module,*

et le théorème qui rend ce concept important :

Théorème 1.11 *La suite des puissances d'une matrice carrée tend vers zéro si et seulement si son rayon spectral est inférieur à un. Si le rayon spectral est supérieur à un, cette suite diverge.*

Structures euclidienne et métrique

On supposera aussi connue la *structure euclidienne* de $\mathbb{R}^n$, ce qui est un bien grand mot pour désigner la notion de produit scalaire

$$(x, y) = \sum_{i=1}^{n} x_i y_i \,,$$

de norme

$$\|x\| = (x, x)^{\frac{1}{2}} = \left(\sum_{i=1}^{n} x_i^2 \right)^{\frac{1}{2}} \,,$$

l'*inégalité triangulaire*

$$\|x + y\| \leq \|x\| + \|y\|$$

et la très utile *inégalité de Cauchy-Schwarz*

$$|(x, y)| \leq \|x\| \|y\| \,,$$

où l'inégalité est *stricte* si les vecteurs x et y ne sont pas colinéaires. (i.e. à moins qu'il n'existe un nombre a tel que $y = ax$.)

1.2.2 Matrices positives

Ce paragraphe concerne les formes quadratiques de $\mathbb{R}^n$. Il relève encore de la structure métrique dans la mesure où la forme quadratique (x, Ax) introduite ci-dessous définit une *norme*, parfois notée $\|x\|_A^2$ dès que A est positive définie. Il est important pour un cours d'optimisation de connaître la notion de matrice positive définie et semi-définie, et d'en connaître les principales propriétés. C'est pourquoi nous isolons ces quelques résultats dans un paragraphe à part.

Soient donc ci-dessous A, B,... des matrices carrées symétriques. On considèrera des *formes quadratiques* de la forme

$$x^t Ax = (x, Ax) = \sum_{i,j=1}^{n} a_{ij} x_i x_j$$

On fait remarquer au passage que comme $x_i x_j = x_j x_i$, si A n'était pas symétrique, la forme quadratique ci-dessus ne dépendrait que de sa *partie symétrique* $\frac{1}{2}(A + A^t)$

Définition 1.11 (Matrice positive) *La matrice A est dite* positive définie*, ce qu'on note $A > 0$, si $\forall x \neq 0$, $x^t Ax > 0$. Elle est dite* positive semi-définie*, ce qu'on note $A \geq 0$, si $\forall x$, $x^t Ax \geq 0$.*

Remarquons que de telles matrices existent, par exemple l'identité est positive définie. Plus généralement, de manière évidente, si L est une matrice (rectangulaire) quelconque, $L^t L$ est positive semi-définie, et définie si L est injective (rang égal à son nombre de colonnes), et LL^t est aussi positive semi-définie, et définie si L est surjective (rang égal à son nombre de lignes.)

De façon très naturelle, on écrira que $A > B$ si $A - B > 0$, et que $A \geq B$ si $A - B \geq 0$.

On sait que les matrices symétriques admettent des valeurs propres réelles, et sont diagonalisables (sur une base orthogonale). Soit

$$A = M^t \Lambda M \tag{1.1}$$

la forme diagonalisée de A, où Λ est la matrice diagonale des valeurs propres, et $M^{-1} = M^t$ parce que M est orthogonale. On montre facilement la propriété suivante :

Proposition 1.12 *Une matrice symétrique est positive semi-définie si et seulement si ses valeurs propres sont toutes positives ou nulles. Elle est positive définie si et seulement si ses valeurs propres sont toutes positives.*

Une conséquence est qu'une matrice positive définie est inversible, et son inverse est encore positive définie. C'est, par contre, un exercice a priori non banal de démontrer le résultat suivant :

Proposition 1.13 *Si $A > B > 0$, alors $B^{-1} > A^{-1} > 0$.*

La forme (1.1) permet de montrer divers propriétés intéressantes.

Le changement de base pour passer sous la forme diagonale respectant les normes, on voit immédiatement que, si la plus grande valeur propre de A est μ, on a toujours

$$\|Ax\| \leq \mu \|x\| ,$$

(μ est la *norme d'opérateur* de A notée $\|A\|$) et donc que $(x, Ax) \leq \mu \|x\|^2$, et aussi que si la plus petite valeur propre est ν, on a

$$(x, Ax) \geq \nu \|x\|^2 .$$

On appelle ν la *constante de coercivité* de A. Cette dernière inégalité s'écrit aussi $A \geq \nu I$. On a de même $A^{-1} \leq 1/\nu I$, de même que $\|A^{-1}\| = 1/\nu$, car la forme diagonale montre que la plus grande valeur propre de A^{-1} est $1/\nu$, et sa plus petite valeur propre $1/\mu$.

Remarquons aussi la propriété suivante :

Proposition 1.14 *Si A et B sont des matrices carrées symétriques et $0 \leq A \leq B$, alors $\|A\| \leq \|B\|$.*

En effet, soit x le vecteur propre associé à la plus grande valeur propre de A, de sorte que $Ax = \|A\|x$. On a donc $(x, Ax) = \|A\|\|x\|^2$. Mais par hypothèse, $(x, Bx) \geq (x, Ax)$, ce qui en majorant le premier terme par Cauchy Schwarz mène à $\|B\|\|x\|^2 \geq \|A\|\|x\|^2$, d'où le résultat annoncé.

En particulier, de $A \leq \beta I$ (où I est la matrice identité), on pourra conclure $\|A\| \leq \beta$.

Appelons naturellement $\Lambda^{1/2}$ la matrice diagonale des racines carrées des valeurs propres de A, et posons

$$L = \Lambda^{\frac{1}{2}} M \,.$$

On a immédiatement

$$A = L^t L \,. \tag{1.2}$$

Donc toute matrice positive semi-définie se met sous cette forme. On peut d'ailleurs remarquer que si certaines valeurs propres sont nulles, la ligne correspondante de L est nulle, et peut être *retirée*, ce qui fait de L une matrice rectangulaire, sans perdre la relation (1.2).

Une conséquence utile est la suivante :

Proposition 1.15 *Si $A \geq 0$ et $x^t A x = 0$, alors $Ax = 0$.*

Définition 1.12 (Racine carrée) *On notera*

$$A^{\frac{1}{2}} := M^t \Lambda^{\frac{1}{2}} M$$

Ainsi, $A^{1/2}$ est symétrique, positive semi-définie, et satisfait $(A^{1/2})^2 = A$. Si $A > 0$, $A^{1/2} > 0$.

1.2.3 Propriétés topologiques

On dote naturellement $\mathbb{R}^n$ de la norme et de la distance euclidiennes :

$$\|x\| = \left(\sum_1^n x_k^2 \right)^{\frac{1}{2}} , \qquad d(x, y) = \|y - x\| \,.$$

On va étendre à $\mathbb{R}^n$ les définitions et propriétés topologiques de $\mathbb{R}$, simplement en remplaçant partout les intervalles ouverts par des *boules ouvertes* :

Définition 1.13 (Boule ouverte) *Étant donnés un vecteur x de $\mathbb{R}^n$ et un réel positif r, on appelle* boule ouverte *de centre x et de rayon r, et on note $B(x, r)$, l'ensemble des vecteurs y de $\mathbb{R}^n$ qui sont à une distance inférieure à r de x :*

$$B(x, r) = \{ y \in \mathbb{R}^n \mid d(x, y) < r \} \,.$$

Les propriétés $(\mathcal{O})$ et $(\mathcal{F})$ du paragraphe 1.1.2 servent de *définition* d'un ouvert et d'un fermé de $\mathbb{R}^n$. Ce qu'on formalise dans la définition ci-dessous :

Définition 1.14 (Ouverts et fermés de $\mathbb{R}^n$) *La définition 1.5 est étendue à $\mathbb{R}^n$ en remplaçant dans la propriété $(\mathcal{O})$ l'intervalle ouvert de demi-longueur ε par une boule ouverte de rayon ε.*

On garde aussi les mêmes définitions pour l'intérieur, l'adhérence, la frontière et l'extérieur d'un ensemble. De même, les définitions de min et de min local s'étendent immédiatement.

On perd la caractérisation simple des ouverts. Mais on garde le théorème essentiel de Bolzano-Weierstrass, i.e. la compacité des fermés bornés. Aussi simple qu'il soit, nous indiquons sa démonstration en un lemme et un théorème. On passe par les pavés :

Définition 1.15 (Pavés) *On appelle* pavé fermé *un sous-ensemble de $\mathbb{R}^n$ de la forme*

$$P = \{y \in \mathbb{R}^n \mid y_k \in [x_k - r_k,\, x_k + r_k]\,, k = 1, \ldots, n\}\,,$$

où x est un vecteur de $\mathbb{R}^n$ (appelé centre *du pavé) et les r_k sont des nombres positifs.*

À lévidence, un pavé peut aussi être caractérisé par ses deux sommets extrêmes, deux vecteurs a et b de $\mathbb{R}^n$, et

$$P = \{x \in \mathbb{R}^n \mid a_k \leq x_k \leq b_k\,, k = 1, \ldots, n\}\,.$$

Lemme 1.16 *Les pavés fermés sont compacts.*

Démonstration Étant donnée une suite $\{x(k)\}_{k\in\mathbb{N}}$, extraire une sous-suite telle que la première coordonnée converge, de cette sous-suite extraire à nouveau une sous-suite telle que la deuxième coordonnée converge (dans cette nouvelle sous-suite, la première coordonnée garde la même limite que dans la première sous-suite) et ainsi de suite n fois.

Théorème 1.17 (Compacts de $\mathbb{R}^n$) *Les fermés bornés de $\mathbb{R}^n$ sont compacts.*

Démonstration Comme le théorème 1.9, en remplaçant l'intervalle fermé par un pavé fermé.

1.3 Dérivation

1.3.1 Continuité

On rappelle pour mémoire la définition :

Définition 1.16 (Continuité) *Une fonction $u(\cdot)$ est dite* continue *sur un ouvert Ω si pour tout x dans Ω, et pour toute suite $\{x_n\}$ tendant vers x, $u(x_n)$ tend vers $u(x)$.*

"Moralement", une fonction continue est une fonction dont le graphe peut être dessiné "sans lever le crayon". Malheureusement, il peut y avoir des fonctions continues bien trop irrégulières pour qu'on puisse en dessiner le graphe, puisque, par exemple, on connait des fonctions réelles continues partout sur un intervalle mais dérivables nulle part. (C'est à dire que le "graphe" n'en admet nulle part de tangente...)

Les propositions ci-dessous sont peut-être utiles à rappeler aussi :

Proposition 1.18 *L'image inverse d'un ouvert par une fonction continue est un ouvert, l'image inverse d'un fermé par une fonction continue est un fermé.*

On se rend facilement compte que la première propriété ci- dessus est *équivalente* à la définition de la continuité. Elle est souvent prise comme définition. En fait, la deuxième propriété lui est trivialement équivalente et pourrait donc aussi bien être retenue comme définition.

Proposition 1.19 *L'image (directe) d'un compact par une fonction continue est un compact.*

Il serait *faux* de croire que l'image directe d'un fermé par une fonction continue soit nécessairement un fermé. Un contre-exemple est fourni par la fonction $1/x$ qui envoie $[1,\, \infty) \subset \mathbb{R}$ (qui est fermé) sur $(0,\, 1]$, qui n'est ni ouvert ni fermé.

1.3.2 Dérivées et dérivées partielles

Soit d'abord $u(\cdot) : I \subset \mathbb{R} \to \mathbb{R}$ une fonction réelle (scalaire) d'une variable réelle définie sur un intervalle ouvert I de $\mathbb{R}$. Le lecteur est supposé connaître la notion de dérivée. Nous choisissons de réécrire sa définition sous la forme suivante :

Définition 1.17 (Dérivée) *La dérivée de $u(\cdot)$ en t est, s'il existe, le nombre (unique : exercice) $u'(t)$ tel que l'on ait*

$$u(t+\tau) = u(t) + u'(t)\tau + o(\tau)\,, \tag{1.3}$$

où $o(\cdot)$ est une fonction qui tend vers zéro plus vite que son argument. (C'est à dire que $o(\tau)/\tau$ tend vers zéro avec τ.)

Si u admet en tout point d'un intervalle ouvert I une dérivée $u'(t)$ qui est elle même une fonction *continue* de t, on dira que "u est (de classe) C^1", ou "$u \in C^1$".

Passons à une fonction de plusieurs variables, $u(\cdot) : \Omega \subset \mathbb{R}^n \to \mathbb{R}$ d'un ouvert Ω de $\mathbb{R}^n$ dans $\mathbb{R}$. Le lecteur connaît aussi la notion de dérivée partielle. Formellement, on peut écrire que $\partial u/\partial x_i$ est la dérivée de l'application partielle

$$x_i \mapsto u(x_1, x_2, \ldots, x_i, \ldots, x_n).$$

La notation $\dfrac{\partial u}{\partial x_i}$, que nous utiliserons, présente un gros inconvénient qu'il faut souligner ici. C'est l'usage du *nom donné à l'argument* (ici x_i) dans le *nom de la fonction* dérivée partielle. Ainsi, comment doit-on écrire l'expression (1.3) pour évaluer l'accroissement de u entre les points

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad \text{et} \quad \begin{pmatrix} y_1 + z \\ y_2 \\ \vdots \\ y_n \end{pmatrix} ?$$

Nous la noterons (en appelant e_1 le vecteur $(1,\, 0,\, \ldots,\, 0)^t$)

$$u(y + ze_1) = u(y) + \frac{\partial u}{\partial x_1}(y)z + o(z)\,,$$

et surtout pas

$$u(y + ze_1) = u(y) + \frac{\partial u}{\partial y_1}(y)z + o(z)\,.$$

Car on ne saurait changer le nom d'une fonction (ici la dérivée partielle par rapport à la première variable) à chaque fois qu'on change le nom de son argument. Si on faisait ainsi, que deviendrait cette dérivée partielle au point $(1,\, 1,\, \ldots,\, 1)^t$?

Pour cette raison, Dieudonné propose de noter les dérivées partielles $D_i u(\cdot)$ pour la dérivée par rapport à la variable de rang i. En cas de besoin, on invite le lecteur à avoir recours à cette notation qui évite les ambigüités.

Nous placerons d'habitude les vecteurs en colonne, et les dérivées partielles par rapport aux coordonnées d'un vecteur en ligne, réservant la notation $u'(x)$ à cette présentation :

$$u'(x) = \begin{pmatrix} \frac{\partial u}{\partial x_1} & \frac{\partial u}{\partial x_2} & \cdots & \frac{\partial u}{\partial x_n} \end{pmatrix}.$$

Ainsi nous aurons, utilisant un produit matriciel ordinaire, la formule fondamentale (1.3) préservée :

$$u(x+h) = u(x) + u'(x)h + o(\|h\|)\,. \tag{1.4}$$

Nous utiliserons la notation ∇u pour désigner le vecteur colonne des dérivées partielles, donc le transposé de u'. Ainsi (1.4) s'écrit aussi

$$u(x+h) = u(x) + (\nabla u(x), h) + o(\|h\|)\,,$$

où (y, z) désigne le produit scalaire des vecteurs y et z.

Si la fonction u est elle même vectorielle :

$$u(x) = \begin{pmatrix} u_1(x) \\ u_2(x) \\ \vdots \\ u_m(x) \end{pmatrix},$$

on fera des $u_i'(x)$ les lignes de la matrice de type $m \times n$:

$$u' = \begin{pmatrix} \frac{\partial u_1}{\partial x_1} & \frac{\partial u_1}{\partial x_2} & \cdots & \frac{\partial u_1}{\partial x_n} \\ \frac{\partial u_2}{\partial x_1} & \frac{\partial u_2}{\partial x_2} & \cdots & \frac{\partial u_2}{\partial x_n} \\ & \vdots & & \\ \frac{\partial u_m}{\partial x_1} & \frac{\partial u_m}{\partial x_2} & \cdots & \frac{\partial u_m}{\partial x_n} \end{pmatrix},$$

et la formule fondamentale (1.4) reste correcte.

Dans la notation de Dieudonné,

$$u' = \begin{pmatrix} D_1u_1 & D_2u_1 & \cdots & D_nu_1 \\ D_1u_2 & D_2u_2 & \cdots & D_nu_2 \\ & \vdots & & \\ D_1u_m & D_2u_m & \cdots & D_nu_m \end{pmatrix}.$$

1.3.3 Dérivation en chaîne et dérivées directionnelles

Dérivées en chaîne Soit $u(\cdot)$ et $v(\cdot)$ deux fonctions réelles, et posons $w(t) := u(v(t))$. On note d'habitude la fonction w comme $u \circ v$. Nous supposons en outre que u et v sont de classe C^1. Alors w l'est aussi, et la formule (1.3) donne une façon facile de démontrer (exercice) la formule connue du lecteur :

$$w'(t) = u'(v(t))v'(t)\,.$$

Remarquons qu'avec les conventions ci-dessus pour les fonctions vectorielles de variables vectorielles, cette formule demeure. Si $v : \mathbb{R}^p \to \mathbb{R}^n$ et $u : \mathbb{R}^n \to \mathbb{R}^m$, alors u' est de type $m \times n$ et v' de type $n \times p$, de sorte que le produit matriciel peut bien être fait, et w' est de type $m \times p$. Tout ceci découle nécessairement de la formule (1.4).

Dérivée directionnelle Appliquons cela à la dérivée de la "coupe" d'une fonction le long d'une droite. Soit $\xi(t) = x + th$ une droite de $\mathbb{R}^n$. Ici, x et h sont fixés dans $\mathbb{R}^n$, et t varie dans $\mathbb{R}$. Posons $U(t) = u(\xi(t))$. C'est donc la restriction de la fonction u à la droite de direction h passant par x. On a en appliquant les règles ci-dessus :

$$U'(t) = u'(\xi(t))h\,,$$

et nous l'utiliserons surtout en $t = 0$:

$$U'(0) = (\nabla u(x), h)\,. \tag{1.5}$$

Exercice 1.4 (important) *: En déduire que*
- *la* ligne de plus grande pente *est parallèle à ∇u (et opposée pour descendre),*
- *cette direction est* orthogonale à la courbe de niveau *qui est la courbe* $\{y \mid u(y) = u(x)\}$.

.

Différentielles Les règles de calcul précédentes justifient qu'il soit commode d'utiliser la notation $du = u'(x)dx$, puis si x lui-même est fonction de, disons, a, $dx = x'(a)da$, d'où en reportant $du = u'(x)x'(a)da$. Nous ne chercherons pas à justifier plus formellement l'usage de la notation différentielle qui est très commode. Rappelons que la "rigueur mathématique" ne consiste pas à respecter scrupuleusement des règles formelles de notation, elle consiste à ne pas se tromper. Il suffit donc de n'utiliser ce type de calcul qu'à bon escient, et de revenir aux théorèmes prouvés quand on n'est pas sûr de soi.

Intégration par parties Pour une fonction réelle $u(\cdot)$ dérivable *en tout* $t \in I$ où I est un intervalle ouvert, on a pour tout t_1 et t_2 de I la célèbre formule d'intégration

$$u(t_2) - u(t_1) = \int_{t_1}^{t_2} u'(t)\,dt\,. \tag{1.6}$$

Cette formule est moins naïve qu'un long usage ne le fait penser. Elle suppose suffisamment de régularité de la part de u.

Nous donnons ci-dessous une application importante de cette formule.

Développement au second ordre Soit $u(\cdot)$ une fonction réelle deux fois dérivable continument. ("De classe C^2".) Nous pouvons utiliser la formule (1.6) en y évaluant $u'(t)$ à l'aide de la formule (1.3) appliquée à u'. Il vient (exercice)

$$u(t+\tau) = u(t) + u'(t)\tau + \frac{1}{2}u''(t)\tau^2 + o(\tau^2)\,. \tag{1.7}$$

En fait, on utilisera aussi une forme un peu différente des développements limités, avec le *reste de Lagrange*, que nous rappelons ici :

$$u(t+\tau) = u(t) + u'(t)\tau + \frac{1}{2}u''(t+\theta\tau)\tau^2\,, \tag{1.8}$$

où θ est un nombre compris entre 0 et 1..

Exercice 1.5 *Étendre ces formules à un ordre n quelconque.*

Ces formules se généralisent à une fonction scalaire d'une variable vectorielle. Soit donc $u(\cdot) : \Omega \subset \mathbb{R}^n \to \mathbb{R}$. Soit D^2u la matrice symétrique de ses dérivées secondes. On a l'importante formule :

$$u(x+h) = u(x) + u'(x)h + \frac{1}{2}h^t D^2u(x)h + o(\|h\|^2)\,, \tag{1.9}$$

ou encore

$$u(x+h) = u(x) + u'(x)h + \frac{1}{2}h^t D^2u(x+\theta h)h \tag{1.10}$$

Comme toujours, ces formules se déduisent de celles du cas scalaire en regardant la fonction $U(t) = u(x+th)$.

Rien n'interdit d'étendre ce type de formule à des fonctions vectorielles (facile) et à des ordres plus élevés. Il y faut des notations plus complexes...ou beaucoup de courage.

1.4 Existence, unicité, CNS, et toutes ces sortes de choses

Ici seulement commence un cours d'optimisation, fût-il élémentaire.

1.4.1 Existence et conditions

Nous rappelons le théorème fondamental suivant :

Théorème 1.20 (Weierstrass) *Une fonction réelle continue sur un compact y atteint son minimum et son maximum.*

Démonstration Soit K un compact de $\mathbb{R}^n$ et $u(\cdot)$ une fonction continue de K dans $\mathbb{R}$. Soit encore $u^* = \inf_{x\in K}\{u(t)\}$. Soit $\{x_n\}$ une suite minimisante, i.e. telle que $u(x_n) \to u^*$ quand $n \to \infty$. Comme K est compact, il existe un point d'accumulation x^* dans K et une sous-suite $\{x_{n'}\}$ qui converge vers x^*. Comme $\{x_{n'}\}$ est une sous-suite d'une suite minimisante, $u(x_{n'})$ tend, comme $u(x_n)$, vers u^*, mais comme $u(\cdot)$ est continue et que $\{x_{n'}\} \to x^*$, $u(x_{n'}) \to u(x^*)$. La limite étant unique, on en déduit que $u(x^*) = u^*$. Donc la fonction atteint son minimum en x^*. La preuve pour le maximum se fait de la même façon. (Ou en appliquant le résultat juste démontré à $-u(\cdot)$.)

Ce théorème est bien "optimal" au sens où on pourra construire des contre-exemples (exercice) en renonçant soit au caractère fermé de K, soit à son caractère borné, soit au caractère continu de la fonction u. Par contre, on a défini des propriétés plus faibles que la continuité, comme la semi-continuité inférieure ou supérieure, qui suffisent pour avoir l'existence d'un min ou d'un max respectivement. Le contre-exemple construit en renonçant à la continuité ne sera simplement pas semi-continu inférieurement.

Cas sans contrainte

Le théorème ci-dessus parle d'un compact, donc d'un ensemble fermé. Pourtant, la majeure partie de l'analyse traditionnelle s'intéresse aux conditions (nécessaires ou suffisantes) prévalant en un minimum atteint dans un ouvert, ce qu'on appellera le cas sans contrainte. (On verra ci-dessous que la considération de contraintes fait naturellement sortir de ce cadre.)

Les dérivées ont été inventées pour le théorème suivant :

Théorème 1.21 *Soit $u(\cdot)$ une fonction dérivable d'un ouvert Ω dans $\mathbb{R}$. Une condition nécessaire pour qu'elle atteigne son minimum en $x^* \in \Omega$ est que $u'(x^*) = 0$.*

Démonstration On utilise la formule (1.4) pour évaluer u au voisinage de x^*. Si $u'(x^*) \neq 0$, il existe h tel que $u'(x^*)h < 0$ (par exemple $h = -\varepsilon u'(x^*)$) et de module suffisamment petit pour que $|u'(x^*)h| > o(\|h\|)$ (dans notre exemple, ε suffisamment petit positif), ceci par définition de $o(\cdot)$. Ainsi, $u(x^* + h) < u(x^*)$, contredisant le fait que x^* soit un minimum.

Chacun sait que si $u' = 0$, on va voir la dérivée seconde pour décider si on a un minimum, un maximum (local) ou un col. Ceci étant, il n'y a aucune nécessité à ce que la dérivée seconde soit positive pour avoir un minimum, même strict. La fonction de $\mathbb{R}$ dans $\mathbb{R}$ $u(t) = t^4$ à l'origine est un contre-exemple évident. Contre-exemple plus intéressant : la fonction

$$u(t) = \begin{cases} \exp(-\frac{1}{t^2}) & \text{si } t \neq 0\,, \\ 0 & \text{si } t = 0\,, \end{cases}$$

est continue et infiniment dérivable en 0, où elle a un minimum strict. Pourtant *toutes* ses dérivées sont nulles en 0.

Le théorème qu'on peut affirmer cependant est le suivant :

Théorème 1.22 *Une condition nécessaire pour qu'une fonction $u(\cdot)$ deux fois continument différentiable atteigne un minimum relatif en x^* est que la matrice des dérivées secondes $D^2u(x^*)$ soit positive semi définie.*

Démonstration Si $D^2u(x^*)$ n'est pas positive semi-définie, elle a au moins une valeur propre négative. On peut donc trouver un vecteur h tel que $h^tD^2u(x^*)h < 0$. Il suffit alors d'exploiter la formule (1.9) comme la formule (1.4) dans la condition nécéssaire du premier ordre avec ce vecteur d'accroissement.

On a enfin comme condition *suffisante* :

Théorème 1.23 *Si $u'(x^*) = 0$ et $u''(x^*) > 0$ (dans le cas où $x \in \mathbb{R}^n$, comprendre que la matrice carrée symétrique $D^2u(x^*)$ est positive définie), alors x^* est un minimum local.*

Démonstration Il suffit, à nouveau, d'utiliser la formule (1.9).

On étendra le théorème ci-dessus pour les fonctions de $\mathbb{R}$ à "première dérivée non nulle d'ordre pair". (En se souvenant que ce n'est qu'une condition suffisante de minimum local).

Contraintes

Terminons en examinant ce qu'on peut dire si la variable x, au lieu d'être libre de parcourir tout $\mathbb{R}^n$, est contrainte à rester dans un ensemble fermé. Commençons par examiner le cas d'une fonction $u(\cdot)$ d'une seule variable réelle, considérée sur un intervalle *fermé* $I = [a, b]$. On peut affirmer de façon évidente la proposition suivante :

Proposition 1.24 *Si u atteint son minimum sur I en t^*, alors, soit t^* appartient à l'intérieur de I et $u'(t^*) = 0$, soit $t^* = a$ et $u'(t^*) \geq 0$, soit $t^* = b$ et $u'(t^*) \leq 0$.*

On laisse le lecteur prouver ce résultat en détail.

Il s'agit de trouver une façon d'exprimer cela qui soit plus élégante, et surtout se généralise à un fermé de $\mathbb{R}^n$. Dans un premier temps, nous nous limitons à la construction ci-dessous. On verra comment la simplifier dans le paragraphe sur la convexité.

Nous considérons donc le problème

$$\min_{x \in C} u(x)$$

où C est un ensemble fermé donné.

En tout point x de C, on introduit les *directions admissibles* en x, définies de la façon suivante : $h \in \mathbb{R}^n$ est une direction admissible en x si il existe $\varepsilon > 0$ tel que, pour tout $t \leq \varepsilon$, $x + th \in C$. On laisse le lecteur démontrer que si x est dans l'intérieur de C, toute direction (tout vecteur de $\mathbb{R}^n$) est admissible en x. Par contre, il n'en va pas de même si x est sur la frontière de C.

On a le résultat suivant :

Théorème 1.25 *: Si la fonction $u(\cdot)$ est dérivable et atteint son minimum sur C en $x^* \in C$, nécessairement, on a*

$$(\nabla u(x^*), h) \geq 0 \tag{1.11}$$

pour toute direction h admissible en x^*.

Démonstration Il suffit de considérer la fonction d'une variable scalaire $U(t) = u(x^* + th)$. Pour t positif suffisament petit, $x^* + th \in C$, donc elle doit atteindre son minimum en zéro. On applique alors la proposition ci-dessus, et la règle de dérivation en chaîne.

Remarquons qu'en fait, si x^* est intérieur à C, alors u atteint son minimum sur $\overset{\circ}{C}$ en x^*, et il suffit d'appliquer le théorème 1.21. On a gagné quelquechose si x^* est un point frontière de C. Mais on n'a pas à distinguer ces cas pour énoncer le théorème.

1.4.2 Multiplicateurs, dualité

Contraintes inégalité

Nous évoquons trop rapidement cette utilisation *essentielle* du théorème précédent. Nous n'en tirerons d'algorithme numérique que dans le cas "convexe" (cf. paragraphe suivant).

Considérons le cas où l'ensemble C des x admissibles est donné par une contrainte scalaire $f(x) \leq 0$. Nous supposerons f dérivable, donc continue. Ainsi, $C = \{x \mid f(x) \leq 0\}$ est l'image inverse du fermé $(-\infty, 0]$ par une fonction continue, donc fermé. Montrons le lemme suivant :

Lemme 1.26 *Si $x \in \partial C$, toute direction h telle que $(\nabla f(x), h) < 0$, est admissible.*

Démonstration Par hypothèse, $x \in \partial C$, soit $f(x) = 0$ (exercice). Il suffit alors d'utiliser (1.4).

On en déduit le résultat suivant, que nous étendrons ensuite au cas à plusieurs contraintes :

Proposition 1.27 *Si x^* fournit le minimum de u sous la contrainte $f(x) \leq 0$, et si au minimum x^*, $\nabla f(x^*) \neq 0$, nécessairement $\nabla u(x^*) = -p\nabla f(x^*)$ où p est un nombre positif ou nul, et $pf(x^*) = 0$.*

Démonstration En effet, considérons d'abord le cas où on aurait $f(x^*) < 0$. Alors, f étant continue, x^* serait intérieur à C, et nécessairement, $\nabla u(x^*) = 0$, soit la relation annoncée avec $p = 0$. Si maintenant $f(x^*) = 0$, bien sûr cela est encore vérifié si $\nabla u(x^*) = 0$. Si non, on déduit du théorème général et du lemme ci-dessus qu'une condition nécessaire est que, pour tout h tel que $(\nabla f(x^*), h) < 0$, on ait $(\nabla u(x^*), h) \geq 0$. Ceci est bien vérifié si $\nabla u(x^*) = -p\nabla f(x^*)$ avec $p \geq 0$. Que cette dernière relation soit aussi nécessaire découle du petit calcul suivant. Supposons que cette dernière condition ne soit pas satisfaite. Alors choisissons

$$h = -\frac{\nabla u(x^*)}{\|\nabla u(x^*)\|} - \frac{\nabla f(x^*)}{\|\nabla f(x^*)\|}$$

qui est non nul d'après l'hypothèse même. D'après l'inégalité de Cauchy-Schwarz stricte, $(\nabla f(x^*), h)$ et $(\nabla u(x^*), h)$ sont tous les deux strictement négatifs, violant donc la condition nécessaire. Enfin, il découle de ce développement qu'on a toujours soit $p = 0$ soit $f(x^*) = 0$, et donc toujours $pf(x^*) = 0$.

Le résultat ci-dessus s'étend à m contraintes. Nous n'en donnerons pas la démonstration complète.

Le problème considéré est alors de trouver $x^* \in C$ tel que

$$\min_{x \in C} u(x) = u(x^*)$$

où

$$C = \{x \in \mathbb{R}^n \mid f_i(x) \leq 0\ ,\ i = 1, \ldots, m\}$$

De même que nous avons du distinguer ci-dessus les cas $f(x^*) < 0$ et $f(x^*) = 0$, il nous faut le faire ici contrainte par contrainte. C'est ce que nous faisons en introduisant la terminologie de contrainte *active en* x^*.

Soit donc $\mathcal{I}(x^*)$ l'ensemble des indices des contraintes actives en x^*, c'est à dire $\mathcal{I} = \{i \in [1 \cdots m] \mid f_i(x^*) = 0\}$. Les autres contraintes n'induisent aucune restriction sur les directions admissibles, puisque les f_j correspondantes restent négatives dans un voisinage de x^* par continuité.

Théorème 1.28 *Soient $u(\cdot)$ et $f_i(\cdot)$, $i = 1, \ldots, m$ $m+1$ fonctions (scalaires) dérivables. Si x^* minimise $u(\cdot)$ sous les m contraintes $f_i(x) \leq 0$, et s'il existe au moins un vecteur h de $\mathbb{R}^n$ tel que $\forall i \in \mathcal{I}(x^*)$, $(\nabla f_i(x^*), h) < 0$, alors nécessairement il existe m nombres p_i positifs ou nuls tels que*

$$\nabla u(x^*) + \sum_{i=1}^{m} p_i \nabla f_i(x^*) = 0\,, \text{et } \sum_{i=1}^{m} p_i f_i(x^*) = 0\,.$$

Démonstration Toute direction h de $\mathbb{R}^n$ qui satisfait $(\nabla f_i(x^*), h) < 0$ pour toutes les contraintes actives est admissible (par une extension banale du lemme 1.26). L'affirmation qui nous manque est donc la suivante, qui généralise le petit calcul effectué dans la proposition :

Lemme 1.29 (Farkas) *Soient g_0 et g_i, $i = 1 \ldots r$ des vecteurs de $\mathbb{R}^n$. Toute direction h satisfaisant $(g_i, h) < 0$, $i = 1 \ldots r$ satisfait aussi $(g_0, h) \geq 0$ si et seulement si il existe r nombres positifs ou nuls p_i tels que*

$$g_0 = -\sum_{i=1}^{r} p_i g_i\,.$$

Que cela soit *suffisant* est évident. Le lemme affirme le caractère nécessaire.

Il ne reste plus qu'à appliquer 1.11, et les deux précédents lemmes, (le dernier avec $g_0 = \nabla u(x^*)$ et $g_i = \nabla f_i(x^*)$ pour $i \in \mathcal{I}$) et à prendre $p_j = 0$ pour $j \notin \mathcal{I}$ pour obtenir le résultat annoncé. Notons aussi qu'on a bien par construction, pour tout i, soit $f_i(x^*) = 0$ (cas où $i \in \mathcal{I}$) soit $p_i = 0$ (cas contraire).

Remarquons enfin que comme nécessairement, pour tout i, $f_i(x^*) \leq 0$ (x^* est admissible) et $p_i \geq 0$, chaque produit $p_i f_i(x^*)$ est négatif ou nul. Affirmer que leur somme est nulle est donc équivalent à affirmer que chacun est nul, ce qui est connu sous le nom de *propriété des écarts complémentaires*.

Contraintes égalité

On a un résultat analogue pour les contraintes *égalité*. Nous le citons car il est très important. Sa démonstration dépend du théorème des fonctions implicites, nous ne l'évoquons pas.

Théorème 1.30 (Multiplicateurs de Lagrange) *Si x^* minimise la fonction dérivable $u(\cdot)$ sous les contraintes $f_i(x) = 0$, $i = 1, \ldots, m$, et si les vecteurs $\nabla f_i(x^*)$ sont linéairement indépendants (le* jacobien *f' de f est sujectif en x^*), il existe m nombres λ_i, $i = 1, \ldots, m$, tels que*

$$\nabla u(x^*) + \sum_{i=1}^{m} \lambda_i \nabla f_i(x^*) = 0\,. \tag{1.12}$$

Ce théorème ressemble au précédent, aux notations près : nous avons noté les multiplicateurs λ au lieu de p. La grande différence est qu'on ne sait rien sur le signe des multiplicateurs λ_i. C'est une constante en théorie de la dualité : les multiplicateurs (ou variables duales, ou variables adjointes) sont *signés* pour des contraintes *inégalité*, non signés pour des contraintes égalité. On invite le lecteur à deviner le théorème qui doit être vrai quand on a des contraintes égalité et des contraintes inégalité dans le même problème.

La façon classique d'utiliser ce théorème est la suivante. Les n équations (1.12) (une pour chaque dérivée partielle) permettent, quand tout va bien, de calculer x en fonction des m inconnues λ. En reportant dans les m équations $f(x(\lambda)) = 0$, encore si tout va bien, on peut espérer trouver les λ qui conviennent.

Ceci n'est qu'une façon de souligner qu'il y a $n + m$ inconnues (x, λ) pour $n + m$ équations. On écrit souvent ces conditions à l'aide du *lagrangien*

$$\mathcal{L}(x, \lambda) := u(x) + (\lambda, f(x))$$

sous la forme

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial x} &= 0\,, \\ \frac{\partial \mathcal{L}}{\partial \lambda} &= 0\,. \end{aligned}$$

Une façon algorithmique d'utiliser cette remarque est de résoudre ce système par la méthode de Newton. Cela conduit à l'algorithme de programmation quadratique séquentielle du paragraphe 3.4.2.

Avant de conclure, on donne une forme alternative du théorème de Lagrange :

Théorème 1.30 (Multiplicateurs de Lagrange, 2^{me} forme) *Si x^* minimise la fonction dérivable $u(\cdot)$ sous les contraintes $f_i(x) = 0$, $i = 1, \ldots, m$, il existe $m + 1$ nombres λ_i, $i = 0, 1, \ldots, m$ non tous nuls tels que*

$$\lambda_0 \nabla u(x^*) + \sum_{i=1}^{m} \lambda_i \nabla f_i(x^*) = 0\,.$$

1.4.3 Convexité

Nous donnons ici une présentation étriquée de quelques résultats de la théorie de la convexité. La théorie (plus ou moins) complète remplit des livres entiers, et concerne l'analyse des fonctions *non* différentiables. C'est dire combien notre définition en termes de dérivée seconde est restrictive.

Fonction convexe

Définition 1.18 (Fonction convexe) *Une fonction $u(\cdot)$ deux fois différentiable sera dite* convexe *si pour tout x dans le domaine considéré, $u''(x) \geq 0$ pour une fonction de $\mathbb{R}$, ou $D^2u(x) \geq 0$ pour une fonction de $\mathbb{R}^n$. Elle sera dite* strictement convexe *si les inégalités ci-dessus sont strictes.*

Théorème 1.31 *Si $u(\cdot)$ est une fonction convexe, alors $u'(x^*) = 0$ implique que x^* est un minimum (la condition nécessaire devient aussi* suffisante*). Si u est strictement convexe, ce minimum est strict.*

Démonstration Prenons d'abord le cas d'une fonction de $\mathbb{R}$ dans $\mathbb{R}$, et comme de coutume notons t sa variable, t^* le point de dérivée nulle : $u'(t^*) = 0$. Utilisons la formule (1.6) pour évaluer $u'(t)$. Du fait que $u'(t^*) = 0$, on a

$$u'(t) = \int_{t^*}^{t} u''(s)\, ds\,.$$

Comme u'' est positive ou nulle pour tout s, (ou positive) il en résulte que $u'(t)$ est négative ou nulle (ou négative) pour $t \leq t^*$, et positive ou nulle (ou positive) pour $t \geq t^*$. Utilisons à nouveau la formule (1.6), mais pour évaluer u cette fois :

$$u(\tau) = u(t^*) + \int_{t^*}^{\tau} u'(t)\, dt\,.$$

À nouveau, le signe de $u'(t)$ nous montre que l'intégrale du membre de gauche ci-dessus est toujours positive ou nulle, ou strictement positive si u est strictement convexe. Donc $u(\tau) \geq u(t^*)$ —ou $u(\tau) > u(t^*)$ si u est strictement convexe—, ce qu'il fallait démontrer.

Prenons maintenant le cas d'une fonction de $\mathbb{R}^n$. Admettons que le domaine où nous la considérons est tel que x^* peut être joint à x par un segment de droite. Pour x fixé, posons donc

$$\xi(t) = x^* + t(x - x^*)\,.$$

On pose $U(t) = u(\xi(t))$. On se souvient que $U'(t) = u'(\xi(t))(x - x^*)$ (et donc que $U'(0) = 0$), et que

$$U''(t) = (x - x*)^t D^2 u(\xi(t))(x - x^*)\,.$$

Comme D^2u est une matrice positive semi-définie (ou définie), on en déduit que U est convexe (ou strictement convexe) et on peut lui appliquer la démonstration précédente.

Théorème 1.32 *Si $u(\cdot)$ est une fonction convexe, pour tout x et y de $\mathbb{R}^n$ on a :*

$$u(y) \geq u(x) + u'(x)(y - x) \tag{1.13}$$

Démonstration Il suffit de remarquer que la fonction

$$y \mapsto u(y) - u(x) - u'(x)(y - x)$$

à la même dérivée seconde que $y \mapsto u(y)$. Elle est donc aussi convexe. Sa dérivée en $y = x$ est nulle, elle y est donc minimum par application du théorème précédent, mais elle y est nulle. D'où le résultat annoncé.

Ce résultat dit simplement que le graphe de la fonction est "au dessus" du plan tangent en un point quelconque du graphe.

Définition 1.19 (Fonction α-convexe) *Une fonction de $\mathbb{R}^n$ dans $\mathbb{R}$ deux fois continument dérivable est dite* α-convexe *(ou* coercive *de coefficient de coercivité α) où α est un nombre positif, si sa dérivée seconde est partout supérieure ou égale à α. (Comprendre, dans le cas $n > 1$, $D^2u(x) - \alpha I \geq 0$.)*

Théorème 1.33 *Pour une fonction $u(\cdot)$ α-convexe, on a pour tout x et y de $\mathbb{R}^n$*

$$[u'(y) - u'(x)](y - x) \geq \alpha\|y - x\|^2\,, \tag{1.14}$$

et aussi

$$u(y) \geq u(x) + u'(x)(y - x) + \frac{\alpha}{2}\|y - x\|^2\,. \tag{1.15}$$

Démonstration Il suffit à nouveau de considérer la fonction $U(t) = u(x + t(y - x))$, dont la dérivée seconde est $U''(t) = ((y-x), u''(x+t(y-x))(y-x)) \geq \alpha\|y-x\|^2$ puis d'utiliser cette minoration pour minorer $U'(t)$:

$$U'(t) = U'(0) + \int_0^t U''(s)\, ds \geq U'(0) + t\alpha\|y - x\|^2 .$$

L'inéquation, prise avec $t = 1$ et en utilisant (1.5), est directement (1.14), et en reportant cette minoration dans

$$U(1) = U(0) + \int_0^1 U'(t)\, dt \geq U(0) + U'(0) + \frac{\alpha}{2}\|y - x\|^2$$

on obtient (1.15).

Ceci dit que le graphe est non seulement au-dessus du plan tangent (qui est une fonction de dérivée seconde nulle) mais aussi au-dessus du "paraboloïde" tangent au graphe de dérivée seconde α. Une conséquence importante de cette propriété est la suivante :

Corollaire 1.34 *Une fonction $u(x)$ (deux fois différentiable) α-convexe tend vers l'infini quand x tend vers l'infini, et atteint son minimum sur $\mathbb{R}^n$.*

Démonstration exercice (Pour la deuxième assertion, on se ramènera au théorème de Weierstrass.)

Et les inégalités (1.14) et (1.15) ont les importantes conséquences suivantes :

Corollaire 1.35 *Si la fonction $u(\cdot)$ est α-convexe et atteint son minimum sur $\mathbb{R}^n$ en x^*, on a pour tout $x \in \mathbb{R}^n$:*

$$\|\nabla u(x)\| \geq \alpha\|x - x^*\| \tag{1.16}$$

et

$$u(x) \geq u(x^*) + \frac{\alpha}{2}\|x - x^*\|^2 \tag{1.17}$$

Démonstration On utilise (1.14) entre x et x^* en utilisant le fait que $\nabla u(x^*) = 0$:

$$(\nabla u(x), x - x^*) \geq \alpha\|x - x^*\|^2 ,$$

et on majore le membre de gauche à l'aide de l'inégalité de Cauchy-Schwarz :

$$\|\nabla u(x)\|\|x - x^*\| \geq \alpha\|x - x^*\|^2 .$$

Si $\|x - x^*\| \neq 0$, on simplifie par ce nombre pour obtenir (1.16), qui reste a fortiori vrai si $\|x - x^*\| = 0$. Quant à (1.17), c'est simplement (1.15) avec $\nabla u(x^*) = 0$.

Ensemble convexe

(C'est un barbarisme de traiter les ensembles convexes *après* les fonctions convexes)

Définition 1.20 (Ensemble convexe) *Un ensemble est dit* convexe *si chaque fois qu'il contient deux points x_1 et x_2, il contient la* corde *qui les joint, i.e. les points $\{\xi(t) = x_1 + t(x_2 - x_1)\,, t \in [0,\, 1]\}$.*

Nous ne retiendrons, sans démonstration, que deux propriétés des ensembles convexes :

Proposition 1.36 *Soit C un sous-ensemble convexe d'intérieur non vide de $\mathbb{R}^n$. En tout point $\bar{x}$ de sa frontière ∂C, il existe au moins une* normale extérieure*, c'est à dire un vecteur non nul $\nu \in \mathbb{R}^n$ tel que,*

$$\forall x \in \bar{C}, \quad (\nu, x - \bar{x}) \leq 0\,.$$

C'est à dire que tous les points de C sont "de l'autre côté" du plan orthogonal à ν passant par $\bar{x}$.

Proposition 1.37 *Tout point x a une* projection $\hat{x}$ *(notée $P_C(x)$) sur $\bar{C}$ qui est le point de $\bar{C}$ le plus proche de x. En outre, si x est extérieur à C, sa projection $\hat{x}$ appartient à ∂C. Dans ce cas $x - \hat{x}$ est une normale extérieure à C en $\hat{x}$, et cette dernière propriété caractérise la projection.*

Notons qu'à l'évidence, si $x \in \bar{C}$, d'après la définition même de la projection sur C, il est sa propre projection.

Et enfin un théorème pour montrer comment peuvent être utilisés ces concepts de nature géométrique. Pour simplifier l'écriture, nous supposons que C est fermé, ce qui nous dispense d'écrire à chaque fois que ce serait justifié $\bar{C}$.

Théorème 1.38 *Soit C un ensemble convexe fermé de $\mathbb{R}^n$. La projection sur C est une contraction au sens large, c'est à dire que quelques soient x_1 et x_2, on a $\|P_C(x_1) - P_C(x_2)\| \leq \|x_1 - x_2\|$.*

Démonstration Notons d'abord qu'en combinant la proposition 1.37 et la définition de la normale, on a pour tout x extérieur à C, et tout y dans C

$$(x - P_C(x), y - P_C(x)) \leq 0\,,$$

et cette relation reste vraie si $x \in \bar{C}$, car alors $x - P_C(x) = 0$.

On écrit donc successivement cette relation en prenant l'un des x_i pour x et la projection $\hat{x}_j := P_C(x_j)$ de l'autre pour y :

$$\begin{aligned}(x_1 - \hat{x}_1, \hat{x}_2 - \hat{x}_1) &\leq 0\,,\\ (x_2 - \hat{x}_2, \hat{x}_1 - \hat{x}_2) &\leq 0\,.\end{aligned}$$

On change le sens de l'inégalité en changeant le signe d'un des facteurs du produit scalaire dans chacune des deux lignes ci- dessus, en s'arrangeant pour faire apparaître le même facteur les deux fois :

$$\begin{aligned}(x_1 - \hat{x}_1, \hat{x}_1 - \hat{x}_2) &\geq 0\\ (\hat{x}_2 - x_2, \hat{x}_1 - \hat{x}_2) &\geq 0\end{aligned}$$

et on additionne, pour obtenir

$$(x_1 - x_2 - (\hat{x}_1 - \hat{x}_2), \hat{x}_1 - \hat{x}_2) \geq 0\,,$$

soit encore

$$(x_1 - x_2, \hat{x}_1 - \hat{x}_2) \geq \|\hat{x}_1 - \hat{x}_2\|^2\,.$$

On majore alors le premier produit scalaire à l'aide de l'inégalité de Cauchy-Schwarz pour obtenir

$$\|x_1 - x_2\|\|\hat{x}_1 - \hat{x}_2\| \geq \|\hat{x}_1 - \hat{x}_2\|^2\,,$$

et en simplifiant une fois par $\|\hat{x}_1 - \hat{x}_2\|$, qu'on peut supposer non nul sans quoi la propriété est trivialement établie, le résultat annoncé.

Il est utile pour l'intuition de remarquer que c'est de la géométrie euclidienne que nous avons fait là. Ce résultat servira dans la preuve de convergence d'un des principaux algorithmes d'optimisation à venir.

Inégalité d'Euler, théorème de Kuhn-Tucker

On va maintenant simplifier l'inégalité (1.11) en la spécialisant au cas où l'ensemble des contraintes C est convexe.

Théorème 1.39 (Inégalité d'Euler) *Soit C un convexe fermé de $\mathbb{R}^n$ et $u(\cdot)$ une fonction dérivable de C dans $\mathbb{R}$. Si u atteint son minimum sur C en x^*, alors,*

$$\forall x \in C, \quad (u'(x^*), x - x^*) \geq 0\,. \tag{1.18}$$

Démonstration Comme C est supposé convexe, et que x^* et x appartiennent tous les deux à C, il en va de même de tous les points du segment $[x^*,\, x]$, qui est donc une direction admissible. Il suffit d'appliquer (1.11)

Nous en déduisons l'important corollaire suivant :

Corollaire 1.40 *Pour une fonction convexe, la condition (1.18) est nécessaire et suffisante pour que x^* soit son minimum sur le convexe C. En outre, si elle est α-convexe, ce minimum est unique.*

Démonstration Il suffit d'uiliser la relation (1.13) pour la première assertion, et (1.15) pour la deuxième.

Théorème 1.41 (Kuhn et Tucker) *Soit $u(\cdot)$ une fonction convexe d'un convexe fermé C de $\mathbb{R}^n$ dans $\mathbb{R}$. Soient f_i, $i = 1 \ldots m$ m fonctions convexes de C dans $\mathbb{R}$. On suppose (hypothèse de Slater) qu'il existe $x_0 \in C$ tel que $f_i(x_0) < 0$, $i = 1, \ldots m$. Alors si $u \to \infty$ quand $\|x\| \to \infty$ dans l'ensemble $C \cap \{x \mid f_i(x) \leq 0, i = 1, \ldots m\}$ (ou si cet ensemble est borné), u y admet un minimum x^*, et il existe m nombres positifs ou nuls $p_i^*, i = 1, \ldots m$ tels que (on note $(p, f) = \sum_{i=1}^m p_i f_i$)*

$$\begin{aligned} &\forall p \geq 0, \forall x \in C\,, \\ &\qquad u(x^*) + (p, f(x^*)) \leq u(x^*) + (p^*, f(x^*)) \leq u(x) + (p^*, f(x)) \end{aligned} \tag{1.19}$$

Réciproquement, s'il existe $x^* \in C$ et $p^* \geq 0$ satisfaisant (1.19), alors $f_i(x^*) \leq 0$ pour $i = 1, \ldots, m$ et x^* est un minimum de u sur le domaine considéré.

Commentaires Avant de démontrer ce théorème, deux commentaires s'imposent.

1. Les inéquations (1.19) peuvent aussi s'écrire en termes du Lagrangien $\mathcal{L}(x, p) = u(x) + (p, f)$ sous la forme suivante, où on note P^+ le cône des éléments à coordonnées positives de $\mathbb{R}^n$:

   $$\forall (x, p) \in C \times P^+\,, \quad \mathcal{L}(x^*, p) \leq \mathcal{L}(x^*, p^*) \leq \mathcal{L}(x, p^*)\,.$$

   On dit que (x^*, p^*) forme un *point selle* sur $C \times P^+$.

2. Comme précédemment, le produit scalaire $(p, f(x^*))$ est toujours négatif ou nul, puisque p est à éléments positifs, et les $f_i(x^*)$ sont négatifs ou nuls. On va très vite voir que l'inégalité de gauche n'est que la condition des écarts complémentaires $(p^*, f(x^*)) = 0$, qui dit que chacun des $p_i^* f_i(x^*) = 0$, les contraintes non actives en x^* n'entrent pas dans le lagrangien (y ont un coefficient p_i^* nul).

Démonstration Nous ne démontrerons ce théorème que dans le cas simplifié où C est tout $\mathbb{R}^n$. C'est aussi dans ce cas que l'algorithme que nous en déduirons, l'algorithme d'UZAWA, est praticable.

Le domaine considéré, $\{x \mid f_i(x) \leq 0\,,\ i = 1, \ldots m\}$, est convexe fermé. Du fait qu'il est fermé, si les suites minimisantes y sont bornées, elles ont un point d'accumulation qui est un minimum de u.

Du fait qu'il est convexe, la condition de Slater implique que $x_0 - x^*$ est une direction admissible en x^*. Nous disposons donc déjà du théorème (1.28). Notons en outre que u étant convexe, les f_i aussi, et les p_i^* étant positifs ou nuls, le Lagrangien $\mathcal{L}$ est convexe en x. Donc, par le théorème (1.31), x^* est un minimum de $\mathcal{L}(x, p^*)$. Ceci donne l'inégalité de droite du point selle. L'inégalité de gauche découle immédiatement de la condition des écarts complémentaires qui fait aussi partie du théorème (1.28).

Réciproquement, supposons les inégalités du point selle vérifiées. L'inégalité de gauche donne

$$\forall p \in P^+ \quad (p, f(x^*)) \leq (p^*, f(x^*))\,.$$

Si les $f_i(x^*)$ n'étaient pas tous négatifs ou nuls, disons que $f_j(x^*) > 0$, en faisant tendre p_j^* seul vers l'infini, on violerait sûrement cette inégalité. Donc aussi $(p, f(x^*)) \leq 0$ pour tout p dans P^+. Si $(p^*, f(x^*))$ était non nul, donc négatif, en remplaçant p^* par $p^*/2$ on violerait à nouveau cette inégalité. Donc l'inégalité de gauche implique à la fois le caractère admissible de x^* et la condition des écarts complémentaires.

Du coup, l'inégalité de droite s'écrit

$$\forall x \quad u(x^*) \leq u(x) + (p^*, f(x))\,.$$

Considérons un x admissible, c'est à dire tel que $f_i(x) \leq 0$ pour tout i. Alors, $(p^*, f(x)) \leq 0$, et donc l'inégalité ci-dessus implique a fortiori $u(x^*) \leq u(x)$, ce qu'il fallait démontrer.

L'interprétation économique de ce théorème est célèbre. Si les contraintes $f_i(x) \leq 0$ représentent des ressources disponibles en quantités limitées (une par indice), le lagrangien est ce qu'il faut faire payer à l'utilisateur, les p_i représentant le "prix" des ressources rares. Ce que dit le théorème est que pour un jeu de prix adéquate, p^*, la minimisation, sans tenir compte des contraintes de ressources, de sa fonction de coût ainsi augmentée, mène l'utilisateur à une décision optimale en tenant compte des contraintes.

Chapitre 2

Recherche unidimensionnelle

2.1 Introduction

2.1.1 Objectif

Ce bref chapitre examine une question qui pourrait sembler bien naïve. Elle mérite qu'on s'y attarde un peu parce qu'elle intervient comme technique intermédiaire dans de nombreux algorithmes que nous découvrirons ensuite, c'est la "boucle intérieure" de boucles imbriquées, donc celle qu'il faut soigner le plus.

Soit donc $u(\cdot)$ une fonction d'une seule variable réelle (nous disons "de $\mathbb{R}$ dans $\mathbb{R}$"), dont nous recherchons le minimum sur un intervalle $[a,\, b]$. Nous supposons

- que le minimum t^* recherché est à l'intérieur du segment
- que la fonction u est *unimodale* sur le segment, c'est à dire d'abord décroissante jusqu'à t^*, puis croissante.

Dans la pratique, choisir a et b pourra poser des problèmes, que nous n'examinons pas ici.

Le problème est de déterminer des algorithmes permettant de trouver t^* efficacement, c'est à dire avec une bonne précision mais "pas trop" de calculs.

2.1.2 Pente et dérivée numérique

Suivant les algorithmes proposés, on peut ou non avoir besoin de la "pente" de la fonction, ou plus précisément du signe ou de la valeur de sa dérivée. Dans bien des cas, la dérivée est aussi facile à calculer que la fonction elle-même, et ceci ne pose pas de problème. Mais il y a aussi des problèmes pour lesquels le calcul de la dérivée demande significativement plus de calculs que celui de la fonction. Remarquons que pour une fonction de $\mathbb{R}^n$ dans $\mathbb{R}$, la dérivée consiste en n nombres, soit n fois plus que la fonction.

Il y a aussi des situations pour lesquelles la dérivée en tant que telle n'est pas connue. Typiquement, la fonction u peut n'être donnée que comme un gros programme informatique auquel on fournit t et qui rend $u(t)$. Il est utile de savoir que sont en train d'apparaître des outils informatiques — tels Odyssée— qui prennent en entrée le source d'un programme (FORTRAN pour Odyssée) et produisent en sortie le source (dans le même langage) d'un programme qui calcule les dérivées partielles de la fonction que définit le programme donné. Mais ces outils sont encore du domaine de la recherche, ne marchent que sous certaines conditions sur la façon dont est écrit le programme donné, etc. Supposons, par exemple, que la fonction u soit calculée à l'aide de fonctions intermédiaires tabulées et non disponibles sous forme de combinaison d'opérations et de fonctions "élémentaires". Il

n'y a aucune chance qu'on puisse en calculer formellement la dérivée.

Dans ces cas, on peut avoir recours à la "dérivation numérique", un grand mot pour quelque chose de bien élémentaire. Il s'agit tout simplement d'approximer $u'(t)$ par $(u(t+\delta)-u(t))/\delta$. La difficulté est de choisir δ, assez petit pour que ceci soit une approximation "raisonnable" de la dérivée, mais assez grand pour que la différence $u(t+\delta) - u(t)$ soit calculée de façon significative. On voit que le bon choix de δ dépend de la précision avec laquelle sont faits les calculs. Il faut parfois tatonner pour choisir ce paramètre.

Chaque fois qu'on veut seulement le signe de la dérivée (la réponse à la question "la fonction est-elle croissante ou décroissante en t ?"), pour savoir de quel côté du minimum on se trouve, le "risque" est que ce minimum soit entre t et $t + \delta$, et qu'on ne le remarque pas. Notons à ce propos qu'il est de toutes façons illusoire de chercher le minimum t^* avec une précision meilleure que celle avec laquelle on sait distinguer deux valeurs de t par la valeur qu'elles donnent à $u(t)$. (Sauf si la dérivée est calculable, et avec une meilleure précision que la fonction, cas tout à fait inhabituel.) Mais ceci impose de choisir δ suffisamment petit.

2.2 Méthodes directes

On appelle "méthodes directes" celles qui ne font pas appel au calcul de la dérivée. Nous étendrons cela à celles qui ne demandent que le *signe* de la dérivée.

2.2.1 Dichotomie

La méthode la plus simple, et pas forcément la plus mauvaise, consiste moralement à résoudre $u' = 0$ par dichotomie. On arrive ainsi à l'algorithme suivant :

Algorithme Dichotomie

1. $a_0 := a, \qquad b_0 := b$
2. $n = 1$
3. $m := (a_{n-1} + b_{n-1})/2$
4. Évaluer $u'(m)$
5. si $u'(m) < 0\,, \quad a_n := m\,, \quad b_n := b_{n-1},$
   si $u'(m) > 0\,, \quad a_n := a_{n-1}\,, \quad b_n := m,$
6. Incrémenter n de 1 et retourner au pas 3

On a omis dans l'algorithme ci-dessus le test d'arrêt, qui est un ingrédient *nécessaire* de *tout* algorithme. C'est qu'ici, on sait exactement la précision obtenue au pas n : on peut affirmer que $t^* \in [a_n, b_n]$, donc on a une précision de $(b-a)/2^n$. On peut donc déterminer *a priori* le nombre de pas à effectuer en fonction de la précision souhaitée. On comparera donc n à ce nombre avant de l'incrémenter.

On a choisi, dans la description de l'algorithme ci-dessus, une version compréhensible du point de vue mathématique, et les mathématiques n'aiment pas donner le même nom à des variables différentes. Un informaticien aurait écrit l'algorithme de la façon plus économe suivante :

Algorithme Faire N fois

1. $m := (a + b)/2$
2. Si $u'(m) < 0 \quad a := m,$
   si $u'(m) > 0 \quad b := m$

3. Recommencer au début

Où N est choisi en fonction de la précision souhaitée. Si cette précision est ε, N croit comme le logarithme (à base 2) de $1/\varepsilon$ (exercice : calculer N). Chaque pas demande un calcul de u', ou, si on doit utiliser des dérivées numériques, deux calculs de u. Le nombre de calculs de u à effectuer croit donc comme $2\log_2(1/\varepsilon) = 2\ln(1/\varepsilon)/\ln 2$.

2.2.2 Suites de Fibonacci

Par une méthode directe, on peut faire mieux que la dichotomie, soit une croissance moins rapide que $2\log_2(1/\varepsilon)$. Le principe de la méthode, dite des *suites de Fibonacci* (on verra pourquoi), est le suivant.

On part du segment $[a_0,\, b_0]$, et on suppose qu'on a calculé $u(a_0)$ et $u(b_0)$. On choisit deux point intérieurs $c_0 < d_0$. On calcule encore $u(c_0)$ et $u(d_0)$. La proposition de base est la suivante :

Proposition 2.1 *Si* $u(c_0) < u(d_0)$*, alors* $t^* \in [a_0,\, d_0]$*, si au contraire* $u(c_0) > u(d_0)$*, alors* $t^* \in [c_0,\, b_0]$.

En effet, en parcourant le segment $[a_0,\, b_0]$, dans le premier cas, la fonction u a nécessairement commencé à croitre avant d_0, et donc $t^* < d_0$, et au contraire dans le deuxième cas, elle a continué à décroitre audelà de c_0, donc $t^* > c_0$.

L'algorithme s'en déduit dans son principe : prendre pour $[a_1,\, b_1]$ le nouveau segment contenant sûrement t^*, $[a_0,\, d_0]$ ou $[c_0,\, b_0]$ suivant le cas, et recommencer. Mais au pas suivant, on connait déjà u en un point intérieur, c_0 dans le premier cas et d_0 dans le deuxième. Donc on n'aura plus qu'à choisir un seul autre point intérieur, et calculer u une seule fois, pour itérer.

La difficulté qui subsiste est dans le choix judicieux des points intérieurs à chaque pas. Le premier choix, naturel, est d'imposer qu'ils soient symétriques par rapport au milieu du segment, ainsi la longueur du segment restant après l'évaluation de u et application de la proposition sera indépendante du résultat du test. En outre, le calcul du deuxième nouveau point intérieur à chaque pas est alors trivial, puisqu'il suffit de prendre le symétrique de celui déjà connu.

Mais cette politique fait dépendre toute la suite des points utilisés du choix des deux premiers (en fait d'un des deux premiers, l'autre étant fixé par symétrie), et cette dépendance peut mener à des blocages. Par exemple, le point intérieur connu au pas n (soit c_{n-1} ou d_{n-1} suivant le cas) pourrait se retrouver au milieu, ou tout près du milieu, du segment, une situation qui empêche d'appliquer notre méthode.

Pour analyser cette question, il faut rentrer un peu dans le détail de cette suite de points.

On supposera qu'on s'est arrangé pour qu'à chaque pas, c_n soit plus grand que le milieu $(a_n + d_n)/2$ de $[a_n,\, d_n]$, et d_n soit plus petit que le milieu $(c_n + b_n)/2$ de $[c_n,\, b_n]$. Ainsi, l'algorithme peut être précisé ainsi :

Algorithme Fibonacci (pas $n+1$)

1. Évaluer u au point intérieur manquant (symétrique de celui déjà connu),
2. si $u(c_n) < u(d_n)$,
   faire $a_{n+1} := a_n, \quad b_{n+1} := d_n, \quad d_{n+1} := c_n, \quad c_{n+1} := a_{n+1} + b_{n+1} - d_{n+1}$,
   si $u(c_n) > u(d_n)$,
   faire $a_{n+1} := c_n, \quad b_{n+1} := b_n, \quad c_{n+1} := d_n, \quad d_{n+1} := a_{n+1} + b_{n+1} - c_{n+1}$.

Appelons $l_0 := b_0 - a_0$ la longueur du segment initial, et de même l_n la longueur du segment $[a_n,\, b_n]$. Du fait de la symétrie, nous avons déjà remarqué que la longueur des segments successifs

ne dépend pas du résultat du test. Examinons donc deux pas consécutifs en supposant que $u(c_{n-1}) < u(d_{n-1})$ et $u(c_n) < u(d_n)$. Ainsi, $[a_n, b_n] = [a_{n-1}, d_{n-1}]$, et $d_n = c_{n-1}$, puis $[a_{n+1}, b_{n+1}] = [a_n, d_n] = [a_{n-1}, c_{n-1}]$. Ainsi, $l_{n+1} = c_{n-1} - a_{n-1}$, tandis que $l_n = d_{n-1} - a_{n-1}$ qui, par symétrie, est aussi $l_n = b_{n-1} - c_{n-1}$. On est donc arrivé à la relation de récurence fondamentale de cette étude :

$$l_{n-1} = l_n + l_{n+1}\,. \tag{2.1}$$

En faisant tourner cette récurence *à l'envers*, c'est à dire en posant $f_k = l_{N-k}$ pour un nombre N suffisamment grand, on voit qu'on arrive à la récurrence

$$f_{k+1} = f_k + f_{k-1}\,, \tag{2.2}$$

qui initialisée en $f_0 = 0$, $f_1 = 1$ donne la suite des nombres de Fibonacci. (exercice : faire un peu de bibliographie pour retrouver qui était Fibonacci, et pourquoi il s'est intéressé à cette suite.)

On peut facilement calculer les premiers termes de la suite de Fibonacci :

$$\begin{array}{rcr} f_0 &=& 0 \\ f_1 &=& 1 \\ f_2 &=& 1 \\ f_3 &=& 2 \\ f_4 &=& 3 \\ f_5 &=& 5 \\ \vdots & & \\ f_{16} &=& 987 \\ f_{21} &=& 10946 \\ f_{26} &=& 121393 \end{array}$$

À l'évidence, la récurrence de Fibonacci génère une suite de nombres entiers positifs qui croit très vite et tends vers l'infini. On peut montrer, et c'est important, qu'elle croit comme ρ_+^k où $\rho_+ = (\sqrt{5}+1)/2$ est connu comme le *nombre d'or*, et est la racine positive de l'équation caractéristique de la récurrence (2.2) :

$$\rho^2 = \rho + 1\,.$$

On aura aussi besoin de $r_+ = 1/\rho_+ = (\sqrt{5}-1)/2$, qui satisfait, lui

$$r^2 = 1 - r\,,$$

dont on voit qu'elle est l'équation caractéristique de la récurence (2.1) écrite $l_{n+1} = l_{n-1} - l_n$, et des racines négatives des mêmes équations $\rho_- = (1-\sqrt{5})/2$ et $r_- = (-1-\sqrt{5})/2$.

La politique "optimale" consiste à avoir au dernier pas une distance δ entre les deux points intérieurs, pour être aussi près que possible de diviser le dernier intervalle par deux. Et le résultat de ce test doit laisser une longueur ε. (Où δ est celui évoqué au titre de la dérivation numérique, et ε la précision souhaitée.)

On en déduit

$$l_N = \varepsilon = l_{N-1}/2 + \delta/2\,,$$

soit $l_{N-1} = 2\varepsilon - \delta$. À partir de là, on peut remonter la suite en utilisant la récurrence (2.1) :

$$\begin{array}{lcccc} l_N & = & \varepsilon & = & f_2\varepsilon \\ l_{N-1} & = & 2\varepsilon - \delta & = & f_3\varepsilon - f_1\delta \\ l_{N-2} & = & 3\varepsilon - \delta & = & f_4\varepsilon - f_2\delta \\ l_{N-3} & = & 5\varepsilon - 2\delta & = & f_5\varepsilon - f_3\delta \\ \vdots & & & & \\ l_0 & = & & & f_{N+2}\varepsilon - f_N\delta \end{array}$$

On doit donc

Algorithme Fibonacci (complet)

1. Déterminer le plus petit entier N tel que $f_{N+2}\varepsilon - f_N\delta > l_0 = b_0 - a_0$,
2. calculer $\varepsilon' = (l_0 + f_N\delta)/f_{N+2}$,
3. choisir $d_0 = a_0 + f_{N+1}\varepsilon' - f_{N-1}\delta$ et $c_0 = b_0 - f_{N+1}\varepsilon' + f_{N-1}\delta$,
4. dérouler l'algorithme de Fibonacci ci-dessus, de $n = 0$ à $N - 1$. (Soit N pas)

Dans cet algorithme, on a évalué u en $N + 3$ points, pour réduire le segment contenant t^* dans un rapport de l'ordre de $1/(\rho_+)^N$. Il est prudent de suivre la suite des nombres de Fibonacci pour placer les points intermédiaires à chaque pas plutôt que de se fier au "symétrique", ce qui évite de laisser s'accumuler de petites erreurs. En effet, cette procédure est *très sensible* à de petites erreurs sur la position des points intérieurs, comme la suite de l'analyse va nous le montrer.

2.2.3 Section dorée

La suite de Fibonacci "à l'envers" engendrée par (2.1) est de la forme

$$l_n = \alpha_+ r_+^n + \alpha_- r_-^n$$

où α_+ et α_- dépendent des deux termes initiaux. Comme r_+ est de module inférieur à 1, le terme en r_+^n tend rapidement vers 0. Par contre, r_- est de module supérieur à 1, et même plus précisément $r_- < -1$, de sorte que le terme en r_-^n diverge rapidement avec des signes alternés. C'est ce qui explique la grande sensibilité de l'algorithme "de Fibonacci" au choix de c_0 et d_0.

La seule façon de pouvoir poursuivre l'algorithme un nombre arbitraire de pas est de s'assurer que $\alpha_- = 0$, c'est à dire de choisir $l_1 = r_+ l_0$. Ainsi, $l_2 = (1 - r_+)l_0 = r_+^2 l_0$, …, $l_n = r_+^n l_0$.

On a ainsi un algorithme bien plus facile à mettre en œuvre, qui consiste à appliquer l'algorithme de Fibonacci ci-dessus, en plaçant à chaque pas d_n à une distance $r_+ l_n$ de a_n, et c_n à une distance $(1 - r_+)l_n = r_+^2 l_n$, où on rappelle que

$$\begin{aligned} r_+ &= \frac{\sqrt{5}-1}{2} \simeq 0,618\,, \\ 1 - r_+ = r_+^2 &= \frac{3-\sqrt{5}}{2} \simeq 0,382\,. \end{aligned}$$

Cet algorithme est à très peu de chose près aussi efficace que le précédent et beaucoup plus simple. On n'a pas besoin de déterminer à l'avance le nombre de pas qu'on va effectuer, il suffit de mettre un test d'arrêt en comparant la longueur du segment $[a_n, b_n]$ restant après le pas n à la précision ε désirée. Aussi le récapitulons-nous à fin de référence.

Algorithme Section dorée

1. Faire $[a_0, b_0] = [a, b]\,, \quad l_0 = b - a\,, \quad c_0 = a + r_+^2 l_0\,, \quad d_0 = a + r_+ l_0.$
2. Évaluer u en a_0, b_0, c_0, d_0.
3. Faire $n := 0$.
4. Calculer $l_{n+1} = r_+(b_n - a_n)$.
5. Si $u(c_n) < u(d_n)$,
   faire $\quad a_{n+1} := a_n, \quad b_{n+1} := d_n, \quad d_{n+1} := c_n, \quad c_{n+1} := a_{n+1} + r_+^2 l_{n+1},$
   si $u(c_n) > u(d_n)$,
   faire $\quad a_{n+1} := c_n, \quad b_{n+1} := b_n, \quad c_{n+1} := d_n, \quad d_{n+1} := a_{n+1} + r_+ l_{n+1}.$
6. Si $l_{n+1} < \varepsilon$ ou $(\sqrt{5} - 2)l_{n+1} < \delta$, donner $a_{n+1} < t^* < b_{n+1}$ et arrêter,
   si non, évaluer u en celui des points c_{n+1} ou d_{n+1} où elle n'est pas encore connue, incrémenter n de 1 et retourner en 4.

exercice : Pourquoi le deuxième critère dans le test d'arrêt ? (Qui limite la précision qu'il est possible d'obtenir à un peu plus de 4δ. On peut, si vraiment nécessaire, ajouter trois pas de dichotomie.)

2.3 Méthodes indirectes

Comme nous l'avons dit, nous regroupons sous ce titre douteux les méthodes qui reposent de façon plus essentielle sur le calcul de dérivées. Rappelons que dans bien des cas, la recherche unidimensionnelle est effectuée dans un algorithme de minimisation dans $\mathbb{R}^n$ pour trouver le meilleur point dans une *direction de recherche* h choisie par ailleurs. Dans ces cas, notre fonction $u(t)$ de ce chapitre est en fait de la forme $u(x + th)$, et ce qui joue le rôle de $u'(t)$ est la dérivée directionnelle $(\nabla u(x + th), h)$.

2.3.1 "Backtracking"

Sous ce vocable anglosaxon, nous visons une méthode simplissime dont l'objectif n'est pas vraiment de trouver le minimum en t de $u(t)$, mais un point "suffisamment bon". Cette méthode sera recommandée dans l'algorithme de Newton "protégé" de l'optimisation multivariable.

On est au voisinage d'un t donné, et on a calculé $u'(t)$. On cherche un nouveau $t' = t + \theta$. L'idée est de partir avec une estimée trop lointaine $t + \theta_0$, avec $\theta_0 u'(t) < 0$, et de reculer jusqu'à ce que la pente $(u(t + \theta) - u(t))/\theta$ soit suffisamment grande en valeur absolue, comparée à $u'(t)$. On va donc choisir deux nombres positifs $r < 0,5$ et $\rho < 1$ (en pratique on prend ρ vers $0,8$) et de faire

Algorithme

1. choisir θ_0 "siffisamment grand"
2. faire $n := 0$,
3. itérer jusquà ce que $u(t + \theta_n) \leq u(t) + ru'(t)\theta_n : \theta_{n+1} = \rho\theta_n$.

2.3.2 Méthode de Newton

La méthode de dichotomie présentée comme méthode "directe" consiste en fait à résoudre l'équation $u'(x) = 0$ par une méthode de dichotomie. On peut bien sûr résoudre cette même équation par la méthode de Newton. On rappelle que cette méthode itérative consiste à prendre comme prochaine estimée de la solution d'une équation le point qui serait la solution si la fonction à annuler coïncidait avec son approximation au premier ordre.

On verra la méthode de Newton dans un cas un peu plus général au chapitre suivant, nous nous contentons ici d'écrire l'algorithme auquel elle mène pour l'application présente :

Algorithme Newton unidimensionnel

1. Choisir t_0 suffisamment proche de t^*,
2. faire $n := 0$,
3. itérer jusqu'à ce que $|u'(t_n)| \leq \varepsilon$

$$t_{n+1} = t_n - u''(t_n)^{-1} u'(t_n)\,.$$

(Ajouter une clause limitant le nombre total d'itérations permis serait une prudence élémentaire.)

On sait que l'algorithme de Newton converge quadratiquement (on le reverra au chapitre suivant), ce qui est *très rapide*. Sa grande faiblesse est qu'il est très sensible au choix de la condition initiale, et peut facilement être mis en défaut si elle est trop éloignée de la solution recherchée. Aussi, on suggère fréquemment de ne l'utiliser que pour affiner une solution approchée obtenue avec une méthode plus rustique.

On voit aussi sur la formule que cet algorithme aura des problèmes si $u''(t)$ est trop proche de 0 au voisinage de t^*. Il est toujours plus difficile de trouver un minimum très "plat" (avec une très petite dérivée seconde) qu'un minimum bien marqué. Mais cet algorithme, utilisant explicitement la dérivée seconde, peut y être plus sensible qu'un autre. Il peut être prudent de tester le module de cette dérivée seconde (qui devrait rester positive en tout état de cause).

2.3.3 Approximation polynômiale

Approximation parabolique

Dans beaucoup d'applications, notamment celles liées au gradient "à pas optimal" ou "conjugué", on connait u et sa dérivée en a. Si dans le cas du gradient conjugué il est important de trouver le minimum avec une certaine précision, il n'en va pas de même pour le gradient à pas optimal, et pour quelques autres algorithmes où une approximation moyenne de l'optimum suffit à chaque pas. (Relaxation,...) Dans ces cas, on peut utiliser la méthode suivante.

Supposons que nous connaissions u et sa dérivée en a. On choisit b tel que $[a,\, b]$ ait une bonne chance d'encadrer le minimum t^* (mais ce n'est pas critique dans cette méthode), on évalue $u(b)$, on approxime $u(t)$ par la parabole qui aurait même valeur en a et b et même dérivée en a. Et on prend comme estimée de t^* le point $\hat{t}$ où cette parabole atteint son minimum.

Ceci mène à l'estimée suivante :

$$u(t) \simeq u(a) + (t-a)u'(a) + \alpha\frac{(t-a)^2}{2}\,,$$

soit

$$t^* \simeq \hat{t} = a - \frac{u'(a)}{\alpha}\,,$$

où α est estimée par

$$u(b) = u(a) + (b-a)u'(a) + \alpha\frac{(b-a)^2}{2}$$

ce qui conduit finalement au choix

$$\hat{t} = a - \frac{u'(a)(b-a)^2}{2[u(b) - u(a) - (b-a)u'(a)]}\,.$$

On ne traite pas cette méthode comme un algorithme au sens propre du terme, car l'idée n'est pas de l'appliquer itérativement, mais une seule fois dans un algorithme englobant qui requiert d'aller au minimum dans une direction donnée à chacun de ses pas. On évite ainsi des itérations emboitées, au prix d'une approximation parfois médiocre de ce minimum.

Au fur et à mesure que l'algorithme englobant se rapproche du minimum recherché, cette méthode approxime de mieux en mieux le t^* du pas en cours, comme l'indique le résultat ci-dessous.

Théorème 2.2 *Si la fonction u est trois fois continument différentiable, α-convexe sur $[a, b]$, et si ce segment contient t^* et a une longueur $b-a$ qui tend vers zéro comme h, $|\hat{t} - t^*|$ tend vers zéro comme h^2. (Donc l'erreur relative $|\hat{t} - t^*|/h$ tend vers zéro comme h.)*

Démonstration Nous donnons une preuve directe de ce résultat, mais il peut aussi se déduire de la preuve plus simple que nous donnons du théorème suivant.

Développons u au voisinage de t^*, en notant $u^* = u(t^*)$ et $u''^* = u''(t^*) > \alpha$:

$$u(t) = u^* + \frac{1}{2}u''^*(t - t^*)^2 + \varepsilon(h^3)$$

où $\varepsilon(z)$ désigne une quantité qui tend vers zéro comme z. On a aussi

$$u'(t) = u''^*(t - t^*) + \varepsilon(h^2)\,.$$

Reportons ces expressions dans la formule pour $\hat{t}$. On trouve facilement

$$u'(a)(b-a)^2 = u''^*(a - t^*)(b-a)^2(1 + \varepsilon(h))$$

et

$$u(b) - u(a) - u'(a)(b-a) = \frac{1}{2}u''^*(b-a)^2(1 + \varepsilon(h))\,,$$

soit

$$\hat{t} = a - \frac{u''^*(a - t^*)(b-a)^2(1 + \varepsilon(h))}{\frac{1}{2}u''^*(b-a)^2(1 + \varepsilon(h))} = a - (a - t^*)(1 + \varepsilon(h)) = t^* + \varepsilon(h^2)\,.$$

Donc, si cette procédure est utilisée, par exemple, dans un algorithme de gradient, au début, $\hat{t}$ est une approximation grossière du t^* du pas en cours, mais on sait que cela suffit, et au fur et à mesure que l'algorithme progresse, l'erreur relative tend vers zéro, permettant une bonne convergence de l'algorithme global.

Approximations cubiques

En fait, la méthode la plus fréquemment utilisée est celle de l'approximation cubique, où la fonction u est approximée par une cubique, donc. En effet, sa dérivée est alors un polynôme de degré deux, et on a encore une formule explicite pour son minimum $\hat{t}$. Ainsi, au prix de calculs guère plus lourds, on a une estimée meilleure d'un ordre (une erreur relative d'ordre deux), ce qui est excellent.

On prendra donc pour approximation de u

$$u(t) \simeq \alpha t^3 + \beta t^2 + \gamma t + \delta\,,$$

et donc

$$u'(t) \simeq 3\alpha t^2 + 2\beta t + \gamma\,,$$

ce qui mène à l'approximation $t^* \simeq \hat{t}$ avec

$$\hat{t} = \frac{\sqrt{\beta^2 - 3\alpha\gamma} - \beta}{3\alpha} .$$

(On a supposé que $\gamma = u'(0) < 0$ et $\beta = u''(0) > 0$.)

Il reste à calculer α, β et γ. Plusieurs méthodes sont possibles (d'où le pluriel dans le titre de ce sous-paragraphe).

Soit, en calculant $u(b)$, on a facilement aussi la dérivée $u'(b)$, et cela fait assez d'information, soit on considère que calculer une dérivée est plus difficile que la fonction elle même, et on peut alors préférer calculer $u(c)$ en un point intermédiaire $c \in [a,\, b]$.

Dans le premier cas, on peut évaluer les constantes par les formules suivantes :

$$\alpha = \frac{2(u(b) - u(a)) - (b-a)(u'(b) + u'(a))}{(b-a)^3} ,$$

$$2\beta = \frac{u'(b) - u'(a)}{b-a} - 3\alpha(b+a) ,$$

$$\gamma = u'(a) - 3\alpha a^2 - 2\beta a ,$$

voire, pour préserver la symétrie (ce qui peut être numériquement utile) la formule symétrisée pour γ en faisant $1/2$ de cette expression plus la même en b.

Dans le deuxième cas, nous introduisons les quantités

$$\Delta(a,b) = \frac{u(a) - u(b) - u'(a)(a-b)}{(a-b)^2}$$

et de même pour $\Delta(a,c)$, et on peut montrer les formules suivantes :

$$\alpha = \frac{\Delta(a,c) - \Delta(a,b)}{b-c} ,$$

$$\beta = \frac{c\Delta(a,b) - b\Delta(a,c)}{b-c} - 2a\alpha ,$$

$$\gamma = u'(a) - 3\alpha a^2 - 2\beta a .$$

On a le résultat logique :

Théorème 2.3 *Si la fonction u est quatre fois continument différentiable, α-convexe sur $[a,\, b]$, et si ce segment contient t^* et a une longueur $b-a$ qui tend vers zéro comme h, $|\hat{t} - t^*|$ tend vers zéro comme h^3. (Donc l'erreur relative $|\hat{t} - t^*|/h$ tend vers zéro comme h^2.)*

Démonstration Appelons $\hat{u}(t)$ notre approximation polynômiale de u, et $\varepsilon(t) = u(t) - \hat{u}(t)$ l'erreur d'approximation. Le fait essentiel est le suivant :

Proposition 2.4

$$\forall t \in [a,b] , \quad \varepsilon'(t) \to 0 \quad \text{comme } h^3 .$$

En effet, le développement de Taylor à l'ordre 3 en a avec reste exact nous apprend que u peut sécrire

$$u(t) = \bar{u}(t) + \frac{(t-a)^4}{24} u^{(4)}(t') \,, \quad t' \in [a,t]$$

où $\bar{u}$ est un polynôme de degré 3. Si, dans les formules servant à calculer $\hat{u}$ on remplace $u(b)$ et $u(c)$ par $\bar{u}(b)$ et $\bar{u}(c)$, on obtient exactement $\bar{u}$ à la place de $\hat{u}$. Mais on peut vérifier que $\hat{u}(t)$ sécrit aussi (exercice)

$$\hat{u}(t) = u_a(t) + \frac{(t-a)^2(t-c)}{(b-a)^2(b-c)} u(b) + \frac{(t-a)^2(t-b)}{(c-a)^2(c-b)} u(c)$$

où $u_a(t)$ est un polynôme de degré 3 qui ne dépend que de $u(a)$ et $u'(a)$, mais pas de $u(b)$ et $u(c)$. (u_a et sa dérivée coincident avec $u(a)$ et $u'(a)$ en a, et u_a s'annulle en b et en c, ce qui au vu des formules ci-dessus le définit complètement.) Les quantités $u(b)$ et $u(c)$ interviennent (linéairement) dans la formule ci-dessus avec des coefficients uniformément bornés quand $h \to 0$. Donc les coefficients du polynôme $\hat{u}$ approchent ceux de $\bar{u}$ comme $\bar{u}$ approche u en b et en c, c'est à dire comme h^4. Donc leur différence *et sa dérivée* approchent zéro comme h^4. Ainsi $\hat{u}'$ approche $\bar{u}'$ en h^4 uniformément en t. Mais le développement de Taylor de u' en a nous apprend que $\bar{u}'$ approche u' comme h^3 uniformément en t. La proposition est démontrée.

Par définition de $\hat{t}$ on a $\hat{u}'(\hat{t}) = 0$, et donc $u'(\hat{t}) = \varepsilon'(\hat{t})$. En outre, $u'(t^*) = 0$, et donc, comme u est supposée α-convexe, par (1.14) $u'(\hat{t}) \geq \alpha|\hat{t} - t^*|$. Soit bien

$$|\hat{t} - t^*| \leq \frac{1}{\alpha} \varepsilon'(\hat{t})$$

d'où, avec la proposition, le résultat annoncé.

On remarque que cette méthode de preuve, élémentaire si non élégante, s'étend à un ordre quelconque.

Il reste une question ouverte : celui du meilleur choix de c dans la dernière méthode. Y a-t-il un choix de c qui annule le terme d'ordre 3 dans l'erreur $\hat{t} - t^*$? C'est peu probable, parce que ce terme est un polynôme de degré trois en t^*, et fait donc intervenir quatre coefficients. Mais à notre connaissance, cette question n'a jamais été regardée. Il faut dire que les calculs demandent à être faits à la machine, car ils sont *gros* ! (Il faut développer u au moins à l'ordre cinq.)

Chapitre 3

Optimisation dans $\mathbb{R}^n$

3.1 Bonnes fonctions

Nous présentons ci-dessous divers algorithmes de recherche de minimum dont certains, tel le gradient à pas optimal, sont très robustes, i.e. convergent dans bien des situations délicates. Cependant, tant parce qu'on ne peut pas toujours faire beaucoup mieux que par souci de simplicité mathématique, nous ne démontrerons jamais la convergence que pour une classe de fonctions u assez restreinte, que nous appellerons les *bonnes fonctions* (appellation totalement indigène).

Définition 3.1 (Bonnes fonctions) *Nous appellerons* bonnes fonctions *les fonctions $u(\cdot)$ de $\mathbb{R}^n$ dans $\mathbb{R}$ α convexes et de dérivée première β-Lipshitz-continue :*

$$\forall x, y \in \mathbb{R}^n, \quad \|u'(y) - u'(x)\| \leq \beta \|y - x\| \,. \tag{3.1}$$

Si nous définissons la convexité via la positivité de la dérivée seconde, —mais la définition ci-dessus est plus générale— c'est dire que $u(\cdot)$ doit être deux fois continument dérivable, et qu'il doit exister deux réels positifs α et β tels que

$$\forall x \in \mathbb{R}^n, \quad \alpha I \leq D^2 u(x) \leq \beta I \,,$$

ou encore que les valeurs propres de $D^2 u$ sont comprises pour tout x entre α et β.

Rappelons que $u(\cdot)$ étant α-convexe, elle satisfait outre (3.1) les inégalités (1.14), (1.15), (1.16) et (1.17), et son minimum est atteint sur $\mathbb{R}^n$.

3.2 Optimisation non contrainte

3.2.1 Relaxation

Nous commençons par un algorithme "direct", c'est à dire qui ne nécessite pas le calcul des dérivées de u. Il n'est à recommander que si ce calcul est vraiment difficile, et (si possible) seulement en petite dimension.

L'algorithme est excessivement simple (voire simpliste), et consiste à faire des minimisations unidimensionnelles en chacune des variables x_i successivement. Un test d'arrêt raisonnable porte sur la décroissance de u aprés qu'on ait fait cette minimisation sur chacune des coordonnées.

Soit donc x^k une estimée de la solution. Nous introduisons les "pas fractionnaires" $x^{k+i/n}$, que nous noterons $x^{k,i}$ pour simplifier, de la façon suivante : $x^{k,1}$ ne diffère de x^k que par sa première

coordonnée obtenue en minimisant $u(\cdot)$ par rapport à cette première coordonnée toutes les autres étant figées à leur valeur dans x^k. On passe ensuite à $x^{k,2}$ en figeant toutes les coordonnées à leur valeur dans $x^{k,1}$, sauf la deuxième par rapport à laquelle on minimise u, et ainsi de suite. Et on posera $x^{k+1} = x^{k,n}$. Nous décrivons formellement cet algorithme. Nous notons e_i le vecteur de base numéro i de $\mathbb{R}^n$.

Algorithme Relaxation

1. Choisir une estimée initiale x^0 et faire $k := 0$
2. faire $x^{k,0} := x^k$.
3. pour $i = 1 \dots n$, calculer $x^{k,i}$ par
$$u(x^{k,i}) = \min_{\theta} u(x^{k,i-1} + \theta e_i) .$$
4. faire $x^{k+1} := x^{k,n}$
5. si $u(x^k) - u(x^{k+1}) < \varepsilon$, stop, sinon incrémenter k de 1 et retourner en 2.

Théorème 3.1 (Convergence de l'algorithme de relaxation) *Si $u(\cdot)$ est une bonne fonction (au sens de la définition ci-dessus), l'algorithme de relaxation converge vers l'argument x^* du minimum.*

Démonstration On remarquera d'abord que par l'α-convexité, et à cause de (1.17) par exemple, la suite $\{x^k\}$ est bornée.

Par construction, on a $u(x^{k,i}) < u(x^{k,i-1})$, et donc aussi $u(x^{k+1}) < u(x^k)$. Comme par α-convexité, u est bornée inférieurement, $\|u(x^k) - u(x^{k+1})\| \to 0$ quand $k \to \infty$. Plus précisément, la définition de $x^{k,i}$ implique que $u'(x^{k,i})e_i = 0$, de sorte que l'inégalité (1.15) donne

$$u(x^{k,i-1}) - u(x^{k,i}) \geq \frac{\alpha}{2}\|x^{k,i-1} - x^{k,i}\|^2 .$$

En sommant ces inégalités de $i = 1$ à n, il vient

$$u(x^k) - u(x^{k+1}) \geq \frac{\alpha}{2}\|x^k - x^{k+1}\|^2 .$$

Donc, en particulier, $\|x^k - x^{k+1}\|$ tend vers zéro, et donc aussi chacune de ses composantes $\|x^{k,i-1} - x^{k,i}\|$.

La deuxième inégalité d'α-convexité (1.14) donne, en utilisant $\nabla u(x^*) = 0$:

$$\alpha\|x^k - x^*\|^2 \leq (\nabla u(x^k), x^k - x^*) = \sum_{i=1}^{n} u_i'(x^k)(x_i^k - x_i^*) ,$$

où $u_i' = u'(x)e_i$ désigne bien sur la dérivée partielle en x_i. Mais à nouveau, par la définition de $x^{k,i}$, $u_i'(x^{k,i}) = 0$. De sorte que l'inégalité (3.1) donne

$$\|u_i'(x^k)\| \leq \beta\|x^k - x^{k,i}\| .$$

Nous utilisons cette évaluation dans l'inégalité précédente, pour obtenir

$$\alpha\|x^k - x^*\|^2 \leq \beta \sum_{i=1}^{n} \|x^k - x^{k,i}\| \, \|x_i^k - x_i^*\| .$$

Nous savons que les $\|x^k - x^{k,i}\|$ tendent vers zéro et que les $\|x_i^k - x_i^*\|$ sont bornés. Donc $\|x^k - x^*\| \to 0$, ce qu'il fallait démontrer.

Cette preuve ne dit pas que la méthode converge "bien". Elle converge même souvent *très mal*, et il n'est guère réaliste d'en espérer une bonne approximation de x^*. Tout au plus permet-elle d'améliorer parfois significativement la performance $u(x)$ d'une estimée initiale à moindre frais. Les méthodes qui suivent sont presque toujours préférables.

3.2.2 Gradient à pas optimal

L'algorithme

L'algorithme de gradient à pas optimal consiste à se déplacer "dans la direction de plus grande pente", c'est à dire dans la direction opposée au gradient, jusqu'au point "le plus bas" dans cette direction. On a ainsi la description formelle suivante :

Algorithme Gradient à pas optimal

1. Choisir une estimée initiale x^0, faire $k := 0$.
2. Calculer $\nabla u(x^k)$. Si $\|\nabla u(x^k)\| < \varepsilon$ stop. Si non,
3. Calculer $x^{k+1} := x^k - \theta^k \nabla u(x^k)$ où le pas $\theta^k > 0$ est déterminé par

$$u(x^{k+1}) = \min_{\theta} u(x^k - \theta \nabla u(x^k))$$

4. incrémenter k de 1 et retourner en 2

Théorème 3.2 (Convergence de l'algorithme du gradient à pas optimal)
Soit $u(\cdot)$ une bonne fonction. L'algorithme du gradient à pas optimal converge vers l'argument x^* du minimum de u.

Demonstration Nous décomposons cette preuve pour souligner ce que nous apporte chaque hypothèse.

1. La fonction u décroit à chaque pas, mais comme elle est α-convexe, elle est bornée inférieurement. Donc $\|u(x^k) - u(x^{k+1})\| \to 0$.
2. Comme u'' est bornée par βI, et plus précisément grâce à l'inégalité (3.1), la décroissance au pas k est d'au moins $\|\nabla u(x^k)\|^2/2\beta$, comme l'indique le lemme que nous démontrons séparément ci-dessous, utilisé avec $\hat{h} = -g/\|g\|$, et donc $\gamma = 1$. En conséquence, en tenant compte du (1) ci-dessus, $\nabla u(x^k) \to 0$.
3. Par l'α-convexité, et plus précisément par l'inégalité (1.16), on en déduit que $x^k \to x^*$, ce qu'il fallait démontrer.

Remarque 3.1 *On remarque qu'on n'a pas vraiment besoin de l'α-convexité de u. Il suffit (exercice) de supposer que u est strictement convexe et tend vers l'infini à l'infini.*

Il reste à démontrer le lemme. Nous le démontrons dans un cadre un peu plus large, qui nous servira par la suite.

Lemme 3.3 *Soit $g = \nabla u(x)$, et $\hat{h}$ un vecteur de $\mathbb{R}^n$ de norme unité, satisfaisant l'inégalité*

$$(g, \hat{h}) \leq -\gamma \|g\| \tag{3.2}$$

avec $0 < \gamma \leq 1$.

Soit $t^+ \in \mathbb{R}^+$ déterminé par

$$u(x + t^+\hat{h}) = \min_{t \in \mathbb{R}^+} u(x + t\hat{h}) \,,$$

et soit $x^+ = x + t^+\hat{h}$.

Sous l'hypothèse (3.1), on a

$$u(x) - u(x^+) \geq \frac{\gamma^2}{2\beta} \|g\|^2 \tag{3.3}$$

Démonstration Introduisons la fonction $U(t) = u(x + t\hat{h})$. Notons que

$$U'(t) = (\nabla u(x + t\hat{h}), \hat{h})$$

et donc que $U'(0) \leq -\gamma\|g\|$.

Par (3.1), nous avons

$$\|\nabla u(x + t\hat{h}) - g\| \leq \beta t \,,$$

soit, avec l'inégalité de Cauchy-Schwarz,

$$(\nabla u(x + t\hat{h}) - g, \hat{h}) \leq \beta t$$

soit encore

$$U'(t) = (\nabla u(x + t\hat{h}), \hat{h}) \leq (g, \hat{h}) + \beta t \leq -\gamma\|g\| + \beta t \,.$$

On utilise cette minoration pour évaluer $U(\tau)$:

$$U(\tau) \leq U(0) + \int_0^\tau (\beta t - \gamma\|g\|)\, dt = u(x) + \beta\frac{\tau^2}{2} - \gamma\|g\|\tau \,.$$

Enfin, on utilise cette minoration en $\tau = \frac{\gamma}{\beta}\|g\|$ pour obtenir

$$u(x^+) = \min_\tau U(\tau) \leq U(\frac{\gamma}{\beta}\|g\|) \leq u(x) - \frac{\gamma^2}{2\beta}\|g\|^2 \,,$$

ce qu'il fallait démontrer.

Cet algorithme est très robuste, et beaucoup plus efficace que l'algorithme de relaxation. On démontre sa convergence pour les "bonnes fonctions", mais c'est un "bon algorithme" au sens où "une bonne théorie est une théorie qui continue à donner de bons résultats quand on l'utilise en dehors des hypothèses sous lesquelles elle a été établie" [1]. Naturellement, en l'absence de convexité, il converge vers le minimum local dans le "bassin d'attraction" duquel on est parti. Il convient donc éventuellement de refaire fonctionner l'algorithme avec divers conditions initiales.

Par contre, on ne doit pas attendre une bonne convergence *près de l'optimum*. C'est à dire que l'algorithme est efficace pour faire décroitre rapidement la fonction, mais pas pour avoir une bonne approximation de x^*. Au point que Claude Lemaréchal a pu écrire dans un autre cours "on devrait **interdire** cette méthode". C'est un peu... totalitaire !. Mais pour approximer finement x^*, il vaut mieux finir avec une méthode du second ordre comme celle de l'algorithme que nous présenterons ensuite.

1. Holt Ashley

Préconditionnement

On remarque que la preuve de convergence demeure si la direction de descente choisie "fait un angle aigu" avec $-\nabla u$, comme le lemme le montre. Cette remarque a de nombreuses applications, qui tournent autour de ce qu'on appelle le "préconditionnement".

Supposons qu'on fasse un changement de variable $x = A\xi$, avec une matrice de changement de variable A non singulière. On est conduit à considérer la fonction $v(\xi) = u(A\xi)$. On peut montrer (exercice) que c'est une "bonne fonction", et qu'on peut donc lui appliquer l'algorithme du gradient. Cela donne-t-il la même suite de points que l'algorithme initial ? La réponse est non comme nous allons le voir.

Nous avons $v'(\xi) = u'(A\xi)A$, soit en transposant $\nabla v(\xi) = A^t \nabla u(A\xi)$ et donc, $v(\xi - \theta \nabla v) = u(A\xi - \theta A A^t \nabla u)$. On a donc comme direction de descente $-AA^t g$. Si A est régulière, AA^t est positive définie, et il existe donc $\gamma > 0$ tel que $(g, AA^t g) \geq \gamma \|g\|^2$. ($\gamma$ est le carré de la plus petite *valeur singulière* de A.) Donc, de la façon dont a été faite la preuve ci-dessus, on déduit immédiatement que l'algorithme converge encore. Mieux ?, moins bien ? Cela dépend évidemment du choix de A, ou de la matrice symétrique AA^t, qui est appelée matrice de *préconditionnement*.

Les algorithmes de gradient conjugué, par exemple, peuvent être vus come des algorithmes de gradient à préconditionnement soigné, et la méthode de Newton protégée ci-dessous comme un préconditionnement "optimal".

Plus simplement, on recommande en général d'utiliser au moins un préconditionnement diagonal de la forme $\xi_i = x_i/\bar{x}_i$ où les $\bar{x}_i$ jouent le rôle de facteur d'échelle, rendant en quelque sorte les ξ_i sans dimension, et doivent être choisis de façon que la sensibilité de $v(\xi)$ aux différents ξ_i soit à peu près la même. Ce qui est obtenu en prenant des $\bar{x}_i$ inversement proportionnels aux (ordre de grandeur des) $\nabla_i u$.

3.2.3 Méthode de Newton

La méthode "pure"

Nous présentons maintenant une méthode "du second ordre", en ce qu'elle utilise les dérivées secondes de u, mais aussi en ce qu'elle converge (plus vite que) quadratiquement. C'est dans cette famille de méthodes qu'il faut chercher si on veut approximer finement l'argument du minimum x^*.

Le principe est simple, il consiste à résoudre l'équation $u'(x) = 0$ par la méthode de Newton. Nous la rappelons d'abord pour la solution d'une équation $g(x) = 0$ où $g(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$.

La méthode de Newton consiste à approximer g au premier ordre autour du dernier point connu, et à prendre comme prochaine estimée de la solution x^* de $g(x) = 0$, le point où cette approximation linéaire s'annule. Ceci mène à

$$g(x) \simeq g(x^k) + g'(x^k)(x - x^k)\,,$$

soit

$$x^{k+1} = x^k - [g'(x^k)]^{-1} g(x^k) \tag{3.4}$$

Cette méthode converge quadratiquement comme le montre le résultat suivant :

Théorème 3.4 (Convergence de la méthode de Newton) *Si la fonction g est deux fois continument différentiable au voisinage de x^*, avec une dérivée première inversible en x^*, il existe un voisinage de x^* dans lequel la méthode de Newton converge quadratiquement.*

Démonstration Par continuité de g', il existe un voisinage de $\mathcal{V}$ de x^* dans lequel la matrice $g'(x)$ est inversible, et plus précisément a une plus petite valeur singulière bornée inférieurement par un nombre positif α, de sorte que $[g'(x)]^{-1}$ existe et a une norme bornée supérieurement par $1/\alpha$. De même, dans ce voisinage, $g''(x)$ existe et a une norme bornée par un nombre positif γ.

Écrivons l'itération de Newton comme

$$x^{k+1} - x^* = [g'(x^k)]^{-1}[g'(x^k)(x^k - x^*) - g(x^k)]$$

Par le développement limité (1.10), et en se souvenant que par définition $g(x^*) = 0$, le dernier crochet ci-dessus peut se ré-écrire comme ci-dessous, coordonnée par coordonnée :

$$g_i'(x^k)(x^k - x^*) - g_i(x^k) = \frac{1}{2}\left((x^k - x^*),\, D^2 g_i(x^k + \theta(x^* - x^k))(x^k - x^*)\right) ,$$

de sorte que ce crochet est borné en norme par

$$\|g'(x^k)(x^k - x^*) - g(x^k)\| \leq \frac{\gamma}{2}\|x^k - x^*\|^2 .$$

Donc, en utilisant la borne sur $\|(g')^{-1}\|$,

$$\|x^{k+1} - x^*\| \leq \frac{\gamma}{2\alpha}\|x^k - x^*\|^2 .$$

Il reste à s'assurer que si on part suffisamment prés de x^*, la suite engendrée ne sort pas de $\mathcal{V}$, ce qui se fait en bornant la somme des $\|x^k - x^*\|$ pour $\|x^0 - x^*\|$ suffisamment petit. Ainsi, le théorème est démontré.

On voit que si $g(x) = \nabla u(x)$, nous avons donné un algorithme de recherche de minimum dans un ouvert, à savoir

$$x^{k+1} = x^k - \left[D^2 u(x^k)\right]^{-1} \nabla u(x^k) , \tag{3.5}$$

et montré sa convergence quadratique si u est trois fois continument différentiable. La grande faiblesse de cet algorithme *très rapide* est sa *très grande sensibilité aux conditions intiales*. C'est pourquoi on conseille souvent de commencer la recherche du minimum avec une méthode plus robuste comme celle du gradient, et de finir avec la méthode de Newton.

Remarque 3.2 *Une remarque importante est que dans cette version "pure", on n'a pas besoin d'inverser $D^2u(x^k)$ en dépit de ce que semble indiquer la formule (3.5). En effet, il suffit de résoudre le système linéaire*

$$D^2 u(x^k)(x^{k+1} - x^k) = -\nabla u(x^k) , \tag{3.6}$$

ce qui peut être significativement moins long. Cela n'évite pas le calcul de la matrice des dérivées secondes $D^2u(x^k)$ à chaque pas.

La méthode de Newton "protégée"

Une méthode recommandée consiste à considérer $h = -[g'(x^k)]^{-1}g(x^k)$ comme une *direction de descente*, et, comme précédemment effectuer une recherche unidimensionnelle de minimum dans cette direction, sachant que l'optimum devrait se trouver près de 1. Le caractère positif défini de $g'(x) = D^2u(x)$, si u est α-convexe, garantit la convergence de cet algorithme en vertu de la preuve de convergence de l'algorithme de gradient. En pratique, la direction de recherche est si bonne qu'il suffit de faire du "backtracking" (cf. le paragraphe 2.3.1) depuis le "pas de Newton" 1.

Il n'est pas très sûr que le poids du calcul de $[D^2u(x^k)]^{-1}$ en vaille la peine si on est vraiment trop loin de l'optimum. Divers aménagement de la méthode de Newton visent à alléger cette étape. Avant de les évoquer, notons que, toujours si la fonction u est convexe, la matrice $\nabla g = D^2u$ à inverser est positive définie, de sorte qu'on dispose pour cette inversion de la très efficace méthode de Cholesky.

Les méthodes de Newton modifiées, quasi-Newton

Le coût principal en calcul dans la méthode de Newton est dans l'inversion de $D^2u(x^k)$ à chaque pas. Remarquons que la preuve de convergence demeure inchangée si on remplace $D^2u(x^k)$ par $D^2u(x^*)$, qui lui, est constant, et ne demanderait donc qu'une inversion unique. Bien sûr, on ne connait pas x^*, et donc pas non plus $D^2u(x^*)$ (sauf si $D^2u(x)$ est constante, mais alors u est une simple fonction quadratique, et l'algorithme de Newton converge en un pas !) Par contre, la preuve demeure encore si on remplace $D^2u(x^k)$ par une matrice H_k telle que

$$\|H_k - D^2u(x^*)\| \leq \eta\|x^k - x^*\|\,.$$

On aura en effet,

$$x^{k+1} - x^* = H_k^{-1}[H_k(x^k - x^*) - g(x^k)]\,,$$

et le crochet s'écrit maintenant

$$H_k(x^k - x^*) - g(x^k) = D^2u(x^*)(x^k - x^*) - g(x^k) + (H_k - D^2u(x^*))(x^k - x^*)$$

dont le module est encore majoré par un terme quadratique en $\|x^k - x^*\|$.

On va donc chercher à avoir, à moindre prix, une suite H_k tendant vers $D^2u(x^*)$ avec x^k. Une idée simplissime, mais assez efficace si elle est utilisée avec doigté, consiste à ne remettre à jour $D^2u(x)$, et donc $D^2u(x)^{-1}$, que de temps en temps. Typiquement tout les deux à cinq pas.

Des méthodes plus sophistiquées existent, sous le nom de méthodes de "quasi Newton", qui s'apparentent au gradient conjugué, et cherchent une formule itérative pour approximer H_k^{-1}. Nous n'en parlerons pas plus ici.

3.3 Optimisation sous contraintes inégalité

3.3.1 Position du problème

La plupart des problèmes d'optimisation, notamment en théorie de la décision, (contrôle, économie théorique, recherche opérationnelle, etc.) se présentent avec des contraintes sur les variables de décision x. D'une manière abstraite, ces contraintes se traduisent par l'existence d'un ensemble $C \subset \mathbb{R}^n$ de *variables admissibles*. On cherche donc $x^* \in C$ tel que

$$\forall x \in C \quad u(x) \geq u(x^*)\,.$$

L'algorithme du gradient projeté ci-dessous considère le problème sous cette forme, et utilisera la projection sur C en le supposant convexe fermé. Ceci suppose que cette projection soit facile à faire, ce qui est rarement le cas.

En pratique, le cas le plus courant est celui où l'ensemble des variables admissibles est défini indirectement par des contraintes de la forme

$$C = \{x \in \mathbb{R}^n \mid f_i(x) \leq 0\,, \quad i = 1, 2, \ldots, p\}\,,$$

qu'on écrira aussi $f(x) \leq 0$. Les méthodes intéressantes sont alors celles qui sont explicites en fonction de f.

La théorie de la dualité (multiplicateurs de Lagrange, de Kuhn et Tucker, etc) est à la base de plusieurs algorithmes visant à répondre à ce type de problème. Nous ne présenterons, dans cette famille, que l'algorithme d'Uzawa, réputé le plus robuste.

Enfin, par souci de présenter les méthodes les plus employées, nous donnerons des méthodes de pénalisation, qui échangent une plus grande simplicité théorique contre une efficacité douteuse.

3.3.2 Gradient projeté

Projection sur un convexe simple

On a rappelé dans le chapitre premier l'existence de la projection $P_C(x)$ d'un vecteur x sur un convexe fermé C. Cette projection est une opération simple dans un petit nombre de cas. Typiquement trois cas :

- C est un pavé, de la forme $a_i \leq x_i \leq b_i$, où les a_i et b_i sont donnés. Alors la projection se fait coordonnée par coordonnée et consiste simplement à "ramener x_i dans $[a_i,\, b_i]$". On peut écrire cela comme
$$(P_C(x))_i = \max\{a_i,\, \min\{x_i,\, b_i\}\}\,.$$
- C est une boule, de la forme $\|x - \xi\| \leq \rho$ où le vecteur ξ de $\mathbb{R}^n$ et le réel positif ρ sont donnés. Alors la projection consiste à ramener x dans la boule le long du rayon, soit
$$P_C(x) = \xi + \min\{1, \frac{\rho}{\|x - \xi\|}\}(x - \xi)\,.$$
- C est un demi-espace, de la forme $(p, x) \leq a$, où le vecteur p de $\mathbb{R}^n$ et le réel a sont donnés. La projection consiste à ramener x dans le demi espace parallèlement à p :
$$P_C(x) = x - \max\left\{0, \frac{(p,x) - a}{(p,p)}\right\} p\,.$$

Exercice 3.1 *Vérifier les formules ci-dessus.*

L'algorithme

L'algorithme du gradient projeté décrit ci-dessous n'existe que dans cette version à pas fixe. A notre connaissance, il n'y a pas de version "à pas optimal".

Naturellement, si le convexe des contraintes sur lequel on projette est tout $\mathbb{R}^n$, la projection est l'identité, et donc la preuve ci-dessous montre la convergence de l'algorithme de gradient à pas fixe sans projection, pour le problème sans contraintes. (Pourvu, toujours, qu'on ait choisi le pas convenablement.) En fait, dans le cas sans contrainte, l'algorithme à pas fixe n'est *vraiment pas* à recommander.

L'algorithme de gradient projeté est fondé sur la remarque suivante. L'inégalité d'Euler pour l'optimisation dans un convexe (1.18) nous dit que si x^* fournit le minimum de u sur C, alors
$$\forall t > 0 \quad P_C(x^* - t\nabla u(x^*)) = x^*\,. \tag{3.7}$$
En effet, soit x^* est intérieur à C, et alors $\nabla u(x^*) = 0$, ce qu'implique (3.7) car pour un point intérieur, et si $\nabla u(x^*) \neq 0$, il existe t suffisamment petit pour que $x^* - t\nabla u(x^*) \in C$, de sorte qu'il

serait sa propre projection. Soit x^* est un point frontière, et (3.7) est équivalent à l'affirmation que $-\nabla u(x^*)$ est une normale extérieure à C, ce que dit (1.18).

L'algorithme s'écrit comme une méthode de recherche du point fixe dans (3.7) :

Algorithme Gradient projeté

1. Choisir une éstimée initiale x^0, un pas $t > 0$, faire $k := 0$,
2. calculer $x^{k+1} = P_C(x^k - \theta\nabla u(x^k))$,
3. si $\|x_{k+1} - x_k\| \leq \varepsilon$ stop, si non, incrémenter k de 1 et retourner en (2)

On a le résultat de convergence suivant :

Théorème 3.5 (Convergence de l'algorithme du gradient projeté) *Si $u(\cdot)$ est une bonne fonction, et si $0 < \theta < 2/\beta$, l'algorithme de gradient projeté converge vers le minimum x^* de u sur C.*

Remarque 3.3 *Le test d'arrêt ne peut pas porter sur le module du gradient de u, dont on ne sait pas a priori s'il sera grand ou petit. Le test proposé ici vérifie donc qu'il soit "bien orthogonal" au bord de C si x^k est sur ce bord, et petit si-non. En fait, pour l'utilisation avec une fonction u dont la convexité est douteuse, il vaut mieux faire porter ce test sur la différence $\|x^{k+1} - x^{k-4}\|$ par exemple, c'est à dire prendre en considération l'évolution de l'algorithme sur* plusieurs *pas.*

Démonstration Puisque nous avons reconnu dans l'algorithme une itération de point fixe (aussi dite de Picard), montrons que la fonction $x \mapsto P_C(x - \theta\nabla u(x))$ est une contraction pour θ choisi comme dans le théorème. On sait (théorème 1.38) que la projection sur C est une contraction au sens large. Il suffit donc de montrer que $\varphi_\theta(x) := x - \theta\nabla u(x)$ est une contraction stricte. On a $\nabla\varphi_\theta(x) = I - \theta D^2u(x)$. Cette matrice est symétrique. Sa norme est donc son rayon spectral, le module de sa valeur propre de plus grand module. Or ses valeurs propres sont de la forme $1 - \theta\lambda(D^2u(x))$, où les $\lambda(D^2u(x))$ sont les valeurs propres de $D^2u(x)$. Par hypothèses, celles-ci sont comprises entre α et β. Il suffit donc de choisir θ de façon que $|1 - \theta\alpha| < 1$ et $|1 - \theta\beta| < 1$, soit $\theta > 0$ et $\theta < 2/\beta$ respectivement. (Cette dernière condition assurant aussi $\theta < 1/\alpha$.) Ce qui établit la convergence de l'algorithme sous la condition annoncée.

On peut se demander quel est le θ "optimal", ou au moins celui pour lequel le module de Lipshitz de φ_θ est minimal. On voit assez facilement qu'il est tel que $1 - \theta\alpha = \theta\beta - 1$, soit $\theta = 2/(\alpha + \beta)$. Et alors, le module de Lipshitz de φ_θ est $1 - 2\alpha/(\alpha + \beta)$.

On voit que ce nombre est d'autant plus proche de 1 que α est petit. C'est une constante des problèmes d'optimisation que trouver le minimum est d'autant plus difficile que le module d'alpha convexité est petit.

En fait, ce théorème souligne surtout la principale difficulté liée à l'utilisation de cet algorithme. C'est celle du choix du pas θ. En effet, en général, α et β ne sont pas connus —bien heureux s'ils existent—, et il faut donc des heuristiques d'adaptation du pas θ.

Remarquons d'abord qu'à en croire le calcul précédent, un pas trop petit peut ralentir considérablement la convergence, pas l'empêcher comme peut le faire un pas trop grand. Il faut quand même trouver un moyen d'apprécier le pas qu'on peut se permettre. Une règle qu'on peut proposer est la suivante. Comparer la décroissance de u obtenue, $u(x^k) - u(x^{k+1})$, à son estimation au premier ordre $u'(x^k)(x^k - x^{k+1}) = \theta\|\nabla u(x^k)\|^2$ (qui doit être plus grande). Si ces deux quantités coincident "très bien", disons à 10% près, on peut sans doute doubler le pas. Si elles coincident mal, disons plus mal que dans un rapport deux, il faut sans doute diviser le pas par deux.

Cet algorithme n'est sans doute pas très performant par rapport à ceux que nous avons vus jusqu'ici. Il faut bien comprendre que la minimisation sous contrainte est un problème plus difficile que la minimisation sans contrainte, et qu'on ne peut espérer trouver des algorithmes aussi efficaces.

3.3.3 Algorithme d'Uzawa

L'algorithme d'Uzawa exploite le point selle du théorème de Kuhn et Tucker. Il va donc chercher à minimiser le lagrangien par rapport à x et à le maximiser par rapport à la variable duale, disons p. Plus précisément, il consiste à remettre à jour l'estimée courante de la variable duale par un pas de gradient —et le gradient du lagrangien par rapport à p est particulièrement simple—, puis à p fixé, à aller jusqu'au minimum en x. Ce sera donc un "méta algorithme", puisque nous ne dirons pas comment effectuer la minimisation en x. Observons pourtant, —et c'est tout ce que la dualité fait pour nous—, que dans cette minimisation, les contraintes $f(x) \leq 0$ ont disparu.

Comme dans le théorème 1.41, nous permettons ici à certaines contraintes de rester "abstraites" sous la forme $x \in C$ tandis que d'autres sont rendues "concretes" sous la forme $f(x) \leq 0$. La minimisation du lagrangien est alors à effectuer *dans* C. Si C est tout $\mathbb{R}^n$, on a alors affaire à un problème de minimisation *sans contrainte*. Donc un algorithme de gradient à pas optimal, par exemple, est possible. La dualité a ainsi ramené un problème de minimisation sous contraintes à une suite de problèmes de minimisation sans contrainte.

On note P_+ la projection sur le cône positif de $\mathbb{R}^n$, qui consiste juste à ramener à zéro toute composante négative du vecteur à projeter.

Algorithme Uzawa

1. Choisir une estimée initiale $p^0 \geq 0$ Faire $k := 0$.
2. Calculer x^k par
$$u(x^k) + (p^k, f(x^k)) = \min_{x \in C} [u(x) + (p^k, f(x))] .$$
3. faire
$$p^{k+1} = P_+[p^k + \rho f(x^k)] .$$
Si $\|p^{k+1} - p^k\| \leq \varepsilon$, stop. Sinon, retourner en 2

Cet algorithme est en fait un algorithme de gradient (en p) projeté (sur le cône positif). On démontre sa convergence d'une façon analogue à la preuve de convergence de cet algorithme. (Cf théorème 3.5)

Théorème 3.6 (Convergence de l'algorithme d'Uzawa) *Si $u(\cdot)$ est une bonne fonction, et $f(\cdot)$ est Lipshitz-continue de coefficient γ, pour $\rho < 2\alpha/\gamma^2$, la suite $\{x^k\}$ engendrée par l'algorithme d'Uzawa converge vers l'optimum x^*.*

Démonstration Remarquons d'abord que la condition des écarts complémentaires entraine que pour tout $\rho > 0$,
$$p^* = P_+[p^* + \rho f(x^*)] .$$
Ainsi, par le théorème 1.38, on a
$$\|p^{k+1} - p^*\| \leq \|p^k - p^* + \rho(f(x^k) - f(x^*))\| .$$
En élevant au carré, il vient
$$\|p^{k+1} - p^*\|^2 \leq \|p^k - p^*\|^2 + 2\rho(p^k - p^*, f(x^k) - f(x^*)) + \rho^2 \|f(x^k) - f(x^*)\|^2 \qquad (3.8)$$

Nous allons majorer le double produit (le terme central du deuxième membre). Remarquons que, grâce au fait que les p_i sont positifs, $u + (p, f)$ est encore convexe, et même α-convexe, en x. Par (1.18), la minoration (1.17) reste valide pour la minimisation dans un convexe fermé. On a donc

$$\begin{aligned} u(x^k) + (p^k, f(x^k)) &\leq u(x^*) + (p^k, f(x^*)) - \frac{\alpha}{2}\|x^k - x^*\|^2 , \\ u(x^*) + (p^*, f(x^*)) &\leq u(x^k) + (p^*, f(x^k)) - \frac{\alpha}{2}\|x^k - x^*\|^2 . \end{aligned}$$

Sommons membre à membre et faisons passer ce qu'il faut à gauche, pour obtenir

$$(p^k - p^*, f(x^k) - f(x^*)) \leq -\alpha\|x^k - x^*\|^2 .$$

Ensuite, l'hypothèse que f est Lipshitz de coefficient γ donne

$$\|f(x^k) - f(x^*)\| \leq \gamma\|x^k - x^*\| .$$

En reportant ces deux majorations dans (3.8), il vient

$$\|p^{k+1} - p^*\|^2 \leq \|p^k - p^*\|^2 + (\rho^2\gamma^2 - 2\rho\alpha)\|x^k - x^*\|^2 .$$

Si on a choisi $\rho < 2\alpha/\gamma^2$, il existe $\delta > 0$ tel que $\rho^2\gamma^2 - 2\rho\alpha < -\delta$, d'où

$$\|p^{k+1} - p^*\|^2 \leq \|p^k - p^*\|^2 - \delta\|x^k - x^*\|^2$$

qu'on peut ré-écrire

$$\delta\|x^k - x^*\|^2 \leq \|p^k - p^*\|^2 - \|p^{k+1} - p^*\|^2 .$$

Donc, la suite $\|p^k - p^*\|^2$ est décroissante. Comme elle est composée d'éléments positifs, elle converge, donc la différence du deuxième membre ci-dessus tend vers zéro. Donc $\|x^k - x^*\|$ tends vers zéro, ce qu'il fallait démontrer.

Terminons en évoquant l'interprétation "économique" du théorème de Kuhn et Tucker, et donc de l'algorithme d'Uzawa. Si les containtes $f_i(x) \leq 0$ représentent des ressources "rares" dont on ne peut utiliser que la quantité disponible, et les p_i^* associés en sont les prix unitaires à l'équilibre, on voit que le lagrangien est un coût économique total prenant en compte le "prix" des denrées rares utilisées, et que l'algorithme consiste tout simplement à diminuer le prix des ressources qu'on n'utilise pas jusqu'à saturation, et à augmenter le prix de celles pour lesquelles la demande dépasse la disponibilité. Rien que de très naturel.

Dualisation partielle Une remarque importante doit être faite à ce stade. On a énoncé le théorème de Kuhn et Tucker pour un problème de la forme

$$\forall x \in K \quad u(x) \geq u(x^*) .$$

avec

$$K = C \cap \{x \in \mathbb{R}^n \mid f(x) \leq 0\} ,$$

et on a "dualisé" —c'est à dire introduit dans le lagrangien— les seules contraintes "concretes" $f(x) \leq 0$. Les autres sont restées "abstraites" et prises en compte par la condition $x \in C$ dans les inéquations du point selle (1.19) et l'algorithme d'Uzawa.

Si C n'est en effet pas tout $\mathbb{R}^n$, i.e. une partie des contraintes est restée non dualisée, l'étape de calcul de x^k dans l'algorithme est un problème de minimisation sous ces contraintes. Cette minimisation peut être faite à l'aide d'un autre algorithme qu'Uzawa et la dualité, menant à une "hybridation

d'algorithmes". Si C est un convexe simple, cette minimisation peut être effectuée par un algorithme de gradient projeté par exemple.

Bien sûr, le choix des contraintes à exprimer d'une façon ou de l'autre (à dualiser ou à ne pas dualiser) est laissé à l'utilisateur, et il n'est efficace d'utiliser une dualisation partielle qu'en ne laissant non dualisées que des contraintes simples. Typiquement, si on a un nombre important de contraintes de borne $a_i \leq x_i \leq b_i$, qui impliqueraient deux fois plus de multiplicateurs. On peut alors préférer renoncer à les dualiser et les prendre en compte directement dans la minimsation du lagrangien par projection.

3.3.4 Pénalisation

Les méthodes présentées ici essayent de ramener le problème contraint à un problème non contraint de façon plus "naïve", mais qui peut marcher. En particulier, on les utilise de façon non itérative, contrairement à Uzawa. On ne résoudra donc qu'un, ou quelques, problème(s) de minimisation sans contrainte. L'idée est la suivante : on va "faire payer" au critère le fait pour x de sortir de C. Et si ce "prix" est très élevé, l'optimum se trouvera dans C.

Pénalisation extérieure quadratique

Supposons donc que l'ensemble des x admissibles est donné par

$$f_i(x) \leq 0, \quad i = 1, \ldots, p$$

que nous écrivons aussi $f(x) \leq 0$, où $f : \mathbb{R}^n \to \mathbb{R}^p$. Introduisons en outre la fonction $f^+(x)$ appelée *partie positive* de f, définie par

$$f_i(x) = \max\{0\,,\, f_i(x)\}\,, \quad i = 1, \ldots, p\,.$$

Notons la proposition :

Proposition 3.7 *Si f est dérivable, $\|f^+\|^2$ l'est aussi, et on a*

$$\frac{d\|f^+(x)\|^2}{dx} = 2(f^+(x))^t f'(x)$$

Démonstration Notons d'abord que la proposition n'est pas (entièrement) évidente, car f^+, elle, n'est en général pas dérivable aux points où $f(x) = 0$, donc notamment en x^*. Prenons le cas d'une fonction f *scalaire*. Il suffira ensuite d'appliquer le résultat à chaque composante de f.

Remarquons d'abord que si $f(x) > 0$ ou $f(x) < 0$, par continuité l'inégalité reste vraie dans un voisinage de x. Dans le premier cas $f^+ = f$ et dans le deuxième $f^+ = 0$ dans ce voisinage. La formule $[(f^+)^2]' = 2(f^+)f'$ est alors évidemment vérifiée.

Soit donc x un point où $f(x) = 0$ (c'est à dire un point frontière de C). Soit e_i un vecteur de base de $\mathbb{R}^n$. On a donc dans le "quotient différentiel"

$$\frac{1}{t}[(f^+(x + te_i))^2 - (f^+(x))^2] = \frac{1}{t}(f^+(x + te_i))^2$$

et

$$0 \leq \left|\frac{1}{t}(f^+(x + te_i))^2\right| \leq \left|\frac{1}{t}(f(x + te_i))^2\right|\,,$$

la dernière inégalité parce que dans tous les cas, $|f^+(x)| \leq |f(x)|$. Dans les inégalités ci-dessus, f étant dérivable, le terme de droite tends vers $2|f(x)f'(x)| = 0$, et donc le terme central a une limite, qui est zéro. On a donc démontré que $(f^+)^2$ a en x une dérivée partielle en x_i, qui est nulle. Ceci achève de démontrer la proposition.

Nous considérerons le *critère augmenté*

$$u_\varepsilon(x) = u(x) + \frac{1}{\varepsilon}\|f^+(x)\|^2 .$$

La méthode de pénalisation consiste à utiliser la solution x_ε du problème *sans contrainte* :

$$u_\varepsilon(x_\varepsilon) = \min_{x\in\mathbb{R}^n} u_\varepsilon(x)$$

comme approximée de la solution x^* du problème contraint. Cette méthode est justifiée par le résultat suivant :

Théorème 3.8 *Si $u(\cdot)$ est une bonne fonction, et $f(\cdot)$ est continue, $x_\varepsilon \to x^*$ quand $\varepsilon \to 0$.*

Démonstration On a manifestement

$$u_\varepsilon(x_\varepsilon) \leq u_\varepsilon(x^*) = u(x^*) ,$$

la dernière égalité parce que par définition, $f^+(x^*) = 0$. Si u est α-convexe, elle est bornée inférieurement. Donc l'inégalité

$$u(x_\varepsilon) + \frac{1}{\varepsilon} f^+(x_\varepsilon)^2 \leq u(x^*)$$

implique que $f^+(x_\varepsilon) \to 0$ quand $\varepsilon \to 0$. A fortiori, elle implique aussi que $u(x_\varepsilon)$ reste bornée, donc, toujours avec l'α-convexité, que x_ε reste borné. Ainsi, pour toute suite décroissante de ε_k tendant vers zéro, les x_{ε_k} ont des points d'accumulation. Soit $\bar{x}$ un tel point et ε' une suite telle que les $x_{\varepsilon'} \to \bar{x}$. Par continuité de f^+, $f^+(\bar{x}) = 0$, et $\bar{x}$ est donc admissible.

On a aussi

$$u(x_\varepsilon) \leq u_\varepsilon(x_\varepsilon) \leq u(x^*) .$$

Donc par passage à la limite, (u est continue), $u(\bar{x}) \leq u(x^*)$. Comme $\bar{x}$ est admissible, il en découle que $u(\bar{x}) = u(x^*)$ et $\bar{x}$ est optimal. Par l'α-convexité de u, l'optimum est unique. Donc c'est toute la suite qui converge vers x^*.

On démontre aussi que, si la matrice $f'(x^*)$ est injective, ce qui est une hypothèse naturelle, le produit $(1/\varepsilon)f^+(x_\varepsilon)$ tend vers un vecteur λ de $\mathbb{R}^p$, de sorte qu'à l'optimum on a $u'(x^*) + \lambda^t f'(x^*) = 0$. Ce λ est le *multiplicateur de Lagrange* (ou de Kuhn et Tucker) associé au problème d'optimisation sous contraintes.

Cette méthode reste délicate d'emploi pour plusieurs raisons. D'abord, on ne va pas résoudre beaucoup de fois le problème pénalisé : c'est un travail potentiellement important. Donc quel ε choisir est non trivial. De plus, si u est trop "plat" au voisinage de C, son rôle dans u_ε va être masqué par le terme de pénalisation, nuisant à la précision de l'estimation de x^*. (Certes, l'hypothèse d'α-convexité limite ce risque en bornant inférieurement la courbure de u. Mais ceci souligne la dépendance de la méthode à cette hypothèse qui est d'habitude difficile à tester.)

Autres pénalisations On a pénalisé le critère avec la fonction de pénalisation $\|f^+\|^2$. Ceci pour avoir un critère u_ε dérivable. Le prix à payer est que, génériquement, si le minimum recherché est sur la frontière de C, on l'approchera par des points extérieurs. Les x_ε sont *non admissibles*.

On aurait pu prendre $f^+(x)$ tout simplement. Alors le critère n'est plus dérivable, mais il est possible qu'un ε fini permette déjà d'obtenir le minimum exact. Le caractère non différentiable du critère en x^* est néanmoins une grave difficulté. En fait, il vaut mieux se référer alors à la théorie de la dualité.

Pénalisation intérieure

Une autre approche possible est d'utiliser comme fonction de pénalisation une fonction "barrière", $\varphi(x)$, qui tend vers l'infini quand x tend vers la frontière de C par l'intérieur. On pourrait penser à $\varphi(x) = -\sum_i 1/\varphi_i(x)$ par exemple, mais nous verrons à la section suivante qu'un meilleur choix est $\varphi(x) = -\sum_i \ln(-f_i(x))$. Puis on va considérer $u_\varepsilon(x) = u(x) + \varepsilon\varphi(x)$, avec ε assez petit pour que ceci ne change guère le comportement du critère tant que les f_i sont tous loins d'être nuls. On parle alors d'algorithmes de "points intérieurs", parce qu'on aborde l'optimum recherché par des points intérieurs à C.

Soit donc d'une manière un peu plus générale, un critère perturbé $u_\varepsilon(x) = u(x) + \varepsilon\varphi(x)$ ou $\varphi(\cdot)$ est définie pour les x *intérieurs* au domaine des contraintes C (c'est à dire les x tels que $f_i(x) < 0$ pour $i = 1, \dots p$), convexe (continue) positive dans $\overset{\circ}{C}$, et tend vers l'infini quand $x \to \partial C$.

Alors, pour tout ε positif, u_ε atteint son minimum dans $\overset{\circ}{C}$, en un unique point x_ε dès lors que u est strictement convexe.

On a alors le résultat attendu :

Théorème 3.9 *Sous les hypothèses ci-dessus, et si u est une bonne fonction, $x_\varepsilon \to x^*$ quand $\varepsilon \to 0$.*

Demonstration Il faut faire attention qu'on ne peut pas manipuler $u_\varepsilon(x^*)$ parce que x^* peut être (est généralement) sur la frontière de C et donc φ peut n'y être pas définie. Soit donc δ un nombre positif (arbitrairement petit) et x_δ un point *itérieur* à C tel que $u(x_\delta) \leq u(x^*) + \delta$. La suite d'inégalités ci-dessous est facile à établir :

$$u(x_\varepsilon) \leq u_\varepsilon(x_\varepsilon) \leq u_\varepsilon(x_\delta) = u(x_\delta) + \varepsilon\varphi(x_\delta) \leq u(x^*) + \delta + \varepsilon\varphi(x_\delta)\,.$$

En prenant $\varepsilon \leq \delta/\varphi(x_\delta)$ on en déduit $u(x_\varepsilon) \leq u(x^*) + 2\delta$. Et comme δ était arbitraire, on en déduit que $u(x_\varepsilon)$ tend vers $u(x^*)$ quand ε tend vers zéro. Si u est α-convexe, on en déduit par utilisation de (1.17) que x_ε tend vers x^*.

Cette idée est à la base d'une méthode qui est aujourd'hui la meilleure connue si le problème est réellement convexe et si les dérivées secondes sont faciles à calculer. Nous la présentons ci-dessous.

3.3.5 Méthode du chemin central

Avant d'exposer cette utilisation de la pénalisation intérieure, montrons un résultat qui constitue la partie facile de la théorie de la dualité. Le problème considéré est toujours le même. Posons comme précédemment $\mathcal{L}(x,p) = u(x) + \sum_i p_i f_i(x)$.

Lemme 3.10 *Soit x^* la solution du problème d'optimisation sous contraintes inégalité. Soit $p \in \mathbb{R}^m$. On a*

$$\forall p \geq 0\,, \quad \min_x \mathcal{L}(x,p) \leq u(x^*)\,. \tag{3.9}$$

Démonstration On a la suite suivante d'inégalités faciles (la seconde vient de ce que pour $x \in C$, $\sum_i p_i f_i(x) \leq 0$) :

$$\min_x \mathcal{L}(x,p) \leq \min_{x\in C} \mathcal{L}(x,p) \leq \min_{x\in C} u(x) = u(x^*)\,.$$

Nous utilisons la méthode de pénalisation intérieure avec les fonctions barrières $-\ln(-f_i(x))$. Posons donc

$$u_\varepsilon = u - \varepsilon \sum_i \ln(-f_i(x))\,.$$

C'est une fonction fortement convexe. Notons x_ε son unique minimum sur $\mathbb{R}^n$. Nous affirmons le théorème facile, mais surprenant :

Théorème 3.11 *On a*

$$u(x_\varepsilon) - p\varepsilon \leq u(x^*) \leq u(x_\varepsilon)\,.$$

(Où p est ici le nombre de contraintes scalaires : $i = 1, \ldots, p$.)

Demonstration Le point x_ε est caractérisé par

$$\nabla u_\varepsilon(x_\varepsilon) = \nabla u(x_\varepsilon) - \varepsilon \sum_i \frac{1}{f_i(x_\varepsilon)} \nabla f_i(x_\varepsilon) = 0\,.$$

Posons alors $p_i = -\varepsilon / f_i(x_\varepsilon)$. Ce sont des nombres positifs. L'égalité ci-dessus s'écrit

$$\nabla u(x_\varepsilon) + \sum_i p_i \nabla f_i(x_\varepsilon) = \nabla_x \mathcal{L}(x,p) = 0\,.$$

Maintenant, ce $\mathcal{L}(x,p)$ est une fonction convexe de x. Donc la condition ci-dessus implique qu'elle atteint son minimum en x_ε. Utilisons alors le petit lemme précédent, en remarquant en outre que $p_i f_i(x_\varepsilon) = \varepsilon$. Le théorème en découle immédiatement.

Ainsi non seulement x_ε approche x^* quand $\varepsilon \to 0$, mais en outre on sait avec quelle précision $u(x^*)$ est atteint.

Voici quel est l'usage qu'on fait de ce résultat dans la méthode du chemin central. On commence typiquement avec $\varepsilon = 1$ (bien qu'une autre valeur soit bien sûr possible, et peut-être préférable dans certains cas.) On résoud le problème $\min u_\varepsilon$ sans contrainte par la méthode de Newton. La première application de cette méthode peut présenter ses difficultés habituelles et demander un peu de soin. Ensuite, on fait décroître ε, typiquement d'un ordre de grandeur à chaque fois, et on résoud à nouveau le problème sans contrainte en prenant la solution à l'étape précédente comme initialisation de la méthode de Newton. Cette fois on a une convergence très rapide de cette méthode. Et on continue à faire décroître ε jusqu'à ce qu'on ait la précision désirée. (On peut même améliorer le choix du x initial dans la méthode de Newton en extrapolant la courbe des x_ε antérieurs.)

On sait que le problème d'optimisation sans contrainte résolu à chaque itération par la méthode de Newton est de plus en plus mal conditionné au fur et à mesure que ε décroît. Mais pour une raison pas totalement expliquée, il semble que le choix d'initialisation de la méthode de Newton suggéré naturellement immunise l'algorithme contre les effets négatifs de ce mauvais conditionnement.

Cette méthode, due à Nesterov et Nemirovsky sur une idée de Karmarkar, puis améliorée par S. Boyd, a donné, entre les mains de ce dernier, des résultats spectaculaires sur de très nombreux problèmes. Elle est *la* méthode à appliquer . . .quand elle s'applique. (La convexité de u et des f_i est essentielle, et il faut que le calcul des dérivées secondes soit raisonnablement simple.)

3.4 Optimisation sous contraintes égalité

On considère à présent le cas où les contraintes sont de la forme

$$\mathcal{C} = \{x \in \mathbb{R}^n \mid f_i(x) = 0\,, \quad i = 1, \ldots, p\}\,. \tag{3.10}$$

Il n'y a plus lieu de supposer une quelconque convexité pour les f_i ni u parce qu'on sort de toutes façons du cadre convexe. (C'est pour rappeler cela que nous n'avons pas utilisé la notation C —mais $\mathcal{C}$— pour l'ensemble des x admissibles.)

A quelques indications près, on laissera le soin au lecteur d'imaginer comment hybrider les algorithmes que nous allons discuter avec ceux du cas inégalité si une partie des contraintes est d'un type et une partie d'un autre.

3.4.1 Contraintes affines

On considère donc maintenant le cas où les f_i dans (3.10) sont affines, ou, de manière équivalente, il nous est donné une matrice F de type $p \times n$ et un vecteur f de dimension p, qui définissent les x admissibles comme devant vérifier

$$Fx = f \tag{3.11}$$

En outre, on supposera toujours que F est de rang p, donc surjective, ce qui impose en particulier que $p < n$. En effet, si ce n'est pas le cas, soit f est dans l'image de F, mais alors une partie des contraintes est redondante : on peut supprimer des lignes qui sont linéairement dépendantes des autres, soit f n'est pas dans l'image de F, et alors il n'y a pas de x admissible.

Algorithmes de gradient

La variété affine admissible est convexe. Donc l'algorithme de gradient projeté peut être conservé à l'identique. Il reste juste à faire remarquer que la projection est facile à calculer, au moins s'il n'y a pas trop de contraintes, et s'exprime par

$$P_{\mathcal{C}}(x) = (I - F^\dagger F)x + F^\dagger f \tag{3.12}$$

où

$$F^\dagger = F^t(FF^t)^{-1}$$

est une inverse à droite (l'inverse de Penrose) de la matrice surjective F.

En fait on peut faire mieux. En effet, la projection sur Ker F du gradient de u est alors le gradient de la restriction de u à $\mathcal{C}$. On peut donc appliquer un algorithme de gradient à pas optimal à cette restriction :

Algorithme Gradient projeté à pas optimal

1. Choisir une estimée initiale x^0, faire $k := 0$.
2. Calculer $\nabla u(x^k)$ et $h^k := (I - F^\dagger F)\nabla u(x^k)$
3. Si $\|h^k\| < \varepsilon$ stop. Si non,
4. Calculer $x^{k+1} := x^k - \theta^k h^k$ où le pas $\theta^k > 0$ est déterminé par

$$u(x^{k+1}) = \min_\theta u(x^k - \theta h^k)$$

5. incrémenter k de 1 et retourner en 2

Remarque 3.4 *Il est prudent d'ajouter une correction pour s'assurer que les x^k calculés restent bien dans la variété admissible, ce qui peut être perdu autrement en raison des erreurs d'arrondi. On peut intercaller à une fréquence à choisir un pas de projection sur $\mathcal{C}$ entre deux pas de gradient, soit le faire systématiquement à tous les pas, remplaçant la formule $x^{k+1} := x^k - \theta^k h^k$ par $x^{k+1} := (I - F^\dagger F)(x^k - \theta^k h^k) + F^\dagger f$.*

Les propriétés de convergence de cet algorithme se déduisent de son interprétation comme algorithme de gradient à pas optimal sur la restriction de u à $\mathcal{C}$.

Algorithme d'Uzawa

On a indiqué plus haut que le théorème de Kuhn et Tucker demeure pour des contraintes égalité affines, à ceci près que le signe des multiplicateurs n'est plus fixé.

En conséquence, l'algorithme d'Uzawa demeure, en supprimant l'opération de projection sur le cone positif. La preuve de convergence est inchangée.

Programmation quadratique

Un problème classique, dont on verra qu'il joue un rôle par la suite, est le problème d'optimiser une forme quadratique sous des contraintes affines. Il s'agit donc de minimiser

$$u(x) = \frac{1}{2} x^t Q x + q^t x$$

où Q est une matrice symétrique positive définie et q un vecteur de $\mathbb{R}^n$, sous les contraintes (3.11).

On laisse le lecteur démontrer (exercice) que la solution s'obtient conjointement avec le multiplicateur de Lagrange p en résolvant le système linéaire

$$\begin{array}{lcl} Qx + F^t p & = & -q\,, \\ Fx & = & f \end{array} \tag{3.13}$$

Théorème 3.12 *Le systèmes d'équations (3.13) a une solution unique quelque soit F si et seulement si F est surjective et la restriction de Q au noyau de F est définie (positive ou négative).*

Preuve Il suffit de vérifier si le système *homogène*, c'est à dire obtenu en remplaçant q et f par 0, a zéro pour seule solution. Soit donc (x, p) satisfaisant le système homogène. Multiplions la première équation à gauche par x^t et tenons compte de la deuxième pour constater qu'on doit avoir $x^t Q x = 0$. Mais ce x appartient nécessairement au noyau de F. Donc $x^t Q x = 0$ implique $x = 0$ si et seulement si la restriction de Q à ce noyau est définie. Mais dans ce cas, on doit aussi avoir $F^t p = 0$, qui implique $p = 0$ si et seulement si F est surjective.

Si Q est inversible, le système (3.13) admet la solution

$$x^* = Q^{-1} F^t (F Q^{-1} F^t)^{-1} (F Q^{-1} q + f) + Q^{-1} q\,.$$

Cependant, demander l'inversibilité de Q est trop, puisque seule doit être positive définie sa restriction au noyau de F. On peut donner une formule explicite qui n'utilise que cette hypothèse, en fonction de la décomposition en valeurs singulières de F :

$$F = U \Sigma V = U [\Sigma_1 \quad 0] \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}$$

(où U et V sont des matrices orthogonales de type $p \times p$ et $n \times n$ respectivement et Σ_1 est la matrice diagonale des valeurs singulières, V_2^t engendre Ker F) comme

$$x^* = (I - V_2^t(V_2QV_2^t)^{-1}V_2Q)V_1^t\Sigma_1^{-1}U^t f - V_2^t(V_2QV_2^t)^{-1}V_2q\,.$$

En pratique, on a construit des algorithmes d'élimination très spécialisés, plus rapides que le calcul d'une décomposition en valeurs singulières suivi de l'inversion de $V_2QV_2^t$.

3.4.2 Contraintes nonlinéaires

Nous sommes maintenant dans les notations de (3.10).

Algorithme à la Uzawa

Le théorème de Kuhn et Tucker n'est plus valide si les contraintes ne sont pas affines. On peut tenter d'appliquer encore l'algorithme d'Uzawa. On n'est assuré ni de sa convergence, ni du fait que s'il converge ce soit vers l'optimum. Voici une brève analyse de cette question.

Remarquons que la fonction étendue

$$\varphi(x) = \sup_\lambda(\lambda, f(x))$$

vaut 0 si x est admissible, et $+\infty$ si non. Ainsi le problème posé, de minimiser $u(x)$ sous les contraintes $f(x) = 0$, est équivalent au problème de minimiser $u(x) + \varphi(x)$, soit encore de chercher

$$\min_{x\in\mathbb{R}^n} \sup_\lambda(u(x) + (\lambda, f(x)))\,.$$

On a encore un minsup, et ceci justifie qu'on fasse un algorithme de type gradient ascendant en λ. Mais ce que calcule l'algorithme d'Uzawa, c'est le

$$\max_\lambda \min_{x\in\mathbb{R}^n} (u(x) + (\lambda, f(x)))\,.$$

En général, ces deux quantités sont différentes, la deuxième inférieure à la première, ce qu'on appelle le "saut de dualité".

Cette méthode ne doit donc être tentée qu'avec circonspection, et si elle converge il convient de vérifier si f s'annulle au point trouvé. (Ce qui n'est garanti dans le cas convexe que par le fait que $\min_x \sup_p = \max_p \min_x$.)

Programmation quadratique séquentielle

Si les conditions requises sont satisfaites, c'est à dire si les $\nabla f_i(x^*)$ sont linéairement indépendants, le point recherché est caractérisé par le théorème de Lagrange. C'est à dire qu'il est, avec le multiplicateur (vectoriel) de Lagrange λ, solution du système

$$\begin{array}{lcl} \nabla u(x) + \nabla f(x)\lambda & = & 0\,, \\ f(x) & = & 0\,. \end{array} \tag{3.14}$$

Ici, $\nabla f(x)$ désigne la matrice jacobienne de f transposée $(f'(x))^t$, de même que $\nabla u(x)$ désigne le transposé de $u'(x)$.

Une idée naturelle, et fructueuse, consiste à essyer de résoudre ce système d'équations non linéaires par la méthode de Newton.

Nous introduisons quelques notations à cet effet. On notera $Q := D_x^2(u(x)+(\lambda, f(x)))$ la matrice symétrique des dérivées secondes en x du lagrangien, et $F := f'(x)$ la matrice jacobienne de f. En outre, pour alléger encore les notations, on notera $f^k := f(x^k)$ et de même pour toutes les fonctions de x et λ.

Avec ces notations, l'algorithme de Newton s'écrit

$$\begin{array}{lcl} Q^k x^{k+1} + (F^k)^t \lambda^{k+1} & = & Q^k x^k - \nabla u^k \,, \\ F^k x^{k+1} & = & F^k x^k - f^k \,. \end{array}$$

On remarque que ce système, où les membres de droite sont connus à l'étape k, et les inconnues sont x^{k+1} et λ^{k+1}, a exactement la même forme que le système (3.13). Il peut donc être résolu à l'aide un algorithme de programmation quadratique —d'où le nom de cette méthode.

Ajoutons que la théorie de la seconde variation montre que les conditions énoncées au théorème 3.12 sont excatement ici la condition de qualification des contraintes d'une part, et la condition suffisante du deuxième ordre pour un minimu local sous contraintes d'autre part.

On peut donc énoncer le théorème :

Théorème 3.13 *Si l'optimum x^* recherché existe, et si la condition de qualification des contraintes y est satisfaite ainsi que la condition suffisante locale du deuxième ordre, il existe un voisinage de x^* dans lequel l'algorithme de programmation quadratique séquentielle converge.*

Bien sûr, avec ses qualités —très grande vitesse de convergence—, cet algorithme partage les défauts de la méthode de Newton : faible robustesse, et nécessité de ce fait de partir avec une assez bonne estimée de x^*.

Chapitre 4

Programmation linéaire et programmation dynamique

Les deux sujets qu'effleure ce chapitre, à titre d'introduction, ont en commun de se situer à la frontière de l'optimisation continue et de l'optimisation combinatoire.

La programmation linéaire parle de variables continues sous des contraintes continues, mais le premier pas dans l'étude de ce problème est de montrer qu'un nombre fini de points sont candidats à être optimaux, et l'algorithme du simplexe peut être vu comme une façon habile de parcourir cet ensemble fini.

De son côté, la programmation dynamique sera d'abord présentée comme un problème de recherche du plus court chemin dans un graphe, donc fondamentalement combinatoire. Mais on verra ensuite qu'un procédé classique ramène un problème en variable d'espace continue (et variable de temps discrète) à un tel graphe.

4.1 Programmation linéaire

4.1.1 Position du problème

Bien des modèles en ingéniérie et en économie mènent à considérer des problèmes où critère et contraintes sont linéaires (donc pas α-convexe) et les variables positives. Ces problèmes ont reçu le (vieux) nom de "programmes linéaires". Ils ont été étudiés dès les origines de ce qu'on devait appeler la "recherche opérationnelle", notamment par G.B. Dantzig dès le début des années 1950 (le plus ancien article cité remonte à 1951), et le développement des méthodes numériques afférentes est contemporain de l'apparition des ordinateurs.

Comme modèles simplifiés très naturels de situations concrètes (coûts et ressources consommées proportionnels aux quantités), ces problèmes ont justifié un effort algorithmique considérable, et ce jusque dans les années plus récentes, comme le bruit fait par les Bell labs autour de la "méthode de Karmarkar" l'a montré.

Nous nous contenterons ici de donner un aperçu des résultats de base et des idées sous-jacentes à l'algorithme du simplexe, comme introduction à l'utilisation des programmes existants. En effet, écrire un nouveau programme de programmation linéaire, que ce soit par l'algorithme du simplexe ou une autre méthode, est un exercice *strictement réservé aux professionnels* de la chose, tant les "packages" qui existent sont (nombreux et) perfectionnés.

On utilisera les inégalités entre vecteurs

$$x \geq y \quad \Leftrightarrow \quad x_i \geq y_i \,,\ i = 1, \ldots,$$

$$x > y \quad \Leftrightarrow \quad x \geq y \text{ et } x \neq y \,,$$

$$x >> y \quad \Leftrightarrow \quad x_i > y_i \,,\ i = 1, \ldots \,.$$

Le problème peut en général être formulé de la façon suivante.

Le critère à minimiser est déterminé par un vecteur c de $\mathbb{R}^n$, et est donné par

$$u(x) = (c, x) = \sum_{i=1}^{n} c_i x_i$$

Les contraintes sont d'une part

$$x \geq 0 \,,$$

et d'autre part deux jeux de contraintes définies par deux matrices A et B, de dimensions respectives $p \times n$ et $q \times n$, et deux vecteurs a et b de dimension p et q définissant les contraintes égalité et inégalité par

$$Ax = a \,, \qquad Bx \leq b \,.$$

Dans les discussions qui suivent, il faut avoir à l'esprit que n, p et q peuvent être de l'ordre de plusieurs milliers, voire dizaines de milliers.

La première remarque est qu'au moins au plan théorique, on peut remplacer les contraintes inégalité par des contraintes égalité et inversement par les artifices suivants. En introduisant q variables supplémentaires $y \in \mathbb{R}^q$, on peut remplacer $Bx \leq b$ par

$$Bx + y = b \,, \quad y \geq 0 \,.$$

Au contraire, si on veut privilégier les contraintes inégalité, on peut remplacer $Ax = a$ par

$$Ax \leq a \text{ et } Ax \geq a \,.$$

Bien sur, la première de ces opérations *augmente de q le nombre de variables*, tandis que la deuxième *augmente de p le nombre de contraintes*. Deux opérations qui ne sont pas souhaitables au vu des dimensions que nous évoquions. Ce sont par contre des artifices utiles pour étudier les propriétés théoriques du problème.

Nous utiliserons dans l'étude théorique la *forme standard égalité* caractérisée par m contraintes égalité *signées* :

$$Ax = b \,, \qquad b \geq 0 \,,$$

ce qui est toujours possible, quitte à multiplier certaines contraintes par -1, et toujours les contraintes de positivité

$$x \geq 0 \,.$$

4.1.2 Étude du polyèdre

Pour comprendre le problème de programmation linéaire, il faut étudier l'ensemble des points défini par les contraintes linéaires et de positivité, appelé *polyèdre*, que nous prenons sous la forme standard égalité $Ax = b$. Nous noterons P cet ensemble.

Nous appelons encore n la dimension de x, sachant qu'elle a pu être augmentée pour donner cette forme aux contraintes. Nous appelons m le nombre de contraintes, et supposons que $m < n$. Nous supposons même plus que cela, à savoir que

$$\text{rang} A = m\,. \tag{4.1}$$

En effet, si-non A a des lignes linéairement dépendantes d'autres, et soit la même dépendance linéaire se retrouve dans les coordonnées de b, et cette ligne est surnuméraire, et peut être omise sans rien changer au problème, soit les coordonnées de b n'exhibent pas la même dépendance, et le polyèdre est vide.

Nous introduisons à cette fin la terminologie suivante :

Définition 4.1 (Direction de $\mathbb{R}^n$) *Nous appelons* direction *l'ensemble des vecteurs portés par une même demi-droite, c'est à dire multiples positifs d'un d'entre-eux.*

Ainsi, soit $h \in \mathbb{R}^n$, il définit une direction $\{\theta h \mid \theta \in \mathbb{R}_+\}$, et tout vecteur de cette direction définit la même direction.

Une direction sera réputée "admissible" si elle est composée de vecteurs à coordonnées positives (ce qui est le cas dès qu'un de ses vecteurs l'est) et si elle appartient au noyau de A. Ainsi, si $x \in \mathsf{P}$, $x + w \in \mathsf{P}$ pour tout w dans cette direction. Ce sont des directions dans lesquelles P est non borné. (Très précisément, ces directions définissent des "points à l'infini" du polyèdre au sens de la géométrie projective.)

Le résultat essentiel que nous visons est le suivant :

Théorème 4.1 *Les points d'un polyèdre peuvent tous être obtenus comme combinaison convexe d'un nombre fini de ses points (appelés* sommets*) plus une somme d'éléments d'un nombre fini de directions (appelées* directions admissibles extrêmales *ou* sommets à l'infini*). Réciproquement, toute combinaison de cette forme appartient au polyèdre.*

Ainsi tout polyèdre est caractérisé par un nombre fini de sommets, à distance finie ou à l'infini, et est constitué par l'ensemble de leurs combinaisons convexes. En particulier, s'il est borné, un polyèdre se réduit au *polytôpe* de toutes les combinaisons convexes de ses sommets (en nombre fini).

Pour démontrer ce résultat, nous introduisons la définition suivante :

Définition 4.2 (Points extrémaux) *Étant donné un ensemble convexe* C*, on appelle* points extrémaux *de* C *les points de* C *qui ne peuvent être représentés comme une combinaison convexe propre d'autres points de* C.

Ainsi, si $\hat{x}$ est un point extrémal de C, et si x_1 et x_2 sont deux points de C tels que $\hat{x} = \lambda x_1 + (1 - \lambda)x_2$, alors nécessairement $\lambda = 0$ ou $\lambda = 1$, et $\hat{x}$ coincide avec x_1 ou x_2.

La même définition s'appliquera aux directions admissibles, sachant que deux vecteurs colinéaires et de même sens (proportionnels dans un rapport positif) représentent la même direction.

Lemme 4.2 *Un point admissible (i.e. tel que $x \geq 0$ et $Ax = b$) [resp une direction admissible h, i.e. telle que $h \geq 0$ et $Ah = 0$] est extrémal[e] si et seulement si les colonnes de A correspondant aux coordonnées* non nulles *de x [resp h] sont linéairement indépendantes [resp. ont un défaut de rang égal à 1].*

Les coordonnées non nulles de x sont dites "de base". On notera que la propriété ci-dessus implique notamment qu'un point extrémal a au plus m coordonnées de base.

Démonstration du lemme Pour simplifier l'écriture, nous supposons que ce sont les p premières colonnes de A qui sont concernées, et donc les $n - p$ dernières coordonnées de x qui sont nulles. Nous partitionnons A et x en

$$A = [\bar{A}\ \tilde{A}]\,, \qquad \begin{pmatrix} \bar{x} \\ \tilde{x} \end{pmatrix}$$

Soit un point de P qui satisfait la condition énoncée, c'est à dire que $x = (\bar{x} \quad 0)$, et donc $\bar{A}\bar{x} = b$. Si ce point est combinaison convexe propre de deux autres points de P alors, il est intérieur au segment joignant ces deux points, ce qui veut dire qu'il existe un vecteur $w \neq 0$ tel que $x + w$ et $x - w$ appartiennent à C. En particulier, ceci implique que les composantes de w hors base soient nulles (si non, soit $x+w$ soit $x-w$ aurait des composantes négatives), et donc aussi que $\bar{A}\bar{w} = 0$, où $\bar{w}$ désigne bien sûr les m premières coordonnées de w. Mais par hypothèse, $\bar{A}$ est injective, donc $w = 0$, ce qui contredit l'hypothèse.

Réciproquement, soit x un point de P, $\bar{x}$ l'ensemble de ses coordonnées non nulles, que nous regroupons au début de la numérotation, et $\bar{A}$ la matrice des colonnes corrsepondantes. Si les colonnes de $\bar{A}$ ne sont pas linéairement indépendantes (A n'est pas injective), il existe un vecteur $\bar{w}$ non nul tel que $\bar{A}\bar{w} = 0$. Comme les coordonnées de $\bar{x}$ sont toutes strictement positives, il existe ε assez petit pour que, en choisissant $\|\bar{w}\| \leq \varepsilon$ ce qui est toujours loisible, les coordonnées de $\bar{x} + \bar{w}$ et de $\bar{x} - \bar{w}$ soient encore toutes positives. Alors, en complètant $w = (\bar{w} \quad 0)$, on voit que $Aw = 0$, et que $x + w$ et $x - w$ appartiennent tous les deux à C. Donc $x = 1/2(x + w) + 1/2(x - w)$ n'est pas un point extrémal de P.

On laisse le lecteur répéter la même preuve pour les directions admissibles extrémales, en se souvenant que la différence entre le nombre des vecteurs considéré et le rang du système qu'ils forment, ou *défaut de rang*, est la dimension du noyau de la matrice dont ils sont les colonnes.

Cette caractérisation montre qu'il ne saurait y avoir qu'un nombre fini de points extrémaux à P. Il suffit de tester tous les ensembles de m colonnes de A, et pour ceux qui sont linéairement indépendants, de verifier si $A^{-1}b$ est positif. On fera de même avec les ensembles de $m + 1$ colonnes de rang m pour chercher les directions admissibles optimales.

Démonstration du théorème Soit x un point de C, et supposons que ce ne soit pas un point extrémal. Comme précédement, il existe un vecteur w du noyau de A ayant les mêmes coordonnées nulles que x, et tel que $x - w$ et $x + w$ appartiennent à C. Regardons $x + tw$, $t > 0$. Si w n'est pas une direction admissible de C (il a des coordonnées négatives), pour un certain t^+ une des coordonnées de ce vecteur s'annule, les autres étant encore positives. On peut faire de même avec $x - tw$. (Si w est une direction admissible de C, $-w$ ne l'est pas, et réciproquement.) Donc, soit t^+ et t^- sont tous les deux définis, et x peut être représenté comme une combinaison convexe de deux vecteurs qui ont une coordonnée de plus nulle chacun : $x = t^-/(t^+ + t^-)x^+ + t^+/(t^+ + t^-)x^-$, soit on a une représentation comme une somme $x = x^- + t^-w$ où w est une direction admissible, et x^- a une coordonnée nulle de plus que x. En répétant ce processus pour les éléments de cette représentation récursivement, on obtient le théorème. (On ne fait qu'un nombre fini de fois cette opération, puisque le nombre de coordonnées

non nulles diminue chaque fois, et la construction s'arrête quand A n'a plus de noyau —ou un noyau de dimension 1 pour les directions admissibles—.)

On déduit de ce théorème le corollaire fondamental suivant :

Corollaire 4.3 *Si le polyèdre des contraintes n'est pas vide, soit le critère n'a pas d'infimum fini, soit un des points extrémaux (ou sommets) du polyèdre est solution.*

Démonstration Soit il existe une direction admissible w telle que $(c, w) < 0$. Alors, comme on peut ajouter tw, t positif arbitraire, à tout point du polyèdre sans en sortir, clairement, le critère (c, x) peut être rendu arbitrairement grand négatif. Soit, pour toute direction admissible w, $(c, w) \geq 0$. Alors, si un point du polyèdre a w dans sa décomposition comme au théorème ci-dessus, on peut retirer cette composante. On reste dans le polyèdre, et on améliore le critère. Nous ne considérons donc que des points combinaison convexe des sommets x_1 à x_N :

$$x = \sum_{i=1}^{N} \lambda_i x_i, \qquad \lambda_i \geq 0, \qquad \sum_{i-1}^{N} \lambda_i = 1\,.$$

Alors,

$$(c, x) = \sum_{i=1}^{N} \lambda_i (c, x_i)\,.$$

Si le critère (c, x) est minimum sur C, tous les (c, x_i) sont supérieurs ou égaux à (c, x). Donc seuls peuvent être non nuls les λ_i pour lesquels on a l'égalité, et ce sont là des sommets solution. Ou bien — et c'est le cas générique— un seul sommet est solution, et il ne peut être représenté comme ci-dessus de manière non dégénérée.

4.1.3 L'algorithme du simplexe

Nous ne donnons ici qu'un bref aperçu du principe de l'algorithme le plus célèbre pour résoudre numériquement le programme linéaire, l'algorithme du simplexe. Il en existe d'autres aujourd'hui, plus rapides ... dans la plupart des configurations.

Remarquons d'abord qu'en principe, puisque le nombre de sommets est fini et qu'on sait les déterminer tous, il suffit de comparer la valeur du critère à tous ces sommets et prendre le meilleur. Ce qui s'oppose à cette approche naïve est le nombre potentiellement très grand de sommets du polyèdre. On considère couramment des problèmes où n et m sont tous les deux en milliers. Or il y a $n!/m!(n-m)!$ combinaisons de m colonnes à tester. C'est à dire, si $n = 2000$ et $m = 1000$, un nombre de l'ordre de 10^{600} combinaisons.

Il faut faire autrechose. L'algorithme du simplexe va explorer des sommets d'une façon moins naïve, et d'habitude plus efficace. On sait malheureusement exhiber des problèmes pour lesquels cet algorithme explore *tous* les sommets. Mais on sait que ce n'est *génériquement* pas le cas, et que le simplexe est génériquement polynomial en $m + n$.

Le principe de l'algorithme est donc le suivant. Étant donné le choix d'une "base", c'est à dire de m colonnes indépendantes de A formant $\bar{A}$, de sorte que $A = [\bar{A}\ \tilde{A}]$, et un découpage correspondant des coordonnées de tout x en coordonnées en base $\bar{x}$ et hors base $\tilde{x}$, on a nécessairement

$$\bar{A}\bar{x} + \tilde{A}\tilde{x} = b\,,$$

soit encore

$$\bar{x} = -\bar{A}^{-1}\tilde{A}\tilde{x} + \bar{A}^{-1}b\,. \tag{4.2}$$

Donc, en paramétrisant x via $\tilde{x}$:

$$(c, x) = (\tilde{c}^t - \bar{c}^t \bar{A}^{-1} \tilde{A})\tilde{x} + \bar{c}^t \bar{A}^{-1} b \,,$$

que nous réecrivons avec une notation évidente

$$(c, x) = (\tilde{w}, \tilde{x}) + \bar{c}^t \bar{A}^{-1} b \,.$$

Si l'itérée x^k de l'algorithme est un sommet correspondant à cette base, soit $\tilde{x}^k = 0$, on va chercher à améliorer le critère en repérant la coordonnée de $\tilde{w}$ la plus négative, disons $\tilde{w}_M$ et en donnant une valeur positive à la coordonnée $\tilde{x}_M$ correspondante de x. Soit donc e_M le vecteur de base numéro M de $\mathbb{R}^n$, et essayons des x de la forme

$$x = x^k + t e_M \,.$$

On est sûr de faire décroitre le critère ce faisant. Le t maximum permis est atteint la première fois qu'une autre coordonnée de $\bar{x}$, tel que calculé par (4.2) passe par zéro. On s'arrête à cette valeur de t, on fait ainsi rentrer la coordonnée M dans la base et sortir celle qui s'est annulée. Et on itère.

L'algorithme s'arrête quand toutes les coordonnées de $\tilde{w}$ sont positives : on ne peut plus améliorer le critère.

Il reste à dire comment *initialiser* l'algorithme. En effet, nous avons indiqué comment passer d'un sommet du polyèdre à un autre, mais comment trouver un premier sommet ? Nous allons indiquer comment utiliser le même algorithme pour trouver un sommet initial, et en même temps déterminer s'il existe un tel sommet, c'est à dire si l'ensemble des états admissibles est non vide. Il peut en effet se faire que les contraintes $Ax = b$, $x \geq 0$ soient incompatibles, mais ceci même est difficile à découvrir.

La méthode va consister à examiner un autre problème linéaire qui présente la particularité d'avoir un sommet évident, et de chercher, s'il existe, un sommet du problème d'origine.

Supposons qu'on s'est ramené à $b \geq 0$, ce qu'on peut toujours faire en changeant le signe des lignes de A si nécessaire. Considérons le problème de programmation linéaire portant sur les variables (positives) (x, w), $x \in \mathbb{R}^n$, $w \in \mathbb{R}^m$, dont les contraintes sont

$$Ax + w = b$$

et le critère

$$u(x, w) = \sum_{i=1}^{m} w_i \,.$$

Comme promis, les contraintes admettent une solution évidente, qui est un sommet : $x = 0$, $w = b$. Et comme ce critère est toujours positif ou nul, son optimum sera zéro si et seulement s'il existe une solution des contraintes avec $w = 0$, soit un point admissible du problème d'origine. En outre, évoluant de point extrémal en point extrémal, le simplexe nous donnera un point extrémal, qui pourra à son tour servir d'initialisation pour le problème d'origine.

Il y a beaucoup à dire à partir de là (on a écrit des livres entiers). Que faire dans le simplexe si la nouvelle base n'est pas indépendante ? si deux coordonnées s'annulent en même temps ? etc. On laisse le lecteur imaginer des parades simples à ces questions.

Plus intéressant : comment simplifier le calcul de $\bar{A}^{-1}$ pour la nouvelle base en utilisant le fait qu'on connaissait cette inverse pour une matrice ne différant que par une colonne ?

Pour toutes ces questions, nous renvoyons le lecteur aux livres spécialisés.

4.1.4 Rudiments de dualité

On ne peut pas parler de programmation linéaire sans évoquer la dualité, qui constitue, comme on va le voir, un outil très utile.

Dans ce numéro, et pour l'élégance des formules obtenues, nous allons supposer qu'on cherche à maximiser un critère linéaire $u = (c, x)$. Cela revient évidemment à changer c en $-c$, mais comme nous n'avons jamais fait d'hypothèse sur le signe des éléments de c, cela est sans conséquence.

Partons donc d'un problème standard, que nous écrivons

$$\max_x c^t x\,, \qquad x \geq 0\,, \qquad Ax = b\,,$$

en notant de façon explicite $c^t x$ le produit scalaire (c, x). Supposons que nous connaissions un vecteur y de $\mathbb{R}^p$ (p est le nombre de contraintes) tel que

$$y^t A \geq c^t\,.$$

Alors, pour tout $x \geq 0$, on a $c^t x \leq y^t A x$. Mais par ailleurs, pour tout x admissible, $Ax = b$, de sorte que l'inégalité ci-dessus donne

$$c^t x \leq y^t b\,. \tag{4.3}$$

Avant d'aller plus loin, supposons que la contrainte $Ax = b$ provient en fait d'une contrainte inégalité, et qu'on a augmenté l'état de "variables d'écart" pour en faire des contraintes égalité, c'est à dire, avec un abus de notations évident, que le vecteur x ci-dessus est en fait de la forme

$$x = \begin{pmatrix} x \\ \xi \end{pmatrix}\,,$$

que A est donc de la forme

$$A = [\ A \quad I\ ]\,,$$

de sorte que la contrainte est

$$Ax + \xi = b, \qquad \xi \geq 0, \qquad x \geq 0\,,$$

donc équivalente à

$$Ax \geq b\,, \qquad x \geq 0\,. \tag{4.4}$$

La "contrainte" sur y se lit alors

$$y^t[\ A \quad I\ ] \geq [\ c^t \quad 0\ ]$$

soit

$$y^t A \geq c^t\,, \qquad y \geq 0 \tag{4.5}$$

Ainsi, pour tout x admissible au sens de (4.4) et tout y admissible au sens du *problème dual* dont l'admissibilité est définie par (4.5), on a (4.3). Le problème

$$\min_y (b, y)$$

soumis aux contraintes (4.5) est appelé *problème dual* de celui d'origine (rappelons que nous en avons ici fait un problème de maximisation sous contraintes inégalité).

On remarque la parfaite symétrie entre problème primal et dual, de sorte que toute affirmation concernant leur intéraction peut être énoncée dans un sens ou dans l'autre.

On voit que si on trouve x et y admissibles pour les problèmes primal et dual tels que $(c, x) = (b, y)$, alors ils sont nécessairement solution de ces problèmes. Cette remarque est la base de méthodes efficaces pour résoudre le problème de programmation linéaire, visant à minimiser la différence, toujours positive ou nulle, $(b, y) - (c, x)$ parmi les x et y admissibles.

On voit aussi aisément que si le problème dual a un ensemble d'états admissibles non vide, le problème primal a son critère borné supérieurement (et mutatis mutandis pour le problème dual si le problème primal admet des états admissibles).

Ces constatations élémentaires ont des réciproques, que nous ne démontrerons pas :

Théorème 4.4 *Chacun des deux problèmes, primal et dual, a un ensemble d'états admissibles non vide et un suprémum fini (donc une solution) si et seulement si il en va de même de l'autre. Si un des deux problèmes a un extremum infini, l'autre n'a pas d'état admissible.*

Donc les deux problèmes ont une solution ou n'en ont pas simultanément, l'absence de solution pouvant provenir de l'absence d'états admissibles, ou du fait que le critère n'est pas borné.

Théorème 4.5 *Si les problèmes de programmation linéaire primal et dual ont une solution, il existe x^* et y^* tels que $(c, x^*) = (b, y^*)$. Réciproquement, toute paire admissible (x^*, y^*) satisfaisant cette égalité est optimale.*

Enfin, faisons remarquer que la connaissance de y^*, par exemple, permet de trouver x^*. En effet, remarquons qu'on a donc toujours, pour tout x et y admissibles

$$c^t x \leq y^t A x \leq y^t b\,,$$

de sorte que si $c^t x = y^t b$, non seulement x et y coincident avec les optimums, mais aussi, toutes les inégalités ci-dessus sont des égalités. Or, l'égalité

$$(y^*, Ax^* - b) = 0$$

alors que $y^* \geq 0$ et $Ax - b \leq 0$ montre que pour toute coordonnée de y^* non nulle, la contrainte correspondante en x est "saturée" en x^* (satisfaite avec l'égalité). De même, l'égalité $(c^t - (y^*)^t A)x^* = 0$, que nous pouvons réécrire

$$(A^t y^* - c, x^*) = 0$$

alors que $A^t y^* - c \geq 0$ et $x^* \geq 0$ implique que pour toute contrainte duale non saturée, la coordonnée corrsepondante de x^* est nulle. Comme nous l'avons vu plus tôt, connaître les contraintes saturées (les ξ_i nuls) et les x_i nuls suffit à déterminer x^* par l'équation linéaire qu'on en déduit.

L'utilité de cette théorie est double. D'une part, comme nous l'avons indiqué, elle est le fondement de méthodes numériques efficaces, ou d'améliorations de l'algorithme du simplexe. D'autre part, elle permet toujours de choisir si l'on préfère résoudre le problème primal ou dual. L'un a plus de contraintes (inégalité) que de variables, et l'autre plus de variables que de contraintes. Suivant les packages de résolution disponibles, l'un peut être préférable à l'autre.

4.2 Programmation dynamique

Nous allons partir d'une vision combinatoire de la programmation dynamique, pour aboutir à une utilisation dans des problèmes authentiquements "dynamiques" en variables continues, donc voisins des préoccupations du reste de ce cours.

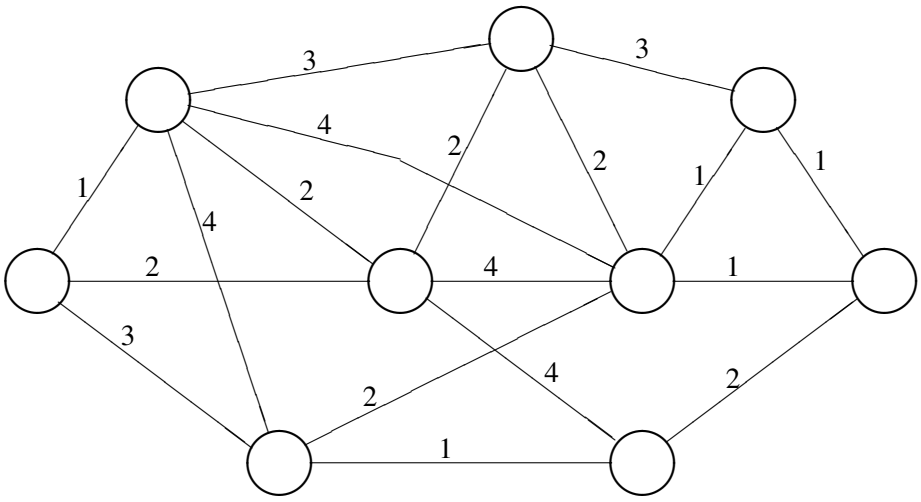

FIGURE 4.1 – Graphe et “poids” des arcs

4.2.1 Plus court chemin dans un graphe orienté

Le problème le plus simple

Nous considérons un graphe orienté, ici de gauche à droite, comme celui de la figure 1, dont chaque arc est muni d'une “longueur” ou “poids”.

Le problème posé est de trouver le chemin de “longueur” ou “poids” minimum du nœud initial, le plus à gauche, au nœud terminal, le plus à droite. C'est manifestement un problème “fini” : le graphe ne comporte qu'une vingtaine de chemins. On pourrait donc tous les lister et choisir le plus court. Mais on voit bien que le nombre de chemins croit combinatoirement avec la taille du graphe, et une procédure aussi rustique ne s'étendra pas à des situations beaucoup plus complexes. (Le seul travail de répertorier tous les chemins devient exorbitant.)

Nous allons indiquer une procédure qui ne croit que linéairement avec le nombre de nœuds, ou plus précisément comme le produit du nombre de nœuds par le nombre moyen d'arcs par nœud.

Le principe de la méthode est de marquer chaque nœud avec la longueur du chemin minimal de ce nœud jusqu'à la fin. Cette procédure sera rapide en raison de la remarque banale mais essentielle qui suit :

Proposition 4.6 (Principe de Bellman) *Le chemin optimal a la propriété (dite “principe d'optimalité” de Bellman)*[1] *qu'entre tout nœud $\mathcal{N}$ par où il passe et la fin du chemin, il est optimal pour le problème d'aller de ce nœud $\mathcal{N}$ jusqu'à la fin. (Ce que nous appellerons le* sous-problème initialisé en $\mathcal{N}$*.)*

Démonstration Comme la longueur totale du chemin est la somme de la longueur parcourue du nœud initial au nœud $\mathcal{N}$ plus la longueur de $\mathcal{N}$ jusqu'à la fin, si on pouvait trouver un chemin plus court pour cette dernière longueur, —le sous problème initialisé en $\mathcal{N}$— on pourrait, en le concaténant avec

1. Il s'agit de Richard Bellman, dont on peut contester qu'il ait inventé la programmation dynamique, pas qu'il en ait compris le premier toute la puissance et toute la généralité.

le chemin optimal entre le début et $\mathcal{N}$, trouver un chemin global plus court que le chemin optimal, ce qui est une contradiction.

Cette démonstration semble être une tautologie tant la propriété est évidente. Pourtant, nous allons nous servir de ce "principe de Bellman" en le reformulant un peu.

Pour tout nœud, *si* le chemin optimal passe par ce nœud, de ce nœud jusqu'à la fin, il utilise le chemin optimal pour ce sous problème. En particulier, depuis un nœud $\mathcal{N}$, si la longueur du chemin optimal jusquà la fin —c'est à dire la *valeur* optimale des sous problèmes si non leur solution complète— est connue pour tous les nœuds immédiatement aval (c'est à dire séparés par un seul arc), alors résoudre le sous problème initialisé en $\mathcal{N}$ est immédiat. En effet, *si* le chemin optimal (depuis $\mathcal{N}$) passe par un certain $\mathcal{N}'$ aval, la longueur en est la somme de la longueur de l'arc séparant $\mathcal{N}$ de $\mathcal{N}'$ ajoutée à la valeur optimale du sous-problème initialisé en $\mathcal{N}'$. Ainsi, depuis $\mathcal{N}$ il suffit de comparer ces valeurs, et de retenir la meilleure (la plus petite). On aura ainsi à peu de frais la valeur du sous-problème initialisé en $\mathcal{N}$.

On obtient ainsi l'algorithme suivant :

Algorithme Programmation dynamique simple

1. Marquer le nœud terminal avec la valeur 0.
2. En tout nœud dont tous les nœuds immédiatement aval sont déjà marqués, faire :
   - pour chaque nœud immédiatement aval, calculer la somme de la longueur de l'arc vers ce nœud et de la valeur de ce nœud.
   - Prendre la plus petite de ces sommes pour valeur du nœud courant, et le marquer.
   - Marquer le, ou les, arc(s) donnant la valeur retenue.
3. Retourner en (2) jusquà ce que tous les nœuds soient marqués
4. depuis le nœud initial, (comme depuis tout nœud du graphe) tout chemin n'empruntant que des arcs marqués est optimal, et a une longueur égale à la valeur marquée à ce nœud.

À titre d'exemple, nous avons dans la figure 2 marqué à chaque nœud sa valeur, et renforcé les arcs optimaux. On voit que le problème posé n'avait pas une solution unique, mais cette procédure n'en est nullement affectée.

Extensions du problème du plus court chemin

On peut étendre de nombreuses façons cet algorithme. La plus élémentaire est la suivante. On peut considérer *plusieurs* nœuds terminaux et plusieurs nœuds initiaux possibles. De plus, on peut supposer qu'un "poids" est attaché à chacun des nœuds en plus des arcs.

Dans ce dernier cas, on pourrait aussi bien attacher ce poids du nœud à tout arc qui le rejoint, se ramenant de manière triviale au problème où seuls les arcs ont un poids. On préfère, pour des raisons qui apparaîtront plus loin, considérer d'une part le poids attaché à chaque nœud terminal, et considérer que c'est la *valeur* de ce nœud pour l'algorithme de programmation dynamique, et associer les poids des autres nœuds aux arcs qui les quittent.

On aboutit ainsi à l'algorithme suivant.

Algorithme Programmation dynamique

1. Marquer les nœuds terminaux avec leur valeur donnée.
2. En tout nœud dont tous les nœuds immédiatement aval sont déjà marqués, faire :

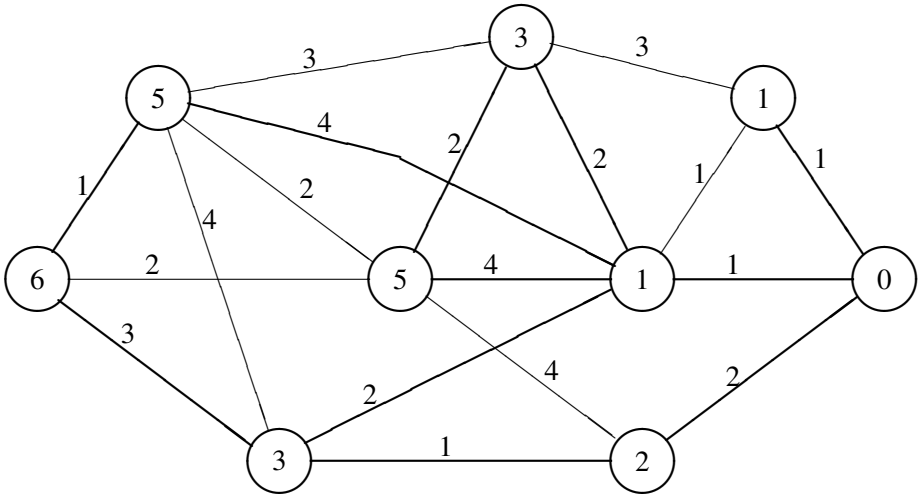

FIGURE 4.2 – Le graphe de la figure 1 avec les valeurs et les arcs optimaux

- pour chaque nœud immédiatement aval, calculer la somme de la longueur de l'arc vers ce nœud et de la valeur de ce nœud.
- Prendre la plus petite de ces sommes pour valeur du nœud courant, et le marquer.
- Marquer le, ou les, arc(s) donnant la valeur retenue.

3. Retourner en (2) jusquà ce que tous les nœuds soient marqués
4. Tout chemin partant d'un nœud initial de valeur minimum et n'empruntant que des arcs marqués est optimal, et a une longueur égale à la valeur marquée à ce nœud.

Dans l'exemple de la figure 3, qui a deux nœuds initiaux possibles et trois nœuds terminaux, on a seulement attaché des poids aux nœuds terminaux, puisque des poids sur les nœuds intermédiaires auraient seulement augmenté d'autant le poids de chaque arc les quittant, sans augmenter la généralité de l'exemple.

On donne directement le graphe marqué avec les valeurs et les chemins optimaux. On conseille au lecteur de refaire l'algorithme lui-même pour constater combien il est simple et rapide. Pourtant ce graphe a plus de 110 chemins possibles.

De très nombreux problèmes combinatoires peuvent se ramener à un problème de recherche de chemin de poids minimal dans un graphe. La caractéristique essentielle pour mettre à jour une telle structure, et l'exploiter, est le sens de parcours unique. En général il provient de ce que le problème peut être organisé en "étapes" dont l'ordre est imposé par la nature du problème, ou immatériel quant à la solution cherhée de sorte qu'il peut être fixé arbitrairement. Cet aspect d'étapes successives, ou dynamique, va être examiné maintenant.

4.2.2 Système dynamique et programmation dynamique

Système dynamique

Nous examinons maintenant un cas particulier extrêmement important. Supposons que les nœuds du graphe, appelés ici *états*, peuvent être repérés par un numéro d'étape $k \in 0, 1, \ldots, N$, et pour chaque étape k soit X_k l'ensemble des états possibles à cette étape. L'hypothèse ici est que les arcs

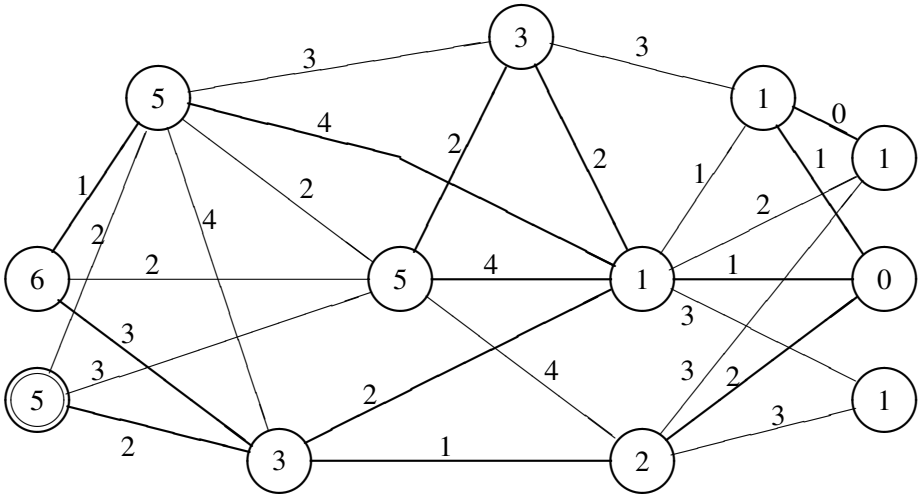

FIGURE 4.3 – Un graphe à plusieurs nœuds initiaux et terminaux

relient toujours un état d'une étape à un de l'étape suivante, et que ceci correspond au sens de parcours imposé du graphe. Indiçons les arcs issus d'un nœud $(k, x(k))$ —où $x(k) \in \mathsf{X}_k$ est un nœud de l'étape k— par un indice $u \in \mathsf{U}(k, x(k))$ appelé *commande*. $\mathsf{U}(k, x(k))$ est simplement un ensemble dont le cardinal est égal au nombre d'arcs quittant le nœud $(k, x(k))$. On voit que le graphe définit une *équation dynamique* de la forme

$$x(k+1) = f(k, x(k), u(k)) \tag{4.6}$$

puisque pour chaque nœud $(k, x(k))$ et chaque commande $u \in \mathsf{U}(k, x(k))$, il définit à quel nœud ou état conduit cet arc, état toujours situé à l'étape $k+1$.

Un chemin dans le graphe est une suite d'états $\{x(k), k = 0, \ldots, N\}$, appelée *trajectoire*. Une trajectoire peut aussi être caractérisée par l'état initial $x(0)$ et une suite de commandes $\{u(k), k = 0, \ldots, N-1\}$, qui, via l'équation (4.6), engendre une trajectoire unique.

Notons encore $L(k, x, u)$ le "poids" ou la longueur —nous dirons ici le *coût*— de l'arc issu du nœud (k, x) indicé par u, et pour tout nœud terminal $x \in \mathsf{X}_N$, $K(x)$ le coût attaché à ce nœud. Ainsi, le coût d'une trajectoire, qu'il s'agit de minimiser, est donné par

$$J = K(x(N)) + \sum_{k=0}^{N-1} L(k, x(k), u(k))\,. \tag{4.7}$$

Il y a équivalence complète entre un graphe structuré comme on l'a dit et une équation de la forme (4.6), et donc entre un problème de plus court chemin dans un tel graphe et le problème de minimisation de J donné par (4.7) avec la dynamique (4.6). Bien des problèmes seront formulés sous la forme (4.6),(4.7). Un bon moyen de les résoudre est de faire l'analogie avec le graphe, et de faire l'algorithme de programmation dynamique sur ce graphe.

Le terme même de "programmation dynamique" vient de là. L'équation (4.6) définit ce qu'on appelle un *système dynamique*. L'indice k est généralement interprété comme représentant le temps. L'équation (4.7) définit une fonctionnelle additive de la trajectoire. On recherche la commande, et éventuellement l'état initial, qui minimise ce critère ou coût.

Une forme équationnelle de la programmation dynamique

Ayant décrit le graphe et le critère par des équations, on peut décrire dans ce langage l'algorithme de la programmation dynamique. Notons $V(k,x)$ le "marquage" associé au nœud (k,x). On l'appelle plutot ici la fonction "performance", ou "de Bellman". L'algorithme que nous avons décrit s'écrit alors :

$$\forall k \in \{0,\dots,N-1\}\,, \forall x \in \mathsf{X}_k\,, \quad V(k,x) = \min_{u \in \mathsf{U}(k,x)} \left[L(k,x,u) + V(k+1, f(k,x,u))\right], \quad (4.8)$$

$$\forall x \in \mathsf{X}_N\,, \quad V(N,x) = K(x)\,. \quad (4.9)$$

L'algorithme de la programmation dynamique consiste donc à appliquer la formule (4.8) pour calculer V de proche en proche, en commençant par initialiser V avec (4.9), puis en reculant en k. Il faut avoir deux tableaux des valeurs de $V(k,\cdot)$ en mémoire, celui qu'on est en train de remplir et celui qui est utilisé dans le deuxième membre de (4.8). In fine, le tableau des $V(0,x)$ donne pour tout état initial possible le coût minimum possible.

Il faut aussi en même temps remplir un grand tableau des valeurs de u qui donnent le minimum à chaque (k,x). Ce tableau donne la *stratégie* (ou *commande en boucle fermée*) optimale, en ce que pour chaque état (k,x) il donne la commande optimale si on se trouve en cet état. Cet aspect est particulièrement utile si l'équation (4.6) constituait une approximation d'un phénomène physique, de sorte qu'on est susceptible de constater au cours de la mise en œuvre qu'on est dans un état $(k,x(k))$ différent de ce que à quoi on s'attendait. L'algorithme ci-dessus a donné une commande conseillée pour *tout* état dans le graphe. Certes, l'écart entre le phénomène réel et le modèle fait que cette commande n'est plus tout à fait optimale, mais si cet écart est faible, elle a toutes les chances de rester une "bonne" commande.

Système à état et continu

Tout le développement de la programmation dynamique a été fait en termes de graphe, avec donc l'hypothèse constante que dans l'équation (4.6), x et u prennent leurs valeurs dans des ensembles X_k et $\mathsf{U}(k,x)$ finis. Cependant, ces équations ainsi que (4.7) gardent un sens si ces ensembles sont des sous ensembles de $\mathbb{R}^n$ et $\mathbb{R}^m$, disons, respectivement. C'est à dire qu'alors l'état est constitué de n nombres réels et la commande de m nombres réels, les uns et les autres éventuellement bornés.

Les équations de la programmation dynamique (4.8)(4.9) gardent également un sens. Et on va démontrer le résultat suivant :

Théorème 4.7 *S'il existe une fonction rélle $V(\cdot,\cdot)$ satisfaisant les équations (4.8) et (4.9), en désignant par $\varphi(k,x)$ un argument du minimum dans (4.8), si la commande $u(k) = \varphi(k,x(k))$ est admissible pour x_0 (au sens où elle engendre une trajectoire qui respecte les contraintes $x(k) \in \mathsf{X}_k$, ce dont on peut s'assurer par un choix convenable des $\mathsf{U}(k,x)$), alors cette commande est optimale pour le problème défini par (4.6)(4.7) initialisé en $x(0) = x_0$.*

Démonstration Soit $\{u(0), u(1), \dots, u(N-1)\}$ une commande admissible, engendrant une trajectoire $\{x_0, x(1), \dots, x(N)\}$. En tout point de cette trajectoire, d'après (4.8), on a

$$V(k,x(k)) \leq L(k,x(k),u(k)) + V(k+1, f(k,x(k),u(k)))\,,$$

ou encore

$$V(k,x(k)) - V(k+1,x(k+1)) \leq L(k,x(k),u(k))\,.$$

Sommons cette inégalité de $k = 0$ à $N - 1$, il vient

$$V(0, x_0) - V(N, x(N)) \leq \sum_{k=0}^{N-1} L(k, x(k), u(k)) .$$

Utilisons alors (4.9) pour exprimer $V(N, x(N))$, que nous faisons repasser à droite, il reste

$$V(0, x_0) \leq K(x(N)) + \sum_{k=0}^{N-1} L(k, x(k), u(k)) ,$$

soit

$$V(0, x_0) \leq J(x_0, \{u\}) . \tag{4.10}$$

Maintenant, si la suite des $\{u(k)\}$ coincide pour tout k avec $\varphi(k, x(k))$, les inégalités dans les calculs ci-dessus sont toutes remplacées par des égalités, et on conclu que

$$V(0, x_0) = J(x_0, \varphi) . \tag{4.11}$$

La comparaison des relations (4.10) et (4.11) établit le théorème.

Système en temps continu

Dans la pratique, le système (4.6) est souvent issu de la discrétisation en temps d'un système en temps continu, ou système différentiel, de la forme

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

Si on discrétise ce système avec un pas de temps h, en notant $x_k = x(kh)$ et $u_k = u(kh)$, on a au premier ordre

$$x_{k+1} = x_k + hF(kh, x_k, u_k)$$

qui est bien une équation de la forme (4.6), avec

$$f(k, x, u) = x + hF(kh, x, u) .$$

De même, le système différentiel peut être muni d'un critère intégral à minimiser, de la forme

$$J = K(x(T)) + \int_0^T l(t, x(t), u(t))\, dt$$

qui peut être approximé au premier ordre par une somme finie (où $T = Nh$)

$$J = K(x_N) + \sum_{k=0}^{N-1} hl(kh, x_k, u_k)$$

qui est bien de la forme (4.7).

Ainsi, la programmation dynamique apparait comme un moyen d'aborder l'optimisation d'un critère intégral pour un système différentiel, un problème connu sous le nom de "commande optimale", ou en Franglais de "contrôle".

L'approximation au premier ordre proposée ci-dessus n'est convenable que si on choisit un pas de temps "assez petit". De même, l'application pratique demandera souvent qu'on discrétise aussi x et u, se ramenant en fait à un problème fini. Et encore, cette discrétisation elle-même demande à être faite avec soin. Enfin, ces discrétisations ne mèneront à un problème faisable en pratique que si les dimensions des espaces d'état et de commandes sont assez petites pour que le nombre de "nœuds" du graphe soit raisonnable.

En fait, ce que nous obtenons ici est *une* discrétisation de l'équation de Hamilton Jacobi Bellman, l'équation aux dérivées partielles équivalente pour ce problème continu à l'équation de Bellman pour le problème à temps discret. Une analyse plus approfondie des schémas numériques nécessiterait beaucoup plus de mathématiques. Nous nous limitons donc à cette aproche naïve.

Mais quelles que soient les limitations de cette méthode, elle reste extrêmement utile dans des cas où elle s'applique. En particulier par le fait qu'elle se prête à prendre en compte toutes sortes de contraintes sur les états admissibles, et des données sans bonnes propriétés mathématiques. (Par exemple, les données peuvent dépendre de fonctions tabulées, etc.)