Recherche Opérationnelle : aspects mathématiques et applications

J. Frédéric Bonnans & Stéphane Gaubert

Septembre 2018

INRIA Saclay - Île-de-France & Centre de Mathématiques Appliquées (CMAP), École Polytechnique, CNRS.
e-mail : Frederic.Bonnans@inria.fr, Stephane.Gaubert@inria.fr

Table des matières

Préambule

La recherche opérationnelle est un ensemble de techniques portant sur la formalisation de problèmes d'organisation, et l'étude de leur résolution par des algorithmes appropriés. Cette discipline, apparue lors de la seconde guerre mondiale [1], s'est diffusée rapidement dans l'ensemble de l'économie. Les années 1950 ont en effet vu se développer la programmation linéaire [2] (Dantzig [Dan63]) et la programmation dynamique (Bellman, [Bel61]). Ces techniques sont allées de pair avec les développements de la théorie des graphes (Berge [Ber70]), de la théorie des jeux (Von Neumann et Morgenstein [vNM44]) et de la programmation non-linéaire et convexe (Arrow, Hurwicz et Uzawa [AHU58]). Le domaine s'est depuis considérablement développé, en interaction avec plusieurs disciplines : mathématiques, informatique, économie, et physique statistique.

Le but de ce cours est d'éclairer les aspects de la recherche opérationnelle relevant des mathématiques appliquées. Il s'agit d'apprendre à modéliser les problèmes de décision qui se posent dans l'industrie (gestion, organisation, investissement,...), afin de reconnaître les problèmes qui peuvent être aujourd'hui résolus. Pour cela, nous présentons quelques grandes familles de méthodes, en mettant l'accent sur les techniques d'optimisation en variables entières et continues, en en particulier, sur la programmation linéaire, qui constitue le socle sur lequel s'appuient la plupart des algorithmes.

Ce cours est constitué de deux parties intimement liées : "modélisation et outils combinatoires", et "outils de programmation convexe et entière", allant ainsi de questions élémentaires de modélisation à des aspects plus avancés.

Modélisation et outils combinatoires En guise de motivation, le premier chapitre donne des exemples de problèmes répertoriés de recherche opérationnelle. Le chapitre 2 fournit quelques résultats de base relatifs aux sous-ensembles convexes de $\mathbb{R}^n$: théorèmes de séparation, représentation des polyèdres en termes de directions et points extrêmes. Afin de relier problèmes continus et discrets, nous donnons quelques résultats relatifs aux matrices totalement unimodulaires. Ce chapitre se poursuit par l'exposé de la théorie de la dualité avec application à la programmation linéaire conique.

Le chapitre 3 est consacré aux problèmes de flots : caractère entier des solutions, dualité entre flot maximum et coupe minimale, problèmes de flot à coût minimum.

Le chapitre 4 introduit au principe de programmation dynamique. On se limite ici au cadre déterministe, qui permet d'aborder de nombreuses applications en combinatoire. On laisse ici de coté l'extension au cas stochastique (processus de décision Markoviens) et aux jeux à deux joueurs. On traite cependant de critères avec paiement moyen, dits "ergodiques", moins classiques, utile dans l'étude de problèmes en temps long.

Après avoir de la sorte fait un tour d'horizon de problèmes spéciaux plus accessibles, car réductible à des problèmes de flots, de programmation dynamique, ou de programmation linéaire non-entière, nous abordons dans le chapitre 5 l'une des principales techniques de résolution des problèmes combinatoires généraux, fondée sur la séparation et l'évaluation (branch and bound). Nous discutons notamment les deux grandes familles

1. Le terme "opérationnel" fait référence, à l'origine de cette discipline (1939-1945), au caractère militaire des applications, comme dans l'expression "théâtre d'opérations". Il a ensuite pris le sens d'"effectif".

2. Le terme "programmation" a pour origine la programmation budgétaire ; lorsque l'emploi optimal des ressources se traduisait en relations linéaires, on a pris l'habitude de parler de programmation linéaire. La programmation est ensuite devenue synonyme (dans ce contexte) d'optimisation.

de relaxation (permettant l'évaluation de minorants), fondées sur la relaxation de contraintes d'intégrité et sur la dualité lagrangienne.

Outils de programmation convexe et entière Le chapitre 6 traite l'algorithme du simplexe, qui permet la résolution numérique de programmes linéaires. Celui-ci est discuté dans le cadre plus général des algorithmes de gradient réduit. On présente la stratégie de plus forte pente (steepest edge), ainsi que le simplexe dual. Malgré le développement des méthodes concurrentes (points intérieurs), l'algorithme du simplexe reste un outil irremplaçable dans la résolution de problèmes de recherche opérationnelle. La question de l'existence d'une règle de pivotage polynomiale, restée ouverte depuis l'invention même de l'algorithme, et qui est reliée à la conjecture de Hirsch sur le diamètre des polyèdres, fait que cet algorithme reste un sujet d'étude actuel.

Le chapitre 7 présente la théorie des coupes d'intégrité. On y explique la construction des coupes de Gomory et de Chvátal, et leur application à des problèmes spécifiques, puis on montre comment construire la clôture entière d'un polyèdre en répétant les coupes de Chvátal.

Le chapitre 8 présente les méthodes de décomposition par génération de colonnes et contraintes, avec application aux problèmes multiflots. Ce chapitre comprend aussi une introduction à la programmation linéaire avec recours, pour les problèmes d'optimisation stochastique.

La dernière partie du cours est consacrée à deux sujets plus avancés. Le chapitre 9 est consacré aux problèmes d'optimisation sous contraintes d'inégalités linéaires matricielles, ou problèmes "SDP". Ceux-ci permettent de traiter de nombreux problèmes issus du contrôle, des statistiques, ou de la mécanique. Ils interviennent aussi dans les relaxations de problèmes combinatoires tels que le problème de la coupe maximale. Le chapitre 10 présente la méthode des points intérieurs, qui est d'une grande efficacité pratique, tout en fournissant des bornes de complexité polynomiale pour les programmes linéaires.

Enfin une annexe donne un exemple d'algorithme glouton, en traitant le calcul d'un arbre couvrant de poids minimum (qui intervient dans l'élaboration de bornes pour le problème du voyageur de commerce).

Il existe plusieurs ouvrages introductifs ou de référence en recherche opérationnelle, principalement en langue anglaise. On recommande notamment la lecture des ouvrages généralistes de J. Cook, W.H. Cunningham, W.R. Pulleybank, A. Schrijver [CCPS98] et de J. Lee [Lee04], ou bien de J. Kleinberg et É. Tardos [KT06] pour une sélection de sujets orientée vers l'algorithmique. On pourra aussi consulter l'ouvrage de M. Gondran et M. Minoux [GM09], en langue française. L'introduction à la programmation linéaire de V. Chvátal [Chv83] est un classique ; on renvoie aussi à l'ouvrage plus récent et parfois plus avancé de J. Matousek et B. Gärtner [MG07]. L.A. Wolsey a donné une introduction à la programmation entière [Wol98], dont la lecture peut être approfondie à l'aide des ouvrages de G. Nemhauser et L.A. Wolsey [NW99], et de Schrijver [Sch86]. L'ouvrage de R.K. Ahuja, T.L Magnanti et J.B. Orlin [AMO93] est canonique en matière d'algorithmes de flots. L'ouvrage de Barvinok [Bar02] comprend une sélection de résultats d'optimisation convexe, avec notamment un traitement introductif de l'optimisation semi-définie et de ses applications en combinatoire. Une référence approfondie à ce sujet est l'ouvrage très récent de B. Gärtner et J. Matousek [GM12]. L'ouvrage d'A. Ben-Tal et A. Nemirovski [BTN01] introduit à la programmation conique et semi-définie, en l'illustrant par de multiples applications. Enfin, l'ouvrage fondamental d'A. Schrijver [Sch03] fait un large état de l'art de l'approche polyédrale en optimisation combinatoire. Nous nous sommes souvent appuyés sur ces différents textes pour rédiger ce cours.

Nous remercions les collègues qui nous ont fait part de commentaires sur ce cours, et aussi ceux qui nous ont signalé des travaux ou fait des remarques qui ont inspiré ou enrichi certains exercices et problèmes ou corrections. Nous remercions spécialement G. Allaire, X. Allamigeon, P. Benchimol, M. Bouhtou, A. Chambolle, É. Gourdin, V. Leclère, L. Massoulié, F. Meunier et B. Rottembourg.

Chapitre 1

Premiers pas en recherche opérationnelle

1.1 Quelques problèmes d'optimisation

En guise de motivation, nous présentons quelques problèmes d'optimisation, de nature presque toujours combinatoire, qui nous serviront de fil conducteur dans la suite. Nous introduirons au fur et à mesure quelques définitions et notions de base, surtout relatives au graphes, permettant de formuler les problèmes.

Exemple 1.1 (Voyageur de commerce) Un voyageur de commerce souhaite effectuer une tournée comprenant n villes, en minimisant la distance parcourue. Plus précisément, désignons les villes par des entiers de 1 à n, et notons c_{ij} le coût pour aller de la ville i à la ville j (le coût peut représenter une distance, un temps, etc.) Une *tournée*, est par définition une suite de villes comprenant exactement une occurrence de chaque ville. On peut la représenter par une permutation circulaire γ de l'ensemble $\{1, \ldots, n\} : \gamma(i)$ indique la ville où aller quand on est dans la ville i. Le *problème du voyageur de commerce* consiste donc à trouver une permutation circulaire γ minimisant le coût total

$$\sum_{1 \leqslant i \leqslant n} c_{i\gamma(i)} \ . \tag{1.1}$$

Ce problème apparaît souvent en pratique, parfois modulo un changement immédiat de vocabulaire (hélicoptère faisant une tournée de plates-formes de forage), parfois de manière déguisée. Considérons par exemple une machine, devant produire de manière cyclique n pièces. Désignons par p_i le temps de traitement de la pièce i, et par t_{ij} le temps nécessaire pour reconditionner la machine, lorsque la pièce j succède à la pièce i. Minimiser le temps de cycle de la machine revient à résoudre un problème de voyageur de commerce dans lequel :

$$c_{ij} = p_i + t_{ij} \ .$$

La difficulté du problème du voyageur de commerce provient de son caractère *combinatoire* : il y a en effet $(n-1)!$ permutations circulaires, donc $(n-1)!$ tournées. Le nombre de tournées, qui croît très rapidement avec n, interdit de parcourir exhaustivement les solutions. Par exemple, pour $n = 25$, il y a plus de 6×10^{23} tournées, un ordinateur examinant 10^9 tournées par seconde (ce qui est optimiste) mettrait plus de 10^7 années pour examiner toutes les solutions.

Remarque 1.2 On parle de problème du voyageur de commerce *symétrique* pour qualifier les cas particuliers où $c_{ij} = c_{ji}$. Si en outre c prend des valeurs positives et vérifie l'inégalité triangulaire $c_{ij} \leqslant c_{ik} + c_{kj}$, on parle de problème de voyageur de commerce *métrique*. •

On peut souvent décrire agréablement un problème de recherche opérationnelle en utilisant la notion de graphe.

Définition 1.3 (Graphe orienté) Un *graphe orienté* est donné par un couple $(\mathcal{N}, \mathcal{A})$: $\mathcal{N}$ est un ensemble dont les éléments sont appelés *nœuds*, et $\mathcal{A}$ est un sous-ensemble de $\mathcal{N} \times \mathcal{N}$, dont les éléments sont appelés *arcs*.

Les arcs (et parfois les nœuds) d'un graphe peuvent être munis de diverses *valuations* : on notera par exemple c_{ij} un *coût* associé à l'arc (i, j), qui pourra représenter la distance de i à j, le temps de parcours de i à j, etc. Un intérêt des graphes est de permettre de parler de *chemin*. Un chemin de *longueur* k est une suite de nœuds, $i_0, \ldots, i_k \in \mathcal{N}$, telle que $(i_0, i_1), \ldots, (i_{k-1}, i_k) \in \mathcal{A}$. On dit que i_0 est son *origine*, que i_k sa *destination*. Un chemin de longueur infinie est une suite infinie de nœuds $i_0, i_1, \ldots \in \mathcal{N}$, telle que $(i_0, i_1), (i_1, i_2), \ldots \in \mathcal{A}$. Toute valuation définie sur les arcs peut être étendue aux chemins de longueur finie, le plus souvent de manière additive. Ainsi, le coût d'un chemin $(i_0, \ldots, i_k)$ vaut par définition $c_{i_0 i_1} + \cdots + c_{i_{k-1} i_k}$. Un chemin tel que $i_0 = i_k$ est appelé *circuit*. Un circuit composé d'un seul arc est une *boucle*. Un chemin $(i_0, \ldots, i_k)$ est dit *élémentaire* si les nœuds $i_0, \ldots, i_k$ sont distincts. Un circuit $(i_0, \ldots, i_k)$ est dit *élémentaire* si les nœuds $i_0, \ldots, i_{k-1}$ sont distincts. On emploiera librement (et sans avoir besoin de le définir) le vocabulaire usuel à propos des chemins, lorsqu'il n'est pas ambigu. Ainsi, nous parlerons de chemin allant de i à j (pour un chemin d'origine i et de destination j), nous dirons que le chemin $(i_0, \ldots, i_k)$ visite les nœuds $i_0, \ldots, i_k$, mais que le circuit $(i_0, \ldots, i_k)$ visite les nœuds $i_0, \ldots, i_{k-1}$, etc. Une tournée, aussi appelée *circuit hamiltonien*, est un circuit visitant chaque nœud une et une seule fois.

Exemple 1.4 (Voyageur de commerce dans un graphe orienté)
Le problème du voyageur de commerce est parfois formulé à partir d'un graphe orienté dont les arcs sont munis d'un coût : il s'agit alors de trouver un circuit hamiltonien de coût minimal, c'est-à-dire, avec les notations précédentes, une permutation circulaire γ des nœuds de $\mathcal{N}$ telle que

$$(i, \gamma(i)) \in \mathcal{A}, \text{ pour tout } i \in \mathcal{N} \ , \tag{1.2}$$

minimisant à nouveau le coût $\sum_{i \in \mathcal{N}} c_{i\gamma(i)}$.

Ce problème se ramène facilement à celui décrit dans l'exemple 1.1. Il suffit en effet de poser $\mathcal{N} = \{1, \ldots, n\}$ et $c_{ij} = +\infty$ lorsque $(i, j) \notin \mathcal{A}$: si une permutation circulaire γ ne vérifie pas l'une des contraintes $(i, \gamma(i)) \in \mathcal{A}$, sont coût sera infini, et donc elle ne contribuera pas à la valeur du problème d'optimisation. On peut même remplacer $+\infty$ par un nombre suffisamment grand K : l'absence de tournée γ vérifiant (1.2) se traduira alors par une valeur élevée de l'infimum du Problème de l'exemple 1.1. De manière précise, pour toute permutation circulaire γ de $\{1, \ldots, n\}$ ne vérifiant pas une des contraintes (1.2), la distance totale parcourue (1.1) vaudra au moins $K + (n-1) \min_{(i,j) \in A} c_{ij}$, alors que la distance totale parcourue vaudra au plus $n \max_{(i,j) \in A} c_{ij}$ si les contraintes (1.2) sont toutes satisfaites. En prenant $K > n \max_{(i,j) \in A} c_{ij} - (n-1) \min_{(i,j) \in A} c_{ij}$, on pourra donc distinguer les deux situations si l'on sait calculer l'infimum du problème de l'exemple 1.1. Ainsi, le problème du voyageur de commerce contient le problème consistant à décider si un graphe admet un circuit hamiltonien.

Remarque 1.5 On distingue plus généralement en recherche opérationnelle *problèmes d'optimisation* et *problèmes de décision*. Un problème de décision consiste à répondre oui ou non à une question. À tout problème d'optimisation peut être associé le problème de décision consistant à déterminer s'il existe une solution pour laquelle le critère est strictement inférieur à une valeur donnée. Ainsi, le problème de décision associé au problème de voyageur de commerce est : "existe-t-il une tournée de coût strictement inférieur à une valeur donnée" ? En particulier, le *problème de faisabilité* (existence d'une solution vérifiant les contraintes) est un cas particulier de problème de décision, correspondant à la valeur $+\infty$. Si l'on sait résoudre le problème d'optimisation, on sait bien-sûr résoudre le problème de décision. Réciproquement, si l'on sait résoudre le problème de décision associé, on peut approcher et parfois même calculer exactement la valeur du problème d'optimisation. Si l'on sait par exemple que le critère prend des valeurs entières comprises entre a et b, il suffira pour cela de résoudre le problème de décision associé de l'ordre de $\log_2(b-a)$ fois, pour des valeurs successives obtenues par dichotomie. •

Exemple 1.6 (Problème d'affectation optimale) Imaginons un ensemble de n individus devant être affectés a un ensemble de n postes. L'individu i assigne un score s_{ij} à chaque poste j, mesurant sa satisfaction s'il est affecté à ce poste. Une *affectation* se représente par une permutation σ de l'ensemble $\{1, \ldots, n\}$: on pose $\sigma(i) = j$ si l'individu i est affecté au poste j. Il s'agit de trouver une affectation maximisant la satisfaction collective. On peut mesurer la satisfaction collective par la somme des satisfactions individuelles :

$$\sum_{1 \leqslant i \leqslant n} s_{i\sigma(i)} \ . \tag{1.3}$$

Le *problème d'affectation optimale* consiste à maximiser cette expression sur l'ensemble des permutations σ de $\{1, \ldots, n\}$.

Remarque 1.7 On notera la similarité entre les critères du problème d'affectation et du problème du voyageur de commerce. La nouveauté est que l'on optimise maintenant ce critère sur l'ensemble de toutes les permutations σ, alors que dans le cas du voyageur de commerce, on optimisait uniquement sur l'ensemble des permutations *circulaires* γ. Nous verrons que ce "petit" changement simplifie considérablement le problème (Application 3.16). •

Remarque 1.8 D'autres mesures de la satisfaction collective sont possibles. Par exemple, si l'on définit la satisfaction de la société comme la satisfaction de l'individu le moins bien servi, on obtient le *problème d'affectation goulot* qui consiste à maximiser le critère :

$$\min_{1 \leqslant i \leqslant n} s_{i\sigma(i)} \ ,$$

toujours sur l'ensemble des permutations σ de $\{1, \ldots, n\}$. •

Exemple 1.9 (Problème de transport) L'un des plus anciens problèmes de recherche opérationnelle est le célèbre *problème des déblais et remblais* de Monge, dons nous donnons ici une version discrète. Considérons n tas de sable, devant servir à combler m trous. Notons $\alpha_i > 0$ la masse du i-ième tas de sable, et $\beta_j > 0$ la masse de sable nécessaire pour combler le j-ième trou. Nous supposerons que la masse des tas de sable permet de combler exactement l'ensemble des trous :

$$\alpha_1 + \cdots + \alpha_n = \beta_1 + \cdots + \beta_m \ .$$

Notons x_{ij} la quantité de sable transportée de i à j. La matrice $x = (x_{ij})$, appelée *plan de transport*, satisfait les contraintes suivantes :

$$0 \leqslant x_{ij} \tag{1.4a}$$

$$\alpha_i = \sum_{1 \leqslant j \leqslant m} x_{ij}, \qquad \forall 1 \leqslant j \leqslant n \tag{1.4b}$$

$$\beta_j = \sum_{1 \leqslant i \leqslant n} x_{ij}, \qquad \forall 1 \leqslant j \leqslant m \tag{1.4c}$$

On note c_{ij} le coût de transport, par unité de masse, du tas i au trou j : par exemple, c_{ij} pourra être la distance euclidienne du tas i au trou j. Il s'agit de trouver une matrice $x = (x_{ij})$ vérifiant (1.4) et minimisant le coût total de transport :

$$\sum_{\substack{1 \leqslant i \leqslant n \\ 1 \leqslant j \leqslant m}} c_{ij} x_{ij} \ .$$

Dans le cas du sable, il est naturel de supposer que les données α_i, β_j, ainsi que les masses transportées x_{ij} sont des réels. Cependant, on pourrait transporter des colis ou des containers. Dans ce cas, les tas de sable représenteront des fournisseurs, et les trous des clients. Les α_i et β_j seront alors des entiers positifs, et l'on exigera que les quantités transportées x_{ij} soient entières. Nous montrerons que cette restriction ne complique pas le problème : dès que les α_i et les β_j sont entiers, l'optimum est atteint par un plan de transport $x = (x_{ij})$ entier (Corollaire 3.14).

Remarque 1.10 Si l'on se limite aux solutions entières, le cas particulier du problème de transport de masse où

$$n = m \text{ et } \alpha_i = \beta_j = 1 \; \forall 1 \leqslant i, j \leqslant n \ ,$$

est équivalent au problème d'affectation optimale. En effet, pour ces valeurs particulières de α et β, on vérifie que les solutions $x = (x_{ij})$ à coordonnées entières de (1.4) sont exactement les matrices de permutations. Autrement dit, $x = (x_{ij}) \in \mathbb{Z}^{n \times n}$ vérifie (1.4) si et seulement si il existe une permutation σ de l'ensemble $\{1, \ldots, n\}$ telle que $x_{ij} = 1$ si $j = \sigma(i)$ et $x_{ij} = 0$ sinon. On a alors $\sum_{\substack{1 \leqslant i \leqslant n \\ 1 \leqslant j \leqslant n}} c_{ij} x_{ij} = \sum_{1 \leqslant i \leqslant n} c_{i\sigma(i)}$. En posant $s_{ij} = -c_{ij}$, on voit que trouver un plan de transport entier optimal revient à trouver une permutation maximisant (1.3). •

Exemple 1.11 (Identification de matrice de trafic) Soient n émetteurs et m récepteurs, qui peuvent être des points d'un réseau télécom ou bien des nœuds du réseau routier. On note x_{ij} la quantité transmise de l'émetteur i au récepteur j. On suppose que pour tout $1 \leqslant i \leqslant n$ on mesure le trafic global α_i émis en i, et que, pour tout $1 \leqslant j \leqslant m$, on mesure le trafic global β_j reçu en j. La matrice de trafic $x = (x_{ij})$ vérifie donc précisément les contraintes (1.4). On s'intéresse à la situation où la matrice de trafic $x = (x_{ij})$ ne peut être mesurée, mais où l'on voudrait l'approcher en exploitant une estimation a priori de celle-ci. Imaginons par exemple, dans le cas du transport routier, que l'on ait obtenu la matrice de trafic $a = (a_{ij})$ de l'an dernier en faisant un sondage auprès des conducteurs : on voudrait actualiser la matrice a en une matrice x à partir des nouveaux bilans α_i et β_j, sans faire de nouveau sondage. Ce problème est a priori mal posé, car la matrice de trafic $x = (x_{ij})$ vérifie les $n + m$ équations données dans (1.4), alors qu'il y a nm inconnues. Une manière de lever cette sous-détermination est de chercher une matrice de trafic $x = (x_{ij})$ proche en un sens convenable de la matrice de l'année précédente. Afin de mesurer cette proximité, introduisons la fonction *entropie* :

$$\phi(r) = r \log r - r \ .$$

Cette fonction est bien adaptée à des problèmes en variables positives : on a $\phi'(r) = \log r$, donc ϕ est strictement convexe, minimale en $r = 1$, et sa dérivée prend toutes les valeurs de $-\infty$ à $+\infty$. Une estimation raisonnable de la matrice de trafic de cette année est fournie par la matrice x satisfiant (1.4) et minimisant le critère :

$$\sum_{\substack{1 \leqslant i \leqslant n \\ 1 \leqslant j \leqslant m}} \phi(\frac{x_{ij}}{a_{ij}}) a_{ij} \ . \tag{1.5}$$

Le critère a été construit pour qu'il soit minimum lorsque $x = a$. Lorsque $a_{ij} = 0$, on convient que $\phi(x_{ij}/a_{ij})a_{ij} = +\infty$ si $x_{ij} > 0$, et $\phi(x_{ij}/a_{ij})a_{ij} = 0$ si $x_{ij} = 0$, de sorte que toute solution admissible de coût fini satisfait :

$$a_{ij} = 0 \implies x_{ij} = 0 \ .$$

On a donc affaire à une version *non-linéaire* du problème de transport, dans lequel on a remplacé le coût de transport linéaire $\sum_{ij} c_{ij} x_{ij}$ par le coût non-linéaire, mais convexe, (1.5). Nous verrons plus loin que cette non-linéarité conduit à perdre le caractère combinatoire du problème, en particulier, lorsque les α_i et β_j sont entiers, l'optimum est maintenant atteint par une matrice de trafic x qui n'est pas nécessairement entière.

Exemple 1.12 (Problème du sac à dos) Considérons n objets pouvant rentrer dans un sac. Notons $u_i > 0$ l'utilité de l'objet i, $p_i > 0$ son poids, et M un poids maximum que l'on est disposé à porter. Le *problème du sac à dos* consiste à trouver un sous-ensemble I de $\{1, \ldots, n\}$ tel que le poids emporté $\sum_{i \in I} p_i$ soit au plus M, maximisant l'utilité totale $\sum_{i \in I} u_i$.

Exemple 1.13 (Plus court chemin, ou chemin de coût minimum) Le problème très classique du *chemin de coût minimum*, aussi appelé problème du *plus court chemin*, s'énonce ainsi. Soit $(\mathcal{N}, \mathcal{A})$ un graphe orienté : un nœud $i \in \mathcal{N}$ représente un ville, et un arc $(i, j) \in \mathcal{A}$ représente une route de la ville i à la ville j. On note c_{ij} le coût pour aller de i à j en empruntant cette route. On se donne une origine $o \in \mathcal{N}$ et une

destination $d \in \mathcal{N}$. Il s'agit de trouver un chemin $i_0, \ldots, i_k$ de o à d (le nombre de routes empruntées, k, étant quelconque), minimisant le coût total :

$$c_{i_0 i_1} + \cdots + c_{i_{k-1} i_k} \ . \tag{1.6}$$

Lorsqu'il y a un circuit γ de coût strictement négatif, ce problème dégénère. Supposons en effet qu'il existe un chemin π de o à un nœud quelconque de γ, ainsi qu'un chemin π' de ce nœud à d. Le chemin π'' obtenu en composant le chemin π avec un nombre arbitraire de copies du circuit γ, puis avec π', est encore un chemin de o à d, et son coût prend des valeurs négatives arbitrairement grandes. Dans ce cas, l'infimum du coût (1.6) vaudra $-\infty$. Ainsi, le problème du chemin de coût minimum est relié au problème de l'existence d'un *circuit de poids négatif* dans un graphe. Nous verrons d'ailleurs que le problème du chemin de coût minimum se simplifie considérablement lorsque les coûts des arcs prennent des valeurs positives ou nulles, ce qui assure l'existence d'un plus court chemin.

Remarque 1.14 (Plus court chemin avec contraintes) On peut compliquer le problème du plus court chemin en rajoutant des contraintes. Par exemple, si pour chaque arc (i, j), on se donne un temps de parcours t_{ij}, et une durée maximale T, on peut chercher un chemin minimisant le coût total (1.6) sous la contrainte

$$t_{i_0 i_1} + \cdots + t_{i_{k-1} i_k} \leqslant T \ . \tag{1.7}$$

Les problèmes de plus court chemin avec une ou plusieurs contraintes sont fréquents en pratique (penser à l'achat d'un billet d'avion avec correspondances : on peut chercher à minimiser le prix en tenant compte de contraintes de temps, de robustesse, etc.). •

Pour présenter l'exemple suivant, il nous faut introduire quelques notions relatives aux graphes *non-orientés*.

Définition 1.15 (Graphe non-orienté) Un *graphe non-orienté* est un couple $(\mathcal{V}, \mathcal{E})$, où $\mathcal{V}$ est un ensemble dont les éléments sont appelés sommets, et $\mathcal{E}$ est un sous-ensemble de parties de $\mathcal{V}$ à un ou deux éléments, dont les éléments sont appelés *arêtes*[1].

Une arête $\{i, j\}$ est représentée par un trait non-orienté reliant i et j. Comme les arcs, les arêtes peuvent être munies de diverses valuations : on notera $c_{\{i,j\}}$ la valuation de l'arête $\{i, j\}$. On simplifiera souvent $c_{\{i,j\}}$ en c_{ij}, en prenant garde que dans le cas non-orienté, on a par définition $c_{ij} = c_{ji}$, car $\{i, j\}$ et $\{j, i\}$ désignent le même ensemble. La plupart des notions et problèmes définis dans le cas orienté admettent le plus souvent des variantes évidentes dans le cas non-orienté. Ainsi, un *chemin (non-orienté)* est une suite $i_0, \ldots, i_k$ telles que $\{i_0, i_1\}, \ldots, \{i_{k-1}, i_k\} \in \mathcal{E}$. On dit que i_0 et i_k sont les *extrémités* du chemin. Signalons quelques différences de vocabulaire en passant du cas orienté au cas non-orienté : on parle de sommet au lieu de nœud, d'arête au lieu d'arcs, de cycle au lieu de circuit. Les notions suivantes sont spécifiques au cas non-orienté.

Définition 1.16 (Connexité) Un graphe non-orienté est dit *connexe* si deux sommets quelconques peuvent être reliés par un chemin (non-orienté).

Définition 1.17 (Arbre) Un *arbre* est un graphe non-orienté connexe sans cycle.

Exemple 1.18 (Arbre couvrant de coût minimum) Soit un graphe non-orienté $(\mathcal{V}, \mathcal{E})$. On appelle *arbre couvrant* un arbre ayant pour ensemble de sommets $\mathcal{V}$ et composé d'arêtes de $\mathcal{E}$. Attribuons maintenant un coût $c_{\{i,j\}}$ à chaque arête $\{i, j\} \in \mathcal{E}$. Il s'agit de trouver un arbre $(\mathcal{V}, \mathcal{T})$, avec $\mathcal{T} \subset \mathcal{E}$, minimisant le coût :

$$\sum_{\{i,j\} \in \mathcal{T}} c_{\{i,j\}} \ .$$

Ce problème apparaît par exemple lorsque l'on veut relier certains points par une liaison électrique en minimisant la longueur de fil.

1. Les lettres $\mathcal{V}$ et $\mathcal{E}$ renvoient aux mots anglais "vertices" et "edges", respectivement.

Remarque 1.19 Les problèmes réels de câblage conduisent souvent à un problème plus général et plus difficile, qui est celui de l'*arbre de Steiner*, dans lequel on se donne en outre un sous-ensemble de sommets dits *obligatoires* (les autres sommets sont dits *facultatifs*). Il s'agit de trouver un arbre de coût minimum couvrant tous les sommets obligatoires mais pouvant éventuellement comprendre des sommets facultatifs, c'est-à-dire, avec les notations précédentes, un arbre $(\mathcal{V}', \mathcal{T}')$ tel que $\mathcal{V}' \subset \mathcal{V}$, $\mathcal{T}' \subset \mathcal{T}$, tel que pour tout sommet obligatoire i, on ait $i \in \mathcal{V}'$, et qui soit de coût minimum. •

Définition 1.20 (Coupe) On appelle *coupe* d'un graphe orienté $(\mathcal{N}, \mathcal{A})$, associée à une partition $I^+ \cup I^- = \mathcal{N}$, l'ensemble des arcs reliant un nœud de I^- à un nœud de I^+. On définit de manière analogue la notion de coupe dans le cas non-orienté.

Exemple 1.21 (Coupe minimale séparant deux points) Soit $(\mathcal{N}, \mathcal{A})$ un graphe orienté, dans lequel on distingue deux nœuds, s et p (pour "source" et "puits"). Nous dirons que la coupe associée à une bipartition $\mathcal{N} = I^+ \cup I^-$ de l'ensemble des nœuds *sépare* s de p si $s \in I^-$ et $p \in I^+$. Supposons chaque arc (i, j) muni d'une valuation u_{ij}, appelée *capacité* de l'arc (i, j), et définissons la *capacité* de la coupe :

$$u(I^-, I^+) := \sum_{i \in I^-, j \in I^+} u_{ij} \ .$$

Un problème de base consiste à trouver une coupe séparant s de p minimisant la capacité $u(I^+, I^-)$. Dans le cas le plus simple où les capacités valent toutes 1, ce problème revient à calculer le nombre minimal d'arcs que l'on doit supprimer pour qu'il n'y ait plus de chemin de s à p. C'est donc une mesure de la connectivité ou de la robustesse du réseau (nombre de défaillances admissible). Plus généralement, la capacité u_{ij} représentera un débit maximum admissible sur l'arc en régime stationnaire : la capacité minimale d'une coupe s'interprète comme la capacité du *goulot d'étranglement* dans un problème de transport de s à p. Nous verrons que lorsque les valuations u_{ij} sont positives, le problème de la coupe minimale se résout facilement, car il intervient dans la formulation duale des problèmes de flots (Théorème de Ford-Fulkerson 3.18).

Exemple 1.22 (Coupe maximale) Soit $(\mathcal{V}, \mathcal{E})$ un graphe non-orienté sans boucles. Munissons chaque arête $\{i, j\}$ d'un coût $c_{\{i,j\}}$, et cherchons un vecteur $\sigma = (\sigma_i) \in \{\pm 1\}^{\mathcal{V}}$ minimisant le critère :

$$E(\sigma) := \sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} \sigma_i \sigma_j \ . \tag{1.8}$$

Ce problème apparaît en physique statistique, dans le cadre du modèle d'Ising : σ représente alors une configuration de spins, qui ne prennent que les valeurs ± 1. L'expression (1.8) exprime l'énergie totale du système est somme des énergies d'interactions entre voisins. Une configuration d'énergie minimale est appelée état fondamental. Les coefficients d'interaction $c_{\{i,j\}}$ peuvent dépendre de l'arête $\{i, j\}$ lorsque le milieu présente des impuretés (cas des "verres de spins"). Signalons aussi que le même problème de minimisation apparaît en traitement d'image : les valeurs $\sigma_i = \pm 1$ représentent alors les valeurs noir ou blanc des différents pixels d'une image à reconstruire.

Étant donnée une configuration σ, définissons la partition

$$\mathcal{V} = I^+ \cup I^- \ , \text{ où } \ I^\pm = \{i \in \mathcal{V} \mid \sigma_i = \pm 1\} \ .$$

On a

$$E(\sigma) = \sum_{\{i,j\} \in \mathcal{E}} c_{\{i,j\}} - 2 \sum_{i \in I^-,\ j \in I^+} c_{\{i,j\}} \ .$$

Autrement dit, l'énergie du système est, à une constante additive et à un facteur 2 près, l'opposé de l'énergie de "l'interface" :

$$c(I^-, I^+) := \sum_{i \in I^-,\ j \in I^+} c_{\{i,j\}} \ ,$$

et donc, minimiser $E(\sigma)$ revient à maximiser $c(I^-, I^+)$ sur l'ensemble des partitions $I^+ \cup I^- = \mathcal{V}$. Nous reconnaissons dans $c(I^-, I^+)$ la version non-orienté de la capacité de la *coupe* associée à la partition $I^+ \cup I^- = \mathcal{V}$. Le problème consistant à calculer une coupe de capacité maximale est difficile si les c_{ij} sont positifs ou nuls, et a fortiori s'ils ont un signe quelconque (si les coefficients c_{ij} sont négatifs ou nuls, i.e., dans le cas "ferromagnétique", l'infimum de $E(\sigma)$ est trivialement obtenu pour $\sigma \equiv 1$). Ce problème peut être abordé en termes d'optimisation sur le cône des matrices symétriques positives (programmation "SDP", Chapitre 9).

Remarque 1.23 Le terme d'interaction avec les plus proches voisin est souvent augmenté d'un terme d'interaction avec un champ magnétique extérieur, on obtient alors une énergie de la forme

$$E'(\sigma) := \sum_{\{i,j\}\in\mathcal{E}} c_{\{i,j\}}\sigma_i\sigma_j + \sum_{i\in\mathcal{V}} b_i\sigma_i \ . \tag{1.9}$$

Le problème consistant à minimiser une énergie de la forme $E'(\sigma)$ au lieu de $E(\sigma)$ n'est pas plus général. En homogénéisant la formulation, c'est à dire, en rajoutant un sommet à $\mathcal{V}$, noté 0, et en rajoutant une variable de spin correspondante, $\sigma_0 = 1$, on peut en effet écrire

$$E'(\sigma) := \sum_{\{i,j\}\in\mathcal{E}} c_{\{i,j\}}\sigma_i\sigma_j + \sum_{i\in\mathcal{V}} b_i\sigma_i\sigma_0 \ ,$$

ce qui correspond à une forme quadratique de type $E(\sigma)$, dans un graphe augmenté d'arêtes $\{i, 0\}$ pour $i \in \mathcal{V}$. Comme $E(\sigma) = E(-\sigma)$, on ne change rien au problème consistant à minimiser E si l'on rajoute la contrainte que l'une des coordonnées de σ, par exemple σ_0, vaut 1. Le problème consistant à maximiser $E'(\sigma)$, devient facile lorsque les $c_{\{i,j\}}$ sont négatifs ou nuls, à cause de l'interprétation en termes de capacités et de flots mentionnée dans l'exemple 1.21. En renvoie à ce sujet à l'exercice 3.25, qui fournit une application au traitement d'image.

•

1.2 Rudiments de complexité

Nous récapitulons dans le tableau suivant les problèmes déjà évoqués, en portant une indication de difficulté, et en donnant une liste non-limitative de techniques mathématiques traitées dans la suite du cours.

Problème	difficulté	outils
Voyageur de commerce	NP-dur	coupes, sép.-évaluation
Affectation optimale	(fortement) polynomial	flots (alg. Hongrois)
Transport	(fortement) polynomial	flots
Matrice de trafic	convexe (facile)	optimisation convexe
Sac à dos	NP-dur : pseudo-pol.	prog. dynamique
Plus court chemin	(fortement) polynomial	prog. dynamique
Plus court chemin contraint	NP-dur : pseudo-pol.	prog. dynamique
Arbre couvrant de coût min.	(fortement) polynomial	alg. glouton, matroïdes
Arbre de Steiner	NP-dur	séparation et évaluation
Coupe min. (capacités $\geqslant 0$)	(fortement) polynomial	flots
Coupe maximale	NP-dur	SDP

Afin d'interpréter ce tableau, il convient de dire quelques mots des notions de *complexité*. Celles-ci sont très utiles pour hiérarchiser la difficulté des différents problèmes d'optimisation (et ainsi éviter les erreurs d'appréciation). Ces notions relèvent de l'informatique, et nous nous limitons ici à un résumé non technique, en renvoyant au cours [CHS01] ou aux ouvrages classiques [GJ79, Pap95, GLS93] pour des définitions précises, ainsi qu'à [AB09] pour une monographie avancée récente.

On dit qu'un algorithme est *polynomial* si son temps d'exécution peut être borné par une fonction polynomiale de la taille des données. On dit qu'un algorithme est *linéaire* si son temps d'exécution peut être borné par une fonction affine de la taille des données.

Les notions de "taille des données" et de "temps de calcul" présupposent un modèle de calcul. En l'absence d'indication spécifique, on suppose le modèle de Turing ou "bit model". Dans celui-ci, la taille des données est mesurée par le nombre de bits d'espace mémoire que celles-ci occupent. Par exemple, pour le problème du voyageur de commerce, la donnée pourra être une matrice $n \times n$ dont les coefficients sont des entiers représentant les distances entre les différentes villes (et dont la diagonale est nulle). Si l'on code chaque entier sur K bits, l'espace mémoire occupé sera donc essentiellement $K(n^2 - n)$. On prendra garde que dans le modèle de Turing (contrairement à un certain usage en analyse numérique), on suppose que les données sont des entiers codés de manière exacte. Lorsque l'on a une borne sur le temps d'exécution dépendant de manière polynomiale de la taille de l'entrée mesurée en supposant que les entiers du problème sont codés en base unaire (i.e., que l'entier n, codé par n bâtonnets, occupe un espace mémoire d'ordre n), on parle d'algorithme *pseudo-polynomial*.

Illustrons ces notions par quelques exemples plus ou moins faciles. 1) Les opérations arithmétiques usuelles sur les entiers (somme, produit) peuvent être effectuées en temps polynomial. 2) l'entier 2^{2^n} ne peut être calculé en temps polynômial (le résultat occupe 2^n bits et il faut au moins une unité de temps pour écrire un bit). 3) Un exemple important d'algorithme pseudo-polynomial mais non polynomial est la méthode du crible pour trouver un facteur premier d'un entier n : on divise n par les entiers successifs inférieurs ou égaux à $\sqrt{n}$. Comme l'entier n occupe seulement une place $\log_2 n$ en mémoire, le nombre de tests étant exponentiel en l'espace mémoire, l'algorithme du crible n'est pas polynomial, mais il est cependant pseudo-polynomial (parce qu'un test de divisibilité peut se faire en temps polynomial). 4) Le déterminant d'une matrice d'entiers peut se calculer de manière polynomiale. Ceci peut sembler très intuitif si l'on note que l'élimination de Gauss s'effectue en $O(n^3)$ opérations arithmétiques, et que l'on sait bien borner le déterminant en fonction des données, la valeur absolue du déterminant d'une matrice étant bornée par le produit des normes euclidiennes des colonnes (théorème de Hadamard). Il faut cependant travailler un peu plus pour prouver la polynomialité du calcul du déterminant, car l'algorithme de Gauss produit des rationnels, avec des numérateurs et dénominateurs dont les longueurs peuvent croître en cours d'algorithme. Une variante de l'algorithme de Gauss, l'algorithme de Bareiss, qui n'effectue que des divisions exactes, conduit à un algorithme polynomial, et même *fortement polynomial*.

Cette dernière notion renvoie à un modèle de calcul plus fin que le modèle de Turing, dit modèle arithmétique, dans lequel on tient compte du nombre d'opérations arithmétiques. Un algorithme est dit *fortement polynomial* [GLS93] si le nombre d'opérations arithmétiques qu'il effectue est borné par un polynôme en le nombre d'entiers de l'entrée du problème, et si ces opérations arithmétiques ne sont appliquées qu'à des entiers dont la taille est bornée polynomialement en la taille de l'entrée du problème. Ainsi, le nombre d'opérations arithmétiques d'un algorithme fortement polynomial ne dépend pas du "nombre de chiffres significatifs" des entiers qui constituent l'entrée. Un exemple de problème fortement polynomial est le problème du plus court chemin (traitable par programmation dynamique). L'existence d'un algorithme fortement polynomial pour résoudre un programme linéaire est un problème ouvert majeur (alors que des algorithmes polynomiaux, comme les points intérieurs, sont connus).

Dans le modèle de Turing, la notion de temps d'exécution repose sur une abstraction d'un ordinateur usuel, appelée machine de Turing déterministe, ayant un seul processeur, une mémoire interne finie (correspondant à ce qu'on appelle aujourd'hui la mémoire vive), et une mémoire externe (bande) potentiellement infinie. Le temps d'exécution est défini comme le nombre d'opérations élémentaires. Il existe des variantes des machines de Turing déterministes : le temps d'exécution d'un algorithme est sensible à la variante utilisée, mais son caractère polynomial ou non n'en dépend pas. La "thèse de Cobham et d'Edmonds" [Cob65, Edm65] est que les problèmes "faciles" sont ceux pour lesquels il existe un algorithme tournant en temps polynomial. Cette thèse, interprétée avec quelques précautions (si le polynôme a un degré élevé, ou des coefficients astronomiques, la borne polynomiale n'a d'intérêt que théorique), s'est avérée remarquablement fructueuse, car elle rend bien compte de la pratique des problèmes combinatoires.

On note P la classe des problèmes de décision polynomiaux. Une classe semble-t-il beaucoup plus générale est la classe NP, formée des problèmes pour lesquels la validité d'une solution peut être vérifiée en temps polynômial. Par exemple, le problème appelé Circuit Hamiltonien qui consiste à décider s'il existe un circuit visitant chaque sommet d'un graphe une et une seule fois, est un problème dans NP, car si l'on a un circuit quelconque, vérifier s'il est Hamiltonien peut se faire en temps polynomial (et même linéaire, il suffit de

compter le nombre de visites en chaque sommet). Si la notion de vérifiabilité est intuitive, sa formalisation mathématique est non-triviale, elle repose sur une généralisation non-déterministe des machines de Turing (le NP renvoie à "temps polynomial sur une machine de Turing non-déterministe").

La classe NP comprend un nombre considérable de problèmes de décision réputés difficiles, aussi bien théoriques que pratiques : satisfaire un ensemble de contraintes booléennes (Sat), savoir si l'on peut colorier un graphe avec 3 couleurs (3-Coloriage), etc. Un problème *NP-dur* est un problème qui est au moins aussi difficile que tous les problèmes de NP, ce qui entraîne que si l'on savait résoudre ce problème en temps polynomial, on saurait résoudre tous les problèmes de NP en temps polynomial. Les problèmes NP-complets sont des problèmes de décision qui sont au moins aussi difficiles que tous les problèmes de NP tout en étant dans NP, autrement dit, ce sont les problèmes de NP de difficulté maximale. L'existence de tels problèmes est un théorème dû à Cook et Levin [Coo71, Lev73]. Par exemple, Sat, Circuit Hamiltonien, et 3-Coloriage, sont des problèmes NP-complets. De nombreux problèmes d'optimisation combinatoire sont NP-difficiles : par exemple le problème de voyageur de commerce, qui est un problème d'optimisation, est NP-difficile, parce que le problème de décision associé, Circuit Hamiltonien, est NP-complet. Un problème ouvert [Coo00] est de montrer que P$\neq$ NP, c'est-à-dire qu'il y a des problèmes de NP qui ne peuvent être résolus en temps polynomial sur une machine de Turing déterministe. Prévenons ici le lecteur que la théorie de la complexité présente des subtilités qui ne peuvent se comprendre sans rentrer dans le détail technique des définitions. En particulier, la formalisation de "au moins aussi difficile que" repose sur une notion de réduction, dont il existe deux variantes, la réduction au sens de Turing (un problème A se réduit à un problème B si on peut écrire un programme résolvant le problème A en appelant plusieurs fois si nécessaire un programme résolvant le problème B, un tel programme étant appelé oracle) est en général utilisée pour définir les problèmes NP-durs alors qu'une réduction plus fine (réduction de Karp, qui n'autorise à appeler l'oracle qu'une seule fois, et en dernier lieu) est utilisée pour définir les problèmes NP-complets. Nous renvoyons encore une fois à [GJ79, Pap95, GLS93, AB09].

Chapitre 2

Convexité, polyédralité et dualité

Ce chapitre fournit quelques résultats de base d'analyse convexe. Les problèmes d'optimisation convexe (et notamment, les problèmes de programmation linéaire) sont en effet très bien résolus. On les rencontre fréquemment en recherche opérationnelle, soit de manière directe, soit comme problèmes auxiliaires dans l'étude de problèmes plus difficiles, ou bien comme cas spéciaux remarquables. Les résultats relatifs aux polyèdres, ainsi qu'à la théorie de la dualité en optimisation, rappelés ici (Sections 2.2–2.4), sont particulièrement utiles. Les résultats plus avancés (calcul sous-différentiel, Section 2.5), peu utilisés dans les chapitres suivants, peuvent être sautés en première lecture.

2.1 Séparation d'ensembles convexes en dimension finie

Définition 2.1 (Séparation des convexes) On dit qu'une forme linéaire q sur $\mathbb{R}^n$ *sépare* (au sens large) deux parties C_1 et C_2 de $\mathbb{R}^n$ si

$$q \neq 0 \quad \text{et} \quad q \cdot x_1 \leqslant q \cdot x_2, \quad \text{pour tout} \quad x_1 \in C_1,\ x_2 \in C_2. \tag{2.1}$$

Dans cette définition, et dans la suite, nous faisons l'abus de langage consistant à identifier le vecteur $q \in \mathbb{R}^n$ et la forme linéaire $x \mapsto q \cdot x$ qu'il représente.

Lemme 2.2 *Soit une partie C convexe de $\mathbb{R}^n$. Alors $y \in \operatorname{int} C$ ssi il n'existe pas de forme linéaire séparant y de C.*

Démonstration. a) Montrons que si $y \in \operatorname{int} C$, il n'existe pas de forme linéaire séparant y de C. Soit $\rho > 0$ tel que $B(y, \rho) \subset C$. S'il existe une forme linéaire q séparant y de C, posons $\varepsilon := \rho/\|q\|$. Alors $y - \varepsilon q \in C$, et donc avec (2.1), $0 \geqslant \varepsilon\|q\|^2$ ce qui donne la contradiction recherchée.
b) Soit $y \notin \bar{C}$ (fermeture de C). Notons z le projeté de y sur le convexe fermé $\bar{C}$, c'est-à-dire l'unique point de $\bar{C}$ minimisant la distance euclidienne à y, qui existe en vertu d'un résultat classique d'optimisation (la fonction $x \mapsto \|x - y^k\|^2$ est continue et tend vers l'infini quand $x \to \infty$: elle atteint donc son minimum en un point de $\bar{C}$, et ce point de minimum est unique car cette fonction est strictement convexe). Posons $q := z - y$. Puisque $y \notin \bar{C}$, on a $q \neq 0$. Si $x \in \bar{C}$ et $\alpha \in]0, 1]$, on a $z + \alpha(x - z) \in \bar{C}$, et donc

$$0 \leqslant \lim_{\alpha \downarrow 0} \frac{\|z + \alpha(x - z) - y\|^2 - \|z - y\|^2}{2\alpha} = q \cdot (x - z). \tag{2.2}$$

Or $q \cdot (z - y) = \|q\|^2 > 0$, donc $q \cdot (x - y) \geqslant 0$ pour tout $x \in \bar{C}$. Ceci prouve que la forme linéaire non nulle q sépare y de C.
c) Finalement, soit $y \in \partial C$; il faut construire une forme linéaire séparant y de C. Il existe une suite $y^k \to y$, avec $y^k \notin \bar{C}$ pour tout k. D'après la discussion du cas b), il existe une forme linéaire non nulle q_k séparant y^k de C. Divisant q^k par $\|q^k\|$, on se ramène au cas où q_k est de norme 1. Extrayant une sous suite si nécessaire,

on peut supposer que $q^k \to \bar{q} \neq 0$. Passant, pour $x \in C$ fixé, à la limite dans l'inégalité $q^k \cdot (y^k - x) \leqslant 0$, on obtient la relation désirée. ■

Corollaire 2.3 *Soient C_1 et C_2 deux parties convexes de $\mathbb{R}^n$, d'intersection vide. Alors il existe une forme linéaire q séparant C_1 et C_2.*

Démonstration. L'ensemble $C := C_1 - C_2$ (ensemble des différences d'éléments de C_1 et C_2) est convexe et ne contient pas 0. Le lemme 2.2 assure l'existence d'une forme linéaire $q \neq 0$ séparant 0 de C, soit $0 \leqslant q \cdot (x^1 - x^2)$, pour tout $x^1 \in C_1$ et $x^2 \in C_2$, ce qui est la propriété recherchée. ■

Corollaire 2.4 (Séparation stricte) *Soient C un convexe fermé de $\mathbb{R}^n$, et $\bar{x} \in \mathbb{R}^n \setminus C$. Alors il existe une forme linéaire q séparant strictement $\bar{x}$ et C, au sens où*

$$q \cdot \bar{x} < \inf\{q \cdot x;\ x \in C\}. \tag{2.3}$$

Démonstration. Le complémentaire de C étant ouvert, il existe $\varepsilon > 0$ tel que $C_1 := B(\bar{x}, \varepsilon)$ (boule ouverte de centre $\bar{x}$ et rayon ε, pour la norme euclidienne) n'intersecte pas C. On applique alors le corollaire 2.3 avec $C_2 = C$. Comme $\sup\{q \cdot x;\ x \in C_1\} = q \cdot \bar{x} + \varepsilon\|q\|$, il vient $q \cdot \bar{x} \leqslant \inf\{q \cdot x;\ x \in C\} - \varepsilon\|q\|$. ■

Remarque 2.5 On peut interpréter le Lemme 2.2 en termes d'hyperplan d'appui. Soient en effet $q \in \mathbb{R}^n$ et $\gamma \in \mathbb{R}$. On dit que l'hyperplan affine $H = \{y \in \mathbb{R}^n \mid q \cdot y = \gamma\}$ est un *hyperplan d'appui* du convexe C au point $x \in C$ si $\gamma = q \cdot x \leqslant q \cdot y$, pour tout $y \in C$. Le Lemme 2.2 montre qu'un sous-ensemble convexe de $\mathbb{R}^n$ admet un hyperplan d'appui en tout point frontière. •

Définition 2.6 (Combinaison convexe) On dit que $x \in \mathbb{R}^n$ est *combinaison convexe* ou *barycentres* des points $x^1, \ldots, x^p$ de $\mathbb{R}^n$ s'il existe des scalaires $\alpha_1, \ldots, \alpha_p$ positifs et de somme 1 tels que $x = \alpha_1 x^1 + \cdots + \alpha_p x^p$.

Définition 2.7 (Enveloppe, fermeture convexe) Soit X une partie de $\mathbb{R}^n$. On appelle *enveloppe convexe* de X le plus petit ensemble convexe de $\mathbb{R}^n$ contenant X, noté $\mathrm{conv}(X)$. Ce n'est rien d'autre que l'ensemble des combinaisons convexes de points de X. On note $\overline{\mathrm{conv}}(X)$ la fermeture de l'enveloppe convexe de X, appelée aussi *fermeture convexe* de X.

Définition 2.8 (Point extrême) On dit qu'un point x d'un convexe C est *extrême* si $x = \alpha y + \beta z$, avec $y, z \in C$, $\alpha > 0$, $\beta > 0$, et $\alpha + \beta = 1$, entraîne $x = y = z$ (autrement dit, x ne peut s'écrire comme combinaison convexe non triviale de deux points distincts de C).

Nous utiliserons l'observation suivante.

Lemme 2.9 *Si H est un hyperplan d'appui d'un convexe $C \subset \mathbb{R}^n$, alors tout point extrême de $H \cap C$ est point extrême de C.*

Démonstration. Soit $H = \{y \in \mathbb{R}^n \mid c \cdot y = \gamma\}$ avec $c \in \mathbb{R}^n$, $c \neq 0$, et $\gamma \in \mathbb{R}$, un hyperplan d'appui de C. Si $x = \alpha y + \beta z$, avec $y, z \in C$, $\alpha > 0$, $\beta > 0$, $\alpha + \beta = 1$, et si $x \in C \cap H$, il vient $\gamma = c \cdot x = \alpha c \cdot y + \beta c \cdot z$, et comme $\gamma \leqslant c \cdot y$ et $\gamma \leqslant c \cdot z$, on a nécessairement $\gamma = c \cdot y = c \cdot z$, donc $y, z \in C \cap H$. Si l'on suppose que x est un point extrême de $C \cap H$, il vient donc $x = y = z$, ce qui montre que x est un point extrême de C. ■

Définition 2.10 (Dimension, intérieur relatif d'un convexe) On appelle *dimension* d'un convexe non vide X la dimension du plus petit espace affine E le contenant (celui-ci est bien défini puisque la classe des espaces affines est stable par intersection). On appelle *intérieur relatif* de X l'intérieur de X dans E (qui est non vide si $X \neq \emptyset$).

Corollaire 2.11 (Théorème de Minkowski) *Tout ensemble compact convexe de $\mathbb{R}^n$ coïncide avec l'enveloppe convexe de ses points extrêmes.*

Démonstration. Soit K un compact convexe de $\mathbb{R}^n$. On va montrer le théorème par récurrence sur la dimension de K. Quitte à remplacer $\mathbb{R}^n$ par un sous-espace affine, on peut supposer que K est de dimension n. Si $n = 0$, K est réduit à un point, et le théorème est vérifié. Supposons donc le théorème démontré pour les compacts convexes de dimension au plus $n-1$, et montrons que tout point x de K est barycentre d'un nombre fini de point extrêmes de K. Si x est un point frontière de K, la remarque 2.5 fournit un hyperplan d'appui H de K en x. Comme $K \cap H$ est un compact convexe de dimension au plus $n-1$, par hypothèse de récurrence, x est barycentre d'un nombre fini de points extrêmes de $K \cap H$, qui sont aussi des points extrêmes de K d'après le Lemme 2.9. Prenons maintenant un point quelconque x de K, et soit D une droite affine passant par x. L'ensemble $D \cap K$ est un segment de la forme $[y, z]$, où les points y, z sont des points frontières de K. D'après ce qui précède, y et z sont barycentres d'un nombre fini de points extrêmes de K. Comme x est lui même barycentre de y et z, le théorème est démontré. ■

2.2 Quelques résultats sur les polyèdres

Définition 2.12 (Polyèdre, polytope, cône polyédral) Un *polyèdre* est une partie d'un espace euclidien, intersection d'un nombre fini de demi-espaces affines fermés. En d'autres termes, une partie P de $\mathbb{R}^n$ est un polyèdre ssi il existe un entier $p \in \mathbb{N}$ et A matrice de taille $p \times n$, $b \in \mathbb{R}^p$ tels que $P = \{x \in \mathbb{R}^n;\ Ax \leqslant b\}$ (si $p = 0$, P est l'espace entier). Un *polytope* est un polyèdre borné.

Si P est le polyèdre $\{x \in \mathbb{R}^n;\ Ax \leqslant b\}$, et $x \in P$, on appelle $I(x) = \{1 \leqslant i \leqslant m \mid A_i x = b_i\}$ l'*ensemble des contraintes actives* en x. Ici A_i désigne la i-ième ligne de A.

Un *cône* est un ensemble stable par multiplication par un nombre strictement positif ; $C \subset \mathbb{R}^n$ est donc un cône ssi, pour tout $\alpha > 0$, on a $C = \alpha C$, où $\alpha C := \{\alpha x,\ x \in C\}$. On appellera *cône polyédral* les cônes qui sont des polyèdres. À titre d'exercice, on pourra vérifier qu'un tel ensemble est caractérisé par un nombre fini d'inégalités linéaires homogènes, de la forme $\{x \in \mathbb{R}^n;\ Ax \leqslant 0\}$.

Un cône contenant 0 est convexe ssi il contient les combinaisons linéaires positives de ses éléments. Nous appellerons *direction* une demi-droite (vectorielle), de la forme $\mathbb{R}_+ x$, où $x \in \mathbb{R}^n \setminus \{0\}$ est un *représentant* de la demi-droite.

Nous dirons qu'une famille de vecteurs E *génère* un cône convexe C si $E \subset C$, et si tout élément de C est combinaison linéaire positive d'un nombre fini de vecteurs de E. On dit que C est le cône convexe *engendré* par E. On notera $\mathrm{cone}(E)$ le cône convexe engendré par $E \subset \mathbb{R}^n$.

Ainsi, la notion de cône convexe engendré par un ensemble est similaire à celle d'espace vectoriel réel engendré par un ensemble, à ceci près que dans le cas des cônes convexes, on exige que les coefficients soient positifs ou nuls.

Théorème 2.13 (de Carathéodory) *Toute combinaison linéaire positive d'une famille finie de vecteurs de $\mathbb{R}^n$ est aussi combinaison linéaire positive d'une sous famille linéairement indépendante. Autrement dit, si $\{x^i,\ i \in I\}$ est une partie finie de $\mathbb{R}^n$, et si $y = \sum_{i \in I} \alpha_i x^i$, avec $\alpha_i \geqslant 0$ pour tout $i \in I$, alors il existe $J \subset I$ tel que la famille $\{x^j,\ j \in J\}$ est linéairement indépendante, et des nombres positifs β_j, $j \in J$, tels que $y = \sum_{i \in J} \beta_j x^j$.*

Démonstration. Soit $y \in \mathrm{cone}(\{x^1, \ldots, x^k\})$. Si y est nul, on conclut immédiatement. Sinon, quitte à remplacer $\{x^1, \ldots, x^k\}$ par un sous-ensemble, on peut supposer que y est combinaison linéaire à coefficients strictement positifs de $x^1, \ldots, x^k$, soit $y = \lambda_1 x^1 + \cdots + \lambda_k x^k$, avec $\lambda_1, \ldots, \lambda_k > 0$. Si les $x^1, \ldots, x^k$ sont linéairement dépendants, on a $\mu_1 x^1 + \cdots + \mu_k x^k = 0$, où les $\mu_1, \ldots, \mu_k$ sont des réels non tous nuls. L'ensemble $\Lambda = \{\nu \in \mathbb{R} \mid \lambda_j + \mu_j \nu \geqslant 0,\ \forall 1 \leqslant j \leqslant k\}$ est non-vide (il contient 0), fermé, et est strictement inclus dans $\mathbb{R}$ (car $\mu \neq 0$). Soit $\bar{\nu}$ un point frontière de Λ. Alors l'ensemble $I := \{1 \leqslant i \leqslant k \mid \lambda_i + \mu_i \bar{\nu} > 0\}$, est strictement inclus dans $\{1, \ldots, k\}$. Comme $y = \sum_{1 \leqslant i \leqslant k} (\lambda_i + \mu_i \bar{\nu}) x^i = \sum_{i \in I} (\lambda_i + \mu_i \bar{\nu}) x^i$, le vecteur y est combinaison

linéaire à coefficients strictement positifs d'au plus $k-1$ des vecteurs $x^1, \dots, x^k$. On conclut par récurrence sur k. ■

Corollaire 2.14 *Une combinaison linéaire positive d'un nombre fini de vecteurs de $\mathbb{R}^n$ peut s'écrire comme combinaison linéaire positive d'au plus n de ces vecteurs.*

Corollaire 2.15 *Le barycentre d'un nombre fini de vecteurs de $\mathbb{R}^n$ peut s'écrire comme barycentre d'au plus $n+1$ de ces vecteurs.*

Démonstration. À tout vecteur x de $\mathbb{R}^n$, associons le vecteur formé en ajoutant une $n+1$ème composante égale à 1 : $\tilde{x} := \begin{pmatrix} x \\ 1 \end{pmatrix}$. Soit $E \subset \mathbb{R}^n$. On vérifie facilement que $y \in \mathrm{conv}(E)$ ssi $\tilde{y}$ est combinaison linéaire positive de vecteurs de $E \times \{1\}$. La conclusion découle du théorème de Carathéodory 2.13. ■

Lemme 2.16 *L'ensemble des combinaisons linéaires positives d'une partie finie E de $\mathbb{R}^n$ est fermé.*

Démonstration. Soit x^k une suite dans $\mathrm{cone}(E)$ de limite $\bar{x}$. D'après le théorème de Carathéodory, chaque x^k peut s'écrire sous la forme $\sum_{i=1}^p \lambda_i u^i$, avec $u_1, \dots, u_p$ famille linéairement indépendante de E. Comme E est fini, extrayant si nécessaire une sous suite, on peut supposer que cette famille reste identique pour tout k ; c'est une base d'un espace vectoriel V contenant la suite $\{x^k\}$; la convergence de x^k entraîne celle des vecteurs de coordonnées λ^k dans cette base vers un certain $\bar{\lambda}$, et donc $\bar{x} = \sum_{i=1}^p \bar{\lambda}_i u^i$ appartient à $\mathrm{cone}(E)$. ■

La notion de point extrême, qui a fait l'objet de la définition 2.8, n'a guère d'intérêt dans le cas d'un cône convexe C : pour $0 < \lambda < 1 < \mu$, on peut en effet écrire un point $x \in C$ comme barycentre de λx et de μx, ce qui montre que x n'est pas extrême, à moins que $x = 0$. Aussi, la notion d'extrémalité doit être adaptée comme suit dans ce cas.

Définition 2.17 (i) On dira qu'un vecteur $x \neq 0$ représente une *direction extrême* du cône convexe C si $x \in C$ et si $x = \alpha y + \beta z$ avec $y, z \in C \setminus \{0\}$, $\alpha > 0$ et $\beta > 0$, entraîne que $x = \lambda y = \mu z$, avec $\lambda, \mu \geqslant 0$. Autrement dit, la direction de x n'est pas somme de deux directions distinctes de C.
(ii) Un cône est dit *pointé* s'il ne contient pas de droite vectorielle.

Lemme 2.18 *Un cône polyédral pointé est généré par ses directions extrêmes. Celles-ci sont en nombre fini, et correspondent aux vecteurs non nuls du cône dont l'ensemble de contraintes actives est maximal (pour la relation d'inclusion des ensembles).*

Démonstration. Notons $\mathcal{I} := \{I(x);\ x \in C \setminus \{0\}\}$ l'ensemble des ensembles de contraintes actives d'éléments non nuls de C. Soient $I_1, \dots, I_q$ les éléments maximaux de $\mathcal{I}$ pour la relation d'inclusion des ensembles, et $x^1, \dots, x^q$ des éléments non nuls de C tels que $I(x^k) = I_k$ pour tout k. Montrons que la famille $\{x^1, \dots, x^q\}$ génère C.

Montrons d'abord que si $x \in C$ est tel que $I(x) = I_1$, alors $x = \alpha x^1$ pour un certain $\alpha \in \mathbb{R}_+$. En effet, $x^\varepsilon := x^1 - \varepsilon x$ appartient à C pour $\varepsilon > 0$ assez proche de 0, puisque $I(x) = I(x^1)$. Comme C est pointé, il ne contient pas $-x$. Il existe donc $x' \in [x^\varepsilon, -x[$ tel que $I(x')$ contient strictement I^1. Si $x \notin \mathbb{R}_+ x^1$, alors $x' \neq 0$ ce qui contredit la maximalité de I_1.

Nous en déduisons que les directions $\{x^1, \dots, x^q\}$ sont extrêmes. En effet, supposons par exemple que $x^1 = \alpha y + (1-\alpha)z$, avec $\alpha \in]0,1[$ et y, z dans $C \setminus \{0\}$. Pour tout $i \in I(x^1)$, on a $0 = A_i x^1 - b_i = \alpha(A_i y - b_i) + (1-\alpha)(A_i z - b_i)$. Comme $A_i y \leqslant b_i$ et $A_i z \leqslant b_i$, ceci implique $i \in I(y)$ et $i \in I(z)$, donc y et z représentent la même direction que x.

Par ailleurs, si $x \in C$, et $I(x)$ n'est pas maximal, on peut supposer que $I(x) \subset I(x^1)$. Posons $x^t := x - t x^1$. Pour $t > 0$ assez petit, $x^t \in C$. Puisque C est pointé et fermé, il ne contient ni $-x^1$, ni $x/t - x^1 = x^t/t$ pour $t > 0$ assez grand. Comme C est un cône, ceci veut dire que $x^t \notin C$, pour $t > 0$ assez grand. Il existe donc $t > 0$ tel que $x^t \in C$ et $I(x^t) \supset I(x)$ strictement. Comme $x = t x^1 + x^t$, nous avons montré que tout

$x \in C$ distinct d'une direction extrémale s'exprime comme combinaison linéaire positive de deux points (en l'occurrence x^1 et x^t), dont chacun a au moins une contrainte active de plus que x. Appliquant cette propriété pour les deux points et substituant leurs expressions autant de fois que possible, on obtient à la fin une expression de x comme combinaison linéaire positive d'éléments de C dont les indices actifs sont maximaux, et donc x est bien combinaison linéaire positive de $\{x^1, \ldots, x^q\}$. ■

La *somme de Minkowski* de deux parties E et F de $\mathbb{R}^n$ est définie comme

$$E + F := \{e + f;\ e \in E,\ f \in F\}. \tag{2.4}$$

On note $\mathrm{lin}(P)$ et on appelle *espace de linéalité* d'un polyèdre P le plus grand espace vectoriel inclus dans $P - x^0$, pour tout $x^0 \in P$ (la définition est indépendante du choix de x^0). On appelle cône asymptote du polyèdre P l'ensemble

$$\{y \in \mathbb{R}^n;\ x + y \in P,\ \text{pour tout } x \in P\}.$$

Si P est non vide et de la forme $\{x \in \mathbb{R}^n;\ Ax \leqslant b\}$, alors $\mathrm{lin}(P)$ n'est autre que le noyau de A, et le cône asymptote de P est $\{x \in \mathbb{R}^n;\ Ax \leqslant 0\}$.

Corollaire 2.19 *On obtient des générateurs d'un cône polyédral C par l'union des directions extrêmes du cône pointé $C' := C \cap (\mathrm{lin}(C)^\perp)$, et d'une famille génératrice de* $\mathrm{lin}(C)$ *(par exemple une base et son opposé).*

Démonstration. Il est clair que C' est un cône polyédral pointé et inclus dans C. Or tout $x \in C$ a une décomposition unique en somme d'éléments de $\mathrm{lin}(C)$ et $\mathrm{lin}(C)^\perp$, soit $x = y + z$. Mais $z = x - y \in C$ par définition de $\mathrm{lin}(C)$, donc $C = C' + \mathrm{lin}(C)$. D'après le lemme 2.18, une famille génératrice de C sera donc obtenue en adjoignant aux directions extrêmes C' (en nombre fini d'après le lemme 2.18) un générateur minimal (au sens des cônes) de $\mathrm{lin}(C)$. ■

Lemme 2.20 *Un polyèdre P peut se décomposer sous la forme $P = Q + C$, où Q est l'enveloppe convexe d'un nombre fini de points, et C est un cône engendré par un nombre fini de directions, dit cône asymptote.*

Démonstration. Le cône polyédral

$$E := \{(x, \alpha) \in \mathbb{R}^n \times \mathbb{R}_+;\ Ax \leqslant \alpha b\} \tag{2.5}$$

est, d'après le corollaire 2.19 généré par un nombre fini de vecteurs notés (x^i, α^i), $i = 1$ à m. On peut supposer que chaque α^i vaut 1 si $i \leqslant q$, et 0 sinon. Soit Q l'enveloppe convexe des x^i pour $i \leqslant q$, et C le cône généré par les x^i pour $q < i \leqslant m$. Alors $x \in P$ ssi $(x, 1) \in E$, soit $x = \sum_{i=1}^m \beta_i x^i$, avec $\beta \in \mathbb{R}_+^m$ tel que $\sum_{i \leqslant q} \beta_i = 1$. La conclusion s'ensuit. ■

Le lemme 2.20 a une réciproque, qui s'appuie sur la notion suivante. Soit C une partie convexe de $\mathbb{R}^n$. On appelle *polaire* de C l'ensemble défini par

$$C^o := \{y \in \mathbb{R}^n;\ y \cdot x \leqslant 1,\ \text{pour tout}\ \ x \in C\}. \tag{2.6}$$

Cet ensemble est convexe, fermé et contient 0.

Théorème 2.21 (Minkowski-Weyl) *La classe des polyèdres coïncide avec celle des ensembles de la forme*

$$P := \mathrm{conv}\{x^i,\ i \in I\} + \mathrm{cone}\{y^j,\ j \in J\}, \tag{2.7}$$

où $(x^i)_{i \in I}$ et $(y^j)_{j \in J}$ sont des familles finies de vecteurs.

Démonstration. Combinant les lemmes 2.20 et 2.18, on vérifie que tout polyèdre est de la forme (2.7). Réciproquement, soient P satisfaisant (2.7), et $\bar{x} \in P$. Alors $P = \bar{x} + P'$, où $P' = \mathrm{conv}\{(x^i - \bar{x}),\ i \in I\} + \mathrm{cone}\{y^j,\ j \in J\}$. La classe des ensembles de la forme (2.7) est donc invariante par translation, comme

celle des polyèdres. On peut donc sans perte de généralité supposer que $0 \in P$. La polaire de P a pour expression

$$P^o := \{z \in \mathbb{R}^n;\ z \cdot x^i \leqslant 1;\ i \in I; \quad z \cdot y^j \leqslant 0;\ j \in J\}. \tag{2.8}$$

C'est donc un polyèdre, d'après le lemme 2.20, de la forme (2.7). De même, le bipolaire $P^{oo} := (P^o)^o$ est un polyèdre. Nous terminons la preuve en montrant que $P^{oo} = P$. Par définition de la polarité, on a $P \subset P^{oo}$ et il faut montrer l'inclusion inverse. Posons

$$Q := \text{conv}\{x^i,\ i \in I\}; \quad C := \text{cone}\{y^j,\ j \in J\}. \tag{2.9}$$

D'après le lemme 2.16, C est fermé, donc $P = Q + C$ est fermé car somme d'un compact et d'un fermé. Soit $\hat{x} \notin P$; D'après le corollaire 2.4, il existe une forme linéaire séparant strictement $\hat{x}$ du convexe fermé P :

$$a \cdot \hat{x} > a \cdot x, \quad \text{pour tout} \quad x \in P. \tag{2.10}$$

Posons $\gamma := a \cdot \hat{x}$. En raison de (2.7), la relation (2.10) équivaut à

$$\gamma > a \cdot x^i,\ i \in I; \qquad \text{et, si } J \neq \emptyset, \quad 0 \geqslant a \cdot y^j,\ j \in J. \tag{2.11}$$

Comme de plus $0 \in P$, on a $\gamma > 0$. Par mise à l'échelle de a, on peut imposer $\gamma > 1$, et $a \cdot x^i \leqslant 1$, $i \in I$. Les relations ci-dessus prouvent que $a \in P^o$. Comme $a \cdot \hat{x} = \gamma > 1$, $\hat{x} \in P^{oo}$ est impossible, comme on voulait le montrer. ■

Énonçons le cas particulier du théorème 2.21 concernant les polyèdres bornés.

Corollaire 2.22 *La classe des polytopes coïncide avec celle des combinaisons convexes d'un nombre fini de points.*

Lemme 2.23 *La classe des polytopes, comme celle des polyèdres, est stable par addition.*

Démonstration. On utilise le Théorème de Minkowski-Weyl 2.21. Explicitons l'argument pour une somme de polytopes. Soient $Q = \text{conv}\{x^1, \ldots, x^s\}$ et $Q' = \text{conv}\{y^1, \ldots, y^t\}$, $x = \sum_{i=1}^s \alpha_i x^i$ et $y = \sum_{j=1}^t \beta_j y^j$, avec α comme β positifs de somme 1. Alors $x + y = \sum_{i,j} \alpha_i \beta_j (x^i + y^j)$ (on le vérifie en réexprimant cette dernière expression sous la forme de combinaisons des x^i et y^j). Comme $x^i + y^j \in Q + Q'$, pour tout (i, j), et que $\sum_{i,j} \alpha_i \beta_j = 1$, ceci montre que $Q + Q'$ est combinaison convexe des $x^i + y^j$. ■

La proposition suivante permet d'interpréter de manière géométrique les vecteurs x^i intervenant dans la décomposition du théorème de Minkowski-Weyl.

Proposition 2.24 *Soit P un polyèdre ne contenant aucune droite affine. Alors le plus petit ensemble $\{x^i,\ i \in I\}$ tel que P puisse s'écrire sous la forme du théorème 2.21, avec $Q := \text{conv}(\{x^i,\ i \in I\})$, est précisément l'ensemble des points extrêmes de P.*

Démonstration. Soit z un point extrême de P, et considérons une représentation de P de la forme (2.7), de sorte que $z = \sum_{i \in I} \alpha_i x^i + \sum_{j \in J} \lambda_j y^j$, avec $\alpha_i, \lambda_j \geqslant 0$ et $\sum_{i \in I} \alpha_i = 1$. Posons $y = \sum_{j \in J} \lambda_j y^j$. On peut écrire $z = \sum_{i \in I, \alpha_i \neq 0} \alpha_i (x^i + y)$, et comme z est extrême, ceci entraîne que $z = x^i + y$ pour tout $i \in I$ tel que $\alpha_i \neq 0$. Si $y \neq 0$, on peut écrire $y = \beta' y' + \beta'' y''$, où $y' \neq y''$ sont des multiples positifs de y, et $0 < \beta', \beta''$ avec $\beta' + \beta'' = 1$. On a alors $z = \beta'(x^i + y') + \beta''(x^i + y'')$, ce qui contredit l'extrémalité de z. Nous venons de montrer que $z = x^i$: tout point extrême de P apparaît ainsi nécessairement dans une représentation de la forme (2.7).

Réciproquement, soit $\{x^i,\ i \in I\}$ un ensemble minimal (pour l'inclusion) permettant d'écrire P sous la forme (2.7). Pour tout $k \in I$, considérons $P_k := \text{conv}\{x^i,\ i \in I \setminus \{k\}\} + \text{cone}\{y^j,\ j \in J\}$. L'hypothèse de minimalité entraîne que $x^k \notin P_k$. L'ensemble P_k est encore un polyèdre (d'après le théorème de Minkowski-Weyl), en particulier, c'est un ensemble convexe fermé, donc on peut trouver un vecteur $q \in \mathbb{R}^n$ tel que l'on ait la séparation stricte (Corollaire 2.4) :

$$q \cdot x^k < \inf\{q \cdot z,\ z \in P_k\}\ . \tag{2.12}$$

Afin de montrer que x^k est un point extrême de P, il suffit de montrer que x^k se représente de manière unique comme combinaison convexe des x^i et combinaison positive des y^j, i.e., que pour tous $\alpha_i, \lambda_j \geqslant 0$, avec $\sum_{i\in I} \alpha_i = 1$, on a

$$x^k = \sum_{i\in I} \alpha_i x^i + \sum_{j\in J} \lambda_j y^j \implies \alpha_k = 1,\ \alpha_i = 0,\ \forall i \in I \setminus \{k\} \text{ et } \lambda_j = 0,\ \forall j \in J. \tag{2.13}$$

Notons que $q \cdot y^j \geqslant 0$, pour tout $j \in J$, sinon, la fonction $x \mapsto q \cdot x$ ne serait pas minorée sur P_k, ce qui contredirait la propriété de séparation stricte (2.12). Par ailleurs, celle-ci implique $q \cdot x^i > q \cdot x^k$, pour $i \neq k$. On déduit aussitôt que $\alpha_i = 0$ pour $i \neq k$, et donc $\alpha_k = 1$. En simplifiant x^k de part et d'autre de l'égalité $x^k = \sum_{i\in I} \alpha_i x^i + \sum_{j\in J} \lambda_j y^j = x^k + \sum_{j\in J} \lambda_j y^j$, il vient $\sum_{j\in J} \lambda_j y^j = 0$. Nous voulons montrer que $\lambda = 0$. Si ce n'est pas le cas, soit k tel que $\lambda_k > 0$; l'expression $-y_k = (\lambda_k)^{-1} \sum_{j\in J\setminus\{k\}} \lambda_j y^j$ montre que $-y_k$ appartient au cône asymptote de P. Comme, par définition, y_k appartient à ce cône, P contient une droite affine, ce qui donne la contradiction recherchée. ■

Corollaire 2.25 *Une fonction concave minorée sur un polyèdre de $\mathbb{R}^n$ admet un point de minimum sur ce polyèdre.*

Démonstration. D'après le Lemme 2.20, un polyèdre P peut s'écrire $P = Q + C$, où Q est un polytope, et C un cône polyédral. Pour tout $x \in P$, on a donc $x = u + v$, avec $u \in Q$, $v \in C$, et $u + \lambda v \in P$, pour tout $\lambda \geqslant 0$. Si la fonction f est concave, la fonction $\phi : \lambda \to \lambda^{-1}(f(u + \lambda v) - f(u))$ est décroissante. S'il existe $\bar\lambda$ tel que $\phi(\bar\lambda) < 0$, on a

$$\inf_{\lambda\geqslant\bar\lambda} f(u + \lambda v) = \inf_{\lambda\geqslant\bar\lambda} f(u) + \lambda\phi(\lambda) \leqslant \inf_{\lambda\geqslant\bar\lambda} f(u) + \lambda\phi(\bar\lambda) = -\infty\ .$$

Donc, l'hypothèse que f est minorée sur P entraîne que $f(x) \geqslant f(u)$. Comme Q est un polytope, Q est l'ensemble des barycentres d'un ensemble fini de points, $\{u_1, \ldots, u_k\}$. Donc $x = \alpha_1 u_1 + \cdots + \alpha_k u_k$, où les α_i sont des réels positifs de somme 1. En utilisant à nouveau la concavité de f, il vient $f(x) \geqslant \alpha_1 f(u_1) + \cdots + \alpha_k f(u_k) \geqslant \min(f(u_1), \ldots, f(u_k))$, ce qui montre que le minimum de f sur P est atteint en l'un des points $u_1, \ldots, u_k$. ■

En combinant la preuve du corollaire 2.25 avec la proposition 2.24, on obtient :

Corollaire 2.26 *Une fonction concave minorée sur un polyèdre de $\mathbb{R}^n$ ne contenant pas de droite affine admet un point de minimum qui est un point extrême du polyèdre.*

Remarque 2.27 Soit $P = Q + C$ un polyèdre, où Q est un polytope, et C est son cône asymptote. L'analyse de la démonstration ci-dessus permet de montrer que la fonction concave f est minorée sur le polyèdre P ssi pour tout $u \in Q$, la fonction $v \mapsto \lim_{\lambda\to+\infty} \lambda^{-1}(f(u + \lambda v) - f(u))$ est positive sur C (le lecteur pourra montrer en guise d'exercice que cette fonction, appelée fonction récession, ne dépend pas du choix de u, en raison de la concavité de f). En particulier, le programme linéaire $\mathrm{Min}\{c \cdot x;\ x \in P\}$ a une valeur finie ssi $c \cdot x \geqslant 0$ pour tout $x \in C$, et dans ce cas il atteint son minimum sur P en au moins un des points extrêmes de Q. •

Remarque 2.28 On applique souvent le corollaire 2.25 dans le cas particulier où la fonction f est linéaire. •

2.3 Intégrité des points extrêmes

Le Corollaire 2.26 montre que le minimum d'une fonction concave sur un polyèdre est atteint en un point extrême, si le polyèdre ne contient pas de droite et si l'infimum de cette fonction sur le polyèdre est fini. Nous nous intéressons maintenant, pour cette raison, au caractère entier des points extrêmes d'un polyèdre

$$P = \{x \in \mathbb{R}^n \mid Ax \leqslant b\} \tag{2.14}$$

avec $A \in \mathbb{R}^{m\times n}$ et $b \in \mathbb{R}^m$.

Lemme 2.29 *Un point extrême du polyèdre P défini par* (2.14) *est nécessairement solution d'un système $A'x = b'$, où A' est une sous-matrice inversible formée de n lignes de A, et b' est le vecteur formé des lignes correspondantes de b.*

Démonstration. Soit x un point de P. Rappelons que $I(x) = \{1 \leqslant i \leqslant m \mid A_i x = b_i\}$ désigne l'ensemble des contraintes actives en x. Si la famille $\{A_i\}_{i\in I(x)}$ n'est pas de rang n, on peut trouver un vecteur non-nul y tel que $A_i y = 0$ pour tout $i \in I(x)$. Comme x est le milieu des points $x - \epsilon y$ et $x + \epsilon y$, qui sont bien des éléments de P si ϵ est assez petit, on contredit le caractère extrême de x. Ainsi, on peut trouver un sous ensemble $I' \subset I(x)$ de cardinal n tel que la matrice $n \times n$ dont les lignes sont les A_i, avec $i \in I'$, est inversible. Le système $A_i x = b_i, i \in I'$, caractérise alors x. ■

Remarque 2.30 Le Lemme 2.29 montre de manière élémentaire qu'un polyèdre n'a qu'un nombre fini de points extrêmes. (On peut retrouver cette conclusion à partir du théorème de Minkowski-Weyl.) •

Exercice 2.31 Montrer réciproquement que si x est un point de P vérifiant $A'x = b'$, avec A' et b' comme dans le Lemme 2.29, alors x est un point extrême.

Le Lemme 2.29 suggère d'étudier les cas où la solution d'un système linéaire est entière :

Proposition 2.32 *Soit $A \in \mathbb{Z}^{n\times n}$ une matrice inversible. Les assertions suivantes sont équivalentes :*

1. $\det A = \pm 1$ *;*
2. *si $b \in \mathbb{Z}^n$, alors $A^{-1}b \in \mathbb{Z}^n$.*

Démonstration. L'implication 1⇒2 résulte aussitôt des formules de Cramer. Réciproquement, supposons que A vérifie l'assertion 2. Montrons d'abord que A^{-1} est à coefficients entiers. En prenant pour b le i-ième vecteur de la base canonique de $\mathbb{R}^n$, on voit que la i-ième colonne de A^{-1}, qui coïncide avec $A^{-1}b$, est à coefficients entiers. Comme ceci est vrai pour tout $1 \leqslant i \leqslant n$, on a $A^{-1} \in \mathbb{Z}^{n\times n}$. Donc $\det A^{-1} \in \mathbb{Z}$, et $1 = \det A \det A^{-1}$ montre que $\det A$ divise 1, c'est-à-dire que $\det A = \pm 1$. ■

Définition 2.33 (Matrices totalement unimodulaires) On dit que $A \in \mathbb{Z}^{n\times n}$ est *unimodulaire* quand $\det A = \pm 1$, et que $B \in \mathbb{Z}^{m\times n}$ est *totalement unimodulaire* quand toute sous matrice carrée extraite de B est de déterminant ± 1 ou 0.

En prenant des sous-matrices 1×1, on voit en particulier que les coefficients d'une matrice totalement unimodulaire valent nécessairement ± 1 ou 0. L'introduction des matrices totalement unimodulaires est motivée par le résultat suivant.

Corollaire 2.34 (Optimalité des solutions entières) *Soient $D \in \mathbb{Z}^{m\times n}$ une matrice totalement unimodulaire, $h \in (\mathbb{Z} \cup \{+\infty\})^m$, $h' \in (\mathbb{Z} \cup \{-\infty\})^m$, $g \in (\mathbb{Z} \cup \{+\infty\})^n$, et $g' \in (\mathbb{Z} \cup \{-\infty\})^n$. Alors, les points extrêmes du polyèdre*

$$Q = \{x \in \mathbb{R}^n \mid h' \leqslant Dx \leqslant h, \quad g' \leqslant x \leqslant g\} \tag{2.15}$$

sont nécessairement entiers. En particulier, si Q ne contient pas de droite affine, pour toute fonction concave f minorée sur Q, le minimum de f sur Q est atteint en (au moins) un point entier de Q :

$$\inf_{x\in Q} f(x) = \min_{x\in Q\cap\mathbb{Z}^n} f(x) \ . \tag{2.16}$$

Démonstration. On peut écrire

$$Q = \{x \in \mathbb{R}^n \mid Ax \leqslant b\} \ , \tag{2.17}$$

où b est un vecteur entier fini et A est une matrice dont chaque ligne est soit de la forme $\pm D_i$, avec D_i une ligne quelconque de D, soit de la forme $\pm e_j$, où e_j est le j-ième vecteur de la base canonique de $\mathbb{R}^n$, et $1 \leqslant j \leqslant n$ est un indice quelconque.

On montre d'abord que A est totalement unimodulaire. Soit donc M une sous-matrice $k \times k$ extraite de A, et montrons, par récurrence sur k que $\det M \in \{\pm 1, 0\}$. Si $k = 1$, cela résulte aussitôt de la totale unimodularité de D. Supposons maintenant le résultat prouvé pour toutes les sous-matrices carrées de A de dimension au plus $k-1$, et montrons le pour M. Si M contient une ligne égale à un vecteur $\pm e_j$, on développe $\det M$ par rapport à cette ligne, et par récurrence, le résultat est prouvé. Sinon, M coïncide, au changement du signe de certaines lignes près, avec une sous-matrice de D, et comme D est totalement unimodulaire, $\det M \in \{\pm 1, 0\}$, ce qui achève la preuve de la totale unimodularité de A.

Comme Q est donné par (2.17), avec b entier et A totalement unimodulaire, il résulte du Lemme 2.29 et de la Proposition 2.32 que les points extrêmes de Q, s'ils existent, sont entiers.

La dernière assertion résulte du Corollaire 2.26. ■

Remarque 2.35 On notera que le Corollaire 2.34 ne pose aucune condition sur f, hormis la concavité. En particulier, si $f(x) = c \cdot x$ est linéaire, le caractère entier des solutions optimales n'est pas lié au caractère entier du vecteur de coût c. •

Remarque 2.36 Le corollaire 2.34 ne dit surtout pas que toutes les solutions optimales sont entières. D'ailleurs, lorsque f est linéaire, il ne peut en être ainsi à moins que la solution optimale ne soit unique, car tout barycentre de solutions optimales d'un programme linéaire est solution optimale. •

Il existe de nombreux résultats sur les matrices totalement unimodulaires. Nous nous bornons ici à donner une condition suffisante très utile.

Proposition 2.37 (Poincaré) *Si A est une matrice à coefficients ± 1 ou 0, avec au plus un coefficient 1 par colonne, et au plus un coefficient -1 par colonne, alors A est totalement unimodulaire.*

Démonstration. Comme la propriété que vérifie A passe aux sous-matrices, il suffit de vérifier que si A est carrée, $\det A \in \{\pm 1, 0\}$. Cela est vrai si A est de taille 1. Si A a une colonne nulle, $\det A = 0$. Si A a une colonne avec seulement un coefficient non-nul, on développe le déterminant par rapport à cette colonne, et l'on conclut par récurrence que $\det A \in \{\pm 1, 0\}$. Il ne reste qu'a considérer le cas où chaque colonne de A a exactement un coefficient 1 et exactement un coefficient -1 : alors, chaque colonne est de somme nulle, donc $\det A = 0$. ■

Nous appliquerons la Proposition 2.37 aux problèmes de flots dans la section 3.1. Donnons pour l'instant en exercice un cas où l'on peut conclure directement à la totale unimodularité :

Exercice 2.38 (Problème de couverture) Un centre d'appel téléphonique a une courbe de charge : c_t est le nombre de clients devant être servis à l'instant discret $t \in \{1, \ldots, T\}$. Des conseillers de clientèle répondent aux appels. On simplifie le problème en supposant que tous les appels sont de même type. On supposera qu'il y a k horaires de travail possibles, l'horaire i étant caractérisé par un intervalle $[\alpha_i, \beta_i]$, avec $1 \leqslant \alpha_i \leqslant \beta_i \leqslant T$, ce qui revient à faire abstraction des pauses. On note S_i le salaire à verser à un conseiller de clientèle travaillant de l'instant α_i à l'instant β_i. On pose $u_{it} = 1$ si $\alpha_i \leqslant t \leqslant \beta_i$, et $u_{it} = 0$ sinon. Justifier le problème

$$\inf_{\substack{x \in \mathcal{N}^k \\ \sum_{1 \leqslant i \leqslant k} x_i u_{it} \geqslant c_t, \ \forall 1 \leqslant t \leqslant T}} \sum_{1 \leqslant i \leqslant k} x_i S_i \ . \tag{2.18}$$

1. Montrer que l'ensemble des solutions admissibles de ce problème peut s'écrire comme l'ensemble des points entiers d'un polyèdre de la forme

$$Q = \{x \in \mathbb{R}^n \mid c \leqslant Dx \quad 0 \leqslant x\} \tag{2.19}$$

où la matrice D est une matrice d'intervalles, c'est-à-dire une matrice à coefficients $0, 1$ telle que les 1 apparaissent consécutivement sur une colonne.

2. Montrer qu'une matrice d'intervalles est totalement unimodulaire.

3. Conclure qu'il est équivalent de résoudre le problème en nombres entiers (2.18) et le problème de programmation linéaire (à inconnues réelles)

$$\inf_{x \in Q} \sum_{1 \leqslant i \leqslant k} x_i S_i \ . \tag{2.20}$$

Une fois vu le chapitre suivant, vous pourrez retrouver ce résultat de manière intuitive, en modélisant le problème de couverture par un problème de flot à coût minimum.

Exercice 2.39 (Théorème de Birkhoff) On considère un graphe biparti complet avec $2n$ sommets, que l'on peut voir comme un graphe avec deux sortes de sommets, les sommets féminins, notés $\{1, \ldots, n\}$, et les sommets masculins, notés $\{1', \ldots, n'\}$, tel qu'une arête $\{i, j'\}$ relie chaque sommet fille i à chaque sommet garçon j'. On munit cette arête d'un poids w_{ij}. On appelle *couplage* un ensemble d'arêtes C tel que chaque sommet appartienne au plus à une arête de C. Le poids d'un couplage est la somme des poids de ses arêtes. Le couplage est dit parfait si chaque sommet appartient à exactement une arête.

1. Montrer que le poids maximum d'un couplage parfait se modélise par le PLNE

$$\bar{w} = \max_{\substack{x \in \mathbb{N}^{n \times n} \\ \forall i,\ \sum_j x_{ij} = 1, \\ \forall j,\ \sum_i x_{ij} = 1}} \sum_{1 \leqslant i \leqslant n,\ 1 \leqslant j \leqslant n} w_{ij} x_{ij}$$

2. Montrer que le problème relâché continu

$$\max_{\substack{x \in \mathbb{R}_+^{n \times n} \\ \forall i,\ \sum_j x_{ij} = 1, \\ \forall j,\ \sum_i x_{ij} = 1}} \sum_{1 \leqslant i \leqslant n,\ 1 \leqslant j \leqslant m'} w_{ij} x_{ij} \tag{2.21}$$

a même valeur que le PLNE.

Indication : écrire les contraintes égalité sous la forme $Dx = b$, et appliquer le théorème de Poincaré (prop. 2.37) pour montrer que D est totalement unimodulaire (on retrouvera ce résultat avec la théorie des flots, voir l'exemple 3.16).

3. On appelle matrice *bistochastique* une matrice admissible pour le problème (2.21). Montrer qu'une telle matrice est enveloppe convexe de matrices de permutations (Théorème de Birkhoff).

4. À un bal, il y a n garçons et n filles, chaque garçon ayant été présenté à r filles, et chaque fille ayant été présentée à r garçons. On suppose $r \geqslant 1$. Déduire du théorème de Birkhoff qu'il est possible de former n couples de danseurs, de sorte que les danseurs d'un même couple aient déjà été présentés l'un à l'autre.

Indication. Considérer la matrice x_{ij} telle que $x_{ij} = 1$ lorsque i a été présenté à j' et 0 sinon.

Exercice 2.40 (Théorème de Gale et Shapley sur les mariages stables) Une autre approche du problème de mariages consiste à remplacer la fonction d'utilité de la société par des préférences individuelles : on suppose que chaque fille a un ordre de préférence (total) entre les différents garçons qu'elle connaît, et qu'il en est de même pour les garçons. Un ensemble de mariages est dit *stable* si lorsqu'une fille,

disons Alice, et un garçon, disons Bob, ne sont pas mariés entre eux, soit Alice est mariée à un garçon qu'elle préfère à Bob, soit Bob est marié à une fille qu'il préfère à Alice.

On propose l'algorithme suivant.

Chaque matin, chaque garçon invite à dîner la fille qu'il préfère parmi celles qui ne lui ont pas déjà refusé une invitation. Chaque fille qui a été invitée dîne le soir avec le garçon qu'elle préfère parmi ceux qui l'ont invitée ce jour là.

L'algorithme se poursuit tant qu'au moins une invitation change. Les couples qui ont dîné entre eux le dernier soir sont mariés.

1. Montrer que cet algorithme termine.

2. Montre que toute fille qui est invitée un soir à dîner est sûre de dîner le lendemain avec un garçon qui lui plaise au moins autant.

3. Montrer que cet algorithme fournit un mariage stable.

2.4 Dualité

2.4.1 Dualité faible

Soit un problème d'optimisation avec contraintes de type égalité, inégalité, et ensembliste :

$$\operatorname*{Min}_x f(x); \quad g(x) = 0; \quad h(x) \leqslant 0; \quad x \in X, \tag{P}$$

avec $X \neq \emptyset$, $f : X \to \mathbb{R}$, $g : X \to \mathbb{R}^p$ et $h : X \to \mathbb{R}^q$. On note l'ensemble des *points réalisables*, la *valeur* et l'ensemble des *solutions* du problème par

$$\begin{aligned} F(P) &:= \{x \in X;\ g(x) = 0;\ h(x) \leqslant 0\}; && (2.22)\\ \mathrm{val}(P) &:= \inf\{f(x);\ x \in F(P)\}; && (2.23)\\ S(P) &:= \{x \in F(P);\ f(x) = \mathrm{val}(P)\}. && (2.24) \end{aligned}$$

La notation $F(\cdot)$ fait référence au terme anglais "feasible". Le *lagrangien* associé à ce problème est

$$L(x, \lambda, \mu) := f(x) + \lambda \cdot g(x) + \mu \cdot h(x), \tag{2.25}$$

où $\lambda \in \mathbb{R}^p$ et $\mu \in \mathbb{R}^q_+$. On vérifie facilement que

$$\sup_{\lambda \in \mathbb{R}^p, \mu \in \mathbb{R}^q_+} L(x, \lambda, \mu) = \begin{cases} f(x) & \text{si } g(x) = 0 \text{ et } h(x) \leqslant 0, \\ +\infty & \text{sinon.} \end{cases}$$

Ceci permet la réécriture du problème (P) comme

$$\operatorname*{Min}_{x \in X} \sup_{\lambda \in \mathbb{R}^p, \mu \in \mathbb{R}^q_+} L(x, \lambda, \mu) \tag{P'}$$

Rappelons maintenant l'inégalité élémentaire suivante :

Proposition 2.41 (Dualité faible) *Si X et Λ sont des ensembles, et si L est une fonction à valeurs réelles définie sur $X \times \Lambda$, on a*

$$\sup_{\lambda \in \Lambda} \inf_{x \in X} L(x, \lambda) \leqslant \inf_{x \in X} \sup_{\lambda \in \Lambda} L(x, \lambda) \ .$$

Démonstration. Pour tout $(\bar{x}, \bar{\lambda}) \in X \times \Lambda$, on a $\inf_{x \in X} L(x, \bar{\lambda}) \leqslant L(\bar{x}, \bar{\lambda}) \leqslant \sup_{\lambda \in \Lambda} L(\bar{x}, \lambda)$, donc $\inf_{x \in X} L(x, \bar{\lambda}) \leqslant \sup_{\lambda \in \Lambda} L(\bar{x}, \lambda)$. Prenant le supremum en $\bar{\lambda}$ à gauche et l'infimum en $\bar{x}$ à droite, on obtient l'inégalité cherchée. ■

Appliquant cette proposition au lagrangien du problème (P), il vient

$$\sup_{\lambda\in\mathbb{R}^p,\mu\in\mathbb{R}^n_+}\inf_{x\in X} L(x,\lambda,\mu)\leqslant \mathrm{val}(P')=\mathrm{val}(P). \tag{2.26}$$

La minoration de la valeur ainsi obtenue s'interprète comme la valeur du problème suivant, appelé *problème dual*

$$\operatorname*{Max}_{\lambda\in\mathbb{R}^p,\mu\in\mathbb{R}^n_+} d(\lambda,\mu), \tag{D}$$

où le *critère dual* est

$$d(\lambda,\mu):=\inf_{x\in X} L(x,\lambda,\mu).$$

Remarque 2.42 Le critère dual étant un infimum de fonctions affines de (λ,μ), est donc concave. Notons que maximiser d revient à minimiser $-d$; le problème dual est essentiellement un problème de minimisation d'une fonction convexe. En particulier, $S(D)$ est convexe. •

Remarque 2.43 L'inégalité de dualité faible s'interprète en terme de *valeur d'information*. Imaginons deux joueurs, appelés "Min" et "Max". Min choisit $x\in X$, et Max choisit $\lambda\in\Lambda$. Min paie alors un montant $L(x,\lambda)$ à Max. Montrons que $\inf_{x\in X}\sup_{\lambda\in\Lambda} L(x,\lambda)$ représente le gain de Min, si Min choisit x en premier, et si Max choisit ensuite λ, en ayant observé x. En effet, connaissant x, Max choisit une action λ pour maximiser son paiement. Il paie donc $\sup_{\lambda\in\Lambda} L(x,\lambda)$. Le joueur Min va choisir une action x minimisant ce sup. Il recevra donc l'inf-sup (2.43). Par symétrie, le sup-inf représente le gain de Min s'il choisit x *après* avoir observé le λ choisi par Max. La différence entre les deux membres de l'inégalité de dualité faible mesure donc c'est-à-dire l'avantage à jouer en dernier, qui est la valeur de l'information. •

Exemple 2.44 Soit le programme linéaire sous *forme standard*

$$\mathrm{Min}\, c\cdot x;\quad Ax=b,\ \ x\geqslant 0, \tag{LP}$$

avec ici $X=\mathbb{R}^n$ et A matrice de taille $n\times p$. Le lagrangien (notant s le multiplicateur associé aux contraintes d'inégalités et écrivant les inégalités $-x\leqslant 0$) est

$$L(x,\lambda,s):=c\cdot x+\lambda\cdot(Ax-b)-s\cdot x=(c+A^\top\lambda-s)\cdot x-b\cdot\lambda. \tag{2.27}$$

On a $\inf_x L(x,\lambda,s)=-b\cdot\lambda$ si $c+A^\top\lambda-s=0$, et $-\infty$ sinon. Comme le problème dual est un problème de maximisation, il s'écrit

$$\operatorname*{Max}_{\lambda\in\mathbb{R}^p,s\in\mathbb{R}^+_n} -b\cdot\lambda;\quad c+A^\top\lambda=s. \tag{LD}$$

Remarque 2.45 On peut choisir de dualiser (LP) en prenant $X=\mathbb{R}^n_+$. On parle alors de *dualisation partielle*. Le lagrangien a pour expression $L(x,\lambda,s):=c\cdot x+\lambda\cdot(Ax-b)$, et $\inf_{x\geqslant 0} L(x,\lambda,s)=-b\cdot\lambda$ si $c+A^\top\lambda\geqslant 0$, et $-\infty$ sinon. Le nouveau dual est donc

$$\operatorname*{Max}_{\lambda\in\mathbb{R}^p} -b\cdot\lambda;\quad c+A^\top\lambda\geqslant 0.$$

C'est essentiellement une réécriture de (LD). La propriété est générale : la dualisation partielle de programmes linéaires conduit à une réécriture du problème dual. •

On appelle *saut de dualité* la quantité positive $\mathrm{val}(P)-\mathrm{val}(D)$ (qui peut valoir $+\infty$).

Proposition 2.46 (Optimalité approchée) *Soit $(x,\lambda,\mu)\in X\times\mathbb{R}^p\times\mathbb{R}^q_+$ tels que $L(\cdot,\lambda,\mu)$ atteint son minimum en x, et supposons que $x\in F(P)$.* (i) *Une majoration du saut de dualité est*

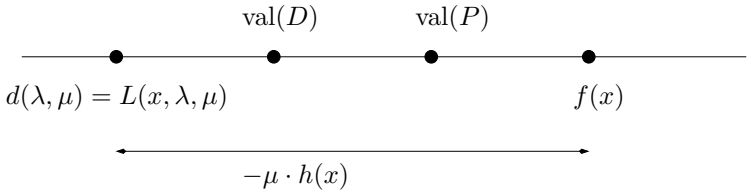
$$\mathrm{val}(P)-\mathrm{val}(D)\leqslant f(x)-L(x,\lambda,\mu)=-\mu\cdot h(x). \tag{2.28}$$

De plus on a la relation

$$f(x) \leqslant \mathrm{val}(D) - \mu \cdot h(x) \leqslant \mathrm{val}(P) - \mu \cdot h(x). \tag{2.29}$$

(ii) *La* condition de complémentarité $\mu \cdot h(x) = 0$ *est satisfaite ssi* $\mathrm{val}(P) = \mathrm{val}(D)$, $x \in S(P)$, *et* $(\lambda, \mu) \in S(D)$.

Le dessin schématique suivant permet de visualiser cette propriété et sa preuve.



Démonstration. (i) On a

$$L(x, \lambda, \mu) = d(\lambda, \mu) \leqslant \mathrm{val}(D) \leqslant \mathrm{val}(P) \leqslant f(x), \tag{2.30}$$

donc $\mathrm{val}(P) - \mathrm{val}(D) \leqslant f(x) - L(x, \lambda, \mu)$; or $g(x) = 0$, d'où (2.28). On a aussi $f(x) = L(x, \lambda, \mu) - \mu \cdot h(x) \leqslant \mathrm{val}(D) - \mu \cdot h(x)$, d'où (2.29) en tenant compte de $\mathrm{val}(D) \leqslant \mathrm{val}(P)$.
(ii) Si la condition de complémentarité est satisfaite, on déduit de (2.28) que $\mathrm{val}(P) - \mathrm{val}(D) = f(x) - L(x, \lambda, \mu) = 0$. D'après (2.29), $f(x) \leqslant \mathrm{val}(P)$, donc $x \in S(P)$, et de même pour $(\lambda, \mu) \in S(D)$. La réciproque se déduit de (2.28). ■

Appliquons cette proposition aux programmes linéaires, reprenant les notations de l'exemple 2.44.

Corollaire 2.47 *Soient $x \in F(LP)$ et $(\lambda, s) \in F(LD)$. Alors $c \cdot x \leqslant \mathrm{val}(P) + x \cdot s$. En outre, $x \cdot s = 0$ si et seulement si $x \in S(LP)$ et $(\lambda, s) \in S(LD)$.*

Démonstration. On applique la proposition 2.46, en notant que pour tout $x \in F(LP)$ et $(\lambda, s) \in F(LD)$, le lagrangien $L(\cdot, \lambda, s)$ atteint son minimum en x. ■

Au problème de maximisation

$$\operatorname*{Max}_x f'(x'); \quad g'(x') = 0; \quad h'(x') \leqslant 0; \quad x' \in X', \tag{P'}$$

on associe le lagrangien $L(x', \lambda', \mu') := f'(x') + \lambda' \cdot g'(x') - \mu' \cdot h'(x)'$. Observons alors que (P') peut se réécrire sous la forme $\mathrm{Max}_x \inf_{\{\lambda'; \mu' \geqslant 0\}} L'(x', \lambda', \mu')$. On peut associer un problème dual en inversant les ordre de maxi-minimisation :

$$\operatorname*{Min}_{\lambda'; \mu' \geqslant 0} \sup_x L'(x', \lambda', \mu') \tag{D'}$$

En particulier, si un problème dual a une expression de type (P'), son dual est appelé problème bidual. Nous vérifierons dans le lemme 2.79 qu'un programme linéaire coïncide avec son bidual.

2.4.2 Dualité forte en programmation linéaire

Lemme 2.48 (Farkas) *Soient c et $a^1, \ldots, a^p$ dans $\mathbb{R}^n$, vérifiant $c \cdot x \geqslant 0$, pour tout $x \in \mathbb{R}^n$, tel que $a^i \cdot x = 0$, $i \leqslant q$, et $a^i \cdot x \geqslant 0$, $q+1 \leqslant i \leqslant p$. Alors il existe $\lambda \in \mathbb{R}^p$ tel que $c = \sum_{i=1}^p \lambda_i a^i$, et de plus $\lambda \geqslant 0$, $i = q+1$ à p.*

Démonstration. L'ensemble $M := \{\sum_i \lambda_i a^i;\ \lambda \geqslant 0,\ i = q+1, \ldots, p\}$ est fermé (lemme 2.16). Il faut montrer que $c \in M$. Dans le cas contraire, le corollaire 2.4 affirme qu'il existe $x \in \mathbb{R}^n$ séparant strictement c et M, soit $c \cdot x < d \cdot x$, pour tout $d \in M$. En prenant tout d'abord $d = 0$, il vient $c \cdot x < 0$. En prenant $d = \pm\gamma a^i$, $i \leqslant q$, et $d = \gamma a^i$, $q+1 \leqslant i \leqslant p$, avec $\gamma \geqslant 0$, le minimum en γ de $d \cdot x$ étant minoré par $c \cdot x$, il vient $a^i \cdot x = 0$, $i \leqslant q$, et $a^i \cdot x \geqslant 0$, $q+1 \leqslant i \leqslant p$, en contradiction avec l'hypothèse. ■

Rappelons qu'on parle de dualité forte si les problèmes primal et dual ont même valeur. Nous reprenons les notations de l'exemple 2.44.

Théorème 2.49 (Dualité forte) (i) *Si le problème primal (LP) admet au moins un point réalisable, il a la même valeur que le problème dual (LD).*
(ii) *Si $x \in F(LP)$ et $(\lambda, s) \in F(LD)$, alors $x \in S(LP)$ et $(\lambda, s) \in S(LD)$ ssi $x \cdot s = 0$.*

Démonstration. (i) D'après la proposition 2.41, $\text{val}(LD) \leqslant \text{val}(LP)$; si $\text{val}(LP) = -\infty$ la conclusion est satisfaite. Supposons donc $\text{val}(LP)$ finie. Nous allons établir la conclusion en montrant que, pour tout $d < \text{val}(LP)$, il existe $(\lambda, s) \in F(LD)$ tel que $-b \cdot \lambda > d$, et donc $\text{val}(LD) > d$.

En effet, il est impossible de satisfaire $Ax = b$ et $c \cdot x = d$ quand $x \geqslant 0$. On peut donc séparer strictement le point (b, d) du convexe $M := \{(Ax, c \cdot x);\, x \in \mathbb{R}^n_+\}$ (fermé d'après le lemme 2.16). Il existe $(\lambda, \alpha) \in \mathbb{R}^p \times \mathbb{R}$ tel que

$$b \cdot \lambda + \alpha d < \lambda \cdot Ax + \alpha c \cdot x, \text{ pour tout } x \in \mathbb{R}^n_+. \tag{2.31}$$

Prenant $x^* \in F(LP)$ (non vide par hypothèse), il vient $\alpha(d - c \cdot x^*) < 0$; comme $d < \text{val}(LP) \leqslant c \cdot x^*$, on a $\alpha > 0$. Divisant λ par α si nécessaire, on peut supposer que $\alpha = 1$. Choisissant $x \geqslant 0$ arbitraire, et posant $s := c + A^\top \lambda$, déduisons de (2.31) que $s \geqslant 0$, donc $(\lambda, s) \in F(LD)$. Enfin prenant $x = 0$ dans (2.31), il vient $d < -b \cdot \lambda$ comme il fallait le montrer.
(ii) C'est une conséquence du point (i) combiné à la proposition 2.46. ■

Exemple 2.50 Soit le problème linéaire $\text{Min}_x\{-x;\ 0 \times x = -1\ ;\ x \in \mathbb{R}_+\}$. Ce problème n'est pas réalisable, et son dual $\text{Max}_\lambda\{\lambda\ ;\ -1 + 0 \times \lambda \geqslant 0\}$ ne l'est pas non plus. On est dans la situation où $-\infty = \text{val}(LD) < \text{val}(LP) = \infty$. Cette dernière situation est le seul cas dans lequel les valeurs primale et duale diffèrent en programmation linéaire.

Exemple 2.51 Les résultats précédents ne sont pas liés à la forme standard du problème (LP). Considérons le format général (de maximisation)

$$\text{Max}\{c \cdot x \mid\ Ax \leqslant b, Cx = d, x \geqslant 0\}\ , \tag{2.32}$$

avec $C \in \mathbb{R}^{p \times m}$, $d \in \mathbb{R}^p$ (et toujours $A \in \mathbb{R}^{n \times m}$, $b \in \mathbb{R}^n$, et $c \in \mathbb{R}^m$). Outre le multiplicateur $\lambda \in \mathbb{R}^n_+$ correspondant à la contrainte $Ax \leqslant b$, on introduit un multiplicateur non signé, $\mu \in \mathbb{R}^p$, correspondant à la contrainte $Cx = d$, un multiplicateur $\nu \in \mathbb{R}^m_+$, correspondant à la contrainte $x \geqslant 0$, ainsi que le lagrangien :

$$L(x; \lambda, \mu, \nu) = c \cdot x + \lambda \cdot (b - Ax) + \mu \cdot (d - Cx) + \nu \cdot x\ ,$$

de sorte que (2.32) s'écrit :

$$\sup_{x \in \mathbb{R}^m} \inf_{\lambda \in \mathbb{R}^n_+,\ \mu \in \mathbb{R}^p,\ \nu \in \mathbb{R}^m_+} L(x; \lambda, \mu, \nu)\ . \tag{2.33}$$

Or

$$\begin{aligned} \sup_{x \in \mathbb{R}^m} L(x; \lambda, \mu, \nu) &= \sup_{x \in \mathbb{R}^m} \lambda \cdot b + \mu \cdot d + (c^\top - \lambda^\top A - \mu^\top C + \nu^\top)x \\ &= \begin{cases} \lambda \cdot b + \mu \cdot d & \text{si } c^\top - \lambda^\top A - \mu^\top C + \nu^\top = 0 \\ +\infty & \text{sinon.} \end{cases} \end{aligned}$$

Le problème dual, obtenu en commutant le sup et l'inf, dans (2.33), est donc :

$$\inf\{\lambda^\top b + \mu^\top d \mid\ \lambda^\top A + \mu^\top C \geqslant c^\top,\ \lambda \geqslant 0\}\ .$$

En guise d'exercice, le lecteur pourra montrer que si le primal (2.32) admet au moins un point réalisable, alors le problème dual a même valeur, et qu'un point réalisable x^* du problème primal est optimal si et seulement si il existe un point réalisable (λ^*, μ^*) du problème dual tel que :

$$\lambda^* \cdot (b - Ax^*) = 0 \text{ et } (\lambda^\top A + \mu^\top C - c^\top)x^* = 0\ .$$

Exercice 2.52 (Théorèmes de König-Egerváry et de König-Frobenius) Cet exercice fait suite à l'exercice 2.39 sur le théorème de Birkhoff, dont il reprend les notations.

1. Montrer par un argument de dualité que

$$\bar{w} = \min_{\substack{y \in \mathbb{R}^n,\, z \in \mathbb{R}^n \\ \forall i,j,\ y_i + z_j \geqslant w_{ij}}} \sum_{1 \leqslant i \leqslant n} y_i + \sum_{1 \leqslant j \leqslant n} z_j \tag{2.34}$$

On s'intéresse maintenant au cas où $w_{ij} \in \{0,1\}$. On note $\mathcal{G}$ le graphe formé des arcs (i,j') tels que $w_{ij} = 1$.

2. Montrer que le problème (2.21) revient alors à trouver un couplage de cardinalité maximum dans $\mathcal{G}$.

3. Montrer que le problème dual (2.34) admet une solution (y,z) entière.

4. Montrer même (preuve plus astucieuse) que ce problème a une solution $y, z \in \{0,1\}^n$.

5. On dit qu'un ensemble S de sommets couvre le graphe $\mathcal{G}$ si chaque arc de $\mathcal{G}$ contient au moins un sommet de S. En déduire très simplement que le cardinal maximal d'un couplage dans $\mathcal{G}$ est égal au cardinal minimal d'un ensemble de sommets couvrant $\mathcal{G}$ (c'est le théorème de König-Egerváry).

6. Déduire le cas particulier suivant (Théorème de König-Frobenius) : pour une matrice A de taille $n \times n$, on a

$$\prod_{1 \leqslant i \leqslant n} A_{i\sigma(i)} = 0,\ \forall \sigma \in \mathfrak{S}_n\ ,$$

si et seulement si il existe un ensemble de lignes I de cardinal p, et un ensemble de colonnes J de cardinal q, tel que $p + q = n + 1$ et $A_{ij} = 0,\ \forall i \in I, \forall j \in J$. Ce résultat généralise (de manière optimale) le fait qu'une matrice qui a une ligne nulle ou une colonne nulle a un déterminant nul !

7. Comment traiter le cas où le nombre de filles et de garçons diffèrent.

L'exercice suivant illustre des techniques générales très utiles qui permettent de représenter comme des programmes linéaires des problèmes d'optimisation dont le critère est convexe affine par morceaux.

Exercice 2.53 (Solutions de systèmes linéaires sous-déterminés : saurez-vous reconnaître les programmes linéaires ?)

On considère le système linéaire

$$Ax = b$$

avec $A \in \mathbb{R}^{p \times n}$, $b \in \mathbb{R}^p$, et $p < n$ (cas sous-déterminé). Afin de sélectionner une solution particulière intéressante de $Ax = b$, on se propose de considérer les différents problèmes d'optimisation :

$$\text{Min} \sum_{1 \leqslant i \leqslant n} |x_i|, \qquad x \in \mathbb{R}^n,\ Ax = b\ ; \tag{2.35}$$

$$\text{Min} \sum_{1 \leqslant i \leqslant n} x_i^2, \qquad x \in \mathbb{R}^n,\ Ax = b\ ; \tag{2.36}$$

$$\text{Min} \max_{1 \leqslant i \leqslant n} |x_i|, \qquad x \in \mathbb{R}^n,\ Ax = b\ ; \tag{2.37}$$

$$\text{Min}\, |N(x)|, \qquad x \in \mathbb{R}^n,\ Ax = b \tag{2.38}$$

où $N(x) := |\{i \mid 1 \leqslant i \leqslant n,\ x_i \neq 0\}|$ désigne le nombre de coordonnées non-nulles de x (rappelons qu'on désigne par $|Y|$ le nombre d'éléments d'un ensemble Y) ;

$$\text{Min}\, (\max_{1 \leqslant i \leqslant n} x_i) - (\min_{1 \leqslant j \leqslant n} x_j), \qquad x \in \mathbb{R}^n,\ Ax = b\ ; \tag{2.39}$$

$$\text{Min } |x|^{[1]} + \cdots + |x|^{[k]}, \qquad x \in \mathbb{R}^n,\ Ax = b\ ; \tag{2.40}$$

où $k \leqslant n$ est fixé, et où pour tout $1 \leqslant q \leqslant n$, $|x|^{[q]}$ désigne le q-ième plus grand nombre de la suite $|x_1|, \ldots, |x_n|$ (par exemple, si $x = (-1, 0, 1, 7)$, $|x|^{[1]} = 7$, $|x|^{[2]} = |x|^{[3]} = 1$, $x^{[4]} = 0$).

Pour chacun de ces problèmes, dire si oui ou non il peut être résolu par la programmation linéaire, et si oui, expliciter le programme linéaire. (Donner le cas échéant l'interprétation pratique du problème.)

2.4.3 Dualité forte et perturbation canonique

Considérons maintenant une famille de problèmes avec paramétrisation additive des contraintes :

$$\underset{x}{\text{Min}}\, f(x); \quad g(x) + \delta^1 = 0; \quad h(x) + \delta^2 \leqslant 0; \quad x \in X, \tag{P_δ}$$

avec $\delta = (\delta^1, \delta^2) \in \mathbb{R}^p \times \mathbb{R}^q$. On note la fonction valeur associée

$$v(\delta) := \inf_{x \in X} \{f(x);\ g(x) + \delta^1 = 0; \quad h(x) + \delta^2 \leqslant 0\}, \tag{2.41}$$

et soit $v_d(\delta)$ la valeur duale associée :

$$v^d(\delta) := \sup_{(\lambda,\mu);\mu \geqslant 0} \inf_x L_\delta(x, \lambda, \mu) \tag{2.42}$$

avec le lagrangien

$$L_\delta(x, \lambda, \mu) := f(x) + \lambda \cdot (g(x) + \delta^1) + \mu \cdot (h(x) + \delta^2). \tag{2.43}$$

Pour le problème "non perturbé" obtenu quand $\delta = 0$, on retrouve le problème (P) et son dual (D) de la section 2.4.1.

Définition 2.54 Soit $f : \mathbb{R}^n \to \bar{\mathbb{R}}$, de valeur finie en $\bar{x}$. On appelle *dérivée inférieure (supérieure) de Dini* de f en $\bar{x}$ dans la direction $h \in \mathbb{R}^n$ les quantités suivantes, finies ou non :

$$f'_-(\bar{x}, h) := \liminf_{t \downarrow 0} \frac{f(\bar{x} + th) - f(\bar{x})}{t}; \quad f'_+(\bar{x}, h) := \limsup_{t \downarrow 0} \frac{f(\bar{x} + th) - f(\bar{x})}{t}. \tag{2.44}$$

Proposition 2.55 *Supposons les problèmes (P) et (D) de même valeur (dualité forte) finie. Alors pour tout $\delta \in \mathbb{R}^{p+q}$, on a*

$$\frac{v^d(t\delta) - v^d(0)}{t} \geqslant \bar{\lambda} \cdot \delta^1 + \bar{\mu} \cdot \delta^2, \tag{2.45}$$

$$v'_-(0, \delta) \geqslant (v^d)'_-(0, \delta) \geqslant \sup_{(\lambda,\mu) \in S(D)} \{\lambda \cdot \delta^1 + \mu \cdot \delta^2\}. \tag{2.46}$$

Démonstration. Fixons $(\bar{\lambda}, \bar{\mu}) \in S(D)$. Alors

$$v^d(\delta) \geqslant \inf_x L_\delta(x, \bar{\lambda}, \bar{\mu}) = \inf_x L_0(x, \bar{\lambda}, \bar{\mu}) + \bar{\lambda} \cdot \delta^1 + \bar{\mu} \cdot \delta^2 = v(0) + \bar{\lambda} \cdot \delta^1 + \bar{\mu} \cdot \delta^2, \tag{2.47}$$

qui puisque $v(0) = v^d(0)$ équivaut à (2.45). La première inégalité de (2.46) est conséquence de $v(t\delta) \geqslant v^d(t\delta)$ pour tout $t \geqslant 0$, et $v(0) = v^d(0)$. Passant à la limite dans (2.45) en $t \downarrow 0$ puis maximisant par rapport à $(\bar{\lambda}, \bar{\mu}) \in S(D)$, on obtient la seconde inégalité.

■

Remarque 2.56 Dans le cas où v a une dérivée notée w en zéro, sous les hypothèses de la proposition, on a donc $w \cdot \delta \geqslant \lambda \cdot \delta^1 + \mu \cdot \delta^2$, pour tout $(\lambda, \mu) \in S(D)$ et $\delta \in \mathbb{R}^p \times \mathbb{R}^q$, ce qui implique $S(D) = \{w\}$. Ainsi le multiplicateur de Lagrange (nécessairement unique !) s'interprète (sous ces hypothèses) comme la dérivée de la valeur du problème (on parle alors de valeur marginale) par rapport à une perturbation additive des contraintes. •

On établira dans la section suivante des résultats plus précis pour des problèmes convexes.

2.5 Calcul sous-différentiel

2.5.1 Notion de sous-différentiel

Notons $\bar{\mathbb{R}} := \mathbb{R} \cup \{-\infty\} \cup \{+\infty\}$. Soit $f : \mathbb{R}^n \to \bar{\mathbb{R}}$. Une fonction $g : \mathbb{R}^n \to \bar{\mathbb{R}}$ est dite *minorante* de f si $g(x) \leqslant f(x)$ pour tout x, et *minorante exacte*, au point $\bar{x} \in \mathbb{R}^n$, si de plus $g(\bar{x}) = f(\bar{x})$. Si g est affine on parlera de *minorante affine*. Soit $\bar{x} \in \mathbb{R}^n$ tel que $f(\bar{x})$ est fini. On appelle *sous-différentiel* de f en $\bar{x}$ l'ensemble des pentes de minorantes affines exactes de f en ce point :

$$\partial f(\bar{x}) := \{q \in \mathbb{R}^n;\ f(x) \geqslant f(\bar{x}) + q \cdot (x - \bar{x}),\ \text{pour tout } x \in \mathbb{R}^n\}. \tag{2.48}$$

Les propriétés suivantes sont immédiates : le sous-différentiel est un ensemble convexe fermé, et $0 \in \partial f(\bar{x})$ ssi f atteint son minimum en $\bar{x}$. Donnons quelques exemples.

Exemple 2.57 Si f a un gradient $\nabla f(\bar{x})$ au point $\bar{x}$, on vérifie facilement que $\partial f(\bar{x})$ est soit vide, soit réduit à $\{\nabla f(\bar{x})\}$. Si f est convexe et différentiable en $\bar{x}$, on sait que l'inégalité de (2.48) est satisfaite pour $q = \nabla f(\bar{x})$, et donc $\partial f(\bar{x}) = \{\nabla f(\bar{x})\}$.

Si par exemple $f(x) = c \cdot x + \frac{1}{2} x \cdot Hx$, avec $c \in \mathbb{R}^n$ et H matrice symétrique, semi définie positive de taille n, alors f est convexe, et $\partial f(x) = \{c + Hx\}$, pour tout $x \in \mathbb{R}^n$.

On note $\Gamma(\mathbb{R}^n)$ la classe des suprema de fonctions affines sur $\mathbb{R}^n$, autrement dit des fonctions de la forme : $\sup\{a \cdot x + b;\ (a, b) \in E\}$, avec $E \subset \mathbb{R}^n \times \mathbb{R}$. Rappelons que l'*épigraphe* de $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ est l'ensemble des points "au dessus du graphe" (graphe compris) :

$$\text{épi}(f) := \{(x, \alpha) \in \mathbb{R}^n \times \mathbb{R};\ \alpha \geqslant f(x)\}. \tag{2.49}$$

L'épigraphe de $f \in \Gamma(\mathbb{R}^n)$ est l'intersection des épigraphes de ces minorantes, qui sont des demi-espaces fermés ; donc épi(f) est convexe et fermé.

Exemple 2.58 (Entropie) Soit l'entropie $h : \mathbb{R} \to \bar{\mathbb{R}} \cup \{\infty\}$, définie par $h(x) = +\infty$ si $x < 0$, $h(0) = 0$, $h(x) = x \log x$ si $x > 0$. On peut vérifier que $h \in \Gamma(\mathbb{R})$. En 0 elle a une valeur finie, mais un sous différentiel vide. Un supremum de fonctions affines n'est donc pas nécessairement sous différentiable en tout point où il a une valeur finie.

Explicitons le sous-différentiel de la fonction maximum.

Lemme 2.59 *La fonction $f : \mathbb{R}^n \to \mathbb{R}$, $f(x) := \max\{x_i,\ i = 1, \ldots, n\}$ est convexe. Notons $I(x) := \{1 \leqslant i \leqslant n;\ x_i = f(x)\}$, et e^i les éléments de la base naturelle de $\mathbb{R}^n$. Alors*

$$\partial f(x) = \text{conv}\{e^i,\ i \in I(x)\}. \tag{2.50}$$

Démonstration. Un maximum de fonctions affines étant convexe, f l'est. Si $i \in I(x)$, de $f(x) \geqslant x_i$ on déduit que $e^i \in \partial f(x)$. Etant convexe, $\partial f(x)$ contient aussi $\text{conv}\{e^i,\ i \in I(x)\}$.

Soit $q \in \partial f(x)$. Si $i \notin I(x)$, et $\varepsilon > 0$ petit, prenant $x' = x + \varepsilon q_i e_i$, il vient $f(x) = f(x') \geqslant f(x) + \varepsilon (q_i)^2$, donc $q_i = 0$. Si $i \in I(x)$, prenant $x' = x - e_i$, il vient $f(x) \geqslant f(x') \geqslant f(x) - q_i$, donc $q_i \geqslant 0$. Enfin choisissant $x' = x + \gamma \mathbf{1}$ ($\mathbf{1}$ vecteur formé de uns, $\gamma \in \mathbb{R}$ quelconque), il vient $f(x) + \gamma = f(x') \geqslant f(x) + \gamma q \cdot \mathbf{1}$, soit $\gamma \geqslant \gamma q \cdot \mathbf{1}$. Comme γ est arbitraire, ceci prouve $q \cdot \mathbf{1} = 1$, donc $q \in \text{conv}\{e^i,\ i \in I(x)\}$, d'où la conclusion. ■

Le résultat précédent peut être vu comme un cas particulier du lemme suivant :

Lemme 2.60 *Soit f convexe, à valeurs finie, positivement homogène : $f(tx) = tf(x)$, pour tout $t \geqslant 0$, et donc $f(0) = 0$. Alors*

$$\partial f(x) = \{q \in \partial f(0);\quad f(x) = q \cdot x\}. \tag{2.51}$$

Démonstration. Soit $q \in \partial f(x)$. Pour tout $\gamma > 0$ et $y \in \mathbb{R}^n$, on a

$$\gamma f(y) = f(\gamma y) \geqslant f(x) + q \cdot (\gamma y - x). \tag{2.52}$$

Divisant par $\gamma \uparrow \infty$, il vient $f(y) \geqslant q \cdot y$, et donc $q \in \partial f(0)$. Par ailleurs, prenant $y = 0$ il vient $0 \geqslant f(x) - q \cdot x$. Comme $q \in \partial f(0)$ implique l'inégalité inverse, ceci prouve $\partial f(x) \subset A$, où $A := \{q \in \partial f(0);\ q \cdot x = f(x)\}$. Réciproquement, si $q \in A$, alors $f(x) + q \cdot (y - x) = q \cdot y \leqslant f(y)$, et donc $q \in \partial f(x)$ ce qui achève la démonstration. ■

Montrons maintenant l'utilité des théorèmes de séparation pour le calcul de sous différentiels.

Lemme 2.61 *Soit Ω une partie compacte (fermée et bornée) de $\mathbb{R}^n$. Alors la fonction $f(x) := \max\{a \cdot x;\ a \in \Omega\}$ (le max est atteint) est convexe, à valeurs finie, positivement homogène, et de plus*

$$\text{(i)}\ \ \partial f(0) = \mathrm{conv}(\Omega); \qquad \text{(ii)}\ \partial f(x) = \mathrm{conv}\{a \in \Omega;\ a \cdot x = f(x)\}. \tag{2.53}$$

Démonstration. (i) Soit $a \in \Omega$; on a $f(x) \geqslant a \cdot x$, donc $a \in \partial f(0)$. Etant convexe, $\partial f(0)$ contient aussi $\mathrm{conv}(\Omega)$.

Soit $q \notin \mathrm{conv}(\Omega)$. La compacité de Ω permet de vérifier que $\mathrm{conv}(\Omega)$ est fermé. Le corollaire 2.3 assure donc l'existence de $y \in \mathbb{R}^n$ séparant strictement q de $\mathrm{conv}(\Omega)$, soit $q \cdot y > \max\{a \cdot y;\ a \in \Omega\}$. Autrement dit, $q \cdot y > f(y)$, donc $q \notin \partial f(0)$. Nous avons montré que $\mathrm{conv}(\Omega) = \partial f(0)$.
(ii) Le lemme 2.60 implique que $\partial f(x) = \{a \in \mathrm{conv}(\Omega);\ a \cdot x = f(x)\}$. Comme $a \cdot x \leqslant f(x)$ pour tout $a \in \Omega$, les combinaisons convexes d'éléments de Ω dans $\partial f(x)$ coïncident avec les combinaisons convexes de $\{a \in \Omega;\ a \cdot x = f(x)\}$, d'où la conclusion. ■

Passons maintenant à un cas plus général.

Lemme 2.62 *Soit Ω un compact de $\mathbb{R}^n \times \mathbb{R}$. Notons $f(x) := \max\{a \cdot x + b;\ (a,b) \in \Omega\}$ (le max est atteint), et soit $I(x) := \{(a,b) \in \Omega;\ a \cdot x + b = f(x)\}$ l'ensemble de contact. Alors, pour tout $\bar{x} \in \mathbb{R}^n$, on a*

$$\partial f(\bar{x}) = \mathrm{conv}\{a;\ (a,b) \in I(\bar{x})\}. \tag{2.54}$$

Démonstration. Si $(a,b) \in I(\bar{x})$, alors $f(x') \geqslant a \cdot x' + b = f(\bar{x}) + a \cdot (x' - \bar{x})$ ce qui prouve que $\partial f(x)$ contient a, et donc contient aussi $A := \mathrm{conv}\{a;\ (a,b) \in I(\bar{x})\}$. La compacité de Ω entraîne celle de A.

Soit maintenant $q \notin A$. Le corollaire 2.3 assure l'existence de $y \in \mathbb{R}^n$ et $\varepsilon > 0$ tels que $q \cdot y > \varepsilon + \max\{a \cdot y;\ a \in A\}$. Soit $x^n = \bar{x} + t_n y$, avec $t_n \downarrow 0$. Il existe $(a_n, b_n) \in \Omega$ tel que $f(x^n) = a_n \cdot x^n + b_n$. Extrayant si nécessaire une sous-suite, on peut supposer que $(a_n, b_n) \to (\bar{a}, \bar{b}) \in \Omega$. De $(a_n, b_n) \in I(x^n)$, on déduit que $(\bar{a}, \bar{b}) \in I(x)$. La relation

$$f(x^n) = t_n a_n \cdot y + a_n \cdot \bar{x} + b_n \leqslant t_n a_n \cdot y + f(\bar{x}) = f(\bar{x}) + t_n\, \bar{a} \cdot y + o(t_n) \tag{2.55}$$

implique $f(x^n) \leqslant f(\bar{x}) + t_n q \cdot y - \varepsilon t_n + o(t_n)$, et donc $\lim_n t_n^{-1}(f(\bar{x} + t_n y) - f(\bar{x})) < q \cdot y$, qui assure que $q \notin \partial f(\bar{x})$. ■

2.5.2 Calcul sous-différentiel

Comme pour le calcul différentiel, il existe un calcul sous-différentiel permettant de déduire (dans certains cas) le sous-différentiel de fonctions composées à partir du sous-différentiel des éléments qui la composent. La section précédente en donne des exemples dans le cas du maximum. Donnons des exemples d'inclusions obtenues facilement pour des classes de fonctions très générales.

On appelle *domaine* de $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ l'ensemble

$$\mathrm{dom}(f) := \{x \in \mathbb{R}^n;\ f(x) < +\infty\}. \tag{2.56}$$

Cet ensemble est convexe si f l'est.

Lemme 2.63 (i) *Soient f_1 et f_2 deux fonctions convexes $\mathbb{R}^n \to \bar{\mathbb{R}}$, de somme $F = f_1 + f_2$ définie par (noter que $\mathrm{dom}(F) = \mathrm{dom}(f_1) \cap \mathrm{dom}(f_2)$)*

$$F(x) = \begin{cases} +\infty \text{ si } f_1(x) = +\infty \text{ ou } f_2(x) = +\infty, \\ f_1(x) + f_2(x) \text{ sinon.} \end{cases} \tag{2.57}$$

Si f_1 et f_2 ont une valeur finie en $\bar{x}$, alors $\partial f_1(\bar{x}) + \partial f_2(\bar{x}) \subset \partial F(\bar{x})$.
(ii) *Soient $A \in L(\mathbb{R}^n, \mathbb{R}^p)$ et $f : \mathbb{R}^p \to \bar{\mathbb{R}}$. Posons $F(x) := f(Ax)$. Si F a une valeur finie en $\bar{x}$, alors*

$$A^\top \partial f(A\bar{x}) \subset \partial F(\bar{x}). \tag{2.58}$$

(iii) *Soit $F(x) := \sup_{i \in I}\{f_i(x)\}$, où $f_i : \mathbb{R}^n \to \bar{\mathbb{R}}$, pour tout $i \in I$. Soient $\bar{x} \in \mathbb{R}^n$ tel que $F(\bar{x})$ est fini. Notons $J(x) := \{i \in I;\, f_i(\bar{x}) = F(\bar{x})\}$. Alors (rappelons que $\overline{\mathrm{conv}}$ désigne la fermeture de l'enveloppe convexe)*

$$\overline{\mathrm{conv}}\{\partial f_j(\bar{x});\ j \in J(\bar{x})\} \subset \partial F(\bar{x}). \tag{2.59}$$

Démonstration. La démonstration (facile) est laissée au lecteur. Notons que pour le point (iii) on vérifie d'abord $\partial f_j(\bar{x}) \in \partial F(\bar{x})$, pour tout $j \in J(\bar{x})$; comme le sous différentiel est convexe fermé, ceci implique (2.59). ■

Ces inclusions sont précieuses dans la mise en œuvre des algorithmes de sous gradient (voir la section 5.2.3 consacrée à la méthode de sur-gradients pour maximiser un critère, ce qui revient au même) : pour minimiser une fonction convexe f, ces algorithmes demandent en effet de calculer un sous gradient de f en chaque point où f est finie.

En revanche, lorsqu'on étudie les conditions d'optimalité, on part d'une relation du type $0 \in \partial f(\bar{x})$ (qui dit que f atteint son minimum en $\bar{x}$) et on voudrait expliciter cette relation en décomposant les éléments de $\partial f(\bar{x})$. On est donc amené à chercher sous quelles conditions les inclusions inverses de celles du lemme 2.63 sont valables. Notons que le lemme 2.62 donne une réponse dans le cas d'un maximum "compact" de fonctions affines.

Le résultat le plus important concerne la composition par une application linéaire :

Lemme 2.64 *Soient $A \in L(\mathbb{R}^n, \mathbb{R}^p)$ et f convexe : $\mathbb{R}^p \to \bar{\mathbb{R}}$. Posons $F(x) := f(Ax)$. Si la condition de stabilité*

$$\text{Il existe } \varepsilon > 0; \quad \varepsilon B \subset \mathrm{dom}(f) - \mathrm{Im}\, A \tag{2.60}$$

est satisfaite, alors pour tout $\bar{x}$ tel que $F(\bar{x})$ est finie, on a $\partial F(\bar{x}) = A^\top \partial f(A\bar{x})$.

Remarque 2.65 Dans (2.60), B est la boule euclidienne ouverte. La condition de stabilité s'interprète ainsi : si on perturbe la fonction F en $F_b(x) := f(Ax + b)$, où $b \in \mathbb{R}^p$, alors $\mathrm{dom}(F_b)$ reste non vide si $\|b\|$ est assez petit. •

Démonstration. (du lemme 2.64) En raison du lemme 2.63(ii), il suffit de démontrer que si $q \in \partial F(\bar{x})$, alors $q \in A^\top \partial f(A\bar{x})$. Pour simplifier les notations on pourra supposer que $\bar{x} = 0$ et $f(0) = 0$. Montrons qu'on peut séparer $\mathrm{épi}(f)$ du sous-espace affine (de $\mathbb{R}^{p+1}$) :

$$H := \{(Ax, q \cdot x);\ x \in \mathbb{R}^n\}. \tag{2.61}$$

D'après le lemme 2.2, il suffit de montrer que $0 \notin \mathrm{int}(\mathrm{épi}(f) - H)$. Dans le cas contraire, on pourrait trouver $(x, y) \in \mathbb{R}^n \times \mathbb{R}^p$ tels que $Ax = y$ et $f(y) - q \cdot x < 0$, donc $F(x) = f(y) < q \cdot x$, ce qui contredit $q \in \partial F(0)$.

Il existe donc $(w, \alpha) \in \mathbb{R}^p \times \mathbb{R}$, non nul, séparant $\mathrm{épi}(f)$ et H :

$$(w, \alpha) \cdot (y, z) \leqslant (w, \alpha) \cdot (Ax, q \cdot x), \text{ pour tout } (x, y) \in \mathbb{R}^n \times \mathbb{R}^p \text{ et } z \geqslant f(y). \tag{2.62}$$

Prenant $x = \bar{x} = 0$ et faisant tendre z vers $+\infty$, on obtient $\alpha \leqslant 0$. Si $\alpha = 0$, alors $w \cdot (y - Ax) \leqslant 0$, pour tout $x \in \mathbb{R}^n$ et $y \in \mathrm{dom}(f)$; comme $w \neq 0$ si $\alpha = 0$, ceci contredit la condition de stabilité (2.60).

Donc $\alpha < 0$. Changeant w en $-w/\alpha$ si nécessaire on peut supposer que $\alpha = -1$. Alors (2.62) équivaut à $w \cdot (y - Ax) + q \cdot x \leqslant f(y)$, pour tout $(x, y) \in \mathbb{R}^n \times \mathbb{R}^p$. Ceci étant vrai pour tout $x \in \mathbb{R}^n$, il vient $q = A^\top w$, et il reste $w \cdot y \leqslant f(y)$, pour tout $y \in \mathbb{R}^p$, soit $w \in \partial f(0)$, et donc $q \in A^\top \partial f(0)$ comme il fallait le montrer. ■

Corollaire 2.66 *Soient f_1 et f_2 deux fonctions convexes $\mathbb{R}^n \to \bar{\mathbb{R}}$, de somme $F = f_1 + f_2$. Si la condition de stabilité suivante est satisfaite :*

$$\text{Il existe } \varepsilon > 0; \quad \varepsilon B \subset \mathrm{dom}(f_1) - \mathrm{dom}(f_2), \tag{2.63}$$

alors pour tout $\bar{x}$ tel que $F(\bar{x})$ est finie, on a $\partial F(\bar{x}) = \partial f_1(\bar{x}) + \partial f_2(\bar{x})$.

Démonstration. On sait par le lemme 2.63(i) que $\partial f_1(\bar{x}) + \partial f_2(\bar{x}) \subset \partial F(\bar{x})$. L'inclusion inverse se déduit du lemme 2.64, en posant $Ax = (x, x)$ et $f(y^1, y^2) := f_1(y^1) + f_2(y^2)$. On a bien $F(x) = f(Ax)$. Montrons que la condition de stabilité $\varepsilon B \subset \mathrm{dom}(f) - \mathrm{Im}(A)$ équivaut à (2.63). En effet, la condition de stabilité (2.60) est l'existence de $\varepsilon_1 > 0$ tel que, pour tout y, y' dans $\varepsilon_1 B$ (avec B boule unité de $\mathbb{R}^n$), il existe $y_1 \in \mathrm{dom}(f_1)$, $y_2 \in \mathrm{dom}(f_2)$, et $x \in \mathbb{R}^n$, tels que $y = y^1 - x$ et $y' - y^2 - x$. Autrement dit, pour tout y, y' dans $\varepsilon_1 B$, il existe $y_1 \in \mathrm{dom}(f_1)$ et $y_2 \in \mathrm{dom}(f_2)$ tels que $y - y' = y^1 - y^2$. Comme $y - y'$ parcourt $\varepsilon B - \varepsilon B = 2\varepsilon B$, ceci est conséquence de (2.63) avec $\varepsilon = 2\varepsilon_1$. Autrement dit (2.63) implique la condition de stabilité avec la constante $\frac{1}{2}\varepsilon$. La réciproque est immédiate, en prenant $y_2 = 0$. ■

Corollaire 2.67 *Soit $f : \mathbb{R}^n \times \mathbb{R}^p \to \bar{\mathbb{R}}$ convexe, de variables notées (x, y). Fixons $\bar{y} \in \mathbb{R}^p$, et définissons $F : \mathbb{R}^n \to \bar{\mathbb{R}}$ par $F(x) := f(x, \bar{y})$. On appelle* sous-différentiel partiel *de f par rapport à x, au point $(x, \bar{y})$, l'ensemble $\partial_x f(x, \bar{y}) := \partial F(x)$. Si la condition de stabilité suivante est satisfaite :*

$$\text{Il existe } \varepsilon > 0; \quad \varepsilon B \subset \mathrm{dom}(f) - \mathbb{R}^n \times \{\bar{y}\}, \tag{2.64}$$

alors le sous différentiel partiel est la restriction aux variables x du sous différentiel de f :

$$\partial_x f(x, \bar{y}) = \{q \in \mathbb{R}^n;\ \text{il existe } w \in \mathbb{R}^p;\ (q, w) \in \partial f(x, \bar{y})\}. \tag{2.65}$$

Démonstration. On applique le lemme 2.64 à la composition $\tilde{f} \circ A$, avec $A : \mathbb{R}^n \to \mathbb{R}^n \times \mathbb{R}^p$, $x \to (x, 0)$ et $\tilde{f}(z) = f(z + (0, \bar{y}))$. Noter que $\mathrm{Im}\, A = \mathbb{R}^n \times \{0\}$ et $\mathrm{dom}\, \tilde{f} = \mathrm{dom}\, f - (0, \bar{y})$. ■

Remarque 2.68 La condition (2.64) revient à dire que, pour tout y voisin de $\bar{y}$, il existe $x \in \mathbb{R}^n$ tel que $f(x, \bar{y}) < +\infty$. •

2.5.3 Continuité et dérivées directionnelles

Nous allons prouver que sur l'intérieur du domaine d'une fonction convexe, celle-ci est localement lipschitzienne, et admet des dérivées directionnelles qui peuvent se calculer à l'aide des sous-différentiels.

Définition 2.69 Une fonction $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ est dite majorée au voisinage de $\bar{x}$ s'il existe $\beta \in \mathbb{R}$ tel que $f(x) \leqslant \beta$, pour x voisin de $\bar{x}$.

Proposition 2.70 *Soit f convexe $\mathbb{R}^n \to \bar{\mathbb{R}}$, de valeur finie en $\bar{x} \in \mathrm{int}(\mathrm{dom}(f))$. Alors f est localement lipschitzienne au voisinage de $\bar{x}$.*

Démonstration. a) Notons d'abord que f est majorée au voisinage de $\bar{x}$. En effet, il existe $\{y^1, \ldots, y^{n+1}\}$ dans $\mathbb{R}^n$ tels que $\bar{x} \in \mathrm{int}(C)$, où $C := \mathrm{conv}(\{y^1, \ldots, y^{n+1}\}$. Changeant si nécessaire y^i en $\bar{x} + t(y^i - \bar{x})$, avec $t \in]0, 1[$ assez petit, on peut supposer que $y^i \in \mathrm{dom}(f)$ pour $i = 1$ à $n + 1$; par convexité de f, on a $\sup_C f \leqslant \max_i f(y^i)$, d'où le résultat.

b) Soient $r > 0$ et $a \in \mathbb{R}$ tels que $f(x) \leqslant a$, quand $x \in \bar{B}(x_0, r)$. Soient $\varepsilon \in]0,1[$ et $h \in X$, avec $\|h\| = \varepsilon r$. Alors $x_0 \pm \varepsilon^{-1}h \in \bar{B}(x_0, r)$, donc

$$\begin{aligned} f(x_0 + h) &\leqslant (1-\varepsilon)f(x_0) + \varepsilon f(x_0 + \varepsilon^{-1}h) \leqslant (1-\varepsilon)f(x_0) + \varepsilon a, \\ f(x_0 + h) &\geqslant (1+\varepsilon)f(x_0) - \varepsilon f(x_0 - \varepsilon^{-1}h) \geqslant (1+\varepsilon)f(x_0) - \varepsilon a, \end{aligned}$$

ce qui assure

$$|f(x_0 + h) - f(x_0)| \leqslant \varepsilon(a - f(x_0)) = r^{-1}(a - f(x_0))\|h\|, \quad \text{pour tout } \|h\| < r. \tag{2.66}$$

En particulier, f est continue en x_0. Réduisant r si nécessaire, on peut donc supposer que $f(x) \geqslant b$, pour tout $x \in \bar{B}(x_0, r)$. Posons $r_1 := \frac{1}{2}r$. Tout $x_1 \in \bar{B}(x_0, r_1)$ est tel que $b \leqslant f(x) \leqslant a$, pour tout $x \in \bar{B}(x_1, r_1)$. Appliquant l'inégalité (2.66) en x_1, avec $r = r_1$, il vient

$$|f(x_1 + h) - f(x_1)| \leqslant r_1^{-1}(a - b)\|h\|, \quad \text{pour tout } \|h\| < r_1. \tag{2.67}$$

Ceci assure que f est lipschitzienne de constante $r_1^{-1}(a-b)$ sur $\bar{B}(x_0, r_1)$, d'où la conclusion. ■

Définition 2.71 Soit $f : \mathbb{R}^n \to \bar{\mathbb{R}}$, de valeur finie en $\bar{x}$. On appelle *dérivée directionnelle* de f en $\bar{x}$ dans la direction $h \in \mathbb{R}^n$ la quantité (si elle existe) finie ou non :

$$f'(\bar{x}, h) := \lim_{t \downarrow 0} \frac{f(\bar{x} + th) - f(\bar{x})}{t}. \tag{2.68}$$

L'existence de la dérivée directionnelle équivaut à l'égalité des dérivées de Dini inférieure et supérieure (définition 2.54) dans la direction h. Si $q \in \partial f(\bar{x})$, une conséquence immédiate de la définition du sous-différentiel est $\liminf_{t\downarrow 0}(f(\bar{x} + th) - f(\bar{x}))/t \geqslant q \cdot h$, et donc

$$\liminf_{t \downarrow 0} \frac{f(\bar{x} + th) - f(\bar{x})}{t} \geqslant \sup\{q \cdot h;\, q \in \partial f(\bar{x})\}. \tag{2.69}$$

Si f est convexe, il est facile de vérifier que la fonction $t \mapsto (f(\bar{x} + th) - f(\bar{x})/t$ est croissante quand $t > 0$. Une fonction convexe a donc des dérivées directionnelles en tout point de son domaine (elles peuvent valoir $-\infty$ comme le montre l'exemple 2.58 de la fonction entropie). Nous allons étudier le cas d'égalité dans (2.69). Dans un premier temps, énonçons un lemme de *valeur moyenne* :

Lemme 2.72 *Soient x et y distincts dans $\mathbb{R}^n$, et $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ une fonction convexe, continue et prenant des valeurs finies sur le segment $[x, y]$. Alors il existe $z \in]x, y[$ et $q \in \partial f(z)$ tels que*

$$f(y) = f(x) + q \cdot (y - x). \tag{2.70}$$

Démonstration. Posons, pour $t \in \mathbb{R}$, $F(t) := f(x + t(y-x)) - (1-t)f(x) - tf(y)$. Cette fonction est convexe et nulle en 0 et 1. Sur $]0,1[$ elle est donc à valeurs négatives, donc F a même minimum sur l'intervalle $[0,1]$ et sur un intervalle de la forme $[\epsilon, 1-\epsilon]$ avec $\epsilon > 0$ assez petit. La fonction F, qui est continue sur cet intervalle, atteint son minimum en un point $\tau \in [\epsilon, 1-\epsilon] \subset]0,1[$, de là $0 \in \partial F(\tau)$. Posons $z := (1-\tau)x + \tau y$. Appliquons le lemme 2.64, avec ici $At = t(y-x)$. La condition de stabilité (2.60) résulte de la continuité de f en tout point de $]x, y[$. Il vient $\partial F(\tau) = (y-x) \cdot \partial f(z) + f(x) - f(y)$, d'où le résultat. ■

Théorème 2.73 *Soit f convexe $\mathbb{R}^n \to \bar{\mathbb{R}}$, de valeur finie en $\bar{x}$, tel que $\bar{x} \in \mathrm{int}(\mathrm{dom}(f))$. Alors $\partial f(\bar{x})$ est non vide et borné, et pour tout $h \in \mathbb{R}^n$:*

$$f'(\bar{x}, h) = \sup\{q \cdot h;\, q \in \partial f(\bar{x})\}. \tag{2.71}$$

Démonstration. D'après la proposition 2.70, il existe $\varepsilon > 0$ tel que f soit lipschitzien de constante L sur $B(\bar{x}, \varepsilon)$. De (2.69) on déduit que $\|q\| \leqslant L$, pour tout $q \in \partial f(\bar{x})$. Pour $t > 0$ assez petit, le point $y_t := \bar{x} + th$ est dans $B(\bar{x}, \varepsilon)$. Le lemme 2.72 assure l'existence de $z_t \in [\bar{x}, y_t]$ et $q_t \in \partial f(z_t)$ tels que

$$f(y_t) = f(\bar{x}) + q_t \cdot (y_t - \bar{x}) = f(\bar{x}) + tq_t \cdot (y - \bar{x}). \tag{2.72}$$

Il vient donc

$$f'(\bar{x}, h) = \lim_{t \downarrow 0} q_t \cdot (y - \bar{x}). \tag{2.73}$$

Passant à la limite dans l'inégalité

$$f(x') \geqslant f(z_t) + q_t \cdot (x' - z_t), \quad \text{pour tout } x' \in \mathbb{R}^n, \tag{2.74}$$

on vérifie que tout point d'adhérence de q_t (il en existe puisque q_t est borné au voisinage de $\bar{x}$) appartient à $\partial f(\bar{x})$ (qui est donc non vide). Passant à la limite dans (2.73), il vient $f'(\bar{x}, h) \leqslant \sup\{q \cdot h;\ q \in \partial f(\bar{x})\}$. On conclut avec (2.69). ■

Exercice 2.74 Soit $N(\cdot)$ une norme quelconque de $\mathbb{R}^n$. Calculer le sous-gradient et les dérivées directionnelles de $N(\cdot)$ en tout point. On utilisera la norme duale $N^*(y) := \sup\{y \cdot x;\ N(x) \leqslant 1\}$.

2.5.4 Dualité convexe

Cette section présente le lien entre existence de multiplicateurs de Lagrange et dualité, puis relie ces résultats à l'analyse de sensibilité de la valeur d'un problème.

Un classe assez générale de problèmes convexes avec contraintes peut s'écrire sous la forme

$$\operatorname*{Min}_{x \in \mathbb{R}^n} f(x); \quad Ax \in K, \tag{P}$$

où K est une partie convexe fermée non vide de $\mathbb{R}^p$. Par exemple, $K = \{b\}$ pour la contrainte d'égalité $Ax = b$, $K = \mathbb{R}^p_-$ pour des contraintes d'inégalité. Introduisons la *fonction indicatrice* associée à K :

$$I_K(x) = \begin{cases} 0 \text{ si } x \in K, \\ +\infty \text{ sinon.} \end{cases} \tag{2.75}$$

Notons que $\partial I_K(x) = N_K(x)$, où $N_K(x)$ est le *cône normal* à K en $x \in \mathbb{R}^n$, défini par

$$N_K(x) := \begin{cases} \emptyset \text{ si } x \notin K, \\ \{y \in \mathbb{R}^n;\ y \cdot (x' - x) \leqslant 0, \text{ pour tout } x' \in K\}, \text{ sinon.} \end{cases} \tag{2.76}$$

On peut réécrire (P) comme le problème de minimisation de $f(x) + I_K(Ax)$, et aussi comme

$$\operatorname*{Min}_{x \in \mathbb{R}^n, y \in \mathbb{R}^p} f(x) + I_K(y); \quad Ax - y = 0. \tag{P'}$$

Le lagrangien $L : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ associé à la dualisation de la contrainte $Ax - y = 0$ est

$$L(x, y, \lambda) := f(x) + I_K(y) + \lambda \cdot (Ax - y). \tag{2.77}$$

Or $\sup_\lambda L(x, y, \lambda)$ vaut $f(x)$ si $y = Ax$ et $y \in K$, et $+\infty$ sinon ; le problème (P) revient à minimiser $\sup_\lambda L(x, y, \lambda)$ par rapport à x. Comme nous l'avons vu en section 2.4.1, on obtient le problème dual en échangeant l'ordre de maxi-minimisation. La minimisation du lagrangien en (x, y) donne le critère dual

$$d(\lambda) := \inf_{x,y} \{f(x) + I_K(y) + \lambda \cdot (Ax - y)\}. \tag{2.78}$$

Les minimisations en x et y se séparent :

$$d(\lambda) := \inf_x \{f(x) + \lambda \cdot Ax)\} + \inf_y \{I_K(y) - \lambda \cdot y\}. \tag{2.79}$$

Introduisons la *conjuguée de Legendre-Fenchel* de f : c'est une application $\mathbb{R}^n \to \bar{\mathbb{R}}$ définie par

$$f^*(x') := \sup_x \{x' \cdot x - f(x)\}. \tag{2.80}$$

Notons qu'on appelle *transformation de Legendre-Fenchel* l'application $f \to f^*$. La conjuguée de l'indicatrice de K est dite *fonction d'appui* de K et notée

$$\sigma_K(\lambda) := \sup_{y \in K} \lambda \cdot y. \tag{2.81}$$

Le critère dual a donc pour expression

$$d(\lambda) := -f^*(-A^\top \lambda) - \sigma_K(\lambda), \tag{2.82}$$

et le problème dual est

$$\operatorname*{Max}_\lambda -f^*(-A^\top \lambda) - \sigma_K(\lambda). \tag{D}$$

Notons une relation importante entre les notions de sous différentiel et de conjugaison :

Lemme 2.75 *Soit $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ ayant une valeur finie en $\bar{x}$. Pour tout $y \in \mathbb{R}^n$, on a l'inégalité de Fenchel-Young $f^*(y) \geqslant y \cdot \bar{x} - f(\bar{x})$, avec égalité ssi $y \in \partial f(\bar{x})$.*

Démonstration. L'inégalité est conséquence de la définition de f^*. L'égalité en $y = \bar{y}$ signifie que $\bar{y} \cdot \bar{x} - f(\bar{x}) \geqslant \bar{y} \cdot x - f(x)$, pour tout $x \in \mathbb{R}^n$, soit $f(x) \geqslant f(\bar{x}) + \bar{y} \cdot (x - \bar{x})$ qui est la définition du sous différentiel. ■

La condition de stabilité du problème (P) est, par définition :

$$\text{Il existe } \varepsilon > 0; \quad \varepsilon B \subset K - A \operatorname{dom}(f). \tag{2.83}$$

Abordons maintenant le lien avec la théorie de la perturbation de problèmes d'optimisation. Plongeons (P) dans la classe de problèmes

$$\operatorname*{Min}_{x \in \mathbb{R}^n} f(x); \quad Ax + b \in K, \tag{P_b}$$

indicés par $b \in \mathbb{R}^p$, et posons $v(b) := \operatorname{val}(P_b)$. La condition de stabilité (2.83) revient à demander que (P_b) est réalisable, quand b est proche de 0. Notons qu'on a toujours $\operatorname{val}(D) \leqslant \operatorname{val}(P) = v(0)$.

Lemme 2.76 *On suppose que v a une valeur finie en 0. Alors*

$$\partial v(0) = \{\lambda \in \mathbb{R}^p;\ d(\lambda) = v(0)\}. \tag{2.84}$$

Autrement dit, $\partial v(0) = S(D)$ si $v(0) = \operatorname{val}(D)$, et $\partial v(0) = \emptyset$ sinon.

Démonstration. D'après le lemme 2.75, $\lambda \in \partial v(0)$ ssi $v^*(\lambda) = -v(0)$. La conclusion découle donc des égalités

$$\begin{array}{rcl} v^*(\lambda) & = & \sup_{b \in \mathbb{R}^p} \{\lambda \cdot b - v(b)\}, \\ & = & \sup_{b,x} \{\lambda \cdot b - f(x) - I_K(Ax + b)\}, \\ & = & \sup_{b,x} \{\lambda \cdot (Ax + b) - I_K(Ax + b) - f(x) - \lambda \cdot Ax\}, \\ & = & \sup_{b',x} \{\lambda \cdot b' - I_K(b') - f(x) - \lambda \cdot Ax\}, \\ & = & \sigma_K(\lambda) + f^*(-A^\top \lambda) = -d(\lambda). \end{array} \tag{2.85}$$

■

Théorème 2.77 (Dualité forte) *Supposons f convexe, et la condition de stabilité* (2.83) *satisfaite. Alors primal et dual ont même valeur. Si cette valeur est finie, v est localement lipschitzienne au voisinage de* 0*, l'ensemble $\Lambda := S(D)$ des solutions du problème dual est non vide et borné, égal à $\partial v(b)$, et*

$$v'(b,d) = \sup\{\lambda \cdot d\,;\ \lambda \in \Lambda\}. \tag{2.86}$$

De plus, toujours si cette valeur est finie, $\bar{x} \in S(P)$ ssi il existe $\lambda \in \mathbb{R}^p$ tel que

$$\lambda \in N_K(A\bar{x}); \quad \partial f(\bar{x}) + A^\top \lambda \ni 0, \tag{2.87}$$

et l'ensemble des λ satisfaisant (2.87) *n'est autre que Λ.*

Démonstration. La condition (2.83) implique $v(0) < +\infty$. Si $v(0) = -\infty$, comme $v(0) = \mathrm{val}(P) \geqslant \mathrm{val}(D)$, la conclusion est satisfaite. Si au contraire $v(0) \in \mathbb{R}$, v est localement lipschitzienne au voisinage de 0 (proposition 2.70) et satisfait la formule de dérivées directionnelles (2.71) (théorème 2.73). On déduit alors (2.86) du lemme 2.76.

Exprimons la différence entre les critères primal et dual au point $(\bar{x}, \lambda)$, en faisant apparaître les inégalité de Fenchel-Young :

$$\begin{array}{rl} \Delta & = f(\bar{x}) + f^*(-A^\top\lambda) + \sigma_K(\lambda) \\ & = \big(f(\bar{x}) + f^*(-A^\top\lambda) + \lambda \cdot A\bar{x}\big) + (\sigma_K(\lambda) - \lambda \cdot A\bar{x})\,. \end{array} \tag{2.88}$$

D'après le lemme 2.75, chacun des deux terme est positif ; le premier est nul ssi $-A^\top\lambda \in \partial f(\bar{x})$, et le second est nul ssi $\lambda \cdot y$ atteint son maximum, pour $y \in K$, en $A\bar{x}$, soit $\lambda \in N_K(A\bar{x})$. La différence est donc nulle ssi (2.87) est satisfait, ce qui montre que (2.87) caractérise Λ. ■

2.5.5 Programmation linéaire conique

On qualifiera de *linéaires coniques* les problèmes d'optimisation de la forme

$$\operatorname*{Min}_{x \in \mathbb{R}^n} c \cdot x; \quad Ax - b \in C, \tag{2.89}$$

où $C \subset \mathbb{R}^p$ est un cône convexe fermé. Les deux exemples fondamentaux développés dans ce livre sont les cas $C = \{0\}_{\mathbb{R}^q} \times \mathbb{R}^p_-$ (programmation linéaire) et celui du cône des matrices symétriques semi définies positives (voir le chapitre 9).

On peut appliquer la théorie de la section précédente en posant $K = b + C$. Notons le *cône polaire* (négatif) de C par

$$C^- := \{z \in \mathbb{R}^n; z \cdot y \leqslant 0, \ \text{ pour tout } \ z \in C\}. \tag{2.90}$$

Exercice 2.78 (i) Vérifier que le cône polaire d'un produit cartésien d'ensemble est le produit cartésien des polaires. (ii) Vérifier par un argument de séparation que tout cône convexe fermé est égal à son bipolaire.

On a $\sigma_K(\lambda) = b \cdot \lambda + I_{C^-}(\lambda)$. De plus, puisque le critère $f(x) = c \cdot x$ est linéaire, on a

$$f^*(z) = \begin{cases} 0 & \text{si } z = c, \\ +\infty & \text{sinon.} \end{cases} \tag{2.91}$$

Le problème dual a donc pour expression (voir (2.82))

$$\operatorname*{Max}_{\lambda \in C^-} -b \cdot \lambda; \quad c + A^\top\lambda = 0. \tag{2.92}$$

Si on préfère exprimer le dual à l'aide du *cône polaire positif* $C^+ = -C^-$, on obtient la formulation équivalente

$$\operatorname*{Max}_{\eta \in C^+} b \cdot \eta; \quad A^\top\eta = c.$$

Le problème dual est donc aussi un problème linéaire conique, qu'on peut mettre sous forme primale en le réécrivant comme un problème de minimisation :

$$\underset{\lambda\in\mathbb{R}^p}{\mathrm{Min}}\, b\cdot\lambda;\quad (\lambda, c+A^\top\lambda)\in K,$$

avec $K := C^-\times\{0\}$. Dualisons le problème précédent. Le cône polaire négatif de K est (utiliser l'exercice 2.78) $C\times\mathbb{R}^n$. Notant $(z,-x)$ le multiplicateur, le lagrangien a pour expression

$$b\cdot\lambda + z\cdot\lambda - x\cdot(c+A^\top\lambda) = -c\cdot x + (b-Ax+z)\cdot\lambda,$$

et le problème bidual (dual du dual) est donc

$$\underset{x,z}{\mathrm{Max}} -c\cdot x;\quad Ax-b=z;\quad z\in C.$$

On peut retrouver cette expression en appliquant directement la formule (2.92) au problème dual. Nous avons démontré :

Lemme 2.79 *L'application de dualité est involutive dans la classe des problèmes linéaires coniques. Autrement dit, le bidual (réécrit comme un problème de minimisation) d'un problème linéaire conique coïncide avec le primal.*

Primal et dual jouent donc des rôles symétriques. On peut donc leur appliquer indifféremment le théorème 2.77 de dualité forte. La condition de stabilité (2.83) du problème primal a pour expression

$$\text{Il existe } \varepsilon>0;\quad \varepsilon B\subset C+b-\mathrm{Im}(A). \tag{2.93}$$

Elle est satisfaite si A est surjective, ou s'il existe $x\in\mathbb{R}^n$ tel que $Ax-b\in\mathrm{int}\, C$.

Corollaire 2.80 *Si* (2.93) *est satisfaite, alors* (2.89) *et* (2.92) *ont même valeur, et si cette valeur commune est finie, l'ensemble des solutions du problème dual* (2.92) *est non vide et borné.*

Dans le résultat correspondant pour le problème dual nous fournissons une caractérisation de la condition de stabilité.

Corollaire 2.81 *Si la condition*

$$\textit{Il existe } \varepsilon>0;\quad \varepsilon B\subset c+A^\top C^-. \tag{2.94}$$

est satisfaite, alors (2.89) *et* (2.92) *ont même valeur, et si cette valeur commune est finie, l'ensemble des solutions du problème primal* (2.89) *est non vide et borné.*

Démonstration. Il suffit de vérifier que (2.94) équivaut à la condition de stabilité du problème dual. Cette dernière a pour expression

$$\varepsilon B\subset C^-\times\{-c\} - \{(\lambda, A^\top\lambda);\ \lambda\in\mathbb{R}^p\}, \tag{2.95}$$

ou encore, pour tout (μ,η) proche de 0 dans $\mathbb{R}^p\times\mathbb{R}^n$, il existe $\lambda\in\mathbb{R}^p$ tel que

$$\mu\in C^--\lambda;\quad \eta=-c-A^\top\lambda. \tag{2.96}$$

La première relation équivaut à $\lambda=\hat\lambda-\mu$, avec $\hat\lambda\in C^-$. Reportant dans la seconde on obtient $c+A^\top\hat\lambda = A^\top\mu-\eta$, relation dont on montre facilement qu'elle est satisfaite ssi (2.95) l'est. ■

2.6 Notes

Pour plus d'information sur la théorie des polyèdres et les questions de totale unimodularité, on pourra se reporter au livre de Schrijver [Sch86], ou bien au texte plus introductif [CCPS98]. La référence de base pour l'analyse convexe en dimension finie est Rockafellar [Roc70] ; voir aussi Hiriart-Urruty et Lemaréchal [HL96]. Les extensions en dimension infinie, basées sur le théorème de Hahn-Banach, sont développées dans Ekeland et Temam [ET76]. Une introduction à l'analyse non lisse en dimension finie est donnée dans [BL00]. L'analyse de sensibilité en optimisation est exposée dans [BS00].

Chapitre 3

Problèmes de flots

Les problèmes de flots remontent aux origines de la recherche opérationnelle : ils comprennent comme cas particulier le problème de transport de masse, énoncé par Monge en 1781 et reformulé par Kantorovitch en termes de programmation linéaire vers 1942 ; leur théorie inclut les résultats fondamentaux sur les problèmes de couplage dans les graphes bipartis, dus à König, développée à partir de 1914, en relation avec des problèmes combinatoires apparaissant en algèbre linéaire (théorème de König-Frobenius). Un résultat central, le théorème de Ford-Fulkerson "flot-maximal=coupe-minimale", démontré en 1956, permet de déterminer le flot maximal entre deux points donnés d'un réseau sachant que les flots sur les arcs sont majorés par des capacités attachées aux arcs. Les problèmes de flots à coût minimum constituent une classe générale permettant de modéliser et de résoudre de nombreux problèmes pratiques. Ils interviennent aussi comme outils dans le traitement de problèmes plus difficiles. La théorie des problèmes de flots offre un exemple remarquable de synthèse entre l'optimisation et la théorie des graphes, l'interprétation combinatoire des conditions d'optimalité et des résultats de dualité en programmation linéaire conduisant naturellement à des algorithmes polynomiaux très efficaces. Nous prenons ici le parti d'aller du général au particulier, en donnant un aperçu de la théorie des problèmes de flots à coût minimum, qui incluent les problèmes de plus court chemins comme cas particulier. Ces derniers peuvent être traités par des méthodes spéciales, notamment de programmation dynamique, que nous abordons au chapitre 4.

3.1 Flots dans un graphe : définitions, premières propriétés

Considérons un graphe orienté $\mathcal{G} = (\mathcal{N}, \mathcal{A})$: rappelons que $\mathcal{N}$ est l'ensemble des *nœuds*, et $\mathcal{A} \subset \mathcal{N} \times \mathcal{N}$ est l'ensemble des *arcs*. On munit chaque arc $(i, j) \in \mathcal{A}$ d'une *capacité* $u_{ij} \in \mathbb{R}_+ \cup \{+\infty\}$ et d'un coût $c_{ij} \in \mathbb{R}$. (Le "u" dans u_{ij} est pour "upper bound".) On se donne aussi en chaque nœud du graphe un flot entrant exogène $b_i \in \mathbb{R}$, compté algébriquement (ce qui signifie que si $b_i < 0$, il s'agit d'un flot sortant en i d'une valeur $-b_i$). On appelle *flot* (positif) une fonction $x \in \mathbb{R}^{\mathcal{A}}$, $(i, j) \mapsto x_{ij}$, vérifiant la *loi des nœuds de Kirchhoff*, exprimant l'égalité des flots entrants et sortants en tous points :

$$b_i + \sum_{j \in \mathcal{N}, (j,i) \in \mathcal{A}} x_{ji} = \sum_{j \in \mathcal{N}, (i,j) \in \mathcal{A}} x_{ij}, \qquad \forall i \in \mathcal{N} \tag{3.1}$$

ainsi que la contrainte de positivité

$$0 \leqslant x_{ij}, \qquad \forall (i, j) \in \mathcal{A} \ . \tag{3.2}$$

Dans toute la suite, on supposera que le graphe n'a pas de boucles, c'est-à-dire pas d'arcs de la forme (i, i), ce qui n'entraîne pas de perte de généralité (car les flots associés aux boucles disparaissent dans le bilan (3.1)).

On parle de flot *algébrique* lorsque l'on impose (3.1) mais pas (3.2). Dans le cas d'un graphe orienté, il est naturel de considérer des flots positifs plutôt que des flots algébriques, ce qui revient à dire que le flot sur un arc orienté de i à j circule en direction de i vers j. Dans la suite, les flots seront toujours positifs, sauf mention explicite du contraire.

Il est commode d'introduire le *flot net sortant* au nœud i :

$$f_i(x) := \sum_{j \in \mathcal{N}, (i,j) \in \mathcal{A}} x_{ij} - \sum_{j \in \mathcal{N}, (j,i) \in \mathcal{A}} x_{ji} \ ,$$

ce qui permet de résumer la loi des nœuds de Kirchhoff par :

$$b = f(x) \ .$$

Un flot tel que $f(x) = 0$ est appelé *circulation*.

En pratique, un flot pourra représenter un fluide circulant en régime stationnaire sur un réseau (eau, gaz, courant électrique), mais aussi des biens (nombre de colis, des containers), si l'on considère des flots à valeurs entières.

En sommant les flot nets sortants $f_i(x)$ sur l'ensemble des nœuds i, on obtient aussitôt :

Observation 3.1 Une condition nécessaire pour l'existence d'un flot est que la somme des flots exogènes entrants soit nulle :

$$\sum_{i \in \mathcal{N}} b_i = 0 \ . \tag{3.3}$$

Nous supposerons toujours que la condition (3.3) est vérifiée.

Un flot est dit *admissible* s'il satisfait les contraintes de capacité :

$$x_{ij} \leqslant u_{ij}, \qquad \forall (i,j) \in \mathcal{A} \ . \tag{3.4}$$

Définition 3.2 (Problème de flot à coût minimum) On appelle problème de *flot à coût minimum* le programme linéaire :

$$\min \sum_{(i,j) \in \mathcal{A}} c_{ij} x_{ij} \ \text{ sous les contraintes } \ (3.1), (3.2), (3.4). \tag{3.5}$$

Le problème de flot à coût minimum admet plusieurs sous-problèmes importants.

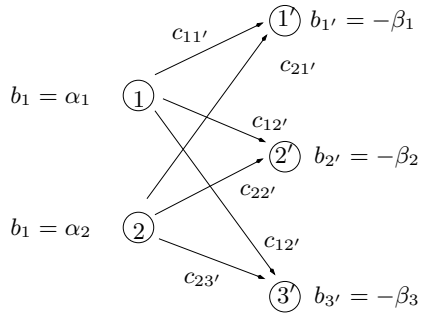
Définition 3.3 (Problème de transport) On appelle *problème de transport* un problème de flot à coût minimum pour lequel les capacités sont infinies, i.e., $u \equiv +\infty$.

La terminologie "transport" est motivée par le problème fondamental suivant.

Exemple 3.4 (Problème de transport de masse) Reprenons à l'aide du formalisme des flots le problème de transport de masse, déjà mentionné dans l'exemple 1.9 du Chapitre 1. Dessinons pour cela un graphe orienté, avec $n + m$ nœuds, les n premiers nœuds étant désignés par $1, \ldots, n$, et les m suivants par $1', \ldots, m'$. Pour tous $1 \leqslant i \leqslant n$ et $1 \leqslant j \leqslant m$, dessinons un arc du nœud i au nœud j'. Si l'on veut être formel, on posera : $\mathcal{N} := \{1, \ldots, n, 1', \ldots, m'\}$, $\mathcal{A} = \{(i, j') \mid 1 \leqslant i \leqslant n, \ 1 \leqslant j \leqslant m\}$. Munissons l'arc (i, j') du coût $c_{ij'}$ et de la capacité $u_{ij'} = +\infty$. Quant au vecteur b de flots exogènes, définissons le par :

$$b_i = \alpha_i, \ 1 \leqslant i \leqslant n, \ \ b_{j'} = -\beta_j, \ 1 \leqslant j \leqslant m \ ,$$

où α_i est la masse du déblai i et β_j la masse du remblai j. Par exemple, lorsque $n = 2$ et $m = 3$, le graphe est :



En un nœud de type i, pour $1 \leqslant i \leqslant n$, la loi des nœuds de Kirchhoff, $b_i = f_i(x)$, donne

$$\alpha_i = \sum_{1 \leqslant j \leqslant m} x_{ij'} \ . \tag{3.6}$$

De même, un nœud de type j, pour $1 \leqslant j' \leqslant n$, $b_{i'} = f_{j'}(x)$, donne

$$\sum_{1 \leqslant i \leqslant n} x_{ij'} = \beta_{j'} \ . \tag{3.7}$$

On voit donc que l'ensemble des flots admissibles est exactement l'ensemble des plans de transport. Le problème de flot à coût minimum (3.5) consiste exactement à minimiser le coût d'un plan de transport.

Remarque 3.5 Une variante du problème de transport de masse consiste à remplacer chaque contrainte (3.6) par

$$\alpha_i \geqslant \sum_{1 \leqslant j \leqslant m} x_{ij'} \ ,$$

ce qui modélise la situation où l'on s'autorise à consommer seulement une partie de la masse α_i du déblai i. On conserve les contraintes (3.7). et l'on suppose maintenant que

$$\alpha_1 + \cdots + \alpha_n \geqslant \beta_1 + \cdots + \beta_m \ .$$

Ce problème se ramène à celui de l'exemple 3.4 : il suffit pour cela de rajouter un remblai imaginaire 0, de masse égale à la masse manquante,

$$\beta_0 := \alpha_1 + \cdots + \alpha_n - \beta_1 - \cdots - \beta_m \ ,$$

avec un coût de transport $c_{i0'} = 0$, pour tout $1 \leqslant i \leqslant n$.

Remarque 3.6 Des contraintes inférieures et supérieures sur le flot

$$\ell_{ij} \leqslant x_{ij} \leqslant u_{ij}$$

peuvent se ramener au cas traité ici où $\ell_{ij} = 0$. Il suffit en effet d'effectuer le changement de variable $x_{ij} = \ell_{ij} + x'_{ij}$, où les nouvelles variables de flots x'_{ij} vérifient maintenant $0 \leqslant x'_{ij} \leqslant u'_{ij}$ avec $u'_{ij} = u_{ij} - \ell_{ij}$, et satisfont une nouvelle loi des nœud de Kirchhoff $f(x') = b'$ avec $b' = b - f(\ell)$. Cette transformation ne fait que rajouter une constante au critère.

Un autre cas particulier fondamental du problème de flot à coût minimum est le *problème du flot maximal*, parfois appelé tout simplement *problème de flot*. Alors que dans les données du problème de transport, on avait des coûts, et pas de capacités, dans les données du problème du flot maximal, on aura des capacités, et pas de coûts. Pour définir le problème de flot maximal, on supposera que $\mathcal{G}$ a deux nœuds distingués, s et p, appelés respectivement *source* et *puits*, tels que s n'a pas de prédécesseur (i.e., $\{i \in \mathcal{N} \mid (i,s) \in \mathcal{A}\} = \emptyset$), et p n'a pas de successeur (i.e., $\{i \in \mathcal{N} \mid (p,i) \in \mathcal{A}\} = \emptyset$). Soit $v \in \mathbb{R}_+$. On appelle *flot admissible de s à*

p de valeur v un vecteur $x = (x_{ij})_{(i,j)\in\mathcal{A}}$ tel que le flot net entrant en s soit égal à v, $f_s(x) = v$, le flot net sortant en p soit égal à $-v$, $f_p(x) = -v$, le flot net en tout autre nœud soit nul, $f_i(x) = 0$ pour $i \in \mathcal{N} \setminus \{s, p\}$, les contraintes de positivité et de capacité étant satisfaites, $0 \leqslant x_{ij} \leqslant u_{ij}$. Le vecteur x est donc solution de (3.1),(3.2),(3.4), avec

$$b_i = \begin{cases} v & \text{si } i = s, \\ -v & \text{si } i = p, \\ 0 & \text{sinon.} \end{cases}$$

Définition 3.7 (Problème du flot maximal) Le problème du flot maximal consiste à trouver un flot admissible de s à p de valeur maximale.

Remarque 3.8 Le problème du flot maximal de s à p est un cas particulier du problème de flot à coût minimal. Rajoutons en effet un arc de p à s, muni d'un coût -1, en munissant tous les autres arcs d'un côut nul, et donnons à ce nouvel arc une capacité $+\infty$, en conservant les autres capacités. Un flot x dans ce nouveau graphe tel que le flot net $f_i(x)$ soit nul en tout nœud i correspond exactement à un flot admissible de s à p dans l'ancien graphe, et le coût du flot est l'opposé de sa valeur.

Remarque 3.9 Dans l'énoncé du problème de flot maximal, l'hypothèse que le nœud source s n'a pas de prédécesseur, et que le nœud puits p n'a pas de successeur, n'est pas restrictive : si l'on cherche un flot d'une origine o à une destination d données, on peut toujours rajouter à un graphe un nœud source imaginaire s et un nœud puits imaginaire p, avec un arc de s à o de coût nul et de capacité infinie, et un arc de d à p de coût nul et de capacité infinie.

On peut décomposer un flot quelconque en termes de chemins. À chaque chemin de s à p et à chaque $v \in \mathbb{R}_+$, on associe un flot x de valeur v, tel que $x_{ij} = v$ si (i, j) appartient au chemin, et $x_{ij} = 0$ sinon. A chaque circuit du graphe on associe selon le même procédé un flot x, tel que $x_{ij} = v$ si (i, j) appartient au circuit, et $x_{ij} = 0$ sinon. Ces flot seront dits unitaires si $v = 1$. Rappelons qu'un chemin est *élémentaire* si tous ses nœuds sont distincts, et qu'un circuit est *élémentaire* si tous ses nœuds à l'exception du dernier sont distincts.

Proposition 3.10 (Flots et chemins) *Tout flot de s à p peut s'écrire comme combinaison linéaire positive de flots unitaires associés à des chemins élémentaires de s à p et de flots unitaires associés à des circuits élémentaires.*

Cette proposition se déduit du résultat suivant, grâce à la construction de la Remarque 3.8.

Proposition 3.11 *Toute circulation dans un graphe est combinaison linéaire positive de flots unitaires associés à des circuits élémentaires.*

Démonstration. Dans le cas d'une circulation, la conservation des flots au nœud j s'écrit $\sum_{(i,j)\in\mathcal{A}} x_{ij} = \sum_{(j,k)\in\mathcal{A}} x_{jk}$. Donc, pour tout arc (i, j) tel que $x_{(i,j)} > 0$, on peut trouver un arc (j, k) tel que $x_{(j,k)} > 0$. Si une circulation x est non-nulle, on en déduit qu'il existe un chemin infini $i_0, i_1, i_2, \ldots$ tel que $x_{i_0 i_1} > 0$, $x_{i_1 i_2} > 0$, etc. Comme le nombre de nœuds du graphe est fini, ce chemin passe deux fois par le même nœud. Autrement dit, $i_{r+c} = i_r$ pour un certain $r \geqslant 0$ et $c \geqslant 1$. Le chemin $(i_r, \ldots, i_{r+c})$ est donc un circuit, et ce circuit est même élémentaire si c est choisi minimal. En notant v le minimum des valeurs du flot sur ce circuit, et en définissant la circulation y valant 1 sur ce circuit, et 0 ailleurs, on voit que $x' := x - vy$ est encore une circulation, et que par rapport à x, x' s'annule sur au moins un arc supplémentaire. Par récurrence sur le nombre d'arcs sur lesquels la circulation ne s'annule pas, on conclut que x est combinaison linéaire positive de flots unitaires associés à des circuits élémentaires. ■

Remarque 3.12 La Proposition 3.10 fournit une bonne illustration du théorème de Minkowski-Weyl (Théorème 2.21). L'ensemble des flots de s à p de valeur unité est un polyèdre, dont les points extrémaux correspondent aux chemins élémentaires de s à p. Les directions extrémales du cône asymptotique de ce polyèdre correspondent aux circuits élémentaires.

Les problèmes de plus court chemin sont des cas particuliers de problèmes de flot à coût minimum. Considérons en effet le problème de calcul du chemin de coût minimum entre une origine o et une destination d d'un graphe orienté, les coûts (ou longueur) des arcs étant donnés par les c_{ij}. Cherchons pour cela un flot x de o à d de valeur 1 et de coût minimum, c'est-à-dire un vecteur $x = (x_{ij})$ tel que $x_{ij} \geqslant 0$ pour tout arc (i, j), tel que le flot net sortant en tout nœud i distinct de o et d soit nul, soit $f_i(x) = 0$ pour $i \in \mathcal{N} \setminus \{o, d\}$, et tel que $f_o(x) = -f_d(x) = 1$, minimisant le coût :

$$\sum_{(i,j)\in\mathcal{A}} c_{ij} x_{ij}$$

En utilisant la Proposition 3.10 (et éventuellement la Remarque 3.9), on voit que l'optimum, s'il existe, est atteint par un flot associé à un chemin de o à d.

Notons que la loi des nœuds de Kirchhoff (3.1) peut s'écrire $Mx = b$, où la matrice $M \in \mathbb{R}^{\mathcal{N}\times\mathcal{A}}$, appelée *matrice d'incidence nœuds-arcs* de $\mathcal{G}$, est définie par

$$M_{i,(j,k)} = \begin{cases} -1 & \text{si } i = k, \\ 1 & \text{si } i = j, \\ 0 & \text{sinon.} \end{cases}$$

Notons que la matrice d'incidence est bien définie, car le graphe est supposé sans boucles. L'ensemble des flots admissibles s'écrit donc

$$\{x \in \mathbb{R}^{\mathcal{A}} \mid Mx = b,\ 0 \leqslant x \leqslant u\} \ . \tag{3.8}$$

Proposition 3.13 *La matrice d'incidence nœuds-arcs d'un graphe est totalement unimodulaire.*

Démonstration. C'est une conséquence immédiate de la Proposition 2.37. ■

Le résultat suivant est fondamental.

Corollaire 3.14 (Optimalité des flots entiers) *Si les flots entrants exogènes b_i sont entiers, et si les capacités u_{ij} sont entières ou infinies, alors, les points extrémaux de l'ensemble* (3.8) *des solutions admissibles d'un problème de flot à coût minimal sont entiers. En particulier, si l'ensemble des flots admissibles est borné et non-vide, il existe un flot entier optimal.*

Démonstration. C'est une conséquence immédiate du Corollaire 2.34 et de la Proposition 3.13. ■

Exercice 3.15 Montrer que le problème de couverture (exercice 2.38) peut se modéliser par un problème de flot à coût minimum, et retrouver ainsi la conclusion de l'exercice 2.38.

Exemple 3.16 (Problème d'affectation optimale) Ce problème, objet de l'Exemple 1.6, a pour expression

$$\max_{\sigma\in\mathfrak{S}_n} \sum_{1\leqslant i\leqslant n} s_{i\sigma(i)} \ . \tag{3.9}$$

Il s'agit d'une version avec contrainte d'intégrité du du problème de transport de masse (exemple 3.4), consistant à maximiser

$$\sum_{1\leqslant i,j\leqslant n} s_{ij} x_{ij} \tag{3.10}$$

sur l'ensemble des x vérifiant :

$$\forall 1 \leqslant i \leqslant n, \qquad 1 = \sum_{1 \leqslant k \leqslant n} x_{ik}, \tag{3.11a}$$

$$\forall 1 \leqslant j \leqslant n, \qquad 1 = \sum_{1 \leqslant k \leqslant n} x_{kj}, \tag{3.11b}$$

$$\forall 1 \leqslant i, j, \leqslant n, \qquad 0 \leqslant x_{ij}. \tag{3.11c}$$

Ce problème de transport de masse est lui même un problème de flot à coût minimum. Il résulte alors du théorème d'optimalité des flots entiers (Corollaire 3.14) que l'optimum de ce problème est atteint pour un x entier, c'est-à-dire, pour un x représentant une matrice de permutation. Ce résultat a une interprétation plaisante en termes de mariage : si l'on envisage de marier n garçons et n filles, s_{ij} étant le bonheur du couple d'individus (i, j), la valeur du problème d'affectation optimale représente la valeur d'un ensemble de n mariages monogames, alors qu'une solution non-entière x de (3.11) correspond à un ensemble de mariages polygames : x_{ij} désigne le pourcentage de temps que i passe avec j. Le théorème d'intégrité des flots démontre ainsi que la monogamie est optimale. Cette conclusion est due au caractère linéaire du critère. On pourrait considérer plus généralement un critère de la forme :

$$\sum_{1 \leqslant i,j \leqslant n} s_{ij} x_{ij}^{\gamma} \ ,$$

avec $\gamma > 0$. Supposons les coefficients s_{ij} positifs. Alors, le cas où $\gamma < 1$ fournit un critère concave, qui modélise la lassitude. Dans ce cas, l'optimum n'est plus nécessairement atteint en un x entier. Par contre, si $\gamma > 1$, le critère est convexe, et d'après le Corollaire 2.34, le maximum du critère est encore atteint en un point entier.

3.2 Algorithmes de flots

Nous présentons maintenant les algorithmes de flots, qui vont nous permettre de traduire de manière efficace le résultat théorique d'intégrité des flots (Corollaire 3.14). Nous commençons par le problème du flot de valeur maximale (Définition 3.7), car l'algorithme est plus simple à présenter dans ce cas. Nous cherchons donc un flot de valeur maximale v d'une source s à un puits p, qui soit admissible, c'est-à-dire qui vérifie la contrainte de capacité (3.4).

3.2.1 Flots maximaux et coupes minimales

Nous allons tout d'abord majorer la valeur v d'un flot admissible. Appelons pour cela *coupe* du graphe $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ une partition $\mathcal{N} = S \cup \overline{S}$ telle que $S \neq \emptyset$ et $S \neq \mathcal{N}$. On dit que cette coupe *sépare* s de p si $s \in S$ et $p \in \overline{S}$. Si I et J sont des sous-ensembles de $\mathcal{N}$, et si y est une fonction de $\mathcal{A}$ dans $\mathbb{R}_+ \cup \{+\infty\}$, nous définissons :

$$y(I, J) = \sum_{i \in I,\ j \in J,\ (i,j) \in \mathcal{A}} y_{ij} \ .$$

En particulier, $u(S, \overline{S}) = \sum_{i \in S,\ j \in \overline{S},\ (i,j) \in \mathcal{A}} u_{ij}$ est appelée *capacité* de la coupe $(S, \overline{S})$.

Lemme 3.17 *La valeur d'un flot admissible de s à p est majorée par la capacité de toute coupe séparant s de p.*

Démonstration. On va montrer que pour toute coupe $(S, \overline{S})$ séparant s de p et pour tout flot x de valeur v

$$x(S, \overline{S}) - x(\overline{S}, S) = v \ . \tag{3.12}$$

En effet, en sommant les lois des nœuds (3.1) pour tout $i \in S$, il vient

$$\sum_{i\in S}\Big(\sum_{j\in\mathcal{N},\,(i,j)\in\mathcal{A}} x_{ij} - \sum_{j\in\mathcal{N},\,(j,i)\in\mathcal{A}} x_{ji}\Big) = v \ , \tag{3.13}$$

Quand i et j appartiennent tous deux à S, x_{ij} se simplifie dans (3.13), et quand i et j appartiennent tous deux à $\overline{S}$, x_{ij} n'apparaît pas dans (3.13), ce qui nous permet de réécrire (3.13) sous la forme (3.12). Le lemme résulte aussitôt de (3.12), de $x(S,\overline{S}) \leqslant u(S,\overline{S})$, et de $x(\overline{S},S) \geqslant 0$. ■

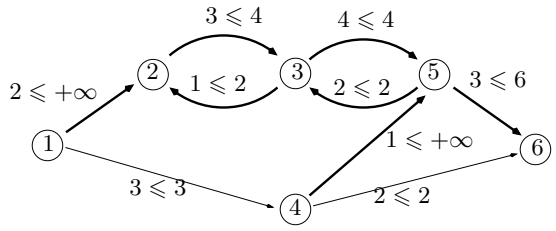
Le théorème fondamental de la théorie des flots, dû à Ford et Fulkerson [FF56], affirme que la majoration du Lemme 3.17 est optimale, i.e. que "max flow=min cut".

Théorème 3.18 (Flot maximal=coupe minimale) *Le supremum des valeurs des flots admissibles de s à p est égal à l'infimum des capacités des coupes séparant s de p.*

La preuve de ce théorème utilise la notion de *graphe résiduel* que nous introduisons maintenant. Afin de rendre plus simples les écritures algébriques, il sera commode de raisonner dans un graphe augmenté, toujours noté $\mathcal{G} = (\mathcal{N},\mathcal{A})$, tel que s'il y a un arc $(i,j) \in \mathcal{A}$, il y a aussi un arc en sens inverse, $(j,i) \in \mathcal{A}$: si l'arc (i,j), mais non l'arc (j,i), est dans le graphe initial, nous rajoutons l'arc (j,i) en le munissant d'une capacité $u_{ji} = 0$, ce qui ne change rien au problème. (Nous ne représenterons pas les arcs de capacité nulle sur les dessins.) Considérons donc un flot admissible x, et appelons, pour tout $(i,j) \in \mathcal{A}$, flot net de i à j la quantité $x_{ij} - x_{ji}$. Si l'on cherche à accroître localement le flot net, on peut augmenter x_{ij} de $u_{ij} - x_{ij}$, et symétriquement faire passer x_{ji} à 0. Aussi appelons-nous *capacité résiduelle* r_{ij} de l'arc (i,j) l'accroissement maximal possible du flot net de i à j, soit

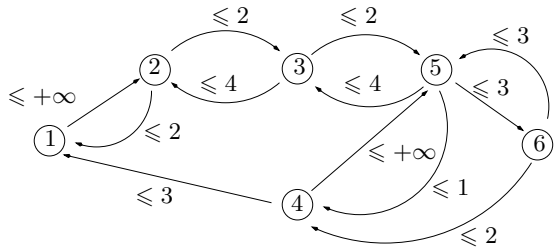
$$r_{ij} = u_{ij} - x_{ij} + x_{ji} \ .$$

Le graphe $\mathcal{G}_r(x) = (\mathcal{N},\mathcal{A}_r)$, où $\mathcal{A}_r = \{(i,j) \in \mathcal{N}^2 \mid r_{ij} > 0\}$ est appelé *graphe résiduel*. La capacité d'une coupe $(S,\overline{S})$ dans le graphe résiduel est par définition $r(S,\overline{S})$. Par exemple, considérons le graphe suivant muni d'un flot et de capacités :



(3.14)

On a représenté sur chaque arc les valeurs "$x_{ij} \leqslant u_{ij}$". Ici, la source est $s = 1$, le puits est $p = 6$, et la valeur du flot est 5. On a indiqué en traits gras dans (3.14) les arcs qui appartiennent au graphe résiduel. Le graphe résiduel lui même peut être représenté comme suit avec ses capacités résiduelles :



(3.15)

Le lemme suivant nous permet de vérifier l'optimalité d'un flot.

Lemme 3.19 *Soit x un flot admissible de valeur v. Les assertions suivantes sont équivalentes :*

1. *x est de valeur maximale,*

2. *il n'existe pas de chemin de s à p dans le graphe résiduel $\mathcal{G}_r(x)$,*

3. *il existe une coupe de capacité 0 séparant s de p dans le graphe résiduel $\mathcal{G}_r(x)$,*

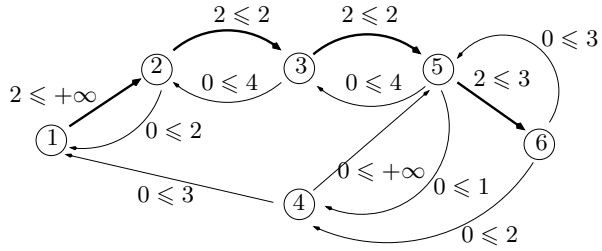
4. *il existe une coupe de capacité v séparant s de p dans le graphe originel.*

Démonstration. Montrons d'abord (non 2)⇒ (non 1). Soit $(\ell_0, \ldots, \ell_k)$ un chemin de s à p dans $\mathcal{G}_r(x)$, que l'on peut toujours supposer élémentaire, et notons $m = \min(r_{\ell_0\ell_1}, \ldots, r_{\ell_{k-1},\ell_k})$ la capacité résiduelle minimale le long du chemin. Pour chaque arc (i, j) du chemin, par définition du graphe résiduel, il est possible d'accroître le flot net de i à j de m en décomposant $m = \alpha_{ij} + \beta_{ji}$, avec $0 \leqslant \alpha_{ij} \leqslant u_{ij} - x_{ij}$ et $0 \leqslant \beta_{ji} \leqslant x_{ji}$, puis, d'une part, en accroissant le flot de α_{ij} sur l'arc (i, j), et d'autre part, en diminuant le flot de β_{ji} sur l'arc (j, i). Formellement, définissons le flot x' :

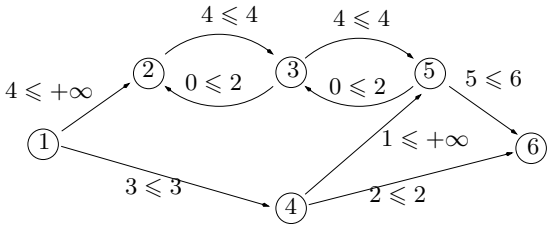
$$x'_{ij} = \begin{cases} x_{ij} + \alpha_{ij} & \text{si } (i,j) \text{ appartient au chemin } (\ell_0, \ldots, \ell_k), \\ x_{ij} - \beta_{ij} & \text{si } (j,i) \text{ appartient au chemin } (\ell_0, \ldots, \ell_k), \\ x_{ij} & \text{sinon.} \end{cases} \tag{3.16}$$

(Comme l'on a supposé le chemin élémentaire, (i, j) et (j, i) ne peuvent tous deux appartenir au chemin, ce qui garantit que les deux premiers cas de (3.16) sont bien exclusifs.) Le flot x' est encore admissible, et sa valeur est $v + m$, ce qui montre que x n'est pas maximal.

Illustrons cette partie de la preuve sur l'exemple (3.14). Dans ce cas, il y a un seul chemin élémentaire de s à p dans $\mathcal{G}_r(x)$, et sa capacité vaut $m = 2$. Ce chemin est représenté comme suit en gras, avec le flot dans le graphe résiduel qui lui est associé :



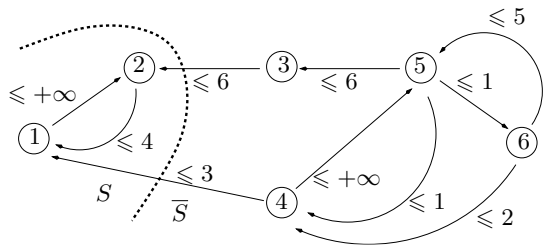
En revenant au graphe initial, partant du flot x de valeur 5, la construction (3.16) fournit le flot x' suivant de valeur 7 :



. (3.17)

Revenons à la preuve, et montrons maintenant (2)⇒(3). Considérons l'ensemble S des nœuds $i \in \mathcal{N}$ tel qu'il existe un chemin de s à i dans le graphe résiduel $\mathcal{G}_r$, et soit $\overline{S} = \mathcal{N} \setminus S$. Par définition de S, $s \in S$. L'assertion 2 entraîne que $p \notin S$. D'autre part, pour tout $i \in S$ et $j \in \overline{S}$ tels que $(i, j) \in \mathcal{A}$, nous avons nécessairement $r_{ij} = 0$: sinon, on aurait $j \in S$, une contradiction. Ainsi, $r(S, \overline{S}) = \sum_{i \in S,\, j \in \overline{S},\, (i,j) \in \mathcal{A}} r_{ij} = 0$.

A titre d'illustration, cet argument appliqué au graphe résiduel du flot (3.17) fournit la coupe suivante :



Montrons ensuite (3)⇒(4). Soit $(S, \overline{S})$ une coupe de capacité nulle séparant s de p dans le graphe résiduel. Notons v la valeur de x. Par définition du graphe résiduel, $u_{ij} = x_{ij}$ et $x_{ji} = 0$ pour tout arc (i, j) du graphe résiduel tel que $i \in S$ et $j \in \overline{S}$, ce qui nous permet de réécrire (3.12) sous la forme

$$\begin{aligned} v &= \sum_{i\in S,\ j\in\overline{S},\ (i,j)\in\mathcal{A}} x_{ij} - \sum_{i\in S,\ j\in\overline{S},\ (j,i)\in\mathcal{A}} x_{ji} \\ &= \sum_{i\in S,\ j\in\overline{S},\ (i,j)\in\mathcal{A}} u_{ij} - \sum_{i\in S,\ j\in\overline{S},\ (j,i)\in\mathcal{A}} 0 = u(S,\overline{S}), \end{aligned}$$

ce qui montre (4).

Enfin, (4)⇒(1) résulte aussitôt du Lemme 3.17. ■

Démonstration du Théorème 3.18. Le calcul de flot maximum est un programme linéaire réalisable puisque le flot nul l'est. Si sa valeur est finie, il a une solution et on conclut avec l'implication (1)⇒(4) du Lemme 3.19. Si elle vaut $+\infty$, on conclut avec le lemme 3.17. ■

3.2.2 Algorithme de Ford et Fulkerson

L'algorithme de Ford et Fulkerson pour calculer le flot admissible maximal dans un graphe consiste à appliquer la preuve du Lemme 3.19 : on part d'un flot admissible x^1 quelconque, par exemple le flot nul, on construit ensuite le graphe résiduel $\mathcal{G}_r(x^1)$ associé à x^1, puis l'on recherche un chemin de s à p dans $\mathcal{G}_r(x^1)$, (ce qui peut se faire en temps linéaire en le nombre d'arcs et de sommets de $\mathcal{G}_r(x^1)$). Si l'on trouve un tel chemin, on construit un nouveau flot x^2 de valeur strictement supérieure à la valeur de x^1, et l'on continue de la sorte. Nous avons donné dans la preuve du Lemme 3.19 une trace de cet algorithme.

Pour appliquer l'algorithme de Ford et Fulkerson en toute généralité, il faut d'abord vérifier que la valeur des flots admissibles de s à p est majorée ce qui peut se faire (toujours en temps linéaire) grâce à l'exercice suivant.

Exercice 3.20 Montrer que la valeur des flots admissibles de s à p n'admet pas de majorant fini si, et seulement si, il existe un chemin de s à p formé d'arcs de capacité infinie.

Nous dirons qu'un flot x est *entier* si $x_{ij} \in \mathcal{N}$, pour tout $(i, j) \in \mathcal{A}$. Nous pouvons maintenant énoncer le résultat de convergence.

Proposition 3.21 *Supposons que les capacités u_{ij} des arcs soient entières ou $+\infty$, supposons en outre qu'il n'existe pas de chemin de s à p dont tous les arcs ont des capacités infinies, et soit v^* la valeur maximale d'un flot admissible de s à p. Alors, l'algorithme de Ford et Fulkerson fournit un flot admissible maximal de s à p, qui est entier, en temps $O(|\mathcal{A}|v^*)$.*

Démonstration. Lorsque les données sont entières, que l'on part d'un flot initial entier, et que l'on applique (3.16) en choisissant des valeurs entières de α_{ij} et β_{ij}, l'algorithme de Ford et Fulkerson produit des flots entiers dont la valeur s'accroît d'au moins une unité à chaque itération. Le nombre d'itérations est donc au plus v^*. Chaque itération prend un temps $O(|\mathcal{A}|)$. ■

Remarque 3.22 La proposition 3.21 permet de déduire le résultat suivant : si u est à valeurs entières ou $+\infty$, et si la valeur des flots admissibles est majorée alors la valeur maximale des flots de s à p est atteinte par un flot entier. Nous avons donc retrouvé de manière algorithmique, dans un cas particulier, le théorème d'optimalité des flots entiers (Corollaire 3.14).

Remarque 3.23 (Approche via la programmation linéaire) La notion de coupe se retrouve à l'aide du théorème de dualité en programmation linéaire. Le problème du flot maximal admissible de s à p consiste en effet à maximiser la valeur v sur l'ensemble des couples (x, v) dans $\mathbb{R}^{\mathcal{A}} \times \mathbb{R}$, tels que $x \geqslant 0$, tel que les lois des nœuds $f_i(x) = 0$ pour $i \in \mathcal{N} \setminus \{s, p\}$, $v = f_s(x)$, et $-v = f_p(x)$, soient vérifiées, ainsi que $x_{ij} \leqslant u_{ij}$. Nous supposons toutes les capacités u_{ij} finies, afin d'alléger la notation. Notons $y = (y_i) \in \mathbb{R}^{\mathcal{N}}$ le multiplicateur associé aux lois des nœuds, $z = (z_{ij}) \in \mathbb{R}^{\mathcal{N}}_+$ le multiplicateur associé aux contraintes $x_{ij} \leqslant u_{ij}$, et introduisons le Lagrangien

$$\begin{aligned} L(x, v; y, z) = \quad & v + \sum_{i \in \mathcal{N} \setminus \{s,p\}} y_i f_i(x) + y_s(f_s(x) - v) + y_p(f_p(x) + v) \\ & + \sum_{(i,j) \in \mathcal{A}} z_{ij}(u_{ij} - x_{ij}) \ , \end{aligned}$$

de sorte que le problème du flot admissible maximum de s à p peut s'écrire :

$$\sup_{x \in \mathbb{R}^{\mathcal{A}}_+, v \in \mathbb{R}} \quad \inf_{y \in \mathbb{R}^{\mathcal{N}},\ z \in \mathbb{R}^{\mathcal{A}}_+} L(x, v; y, z) \ .$$

Le problème dual s'obtient en commutant le sup et l'inf. En faisant apparaître les variables x_{ij} et v en facteur, on obtient la forme suivante du problème dual :

$$\inf_{y \in \mathbb{R}^{\mathcal{N}},\ z \in \mathbb{R}^{\mathcal{A}}_+} \quad \sup_{x \in \mathbb{R}^{\mathcal{A}}_+, v \in \mathbb{R}} \sum_{(i,j) \in \mathcal{A}} x_{ij}(y_i - z_{ij} - y_j) + v(1 - y_s + y_p) + \sum_{(i,j) \in \mathcal{A}} u_{ij} z_{ij}, \tag{3.18}$$

soit

$$(\mathsf{D}) \qquad \inf_{y \in \mathbb{R}^{\mathcal{N}},\ z \in \mathbb{R}^{\mathcal{A}}_+} \sum_{(i,j) \in \mathcal{A}} u_{ij} z_{ij}; \ \ y_i \leqslant z_{ij} + y_j, \ \forall (i,j) \in \mathcal{A}, \ \ 1 + y_p = y_s \ .$$

Soit (y, z) une solution duale. Ajoutons la contrainte $\pm y_i \leqslant M$, où $M \in \mathbb{N}$, $M > \max_i |y_i|$. Alors l'ensemble admissible dual ne contient pas de droite affine. La matrice du problème dual étant totalement unimodulaire (appliquer la proposition 2.37 à la transposée de cette matrice), le corollaire 2.34 assure l'existence d'une solution duale (y, z) entière. L'ensemble des solutions duales étant invariant par ajout d'une même constante à chaque composante de y, on peut supposer que $y_p = 0$ et $y_s = 1$.

Définissons $(y', z') \in \mathbb{R}^{\mathcal{N}} \times \mathbb{R}^{\mathcal{A}}_+$ par $y'_i := \max(0, \min(1, y_i))$, pour tout $i \in \mathbb{R}^{\mathcal{N}}$, et pour tout $ij \in \mathcal{A}$, $z'_{ij} := (y'_i - y'_j)_+$. Alors (y', z') est admissible, et $z'_{ij} \leqslant z_{ij}$ pour tout $ij \in \mathcal{A}$, donc $\sum_{(i,j) \in \mathcal{A}} u_{ij} z'_{ij} \leqslant \sum_{(i,j) \in \mathcal{A}} u_{ij} z_{ij}$, ce qui prouve que (y', z') est solution duale. Nous avons construit une solution duale de composantes dans $\{0, 1\}$.

On retrouve les coupes séparant s de p à partir de cette formulation. En effet, soit (y, z) une solution duale de composantes dans $\{0, 1\}$. Posons $S = \{i \in \mathcal{N} \mid \ y_i = 1\}$, et $\bar{S} = \mathcal{N} \setminus S$. On voit qu'on a nécessairement $s \in S$ et $p \in \bar{S}$, et que le plus petit vecteur z vérifiant $y_i \leqslant z_{ij} + y_j$ n'est autre que le vecteur indicateur z de la coupe $(S, \bar{S})$, qui est tel que $z_{ij} = 1$ si $i \in S$, $j \in \bar{S}$ et $z_{ij} = 0$ sinon. En outre, $\sum_{(i,j) \in \mathcal{A}} u_{ij} z_{ij}$ n'est autre que la capacité $u(S, \bar{S})$ de la coupe $(S, \bar{S})$.

Nous retrouvons ainsi comme conséquence du théorème de dualité faible en programmation linéaire le fait que la valeur du flot maximal admissible de s à p est majorée par la capacité de toute coupe séparant s de p. En fait, le problème dual (D) admet une solution (y, z) telle que z soit la fonction indicatrice d'une coupe minimale. En effet, si $(S, \bar{S})$ est une coupe de capacité minimale, à laquelle sont associés les vecteurs y, z définis ci-dessus, la valeur correspondante du critère du problème dual (D) est $u(S, \bar{S})$, qui est égal à la valeur maximale d'un flot x, d'après le théorème de Ford-Fulkerson, autrement dit, à la valeur du problème primal, ce qui montre que (y, z) est une solution optimale du problème dual.

Exercice 3.24 (Choix de représentants et problèmes de flots) Une ville a r citoyens $M_1, \ldots, M_r$, q clubs $C_1, \ldots, C_q$, et n partis politiques $P_1, \ldots, P_n$. (Rappelons qu'un club, ou un parti politique, est un ensemble de citoyens.) On suppose que chaque citoyen est membre d'au moins un club, et d'au plus un parti politique. Chaque club désigne l'un de ses membres pour le représenter au conseil municipal. Si un membre du conseil municipal représente plusieurs clubs, il a autant de voix que de clubs qui l'ont désigné (par exemple s'il est désigné par deux clubs, son avis compte double). On voudrait savoir s'il est possible de choisir de tels représentants de sorte que pour tout $1 \leqslant i \leqslant n$, le nombre total des voix détenues par les membres du conseil municipal appartenant au parti P_i n'excède pas un quota u_i fixé a priori.

1. Montrer que le problème plus général suivant : "trouver le nombre maximal $N \leqslant q$ de clubs qui peuvent choisir chacun un représentant de sorte que, pour tout $1 \leqslant i \leqslant n$, le nombre total de voix détenues par les membres du conseil municipal appartenant au parti P_i n'excède pas u_i" est équivalent à un problème de flot maximal dans un certain graphe que l'on décrira. On pourra notamment expliciter le graphe pour l'exemple suivant : $r = 7$, $q = 4$, $n = 3$, $C_1 = \{M_1, M_3\}$, $C_2 = \{M_1, M_2, M_3\}$, $C_3 = \{M_2, M_3, M_4\}$, $C_4 = \{M_2, M_4, M_5, M_6, M_7\}$, $P_1 = \{M_1, M_2\}$, $P_2 = \{M_3, M_4\}$, $P_3 = \{M_5, M_6, M_7\}$, $u_1 = 1$, $u_2 = 1$, $u_3 = 2$.

2. Démontrer que pour l'exemple qui précède, on a $N < 4$ (au moins un club ne peut choisir de représentant). Indication : on pourra utiliser le théorème max-flot=min-cut.

3. Peut-on modéliser par un problème de flot la variante dans laquelle deux clubs ne peuvent pas désigner le même représentant ?

Exercice 3.25 (Un problème de coupe minimale apparaissant en traitement d'image). On représente un dessin en noir et blanc avec n pixels par une application $\{1, \ldots, n\} \to \{\pm 1\}$: la valeur d'un pixel est -1 ou $+1$ (pour blanc et noir).

Soit $\alpha : \{1, \ldots, n\} \to \{\pm 1\}, i \mapsto \alpha_i$ une image initiale, qui peut être dégradée. L'image restaurée $\sigma : \{1, \ldots, n\} \to \{\pm 1\}, i \mapsto \sigma_i$ est obtenue en minimisant une énergie du type :

$$J(\sigma) = \sum_{1 \leqslant i \leqslant n} -\alpha_i \sigma_i - \sum_{1 \leqslant i,j \leqslant n} \gamma_{ij} \sigma_i \sigma_j \ ,$$

où les $\gamma_{ij} \geqslant 0$ sont donnés. (En pratique, $\gamma_{ij} > 0$ si i et j sont des pixels voisins, et $\gamma_{ij} = 0$ sinon.)

En effet, dans ce problème de minimisation, le terme $-\alpha_i \sigma_i$ incite la valeur reconstruite σ_i à coïncider avec la valeur initiale α_i, alors que, lorsque $\gamma_{ij} > 0$, le terme $\gamma_{ij} \sigma_i \sigma_j$, incite les valeurs σ_i et σ_j a être identiques (régularisation).

Nous allons montrer que la minimisation de J se réduit au calcul d'un flot maximum dans un graphe.

1. Dire pourquoi il n'y a pas de perte de généralité à supposer que $\gamma_{ij} = \gamma_{ji}$, ce que l'on fera désormais.

2. À toute image σ, on associe la partition de $\{1, \ldots, n\}$ formée des deux ensembles :

$$T^+ = \{i \mid \sigma_i = 1\}, \qquad T^- = \{i \mid \sigma_i = -1\}$$

On forme le graphe orienté complet dont les sommets sont $1, \ldots, n$ (complet signifie que pour chaque $1 \leqslant i, j \leqslant n$, l'arc orienté (i, j) est présent), l'arc (i, j) étant muni de la capacité γ_{ij}.

Montrer que

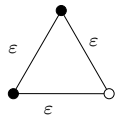
$$\sum_{1 \leqslant i,j \leqslant n} \gamma_{ij}(1 - \sigma_i \sigma_j)/2 = 2C(T^+, T^-)$$

où $C(T^+, T^-)$ désigne la capacité de la coupe (T^+, T^-) (avec la convention que la capacité est nulle si T^+ ou T^- est vide).

3. Montrer qu'en rajoutant un nœud source et un nœud puits au graphe, reliés au nœuds $1, \ldots, n$ par des arcs de capacités bien choisies, on peut interpréter $J(\sigma)$ comme la capacité d'une coupe, à des facteurs additifs et multiplicatifs près, et conclure que la minimisation de l'énergie J se ramène au calcul d'une coupe minimale séparant s de p.

Signalons que ce résultat n'a plus lieu si certains des γ_{ij} sont strictement négatifs. En particulier, lorsque tous les γ_{ij} sont négatifs, on obtient le problème "MAXCUT" (maximiser la capacité une coupe au lieu de la minimiser), qui est NP-dur. (Les problèmes de maximisation et de minimisation ne sont pas équivalents via un changement de signe, car les capacités des arcs sont toujours supposées positives ou nulles.)

4. Discuter la régularisation de la petite image suivante (3 pixels voisins dont deux noirs et un blanc) en fonction de la valeur de $\gamma_{ij} \equiv \epsilon > 0$, $\forall 1 \leqslant i, j \leqslant 3$.



3.2.3 Algorithme d'accroissement sur le chemin le plus court

L'algorithme d'accroissement du flot sur le chemin le plus court, dû à *Dinits* et à *Edmonds et Karp*, est une variante de l'algorithme de Ford et Fulkerson, qui permet de remplacer la borne pseudo-polynomiale de la proposition 3.21 par une véritable borne polynomiale. Ce raffinement consiste tout simplement à sélectionner à chaque étape de l'algorithme de Ford et Fulkerson, le plus court chemin de s à p, c'est-à-dire le chemin de s à p qui a le plus petit nombre d'arcs.

Théorème 3.26 *L'algorithme d'accroissement du flot sur le chemin le plus court effectue au plus* $|\mathcal{N}||\mathcal{A}|$ *accroissements.*

Démonstration. Notons $d_x(i,j)$ la distance de i à j dans le graphe résiduel $\mathcal{G}_r(x)$. On pose $d_x(i,j) = +\infty$ s'il n'existe pas de chemin de i à j. On prétend que si l'on passe de x à x' en effectuant l'accroissement (3.16) sur un chemin de longueur minimale, alors

$$d_x(s,p) \leqslant d_{x'}(s,p) \ . \tag{3.19}$$

Pour voir cela, introduisons, pour $k \geqslant 0$,

$$\mathcal{N}_k = \{j \in \mathcal{N} \mid \ d_x(s,j) = k\} \ ,$$

l'ensemble des nœuds à distance k de s, soit $\Delta = \max\{d_x(s,j) \mid \ j \in \mathcal{N}\}$, et $D = d_x(s,p)$. On a la partition

$$\mathcal{N} = \mathcal{N}_0 \cup \mathcal{N}_1 \cup \cdots \cup \mathcal{N}_\Delta, \quad \text{avec } s \in \mathcal{N}_0, \ p \in \mathcal{N}_D \ .$$

Par construction de la partition, si (i,j) est un arc de $\mathcal{G}_r(x)$, et si $i \in \mathcal{N}_k$, alors $j \in \mathcal{N}_0 \cup \cdots \cup \mathcal{N}_{k+1}$.

Nous savons que x' est construit à partir de x en accroissant le flot sur un chemin $(\ell_0, \ldots, \ell_D)$ de longueur $D = d_x(s,p)$, avec $\ell_0 = s$ et $\ell_D = p$. On a nécessairement

$$\ell_i \in \mathcal{N}_i, \quad \text{pour } i = 0, \ldots, D \ . \tag{3.20}$$

Pour que $d_{x'}(s,p) < d_x(s,p)$, il faudrait que le graphe résiduel de x' contienne au moins un arc allant d'un ensemble $\mathcal{N}_k$ à un ensemble $\mathcal{N}_l$, avec $l \geqslant k+2$, mais vu la définition de x' à partir de x donnée dans (3.16), cela n'est pas le cas : les nouveaux arcs créés sont en effet de la forme (ℓ_i, ℓ_{i-1}), avec $\ell_i \in \mathcal{N}_i$ et $\ell_{i-1} \in \mathcal{N}_{i-1}$. Nous venons de montrer (3.19).

Pour conclure l'analyse, il reste à considérer le cas où $d_{x'}(s,p) = d_x(s,p) = D$. Soit $\mathcal{G}^*(x)$ le graphe obtenu comme union des chemins de s à p de longueur minimale. Comme tout accroissement du flot le long d'un chemin $(\ell_0, \ldots, \ell_D)$ fait disparaître au moins un arc de ce chemin, et apparaître seulement des arcs de la forme (ℓ_i, ℓ_{i-1}), il est clair que $\mathcal{G}^*(x')$ est strictement inclus dans $\mathcal{G}^*(x)$. Il ne peut donc y avoir qu'au plus $|\mathcal{A}|$ accroissements consécutifs à distance de s à p constante dans le graphe résiduel. Comme le nombre d'accroissements de la distance de s à p dans le graphe résiduel est au plus $|\mathcal{N}|$, on en déduit que le nombre d'accroissements de flot effectué par l'algorithme est au plus $|\mathcal{N}||\mathcal{A}|$. ■

Remarque 3.27 L'algorithme d'accroissement du flot sur le chemin le plus court peut s'implémenter en temps $O(|\mathcal{N}||\mathcal{A}|^2)$. Voir [AMO93]. •

Remarque 3.28 Un algorithme de flot très différent, le "preflow-push" algorithm de Goldberg et Tarjan (1986), qui procède en manipulant une suite de vecteurs de "pré-flots" qui peuvent violer la loi des nœuds, a un temps d'exécution de $O(|\mathcal{N}|^2|\mathcal{A}|)$, ce qui est mieux que le temps de $O(|\mathcal{N}||\mathcal{A}|^2)$ pour l'algorithme d'accroissement du flot sur le chemin le plus court. (Pour des graphes très creux, $|\mathcal{N}|$ et $|\mathcal{A}|$ sont du même ordre de grandeur, mais pour des graphes pleins, $|\mathcal{A}|$ est d'ordre $|\mathcal{N}|^2$.) Cela dit, le temps d'exécution dans le pire des cas n'est pas le seul critère de sélection d'un algorithme de flot. (Par exemple, pour rester élémentaire, la borne de la Proposition 3.21 montre que l'algorithme de Ford et Fulkerson, presque immédiat à programmer, sera très efficace lorsque les capacités sont entières et "petites".) Voir [AMO93] pour un état de l'art. •

3.2.4 Problèmes de flot à coût minimum

Nous avons maintenant tous les outils pour traiter le problème général de flot à coût minimum (3.5). La proposition suivante caractérise le cas où le coût des flots est minoré.

Proposition 3.29 *Supposons que le programme de flot à coût minimum* (3.5) *admette au moins un flot admissible. Alors, la valeur de ce programme est finie si, et seulement si, il n'existe pas de circuit formé uniquement d'arcs de capacité infinie, et de coût total strictement négatif.*

Démonstration. D'après la construction de la Remarque 3.9, on peut se limiter au cas où le graphe a un nœud source s et un nœud puits p, avec $b_i = 0$ si $i \notin \{s, p\}$. D'après le résultat de décomposition des flots (proposition 3.10), tout flot va s'écrire comme combinaison linéaire positive de flots associés à des chemins élémentaires de s à p, et de flots associés à des circuits. La proposition en résulte. ■

Remarque 3.30 Nous verrons dans le chapitre sur la programmation dynamique que l'existence d'un circuit de coût total strictement négatif peut se vérifier en temps $O(|\mathcal{N}||\mathcal{A}|)$ par l'algorithme de Ford-Bellman (exercice 4.22).

Exercice 3.31 Montrer que l'on peut vérifier l'existence d'un flot admissible en résolvant un problème de flot de valeur maximale auxiliaire (on rajoutera une nœud source et un nœud puits, avec des arcs et des capacités convenables), et que, si les capacités sont entières, il existe un flot admissible entier.

Nous supposons dans toute la suite que la valeur du programme de flot à coût minimum est finie.

Pour présenter l'algorithme le plus simple de flot à coût minimum, nous devons définir le coût d'un chemin du graphe résiduel $\mathcal{G}_r(x)$. Si (i, j) est un arc de $\mathcal{G}_r(x)$, nous définissons le coût $c_{ij}(x)$ de l'arc (i, j) dans $\mathcal{G}_r(x)$

$$c_{ij}(x) = \begin{cases} \min(c_{ij}, -c_{ji}) & \text{si } x_{ij} < u_{ij} \text{ et } x_{ji} > 0, \\ c_{ij} & \text{si } x_{ij} < u_{ij} \text{ et } x_{ji} = 0 \\ -c_{ji} & \text{si } x_{ij} = u_{ij} \text{ et } x_{ji} > 0. \end{cases} \tag{3.21}$$

La raison de cette définition est que si nous prenons un chemin élémentaire $(\ell_0, \ldots, \ell_k)$ de s à p dans $\mathcal{G}_r(x)$, il est possible de définir un nouveau flot x' par (3.16), en choisissant pour tout arc (i, j) du chemin α_{ij} et β_{ij} tels que $\alpha_{ij} + \beta_{ji} = \epsilon > 0$, avec ϵ assez petit, de sorte que le coût du flot x' soit égal au coût de x augmenté de ϵ fois le coût du chemin $(\ell_0, \ldots, \ell_k)$ dans $\mathcal{G}_r(x)$.

Le résultat suivant nous permet de vérifier l'optimalité d'un flot admissible.

Théorème 3.32 *Un flot admissible x est de coût minimal si, et seulement si, le graphe résiduel $\mathcal{G}_r(x)$ ne contient aucun circuit de coût total strictement négatif.*

Démonstration. Nous allons déduire ce résultat du théorème de dualité en programmation linéaire.

Nous avons déjà vu dans la Remarque 3.23 un calcul pratique de problème dual. En s'inspirant de ce calcul, et en prenant garde aux signes, car il s'agit maintenant d'un problème de minimisation, et non de maximisation, on forme le Lagrangien :

$$L(x;y,z) = \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \sum_{i\in\mathcal{N}} y_i(b_i - f_i(x)) + \sum_{(i,j)\in\overline{\mathcal{A}}} z_{ij}(x_{ij} - u_{ij}) \ .$$

Le vecteur b est défini comme en (3.1), on a posé $\overline{\mathcal{A}} = \{(i,j) \in \mathcal{A} \mid u_{ij} < +\infty\}$, $y \in \mathbb{R}^{\mathcal{N}}$ est le multiplicateur de Lagrange associé à la loi des nœuds (3.1), et $z \in \mathbb{R}^{\mathcal{A}}_+$ est le multiplicateur de Lagrange associé à la contrainte de capacité $x_{ij} \leqslant u_{ij}$. On convient dans la suite que $z_{ij} = 0$ si $(i,j) \in \mathcal{A} \setminus \overline{\mathcal{A}}$ (c'est là une simple commodité de notation).

Raisonnant comme dans la remarque 3.23, on montre que le dual du programme linéaire (3.5) vaut :

$$\begin{array}{cc} \sup & \sum_{i\in\mathcal{N}} b_i y_i - \sum_{(i,j)\in\overline{\mathcal{A}}} u_{ij} z_{ij} \\ -z_{ij} \leqslant -y_i + c_{ij} + y_j, \ \forall (i,j) \in \mathcal{A} \ , & \\ z_{ij} \geqslant 0, \ \forall (i,j) \in \overline{\mathcal{A}} \ , & \end{array} \tag{3.22}$$

Il résulte du théorème de dualité forte en programmation linéaire que le flot admissible x est de coût minimum si, et seulement si, il existe un point admissible (y,z) du problème dual, tel que la condition des écarts complémentaires soit satisfaite :

$$x_{ij}(-y_i + c_{ij} + y_j + z_{ij}) = 0 \qquad \forall (i,j) \in \mathcal{A}, \quad \text{et} \tag{3.23}$$

$$(u_{ij} - x_{ij})z_{ij} = 0 \qquad \forall (i,j) \in \overline{\mathcal{A}} \tag{3.24}$$

(voir à ce sujet l'exemple 2.51). Autrement dit, un flot admissible x est optimal si, et seulement si, il existe $y \in \mathbb{R}^{\mathcal{N}}$ et $z \in \mathbb{R}^{\mathcal{A}}$, avec $z_{ij} = 0$ pour $(i,j) \in \mathcal{A} \setminus \overline{\mathcal{A}}$, tels que (y,z) soit admissible, c'est-à-dire,

$$-z_{ij} \leqslant -y_i + c_{ij} + y_j, \ \forall (i,j) \in \mathcal{A}, \qquad z_{ij} \geqslant 0, \ \ \forall (i,j) \in \overline{\mathcal{A}}, \tag{3.25}$$

et

$$x_{ij} > 0 \quad \Rightarrow \quad -y_i + c_{ij} + y_j = -z_{ij} \ , \tag{3.26}$$

$$x_{ij} < u_{ij} \quad \Rightarrow \quad z_{ij} = 0 \ . \tag{3.27}$$

On prétend que l'optimalité de x est équivalente à l'existence d'un vecteur $y \in \mathbb{R}^{\mathcal{N}}$ tel que

$$-y_i + c_{ij}(x) + y_j \geqslant 0 \ , \ \text{ pour tout arc } (i,j) \text{ de } \mathcal{G}_r(x) \ . \tag{3.28}$$

Pour montrer cela, compte tenu de la définition (3.21) de $c(x)$, il suffit de vérifier qu'à y fixé, l'existence d'un z vérifiant (3.25)–(3.27) est équivalente à

$$x_{ij} > 0 \quad \Rightarrow \quad y_i - c_{ij} - y_j \geqslant 0 \tag{3.29}$$

$$x_{ij} < u_{ij} \quad \Rightarrow \quad y_i - c_{ij} - y_j \leqslant 0 \tag{3.30}$$

Or (3.29) résulte de (3.26) et de $z_{ij} \geqslant 0$, et (3.30) résulte de (3.27) et de la première inégalité de (3.25). Réciproquement, si y satisfait (3.29)–(3.30), on pose

$$z_{ij} = \max(0, \ y_i - c_{ij} - y_j) \ ,$$

qui vérifie bien (3.25)–(3.27). On a montré l'équivalence annoncée.

Pour conclure la preuve du théorème, il suffit de montrer le lemme suivant. ■

Lemme 3.33 *L'existence d'un y vérifiant* (3.28) *est équivalente à l'absence de circuits de coût total strictement négatif dans $\mathcal{G}_r(x)$.*

Démonstration. Notons $a_{ij} := c_{ij}(x)$ pour simplifier les notations. Soit $(i_1, \ldots, i_k, i_1)$ un circuit de $\mathcal{G}_r(x)$. S'il existe un tel vecteur y vérifiant (3.28), on a $-y_{i_1} + a_{i_1 i_2} + y_{i_2} \geqslant 0$, $\ldots$, $-y_{i_k} + a_{i_k i_1} + y_{i_1} \geqslant 0$, et en sommant ces inégalités, $a_{i_1 i_2} + \cdots + a_{i_k i_1} \geqslant 0$, ce qui montre que tout circuit de $\mathcal{G}_r(x)$ a un côut positif ou nul. Montrons la réciproque, en considérant tout d'abord le cas particulier où il existe un nœud p tel qu'il existe un chemin de tout autre nœud à celui ci, dans le graphe $\mathcal{G}_r(x)$. Désignons alors par y_i le poids minimum d'un chemin de i à p, et observons que si tout circuit a un poids positif ou nul, ce minimum est atteint pour un chemin élémentaire, et est donc fini. On a par construction $y_i \leqslant a_{ij} + y_j$. Dans le cas général, on choisit un nœud p quelconque, et l'on définit y_i comme le poids minimum d'un chemin de i à p seulement quand il existe un chemin de i à p. Il reste alors des nœuds j pour lesquels y_j n'est pas défini. Il suffit de choisir un nouveau nœud p parmi ces j, et de recommencer, jusqu'à ce qu'on ait défini un vecteur $y \in \mathbb{R}^{\mathcal{N}}$ vérifiant $y_i \geqslant a_{ij} + y_j$ pour tout arc (i, j) de $\mathcal{G}_r(x)$. ■

On déduit aussitôt du Théorème 3.32 l'algorithme suivant, dit d'*accroissement des circuits*, pour résoudre un problème de flot à coût minimum. On part d'un flot admissible initial x, et l'on construit le graphe résiduel $\mathcal{G}_r(x)$. On recherche alors un circuit de coût strictement négatif dans $\mathcal{G}_r(x)$, ce qui peut se faire à l'aide de l'algorithme de Ford-Bellman décrit au chapitre traitant de la programmation dynamique. Si on en trouve un, de deux choses l'une : soit toutes les capacités u_{ij} des arcs (i, j) du circuit sont infinies, auquel cas la valeur du problème de flot est $-\infty$, soit l'une de ces capacités est finie. Dans ce dernier cas, en augmentant la valeur du flot x sur le circuit, autant que le permettent les capacités, on parvient à un nouveau flot admissible x', pour lequel on réexécute ce qui précède. Si au contraire on ne trouve pas de circuit de coût strictement négatif, l'algorithme s'arrête : on a trouvé la solution optimale.

Lorsque les capacités u_{ij} sont entières ou infinies, et que les flots exogènes b_i sont entiers, cet algorithme termine en un nombre fini d'étapes. En effet, il est possible (exercice 3.31) d'initialiser l'algorithme avec un flot admissible entier, et d'augmenter les valeurs des flots sur les circuits en faisant en sorte de ne produire que des flots entiers. Observons en outre qu'il n'y a qu'un nombre fini de circuits élémentaires et de valeurs possibles des coûts dans tous les graphes résiduels de la forme $\mathcal{G}_r(x)$, et notons donc $\delta < 0$ le coût maximum d'un circuit élémentaire de coût strictement négatif dans l'un de ces graphes résiduels. On voit que le coût diminue d'au moins δ à chaque itération. La convergence en temps fini de l'algorithme en résulte.

Remarque 3.34 La terminaison de l'algorithme d'accroissement des circuits permet de déduire que si la valeur du problème de flot à coût minimum (3.5) est finie, si les capacités u_{ij} sont entières ou infinies, et si v est entier, alors (3.5) a une solution optimale entière. On retrouve ainsi de manière algorithmique le résultat général d'optimalité des flots entiers (Corollaire 3.14).

Remarque 3.35 Dans le cas de l'algorithme de Ford et Fulkerson, on a vu qu'une stratégie judicieuse de choix du chemin augmentant fournissait un algorithme polynomial (cf. Théorème 3.26). Dans le cas des flots à coût minimum, on obtient un algorithme polynomial si l'on choisit à chaque fois le circuit dont le ratio coût/longueur est minimum [AMO93]. •

3.3 Notes

Une référence très complète pour les algorithmes de flots est l'ouvrage de R.K. Ahuja, T.L Magnanti et J.B. Orlin [AMO93]. Celui-ci couvre notamment les algorithmes de "préflots", que nous n'avons pas discuté ici.

Chapitre 4

Programmation dynamique déterministe discrète et problèmes du plus court chemin

La programmation dynamique, développée par Richard Bellman dans les années 50, est une méthode très générale qui s'applique aux problèmes de décision dans le temps, c'est-à-dire à des situations où une suite de décisions doit être prise (la variable de temps étant parfois déguisée). Le domaine d'application immédiat de la programmation dynamique concerne les questions de plus court chemin. Elle apparaît aussi naturellement dans l'étude de divers problèmes de nature économique (gestion de stock, problèmes de maintenance, ...). Elle fournit enfin des algorithmes assez efficaces (pseudo-polynomiaux) adaptés à des problèmes combinatoires avec un petit nombre de contraintes, tels que le problème de sac à dos.

La programmation dynamique, qui est profondément reliée à l'approche Markovienne en théorie des probabilités, repose sur la notion *d'état*. Un état résume l'ensemble des données pertinentes pour résoudre le problème. A chaque état on associe une valeur optimale partant de cet état, et l'équation de programmation dynamique relie la valeur d'un état à un instant donné à celles des états auxquels on peut accéder à l'instant suivant.

Nous nous limitons dans ce chapitre aux questions de programmation dynamique déterministe avec espace de temps discret. Ce cadre permet de voir les idées essentielles avec un minimum de complications techniques. L'approche de ce chapitre peut s'étendre de plusieurs manières : traitement du temps continu (équation d'Hamilton-Jacobi), programmation dynamique stochastique, et cas à deux joueurs (jeux à somme nulle).

4.1 Cadre général

Les divers problèmes de programmation dynamique (déterministe) peuvent se poser dans le cadre abstrait suivant.

(i) On se donne un ensemble d'*états* X, éventuellement infini. (ii) Pour chaque couple d'états $(x, y) \in X \times X$, on se donne un *gain de transition* de x à y, $G_{x,y} \in \mathbb{R} \cup \{-\infty\}$. On convient de poser $G_{x,y} = -\infty$ lorsque la transition de x à y est interdite (dans le cas contraire, la transition est dite autorisée) (iii) On fixe un *état initial* $x_0 \in X$. (iv) Enfin, pour chaque $y \in X$, on se donne une *prime finale*, $b_y \in \mathbb{R} \cup \{-\infty\}$. On convient de poser $b_y = -\infty$ lorsque qu'il est interdit de finir dans l'état y.

On peut distinguer plusieurs types de problème.

Problème 4.1 (en horizon N) *On souhaite maximiser le revenu en N étapes, partant de l'état initial x_0 :*

$$\operatorname*{Max}_{x_1,\dots,x_N \in X} G_{x_0 x_1} + \cdots + G_{x_{N-1} x_N} + b_{x_N} \quad . \tag{4.1}$$

Problème 4.2 (avec temps d'arrêt) *On souhaite maximiser le revenu, partant de l'état initial x_0, le nombre N d'étapes étant quelconque :*

$$\operatorname*{Max}_{N\geqslant 0,\ x_1,\dots,x_N\in X} G_{x_0x_1}+\cdots+G_{x_{N-1}x_N}+b_{x_N}\quad . \tag{4.2}$$

Exemple 4.3 (Plus court chemin) Le problème du plus court chemin, déjà évoqué dans l'exemple 1.13, est un cas particulier du problème 4.2. Soit en effet $(\mathcal{N},\mathcal{A})$ un graphe orienté dont chaque arc (i,j) est muni d'un coût c_{ij}, et soient o et d une origine et une destination prescrites. Chercher le chemin de coût minimum de o à d revient à résoudre (4.2) avec $X=\mathcal{N}$, $G_{ij}=-c_{ij}$ pour $(i,j)\in\mathcal{A}$, et $G_{ij}=-\infty$ pour $(i,j)\notin\mathcal{A}$, $x_0=o$. Il suffit de poser $b_d=0$ et $b_j=-\infty$ pour $j\neq d$ pour interdire au chemin de s'arrêter ailleurs qu'en la destination d. L'entier N représente la longueur du chemin.

Problème 4.4 (avec gain total) *On maximise maintenant sur l'ensemble des suites infinies d'états partant de x_0 :*

$$\operatorname*{Max}_{x_1,x_2,\dots\in X} G_{x_0x_1}+G_{x_1x_2}+\cdots$$

La prime finale n'intervient plus dans ce dernier problème. On prendra garde que la série $G_{x_0x_1}+G_{x_1x_2}+\cdots$ peut être divergente, à moins que l'on ne fasse des hypothèses sur le gain de transition $(x,y)\mapsto G_{xy}$. L'hypothèse la plus simple consiste à supposer que $G_{xy}\leqslant 0$, pour tout $(x,y)\in X\times X$. Dans ce cas, la somme $G_{x_0x_1}+G_{x_1x_2}+\cdots$ est définie sans ambiguïté. Une autre manière de résoudre cette difficulté consiste à considérer la variante suivante.

Problème 4.5 (avec facteur d'actualisation) *On se donne un* facteur d'actualisation $0<\alpha<1$*, et l'on maximise sur l'ensemble des suites infinies d'états partant de x_0 la somme des gains actualisés :*

$$\operatorname*{Max}_{x_1,x_2,\dots\in X} G_{x_0x_1}+\alpha G_{x_1x_2}+\alpha^2 G_{x_2x_3}\cdots$$

On notera que la somme infinie $G_{x_0x_1}+\alpha G_{x_1x_2}+\alpha^2 G_{x_2x_3}\cdots$ a un sens dès que la fonction $(x,y)\mapsto G_{x,y}$ est majorée, ce qui est le cas en particulier lorsque l'ensemble X des états est fini.

Un facteur d'actualisation α trop petit conduit à des stratégies optimales myopes, pour lesquelles seules les gains des premiers pas de temps importent. La prise en compte de considérations de type "développement durable" conduit à donner aux gains futurs la même importance qu'aux gains actuels. Ceci peut être modélisé en considérant la limite quand α tend vers 1 du problème actualisée, ou plus directement, en considérant le critère suivant, dit ergodique.

Problème 4.6 (avec critère ergodique) *On cherche maintenant à maximiser le gain moyen par unité de temps :*

$$\operatorname*{Max}_{x_1,x_2,\dots\in X} \limsup_{N\to\infty} \frac{G_{x_0x_1}+\cdots+G_{x_{N-1}x_N}}{N}\quad .$$

Exemple 4.7 Pour illustrer ce qui précède, considérons le cas d'un chauffeur de taxi maraudant dans la ville imaginaire représentée sur la Figure 4.1. La ville est formée de trois zones, H, un quartier huppé, A un aéroport, et B une banlieue (d'où le chauffeur revient en général à vide), ce que l'on peut modéliser en prenant pour ensemble d'états $X=\{H,A,B\}$. La fonction de gain de transition $(x,y)\mapsto G_{xy}$ et la prime $x\mapsto b_x$ peuvent s'écrire comme suit, sous forme matricielle :

$$G=\begin{array}{c} \\ H\\ A\\ B\end{array}\!\!\begin{array}{c}\begin{array}{ccc} H & A & B\end{array}\\ \begin{pmatrix} 3 & 10 & -\infty\\ 12 & -\infty & 7\\ -5 & -3 & -\infty\end{pmatrix}\end{array},\qquad b=\begin{array}{c} \\ H\\ A\\ B\end{array}\!\!\begin{array}{c}\\ \begin{pmatrix} 0\\ 2\\ -\infty\end{pmatrix}\end{array}.$$

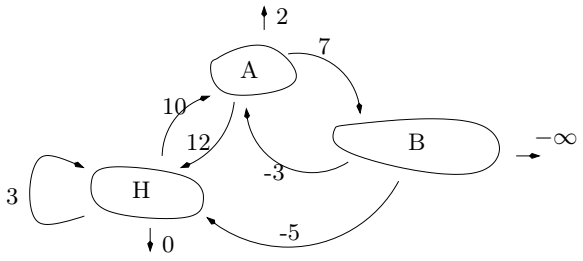


FIGURE 4.1 – Un chauffeur de taxi très déterministe

Exemple 4.8 Les problèmes de programmation dynamique apparaissent souvent dans le formalisme suivant. Considérons un système contrôlé abstrait :

$$x_{k+1} = f(x_k, u_k),\ u_k \in U,$$

où $x_k \in X$ est l'état à l'instant k, l'état initial $x_0 \in X$ est fixé, u_k est le contrôle à l'instant k, U est l'ensemble des valeurs admissibles du contrôle. L'application $f : X \times U \to X$ est la dynamique (discrète). On se donne un *gain* (appelé *Lagrangien* dans le contexte du contrôle optimal) $\ell : X \times U \to \mathbb{R}$, $(x, u) \mapsto \ell(x, u)$, ainsi qu'une prime finale $\phi : \mathcal{X} \to \mathbb{R}$. On cherche à trouver une suite de contrôles $(u_0, \ldots, u_{N-1})$ maximisant un gain en horizon N de la forme

$$\ell(x_0, u_0) + \cdots + \ell(x_{N-1}, u_{N-1}) + \phi(x_N)\ .$$

On se ramène au problème 4.1 en posant :

$$G_{xy} := \sup_{u \in U,\ f(x,u)=y} \ell(x, u), \qquad b_x := \phi(x)\ .$$

Remarque 4.9 On peut s'intéresser plus généralement au cas où la dynamique et les gains dépendent du temps, ce qui revient à considérer une dynamique de la forme $x_{k+1} = f_k(x_k, u_k)$ et un Lagrangien $\ell_k(x_k, u_k)$ dans l'exemple qui précède. Remarquons que l'on peut se ramener formellement à la situation de l'exemple 4.8 en rajoutant le temps comme variable d'état supplémentaire, c'est-à-dire en considérant le nouvel espace d'état $\bar{X} = X \times \{0, \ldots, N\}$, le vecteur d'état $\bar{x}_k := (x_k, k)$, avec la dynamique $\bar{f}((x, k), u) := (f_k(x, u),\ k+1)$ et le Lagrangien $\bar{\ell}(\bar{x}, u) := \ell_k(x, u)$.

4.2 Principe de programmation dynamique pour le problème en horizon fini

Examinons tout d'abord le problème le plus simple, en horizon N. Notons que si X est fini, le nombre de trajectoires possibles est $|X|^N$. Il est donc très coûteux d'énumérer les trajectoires. La programmation dynamique parvient à éviter cette énumération, en "factorisant" le calcul. Elle repose en effet sur la manipulation algébrique suivante, qui revient à distribuer la somme par rapport au sup,

$$\begin{aligned} &\sup_{x_1,\ldots,x_n \in X} G_{x_0 x_1} + \cdots + G_{x_{N-1} x_N} + b_{x_N} \\ &= \sup_{x_1 \in X} G_{x_0 x_1} + \underbrace{\sup_{x_2,\ldots,x_n \in X} G_{x_1 x_2} + \cdots + G_{x_{N-1} x_N} + b_{x_N}}_{\text{valeur du même problème partant de } x_1, \text{ en horizon } N-1} \end{aligned} \tag{4.3}$$

Cette observation suggère, pour résoudre le Problème en horizon fini 4.1, de faire varier l'état initial et l'horizon de planification, ce qui conduit à la définition suivante.

Définition 4.10 (Fonction valeur du Problème en horizon fini) La *fonction valeur* du Problème 4.1 en horizon N est la fonction $v^N : x_0 \mapsto v^N_{x_0}$ qui associe à l'état initial x_0 la valeur (4.1) du problème de maximisation.

Théorème 4.11 (Équation de la programmation dynamique) *La fonction valeur du Problème 4.1 peut se calcule par récurrence comme suit :*

$$v^0_x = b_x,\ \forall x \in X \tag{4.4}$$

$$v^n_x = \sup_{y\in X}(G_{xy} + v^{n-1}_y),\ \forall 1 \leqslant n \leqslant N,\ \forall x \in X\ . \tag{4.5}$$

En outre, quand on est dans l'état x, et qu'il reste n coups à jouer, il est optimal d'aller en y, ssi le supremum est atteint en $y \in X$.

Démonstration. C'est une conséquence immédiate du calcul (4.3). ■ L'équation (4.5) est appelée équation de programmation dynamique. On peut s'en souvenir à l'aide du *principe d'optimalité de Bellman*, qui affirme qu'une sous-trajectoire d'une trajectoire optimale est encore optimale.

Remarque 4.12 Pour calculer (4.5) en toute généralité, on peut convenir de donner à $G_{xy} + v^{n-1}_y$ la valeur $-\infty$ lorsque $G_{xy} = -\infty$ et $v^{n-1}_y = +\infty$, ce qui reflète la convention que $G_{xy} = -\infty$ quand la transition de x vers y est interdite.

Introduisons l'opérateur de Bellman :

$$\mathcal{B} :\ \bar{\mathbb{R}}^X \to \bar{\mathbb{R}}^X,\ (\mathcal{B}(w))_x := \sup_{y\in X}(G_{xy} + w_y)\ , \tag{4.6}$$

où $\bar{\mathbb{R}}^X$ désigne l'ensemble des fonctions de X dans $\bar{\mathbb{R}}$. On peut alors écrire de manière agréable l'équation de programmation dynamique (4.5)

$$v^n = \mathcal{B}(v^{n-1})\ .$$

Comme $G_{xy} = -\infty$ représente l'absence de transition, il faut convenir dans (4.6) que $(-\infty) + (+\infty) = -\infty$. Ce sont évidemment les cas non-dégénérés où l'on applique $\mathcal{B}$ à des fonctions à valeurs finies qui nous intéresserons le plus en pratique. Notons en particulier que $\mathcal{B}$ préserve l'ensemble des fonctions bornées (donc à valeurs finies) sur X dès que la fonction $x \mapsto \sup_{y\in X} G_{xy}$, i.e., la fonction $\mathcal{B}(0)$, est elle même bornée.

Exemple 4.13 Illustrons la méthode de programmation dynamique en calculant la décision optimale du chauffeur de taxi, sur un horizon de deux courses ($N = 2$), partant du quartier H. Il vient :

$$v^0 = b = \begin{pmatrix} 0 \\ 2 \\ -\infty \end{pmatrix},\ \ v^1 = \mathcal{B}(v^0) = \begin{pmatrix} \max(3+0, \underline{10+2}) \\ \max(\underline{12+0}, 7+-\infty) \\ \max(-5+0, \underline{-3+2}) \end{pmatrix} = \begin{pmatrix} 12 \\ 12 \\ -1 \end{pmatrix}, \tag{4.7}$$

où l'on a souligné les termes réalisant le maximum. On a enfin

$$v^2_H \quad = \quad \max(3+12, \underline{10+12}) = 22\ .$$

On en déduit que le gain optimal, partant de H, en horizon 2, est de 22. Le maximum dans l'expression de v^2_H est atteint à droite, le terme souligné correspondant à un mouvement vers A. Il faut donc d'abord aller en A. Pour trouver le coup suivant, on observe que le terme donnant le maximum dans l'expression de v^1_A correspond à un déplacement vers H. On a ainsi trouvé la trajectoire optimale : $H \to A \to H$.

Remarque 4.14 Afin d'évaluer la complexité de la méthode, supposons X fini, et notons m le nombre de transitions possibles, i.e., le nombre de couples (x, y) tels que $G_{x,y} \neq -\infty$. On a évidemment $m \leqslant |X|^2$. Nous supposerons, sans perte de généralité, que chaque état est origine d'au moins une transition, de sorte que $m \geqslant |X|$. Étant donné $w \in \bar{\mathbb{R}}^X$, calculer $\mathcal{B}(w)$ prend un temps de calcul $O(m)$, le calcul (4.6) étant effectué de manière creuse, ce qui signifie que l'on n'inspecte seulement les transitions (x, y) autorisées (telles que $G_{x,y} \neq -\infty$). On en déduit que la résolution du Problème 4.1 par la méthode de la programmation dynamique peut se faire en temps $O(Nm)$. L'espace mémoire requis est $O(|X|)$ si l'on veut seulement déterminer la fonction valeur v^N ainsi que le premier coup à jouer dans chaque état. Si l'on veut construire toute la trajectoire optimale sans dupliquer les calculs, il faut stocker en mémoire les fonctions valeur successives $v^0, \ldots, v^N$, ce qui demande un espace $O(N|X|)$.

4.3 Quelques applications du problème en horizon fini

4.3.1 Gestion de stock

Un stock suit une dynamique de la forme $x_{n+1} = x_n - d_n + u_n$, où d_n désigne la demande à l'instant n, supposée connue pour tout $0 \leqslant n \leqslant N$, $x_n \in \mathbb{Z}$ désigne le niveau de stock à l'instant n, et $u_n \in \mathbb{N}$ le réassortiment. Un stock négatif modélise la possibilité d'emprunter. On note $c(u)$ le coût d'un réassortiment u. On a typiquement : $c(u) = a + bu$, pour $u > 0$, où $a > 0$ représente un coût fixe et b représente un coût proportionnel, et $c(0) = 0$. On se donne également un coût de stockage $s(x)$ dans l'état x typiquement, $s(x) = \max(p^+x, -p^-x)$, avec $p^- > p^+ > 0$, où p^- est un coût unitaire de défaillance, et p^+ le coût de stockage d'une unité excédentaire. On peut prendre en particulier $p^- = +\infty$, soit $s(x) = p^+x$ si $x \geqslant 0$, et $s(x) = +\infty$ sinon, ce qui modélise l'interdiction d'emprunter. Le problème de gestion de stock en horizon N revient à minimiser la somme des coûts de stockage et de réassortiment

$$\sum_{k=0}^{N} c(u_k) + s(x_k)$$

sous les contraintes $x_{k+1} = x_k - d_k + u_k$, $u_k \geqslant 0$, $k = 0, \ldots, N-1$, x_0 étant fixé. Il s'agit donc d'une version non-stationnaire (la dynamique dépend de k) du problème en horizon fini. Ce problème rentre dans le cadre de la remarque 4.9, mais il est plus rapide de ré-établir directement une équation de programmation dynamique. Notons en effet w_x^n l'infimum de $\sum_{k=n}^{N} s(x_k) + c(u_k)$, sous les contraintes $x_{k+1} = x_k - d_k + u_k$, $u_k \geqslant 0$, $k = n, \ldots, N-1$, $x_k = x$. On a

$$w_x^N = 0, \ \ \forall x \in \mathbb{Z}, \ \ w_x^k = \inf_{u \in \mathbb{N}} (s(x) + c(u) + w_{x-d_k+u}^{k+1}) \ , \qquad \forall x \in \mathbb{Z} \tag{4.8}$$

Ceci permet de déterminer de proche en proche $w^N, w^{N-1}, \ldots, w^n$. Noter l'inversion du temps par rapport à (4.5) : dans (4.5), n représente le temps restant à jouer, alors que dans (4.8), n représente le temps physique, qui s'écoule en sens inverse. L'équation (4.8) fournit un algorithme "théorique", qui demande de déterminer les valeurs w_x^k pour tout $x \in \mathbb{Z}$. Pour obtenir un algorithme fini, il faudrait remplacer l'espace d'état $\mathbb{Z}$ par un sous ensemble fini $X = [x^-, x^+] \subset \mathbb{Z}$, en exigeant que le réassortiment garantisse que le stock reste dans X, ce qui est d'ailleurs plus réaliste. Cela revient à remplacer (4.8) par :

$$w_x^k = \inf_{u \in \mathbb{N}, \ x-d_k+u \in X} (s(x) + c(u) + w_{x-d_k+u}^{k+1}), \ \forall x \in X \ .$$

On peut alors effectivement calculer par récurrence les fonctions valeur $(w_x^N)_{x \in X}, \ldots, (w_x^0)_{x \in X}$.

Exercice 4.15 Retrouver l'équation de programmation dynamique (4.8) en appliquant la remarque 4.9.

4.3.2 Problème du sac à dos

La programmation dynamique permet de résoudre des problèmes combinatoires avec un petit nombre de contraintes, dont le prototype est le problème du sac à dos :

$$\underset{z_i\in\{0,1\},\ \sum_{i=1}^N p_i z_i \leqslant M}{\text{Max}} \sum_{i=1}^N u_i z_i \ ,$$

où $u_i > 0$ représente l'utilité de l'objet i, $p_i > 0$ son poids, M le poids maximum admissible du sac à dos, N le nombre d'objets considérés, avec $z_i = 1$ si l'objet i est mis dans le sac à dos, et $z_i = 0$ sinon. On peut voir se problème comme un problème dynamique, consistant à examiner successivement les objets $1, \ldots, N$, et à les accepter ou à les rejeter. Ceci amène à définir la fonction valeur v_q^n, pour $n \in \{1, \ldots, N\}$ et $q \in \{0, \ldots, M\}$,

$$v_q^n = \text{Max} \sum_{i=n}^N u_i z_i, \quad \sum_{i=n}^N p_i z_i \leqslant q, z_i \in \{0,1\} \ .$$

Notons que q représente la capacité résiduelle du sac, une fois que les décisions $z_1, \ldots, z_{n-1} \in \{0,1\}$ ont été prises. L'équation de la programmation dynamique s'écrit comme suit :

$$v_q^n = \begin{cases} \max(v_q^{n+1},\ u_n + v_{q-p_n}^{n+1}) & \text{si } q \geqslant p_n \\ v_q^{n+1} \text{ si } q < p_n \end{cases} \ ,$$

avec $v_q^{N+1} = 0$, pour $q \in \{1, \ldots, M\}$. En calculant successivement $v^{N+1}, \ldots, v^1$, on obtient en particulier v_M^1, qui est la valeur du problème de sac à dos initial. L'algorithme prend un temps $O(NM)$. Il est seulement pseudo-polynomial, car l'entier M occupe en mémoire un espace de l'ordre de $\log_2 M$ bits.

Exercice 4.16 Résoudre par programmation dynamique le problème de sac à dos

$$\underset{x\in\{0,1\}^3}{\text{Max}}\ 4x_1 + 3x_2 + 2x_3; \quad 5x_1 + 4x_2 + 3x_3 \leqslant 10. \tag{4.9}$$

Exercice 4.17 (Sac à dos multidimensionnel) Appliquer le principe de programmation dynamique à la résolution du problème

$$\underset{x\in\{0,1\}^n}{\text{Max}} \sum_{i=1}^n c_i x_i; \sum_{i=1}^n a_{ij} x_i \leqslant b_j,\ j = 1, 2. \tag{4.10}$$

On suppose les b_j et a_{ij} entiers strictement positifs. Évaluer le nombre d'opérations.

Généraliser au cas de m contraintes : $\sum_{j=1}^n a_{ji} x_i \leqslant b_j,\ j = 1, \ldots, m$. Qu'elle est la limitation de la méthode ?

4.4 Problème arrêté ou problème du plus court chemin

Considérons maintenant le problème arrêté 4.2. Notons v_{x_0} la valeur du problème de maximisation (4.2), pour un état initial x_0. On a évidemment

$$v = \sup_{N\geqslant 0} v^N$$

où v^N est la valeur du problème en horizon N (Définition 4.10).

La fonction valeur v du Problème 4.2 vérifie l'équation de programmation dynamique

$$v = \max(b, \mathcal{B}(v)) \tag{4.11}$$

où $\mathcal{B}$ est l'opérateur de Bellman du problème en horizon fini, défini en (4.6). Pour s'en convaincre, il suffit d'expliciter (4.11) :

$$v_x = \max(b_x, \sup_{y \in X} G_{xy} + v_y) \ .$$

Le premier terme du max traduit la possibilité de s'arrêter, le second terme du max traduit la possibilité de continuer. Contrairement au cas du problème en horizon fini, l'équation de programmation dynamique est une équation de point fixe, qui peut a priori avoir plusieurs solutions. Rappelons qu'on appelle solution minimale d'une équation une solution qui minore toutes les autres.

Théorème 4.18 *La fonction valeur v du problème arrêté 4.2 est la solution minimale de l'équation de programmation dynamique* (4.11).

Démonstration. On a déjà montré que v est solution de (4.11). Soit v' une solution quelconque de (4.11). Montrons que $v' \geqslant v$. Soit $(x_0, \ldots, x_N)$ un chemin quelconque issu de x_0. Comme v' vérifie (4.11), on peut écrire $v'_{x_r} \geqslant G_{x_r, x_{r+1}} + v'_{x_{r+1}}$, pour tout $0 \leqslant r \leqslant N-1$, et $v'_{x_N} \geqslant b_{x_N}$. En enchaînant ces inégalités,

$$v'_{x_0} \geqslant G_{x_0,x_1} + \cdots + G_{x_{N-1},x_N} + b_{x_N}$$

et en prenant le supremum sur tous les chemins $(x_0, \ldots, x_N)$ issus de x_0, $v'_{x_0} \geqslant v_{x_0}$. Ainsi $v' \geqslant v$, ce qui montre que v est solution minimale de (4.11). ■

L'exercice suivant donne une condition garantissant la finitude et l'unicité de la solution de (4.11).

Exercice 4.19 On se restreint au cas où X est fini, et l'on note $\mathcal{A} = \{(x,y) \in X \times X \mid G_{x,y} \neq -\infty\}$. Nous supposerons que pour tout $x \in X$, il existe un chemin dans le graphe $(X, \mathcal{A})$ conduisant à un état y tel que $b_y \neq -\infty$. Nous supposerons en outre que tous les circuits du graphe $(X, \mathcal{A})$ ont un poids strictement négatif. Montrer alors que l'équation (4.11) possède une solution unique finie, égale à v. Indication. Considérons $v' \in \mathbb{R}^X$ une solution finie de l'équation de Bellman (4.11), introduisons l'ensemble

$$C = \{x \in X \mid b_x \leqslant \max_y (G_{x,y} + v'_y)\} \ ,$$

et choisissons pour chaque $x \in C$ un nœud $\pi(x)$ tel que

$$v'_x = G_{x,\pi(x)} + v'_{\pi(x)} \ .$$

On définit ainsi une application $\pi : C \to X$. Montrer que quel que soit $x \in C$, il existe un entier k tel que le k-ème itéré $\pi^k(x)$, n'appartienne pas à C. Conclure que $v' \geqslant v$.

La méthode la plus simple pour résoudre l'équation (4.11) est d'effectuer une itération de point fixe. C'est l'algorithme dit d'*itération sur les valeurs*.

Exercice 4.20 (Itération sur les valeurs) Définissons l'opérateur

$$f(w) = \sup(b, \mathcal{B}(w)) \ .$$

Montrer que

$$f^k(w) = \sup(b, \mathcal{B}(b), \ldots, \mathcal{B}^{k-1}b, \mathcal{B}^k w) \ .$$

En déduire que si w^k désigne la suite définie par $w^0 = b$, $w^{k+1} = f(w^k)$, alors w^k est une suite croissante qui converge vers la fonction valeur du problème 4.2. Montrer en outre que si X est fini, et que si pour tout circuit $x_1, x_2, \ldots, x_{N+1} = x_1$, on a $G_{x_1x_2} + \cdots + G_{x_Nx_1} \leqslant 0$, alors $w^k = w^{k+1} = \cdots = v$ pour un certain $k \leqslant N-1$.

En pratique, on programme rarement tel quel cet algorithme d'itération sur les valeurs, mais plutôt la variante suivante de type Gauss-Seidel, appelée algorithme de Ford-Bellman [Bel58], laquelle met à jour au plus tôt toutes les coordonnées dans l'itération sur les valeurs. Nous formulons cette variante dans le cas du problème de plus court chemin (minimisation d'un coût, à destination fixée).

Algorithme 4.21 (de Ford et Bellman) Entrée : un graphe $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, un coût $c : \mathcal{A} \to \mathbb{R}$, une destination $d \in \mathcal{N}$. Variables : $v \in (\mathbb{R} \cup \{+\infty\})^{\mathcal{N}}$, β un Booléen, r un compteur de boucle, $i, k \in \mathcal{N}$.

Initialisation : $r \leftarrow 0$, $\beta \leftarrow$ vrai ; $v_d \leftarrow 0$ et pour tout $i \in \mathcal{N} \setminus \{d\}$, $v_i \leftarrow +\infty$.

Tant que $r < |\mathcal{N}|$ et β faire :

$\beta \leftarrow$ faux, $r \leftarrow r + 1$;

pour tout $i \in \mathcal{N}$ et pour tout $(i, k) \in \mathcal{A}$, si $c_{i,k} + v_k < v_i$, faire $v_i \leftarrow c_{i,k} + v_k$ et $\beta \leftarrow$ vrai.

Si $r < |\mathcal{N}|$, pour tout $i \in \mathcal{N}$, v_i fournit le coût minimum d'un chemin partant de i terminant en d. Si $r = |\mathcal{N}|$, il existe un circuit de coût strictement négatif.

Exercice 4.22 Comment peut-on compléter l'algorithme de Ford-Bellman pour construire un circuit de coût négatif ?

Remarque 4.23 Il existe des variantes de l'algorithme de Ford-Bellman, qui différent par l'ordre dans lequel on parcourt les nœuds i et les arcs (i, k) dans la boucle de l'algorithme 4.21 : on trouvera le terme générique de "label correcting algorithms" dans la littérature. Les algorithmes de cette famille, et en particulier l'algorithme de Ford-Bellman, sont parmi les plus rapides pour calculer les chemins de coût minimum, dans le cas d'un graphe orienté qui a des circuits (de coût positif on nul) et dont les coûts peuvent être négatifs. Dans deux cas spéciaux, on sait faire cependant beaucoup mieux que Ford-Bellman. Lorsque le graphe $\mathcal{G}$ n'a pas de circuit, on effectue un tri topologique, c'est-à-dire que l'on munit les sommets d'un ordre total $\leqslant$ tel que s'il y a un chemin de i à j, alors $i \leqslant j$. Souvent, les nœuds sont déjà naturellement ordonnés (par exemple par temps croissant). Dans le cas contraire, on sait trouver un tel ordre en temps linéaire, voir par exemple [AMO93] pour plus de détails. Une fois les nœuds ordonnés, il suffit d'initialiser $v_d = 0$, et $v_i = +\infty$, comme dans l'algorithme de Ford-Bellman ci-dessus, et d'effectuer une et une seule fois la substitution $v_i \leftarrow \min_{k \in \mathcal{N},\, (i,k) \in \mathcal{A}} c_{i,k} + v_k$ pour chaque i, en parcourant les i par ordre décroissant, pour obtenir la fonction valeur, ce qui prend un temps linéaire. Un autre cas particulier remarquable est celui où les coûts sont positifs ou nuls : dans ce cas, on peut employer un algorithme de type glouton, appelé algorithme de Moore-Dijsktra, voir par exemple [AMO93]. •

Exercice 4.24 (Chemin de coût minimum avec contrainte de temps). Considérons un graphe orienté, dont chaque arc (i, j) est muni d'un coût c_{ij} et d'un temps t_{ij}. Il s'agit de trouver un chemin $i_0, \ldots, i_k$ d'une origine donnée o à une destination donnée d (le nombre de routes empruntées, k, étant quelconque), minimisant le coût total :

$$c_{i_0 i_1} + \cdots + c_{i_{k-1} i_k} \ , \tag{4.12}$$

sous la contrainte

$$t_{i_0 i_1} + \cdots + t_{i_{k-1} i_k} \leqslant T \ , \tag{4.13}$$

où T est un temps maximal fixé. Formuler un algorithme de programmation dynamique pour résoudre ce problème (on supposera que les temps sont des entiers strictement positifs).

Modéliser la généralisation suivante : trouver le parcours en avion le moins cher d'une ville o à une ville d, de temps total au plus T, avec un probabilité de survie d'au moins P sur la totalité du vol.

4.5 Problème actualisé

Tout comme le cas arrêté, le cas actualisé conduit à un problème de point fixe.

Théorème 4.25 (Programmation dynamique avec actualisation) *La fonction valeur $v : x_0 \mapsto v_{x_0}$ du Problème actualisé 4.5 est solution de l'équation de programmation dynamique*

$$v = \mathcal{B}(\alpha v) \ .$$

Démonstration. Il suffit de remplacer la somme finie $G_{x_0x_1} + \cdots + G_{x_{N-1}x_N} + b_{x_N}$ par la série $G_{x_0x_1} + \alpha G_{x_1x_2} + \alpha^2 G_{x_2x_3} \cdots$ dans (4.3). ■

L'opérateur $\mathcal{B}$ vérifie les deux propriétés suivantes, dites de croissance et d'homogénéité additive

$$u \leqslant w \implies \mathcal{B}(u) \leqslant \mathcal{B}(w), \qquad \mathcal{B}(\lambda + u) = \lambda + \mathcal{B}(u) \tag{4.14}$$

où u, w désignent des fonctions quelconques de X dans $\mathbb{R}$, et où pour tout réel λ, $\lambda + u$ désigne la somme de la fonction u et de la fonction constante λ. Ces deux propriétés sont très utiles. Elles entraînent que $\mathcal{B}$ est une contraction au sens large pour la norme sup. En effet, en appliquant (4.14) à l'inégalité

$$-\|u - w\|_\infty + w \leqslant u \leqslant \|u - w\|_\infty + w$$

il vient

$$-\|u - w\|_\infty + \mathcal{B}(w) \leqslant \mathcal{B}(u) \leqslant \|u - w\|_\infty + \mathcal{B}(w)$$

et donc

$$\|\mathcal{B}(u) - \mathcal{B}(w)\|_\infty \leqslant \|u - w\|_\infty \ .$$

Pour que ce calcul ait du sens, il faut bien sûr que les fonctions $\mathcal{B}(u)$ et $\mathcal{B}(v)$ prennent des valeurs finies, ce qui est le cas en particulier si la fonction $x \mapsto \sup_{y \in X} G_{xy}$ est bornée et si les fonctions u et w sont bornées.

Le résultat suivant, de nature algorithmique, résulte aussitôt du théorème de point fixe de Banach et du fait que $w \mapsto \mathcal{B}(\alpha w)$ est contractante de taux α pour la norme sup.

Proposition 4.26 (Itération sur les valeurs, cas actualisé) *Supposons bornée la fonction $x \mapsto \sup_{y \in X} G_{xy}$. Alors, la fonction valeur du problème actualisé est l'unique fonction bornée solution du problème de point fixe $v = \mathcal{B}(\alpha v)$. En outre, pour toute fonction bornée v^0 de X dans $\mathbb{R}$, la suite de fonctions définie par*

$$v^{k+1} = \mathcal{B}(\alpha v^k)$$

converge vers v avec un taux α. Plus précisément :

$$\|v^k - v\|_\infty \leqslant \alpha^k \|v^0 - v\|_\infty \ .$$

4.6 Problème ergodique

Dans la suite, il sera commode de voir l'ensemble X des états comme l'ensemble des nœuds d'un graphe orienté dont les arcs sont les transitions autorisées, $\mathcal{A} := \{(y, z) \in X^2 \mid G_{yz} \neq -\infty\}$. Nous dirons que la fonction de gain G est *irréductible* si ce graphe est fortement connexe, ce qui signifie que l'on peut passer d'un état choisi de manière arbitraire à un autre état choisi de manière arbitraire par une suite finie de transitions autorisées. Une propriété du critère ergodique est de ne tenir compte que du comportement "ultime" de la suite d'états. La proposition suivante est alors très intuitive.

Proposition 4.27 *Si la fonction de gain est irréductible, alors la valeur du problème ergodique 4.6 est indépendante du choix de l'état initial.*

Afin de démontrer cela, notons $\mu(x_0)$ la valeur du problème ergodique 4.6, vue comme une fonction de l'état initial x_0. Le lemme suivant, élémentaire, montre que la valeur ne peut que décroître si l'on remplace l'état initial par un état auquel on peut accéder par une transition autorisée.

Lemme 4.28 *Si $G_{xy} \neq -\infty$, alors*

$$\mu(x) \geqslant \mu(y) \ .$$

Démonstration. Si $G_{x_0x_1}$ est fini, la limite supérieure

$$\limsup_{N\to\infty} \frac{G_{x_0x_1} + \cdots + G_{x_{N-1}x_N}}{N}$$

est inchangée si on remplace la suite infinie $x_0, x_1, \ldots$ par la suite obtenue en effaçant le premier terme x_0 et si l'on remplace N par $N-1$ au dénominateur. On en déduit que

$$\mu(x_0) \geqslant \operatorname*{Max}_{x_2,x_3\ldots} \limsup_{N\to\infty} \frac{G_{x_1x_2} + \cdots + G_{x_{N-1}x_N}}{N-1} = \mu(x_1) \ .$$

■

La proposition 4.27 en résulte. En effet, si les gains $G_{x_0x_1}, \ldots, G_{x_{k-1}x_k}$ sont tous finis, le lemme précédent montre que $\mu(x_0) \geqslant \mu(x_1) \geqslant \cdots \geqslant \mu(x_k)$. Si la fonction de gain est irréductible, il s'ensuit que $\mu(x) \geqslant \mu(y)$ quels que soient x et y. En échangeant les rôles de x et de y, on conclut alors que $\mu(x) = \mu(y)$. ■

Ceci nous conduit à formuler une variante du problème ergodique dans laquelle l'état initial x_0 est laissé libre.

Problème 4.29 (Critère ergodique, état initial libre). *Maximisons le gain moyen par unité de temps :*

$$\operatorname*{Max}_{x_0,x_1,x_2,\ldots\in X} \limsup_{N\to\infty} \frac{G_{x_0x_1} + \cdots + G_{x_{N-1}x_N}}{N} \ .$$

En vertu de la proposition 4.27, cette variante est équivalente au problème initial quand la fonction de gain est irréductible.

Deux approches peuvent être employées pour résoudre le problème 4.29, l'une repose sur une variante de l'équation de programmation dynamique, l'autre sur "l'espace des fréquences". Nous présentons ici la seconde, la première pouvant s'en déduire par un exercice de dualité que nous énonçons ensuite.

Proposition 4.30 *Si l'espace d'états X est fini, alors l'optimum du problème ergodique 4.29 est atteint par une suite $x_0, x_1, \ldots$ périodique.*

Démonstration. Il sera commode de noter

$$G(x_0, \ldots, x_N) = G_{x_0x_1} + \cdots + G_{x_{N-1}x_N}$$

le gain de la suite $x_0, \ldots, x_N$. Appelons circuit de longueur N une suite d'états de la forme $c = (y_0, \ldots, y_{N-1}, y_0)$ et définissons le poids moyen de c :

$$\lambda(c) = \frac{G(y_0, \ldots, y_{N-1}, y_0)}{N} \ .$$

Considérons le problème de maximisation sur l'ensemble des circuits :

$$\lambda := \operatorname*{Max}_c \lambda(c) \ . \tag{4.15}$$

Si $c' = (y'_0, \ldots, y'_{M-1}, y'_0)$, avec $y_0 = y'_0$, notons cc' le circuit obtenu par concaténation, soit $(y_0, \ldots, y_{N-1}, y_0, y'_1, \ldots, y'_{M-1}, y'_0)$. Comme $G(cc') = G(c) + G(c')$, on voit facilement que $\lambda(cc') = (G(c) + G(c'))/(N+M) \leqslant \max(G(c)/N, G(c')/M) \leqslant \max(\lambda(c), \lambda(c'))$. Il en résulte que le supremum du Problème (4.15) ne change pas si l'on se restreint aux circuits c élémentaires, c'est-à-dire aux circuits $c = (x_0, \ldots, x_{N-1}, x_0)$ tels que les états $x_0, \ldots, x_{N-1}$ soient tous distincts. Notons que la longueur N d'un circuit élémentaire vérifie nécessairement $N \leqslant |X|$. Il n'y a donc qu'un nombre fini de tels circuits. On en déduit que l'optimum du Problème (4.15) est atteint par un circuit élémentaire c^*.

Supposons tout d'abord λ fini, ce qui revient à dire qu'il existe au moins un circuit constitué de transitions autorisées. Soustrayons λ à chaque gain G_{xy}. Le problème transformé est tel que $\lambda = 0$, de sorte que $\lambda(c) \leqslant 0$

et donc $G(c) \leqslant 0$ pour tout circuit c, avec bien sûr $\lambda(c^*) = G(c^*) = 0$. De manière analogue au cas des circuits, qualifions d'élémentaire une suite d'états $x_0, \ldots, x_N$ telle que $x_i \neq x_j$ pour $0 \leqslant i \leqslant j \leqslant N$. Toute suite finie d'états $x_0, x_1, \ldots, x_N$ peut être ramenée à une suite élémentaire $x_{i_0}, x_{i_1}, \ldots, x_{i_M}$ en effaçant successivement des circuits, et l'on a alors $G(x_0, \ldots, x_N) \leqslant G(x_{i_0}, \ldots, x_{i_M})$. Comme il n'y a qu'un nombre fini de suites élémentaires (en fait, $M \leqslant |X| - 1$), on en déduit que $G(x_0, \ldots, x_N)$ est supérieurement borné indépendamment de la longueur N de la suite. La valeur du Problème 4.29 est donc négative ou nulle. Elle est atteinte par la suite infinie périodique, de valeur nulle, obtenue en répétant infiniment le circuit c^*.

Enfin, si $\lambda = -\infty$, tout circuit est de gain $-\infty$, et comme toute suite de $|X|$ états de longueur N contient un circuit, la limsup apparaissant dans l'énoncé du Problème 4.29 vaut $-\infty$ pour toute suite. L'énoncé de la proposition est donc trivialement réalisé. ■

Si $x_0, x_1, \ldots$ est une suite périodique, on peut définir la fréquence de visite de l'arc (y, z) :

$$f_{yz} = \lim_{N\to\infty} N^{-1} |\{0 \leqslant k \leqslant N-1 \mid x_k = y,\ x_{k+1} = z\}|$$

de sorte que

$$\lim_{N\to\infty} \frac{G_{x_0 x_1} + \cdots + G_{x_{N-1} x_N}}{N} = \sum_{(y,z)\in\mathcal{A}} G_{yz} f_{yz} \tag{4.16}$$

On vérifie facilement que le vecteur f ainsi obtenu est une mesure de probabilité sur l'espace des arcs

$$f_{yz} \geqslant 0,\ \forall (y,z) \in \mathcal{A}\ , \tag{4.17}$$

$$\sum_{(y,z)\in\mathcal{A}} f_{yz} = 1 \tag{4.18}$$

qui vérifie en outre la relation d'équilibre

$$\forall y \in X,\ \sum_{z\in X,\ (z,y)\in\mathcal{A}} f_{zy} = \sum_{z\in X, (y,z)\in\mathcal{A}} f_{yz} \tag{4.19}$$

Désignons par P le polyèdre des vecteurs de fréquence f, c'est-à-dire l'ensemble des solutions de (4.17), (4.18) et (4.19).

Lemme 4.31 *Le polyèdre P des vecteurs de fréquence a pour points extrêmes les mesures de probabilités uniformes supportées par des circuits élémentaires.*

Démonstration. Un vecteur de fréquence f est une circulation dans le graphe $(X, \mathcal{A})$. D'après la proposition 3.11, f peut s'écrire comme combinaison linéaire positive

$$f = \lambda_1 g^1 + \cdots + \lambda_k g^k$$

avec $\lambda_1, \ldots, \lambda_k > 0$, où $g^1, \ldots, g^k$ sont des flots unitaires associés à des circuits élémentaires, dont nous noterons $n_1, \ldots, n_k$ les longueurs respectives. Les vecteurs $f^i := g^i/n_i, i = 1, \ldots, k$ sont donc des mesures de probabilité uniformes supportées par des circuits élémentaires, et l'on a $f = \lambda_1 n_1 f^1 + \cdots + \lambda_k n_k f^k$. Comme la masse de f et de chaque f^i est 1, il vient $1 = \lambda_1 n_1 + \cdots + \lambda_k n_k$, ce qui montre que f est barycentre de $f^1, \ldots, f^k$. Ceci montre que l'ensemble des points extrêmes de P est contenu dans l'ensemble des mesures de probabilité uniformes supportées par des circuits élémentaires. Il reste à vérifier toute mesure f de cette sorte est bien un point extrême de P. Dans le cas contraire, ce qui précède nous permettrait d'exprimer f comme barycentre de mesures supportées par des circuits élémentaires, et ces circuits seraient nécessairement contenus dans le circuit supportant f, contredisant son caractère élémentaire. ■

Remarque 4.32 On vérifie facilement que le polyèdre P est vide si et seulement le graphe $(X, \mathcal{A})$ est sans circuits.

Théorème 4.33 (Problème ergodique, approche via les fréquences). *Supposons l'espace d'état X fini. Alors, la valeur du problème ergodique 4.29 coïncide avec celle du programme linéaire :*

$$\text{Max} \sum_{(y,z)\in\mathcal{A}} G_{yz} f_{yz}, \quad f \in P \ . \tag{4.20}$$

Démonstration. Traitons d'emblée le cas trivial où le polyèdre P est vide : dans ce cas, la valeur du programme (4.20) est $-\infty$, le graphe $(X, \mathcal{A})$ est sans circuits, et la valeur du problème 4.29 vaut $-\infty$ car toute suite infinie d'états contient nécessairement un circuit.

Si P est non-vide, comme P est un polytope, la valeur du programme linéaire (4.20) est atteinte en un point extrême de P, qui en vertu du lemme 4.31, est une mesure de probabilité uniforme supportée par un circuit élémentaire. D'après l'identité (4.16), on voit que $\sum_{(y,z)\in\mathcal{A}} G_{yz} f_{yz}$ coïncide avec la moyenne de Cesaro des gains de la suite périodique d'états obtenue en répétant ce circuit élémentaire, ce qui conclut la preuve. ■

Exercice 4.34 (Approche duale) Montrer que le problème dual de (4.20) peut s'écrire :

$$\text{Min}\, \lambda, \ \lambda \in \mathbb{R}, \ w \in \mathbb{R}^X, \ G_{xy} + w_y \leqslant \lambda + w_x, \ \forall (x,y) \in \mathcal{A} \ .$$

Montrer qu'un circuit c est de gain moyen $\lambda(c)$ maximal si et seulement si il est composé d'arcs (x, y) tels que $G_{xy} + w_y = \lambda + w_x$ pour toute solution optimale (λ, w) (indication : utiliser la propriété des écarts complémentaires).

Exercice 4.35 Rotation des cultures). Un agriculteur souhaite optimiser la culture d'un champ. On se donne un ensemble fini $\mathcal{C}$ de cultures possibles sur le champ, par exemple $\mathcal{C} = \{B, A, J\}$ respectivement pour "blé", "avoine", et "jachère". On suppose qu'il n'y a qu'une récolte par an. On admet que pour la suite de cultures $c_0, c_1, \ldots, c_k \in \mathcal{C}$, le revenu de l'agriculteur noté $R(c_0, c_1, \ldots, c_k)$, est donné par :

$$R(c_0, c_1, \ldots, c_k) = \\ L(c_0) + K(c_0, c_1) + K(c_1, c_2) + \cdots + K(c_{k-1}, c_k) \ ,$$

où la fonction $K : \mathcal{C} \times \mathcal{C} \to \mathbb{R}$ et la fonction $L : \mathcal{C} \to \mathbb{R}$ sont connues de l'agriculteur.

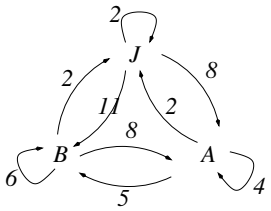
1. Écrire une équation de programmation dynamique permettant de calculer le revenu optimal de l'agriculteur en horizon k.

L'agriculteur souhaite maintenant maximiser son revenu à long terme, i.e. choisir une suite infinie de cultures $c_0, c_1, \ldots \in \mathcal{C}$ maximisant le revenu moyen par année, lequel s'écrit :

$$\lambda(c_0, c_1, \ldots) = \limsup_{k\to\infty} R(c_0, \ldots, c_k)/k \ .$$

On note λ^* le supremum des valeurs de $\lambda(c_0, c_1, \ldots)$.

2. Montrer que l'optimum est atteint par une rotation (suite obtenue en répétant à l'infini un circuit élémentaires de cultures). Expliciter la rotation optimale dans le cas $\mathcal{C} = \{B, A, J\}$ ("blé", "avoine", "jachère"), les revenus annuels étant indiqués sur le graphe suivant :



Par exemple, l'arc $B \stackrel{8}{\to} A$ signifie que $K(B,A) = 8$ (si l'on cultive de l'avoine -semé au printemps- sachant que l'année précédente on a cultivé du blé -semé en hiver-, le revenu de l'année courante est de 8). Calculer dans ce cas λ^* ainsi qu'une suite optimale périodique de cultures.

3. Formuler un programme linéaire permettant de calculer λ^* en utilisant l'approche des fréquences.

4. Expliciter le problème dual et l'interpréter.

5. Montrer que si $\lambda \in \mathbb{R}$ et $u \in \mathbb{R}^{\mathcal{C}}$ sont tels que

$$\max_{c' \in \mathcal{C}} K(c,c') + u(c') = \lambda + u(c), \quad \forall c \in \mathcal{C} \ ,$$

alors nécessairement $\lambda = \lambda^*$.

On suppose maintenant que le revenu de l'agriculteur dépend des cultures des deux années précédentes, soit :

$$R(c_0, c_1, \ldots, c_k) = L'(c_0, c_1) + K'(c_0, c_1, c_2) + K'(c_1, c_2, c_3) + \cdots + K'(c_{k-2}, c_{k-1}, c_k),$$

où les fonction $K' : \mathcal{C} \times \mathcal{C} \times \mathcal{C} \to \mathbb{R}$ et la fonction $L : \mathcal{C} \times \mathcal{C} \to \mathbb{R}$ sont connues de l'agriculteur.

6. Écrire un équation de programmation dynamique permettant de résoudre ce dernier problème. On explicitera le graphe quand $\mathcal{C} = \{B, A, J\}$.

Cet exercice est inspiré d'un article de Nicolas Bacaer [Bac03] qui démontre qu'un horizon de deux ans est nécessaire pour expliquer les divers assolements observés (assolement triennal à l'époque classique, assolement jachère-céréale dans l'antiquité, ...)

Exercice 4.36 On pourrait considérer un problème ergodique avec contrainte, ce qui reviendrait à se donner une seconde fonction R_{xy} et à maximiser le critère

$$\limsup_{N \to \infty} \frac{G_{x_0 x_1} + \cdots + G_{x_{N-1} x_N}}{N}$$

sur l'ensemble des suites telles que

$$\limsup_{N \to \infty} \frac{R_{x_0 x_1} + \cdots + R_{x_{N-1} x_N}}{N} \leqslant \beta$$

où β est un seuil donné (ceci peut modéliser la minimisation d'un coût d'exploitation sachant que certaines tâches de type maintenance ont une fréquence minimale admissible). Il est naturel de considérer le problème analogue à (4.20)

$$\text{Max} \sum_{(y,z) \in \mathcal{A}} G_{yz} f_{yz}, \quad f \in P, \sum_{(y,z) \in \mathcal{A}} R_{yz} f_{yz} \leqslant \beta \ .$$

Trouver l'erreur dans l'argument paradoxal suivant : "on s'est ramené à un PL, soluble en temps polynomial, or les problèmes de type plus court chemin avec contraintes, et donc le problème considéré, sont connus pour être NP-durs." Indication : si P est un polyèdre, et si H est un demi-espace, comparer l'ensemble des points extrêmes de P appartenant à H et l'ensemble des points extrêmes de $P \cap H$.

4.7 Notes

Une référence classique en programmation dynamique est l'ouvrage [Ber01], qui traite également du cas stochastique (processus de décision Markoviens). Le lien entre programmation dynamique déterministe et les algèbres de chemin apparaît dans [GM09]. La modélisation de problèmes de programmation dynamique par des fréquences (mesures d'occupations) est présentée dans [Bor91, HLL99], dans un cadre de contrôle stochastique. Les résultats de ce chapitre concernant le problème ergodique sont des versions simplifiées de résultats de théorie spectrale max-plus ou tropicale [BCOQ92, ABG13]. Les méthodes de programmation dynamique s'étendent au cas des jeux à somme nulle, voir [FV97, NS03].

Chapitre 5

Séparation, évaluation, relaxation

Dans la résolution d'un problème de Recherche Opérationnelle, une des premières tâches est de reconnaître si une méthode exacte efficace (polynomiale) s'applique : ainsi, nous avons mis l'accent dans les chapitres précédents sur les problèmes que l'on peut résoudre exactement à l'aide de la programmation linéaire continue, ou parfois de manière encore plus efficace, à l'aide d'algorithmes de flots. Que faire, cependant, pour trouver la solution exacte, dans des cas plus difficiles ? Nous présentons dans ce chapitre une technique générale en optimisation combinatoire, appelée "séparation et évaluation", ou en anglais, "branch and bound". Il s'agit là d'une méthode d'exploration arborescente, dans laquelle l'ingrédient essentiel est un *minorant* de la valeur de sous-problèmes (obtenus en fixant certaines variables à optimiser). On parvient ainsi à la solution optimale, ou à défaut, à une solution de qualité garantie (on saura borner l'écart en valeur à l'optimum). Les techniques de séparation et évaluation réduisent ainsi un problème d'optimisation à un problème de calcul de borne. Le problème de calcul de borne est, en général, aussi difficile que le problème d'optimisation initial. Cependant, pour certains problèmes de recherche opérationnelle présentant des propriétés de structure (comme le problème du voyageur de commerce), et en dépit du diagnostic théorique de NP-difficulté, des bornes utiles peuvent être obtenues par des méthodes dites de *relaxation*, qui consistent à approcher un problème d'optimisation combinatoire par un problème d'optimisation convexe, voire linéaire.

5.1 Séparation et évaluation (branch and bound)

Considérons le problème combinatoire très général

$$\min_{x \in X} f(x) \ , \tag{5.1}$$

avec X fini (mais gros), et $f : X \to \mathbb{R}$.

Dans la méthode de séparation et évaluation (en anglais, "branch and bound"), la *séparation* consiste à représenter l'ensemble X des points admissibles par les feuilles d'un arbre, que l'on va explorer. Les nœuds internes de l'arbre représentent des décisions partielles (correspondant à fixer certaines variables de décision, mais pas toutes), le nœud racine représente une situation initiale, dans laquelle on n'a encore rien décidé.

Par exemple, si $x = (x_1, \ldots, x_n) \in X := \{0,1\}^n$, une décision partielle pourra consister à fixer la valeur de x_1, ce qui donne deux sous-arbres, correspondant respectivement à $x_1 = 0$ et $x_1 = 1$. On fixe ensuite la valeur de x_2, etc. On obtient ainsi un arbre binaire, dont les feuilles représentent tous les x possibles dans $\{0,1\}^n$.

L'*évaluation* s'intéresse, pour chaque nœud interne s de l'arbre, au *coût conditionnel*

$$v(s) = \min_{s' \text{ est une feuille descendante de } s} f(s') \ . \tag{5.2}$$

(Le mot "descendant" a le sens généalogique, i.e. nous orientons l'arbre avec la racine en haut, et les feuilles en bas.) Le calcul de ce coût conditionnel en s est souvent aussi dur que le problème initial (5.1) (en particulier,

quand s est la racine, (5.2) coïncide avec (5.1)), c'est pourquoi nous allons simplifier le problème (5.2), en nous autorisant l'introduction d'un minorant $b(s)$ du coût conditionnel en s :

$$b(s) \leqslant v(s) \ , \tag{5.3}$$

que l'on devra définir pour tout nœud interne de l'arbre. Nous suivrons l'usage, qui appelle (improprement) *borne inférieure* ou tout simplement *borne* le minorant $b(s)$.

Pour reprendre l'exemple où $X = \{0,1\}^n$, un nœud interne s de l'arbre, à distance k de la racine, correspond à une suite $(y_1, \ldots, y_k) \in \{0,1\}^k$, et le coût conditionnel s'écrit

$$v(y_1, \ldots, y_k) = \min_{x_{k+1},\ldots,x_n \in \{0,1\}} f(y_1, \ldots, y_k, x_{k+1}, \ldots, x_n) \ .$$

5.1.1 Énoncé de l'algorithme de séparation et évaluation

L'algorithme de séparation et évaluation consiste à explorer les nœuds de l'arbre, en partant de la racine. En cours d'exploration, on mémorise m, le coût minimal des points admissibles déjà trouvés, ainsi que le point admissible correspondant. On initialise l'algorithme avec $m = +\infty$. Quand on visite pour la première fois un nœud interne s du graphe, on évalue la borne $b(s)$. Si $b(s) \geqslant m$, il ne sert à rien d'explorer les branches filles du nœud s, car le coût des feuilles qui s'y trouvent n'est pas meilleur que le coût du meilleur point admissible rencontré, et l'on remonte au nœud père de s afin de poursuivre l'exploration de l'arbre (on peut visualiser cela en disant que l'on coupe la branche de l'arbre partant du nœud s). Si au contraire $b(s) < m$, il est possible qu'une branche partant de s contienne un point admissible améliorant m : on poursuit dans ce cas l'exploration de l'arbre, en passant à un nœud fils de s non encore exploré. En termes d'algorithmique, le fait de visiter en priorité un nœud fils caractérise les algorithmes de parcours de graphe dits en *profondeur d'abord* (depth first search). Quand on parvient à une feuille de l'arbre, si elle représente un point admissible $x \in X$ de (5.1), il ne reste qu'à calculer la valeur $f(x)$: si $f(x) < m$, le meilleur point admissible trouvé est x, on pose donc $m = f(x)$, et l'on mémorise x à la place de l'ancien point admissible trouvé. On poursuit alors l'exploration de l'arbre en remontant au nœud père de la feuille courante.

L'algorithme visite au plus une fois chaque feuille. Le pire des cas est celui où la borne b ne permet jamais de couper de branche : l'algorithme revient dans ce cas à énumérer tous les points admissibles de (5.1).

Signalons tout de suite une amélioration simple. Dans certains cas, un point admissible peut être trouvé très facilement, et parfois même, un "bon" point admissible (fournie éventuellement par une heuristique, c'est-à-dire par une méthode approchée) est déjà disponible. Dans ce cas, il est licite (et fortement recommandé) d'initialiser le majorant m de la valeur du problème par la valeur de ce point admissible.

5.1.2 Première illustration : énumération et borne naïve pour le problème du voyageur de commerce

Afin de détailler la méthode, considérons le problème du voyageur de commerce, déjà évoqué dans les exemples 1.1 et 1.4. Nous allons traiter ici la version non-orientée du voyageur de commerce. On considère un graphe non-orienté *complet* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ (complet signifie que $\mathcal{E}$ est formé de toutes les paires de sommets de $\mathcal{V}$). On prendra $\mathcal{V} = \{1, \ldots, n\}$, avec $n = |\mathcal{V}|$. Nous associons à chaque paire de sommets $\{i, j\}$ un coût qu'on note c_{ij} (ce qui est une abréviation pour $c_{\{i,j\}}$, on a donc $c_{ij} = c_{ji}$ puisque le graphe est non-orienté). Le but est de trouver un tour, c'est-à-dire, rappelons le, une suite de sommets $\ell_1, \ldots, \ell_n$ comprenant chaque sommet de $\mathcal{G}$ une et une seule fois, et telle que le coût total

$$c_{\ell_1\ell_2} + c_{\ell_2\ell_3} + \cdots + c_{\ell_n\ell_1} \tag{5.4}$$

soit minimum.

En utilisant l'invariance du critère (5.4) par permutation circulaire, on peut toujours supposer que l'on part du sommet 1, soit $\ell_1 = 1$. Le tour est alors spécifié de manière unique par la suite $\ell_2, \ldots, \ell_{n-1}$. La *séparation* du problème revient à organiser le choix d'un tour en une suite de décisions. La première idée qui vient à l'esprit est peut être de considérer le choix de $\ell_2 \in \mathcal{V} \setminus \{\ell_1\}$ comme une première décision, le choix de

$\ell_3 \in \mathcal{V} \setminus \{\ell_1, \ell_2\}$ comme une seconde décision, et ainsi de suite jusqu'à ℓ_{n-1}. Un sommet interne de l'arbre de séparation correspondra donc à une sous-suite $(1 = \ell_1, \ell_2, \ldots, \ell_k)$, avec $k \leqslant n-2$, que nous appellerons "tour partiel". Il s'agit de minorer le coût conditionnel (5.2), i.e. de minorer le coût total des tours qui commencent par $\ell_1, \ldots, \ell_k$. On peut donner par exemple la borne naïve

$$\begin{aligned} b_1(\ell_1, \ldots, \ell_k) \quad = \quad & c_{\ell_1 \ell_2} + \cdots + c_{\ell_{k-1} \ell_k} + \min_{j \in \mathcal{V} \setminus \{\ell_1, \ldots, \ell_{k-1}\}} c_{\ell_k j} + \\ & \min_{m \in \mathcal{V} \setminus \{\ell_2, \ldots, \ell_k\}} c_{m \ell_1} + (n-k-1)\Big(\min_{\substack{j,\, m \,\in\, \mathcal{V} \,\setminus\, \{\ell_1, \ldots, \ell_k\} \\ j \neq m}} c_{jm} \Big). \end{aligned} \tag{5.5}$$

En effet, le coût du tour (5.4) est la somme du coût du tour partiel $(\ell_1, \ell_2, \ldots, \ell_k)$, soit $c_{\ell_1 \ell_2} + \cdots + c_{\ell_{k-1} \ell_k}$, plus du coût de l'arête $\{\ell_k, \ell_{k+1}\}$, que l'on minore par le premier min dans (5.5), plus du coût de l'arête $\{\ell_n, \ell_1\}$, que l'on minore symétriquement par le second min dans (5.5), et enfin, du coût du chemin $(\ell_{k+1}, \ldots, \ell_n)$. Ce chemin étant de longueur $n-k-1$ et ne comprenant aucun sommet de $\{\ell_1, \ldots, \ell_k\}$, on peut minorer son coût par le dernier min dans (5.5), ce qui montre que $b_1(\ell_1, \ldots, \ell_k) \leqslant v(\ell_1, \ldots, \ell_k)$.

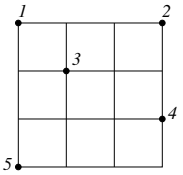


FIGURE 5.1 – Un voyageur de commerce dans Manhattan

Appliquons maintenant l'algorithme de séparation et évaluation, avec la borne b_1, à un petit exemple de voyageur de commerce. Considérons la ville de type Manhattan représentée sur la Figure 5.1. On prendra l'ensemble $\mathcal{V} = \{1, \ldots, 5\}$ dont les éléments correspondent aux points représentés sur le dessin, de coordonnées $P_1 = (0,0)$, $P_2 = (3,0)$, $P_3 = (1,1)$, $P_4 = (3,2)$, et $P_5 = (0,3)$ (dans un repère dirigé vers l'Est et le Sud), et c_{ij} représentera le temps de marche du point P_i au point P_j, c'est-à-dire la norme $\|P_i - P_j\|_1$.

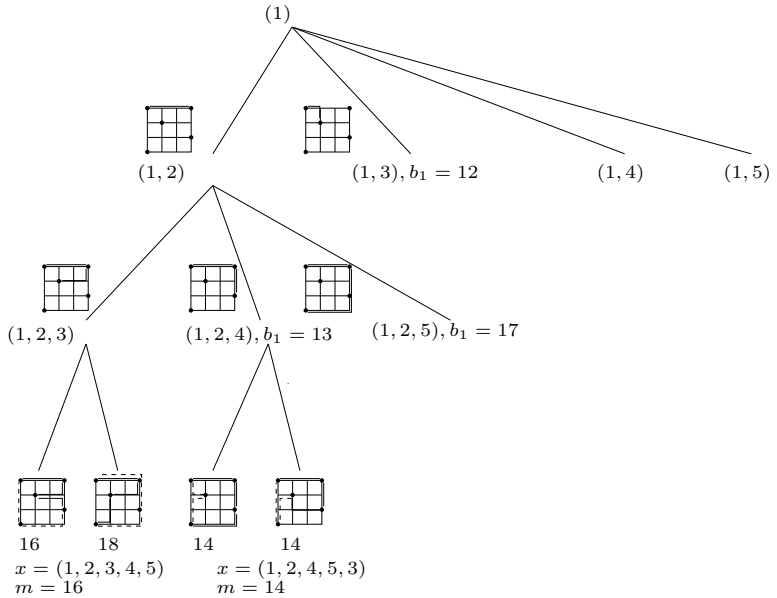


FIGURE 5.2 – Une partie de l'arbre de séparation et évaluation, pour le problème de voyageur de commerce de la Figure 5.1.

Le parcours d'arbre correspondant est représenté sur la Figure 5.2. On part du nœud (1), qui correspond à un tour partiel vide partant du point 1 de la ville. La première étape de l'algorithme consiste à choisir le point suivant de la ville que l'on va visiter, qui peut être 2, 3, 4, ou 5. Choisissons par exemple le point 2, ce qui nous amène au nœud $(1, 2)$ que nous avons représenté à gauche de l'arbre, avec le tour partiel qui lui correspond. Comme nous n'avons pas de point admissible pour l'instant, $m = +\infty$, et comme le test $b_1(1, 2) < m$ est automatiquement vérifié, nous ne calculons pas $b_1(1, 2)$. Poursuivons le parcours en profondeur d'abord : on arrive au nœud $(1, 2, 3)$ (nous avons à nouveau représenté le tour partiel), puis au nœud $(1, 2, 3, 4)$, qui est une feuille, car il détermine de manière unique le point admissible $x = (1, 2, 3, 4, 5)$, de coût $m = 16$. Le tour complet ainsi obtenu est représenté à l'endroit de la feuille. Étant arrivée à une feuille, la recherche en profondeur remonte au nœud père, pour redescendre à la feuille suivante, qui représente le point admissible $(1, 2, 3, 5, 4)$, de coût 18 pire que m. Le prochain nouveau nœud exploré est $(1, 2, 4)$, et l'on calcule $b_1(1, 2, 4) = 13 < m$: on explore donc un premier fils de $(1, 2, 4)$, qui fournit le point admissible $x = (1, 2, 4, 5, 3)$, de coût 14, ce qui est mieux que m : on pose donc $m = 14$. L'autre fils de $(1, 2, 4)$ fournit un autre point admissible de coût 14, $(1, 2, 4, 3, 5)$. Le nouveau nœud suivant visité est $(1, 2, 5)$, avec $b_1(1, 2, 5) = 17 \geqslant m$: on coupe donc le sous-arbre partant de $(1, 2, 5)$, et le nouveau nœud suivant est $(1, 3)$. Nous laissons le lecteur finir le calcul, et montrer ainsi que $x = (1, 2, 4, 5, 3)$ est un tour dont le coût 14 est optimal.

5.1.3 De l'importance de la qualité de la borne

Le lecteur pourrait croire que la borne b_1 est raisonnable, mais voyons comment l'algorithme se comporte quand on augmente le nombre de sommets du graphe. Nous avons programmé l'algorithme qui précède, et résolu des instances similaires de voyageur de commerce dans Manhattan, mais en faisant varier le nombre n de sommets. Voici un jeu de résultats typique, donnant le nombre de nœuds (effectivement explorés) de l'arbre de séparation et évaluation, en fonction de n :

n	7	8	9	10	11	12	13	14	15
arbre	51	805	2175	10598	58414	199276	499887	1250530	3598585

Le cas $n = 15$ prend déjà une minute sur un PC usuel. Nous venons de rencontrer ce qu'on appelle l'*explosion combinatoire*.

Rétrospectivement, le caractère grossier de la borne b_1 apparaît : lorsque $n - k$ est grand, il est mauvais de minorer la longueur du tour partiel complémentaire de $(\ell_1, \ldots, \ell_k)$ par $(n - k - 1)$ fois le minimum des coûts des arêtes entre sommets restants, cela revient à multiplier une erreur par un terme d'ordre n. On voit ici que dans un algorithme de séparation et d'évaluation, il est indispensable d'avoir une borne qui prenne en compte suffisamment la "physique" du problème. Une amélioration immédiate serait de prendre la somme des $(n - k - 1)$ plus petites arêtes. Nous donnons ci-après une borne moins naïve (1-arbre) mais qui peut sembler "sortie du chapeau". Nous verrons plus loin comment construire systématiquement des bornes à l'aide de formulations par programmes linéaires en nombre entiers.

5.1.4 Borne du 1-arbre pour le voyageur de commerce

Cette borne due à Held et Karp repose sur la similarité entre le problème du voyageur de commerce et un autre problème beaucoup plus simple d'optimisation combinatoire, le problème de l'arbre couvrant de poids minimum. Nous rappelons en Annexe A quelques résultats de base sur ce problème et sur sa résolution rapide à l'aide d'un algorithme dit glouton. Pour l'instant, il nous suffit de savoir qu'un arbre couvrant d'un graphe non-orienté $\mathcal{G}$ est un sous-graphe de $\mathcal{G}$, connexe, sans cycle, et tel que tout sommet de $\mathcal{G}$ soit extrémité d'au moins une arête de ce sous-graphe. Le coût d'un arbre couvrant est par définition la somme des coûts de ses arêtes. Observons qu'un arbre couvrant présente des similarités avec un tour : dans les deux cas, on a des sous-graphe connexes, et tout sommet du graphe est extrémité d'au moins une arête du sous-graphe considéré. Cependant, dans un tour, un sommet a exactement deux voisins, alors que dans un arbre, le nombre de voisins d'un sommet est quelconque. Par ailleurs, un arbre couvrant n sommet est composé de $n - 1$ arêtes, alors qu'un tour visitant n sommets est composé de n arêtes.

Ces observations conduisent à la borne suivante. Supposons fixé un tour partiel $(\ell_1, \ell_2, \ldots, \ell_k)$ de $\mathcal{G}$, avec $2 \leqslant k \leqslant n-2$, et construisons le graphe $\mathcal{G}'$ induit par le sous ensemble de sommets $\mathcal{V}' = \mathcal{V} \setminus \{\ell_2, \ldots, \ell_{k-1}\}$, c'est-à-dire le graphe d'ensemble de sommets $\mathcal{V}'$ et d'ensemble d'arêtes $\mathcal{E}' = \{\{i,j\} \in \mathcal{E} \mid i,j \in \mathcal{V}'\}$. Nous notons $\mathrm{ac}(\mathcal{G}')$ le coût minimum d'un arbre couvrant $\mathcal{G}'$. Si l'on complète le tour partiel $(\ell_1, \ldots, \ell_k)$ en un tour $(\ell_1, \ldots, \ell_{n-1}, \ell_n)$, on voit que le sous-graphe de $\mathcal{G}$ formé des arêtes $\{\ell_k, \ell_{k+1}\}$,..., $\{\ell_n, \ell_1\}$, est un arbre couvrant les sommets $\ell_k, \ldots, \ell_n, \ell_1$ (car ce sous-graphe est en fait un chemin). On a donc le minorant suivant du coût de tous les tours prolongeant le tours partiel $(\ell_1, \ldots, \ell_k)$:

$$b_2(\ell_1, \ldots, \ell_k) = c_{\ell_1\ell_2} + \cdots + c_{\ell_{k-1}\ell_k} + \mathrm{ac}(\mathcal{G}') \ . \tag{5.6}$$

Il reste à considérer le cas spécial où le tour partiel est de longueur nulle, soit $k = 1$. Dans ce cas, on forme le sous graphe $\mathcal{G}'$ induit par le sous-ensemble de sommets $\mathcal{V}' = \mathcal{V} \setminus \{\ell_1\}$, et l'on emploie la borne :

$$b_2(\ell_1) \quad = \quad \min_{\substack{j,s \in \mathcal{V}\setminus\{\ell_1\} \\ j \neq s}} c_{\ell_1 j} + c_{\ell_1 s} + \mathrm{ac}(\mathcal{G}') \ . \tag{5.7}$$

Cette dernière borne, classique, est connue sous le nom de *borne du* 1-*arbre* : un 1-arbre d'un graphe est un sous-graphe formé d'une part d'un arbre couvrant tous les sommets hormis un sommet distingué noté "1", et d'autre part deux arêtes reliant le sommet "1" à deux autres sommets. Le coût minimal d'un 1-arbre s'obtient en calculant le coût minimal d'un arbre couvrant tous les sommets sauf le sommet 1, auquel il faut rajouter le coût minimum de deux arêtes distinctes reliant 1 à d'autres sommets, ce qui fournit bien le second membre de (5.7).

Par exemple, pour le graphe de la Figure 5.1, en prenant $\ell_1 = 1$, et en appliquant l'algorithme de Kruskal au sous-graphe de sommets $2, 3, 4, 5$, on obtient le 1-arbre de coût minimal représenté en traits gras sur la Figure 5.3. Le coût de ce 1-arbre est de 8. En lui rajoutant les coûts des deux arêtes en pointillées, on obtient la borne $b_2(1) = 13$. Nous laissons le lecteur, en guise d'exercice, constater que l'arbre de la Figure 5.2 se simplifie beaucoup si l'on utilise la borne b_2 au lieu de b_1.

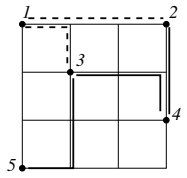


FIGURE 5.3 – Borne du 1-arbre $b_2(1)$, pour le problème du voyageur de commerce de la Figure 5.1. L'arbre couvrant $\mathcal{G}'$ est en traits gras, les deux arêtes connectant cet arbre au sommet 1 sont en traits pointillés.

5.1.5 Du choix de l'arbre et de l'ordre d'exploration des branches

Il y a en général plusieurs manières de représenter l'ensemble des points admissibles X, et le choix est souvent suggéré par la technique servant à fabriquer la borne. Ainsi, nous verrons dans la Sous-section 5.2 qu'on peut modéliser le voyageur de commerce par un programme linéaire en variables entières, en introduisant pour chaque $\{i,j\} \in \mathcal{E}$ une variable x_{ij} valant 1 si l'arête $\{i,j\}$ appartient au tour considéré et 0 sinon. Avec une telle modélisation, on peut construire un arbre de séparation et d'évaluation binaire, dans lequel une décision élémentaire consiste à fixer la valeur d'une variable x_{ij} à 0 ou à 1. La borne du 1-arbre n'apparaît alors plus comme une astuce, mais comme le produit d'une méthode systématique.

Un autre paramètre déterminant est l'ordre dans lequel on visite les sommets : il est judicieux d'examiner le plus tôt possible (i.e., près de la racine de l'arbre) les décisions dont on pense qu'elles ont le plus d'influence sur le coût du point admissible, le but étant de couper les branches le plus haut possible dans l'arbre.

5.2 Relaxation de problèmes combinatoires

Une manière systématique d'obtenir des minorants de la valeur optimale du coût d'un problème combinatoire consiste à *relâcher* (ou relaxer) le problème, c'est-à-dire à grossir l'ensemble admissible de manière à obtenir un problème plus facile, fournissant une borne inférieure pour le problème initial. Quand l'ensemble admissible est représenté par des contraintes, une manière de relâcher est d'oublier tout simplement certaines contraintes. Nous avons déjà vu un exemple de relaxation avec la borne du 1-arbre pour le voyageur de commerce : les tours sont précisément les 1-arbres tels que chaque sommet a deux voisins. Nous allons maintenant présenter des techniques générales de relaxation.

5.2.1 Relaxation continue : principe

L'efficacité des outils de programmation linéaire (que nous présenterons en détail dans les chapitres 6 et 10) suggère souvent de modéliser les problèmes combinatoires par des programmes linéaires en nombre entiers : on obtient alors une borne inférieure en relâchant les contraintes d'intégrité. De manière formelle, on part d'un problème linéaire en nombre entiers,

$$\inf\{c \cdot x \mid Ax \leqslant b,\ x \in \mathbb{Z}^n\} \ ,$$

que l'on le remplace tout simplement par le problème

$$\inf\{c \cdot x \mid Ax \leqslant b,\ x \in \mathbb{R}^n\} \ ,$$

appelé *problème relaxé continu*, ce qui fournit évidemment un minorant de la valeur du problème combinatoire. Si la solution x^* du problème relaxé continu est entière, on est assuré d'avoir trouvé l'optimum du problème initial. Dans le cas contraire, on peut trouver un indice i telle que la coordonnée x_i^* ne soit pas entière, et l'on sait que toute solution x du problème combinatoire, étant entière, vérifie nécessairement

$$x_i \leqslant \lfloor x_i^* \rfloor \qquad \text{ou} \qquad x_i \geqslant \lceil x_i^* \rceil \quad ,$$

où $\lfloor \cdot \rfloor$ désigne la partie entière par le dessous (plus grand entier inférieur à un réel) et où $\lceil \cdot \rceil$ désigne la partie entière par le dessus (plus petit entier supérieur à un réel). Cette observation peut être exploitée dans la génération de l'arbre de séparation et évaluation. Ainsi, le programme linéaire dans lequel on a rajouté une contrainte

$$\inf\{c \cdot x \mid Ax \leqslant b,\ x \in \mathbb{R}^n, x_i \leqslant \lfloor x_i^* \rfloor\}$$

fournit un minorant facilement calculable permettant de décider de l'intérêt d'explorer la branche $x_i \leqslant \lfloor x_i^* \rfloor$, et l'on forme de même un second programme comprenant l'inégalité $x_i \geqslant \lceil x_i^* \rceil$ pour décider de l'intérêt d'explorer l'autre branche.

Dans le cas du sac à dos, cette méthode s'applique de manière très directe, chaque problème relaxé pouvant même être résolu par un calcul explicite (plutôt qu'en faisant appel à la programmation linéaire).

Exercice 5.1 (Résolution d'un problème de sac à dos) Considérons n objets pouvant rentrer dans un sac. Notons $u_i > 0$ l'utilité de l'objet i, $p_i > 0$ son poids, et M un poids maximum que l'on est disposé à porter. Rappelons que le problème du sac à dos consiste à trouver un sous-ensemble I de $\{1, \ldots, n\}$ tel que le poids emporté $\sum_{i \in I} p_i$ soit au plus M qui maximise l'utilité totale $\sum_{i \in I} u_i$. On suppose, sans perte de généralité, que $\sum_{1 \leqslant j \leqslant n} p_j > M$ et que $M \geqslant p_i$, pour tout i. En posant $x_i = 1$ si $i \in I$, et $x_i = 0$ sinon, on modélise ce problème par le PLNE :

$$\max_{\substack{x \in \{0,1\}^n \\ \sum_{1 \leqslant i \leqslant n} p_i x_i \leqslant M}} \sum_{1 \leqslant i \leqslant n} u_i x_i \ . \tag{5.8}$$

1. Proposer une méthode heuristique pour trouver une solution approchée de ce problème.

2. On suppose les indices ordonnés par ratios utilité/poids croissants :

$$\frac{u_1}{p_1} \leqslant \cdots \leqslant \frac{u_n}{p_n} \ . \tag{5.9}$$

Démontrer que le problème relâché continu de (5.8) admet la solution explicite suivante. On choisit $\bar{\imath}$ tel que

$$\sum_{j=\bar{\imath}}^{n} p_j > M \geqslant \sum_{j=\bar{\imath}+1}^{n} p_j \ .$$

La solution est alors donnée par $\bar{x}_{\bar{\imath}+1} = \cdots = \bar{x}_n = 1$, $\bar{x}_{\bar{\imath}} = (M - p_n - \cdots - p_{\bar{\imath}+1})/p_{\bar{\imath}}$, et $\bar{x}_0 = \ldots = \bar{x}_{\bar{\imath}-1} = 0$.

3. Écrire le problème linéaire dual du problème relâché continu de (5.8). Montrer que la résolution de ce problème dual se ramène à la minimisation d'une fonction convexe non-différentiable en dimension 1, le résoudre explicitement, et retrouver ainsi la conclusion de la question précédente.

4. Résoudre à l'aide d'une méthode de séparation et évaluation le problème suivant :

$$\max_{\substack{x \in \{0,1\}^3 \\ 2x_1 + 5x_2 + 3x_3 \leqslant 6}} 8x_1 + 9x_2 + 2x_3 \ .$$

5. Même question avec

$$\max_{\substack{x \in \{0,1\}^4 \\ 5x_1 + 3x_2 + 8x_3 + 7x_4 \leqslant 12}} 17x_1 + 10x_2 + 25x_3 + 17x_4 \ .$$

6. Discuter les avantages et inconvénients de l'algorithme de séparation évaluation par rapport à la programmation dynamique, pour résoudre le problème du sac à dos.

5.2.2 Formulation PLNE du problème de voyageur de commerce

Nous présentons maintenant la relaxation du voyageur de commerce proposée par Dantzig, Fulkerson, et Johnson dans un article de 1954, qui résolvait à l'époque un problème à 49 villes. La postérité des idées de cet article permet de résoudre aujourd'hui exactement des instances à plusieurs milliers de villes [1]. L'idée est encore d'exploiter un programme linéaire en nombre entiers, avec une difficulté supplémentaire liée au fait que celui-ci a un nombre exponentiel de contraintes.

Nous reprenons les notations de la section précédente pour le voyageur de commerce : $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ est un graphe non orienté complet (i.e. $\mathcal{E}$ contient toutes les arêtes reliant deux éléments de $\mathcal{V}$), avec une fonction coût $c : \mathcal{E} \to \mathbb{R}$, $\{i,j\} \mapsto c_{ij}$. A chaque tour, on associe un vecteur $x \in \{0,1\}^{\mathcal{E}}$ tel que $x_{ij} = 1$ si $\{i,j\}$ fait partie du tour, et $x_{ij} = 0$ sinon. Réciproquement, un vecteur $x \in \{0,1\}^{\mathcal{E}}$ représente le sous-graphe de $\mathcal{G}$ ayant pour arêtes les $\{i,j\}$ tels que $x_{ij} = 1$. Il nous faut maintenant exprimer par des contraintes linéaires le fait que $x \in \{0,1\}^{\mathcal{E}}$ représente un tour. Comme chaque sommet d'un tour a exactement deux voisins, x vérifie nécessairement

$$\sum_{k \in \mathcal{V}, \{k,j\} \in \mathcal{E}} x_{kj} = 2 \ , \text{ pour tout } j \in \mathcal{V}. \tag{5.10}$$

Les contraintes (5.10) ne suffisent pas à caractériser un tour, car l'ensemble des arêtes $\{i,j\}$ telles que $x_{ij} = 1$ peut ne pas être connexe : en fait, on peut voir que les $x \in \{0,1\}^{\mathcal{E}}$ solutions de (5.10) représentent exactement les unions disjointes de circuits. Afin d'éliminer les solutions parasites, on peut rajouter les contraintes suivantes, dites de *sous-tour*,

$$\text{pour tout } \mathcal{S} \subset \mathcal{V}, \text{ tel que } \mathcal{S} \neq \emptyset \text{ et } \mathcal{S} \neq \mathcal{V}, \quad \sum_{k,m \in \mathcal{S}, \{k,m\} \in \mathcal{E}} x_{km} \leqslant |\mathcal{S}| - 1 \ . \tag{5.11}$$

1. Pour un historique et un état de l'art récent, on pourra se reporter à [ABCC06] et plus généralement à la toile http ://www.math.princeton.edu/tsp.

On voit facilement que $x \in \{0,1\}^{\mathcal{E}}$ vérifie (5.10) et (5.11) si, et seulement si, il représente un tour : en effet, tout vecteur associé à un tour vérifie ces contraintes, et réciproquement, comme tout $x \in \{0,1\}^{\mathcal{E}}$ vérifiant (5.10) représente une union disjointe de circuits, il suffit de prendre pour $\mathcal{S}$ l'ensemble des sommets de l'un quelconque de ces circuits pour obtenir $\sum_{k,m\in\mathcal{S},\{k,m\}\in\mathcal{E}} x_{km} = |\mathcal{S}|$, et si x vérifie les inégalités de sous-tour (5.11), on a nécessairement $\mathcal{S} = \mathcal{V}$, ce qui montre que x représente un tour.

Ceci nous permet de formuler le problème de voyageur de commerce comme un programme linéaire en nombres entiers

$$(VC): \quad \min \sum_{\{i,j\}\in\mathcal{E}} c_{ij}x_{ij} \quad \text{sous les contraintes } x \in \{0,1\}^{\mathcal{E}},\ (5.10), (5.11),$$

et on obtient aussitôt une borne inférieure en considérant le programme linéaire relaxé en variables continues

$$(VC)_{\text{rel}}: \quad \min \sum_{\{i,j\}\in\mathcal{E}} c_{ij}x_{ij} \quad \text{sous les contraintes } 0 \leqslant x_{ij} \leqslant 1,\ (5.10), (5.11).$$

Malgré le nombre exponentiel de contraintes dans (5.11), il est possible de résoudre efficacement $(VC)_{\text{rel}}$ en procédant comme suit :

- On commence à minimiser $\sum_{\{i,j\}\in\mathcal{E}} c_{ij}x_{ij}$ sous les contraintes $0 \leqslant x_{ij} \leqslant 1$ et (5.10), en oubliant les contraintes de sous-tour (5.11). On trouve ainsi un premier $x \in [0,1]^{\mathcal{E}}$.
- On va ensuite chercher s'il existe un sous ensemble $\mathcal{S} \subset \mathcal{V}$, $\mathcal{S} \neq \emptyset$, $\mathcal{S} \neq \mathcal{V}$, pour lequel l'inégalité de sous-tour (5.11) n'est pas vérifiée, ce qui peut se faire efficacement à l'aide d'algorithmes de flots (Exercice 5.2) ou bien d'algorithmes spéciaux (Remarque 5.3).
- S'il n'en existe pas, on a résolu $(VC)_{\text{rel}}$.
- Si au contraire on a trouvé un $\mathcal{S}$ tel que (5.11) ne soit pas satisfaite, l'on minimise à nouveau $\sum_{\{i,j\}\in\mathcal{E}} c_{ij}x_{ij}$ sous les contraintes $0 \leqslant x_{ij} \leqslant 1$ et (5.10), en rajoutant l'inégalité de sous-tour (5.11) associée à $\mathcal{S}$.

En poursuivant cette suite de minimisations et rajouts successifs d'inégalités, on aboutit finalement à une solution de $(VC)_{\text{rel}}$.

Cette méthode peut s'interpréter géométriquement en parlant de *coupes*[2]. Notons en effet P le polytope des points admissibles du problème $(VC)_{\text{rel}}$, et soit $c : x \mapsto \sum_{\{i,j\}\in\mathcal{E}} c_{ij}x_{ij}$ la forme linéaire que l'on minimise. La méthode esquissée revient à définir une suite décroissante de polytopes $P^1 \supset P^2 \supset \cdots \supset P$. On prend tout d'abord pour P^1 l'ensemble défini par $0 \leqslant x_{ij} \leqslant 1$ et (5.10). On minimise d'abord c sur P^1, et le minimum est atteint en un point x^1. Si x^1 est dans P, on a résolu $(VC)_{\text{rel}}$, sinon détecter une contrainte (5.11) violée par x^1 revient à *séparer* x^1 de P, c'est-à-dire à trouver un demi-espace H^1 particulier tel que $x^1 \notin H^1$, et $P \subset H^1$, et l'étape suivante revient à minimiser c sur le nouveau polytope $P^2 = P^1 \cap H^1$, obtenu en "coupant" P^1 par H, ce qui fournit un nouveau point x^2, etc. L'on fabrique ainsi une suite décroissante de polytopes dont on peut dire intuitivement qu'ils "approchent" P au voisinage du point où c est minimal.

Il est essentiel, pour mettre en œuvre de cette méthode, de pouvoir séparer efficacement à chaque étape, le point x^k du polytope P : en l'occurrence, il s'agit de vérifier si x^k ne vérifie pas l'une des contraintes de sous-tour (5.11). Autrement dit, gérer un nombre considérable de contraintes en programmation linéaire reste possible, si l'on sait décider facilement (par une méthode évitant l'énumération des contraintes) si l'une de ces contraintes n'est pas vérifiée (cette idée sera reprise au chapitre 8). La procédure que nous venons de décrire permet en l'occurrence de résoudre de manière optimale le problème relaxé continu $(VC)_{\text{rel}}$.

On a parfois la chance que la solution x de ce problème relaxé soit entière : dans ce cas, x c'est évidemment solution du problème combinatoire (VC). Sinon, on pourrait commencer à générer un arbre de séparation et évaluation. Cependant, avant cela, l'état de l'art [3] consiste à rajouter d'autres coupes, dites coupes d'intégrité,

2. On se gardera de confondre la notion de coupe d'un polyèdre et la notion de coupe d'un graphe (Définition 1.20). Dans l'exercice 5.2, nous utiliserons des coupes de graphes pour trouver une coupe d'un polyèdre !

3. Le solveur concorde http ://www.tsp.gatech.edu/concorde/index.html permet de résoudre de grandes instances du problème du voyageur de commerce sur un graphe non orienté. Un mode verbeux permet de voir les ajouts de coupes ainsi que le déroulement de la procédure de séparation et évaluation. Sur de petites instances, l'optimum est souvent atteint à l'aide de coupes d'intégrité, sans qu'il soit nécessaire de "brancher".

vérifiées par les points entiers du polytope P. Les principes généraux permettant de trouver de telles coupes font l'objet du chapitre 7.

Exercice 5.2 (Coupe minimale globale, contraintes de sous-tour)
Nous avons défini une coupe d'un graphe non-orienté $(\mathcal{V}, \mathcal{E})$, associée à une partition $\mathcal{V} = \mathcal{S} \cup (\mathcal{V} \setminus \mathcal{S})$, comme l'ensemble des arêtes reliant un sommet de $\mathcal{S}$ et un sommet de $\mathcal{V} \setminus \mathcal{S}$. Si chaque arc $\{i, j\}$ est muni d'une capacité u_{ij}, la *capacité* de la coupe est par définition

$$u(\mathcal{S}, \mathcal{V} \setminus \mathcal{S}) = \sum_{k \in \mathcal{S},\ m \in \mathcal{V} \setminus \mathcal{S}, \{k,m\} \in \mathcal{E}} u_{km} \ .$$

On suppose que les capacités u_{ij} sont toutes positives. Le problème de la *coupe minimale* (globale) consiste à trouver une coupe minimisant la capacité $u(\mathcal{S}, \mathcal{V} \setminus \mathcal{S})$.

1. Montrer que dans le problème $(VC)_{\text{rel}}$, on peut remplacer les contraintes (5.11) par les contraintes

$$\text{pour tout } \mathcal{S} \subset \mathcal{V}, \text{ tel que } \mathcal{S} \neq \emptyset \text{ et } \mathcal{S} \neq \mathcal{V}, \quad \sum_{k \in \mathcal{S},\ m \in \mathcal{V} \setminus \mathcal{S}, \{k,m\} \in \mathcal{E}} x_{km} \geqslant 2 \ . \tag{5.12}$$

2. Montrer, en utilisant le théorème de Ford et Fulkerson 3.18 que l'on peut calculer la capacité minimale d'une coupe en fixant un sommet $s \in \mathcal{V}$ quelconque, et en calculant, pour chaque sommet p différent de s, la valeur maximale d'un flot de s à p dans un certain graphe.

Conclure que l'on peut vérifier, en temps polynomial, si un vecteur x positif donné satisfait toutes les inégalités de sous-tour (5.11) (et, le cas échéant, exhiber une inégalité de sous-tour non satisfaite).

Remarque 5.3 Le problème de la coupe minimale globale a fait l'objet de nombreux travaux. On se reportera à [CGK$^+$96] pour une comparaison des différents algorithmes disponibles.

Remarque 5.4 On dit qu'un polyèdre P admet un *oracle de séparation polynomial* s'il existe un programme de temps d'exécution polynomial qui prend en entrée un vecteur y, répond oui si ce vecteur appartient au polyèdre, et répond non dans le cas contraire en retournant une inégalité affine $c \cdot x \leqslant b$ satisfaite par tous les vecteurs x de P mais pas par y (une telle inégalité définit un demi-espace séparant y de P). Par exemple, l'exercice 5.2 montre le polyèdre intervenant dans la formulation du problème du voyageur de commerce, lequel est défini par les contraintes $0 \leqslant x_{ij} \leqslant 1$, (5.10) et par un nombre *exponentiel* de contraintes de sous-tour (5.11), admet un oracle de séparation polynomial (l'exercice montre que décider si les contraintes de sous-tour sont satisfaites revient à résoudre un problème de flot). Un résultat très général, pour lequel nous renvoyons à [Sch86, Chap. 14], affirme que si un polyèdre P admet un oracle de séparation polynomial, alors on sait minimiser une forme linéaire sur P en temps polynomial. Ce résultat est en fait un sous-produit de l'algorithme de Khachyian à base d'ellipsoïdes, qui a été le premier algorithme permettant de résoudre en temps polynomial des programmes linéaires. Ceci fournit une base théorique permettant de justifier de l'efficacité des méthodes consistant à ajouter successivement des inégalités comme on l'a fait plus haut pour résoudre la relaxation continue du problème de voyageur de commerce, même si l'algorithme de Khachyian n'est pas utilisé en pratique (on utilise plutôt l'algorithme du simplexe ou des méthodes modernes de points intérieurs).

Exercice 5.5 (Problème de tournée sans contrainte de capacité) Une compagnie délivre des objets à n clients, au moyen de K camions qui partent du dépôt chaque matin et rentrent au dépôt chaque soir. On connaît la distance entre deux clients, et entre un client et le dépôt. On suppose que chaque camion a une capacité de b objets, et que a_i objets doivent être livrés au client i. On veut établir un système de tournées minimisant la distance totale parcourue par les camions (sachant qu'un client n'est desservi que par un camion). On se limite pour l'instant au cas sans capacités : la capacité b est supposée infinie, et la quantité a_i devient alors sans intérêt. Montrer que ce problème ce ramène à un problème du voyageur de commerce. (Indication : on pourra considérer un graphe avec K copies du dépôt.)

Exercice 5.6 (Relaxation du problème du voyageur de commerce non-symétrique). Formuler le problème de voyageur de commerce non-symétrique comme un PLNE. Proposer une relaxation faisant intervenir un problème d'affectation optimale. Quel est le défaut de cette relaxation dans le cas particulier symétrique ?

5.2.3 Relaxations lagrangiennes

Principe

Considérons le problème très général

$$\begin{array}{l} \text{minimiser } \ f(x) \ \text{ sous les contraintes :} \\ \quad x \in X, \\ \quad g_i(x) = 0, \ i = 1, \ldots, p, \\ \quad g_i(x) \leqslant 0, \ i = p+1, \ldots, p+q, \end{array} \tag{5.13}$$

où $f, g_1, \ldots, g_{p+q}$ sont des fonctions de $\mathbb{R}^n$ dans $\mathbb{R}$, et X est un sous-ensemble non-vide de $\mathbb{R}^n$. Considérons le lagrangien $\mathcal{L} : X \times \Lambda \to \mathbb{R}$, avec $\Lambda = \mathbb{R}^p \times (\mathbb{R}_+)^q$ et

$$\mathcal{L}(x, \lambda) = f(x) + \lambda_1 g_1(x) + \cdots + \lambda_{p+q} g_{p+q}(x) \ .$$

Soit f^* la valeur optimale de (5.13). Nous avons déjà noté dans la Proposition 2.41 qu'on a toujours l'inégalité de dualité faible

$$f^* = \inf_{x \in X} \sup_{\lambda \in \Lambda} \mathcal{L}(x, \lambda) \geqslant \sup_{\lambda \in \Lambda} \inf_{x \in X} \mathcal{L}(x, \lambda) = \sup_{\lambda \in \Lambda} \mathcal{G}(\lambda) \ , \tag{5.14}$$

où

$$\mathcal{G} : \Lambda \to \mathbb{R} \cup \{-\infty\}; \ \ \mathcal{G}(\lambda) = \inf_{x \in X} \mathcal{L}(x, \lambda) \tag{5.15}$$

est la fonction duale. La méthode de la *relaxation lagrangienne* consiste à employer le second membre de (5.14) comme minorant de la valeur f^* du problème original (5.13). Elle est souvent utile dans des situations où le calcul de $\mathcal{G}(\lambda)$, à λ donné, peut être réalisé efficacement (par exemple par un algorithme de graphe).

N'importe quel multiplicateur de Lagrange $\lambda \in \Lambda$ fournit un minorant $\mathcal{G}(\lambda) \leqslant f^*$, mais il est naturel de chercher le meilleur minorant possible, ce qui revient à maximiser $\mathcal{G}$. Or $\mathcal{G}$, qui est un infimum de fonctions affines, est concave. Si, comme c'est le cas pour la plupart des problèmes combinatoires, X est fini, $\mathcal{G}$ qui est un infimum fini de fonctions affines, est une fonction *non-différentiable* (sauf bien sûr dans des cas dégénérés). Maximiser $\mathcal{G}$ relève donc de l'*optimisation non-différentiable*, qui traite de la minimisation de fonctions convexes non-différentiables (ou symétriquement, de la maximisation de fonctions concaves non-différentiables). On peut employer des algorithmes d'optimisation d'une fonction convexe basés sur l'usage des sous gradients (voir par exemple [HL96, BGLS06]). Présentons la plus simple, dite de la *série divergente* ; c'est une généralisation des méthodes de gradient de l'optimisation différentiable.

Calcul d'un sur-gradient de la fonction duale

Nous avons déjà défini dans la section 2.5 le *sous-différentiel* au point x d'une fonction f :

$$\partial f(x) = \{p \in \mathbb{R}^n \mid \ f(y) - f(x) \geqslant p \cdot (y - x), \qquad \forall y \in \mathbb{R}^n\} \ . \tag{5.16}$$

(La fonction f est de $\mathbb{R}^n \to \bar{\mathbb{R}}$, mais l'on suppose que $f(x)$ est fini pour définir le sous-différentiel en x.) Les éléments de $\partial f(x)$ sont appelés *sous-gradients* au point x : ce sont les pentes des fonctions affines minorant f et coïncidant avec f au point x. L'intérêt du sous-différentiel est que

$$0 \in \partial f(x) \iff x \text{ est un point de minimum de } f,$$

ce qui résulte aussitôt de la définition.

Les notions de sur-différentiel et de sur-gradient sont définies symétriquement en renversant l'inégalité dans (5.16). En particulier, un *sur-gradient* p de f en un point x tel que $f(x)$ est fini est défini par l'inégalité suivante :

$$f(y) - f(x) \leqslant p \cdot (y - x), \qquad \forall y \in \mathbb{R}^n \ . \tag{5.17}$$

Les notions de sous-différentiel et de sous-gradient sont surtout utiles dans le cas d'une fonction f convexe (on peut montrer en particulier, à l'aide du théorème de séparation, que le sous-différentiel en tout point d'une fonction convexe à valeurs finies est non-vide). Comme la fonction duale $\mathcal{G}$ est concave, on s'intéressera plutôt à ses sur-gradients.

Appliquons donc ces notions à la fonction duale $\mathcal{G}$. Il sera commode de prolonger $\mathcal{G}$ à $\mathbb{R}^{p+q}$ en posant $\mathcal{G}(\lambda) = -\infty$, si $\lambda \notin \Lambda$, ce qui définit bien une fonction concave de $\mathbb{R}^{p+q}$ dans $\mathbb{R} \cup \{-\infty\}$. Un sur-gradient de la fonction duale $\mathcal{G}$ se calcule aisément, à l'aide de l'observation suivante.

Proposition 5.7 *Supposons X fini, soit $\mathcal{G}$ la fonction duale définie par* (5.15)*, et pour tout $\lambda \in \Lambda$, posons $\Gamma(\lambda) = \operatorname{argmin}_{x \in X} \mathcal{L}(x, \lambda) = \{x \in X \mid \mathcal{L}(x, \lambda) = \mathcal{G}(x)\}$. Alors, pour tout $x \in \Gamma(\lambda)$, $g(x) = (g_i(x))_{1 \leqslant i \leqslant p+q}$ est un sur-gradient de $\mathcal{G}$ au point λ.*

Démonstration. Pour tout $x \in \Gamma(\lambda)$, et pour tout $\mu \in \Lambda$, on a $\mathcal{G}(\mu) - \mathcal{G}(\lambda) \leqslant \mathcal{L}(x, \mu) - \mathcal{L}(x, \lambda) = g(x) \cdot (\mu - \lambda)$, ce qui montre que $g(x)$ est un sur-gradient de $\mathcal{G}$ au point λ. ■

Remarque 5.8 La Proposition 5.7 est une version faible de la règle de "sous-différentiation sous le signe max", qui fait l'objet du Lemme 2.62. •

Énoncé de la méthode de sur-gradient

Soit P_Λ la projection de $\mathbb{R}^{p+q}$ dans Λ, $\lambda \mapsto (\lambda_1, \ldots, \lambda_p, \lambda_{p+1}^+, \ldots, \lambda_{p+q}^+)$, où l'on a posé $t^+ = \max(0, t)$. L'*algorithme de sur-gradient* (projeté) pour maximiser la fonction concave $\mathcal{G}$ consiste à construire la suite

$$\lambda_{k+1} = P_\Lambda\Big(\lambda_k + \frac{\rho_k}{\|p_k\|} p_k\Big) \ , \tag{5.18}$$

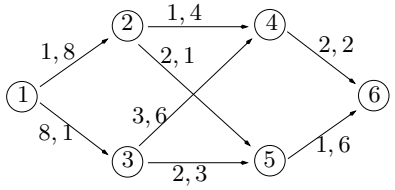
où $\lambda_0 \in \Lambda$ est choisi arbitrairement, où p_k est un sur-gradient quelconque de $\mathcal{G}$ au point λ_k, et où ρ_k est une suite de réels strictement positifs telle que

$$\rho_k \to 0, \quad \sum_i \rho_i = +\infty \ . \tag{5.19}$$

Évidemment, la valeur λ_{k+1} n'est bien définie que si $p_k \neq 0$. Lorsque $p_k = 0$, l'algorithme s'arrête : λ_k est alors le maximum de f (par définition même des sur-gradients la nullité d'un sur-gradient en un point implique que la fonction est maximale en ce point).

Contrairement aux algorithmes de gradient pour des fonctions différentiables, le pas ρ_k doit tendre vers 0 pour assurer la convergence, ce qui rend cet algorithme particulièrement lent.

Application 5.9 (Coût minimum avec contraintes de temps). Illustrons la relaxation lagrangienne en traitant l'exemple du chemin de coût minimum avec contrainte de temps, déjà mentionné dans l'exemple 1.14. Nous considérons donc un graphe orienté $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, l'arc (i, j) étant muni du coût c_{ij} et du temps τ_{ij}. Pour fixer les idées, on cherchera le chemin de coût minimum et de temps total au plus 10, allant du nœud source $s = 1$ au nœud puits $p = 6$, dans le graphe



(5.20)

On a représenté sur chaque arc les valuations c et τ, dans cet ordre, par exemple, l'arc $(1,2)$ coûte $c_{12}=1$ et prend $\tau_{12}=8$ unités de temps.

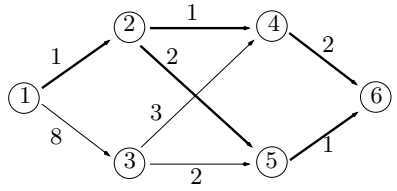
Il nous faut tout d'abord formuler ce problème sous la forme (5.13). Pour cela, nous représenterons un chemin par le vecteur Booléen $x \in \mathbb{R}^{\mathcal{A}}$ tel que $x_{ij}=1$ si (i,j) appartient au chemin, et $x_{ij}=0$ sinon. Le problème de plus court chemin d'une source s à un puits p en temps au plus T s'écrit alors

$$\begin{array}{rl} \text{minimiser} & \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} \\ \text{sous les contraintes :} & x \in \{0,1\}^{\mathcal{A}}, \\ & x \text{ représente un chemin de } s \text{ à } p, \\ & \sum_{(i,j)\in\mathcal{A}} \tau_{ij}x_{ij} \leqslant T \ . \end{array} \tag{5.21}$$

Il y a bien sûr plusieurs manières d'écrire un programme (5.13). Ainsi, c'est volontairement que nous avons laissé la contrainte "x représente un chemin de s à p" sous forme littérale. Nous aurions pu expliciter cette contrainte en écrivant la loi des nœuds de Kirchhoff (3.1). Mais dualiser la loi des nœuds serait une mauvaise idée : le succès de la relaxation lagrangienne dépend précisément de la capacité à identifier le plus "petit" ensemble de contraintes dont la relaxation conduit à un problème plus simple, résoluble de préférence par une méthode combinatoire directe. Ici, c'est la contrainte de temps qu'il faut relâcher, car si l'on oublie cette contrainte, on obtient un pur problème de chemin de coût minimum. En dualisant la contrainte de temps, la fonction $\mathcal{G}:\mathbb{R}_+ \to \mathbb{R}\cup\{-\infty\}$ s'écrit en effet

$$\begin{aligned} \mathcal{G}(\lambda) &= \inf_{\substack{x\in\{0,1\}^{\mathcal{A}} \\ x \text{ représente un chemin de } s \text{ à } p}} \Big(\sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij}\Big) + \lambda\Big(\sum_{(i,j)\in\mathcal{A}} \tau_{ij}x_{ij} - T\Big) \ , \\ &= -\lambda T + \inf_{\substack{x\in\{0,1\}^{\mathcal{A}} \\ x \text{ représente un chemin de } s \text{ à } p}} \sum_{(i,j)\in\mathcal{A}} (c_{ij}+\lambda\tau_{ij})x_{ij} \ . \end{aligned}$$

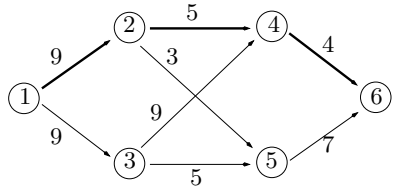
A λ fixé, le calcul de $\mathcal{G}(\lambda)$ revient à résoudre un problème classique de chemin de coût minimum *sans contraintes de temps* dans le graphe de coûts $c_{ij}+\lambda\tau_{ij}$, ce qui peut se faire par programmation dynamique, et qui prend même un temps linéaire dans le cas d'un graphe sans circuits, comme c'est le cas pour l'exemple (5.20). Appliquons maintenant l'algorithme de sur-gradient (5.18) sur cet exemple. Pour $\lambda_1=0$, il nous faut trouver le chemin de coût minimum $1\to 6$ dans le graphe muni des coûts $c_{ij}+0\tau_{ij}$



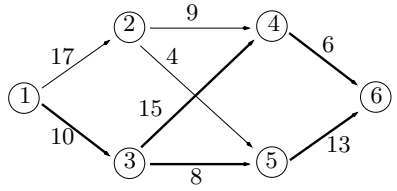
L'ensemble $\Gamma(0)$ des chemins optimaux est réduit aux chemins $(1,2,4,6)$, et $(1,2,5,6)$ représentés en traits gras, qui ont pour coût 4. Ainsi, $\mathcal{G}(0)=4-0T=4$. D'après la Proposition 5.7, on a le sur-gradient

$$p_1 = \tau_{12}+\tau_{24}+\tau_{46}-T = 4 \ .$$

Avec un pas $\rho_1=1$, (5.18) donne $\lambda_2=1$, et le nouveau graphe muni des coûts $c_{ij}+1\tau_{ij}$



Comme le chemin optimal n'a pas changé, on a le même sur-gradient $p_2 = p_1$. En prenant par exemple $\rho_2 = 1$, on a $\lambda_3 = 2$, ce qui donne le graphe



Il y a cette fois ci deux chemins optimaux : $\Gamma(\lambda_3) = \{(1,3,4,6),(1,3,5,6)\}$, de coût 21. En particulier, le sur-gradient correspondant au chemin $(1,3,5,6)$ est $\tau_{13} + \tau_{35} + \tau_{56} - T = 0$. L'algorithme de sur-gradient s'arrête donc : $\max_{\lambda\in\mathbb{R}_+} \mathcal{G}(\lambda) = \mathcal{G}(2) = 31 - 20 = 11$. Comme le chemin $(1,3,5,6)$ est de temps $10 \leqslant T$ et de coût $11 = \mathcal{G}(2)$, nous venons de résoudre non seulement le problème relâché lagrangien, qui consiste à maximiser $\mathcal{G}(\lambda)$, mais également le problème initial (5.21).

Remarque 5.10 Le fait que le relâché lagrangien fournisse une solution du problème initial est exceptionnel. En général, il y a un saut de dualité, mais les x réalisant le min dans la fonction duale (5.15), évaluée en un point de maximum λ de $\mathcal{G}$, peuvent souvent être modifiés sans trop augmenter le coût pour arriver à une solution (sous-optimale) du problème initial. On parle alors d'*heuristiques lagrangiennes.* •

Exercice 5.11 Calculer la fonction duale $\mathcal{G}(\lambda)$ pour le problème (5.20), et retrouver ainsi la valeur de $\max_{\lambda\in\mathbb{R}_+} \mathcal{G}(\lambda)$.

Exercice 5.12 Proposer une relaxation lagrangienne fournissant un majorant pour le problème général du sac à dos (exemple 1.12), et l'appliquer au cas particulier suivant :

$$\max_{\substack{x\,\in\,\{0,1\}^3 \\ 2x_1+3x_2+5x_3\,\leqslant\,6}} 8x_1 + 2x_2 + 9x_3 \ . \tag{5.22}$$

Relaxation lagrangienne pour le problème du voyageur de commerce

Revisitons le problème du voyageur de commerce dans un graphe non-orienté complet $\mathcal{G} = (\mathcal{V},\mathcal{E})$, muni d'une fonction coût $c : \mathcal{E} \to \mathbb{R}_+, \{i,j\} \mapsto c_{ij}$. On peut écrire le problème du voyageur de commerce sous forme de programme linéaire en nombre entiers, équivalent à (VC),

$$\begin{array}{ll} \min & \sum_{\{i,j\}\in\mathcal{E}} c_{ij}x_{ij} \text{ sous les contraintes} \\ & x \in \{0,1\}^{\mathcal{E}}, \\ & \sum_{k\in\mathcal{V},\{k,j\}\in\mathcal{E}} x_{kj} = 2, \text{ pour tout } j \in \mathcal{V}, \\ & x \text{ représente un graphe connexe à } |\mathcal{V}| \text{ arêtes} \\ & \text{couvrant tous les sommets.} \end{array}$$

Distinguons un sommet particulier $1 \in \mathcal{V}$. Si, pour tout $j \in \mathcal{V}\setminus\{1\}$, on relâche les contraintes $\sum_{k\in\mathcal{V},\{k,j\}\in\mathcal{E}} x_{kj} = 2$, on obtient la fonction duale $\mathcal{G} : \mathbb{R}^{\mathcal{V}\setminus\{1\}} \to \mathbb{R}$

$$\begin{array}{ll} \mathcal{G}(\lambda) = & \min \sum_{\{i,j\}\in\mathcal{E}} c_{ij}x_{ij} + \sum_{j\in\mathcal{V}\setminus\{1\}} \lambda_j(\sum_{k\in\mathcal{V},\ \{k,j\}\in\mathcal{E}} x_{kj} - 2) \\ & \text{sous les contraintes} \\ & \quad x \in \{0,1\}^{\mathcal{E}} \\ & \sum_{k\in\mathcal{V},\{k,1\}\in\mathcal{E}} x_{k1} = 2 \\ & \quad x \text{ représente un graphe connexe à } |\mathcal{V}| \text{ arêtes} \\ & \quad \text{couvrant tous les sommets.} \end{array}$$

Au terme $-\sum_{j\in\mathcal{V}\setminus\{1\}} 2\lambda_j$ près, on reconnaît dans $\mathcal{G}(\lambda)$ le coût minimum d'un 1-arbre pour le graphe $\mathcal{G} = (\mathcal{V},\mathcal{E})$, dont chaque arête $\{i,j\}$ est munie du coût $c_{ij} + \lambda_i + \lambda_j$, pour $i,j \in \mathcal{V}\setminus\{1\}$, et dont chaque arête

$\{i,1\}$, pour $i \in \mathcal{V} \setminus \{1\}$, a pour coût $c_{1i} + \lambda_i$. L'algorithme de Kruskal de la Section A.1 permet justement de calculer un 1-arbre de coût minimum en temps $O(|\mathcal{E}| \log |\mathcal{E}|)$, ce qui nous permet de mettre en œuvre efficacement l'algorithme de sur-gradient (5.18) pour calculer la borne suivante : $\sup_{\lambda \in \mathbb{R}^{\mathcal{V}\setminus\{1\}}} \mathcal{G}(\lambda)$. Cette borne remarquable pour le problème du voyageur de commerce, est due à Held et Karp. On peut montrer (cela n'est pas immédiat) que la borne du 1-arbre vue dans la sous-section 5.1 coïncide avec la valeur $\mathcal{G}(0)$ obtenue en ne faisant pas payer la transgression des contraintes $\sum_{k\in\mathcal{V},\{k,j\}\in\mathcal{E}} x_{kj} = 2$, pour tout $j \in \mathcal{V} \setminus \{1\}$.

Exercice 5.13 La version "orientée" du problème du voyageur de commerce consiste à considérer un graphe orienté $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, muni d'une fonction coût $c : \mathcal{A} \to \mathbb{R}_+$, et à chercher un circuit orienté passant une et une seule fois par chaque nœud. On demande de proposer une relaxation lagrangienne, en faisant en sorte que pour chaque vecteur λ de multiplicateurs de Lagrange, le calcul de la fonction duale $\mathcal{G}(\lambda)$ revienne à résoudre un problème d'affectation. Quel est l'inconvénient de cette relaxation, par comparaison à la borne de Held et Karp, dans le cas particulier du problème de voyageur de commerce non-orienté ?

Théorème du bidual : le relaxé lagrangien améliore le relaxé continu

On peut se demander comment une relaxation lagrangienne se compare à une relaxation continue. Un résultat dû à Geoffrion, qui identifie la valeur du relaxé lagrangien à la valeur d'un bidual, va nous permettre de montrer que le relaxé lagrangien fait au moins aussi bien que le relaxé continu, tout en caractérisant le cas d'égalité.

Nous partons du problème combinatoire

$$\operatorname*{Min}_x c \cdot x;\ Ax \leqslant b,\ x \in X\ . \tag{5.23}$$

avec X un sous-ensemble *fini* non-vide de $\mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, et $c \in \mathbb{R}^n$. Nous formons le lagrangien

$$L(x,\lambda) = c \cdot x + \lambda \cdot (Ax - b)\ ,$$

la *fonction duale*

$$\mathcal{G}(\lambda) = \min_{x\in X} L(x,\lambda)$$

et le *relaxé lagrangien*

$$\operatorname*{Max}_\lambda \mathcal{G}(\lambda);\ \lambda \in \mathbb{R}^m_+\ .$$

Théorème 5.14 *la valeur du relaxé lagrangien coïncide avec la valeur du problème*

$$\operatorname*{Min}_x c \cdot x;\ Ax \leqslant b,\ x \in \mathrm{conv}(X) \tag{5.24}$$

Démonstration. Le problème relaxé lagrangien peut s'exprimer comme un programme linéaire moyennant l'introduction d'une variable scalaire auxiliaire t :

$$\operatorname*{Max}_{t\in\mathbb{R},\lambda\in\mathbb{R}^m_+} t; \qquad t \leqslant L(x,\lambda),\ \text{ pour tout }\ x \in X\ ,$$

soit

$$\operatorname*{Max}_{t\in\mathbb{R},\lambda\in\mathbb{R}^m_+} t; \qquad t \leqslant c \cdot x + \lambda \cdot (b - Ax),\ \text{ pour tout }\ x \in X\ . \tag{5.25}$$

Notons α_x le multiplicateur associé à la contrainte $t \leqslant c\cdot x+\lambda\cdot(b-Ax)$, et notons β le multiplicateur associé à la contrainte $\lambda \geqslant 0$. Le problème (5.25) s'écrit

$$\sup_{\substack{t\ \in\ \mathbb{R},\\ \lambda\ \in\ \mathbb{R}^m}}\ \inf_{\substack{\alpha\ \in\ \mathbb{R}^X_+,\\ \beta\ \in\ \mathbb{R}^m_+}}\ t + \sum_{x\in X} \alpha_x(c\cdot x + \lambda\cdot(Ax-b) - t) + \lambda\cdot\beta$$

Le problème dual de (5.25) est donc donné par :

$$\begin{aligned}
&\inf_{\substack{\alpha \in \mathbb{R}_+^X,\\ \beta \in \mathbb{R}_+^m}} \sup_{\substack{t \in \mathbb{R},\\ \lambda \in \mathbb{R}^m}} t + \sum_{x \in X} \alpha_x (c \cdot x + \lambda \cdot (Ax - b) - t) + \lambda \cdot \beta \\
&\quad = \inf_{\substack{\alpha \in \mathbb{R}_+^X,\\ \beta \in \mathbb{R}_+^m}} \sup_{\substack{t \in \mathbb{R},\\ \lambda \in \mathbb{R}^m}} \lambda \cdot (A(\sum_{x \in X} \alpha_x x) - \sum_{x \in X} \alpha_x b + \beta) \\
&\quad + c \cdot (\sum_{x \in X} \alpha_x x) + t(1 - \sum_{x \in X} \alpha_x) \\
&\quad = \inf_{\substack{\alpha \in \mathbb{R}_+^X \\ \sum_{x \in X} \alpha_x = 1,\ A(\sum_{x \in X} \alpha_x x) \leqslant b}} c \cdot (\sum_{x \in X} \alpha_x x) = \inf_{z \in \mathrm{conv}(X); Az \leqslant b} c \cdot z \ ,
\end{aligned}$$

ce qui donne bien (5.24). Le théorème de dualité forte en programmation linéaire montre que la valeur d'un programme coïncide avec la valeur de son dual pourvu que ce programme ait au moins un point admissible, ce qui est bien le cas de (5.25) (les contraintes sont satisfaites quel que soit λ quand $t \to -\infty$). ■

Remarque 5.15 Puisque le relaxé lagrangien peut être vu comme un dual du problème initial, la preuve qui précède permet d'interpréter le problème (5.24) comme un bidual.

On s'intéresse maintenant au cas où X est de la forme $X = \{x \in \mathbb{Z}^n \mid Dx \leqslant g\}$ avec $D \in \mathbb{R}^{p \times n}$ et $g \in \mathbb{R}^p$, X étant en outre toujours supposé fini. Soient

$$P = \{x \in \mathbb{R}^n \mid Ax \leqslant b\}, \qquad Q = \{x \in \mathbb{R}^n \mid Dx \leqslant g\}$$

et notons, pour tout polyèdre $R \subset \mathbb{R}^n$,

$$R_I = \mathrm{co}(R \cap \mathbb{Z}^n)$$

la *clôture entière de* R, notions que nous reverrons au chapitre 7. Observons que les valeurs du problème initial (5.23), de son relaxé lagrangien, et de son relaxé continu, sont données respectivement par

$$\min_{x \in (P \cap Q)_I} c \cdot x \ , \tag{5.26a}$$

$$\min_{x \in P \cap Q_I} c \cdot x \ , \tag{5.26b}$$

$$\min_{x \in P \cap Q} c \cdot x \ , \tag{5.26c}$$

avec

$$(P \cap Q)_I \subset P \cap Q_I \subset P \cap Q \ .$$

Ceci montre que la valeur du relaxé lagrangien est toujours plus précise (supérieure) à celle du relaxé continu, avec égalité si Q est borné et si les points extrêmes de Q sont entiers (dans ce cas $Q = Q_I$).

5.3 Notes

On pourra se reporter au livre de Cook, Cunningham, Pulleybank et A. Schrijver [CCPS98] ainsi que celui de Korte et Vygen [KV02] pour une introduction à l'approche "polyédrale" du problème de voyageur de commerce. Le livre d'Ahuja, Magnanti, et Orlin [AMO93], contient un exposé complet de la méthode de relaxation lagrangienne, illustré par des exemples pratiques, souvent empruntés aux réseaux ; on pourra aussi consulter l'article introductif [Lem03].

Chapitre 6

Algorithme du simplexe

Ce chapitre présente la méthode du simplexe, issue des travaux de Dantzig dans les années 50 et considérablement améliorée depuis pour l'adapter aux problèmes de très grande taille. Avec les méthodes de points intérieurs présentées dans le chapitre 10, c'est l'une des deux classes d'algorithmes utilisés de nos jours. Un avantage déterminant de la variante duale du simplexe, présentée en fin de chapitre, est la possibilité de résoudre rapidement une suite de problèmes obtenus par ajout successifs de contraintes. Cela en fait l'algorithme de choix pour la résolution de programmes linéaires avec contraintes d'intégrité pour lesquels on résout une suite de programmes linéaires "continus", obtenus par branchement ou ajout de coupes ; voir les chapitres 5 et 7.

Le chapitre présente d'abord la classe plus générale des méthodes de gradient réduit, puis expose la méthode du simplexe proprement dite.

6.1 Méthodes de gradient réduit

6.1.1 Optimisation sous contraintes linéaires

Considérons des problèmes de la forme

$$\operatorname*{Min}_x f(x)\ ;\ Ax = b,\ x \geqslant 0, \tag{P}$$

avec $f : \mathbb{R}^n \to \mathbb{R}$ de classe C^1, et A matrice $p \times n$ de rang p. On dit que $\bar{x}$ est solution locale du problème (P) s'il existe $\varepsilon > 0$ tel que tout $x \in F(P) \cap B(\bar{x}, \varepsilon)$ vérifie $f(x) \geqslant f(\bar{x})$. On note $\nabla f(x)$ le vecteur colonne formé des dérivées partielles de f en x. C'est le transposé du vecteur horizontal $Df(x)$.

Lemme 6.1 *Toute solution locale $\bar{x}$ du problème (P) vérifie la condition nécessaire d'optimalité du premier ordre suivante : Il existe $\lambda \in \mathbb{R}^p$ tel que*

$$A\bar{x} = b; \quad \bar{x} \geqslant 0; \quad \nabla f(\bar{x}) + A^\top \lambda \geqslant 0; \quad (\nabla f(\bar{x}) + A^\top \lambda) \cdot \bar{x} = 0. \tag{6.1}$$

Démonstration. a) Posons $c := \nabla f(\bar{x})$, et montrons que $\bar{x}$ est solution du programme linéaire

$$\operatorname*{Min}_x c \cdot x; \quad Ax = b, \quad x \geqslant 0. \tag{6.2}$$

Si ce n'est pas le cas, il existe $y \in F(P)$ tel que $Df(\bar{x})(y - \bar{x}) < 0$. Alors pour $t \in]0, 1[$, $x_t := \bar{x} + t(y - \bar{x})$ est tel que $x_t \in F(P)$ et, si t est assez petit, $f(x_t) = f(\bar{x}) + tDf(\bar{x})(y - \bar{x}) + o(t) < f(\bar{x})$ ce qui contredit l'optimalité locale de $\bar{x}$.
b) Le problème dual de (6.2) est

$$\operatorname*{Max}_{\lambda \in \mathbb{R}_n, s \in \mathbb{R}^+_n} -b \cdot \lambda; \quad \nabla f(\bar{x}) + A^\top \lambda = s. \tag{6.3}$$

D'après le théorème 2.49 de dualité forte pour la programmation linéaire, le dual a même valeur finie que le primal, donc (puisque c'est un programme linéaire) a une solution (λ, s) tel que $x \cdot s = 0$ (en raison du point (ii) du théorème cité), ce qui implique (6.1). ■

6.1.2 Algorithme de gradient réduit : principe

L'idée de l'algorithme du gradient réduit est d'éliminer p composantes de x grâce aux contraintes linéaires, afin de se ramener à un problème à $n - p$ variables.

Soit (B, N) une partition de $\{1, \cdots, n\}$. On note $|B|$ le cardinal de B. Partitionnant $x \in \mathbb{R}^n$ et A suivant leurs indices et colonnes, on obtient

$$x = (x_B, x_N); \quad A = (A_B, A_N); \quad Ax = A_B x_B + A_N x_N.$$

Définition 6.2 Si (B, N) est une partition de $\{1, \cdots, n\}$, on dit que B est une *base* si $|B| = p$, et si A_B est inversible (pour faciliter les raisonnements on pourra représenter B par les p premières composantes). On appelle x_N la *variable hors base*, et x_B la *variable de base*. Cette dernière est déterminée par la variable hors base :

$$x_B(x_N) := A_B^{-1}(b - A_N x_N). \tag{6.4}$$

Le critère peut donc s'exprimer en fonction de x_N. On appelle *critère réduit* la fonction

$$F(x_N) := f(x_B(x_N), x_N), \tag{6.5}$$

et *gradient réduit* le gradient de celle-ci. Le gradient réduit est usuellement obtenu de la manière suivante. Notons $\nabla_B f(x)$ et $\nabla_N f(x)$ les gradients partiels de f par rapport à x_B et x_N. On calcule d'abord l'*estimation du multiplicateur* λ, solution de

$$A_B^\top \lambda = -\nabla_B f(x). \tag{6.6}$$

Effectuant un développement de Taylor de F au premier ordre, il vient

$$\nabla F(x_N) = \nabla_N f(x) + A_N^\top \lambda. \tag{6.7}$$

Soit $\hat{x} \in F(P)$ tel que $\hat{x}_B > 0$ (ce qui signifie que toutes ses coordonnées sont strictement positives) ; on dit alors que la base est *non dégénérée*. Au voisinage de $\hat{x}$, on peut ignorer les contraintes $x_B \geqslant 0$ qui ne sont pas actives, ce qui conduit à l'expression du *problème local* :

$$\operatorname*{Min}_{x_N} F(x_N); \quad x_N \geqslant 0. \tag{P_{loc}}$$

La condition nécessaire d'optimalité du premier ordre (conséquence du lemme 6.1 dans le cas sans contraintes linéaires) du problème local est

$$x_N \geqslant 0; \quad \nabla F(x_N) \geqslant 0; \quad \nabla F(x_N) \cdot x_N = 0. \tag{6.8}$$

Lemme 6.3 *Soit $x \in F(P)$, tel que $x_B > 0$. Alors les conditions nécessaires d'optimalité du premier ordre* (6.1) *et* (6.8) *sont équivalentes.*

Démonstration. De (6.6) et (6.7) on déduit que

$$\nabla f(x) + A^\top \lambda = \begin{pmatrix} 0 \\ \nabla F(x_N) \end{pmatrix}, \tag{6.9}$$

d'où découle l'équivalence de (6.1) et (6.8). ■

Définition 6.4 On appelle direction de descente en $x \in F(P)$ une direction $d \in \mathbb{R}^n$ telle que $Df(x)^\top d < 0$, et $x + \varepsilon d \in F(P)$, pour un certain $\varepsilon > 0$.

L'absence de directions de descente en $x \in F(P)$ équivaut à la condition nécessaire d'optimalité du premier ordre (voir la démonstration du lemme 6.1). Si $x \in F(P)$ et la partition (B, N) de $\{1, \ldots, n\}$ sont tels que A_B est inversible et $x_B > 0$, alors d est une direction de descente ssi les deux conditions suivantes sont vérifiées :

$$\begin{cases} \text{(i)} & \nabla F(x_N)^\top d_N < 0; \quad \exists\, \varepsilon' > 0; \;\; x_N + \varepsilon' d_N \geqslant 0. \\ \text{(ii)} & d_B = -A_B^{-1} A_N d_N. \end{cases} \tag{6.10}$$

On dira que d_N est une *direction de descente réduite* en x_N (en un point $x \in F(P)$ tel que $x_B > 0$) si elle vérifie (6.10) (i).

Algorithme 6.5 (Algorithme de principe du gradient réduit)

1. Initialisation : Choisir $x^0 \in F(P)$; $k \leftarrow 0$.
2. Choisir une partition (B^k, N^k) de $\{1, \ldots, n\}$ telle que A_{B^k} est inversible et $x^k_{B^k} > 0$.
   Poser $B \leftarrow B^k$, $N \leftarrow N^k$.
3. Calculer l'estimation du multiplicateur λ^k, solution de $A_B^\top \lambda^k = -\nabla_B f(x^k)$.
   Calculer le gradient réduit $\nabla F(x^k_N) = \nabla_N f(x^k) + A_N^\top \lambda^k$.
4. Si (6.8) est satisfaite, arrêt.
5. Choisir d_N, direction de descente réduite en x^k_N.
   Calculer d_B, solution de $A_B d_B = -A_N d_N$. Poser $d^k \leftarrow (d_B, d_N)$.
6. Recherche linéaire : choisir $\rho_k > 0$ tel que $f(x^k + \rho_k d^k) < f(x^k)$.
7. $x^{k+1} \leftarrow x^k + \rho_k d^k$; $\quad k \leftarrow k + 1$. Aller en 2.

Pour passer de l'algorithme de principe à une implémentation réelle, il faut préciser (i) le choix de la direction de descente, (ii) la règle de recherche linéaire, (iii) la règle de mise à jour de la base. Comme notre motivation est avant tout la présentation de l'algorithme du simplexe pour la programmation linéaire, nous renvoyons pour les deux premiers points aux notes de fin de chapitre et nous exposons le troisième point dans la suite.

6.1.3 Mise à jour de la base : règles de pivotage

Soit B la base lors de l'itération k de l'algorithme de gradient réduit ; on a donc $x^k_B > 0$. La recherche linéaire détermine un nouveau point x^{k+1}. Si $x^{k+1}_B > 0$, on peut garder la même partition (B, N) à l'itération $k + 1$. Sinon, il existe $\ell \in B$ tel que $x^k_\ell > 0$ et $x^{k+1}_\ell = 0$. Il faut alors mettre à jour la base, en remplaçant ℓ par une variable hors base j telle que $x^{k+1}_j > 0$.

Définition 6.6 On dira que le point x^{k+1} (ou plus rigoureusement le couple (x^{k+1}, B)) est *non dégénéré* si (i) x^{k+1}_B a ℓ pour seule composante nulle, et (ii) il existe $j \in N$ tel que $x^{k+1}_j > 0$, et $\hat{B} := (B \cup \{j\}) \backslash \{\ell\}$ est une base. On appelle ℓ la *variable sortante* et j la *variable entrante*.

Remarque 6.7 La non dégénérescence exprime la possibilité d'obtenir une nouvelle base $\hat{B}$, obtenue en changeant un seul indice de l'ancienne base, et telle que $x_{\hat{B}} > 0$. •

En général il y a plusieurs candidats pour la *variable entrante* j. Le choix est un compromis entre (i) la distance de x^{k+1}_j à 0, qui doit être la plus grande possible pour éviter que l'indice entrant de la base n'en ressorte trop rapidement, et (ii) l'inversibilité de la nouvelle matrice de base (plus précisément la possibilité de calculer les directions de déplacement de manière suffisamment précise). Étudions ce second point grâce à la formule de Woodbury-Shermann-Morrisson établie dans le lemme ci-dessous :

Lemme 6.8 *Soient $A \in \mathcal{M}^n$ et u, v dans $\mathbb{R}^n$ tels que A est inversible. Alors $A + uv^\top$ est inversible ssi $1 + \langle v, A^{-1}u\rangle \neq 0$, et on a*

$$(A + uv^\top)^{-1} = A^{-1} - \frac{A^{-1}uv^\top A^{-1}}{1 + \langle v, A^{-1}u\rangle}. \tag{6.11}$$

Démonstration. Si $u = 0$ la conclusion est triviale ; sinon, puisque $(A+uv^\top)A^{-1}u = (1+\langle v, A^{-1}u\rangle)u$, si $1 + \langle v, A^{-1}u\rangle = 0$, il est clair que $A + uv^\top$ n'est pas inversible. Enfin si $1 + \langle v, A^{-1}u\rangle \neq 0$, on vérifie (6.11) en effectuant le produit du membre de droite par $A + uv^\top$. ■

Ainsi l'inverse d'une mise à jour de rang 1 est une mise à jour de rang 1 de l'inverse.

Lemme 6.9 *Soient ℓ et j les indices des variables sortante et entrante. Notons e_ℓ le vecteur de base d'indice ℓ dans $\mathbb{R}^p$, et A_i la $i$ème colonne de A. Alors la nouvelle matrice de base est inversible ssi*

$$e_\ell^\top A_B^{-1} A_j \neq 0. \tag{6.12}$$

Démonstration. On peut supposer que $B = \{1, \ldots, p\}$ et que la variable entrante est rangée en position ℓ dans la base. Alors la nouvelle matrice de base $\hat{A}_B$ ne diffère de l'ancienne que par la colonne ℓ :

$$\hat{A}_B = A_B + (A_j - A_\ell)e_\ell^\top.$$

En particulier, les deux matrices diffèrent d'une matrice de rang 1. La formule de Woodbury-Shermann-Morrisson (6.11) indique que $\hat{A}_B$ est inversible ssi

$$0 \neq 1 + \langle e_\ell, A_B^{-1}(A_j - A_\ell)\rangle.$$

Mais $A_B^{-1}A_\ell = e_\ell$, d'où le résultat. ■

La quantité $e_\ell^\top A_B^{-1} A_j$ représente la sensibilité de la variable de base ℓ à la variation de la variable hors-base j. On appelle $e_\ell^\top A_B^{-1} A_N$ le *vecteur des pivots*. On choisit en général la variable hors base la plus éloignée de la borne, sous contrainte de pivot pas trop petit. La règle classique due à Markowitz : pour un certain $\alpha \in]0, 1[$ (un choix classique est 1/10), on choisit la variable entrante dans l'ensemble

$$\operatorname*{argmax}_{j \in N} \left\{ x_j ; \;\; |e_\ell^\top A_B^{-1} A_j| \geqslant \alpha \max_{i \in N} |e_\ell^\top A_B^{-1} A_i| \right\}. \tag{6.13}$$

Remarque 6.10 On peut implémenter l'algorithme en explicitant l'inverse de la première matrice de base et en mettant à jour l'inverse grâce à la formule de Woodbury-Shermann-Morrisson (6.11). La mise à jour de l'inverse nécessite $O(p^2)$ opérations, alors que la résolution d'un système linéaire de taille p nécessite en général $O(p^3)$ opérations. •

Remarque 6.11 Concernant l'initialisation de l'algorithme (point et base initiaux), et en particulier la recherche d'un point admissible, on se reportera à la remarque 6.15. •

6.2 Programmation linéaire : algorithme du simplexe

6.2.1 Principe

Présentons maintenant l'algorithme du simplexe pour la résolution de problèmes d'optimisation linéaire. Il s'agit d'un cas très particulier d'algorithme de gradient réduit, puisque les variables hors base sont nulles (au début de chaque itération). Considérons le problème linéaire

$$\operatorname*{Min}_x c \cdot x; \quad Ax = b; \;\; x \geqslant 0. \tag{LP}$$

On suppose A de taille $p \times n$, avec $p \leqslant n$, de rang p (les lignes sont linéairement indépendantes). On rappelle que si $I \subset \{1, \cdots, n\}$, on note par A_J l'ensemble des colonnes de A d'indice dans I, et de même pour x_I (ensemble des composantes de x d'indice dans I). Une base B est une partie de $\{1, \cdots, n\}$ de cardinal p, telle que A_B est inversible. On pose $N := \{1, \cdots, n\} \setminus B$. Introduisons la notion clé de l'algorithme.

Définition 6.12 (i) Soit (B, N) une partition de $\{1, \cdots, n\}$ telle que B est une base ($|B| = p$ et A_B est inversible). On appelle *point de base* associé l'unique vecteur $x \in \mathbb{R}^n$ de composantes $x_B = A_B^{-1}b$, $x_N = 0$. Notons que $Ax = b$; on dit aussi que x est *basique*.
(ii) On dit que la base B est admissible (ou réalisable) si $A_B^{-1}b \geqslant 0$ (c'est à dire si le point de base associé est réalisable).
(iii) On dit que la base B est non dégénérée si $A_B^{-1}b > 0$ (les variables de base du point de base sont strictement positives).

Lemme 6.13 *Tout point extrême de $F(LP)$ est basique.*

Démonstration. Soit $\bar{x}$ un point extrême de $F(LP)$. Notons $J := \{1 \leqslant j \leqslant n;\ \bar{x}_j > 0\}$ et soit E l'espace vectoriel $\mathbb{R}^{|J|}$ d'éléments indicés dans J. Alors l'application $E \to \mathbb{R}^p$, $d \mapsto A_J d = \sum_{j \in J} d_j A_j$ est injective. Sinon, soit $\bar{d}$ un élément non nul du noyau, étendu à $\mathbb{R}^n$ en prenant $\bar{d}_j = 0$ si $j \notin J$. Alors $A\bar{d} = 0$, et donc $\bar{x} \pm \varepsilon\bar{d} \in F(LP)$ pour $\varepsilon > 0$ petit, ce qui contredit l'extrémalité de $\bar{x}$.

Comme A est de rang p, on peut compléter J par $p - |J|$ indices pour former un ensemble B de cardinal p, tel que A_B est inversible ; alors $\bar{x}$ est le point de base associé à B. ■

Corollaire 6.14 *L'ensemble $S(LP)$ contient, s'il n'est pas vide, un point basique.*

Démonstration. Si $S(LP)$ n'est pas vide, puisque $F(LP)$ ne contient pas de droite, le corollaire 2.26 assure que $S(LP)$ contient au moins un point extrême $\bar{x}$. On conclut avec le lemme précédent. ■

Remarque 6.15 En général, on ne connaît pas de point de base du problème qu'on cherche à résoudre. En fait, on ne sait pas si ce problème est réalisable ! On peut cependant ramener le calcul d'un point réalisable à la résolution d'un problème auxiliaire dont on connaît un point de base. Par exemple, considérons le problème

$$\operatorname*{Min}_{x,z} \sum_{i=1}^{p} z_i;\ Ax + wz = b;\ x \geqslant 0;\ z \geqslant 0, \tag{$*$}$$

où $w \in \mathbb{R}^p$, $z \in \mathbb{R}^p$, et le produit wz est pris composante par composante. Il est clair que $\text{val}(*) \geqslant 0$, et que $\text{val}(*) = 0$ ssi (LP) est réalisable. Si le choix du point initial est

$$\begin{cases} x^0 \in \mathbb{R}^n_+; & I := \{1 \leqslant i \leqslant n; (b - Ax^0)_i \neq 0\}, \\ w_i := (b - Ax^0)_i; & \text{si } i \in I,\ 1 \text{ sinon}, \\ z_i^0 := 1 & \text{si } i \in I,\ 0 \text{ sinon}, \end{cases} \tag{6.14}$$

alors le point (x^0, z^0) est réalisable pour $(*)$. Si de plus on choisit $x^0 = 0$, le point (x^0, z^0) est de base, les variables de base correspondant à z. Notons que ce point est dégénéré ssi $I \neq \{1, \ldots, n\}$. •

Algorithme 6.16 (Algorithme de principe du simplexe)

1. Choisir x^0, point de base admissible. $k := 0$.
2. Si $x^k \in S(LP)$, arrêt.
3. Si on peut vérifier que $\text{val}(LP) = -\infty$, arrêt.
4. Calculer x^{k+1}, point admissible associé à une autre base tel que $c \cdot x^{k+1} \leqslant c \cdot x^k$, avec inégalité stricte si x^k n'est pas dégénéré ($x_B^k > 0$).
5. $k \leftarrow k + 1$, aller en 1.

Bien entendu, on va préciser plus loin comment réaliser les étapes précédentes. Établissons d'abord la convergence de cet algorithme de principe.

Théorème 6.17 *L'algorithme de principe ci-dessus ne comporte qu'un nombre fini d'itérations associées à des bases non dégénérées. En particulier, après un nombre fini d'itérations, soit il s'arrête, soit il ne passe plus que par des bases dégénérées.*

Démonstration. Il existe un nombre fini de bases, et à chacune est associée un unique point de base. La décroissance du critère, stricte lors du passage par une base non dégénérée, assure donc qu'on ne peut passer deux fois par par la même base non dégénérée, d'où la conclusion. ■

Voyons maintenant comment réaliser le passage d'un point de base non dégénéré à un autre point de base, de critère strictement inférieur. Soient B une base non dégénérée, N son complément dans $\{1,\dots,n\}$, et $\tilde{x}$ le point de base associé. De la relation $Ax = b$ on peut tirer x_B fonction de x_N :

$$x_B(x_N) := A_B^{-1}(b - A_N x_N). \tag{6.15}$$

Ceci permet d'exprimer le critère fonction de x_N seul :

$$c \cdot x = c_B \cdot A_B^{-1}(b - A_N x_N) + c_N \cdot x_N = (c_N - A_N^\top A_B^{-\top} c_B) \cdot x_N. \tag{6.16}$$

On appelle *coût réduit* la quantité $g := c_N - A_N^\top A_B^{-\top} c_B$. Décomposant son calcul en faisant apparaître l'*estimation du multiplicateur* λ, il vient

$$A_B^\top \lambda = -c_B; \quad g = c_N + A_N^\top \lambda. \tag{6.17}$$

Joignant les deux égalités ci-dessus, on observe que

$$c + A^\top \lambda = s \quad \text{où} \quad s = \begin{pmatrix} 0 \\ g \end{pmatrix}; \;\; x \cdot s = 0. \tag{6.18}$$

Le multiplicateur s satisfait la condition de complémentarité avec le point basique $\tilde{x}$, puisque $s_B = 0$ et $x_n = 0$. Si $g \geqslant 0$, (x, λ, s) est solution du système d'optimalité, et $\tilde{x}$ est solution de (LP). On obtient ainsi un test d'arrêt de l'algorithme.

Si au contraire il existe $j \in N$ tel que $g_j < 0$, soit e_j le $j^{\text{ième}}$ vecteur de base de $\mathbb{R}^n$, et soit la direction $d \in \mathbb{R}^n$ définie par

$$d_i = 0, i \in N \setminus \{j\}; \quad d_j = 1; \;\; d_B = -A_B^{-1} A_N d_N = -A_B^{-1} A_j. \tag{6.19}$$

Posons, pour $\rho > 0$, $x(\rho) := \tilde{x} + \rho d$. Il vient $c \cdot d = c_B \cdot d_B + c_N \cdot d_N = (c_N - A_N^\top A_B^{-\top} c_B) \cdot d_N$, et donc

$$c \cdot x(\rho) := c \cdot \tilde{x} + \rho c \cdot d = c \cdot \tilde{x} + \rho g_j < c \cdot x. \tag{6.20}$$

Comme $Ad = 0$ et $d_N \geqslant 0$, pour $\rho > 0$ petit, $x(\rho)$ est réalisable. Considérons le plus grand pas réalisable $\bar{\rho}$. S'il vaut $+\infty$, puisque $c \cdot d = g_j < 0$, on déduit que $\text{val}(LP) = -\infty$. S'il est fini, notons $\hat{x} := \tilde{x} + \bar{\rho} d$ le point ainsi obtenu, qui a nécessairement un indice ℓ dans B tel que $\hat{x}_\ell = 0$. Notons $\hat{B} := (B \cup \{j\}) \setminus \{\ell\}$. Alors $A_{\hat{B}}$ est inversible d'après le lemme 6.9. Donc $\hat{x}$ est le point de base associé à $\hat{B}$.

Récapitulons l'ensemble des opérations :

Algorithme 6.18 [Réalisation d'une itération de l'algorithme de principe]
On se donne une base admissible non dégénérée B et le point de base associé x.

1. Calcul de l'estimation du multiplicateur, par résolution de $A_B^\top \lambda = -c_B$.
2. Calcul du coût réduit $g \leftarrow c_N + A_N^\top \lambda$.
3. Si $g \geqslant 0$, arrêt : x est solution de (LP).
4. Direction de déplacement : soit $j \in N$ tel que $g_j < 0$; $d_N \leftarrow e_j$; $d_B \leftarrow -A_B^{-1} A_j$.
5. Si $d \geqslant 0$, arrêt : $\text{val}(LP) = -\infty$.
6. Calcul du pas : $\ell \in \text{argmin}\{x_i/|d_i|;\; d_i < 0\}$, et $\rho \leftarrow x_\ell/|d_\ell|$.

7. Nouveau point $x \leftarrow x + \rho d$. $B \leftarrow (B \cup \{j\})\backslash\{\ell\}$; $N \leftarrow (N \cup \{\ell\})\backslash\{j\}$.

Remarque 6.19 (i) Le choix standard de la variable entrante est $j = \text{argmin}\{g_j\}$. Il faut noter que ce choix dépend de la mise à l'échelle des variables.
(ii) Nous avons vu que la nouvelle matrice de base est inversible d'après le lemme 6.9. Toutefois, en pratique les calculs se font usuellement avec des nombres flottants. Il peut alors être nécessaire de procéder à une mise à l'échelle des variables et contraintes (par exemple en normalisant les lignes, puis les colonnes de la matrice de base) pour assurer la précision de la résolution des systèmes linéaires. Les variables de base presque nulles peuvent aussi causer des difficultés. •

6.2.2 Règle de Bland

Examinons maintenant un cas particulier de l'algorithme du simplexe qui assure la convergence même en présence de bases dégénérées (nullité de certaines variables de base). Le problème apparaît si, à l'itération $k-1$, un nombre $r \geqslant 2$ de variables de bases s'annulent simultanément. A l'itération k, la matrice de base a $r-1$ variables nulles. On peut toujours calculer un gradient réduit, mais si certaines variables de base nulles ont une direction de déplacement strictement négative, le pas ρ_k sera nul. Toutefois l'itération a permis un changement de base. La question est de savoir si l'algorithme va effectuer une infinité de telles itérations ou au contraire obtenir après un nombre fini d'itérations un pas positif. Nous allons assurer que le second cas se produit par le choix de la *règle de Bland*, qui précise l'algorithme 6.18 comme suit :

Règle de Bland

(i) Faire rentrer dans la base la variable hors base d'indice minimal, parmi celles de gradient réduit strictement négatif. Autrement dit, la variable entrante dans l'étape 4 de l'algorithme 6.18 est

$$j := \min\{i \in N;\ g_j < 0\}. \tag{6.21}$$

(ii) Dans le cas d'une base dégénérée (plusieurs variables de base nulles), la variable sortante est celle d'indice minimal parmi les variables de base nulles et de déplacement strictement négatif. Autrement dit, on remplace l'étape 7 de l'algorithme 6.18 par

7'. Nouveau point $x \leftarrow x + \rho d$.

$$E := \{i \in B;\ x_i = 0;\ d_i < 0\}; \quad \hat{\ell} := \min\{i \in E\}. \tag{6.22}$$

$B := (B \cup \{j\})\backslash\{\hat{\ell}\}$; $N := (N \cup \{\hat{\ell}\})\backslash\{j\}$.

Théorème 6.20 *La méthode du simplexe utilisant la règle de Bland détermine si* (LP) *a une valeur finie, et dans ce cas obtient une solution de* (LP).

Démonstration. Il suffit de montrer l'impossibilité d'une succession infinie de pas nuls. Supposons au contraire que ceci arrive : nous allons obtenir une contradiction.
a) L'algorithme, après un nombre fini d'itérations, aura un *comportement cyclique* : il existe deux entiers k_0 et $K > 0$ tel que pour tout $k \geqslant k_0$ les itérations k et $k+K$ sont identiques (mêmes base, variable entrante, et variable sortante). En effet, le nombre de bases étant fini, il existe deux itérations k_1 et $k_2 = k_1 + K$, avec K entier strictement positifs, de même base. La règle de choix de la variable entrante et sortante de la base étant fonction de la seule base, les opérations effectuées sont identiques pendant ces deux itérations. Les itérations $k_1 + 1$ et $k_2 + 1$ ont donc encore la même base, et ainsi de suite.
b) Soient $\mu \in \mathbb{R}^p$ et $c^\mu := c + A^\top \mu$. Montrons que *le changement de* c *en* c^μ *laisse invariant le coût réduit*, et donc aussi l'algorithme du simplexe avec la règle de Bland. En effet, quand on change c en c^μ, la nouvelle estimation du multiplicateur est $\lambda^\mu = -(A_B)^{-\top} c^\mu = \lambda - \mu$, et donc le nouveau coût réduit est $g^\mu = c_N^\mu - A_N^\top \lambda^\mu = c_N + A_N^\top \lambda - A_N^\top \mu + A_N^\top \mu = g$. La suite des calculs de l'algorithme (variable entrante, direction, pas, variable sortante) est donc identique.

c) Notons q le plus grand indice des variables entrantes dans la base au cours des itérations $k > k_0$. Soit $k_e > k_0$ (resp. $k_s > k_0$) une itération dans laquelle q est une variable entrante (resp. sortante). Choisissons $\mu = \lambda$ (l'estimation du multiplicateur à l'itération k_e). Il vient, à l'itération k_e, $c_B = 0$, $\lambda^\mu = 0$, et donc $g = c_N + A_N^\top \lambda^\mu = c_N$. D'après la règle de Bland, $g_i = c_i \geqslant 0$ pour tout $i \in N$, $i < q$. Comme $c_B = 0$, on a bien $c_i \geqslant 0$, pour tout $i < q$. Puisque la variable q est entrante, on a aussi $c_q < 0$.
d) Notons B^e et B^s les bases aux itération k_e et k_s, et soit d la direction de déplacement à l'itération k_s, la variable entrante étant d'indice j. Alors la quantité

$$c \cdot d = \sum_{i \in B^s \cap B^e} c_i d_i + \sum_{i \in B^s \setminus B^e, i \neq q} c_i d_i + c_q d_q + c_j \tag{6.23}$$

est positive. En effet la première somme est nulle car $c_i = 0$ pour tout $i \in B^e$. Pour $i \in B^s \setminus B^e$, $i \neq q$ on a $i < q$ (par définition de q, car i est rentrée à une itération $k > k_0$), donc $c_i \geqslant 0$, et $x_i = 0$ (puisque depuis la rentrée de l'indice i il n'y a eu que des pas nuls), de sorte que $d_i \geqslant 0$ d'après la règle de fixation de la variable sortante. La seconde somme est donc positive. Enfin $c_q < 0$ et $d_q < 0$, et $c_j \geqslant 0$ (puisque $j < q$ par définition de q). Or l'algorithme du simplexe impose $c \cdot d < 0$, ce qui constitue la contradiction cherchée. ■

Exercice 6.21 (Pratiquer l'algorithme du simplexe) Soit le problème de programmation linéaire [Chv83] :

$$\begin{aligned} \underset{x \geqslant 0}{\text{Max}} \quad & 5x_1 + 4x_2 + 3x_3; \\ & 2x_1 + 3x_2 + x_3 \leqslant 5 \\ & 4x_1 + x_2 + 2x_3 \leqslant 11 \\ & 3x_1 + 4x_2 + 2x_3 \leqslant 8 \ . \end{aligned}$$

Résoudre ce problème à l'aide de l'algorithme du simplexe. On partira de la solution réalisable $x_1 = x_2 = x_3 = 0$.

Exercice 6.22 (Résoudre par le simplexe la relaxation du sac à dos) Soit la relaxation suivante du problème de sac à dos

$$\underset{x \geqslant 0}{\text{Max}} \sum_{i=1}^{n} p_i x_i; \quad \sum_{i=1}^{n} a_j x_i \leqslant b \tag{6.24}$$

(avec $a_i > 0$, $p_i > 0$ pour tout i). Etudier sa résolution par l'algorithme du simplexe. On pourra introduire une variable d'écart pour se ramener à une contrainte d'égalité, et choisir comme point initial de ne mettre aucun objet dans le sac.

On notera $w_i = p_i/a_i$ les valeurs relatives (à l'encombrement).

6.2.3 Variable entrante de plus forte pente (steepest edge)

Nous avons noté que le choix classique de la variable entrante est la variable non basique de plus faible coût réduit. Un autre choix possible, qui s'avère en général plus efficace, est celui de la plus forte pente dans l'espace $\mathbb{R}^n$. À la variable $j \in N$ est associée la direction $d^j = (d^j_B, d^j_N)$ avec $d^j_N = e_j$ et $d^j_B = -A_B^{-1} A_j$. La *pente* (associée à la norme euclidienne) est définie comme $c \cdot d^j / \|d^j\|$. La variable $q \in N$ de plus forte pente est donc telle que

$$q \in \underset{j \in N}{\text{argmin}} \frac{c \cdot d^j}{\|d^j\|}. \tag{6.25}$$

Comme $c \cdot d^j = g_j$ n'est autre que le coût réduit de la variable j, qui est évalué lors de l'itération du simplexe, la difficulté du calcul réside dans l'estimation rapide de $\|d^j\|$, pour tout $j \in N$. Comme $\|d^j\|^2 = 1 + \|d^j_B\|^2$, il suffirait de calculer d^j_B, mais ceci serait prohibitif quand $|N|$ est grand, ce qui est le plus souvent le cas. Montrons cependant comment mettre à jour à faible coût les $\|d^j\|$, pour $j \in N$, si ces quantités ont été

calculées à l'itération précédente. On peut supposer que $B = \{1, \dots, p\}$. Soient q la variable entrante et $r \leqslant p$ la variable sortante. La nouvelle matrice de base et son inverse sont, d'après (6.11) :

$$\bar{A}_B = A_B + (A_q - A_r)e_r^\top; \quad \bar{A}_B^{-1} = A_B^{-1} - \frac{A_B^{-1}(A_q - A_r)e_r^\top A_B^{-1}}{1 + (A_q - A_r) \cdot A_B^{-\top} e_r}. \tag{6.26}$$

On a donc, notant $\bar{d}^j$ les nouvelles directions

$$\bar{d}_B^j = d_B^j + \frac{A_B^{-1}(A_q - A_r)(A_B^{-\top} e_r)^\top A_j}{1 + (A_q - A_r) \cdot A_B^{-\top} e_r}. \tag{6.27}$$

Posons

$$\left\{ \begin{array}{l} u = A_B^{-\top} e_r, \ \ \beta := (1 + (A_q - A_r) \cdot u)^{-1}, \ \ \alpha_j := \beta u \cdot A_j, \\ w = A_B^{-1}(A_q - A_r), \ \ z = A_B^{-\top} w. \end{array} \right.$$

Alors

$$\bar{d}_B^j = d_B^j + \alpha_j w = d^j + \alpha_j A_B^\top z. \tag{6.28}$$

Utilisant $A_B d^j = -A_j$, il vient

$$\|\bar{d}_B^j\|^2 = \|d_B^j\|^2 - 2\alpha_j z \cdot A_j + \alpha_j^2 \|w\|^2. \tag{6.29}$$

Au total, la méthode nécessite le calcul des $\gamma_j = \|d^j\|^2$ à la première itération, puis à chaque itération, sachant que $w = A_B^{-1}(A_q - A_r)$ est déjà calculé (remise à jour de la matrice de base), la résolution de deux systèmes linéaires ($u = A_B^{-\top} e_r$ et $z = A_B^{-\top} w$), le calcul de β. Enfin, notant $u' := \beta u$, il faut, pour tout $j \in N$, calculer les produits scalaires $\alpha_j := u' \cdot A_j$ et $z \cdot A_j$, ce qui requiert un nombre d'opérations de l'ordre du nombre d'éléments non nuls de A, comme pour le calcul du coût réduit. En dehors de la première itération, les calculs sont donc de même ordre de complexité (variable suivant le creux de la matrice A) que les calculs de l'itération standard du simplexe.

6.2.4 Simplexe dual

On peut mettre en œuvre une méthode de simplexe appliquée à la formulation duale

$$\underset{\lambda, s}{\text{Max}} -b \cdot \lambda; \quad c + A^\top \lambda = s; \quad s \geqslant 0. \tag{6.30}$$

Ceci est très utilisé dans la résolution de problèmes en nombres entiers, dans lesquels on est amené à résoudre une suite de programme linéaires obtenus en ajoutant à la formulation précédente des contraintes linéaires supplémentaires (coupes) ; voir la section 8.1. Nous avons également vu une démarche analogue pour résoudre le relâché continu du programme linéaire en nombre entiers modélisant le problème de voyageur de commerce, lequel fait intervenir un nombre exponentiel de contraintes de sous-tour (section 5.2.2).

Dans de tel cas, la solution du problème précédent fournit un point réalisable du dual du problème suivant, qui est souvent résolu par le simplexe (dual) en peu d'itérations.

Le principe est identique à celui du simplexe primal, en tenant compte du fait que la variable λ a un signe quelconque. La base duale (de taille n) contient toujours λ, qui ne peut sortir de la base puisqu'il est non borné ; il y a donc $n - p$ composantes de s dans la base duale. Pour des raisons qui apparaîtront plus loin, notons N (resp. B) l'ensemble des composantes *basiques* (resp. *hors base*) de s. On a $|B| = p$ et $|N| = n - p$. Montrons que la matrice de base est inversible ssi A_B l'est. En effet, le système linéaire correspondant est $A^\top \lambda - s_N = s_B - c$ (dans cette écriture, on identifie s_B et s_N à leur prolongement par zéro à $\mathbb{R}^n$), et on résout donc d'abord $A_B^\top \lambda = s_B - c_B$, ce qui nécessite bien l'inversibilité de A_B, puis il vient $s_N = A_N^\top \lambda + c_N$.

Notons x le multiplicateur associé aux contraintes linéaires. Le lagrangien du problème dual (sans dualisation de la contrainte de positivité de s) s'écrit

$$L(x, \lambda, s) = -b \cdot \lambda + x \cdot (c + A^\top \lambda - s) = c \cdot x + \lambda \cdot (Ax - b) - s \cdot x. \tag{6.31}$$

Nous retrouvons une expression identique à celle du lagrangien du problème primal. On peut observer que l'estimation du multiplicateur (dans l'algorithme du simplexe) est celle qui rend stationnaire le lagrangien par rapport aux variables de base. Ici les variables de base sont (λ, s_N) et le multiplicateur x est donc obtenu comme solution de $(Ax - b, x_N) = 0$, de solution unique $x_b = A_B^{-1} b$, $x_N = 0$.

Le coût réduit par rapport aux variables hors base (qui se calcule comme la dérivée partielle du lagrangien par rapport aux variables hors base) s_B est $-x_B$. On cherche à le rendre négatif (car le problème dual est une maximisation). Choisir une direction de montée du coût dual revient à faire rentrer dans la base un indice $i \in B$ pour lequel $x_i < 0$. On peut faire rentrer dans la base, par exemple, la composante la plus négative de x_B. Lorsque $x_B \geqslant 0$ on a satisfait les conditions d'optimalité.

La direction de déplacement se calcule ainsi : Il faut résoudre en (d^λ, d^s_N) les équations $A^\top d^\lambda = d^s_N + e_i$ (e_i est le vecteur de base associé à l'indice i de la variable rentrante, et on identifie ici d^s_N et son extension par zéro sur $\mathbb{R}^n$). Ceci se réécrit

$$\left\{ \begin{array}{lcl} A_B^\top d^\lambda & = & e_i, \\ A_N^\top d^\lambda & = & d^s_N. \end{array} \right. \tag{6.32}$$

Pour cela on calcule d'abord d^λ, solution du premier bloc, et il vient alors $d^s_N = A_N^\top d^\lambda$. On voit que, comme dans le simplexe primal, l'essentiel du travail est la factorisation et la mise à jour de la matrice A_B. Le coût d'une itération du simplexe dual est donc identique à celui d'une itération du simplexe primal.

6.2.5 Simplexe avec bornes quelconques

En pratique les variables d'un programme linéaire comportent des bornes quelconques, ce qui amène à introduire le format suivant :

$$\min_x c \cdot x; \quad Ax = b; \quad \ell_i \leqslant x_i \leqslant u_i, \quad i = 1, \ldots, n. \tag{LPB}$$

Certaines des composantes de ℓ (resp. u) peuvent être égales à $-\infty$ (resp. $+\infty$) ce qui correspond à l'absence de borne inférieure (resp. supérieure) sur la composante correspondante de A.

On suppose encore A de taille $p \times n$, de rang $p \leqslant n$, et une base est toujours une partie de $\{1, \cdots, n\}$ de cardinal p, telle que A_B est inversible. On partitionne l'ensemble $N := \{1, \cdots, n\} \setminus B$ des variables hors base en $N = N_1 \cup N_2$ ($N_1 \cap N_2 = \emptyset$), où N_1 (resp. N_2 représente l'ensemble des variables hors base égales à leur borne inférieure (resp. supérieure). Le point basique x associé à la partition (B, N_1, N_2) est tel que

$$x_i = \ell_i, \, i \in N_1, \quad x_i = u_i, \, i \in N_2; \quad A_B x_B = b - A_N x_N. \tag{6.33}$$

Noter qu'une composante sans bornes est nécessairement dans la base. On montre (par le même type d'argument que pour le format standard) que si (LPB) a une valeur finie, et si $F(LPB)$ ne contient pas de droite affine (la remarque 6.23 indique comment se ramener à ce cas), alors il a une solution basique (une solution qui est un point de base). Il est alors facile d'étendre à ce format l'algorithme du simplexe : la variable entrante sera de coût réduit strictement négatif si elle est dans N_1, positif si elle est dans N_2.

Remarque 6.23 Notons X le sous espace vectoriel de $\mathbb{R}^n$ qui a pour éléments les vecteurs dont les composantes sont nulles pour toutes les variables de (LPB) ayant au moins une borne. Si (LPB) est réalisable, l'absence de droite affine contenue dans $F(LPB)$ revient à dire que $X_A := X \cap \operatorname{Ker} A$ se réduit à $\{0\}$.

Si au contraire il existe $d \neq 0$ dans X_A, on voit que si $c \cdot d \neq 0$, le problème ne peut avoir de solution. Si au contraire $c \cdot d = 0$, supposant sans perte de généralité que $d_n = 1$, on voit qu'on peut sans perte de généralité donner une valeur nulle à x_n, ce qui revient à retirer cette variable et donc se ramener à un problème à $n-1$ variables. Itérant si nécessaire, on se ramène (si le problème a une solution) à une formulation réduite dans laquelle $F(LPB)$ ne contient pas de droite affine.

L'algorithme a l'expression suivante :

Algorithme 6.24 [Itération du simplexe avec bornes]
Données : (B, N_1, N_2) base admissible non dégénérée, x point de base associé.

1. Estimation du multiplicateur, par résolution de $A_B^\top \lambda = -c_B$.
2. Calcul du coût réduit $g \leftarrow c_N + A_N^\top \lambda$.
3. Si $g_{N_1} \geqslant 0$ et $g_{N_2} \leqslant 0$, arrêt : x est solution de (LPB).
4. Direction de déplacement, au choix :
   $\left\{ \begin{array}{l} j \in N_1 \text{ tel que } g_j < 0\,;\ d_N \leftarrow e_j;\ d_B \leftarrow -A_B^{-1} A_j. \\ j \in N_2 \text{ tel que } g_j > 0\,;\ d_N \leftarrow -e_j;\ d_B \leftarrow A_B^{-1} A_j. \end{array} \right.$
5. Si pour tout i, $u_i = +\infty$ si $d_i > 0$, et $\ell_i = -\infty$ si $d_i < 0$, arrêt : $\mathrm{val}(LP) = -\infty$.
6. Calcul de l'indice saturant une contrainte :
   $I_1 \leftarrow \{(\ell_i - x_i)/d_i;\ d_i < 0,\ \ell_i > -\infty,\ 1 \leqslant i \leqslant n\}$.
   $I_2 \leftarrow \{(u_i - x_i)/d_i;\ d_i > 0,\ u_i < +\infty,\ 1 \leqslant i \leqslant n\}$.
   Si $I_1 = \emptyset$, $\rho_1 \leftarrow +\infty$. Sinon, $\hat{\ell}_1 \in \mathrm{argmin}\, I_1$; $\rho_1 \leftarrow (\ell_{\hat{\ell}_1} - x_{\hat{\ell}_1})/d_{\hat{\ell}_1}$.
   Si $I_2 = \emptyset$, $\rho_2 \leftarrow +\infty$. Sinon, $\hat{\ell}_2 \in \mathrm{argmin}\, I_2$; $\rho_2 \leftarrow (u_{\hat{\ell}_2} - x_{\hat{\ell}_2})/d_{\hat{\ell}_2}$.
   Si $\rho_1 \leqslant \rho_2$, $\rho \leftarrow \rho_1$, $\hat{\ell} \leftarrow \hat{\ell}_1$; sinon, $\rho \leftarrow \rho_2$, $\hat{\ell} \leftarrow \hat{\ell}_2$.
7. Nouveau point $x \leftarrow x + \rho d$.
8. Mise à jour de la base
   Si $j \in N_1$, $N_1 \leftarrow N_1 \backslash \{j\}$; sinon, $N_2 \leftarrow N_2 \backslash \{j\}$.
   Si $\hat{\ell} = j$ (donc B inchangé !) :
      Si $j \in N_1$, $N_2 \leftarrow N_2 \cup \{j\}$; sinon, $N_1 \leftarrow N_1 \cup \{j\}$.
   Si $\hat{\ell} \neq j$:
      $B \leftarrow (B \cup \{j\}) \backslash \{\hat{\ell}\}$.
      Si $\rho_1 \leqslant \rho_2$: $N_1 \leftarrow N_1 \cup \{\hat{\ell}\}$; sinon, $N_2 \leftarrow N_2 \cup \{\hat{\ell}\}$.

6.2.6 Simplexe dual avec bornes quelconques

On reprend le problème (LPB) de la section précédente. Le lagrangien s'écrit

$$\begin{array}{rl} L(x,\lambda,s,z) & := c \cdot x + \lambda \cdot (Ax - b) + s \cdot (\ell - x) + z \cdot (x - u). \\ & = s \cdot \ell - z \cdot u - \lambda \cdot b + (c + A^\top \lambda - s + z) \cdot x. \end{array} \tag{6.34}$$

Le problème dual a donc pour expression

$$\underset{\lambda,s,z}{\mathrm{Max}}\ s \cdot \ell - z \cdot u - \lambda \cdot b\,; \quad c + A^\top \lambda - s + z = 0; \quad s \geqslant 0;\ z \geqslant 0. \tag{LDB}$$

Comme dans le cas du format standard, λ fait partie des n variables de la base duale N. On note N_1 (resp. N_2) les composantes basiques de s (resp z). L'estimation du "multiplicateur" x étant obtenue en rendant stationnaire le lagrangien par rapport aux variables de base, on obtient

$$x_{N_1} = \ell_{N_1}; \quad x_{N_2} = u_{N_2}. \tag{6.35}$$

On voit que ceci suppose que $N_1 \cap N_2 = \emptyset$. On note $B := \{1, \cdots, n\} \setminus N$ l'ensemble de cardinal m des indices i tels que s_i et z_i soient tous les deux hors base (duale). La stationnarité du lagrangien par rapport à la variable de base λ implique $Ax = b$, et donc puisque x_N est connu ceci détermine x_B :

$$A_B x_B = b - A_N x_N. \tag{6.36}$$

Rappelons que le coût réduit est la dérivée partielle du lagrangien par rapport aux variables hors base. Il vient

$$\left\{ \begin{array}{l} \text{Si } i \in B \cup N_2 : g_i^s = \dfrac{\partial L}{\partial s_i} = \ell_i - x_i \\ \text{Si } i \in B \cup N_1 : g_i^z = \dfrac{\partial L}{\partial z_i} = x_i - u_i \end{array} \right. \tag{6.37}$$

Noter que, si $i \in N_1$, alors $x_i = \ell_i$ donc $g_i^z < 0$, et de même si $i \in N_2$, alors $x_i = u_i$ donc $g_i^s < 0$. La variable entrante j appartient donc nécessairement à B. C'est une composante de s si $x_i < \ell_i$, et de z sinon. Le calcul de la direction de déplacement nécessite la résolution de

$$A^\top d^\lambda = d^s_{N_1} - d^z_{N_2} + d^e, \quad \text{où } d^e = e_j \text{ si } x_j < \ell_j, \text{ et } d^e = -e_j \text{ sinon.} \tag{6.38}$$

On calcule d'abord d^λ, puis $d^s_{N_1}$ et $d^z_{N_2}$ avec les relations suivantes :

$$A_B^\top d^\lambda = d^e; \quad d^s_{N_1} = A^\top_{N_1} d^\lambda; \quad d^z_{N_2} = -A^\top_{N_2} d^\lambda. \tag{6.39}$$

L'algorithme a l'expression suivante :

Algorithme 6.25 [Itération de l'algorithme du simplexe dual avec bornes]
Données : base duale admissible non dégénérée (B, N_1, N_2), point de base associé (λ, s, z).

1. Estimation du "multiplicateur" x par (6.35)-(6.36).
2. Calcul du coût réduit par (6.37).
3. Si $g^s \leqslant 0$ et $g^z \leqslant 0$, arrêt : (λ, s, z) est solution duale de (LPB).
4. Direction de déplacement $j \in B$, telle que
   $d^e \leftarrow e_j$ si $x_j < \ell_j$, $d^e \leftarrow -e_j$ sinon.
   Calcul de la direction de déplacement dual basique par (6.39).
5. Si $d^s \geqslant 0$ et $d^z \geqslant 0$, arrêt : $\mathrm{val}(LP) = +\infty$.
6. Calcul de l'indice saturant une contrainte :
   $I_1 \leftarrow \{-s_i/d^s_i;\ d_i < 0,\ i \in N_1\}$.
   $I_2 \leftarrow \{-u_i/d_i;\ d_i < 0,\ \ i \in N_2\}$.
   Si $I_1 = \emptyset$, $\rho_1 \leftarrow +\infty$. Sinon, $\hat\ell_1 \in \operatorname{argmin} I_1$; $\rho_1 \leftarrow -s_{\hat\ell_1}/d^s_{\hat\ell_1}$.
   Si $I_2 = \emptyset$, $\rho_2 \leftarrow +\infty$. Sinon, $\hat\ell_2 \in \operatorname{argmin} I_2$; $\rho_2 \leftarrow -u_{\hat\ell_2}/d_{\hat\ell_2}$.
   Si $\rho_1 \leqslant \rho_2$, $\rho \leftarrow \rho_1$, $\hat\ell \leftarrow \hat\ell_1$; sinon, $\rho \leftarrow \rho_2$, $\hat\ell \leftarrow \hat\ell_2$.
7. Nouveau point $(\lambda, s, z) \leftarrow (\lambda, s, z) + \rho d$.
8. Mise à jour de la base
   Si $d^e_j > 0$, $N_1 \leftarrow N_1 \cup \{j\}$; sinon, $N_2 \leftarrow N_2 \cup \{j\}$.
   Si $\rho_1 \leqslant \rho_2$, $N_1 \leftarrow N_1 \backslash \{\hat\ell_1\}$; sinon, $N_2 \leftarrow N_2 \backslash \{\hat\ell_2\}$.
   $B \leftarrow (B \cup \{\hat\ell\}) \backslash \{j\}$.

Exercice 6.26 Appliquer le simplexe dual à la relaxation du sac à dos). Résoudre le problème du sac à dos relaxé (avec $a_i > 0$, $p_i > 0$ pour tout i)

$$\operatorname*{Max}_{x \in [0,1]^n} \sum_{i=1}^n p_i x_i; \quad \sum_{i=1}^n a_j x_i \leqslant b \tag{6.40}$$

par l'algorithme du simplexe dual, initialisé en prenant la solution du problème sans la contrainte linéaire.

6.3 Notes

La méthode du gradient réduit, inspirée de celle du simplexe, a été introduite dans [Wol63] ; voir aussi [Bon06]. Cette méthode se généralise au prix de complications techniques aux contraintes non linéaires [Aba86]. La règle de Markowitz est introduite dans [Mar57].

En général, pour les problèmes de grande taille, la matrice A est creuse (elle a beaucoup d'éléments nuls) et on calcule les facteurs LU de A_B (méthode de Gauss) afin de résoudre les systèmes linéaires. Ces facteurs LU sont eux-mêmes stockés sous forme creuse. La mise à jour de la matrice de base de rang 1 permet une mise à jour rapide des facteurs LU, voir Reid [Rei82].

La référence classique sur le simplexe est Dantzig [Dan63]. Le choix du "steepest edge" est présenté dans [FG92]. D'autres règles de pivotage sont discutées par Terlaky et Zhang [TZ93]. Nous avons exposé la règle lexicographique de Bland [Bla77] qui permet (en arithmétique exacte) d'assurer la convergence en un nombre fini d'itérations de l'algorithme du simplexe. Les améliorations critiques des performances sont discutés dans [BFG+00].

Au moins deux programmes libres de très bonne qualité permettent de traiter des applications de programmation linéaire, il s'agit de glpk [1] et de lpsolve [2]. Outre l'algorithme du simplexe, glpk comprend un algorithme de points intérieurs. Les deux programmes permettent de traiter des problèmes en nombre entiers.

1. http ://www.gnu.org/software/glpk/
2. http ://sourceforge.net/projects/lpsolve

Chapitre 7

Coupes d'intégrité

Ce chapitre présente les méthodes de coupe pour la résolution de programmes linéaires en nombres entiers (PLNE). Après avoir illustré le principe par la coupe de Dantzig, nous exposons les coupes de Gomory, les coupes de Chvátal qui les généralisent, et certaines coupes pour des problèmes spécifiques : ensembles stables et sac à dos.

Le chapitre se termine par des compléments sur les polyèdres entiers, les équations linéaires diophantiennes, l'intégrité duale totale et la convergence des coupes de Chvátal.

7.1 Principe des coupes d'intégrité

7.1.1 Schéma général

Etant donné un polyèdre P, on désigne par P_I et on appelle *clôture entière* l'enveloppe convexe des points entiers de ce polyèdre :

$$P_I := \text{conv}\,(P \cap \mathbb{Z}^n)\,. \tag{7.1}$$

Le résultat élémentaire suivant montre qu'un programme linéaire en nombre entiers se ramène à minimiser une fonction linéaire sur la clôture entière du polyèdre considéré. Nous verrons plus loin que la clôture entière P_I est un polyèdre, dès lors que P est décrit par des inégalités à coefficients rationnels. Si l'on savait calculer P_I, on se ramènerait ainsi à un programme linéaire ordinaire.

Lemme 7.1 *Un critère linéaire a le même infimum sur $P \cap \mathbb{Z}^n$ et sur P_I.*

Démonstration. Notons $v_1 := \inf_x\{c \cdot x;\ x \in P \cap \mathbb{Z}^n\}$, $v_2 := \inf_x\{c \cdot x;\ x \in P_I\}$. Comme $P \cap \mathbb{Z}^n \subset P_I$, on a $v_1 \geqslant v_2$. Soit maintenant $x \in P_I$. Alors $x = \alpha_1 x^1 + \ldots + \alpha_q x^q$ avec les α_i strictement positifs de somme un, et $x^i \in P \cap \mathbb{Z}^n$ pour tout i, donc $c \cdot x \geqslant \min_i c \cdot x^i \geqslant v_1$, ce qui prouve que $v_1 \leqslant v_2$. ■

La figure 7.1 illustre le lemme 7.1. Cette figure motive l'introduction de la notion de *coupe d'intégrité*.

Définition 7.2 Une *coupe d'intégrité* du problème linéaire en nombre entiers

$$\text{Min}\, c \cdot x;\ x \in P \cap \mathbb{Z}^n$$

est une inégalité $a \cdot x \leqslant \beta$ satisfaite en tout point de $P \cap \mathbb{Z}^n$ et qui permet d'éliminer certains points de P. On dira que la coupe est *relative* à un point $\bar{x} \in P$ si elle exclut ce point.

L'idée des coupes d'intégrité consiste à résoudre tout d'abord le relâché continu du programme linéaire en nombres entiers, puis, si la solution optimale de ce relâché continu n'est pas entière, à rajouter une coupe excluant cette solution, puis à réitérer. Nous obtenons l'algorithme de principe suivant :

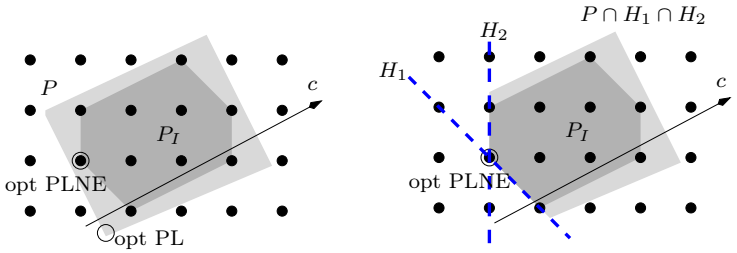


FIGURE 7.1 – Un polyèdre P (quadrilatère gris clair) et sa clôture entière P_I (hexagone gris foncé). L'optimum du programme linéaire en nombre entiers (opt PLNE) est comparé à l'optimum de son relâché continu (opt PL), pour le coût $x \mapsto c \cdot x$. L'ajout de deux coupes d'intégrité (demi-plans H_1 et H_2) conduit à un problème de minimisation sur un nouveau polyèdre (pentagone), qui redonne l'optimum du PLNE (figure de droite).

Algorithme 7.3 (Coupe d'intégrité)
begin coupe
data : A, b, c. Polyèdre $P = \{x \mid Ax \leqslant b\}$.
init : Calculer $x \in \operatorname{argmin}\{c \cdot x;\ x \in P\}$; arrêt si le problème n'est pas réalisable.
while x non entier
Intégrer à la formulation une ou plusieurs coupes d'intégrité, i.e., remplacer P par $P \cap H_1 \cap \cdots \cap H_k$, où $H_1, \ldots, H_k$ sont des demi-espaces contenant $P \cap \mathbb{Z}^n$ et tels que $x \notin H_1, \ldots, H_k$.
Calculer $x \in \operatorname{argmin}\{c \cdot x;\ x \in P\}$; arrêt si le problème n'est pas réalisable.
end while
end coupe

L'algorithme ci-dessus se termine ssi il produit une solution entière ou conclut à la non-réalisabilité. À chaque étape, on résout un programme linéaire. En pratique on combine presque toujours les idées de coupe avec celles de branchement, présentées dans le chapitre 5.

Nous allons maintenant décrire plusieurs méthodes qui permettent de générer des coupes d'intégrité de manière systématique.

7.1.2 Coupes de Dantzig

Considérons le problème de programmation linéaire en nombres entiers (PLNE) sous forme standard

$$\min_x\ c \cdot x; \quad Ax = b; \quad x \in \mathbb{N}^n, \tag{PLNE}$$

avec A de taille $p \times n$ de rang p (donc $p \leqslant n$). La première étape d'un algorithme consiste souvent à résoudre la relaxation continue de ce problème :

$$\operatorname{Min} c \cdot x; \quad x \in \mathbb{R}^n_+;\ Ax = b. \tag{PL}$$

On sait que, si la valeur de (PL) est finie, il a au moins une solution, et même une solution de base $\bar{x}$ (qu'on peut calculer par un algorithme de type simplexe), voir la section 6.2. Rappelons que cela signifie qu'il existe une partition (B, N) de $\{1, \ldots, n\}$, telle que la matrice de base A_B (formée des colonnes de A à indices dans B) est de rang p (donc $|B| = p$ et A_B est inversible), et $\bar{x}_N = 0$ (on note par x_N la collection des $x_i = 0$, pour $i \in N$).

Si $\bar{x}$ est entier, c'est un point de $F(PLNE)$ qui réalise le minimum du critère sur un ensemble contenant $F(PLNE)$; il est donc solution de $(PLNE)$. Discutons dans la suite le cas où $\bar{x} \notin F(PLNE)$.

Si $x \in F(PLNE)$, puisque A_B est inversible, $x_N = 0$ implique $x = \bar{x}$ ce qui est impossible. Comme les éléments de $F(PLNE)$ ont des coordonnées entières positives, on a $x_i \geqslant 1$ pour au moins un $i \in N$. En

particulier, tout $x \in F(PLNE)$ satisfait la *coupe de Dantzig* :

$$\sum_{i \in N} x_i \geqslant 1. \tag{7.2}$$

L'ajout de cette inégalité au problème original laisse invariant l'ensemble des points réalisables (entiers), et permet de réduire l'ensemble réalisable de la relaxation continue. Cette coupe peut donc être exploitée dans l'algorithme 7.3.

Remarque 7.4 Considérons le cas d'un problème linéaire en nombres entiers avec contraintes de borne quelconques, qu'on peut toutefois toujours supposer entières :

$$\operatorname*{Min}_x c \cdot x; \quad Ax = b; \quad x \in \mathbb{Z}^n, \quad \ell \leqslant x \leqslant u, \tag{PLNEB}$$

avec ℓ, u dans $\mathbb{Z}^n$ tels que $\ell_i < u_i$ pour tout $i \in \{1, \ldots, n\}$. Le problème (LPB) de la section 6.2.5 s'interprète alors comme le relaxé continu de $(PLNEB)$. Nous avons introduit la notion de point de base pour ce type de problème, et établi que si la valeur de (LPB) est finie, alors il possède un solution de base (vérifiant (6.33)), avec (B, N_1, N_2) partition en variables de base et hors base sur borne inférieure/supérieure). Si $\bar{x}$ est une solution basique, associée à cette partition, un raisonnement identique à celui qui précède conduit, si la solution du relaxé continu n'est pas entière, à la coupe

$$\sum_{i \in N_1} (x_i - \ell_i) + \sum_{i \in N_2} (u_i - x_i) \geqslant 1. \tag{7.3}$$

•

Exemple 7.5 Soit la variante du problème de type sac à dos (où on demande que le sac soit totalement rempli)

$$\operatorname{Max} 3x_1 + 2x_2 + x_3; \quad 2x_1 + 3x_2 + 2x_3 = 4; \quad x \in \{0,1\}^3. \tag{7.4}$$

La solution du relaxé continu est $x^1 = (1, 2/3, 0)$, et ici $B = \{2\}$, $N_1 = \{3\}$, $N_2 = \{1\}$. La coupe de Dantzig s'écrit $(1 - x_1) + x_3 \geqslant 1$, soit $x_1 \leqslant x_3$. Cette coupe est active à la solution du nouveau relaxé continu. On vérifie alors que ce dernier a la solution entière $x^2 = (1, 0, 1)$, qui est donc solution du problème (7.4).

On ne connaît pas d'algorithme de coupe d'intégrité convergent basé sur les coupes de Dantzig. Ces dernières sont de plus considérées comme peu efficaces, même combinées avec la technique de séparation et évaluation. On leur préfère d'autres types de coupes que nous exposons maintenant.

7.1.3 Coupe de Gomory fractionnaire

Notons l'ensemble réalisable pour la relaxation continue de $(PLNE)$ par

$$P = \{x \in \mathbb{R}^n_+;\ Ax = b\} \tag{7.5}$$

avec A de taille $p \times n$ de rang p. Soit $\mu \in \mathbb{R}^p$ un vecteur. Si $x \in P$, alors $\mu \cdot Ax = \mu \cdot b$. Cette égalité peut s'écrire, notant $a^\mu := A^\top \mu$ et $b^\mu := \mu \cdot b$, comme $a^\mu \cdot x = b^\mu$. Si $x \in F(PLNE)$, on a l'égalité modulo 1 en ne retenant que les parties fractionnaires des coefficients :

$$\sum_{i=1}^{n} (a^\mu_i - \lfloor a^\mu_i \rfloor)\, x_i = b^\mu - \lfloor b^\mu \rfloor \qquad \text{modulo 1.} \tag{7.6}$$

Comme le membre de gauche est positif, celui de droite étant dans $[0, 1[$, on en déduit la *coupe de Gomory fractionnaire*

$$\sum_{i=1}^{n} (a^\mu_i - \lfloor a^\mu_i \rfloor)\, x_i \geqslant b^\mu - \lfloor b^\mu \rfloor. \tag{7.7}$$

On choisit habituellement les poids μ en rapport avec une variable de base non entière. Si (B, N) est une partition optimale des indices pour le relaxé continu, avec pour solution associée $\bar{x}$, on se ramène au cas où $B = \{1, \ldots, p\}$. Fixons i dans B. On note $A = (A_B, A_N)$ la partition correspondante de A. Comme la matrice de base est inversible, il existe des poids μ pour lesquels la relation $a^\mu x = b^\mu$ est de la forme

$$x_i + \sum_{j \in N} a^\mu_j x_j = b^\mu. \tag{7.8}$$

De fait cette relation est la $i$ème ligne du produit de $Ax = b$ par A_B^{-1}. Pour éviter d'expliciter des inverses de matrices (ce qui est numériquement coûteux), on calcule plutôt μ solution de $A_B^\top \mu = e_i$ (e_i est le $i$ème vecteur de la base naturelle de $\mathbb{R}^p$). Comme la $i$ème ligne de $A_B^{-1}A = (I, A_B^{-1}A_N)$ s'obtient en multipliant à gauche A par le vecteur ligne $\mu^\top := e_i^\top A_B^{-1}$, on obtient la relation cherchée. Si $\bar{x}$ est entier il est solution de $(PLNE)$, sinon (puisque $\bar{x}_N = 0$) il existe au moins $i \in B$ tel que $\bar{x}_i = b^\mu \notin \mathbb{N}$. La coupe de Gomory a pour expression (bien noter que la somme ne porte que sur les indices hors base) :

$$\sum_{j \in N} \left(a^\mu_j - \lfloor a^\mu_j \rfloor\right) x_j \geqslant b^\mu - \lfloor b^\mu \rfloor. \tag{7.9}$$

Comme $\bar{x}_i = b^\mu$, les parties fractionnaires de ces deux quantités sont égales, et donc le membre de droite de (7.9) est non nul. La coupe (7.9) exclut donc bien le point $\bar{x}$.

On peut générer autant de coupes de Gomory que de composantes de base non entières. Récapitulons les calculs à faire en pratique : on calcule μ solution de $A_B^\top \mu = e_i$ ($i$ème vecteur de base), puis $a^\mu_N = A_N^\top \mu$; rappelons que $b^\mu = \bar{x}_i$.

Remarque 7.6 Le vecteur a^μ peut être beaucoup moins creux que la ligne correspondante de la matrice A. Si p est beaucoup plus petit que n, on peut limiter l'encombrement mémoire en ne retenant la coupe que si le vecteur a^μ est suffisamment creux. •

Remarque 7.7 Dans le cas du problème $(PLNEB)$ avec bornes quelconques décrit dans la remarque 7.4, on peut appliquer la coupe de Gomory obtenue en ramenant toutes les variables hors base à 0, avec contrainte de non négativité de ces variables (si $x_i = u_i$, introduire les variable $y_i = u_i - x_i$). Réécrivant la contrainte correspondant à (7.8) (exprimé en fonction de x et y) sous la forme

$$x_i + \sum_{j \in N_1} a^\mu_j (x_j - \ell_j) - \sum_{j \in N_2} a^\mu_j (u_j - x_j) = \bar{b}^\mu := b^\mu - \sum_{j \in N_1} a^\mu_j \ell_i - \sum_{j \in N_2} a^\mu_j u_j, \tag{7.10}$$

on obtient l'expression de la coupe de Gomory :

$$\sum_{j \in N_1} \left(a^\mu_j - \lfloor a^\mu_j \rfloor\right)(x_j - \ell_j) + \sum_{j \in N_2} \left(-a^\mu_j - \lfloor -a^\mu_j \rfloor\right)(u_j - x_j) \geqslant \bar{b}^\mu - \lfloor \bar{b}^\mu \rfloor. \tag{7.11}$$

•

Exemple 7.8 Reprenons le problème de l'exemple 7.5. Nous avons déjà calculé la solution du problème relaxé ainsi que (B, N_1, N_2). Le multiplicateur $\mu := 1/3$ permet d'écrire la contrainte de remplissage sous la forme

$$x_2 + \frac{2}{3}x_1 + \frac{2}{3}x_3 = \frac{4}{3} \tag{7.12}$$

ou encore, puisque $N_1 = \{3\}$ et $N_2 = \{1\}$:

$$x_2 - \frac{2}{3}(1 - x_1) + \frac{2}{3}x_3 = \bar{b}^\mu = \frac{4}{3} - \frac{2}{3} = \frac{2}{3}. \tag{7.13}$$

La coupe de Gomory s'écrit donc

$$\frac{1}{3}(1 - x_1) + \frac{2}{3}x_3 \geqslant \bar{b}^\mu - \lfloor \bar{b}^\mu \rfloor = \frac{2}{3}, \tag{7.14}$$

soit $2x_3 \geqslant x_1 + 1$. Cette coupe étant active à la solution du nouveau relaxé continu, ce dernier peut s'écrire, en éliminant x_3,

$$\underset{x}{\text{Max}}\ 3.5x_1 + 2x_2;\ \ 3x_1 + 3x_2 = 3;\ \ x \in [0,1]^2;$$

de solution $x_1 = 1$, $x_2 = 0$, donc $x_3 = 1$; puisque cette solution est entière, elle est solution du problème (7.4).

Remarque 7.9 Le principe des coupes de Gomory s'applique aussi à l'égalité obtenue en multipliant (7.8) par un entier positif α. Supposons les a_i^μ et b^μ rationnels. On obtient la coupe d'expression

$$\sum_{j \in N} \left(\alpha a_j^\mu - \lfloor \alpha a_j^\mu \rfloor\right) x_j \geqslant \alpha b^\mu - \lfloor \alpha b^\mu \rfloor. \tag{7.15}$$

Il suffit de donner à α les valeurs 1 à $M-1$, où M est le plus petit commun multiple des dénominateurs des nombres (ramenés à des fractions irréductibles) $a_1^\mu, \ldots, a_n^\mu$ et de b^μ. En effet, les parties fractionnaires de a^μ et b^μ sont invariantes par ajout de M à α. Notons aussi que, si α est multiple du dénominateur de b^μ, l'inégalité obtenue n'apportera rien puisque le second membre sera nul. L'idéal est d'obtenir une partie fractionnaire importante pour αb^μ, et faible pour les composantes de αa^μ. On peut par exemple choisir α maximisant le ratio entre la partie fractionnaire de αb^μ et la somme des parties fractionnaires de αa^μ. •

Exercice 7.10 (Coupes de Gomory pour le sac à dos) 1. Appliquer la coupe de Gomory au problème de sac à dos

$$\underset{x}{\text{Max}} \sum_{i=1}^{n} c_i x_i; \quad x \in S,$$

où les c_i sont strictement positifs, S étant défini par (7.20). On suppose b et les a_i entiers.

2. Traiter l'application au cas où P est donné par (7.23), les c_i étant égaux à 1.

7.1.4 Coupes de Chvátal

Les coupes d'intégrité de Chvátal, appliquées au polyèdre $P = \{x;\ Ax \leqslant b\}$, A de taille $p \times n$, fournissent un cadre théorique (et non une méthode opératoire) pour la construction des coupes d'intégrité. La clôture entière P_I d'un polyèdre P a été définie en (7.1). Rappelons (lemme 7.1) qu'un critère linéaire a le même infimum sur $P \cap \mathbb{Z}^n$ et sur P_I.

Soit $H := \{x;\ c \cdot x \leqslant \beta\}$ un demi espace contenant P. Alors $P_I \subset H_I$. Les coupes de Chvátal consistent simplement à expliciter H_I pour tous les demi espaces H contenant P. Montrons d'abord que l'inclusion $P \subset H$ implique que c est de la forme $c = \sum_{i=1}^{p} \lambda_i a^i$, avec $\lambda \geqslant 0$.

Lemme 7.11 *Soient le polyèdre non vide $P = \{x;\ Ax \leqslant b\}$, avec A de taille $p \times n$, et le demi-espace $H = \{x;\ c \cdot x \leqslant \beta\}$. Alors $P \subset H$ ssi il existe $\lambda \in \mathbb{R}^p_+$ tel que $c = A^\top \lambda$ et $b \cdot \lambda \leqslant \beta$.*

Démonstration. Dire que $P \subset H$ revient à dire que le programme linéaire

$$\underset{x}{\text{Max}}\, c \cdot x;\ \ Ax \leqslant b \tag{$P_{c,\beta}$}$$

est de valeur inférieure à β. Puisque P est non vide, la valeur de $(P_{c,\beta})$ est égale à celle de son dual d'expression

$$\underset{\lambda \geqslant 0}{\text{Min}}\, b \cdot \lambda;\ \ c = A^\top \lambda. \tag{$D_{c,\beta}$}$$

Si le dual est non réalisable, la condition du lemme n'est pas satisfaite, et le problème primal est non borné, donc $P \not\subset H$. Si au contraire le dual est réalisable, sa valeur étant finie, il a une solution λ qui vérifie la condition du lemme. ■

Si c est rationnel, on peut se ramener au cas d'un vecteur c entier en multipliant (c, β) par un entier convenable. Rappelons le résultat arithmétique élémentaire suivant (*théorème de Bezout*) :

Lemme 7.12 *Soit $c \in \mathbb{Z}^n$, et $E := \{c \cdot d;\ d \in \mathbb{Z}^n\}$. Alors le plus petit élément strictement positif de E est le PGCD (plus grand commun diviseur) des composantes de c.*

Démonstration. Notons α le plus petit élément strictement positif de E. Evidemment α est multiple du PGCD. Il suffit de montrer que α divise toute composante de c. Pour tout $1 \leqslant i \leqslant n$, notons r le reste de la division euclidienne de $|c_i|$ par α, de sorte que $|c_i| = p\alpha + r$ avec p entier et $0 \leqslant r < \alpha$. Comme $\alpha = c \cdot d$ pour un certain $d \in \mathbb{Z}^n$, on a, en notant e_i le i-ième vecteur de la base canonique de $\mathbb{R}^n$, $r = c_i(\pm 1) - pc \cdot d = c \cdot (\pm e_i - pd) \in E$, et comme $r < \alpha$, on a $r = 0$, par définition de α. Donc α divise c_i. ■

Quand le PGCD des composantes de c est l'unité, on dit que ces composantes sont *premières entre elles*, et d'après l'identité de Bezout, on a

$$\text{Il existe } \bar{d} \in \mathbb{Z}^n \text{ tel que } c \cdot \bar{d} = 1. \tag{7.16}$$

L'enveloppe convexe des points entiers de H a alors une expression simple :

Proposition 7.13 *Supposons c de composantes entières, premières entre elles. Alors l'enveloppe convexe H_I des points entiers de $H := \{x \in \mathbb{R}^n;\ c \cdot x \leqslant \beta\}$ a pour expression*

$$H_I = \{x \in \mathbb{R}^n;\ c \cdot x \leqslant \lfloor \beta \rfloor\}\,. \tag{7.17}$$

Cette proposition est illustrée sur la figure 7.2. Elle résulte du lemme suivant.

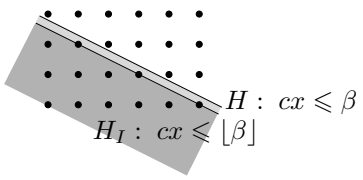


FIGURE 7.2 – Clôture entière d'un demi-espace

Lemme 7.14 *L'hyperplan $\bar{H} := \{x \in \mathbb{R}^n;\ c \cdot x = \lfloor \beta \rfloor\}$ est enveloppe convexe de points entiers de $\bar{H}$ (et donc de H).*

Supposons en effet ce lemme. Alors, $H_I \supset \mathrm{conv}(\bar{H} - \mathbb{N}d) = \bar{H} - \mathbb{R}_+ d = \{x \in \mathbb{R}^n;\ c \cdot x \leqslant \lfloor \beta \rfloor\}$, ce qui démontre la proposition, l'autre inclusion étant triviale. Nous démontrons maintenant le lemme.

Démonstration. L'espace affine $\bar{H}$ s'écrit $\bar{H} = \lfloor \beta \rfloor d + \bar{H}^{\mathrm{vect}}$ avec $\bar{H} := \{x \in \mathbb{R}^n;\ c \cdot x = 0\}$. Comme le vecteur $\lfloor \beta \rfloor d$ est entier, il suffit de traiter le cas où $\beta = 0$, de sorte que $\bar{H} = \bar{H}^{\mathrm{vect}}$. Considérons alors le $\mathbb{Q}$-espace vectoriel de dimension $n-1$, $\bar{H}_{\mathbb{Q}} = \{x \in \mathbb{Q}^n;\ c \cdot x = 0\}$. Celui-ci admet une base formée de $n-1$ vecteurs rationnels $y^1, \ldots, y^{n-1}$. Quitte à multiplier chacun de ces vecteurs par une constante entière, on peut supposer que $y^1, \ldots, y^{n-1}$ sont entiers. Par définition d'une base, les vecteurs $y^1, \ldots, y^{n-1}$ sont linéairement indépendants sur $\mathbb{Q}$, et donc linéairement indépendants sur $\mathbb{R}$ (l'indépendance linéaire sur $\mathbb{Q}$ ou $\mathbb{R}$ est caractérisée par la non-nullité d'un mineur maximal de la matrice $[y^1, \ldots, y^{n-1}]$). Comme $\bar{H}$ est un $\mathbb{R}$-espace vectoriel de dimension $n-1$, $y^1, \ldots, y^{n-1}$ est une base de $\bar{H}$. Donc tout vecteur x de $\bar{H}$ peut s'écrire comme combinaison linéaire réelle positive des vecteurs $\pm y^1, \ldots, \pm y^{n-1}$,

$$x = \sum_i \lambda_i(\epsilon_i y^i) \qquad \lambda_i \geqslant 0, \qquad \epsilon_i = \pm 1$$

En choisissant $\mu \in \mathbb{N}$ tel que $\mu \geqslant \sum_i \lambda_i$, on a bien que

$$x = \sum_i \frac{\lambda_i}{\mu} \mu(\epsilon_i y^i) + (1 - \sum_i \frac{\lambda_i}{\mu})0$$

est combinaison convexe de vecteurs entiers de $\bar{H}$. ■

Une coupe de Chvátal consiste donc à former une combinaison linéaire positive des inégalités (vérifiant la condition de coefficients de c entiers et premiers entre eux) puis à tronquer le second membre.

Indiquons le lien entre les coupes de Chvátal et de Gomory. Si $P \subset H$, et si les vecteurs de P ont tous des composantes positives, alors P est aussi contenu dans le demi espace $H' := \{x;\ \sum_{i=1}^n \lfloor c_i \rfloor x_i \leqslant \beta\}$. Les coupes de Chvátal incluent donc dans ce cas les inégalités du type

$$\sum_{i=1}^n \lfloor c_i \rfloor x_i \leqslant \lfloor \beta \rfloor. \tag{7.18}$$

Proposition 7.15 *La coupe de Gomory appliquée à la ième ligne coïncide, sur le polyèdre $P = \{x \geqslant 0;\ Ax = b\}$, avec la coupe de Chvátal de poids $\mu = A_B^{-\top} e_i$ appliquée à l'inégalité $Ax \leqslant b$.*

Démonstration. Appliquant la procédure d'arrondi à l'inégalité $a^\mu \cdot x \leqslant b^\mu$, il vient

$$\lfloor a^\mu \rfloor \cdot x \leqslant \lfloor b^\mu \rfloor. \tag{7.19}$$

Or si $x \in P$, on a $a^\mu \cdot x = b^\mu$. Soustrayant cette égalité de (7.19), on obtient (7.9). ■

Remarque 7.16 La coupe de Gomory (7.9) est donc formellement différente de la coupe de Chvátal (7.19), mais les deux coïncident au sens où leur intersection avec P est identique. •

Remarque 7.17 Nous prouverons en section 7.5.5 que la conjonction de toutes les coupes de Chvátal est un procédé qui, répété un nombre fini de fois, permet d'obtenir des inégalités caractérisant le polyèdre entier. Ce remarquable résultat théorique suggère l'utilité de ces coupes, ce qui est confirmé par les bonnes performances des coupes de Gomory. •

Exercice 7.18 (Coupes de Chvátal pour le sac à dos) Soit la contrainte de sac à dos $x \in S$, avec (a_i positifs)

$$S := \left\{ x \in \{0,1\}^n;\quad \sum_{i=1}^n a_i x_i \leqslant b \right\} \tag{7.20}$$

1. Soit $C \subset \{1, \ldots, n\}$ un ensemble dépendant ($\sum_{i \in C} a_i > b$) minimal. Montrer que

$$\sum_{i \in E(C)} x_i \leqslant |C| - 1, \tag{7.21}$$

où $E(C) = C \cup \{j; a_j \geqslant \max_{i \in C} a_i\}$.

2. Peut-on obtenir ce type de relation comme inégalité de Chvátal ?

3. Expliciter P_I, en se basant sur les questions précédentes, dans les deux cas suivants :

$$P = \{x \in \mathbb{R}^2_+;\ 5x_1 + 6x_2 \leqslant 7\}. \tag{7.22}$$
$$P = \{x \in \mathbb{R}^2_+;\ 5x_1 + 6x_2 + 7x_3 \leqslant 8\}. \tag{7.23}$$

4. Soit $P = \{x \in \mathbb{R}^n_+;\ \sum_{i=1}^n i x_i \leqslant n\}$. Formuler des inégalités satisfaites sur P_I. On se limitera au cas d'ensembles dépendants de cardinal 2 et 3.

Exercice 7.19 (Inégalités booléennes) Soit la contrainte

$$x \in \{0,1\}^{n+1};\quad \sum_{i=1}^n x_i \leqslant n x_{n+1}. \tag{7.24}$$

1. Montrer (par un raisonnement direct) que les relations précédentes sont équivalentes à

$$x \in \{0,1\}^{n+1}; \quad x_k \leqslant x_{n+1}, \quad k = 1, \ldots, n. \tag{7.25}$$

2. Obtenir (7.25) en s'appuyant sur les inégalités de Chvátal.

7.2 Coupes mixtes

Beaucoup de problèmes comportent des contraintes d'intégrité pour seulement une partie des variables. Supposons par exemple le domaine admissible du type

$$E := \{(x,y) \in \mathbb{N}^n \times \mathbb{R}^m_+; \;\; Ax + Gy \leqslant b\}. \tag{7.26}$$

On est donc amené à étudier des coupes d'intégrité partielle, dites coupes mixtes, qui séparent $\mathrm{conv}(E)$ d'un point n'appartenant pas à $\mathrm{conv}(E)$.

7.2.1 Principe des coupes disjonctives

Commençons par une propriété élémentaire qui est à la base de l'idée de coupe par disjonction.

Lemme 7.20 *Soient S_1 et S_2 des parties quelconques de $\mathbb{R}^n_+$, telles que $\sum_{i=1}^n c_{i,k} x_i \geqslant \gamma_k$ quand $x \in S_k$, pour $k = 1, 2$. Alors*

$$\sum_{i=1}^n \max(c_{i,1}, c_{i,2}) x_i \geqslant \min(\gamma_1, \gamma_2), \qquad \textit{pour tout } x \in S_1 \cup S_2. \tag{7.27}$$

Appliquons ce principe au calcul de coupes pour les contraintes mixtes.

7.2.2 Coupe de Gomory mixte

Nous allons présenter la technique des *coupes de Gomory mixtes*.

Théorème 7.21 *Tout point $(x,y) \in \mathbb{N}^n \times \mathbb{R}^m_+$, solution de l'inégalité*

$$\sum_{i=1}^n a_i x_i + \sum_{j=1}^m g_j y_j = \beta \tag{7.28}$$

avec β non entier, vérifie également la relation

$$\sum_{f_i \leqslant f_0} \frac{f_i}{f_0} x_i + \sum_{f_i > f_0} \frac{1-f_i}{1-f_0} x_i + \sum_{g_j \geqslant 0} \frac{g_j}{f_0} y_j - \sum_{g_j < 0} \frac{g_j}{1-f_0} y_j \geqslant 1, \tag{7.29}$$

où $f_0 := \beta - \lfloor \beta \rfloor$ est la partie fractionnaire de β, et $f_i = a_i - \lfloor a_i \rfloor$ est celle de a_i, $i = 1, \cdots, n$.

Démonstration. On a, pour un certain $k \in \mathbb{Z}$:

$$\sum_{f_i \leqslant f_0} f_i x_i + \sum_{f_i > f_0} (f_i - 1) x_i + \sum_{j=1}^m g_j y_j = k + f_0. \tag{7.30}$$

Suivant que $k \geqslant 0$ ou $k \leqslant -1$, la première ou la seconde inégalité ci-dessous sera satisfaite :

$$\begin{aligned} \sum_{f_i \leqslant f_0} \frac{f_i}{f_0} x_i + \sum_{f_i > f_0} \frac{f_i - 1}{f_0} x_i + \sum_{j=1}^m \frac{g_j}{f_0} y_j &\geqslant 1, \\ \sum_{f_i \leqslant f_0} \frac{f_i}{f_0 - 1} x_i + \sum_{f_i > f_0} \frac{f_i - 1}{f_0 - 1} x_i + \sum_{j=1}^m \frac{g_j}{f_0 - 1} y_j &\geqslant 1. \end{aligned} \tag{7.31}$$

On applique alors le lemme 7.20 (notons que les coefficients de la même composante de x ou y sont de signe opposé) ce qui permet de conclure. ■.

Remarque 7.22 En cas d'absence de variables continues, la coupe de Gomory mixte se réduit à

$$\sum_{f_i \leqslant f_0} \frac{f_i}{f_0} x_i + \sum_{f_i > f_0} \frac{1-f_i}{1-f_0} x_i \geqslant 1. \tag{7.32}$$

Comme $(1-f_i)/(1-f_0) < f_i/f_0$ si $f_i > f_0$, cette inégalité implique $\sum_i f_i x_i \geqslant f_0$ qui n'est autre que la coupe fractionnaire de Gomory. •

Remarque 7.23 Comme la coupe fractionnaire de Gomory, la coupe mixte de Gomory sera typiquement appliquée à une solution du problème relaxé continu, ayant une solution de base $(\bar{x}, \bar{y})$ avec une coordonnée i_0 du vecteur $\bar{x}$ non entière. Après introduction de variables d'écart (continues et positives) et produit par l'inverse de la matrice de base, on se ramène à une égalité du type

$$x_{i_0} + \sum_{i \in N_x} a_i x_i + \sum_{j \in M_y} g_j y_j = \beta, \tag{7.33}$$

où N_x et M_y désignent les indices hors base des variables x et y. Il en résulte la coupe

$$\sum_{i \in N_x; f_i \leqslant f_0} \frac{f_i}{f_0} x_i + \sum_{i \in N_x; f_i > f_0} \frac{1-f_i}{1-f_0} x_i + \sum_{j \in M_y; g_j \geqslant 0} \frac{g_i}{f_0} y_j - \sum_{j \in M_y; g_j < 0} \frac{g_i}{1-f_0} y_j \geqslant 1. \tag{7.34}$$

La coupe de Gomory mixte exclut bien le point courant, dont les coordonnées hors base sont nulles. •

Exemple 7.24 Soit le problème de sac à dos mixte

$$\text{Max } 2x + y; \quad x + y \leqslant 1.5;\ x \in \mathbb{N},\ y \in \mathbb{R}_+. \tag{7.35}$$

La solution du relaxé continu est $\bar{x} = 1.5$, $\bar{y} = 0$. Introduisant la variable d'écart $z \geqslant 0$, on écrit la contrainte sous la forme $x + y + z = 1.5$. Appliquant la relation (7.29) à la contrainte, avec ici $f_0 = \frac{1}{2}$, on obtient la coupe mixte $2y + 2z \geqslant 1$. La nouvelle relaxation continue a alors pour solution $\tilde{x} = 1$, $\tilde{y} = 0.5$ qui est la solution cherchée puisque $\tilde{x}$ est entier.

7.3 Coupes spécifiques

De nombreuses coupes ont été développées pour des structures spécifiques de contraintes. Nous ne donnons qu'une brève introduction à ce sujet.

7.3.1 Ensembles stables

Soit un graphe (non orienté) $G = (V, E)$. Un ensemble de sommets est dit *indépendant* ou *stable* s'il ne contient pas de sommets adjacents. À tout $W \subset V$ associons le vecteur indicateur $x \in \mathbb{R}^{|V|}$ tel que $x_i = 1$ si $i \in W$, et 0 sinon. Alors W est un ensemble stable ssi

$$x_i + x_j \leqslant 1, \quad \text{pour tout } (i,j) \in E. \tag{7.36}$$

On peut expliciter certaines coupes, et les relier aux coupes de Chvátal.

Exemple 7.25 Soit $C \subset G$ un cycle de longueur m. Additionnant les contraintes (7.36) le long du cycle il vient $2\sum_{i \in C} x_i \leqslant m$, et donc $\sum_{i \in C} x_i \leqslant \lfloor m/2 \rfloor$. Si m est pair ceci n'apporte rien de nouveau, mais si m est impair on obtient une nouvelle inégalité (de type Chvátal).

Exemple 7.26 Soit $C \subset G$ une *clique* (sous graphe avec des arêtes entre deux sommets quelconques) avec m sommets. Un au plus des sommets appartient à un ensemble stable, soit $\sum_{i\in C} x_i \leqslant 1$. Comment générer cette inégalité par les coupes de Chvátal ?

Procédons par récurrence. Pour $|C| = m = 3$ c'est vrai d'après la coupe de cycle. Supposons $m > 3$, et la coupe générée pour la clique $C' := C\backslash\{k\}$, où $k \in C$. Additionnons $(|C|-2)$ fois l'inégalité $\sum_{i\in C'} x_i \leqslant 1$, et les inégalités $x_i + x_k \leqslant 1$, pour $i \in C'$. Il vient $(|C|-1)(\sum_{i\in C} x_i) \leqslant 2|C|-3$, donc $\sum_{i\in C} x_i \leqslant \lfloor 2-1/(|C|-1)\rfloor = 1$ comme désiré.

7.3.2 Sac à dos

Construisons certaines coupes spécifiques aux problèmes de sac à dos. Elles sont d'une grande utilité, car on peut les appliquer à chaque contrainte scalaire d'un programme linéaire en variables $\{0,1\}$. La contrainte de sac à dos s'écrit

$$S = \left\{ x \in \{0,1\}^n; \quad \sum_{i=1}^{n} a_j x_i \leqslant b \right\}, \tag{7.37}$$

où $a \in \mathbb{N}^n$ et $b \in \mathbb{N}$. On peut supposer que $a_i \leqslant b$, car dans le cas contraire nécessairement $x_i = 0$ et on peut donc éliminer x_i du modèle.

On dit que $C \subset \{1,\dots,n\}$ est un ensemble *dépendant* si $\sum_{i\in C} a_i > b$. Ceci implique qu'au moins un des x_i, $i \in C$ est nul, soit :

$$\sum_{i\in C} x_i \leqslant |C| - 1. \tag{7.38}$$

Un ensemble dépendant est dit *minimal* s'il ne contient pas strictement un autre ensemble dépendant. Si C est dépendant minimal, on appelle *extension* de C l'ensemble

$$E(C) := C \cup \left\{ k;\ a_k \geqslant \max_{i\in C} a_i \right\}. \tag{7.39}$$

Si C est dépendant minimal, on peut mettre au plus $(|C|-1)$ éléments de $E(C)$ dans le sac à dos. C'est la *coupe de couverture* :

$$\sum_{i\in E(C)} x_i \leqslant |C| - 1. \tag{7.40}$$

Exemple 7.27 Dans l'exemple 7.5, un ensemble minimal suggéré par la relaxation continue est $\{1,2\}$ et l'ajout de la coupe $x_1 + x_2 \leqslant 1$ au problème relaxé permet d'obtenir la solution discrète recherchée.

7.4 Coupes d'optimalité

Reprenons les problèmes $(PLNE)$ et (PL) définis au début de ce chapitre. Soit $\bar{x}$ solution du relaxé continu (PL) associé à une partition (B,N), et notons $\underline{v} := c\cdot\bar{x}$ la valeur du problème (PL).

Partitionnant x en variable de base et hors base, on peut réécrire le problème relaxé continu sous la forme

$$\operatorname*{Min}_x \underline{v} + g\cdot x_N; \quad x_N \geqslant 0; \quad A_B^{-1}(b - A_N x_N) \geqslant 0, \tag{$PR2$}$$

où le gradient réduit a pour expression $g := c_N + A_N^\top \lambda$, avec $\lambda = -A_B^{-\top} c_B$, et vérifie $g \geqslant 0$ (condition d'optimalité du problème continu).

Supposons connu un majorant $\bar{v}$ de la valeur de $(PLNE)$ (typiquement un coût associé à un point réalisable obtenu par une heuristique). Posons $v^* := \bar{v} - \underline{v}$. On obtient

$$x \in S(PLNE) \quad \text{implique} \quad g\cdot x_N \leqslant v^*. \tag{7.41}$$

En particulier, puisque $g \geqslant 0$, ceci implique $g_i x_i \leqslant v^*$, pour tout $i \in N$. Puisqu'on cherche une solution entière, on déduit que

$$x_i \leqslant \lfloor v^*/g_i \rfloor, \quad \text{pour tout } i \in N \quad \text{tel que } g_i > 0. \tag{7.42}$$

Remarque 7.28 La relation ci-dessus permet de réduire le domaine à explorer (ce qui est précieux dans les méthodes de séparation et évaluation). Si $i \in N$ est tel que $v^* < g_i$, on déduit que $x_i = 0$ est satisfait à l'optimum. On peut donc éliminer la variable x_i du problème. •

Remarque 7.29 Plus généralement, pour tout $\alpha > 0$, on peut ajouter la coupe

$$\sum_{i \in N} \lfloor \alpha g_i \rfloor x_i \leqslant \lfloor \alpha v^* \rfloor. \tag{7.43}$$

Choisissant $\alpha = 1/g_i$ (pour $i \in N$ tel que $g_i > 0$) on obtient la coupe plus forte que (7.42) (mais moins maniable !)

$$x_i + \sum_{j \in N \setminus \{i\}} \lfloor g_j/g_i \rfloor x_j \leqslant \lfloor v^*/g_i \rfloor. \tag{7.44}$$

•

7.5 Compléments sur la théorie des coupes

7.5.1 Polyèdres entiers

Nous réunissons dans cette section quelques résultats utiles pour l'étude des polyèdres entiers. Soient un polyèdre et sa clôture entière

$$P := \{x \in \mathbb{R}^n;\ Ax \leqslant b\}; \quad P_I := \mathrm{conv}\{x \in \mathbb{Z}^n;\ Ax \leqslant b\}. \tag{7.45}$$

On dit que le polyèdre P est *entier* si $P = P_I$. Rappelons quelques définitions : l'*espace de linéalité* $\mathrm{lin}(P)$ est le plus grand espace vectoriel inclus dans $P - x^0$, où $x^0 \in P$ (définition indépendante de x^0), le *cône asymptote* de P est $\{x \in \mathbb{R}^n;\ Ax \leqslant 0\}$, et un cône est dit *pointé* s'il ne contient pas de droite. On dira qu'un polyèdre est rationnel s'il admet une paramétrisation de la forme (7.45) avec A et b à coefficients rationnels.

Proposition 7.30 *Soit P un polyèdre rationnel. Si P_I n'est pas vide, c'est un polyèdre rationnel, de même cône asymptote.*

Démonstration. a) Montrons qu'un cône polyédral rationnel $C = \{x \in \mathbb{R}^n;\ Ax \leqslant 0\}$ a un générateur entier, en commençant par le cas où C est pointé. Nous avons vu dans la démonstration du lemme 2.18 que C est engendré par des directions caractérisées par des ensembles maximaux d'indices actifs. Les contraintes correspondantes sont donc nécessairement de la forme $\hat{A}x = 0$, où $\hat{A}$ est de rang $n-1$ (puisque son noyau est de dimension 1). Eliminant si nécessaire les conditions redondantes, on peut supposer que $\hat{A}$ a $n-1$ lignes. La direction est non nulle, donc vérifie par exemple $x_1 > 0$. Le système linéaire $\{\hat{A}x = 0;\ x_1 = 1\}$ a une solution unique, qui est rationnelle. La multipliant par un rationnel convenable on obtient une direction extrême entière. Un cône pointé a donc un générateur entier.
b) Traitons maintenant le cas où C n'est pas pointé. On sait que $\mathrm{lin}(P) = \mathrm{lin}(C) = \mathrm{Ker}\, A$, et que $C = C' + \mathrm{lin}(C)$, où $C' := C \cap (\mathrm{lin}(C)^\perp)$ est pointé. Or $\mathrm{Ker}\, A$ admet un générateur $\{g_1, \ldots, g_q\}$ de coordonnées rationnelles [1] , qui induit la paramétrisation entière de son orthogonal : $(\mathrm{Ker}\, A)^\perp = \{x \in \mathbb{R}^n;\ g_i \cdot x = 0,\ i = 1, \ldots, q\}$, donc aussi de C'. D'après le point a, C' a donc un générateur entier, et il en est de même pour $C = C' + \mathrm{Ker}\, A$.
c) D'après le lemme 2.20, on a $P = Q + C$, où Q est un polytope et C est le cône asymptote. Soit $(y^1, \ldots, y^s)$ un générateur entier de C. D'après le lemme 2.23, l'ensemble suivant est un polytope :

$$B := \left\{ \sum_{i=1}^{s} \beta_i y^i;\quad \beta \in [0,1]^s \right\}. \tag{7.46}$$

1. Il suffit de raisonner sur p lignes de A indépendantes. Soit $\hat{A}$ la matrice correspondante, B un ensemble de colonnes indépendantes de $\hat{A}$ de cardinal p, et $N := \{1, \ldots, n\} \setminus B$. Un générateur de $\mathrm{Ker}\, A$ est donné par les $|N|$ vecteurs de la forme $x_B = -\hat{A}_B^{-1}\hat{A}_i$, $x_N = e_i$ pour $i \in N$, avec e_i $i$ème vecteur de la base naturelle de $\mathbb{R}^n$ et $\hat{A}_i$ $i$ème colonne de $\hat{A}$. L'inverse d'une matrice étant une fonction rationnelle de ses coefficients, ce générateur est bien rationnel.

Nous allons montrer que

$$P_I = (Q + B)_I + C. \tag{7.47}$$

Comme Q et B sont bornés, il en est de même de $Q + B$, donc $(Q + B)_I$ est un polytope ; la conclusion s'obtient alors en combinant (7.47) avec le lemme 2.23.

Pour démontrer (7.47), notons d'abord que

$$(Q + B)_I + C \subset P_I + C = P_I + C_I \subset (P + C)_I = P_I. \tag{7.48}$$

Réciproquement, il suffit de montrer qu'un élément p entier de P est inclus dans $(Q+B)_I + C$. Or $p = q + c$, avec $q \in Q$ et $c \in C$. Il existe donc $\beta \in \mathbb{R}^s_+$ tel que $c = \sum_{i=1}^s \beta_i y^i$. On peut écrire $c = b + c'$, avec $b \in B$ et c' vecteur entier de C (prendre $b = \sum_{i=1}^s (\beta_i - \lfloor \beta_i \rfloor) y^i$). Donc $p = (q + b) + c'$; comme $q + b = p - c'$ est entier, $q + b \in (Q + B)_I$, d'où $p \in (Q + B)_I + C$ comme il fallait le montrer. ■

Théorème 7.31 *Soit P un polyèdre rationnel, tel que P_I est non vide. Alors le problème* $\mathrm{Min}\{c \cdot x; x \in P_I\}$ *a un ensemble de solutions non vide ssi le problème relaxé* $\mathrm{Min}\{c \cdot x;\ x \in P\}$ *a une valeur finie.*

Démonstration. a) Si $\mathrm{Min}\{c \cdot x;\ x \in P\}$ a une valeur v finie, comme $\emptyset \notin P_I \subset P$, on a $v \leqslant \inf\{c \cdot x; \in P_I\} < +\infty$, donc $\mathrm{Min}\{c \cdot x; \in P_I\}$ a une valeur finie. Comme P_I est un polyèdre, ce programme linéaire à valeur finie a des solutions.
b) Si $\inf\{c \cdot x;\ x \in P\} = -\infty$, cela veut dire qu'il existe $\bar{d}$ dans le cône asymptotique $C = \{d \in \mathbb{R}^n;\ Ad \leqslant 0\}$ tel que $c \cdot \bar{d} < 0$. Ce point est de la forme $\sum_i \beta_j y^j$, où la somme est finie, les coefficients β_j sont positifs, et les y^j sont des directions extrêmes de C qu'on peut supposer entières. Approchant les coefficients β_j par des rationnels, on obtient l'existence de $\tilde{d} \in C$ rationnel, tel que $c \cdot \tilde{d} < 0$. Soit $x \in P_I$. Alors $x + t\tilde{d} \in P_I$ pour tout $t \geqslant 0$, et $c \cdot (x + t\tilde{d}) \downarrow -\infty$ quand $t \uparrow +\infty$. Ceci implique que $\mathrm{Min}\{c \cdot x; \in P_I\}$ a pour valeur $-\infty$, donc un ensemble de solutions vide. ■

7.5.2 Equations linéaires diophantiennes

Cette section discute le problème de calcul d'une solution entière (dans $\mathbb{Z}^n$) d'un système linéaire $Ax = b$, avec A de taille $p \times n$, $p \leqslant n$, A et b rationnels. Intéressant en soi, ce résultat joue aussi un rôle clé dans l'analyse des solutions de programmes linéaires en nombres entiers [2].

Définition 7.32 La matrice A, de taille $p \times n$, et de rang plein p (donc $p \leqslant n$) est dite sous *forme de Hermite* si elle est du type $[B\ 0]$, où B est carrée, semi triangulaire inférieure, inversible, et formée d'éléments positifs dont l'unique maximum sur chaque ligne est diagonal.

Théorème 7.33 *On peut ramener toute matrice de rang plein à la forme de Hermite par les "opérations élémentaires sur les colonnes" suivantes :* (i) *échange de deux colonnes,* (ii) *multiplication d'une colonne par -1,* (iii) *ajout d'un multiple entier d'une colonne à une autre.*

Démonstration. On peut sans perte de généralité supposer que les coefficients de la matrice sont entiers. Nous procédons par récurrence sur le nombre de lignes.
a) Si la matrice ne comporte qu'une seule ligne, on se ramène par (ii) au cas où tous les éléments sont positifs, puis avec (iii) par division euclidiennes successives par le plus petit élément, on réduit au cas d'un seul élément non nul, ramené en première position par (i), ce qui est la forme désirée.
b) Dans le cas général, supposons la forme normale obtenue pour les lignes 1 à $p - 1$. Appliquant l'étape précédente aux éléments p à n de la dernière ligne (non tous nuls puisque le rang de la matrice est p), on se ramène à la forme triangulaire avec $B_{pp} > 0$; retranchant un multiple convenable de la colonne p aux colonnes 1 à $p - 1$ si nécessaire (ce qui ne modifie pas les autres lignes), on obtient la forme désirée. ■

2. La raison essentielle est que si P est un polyèdre, dont les faces sont définies comme les ensembles de solutions de problèmes de maximisation de critères linéaires sur P, alors les faces de dimension minimale sont des sous espaces affines ; or le minimum d'un critère linéaire sur un polyèdre est atteint sur une face de dimension minimale. Voir la démonstration du lemme 7.37.

Proposition 7.34 *Soient $A \in \mathbb{Q}^{p\times n}$ et $b \in \mathbb{Q}^p$. Alors $Ax = b$ a des solutions entières ssi $b \cdot \lambda$ est entier, pour tout $\lambda \in \mathbb{Q}^p$ tel que $A^\top \lambda$ est entier.*

Démonstration. a) Si x est solution entière de $Ax = b$, et si $A^\top \lambda$ est entier, alors $b \cdot \lambda = (Ax) \cdot \lambda = x \cdot A^\top \lambda$ est entier. La condition est donc nécessaire.
b) Inversement, supposons la condition satisfaite. Vérifions dans un premier temps que $b \in \mathrm{Im}(A)$. Montrons d'abord que la projection b^1 de b sur $\mathrm{Im}(A)$ est rationnelle. Soient $\{A^i,\ i \in I\}$ une famille maximale de colonnes indépendantes de A, $\hat{A}$ la matrice dont les colonnes sont les A^i, pour $i \in I$, et $r := |I|$. Soit $y \in \mathbb{R}^r$ solution de $\mathrm{Min}\ \frac{1}{2}\|\hat{A}y - b\|^2$. Les solutions de ce problème convexe sont caractérisées par la condition d'optimalité $\hat{A}^\top \hat{A}y = \hat{A}^\top b$. Comme $\hat{A}^\top \hat{A}y$ est inversible (conséquence de l'injectivité de $\hat{A}$), la solution unique est $y = (\hat{A}^\top \hat{A})^{-1}\hat{A}^\top b$; elle est rationnelle, de même que $b^1 = \hat{A}y$.

Posons $\eta := b - b^1$. Comme $\hat{A}^\top \eta = 0$, on a $\eta \cdot b^1 = 0$. Pour $\alpha > 0$ assez petit, $\lambda := \alpha\eta$ est tel que $A^\top \lambda = 0$ et $\lambda \cdot b = \lambda \cdot (b - b^1) = \alpha\|\eta\|^2 \in]0, 1[$ (norme euclidienne), en contradiction avec l'hypothèse si $b \neq b^1$; donc $b \in \mathrm{Im}(A)$.
c) Soit I un ensemble de ligne indépendantes de A. Par hypothèse, $b \cdot \lambda$ est entier, pour tout $\lambda \in \mathbb{Q}^p$ tel que $A^\top \lambda$ est entier, vérifiant de plus $\lambda_i = 0$, pour tout $i \notin I$. Cette propriété s'interprète comme l'hypothèse de la proposition, pour le système réduit aux ligne de I. Il suffit d'obtenir la conclusion dans ce cas (les autres lignes sont compatibles, car $b \in \mathrm{Im}(A)$), ce qui revient à supposer A de rang plein.
d) L'hypothèse de la proposition est invariante sous les opérations élémentaires du théorème 7.33. On peut donc supposer A (de rang plein) sous forme de Hermite ; nous allons vérifier que $x := B^{-1}b$ est entier. En effet, pour $i = 1$ à n, $x_i = \lambda^i \cdot b$, où λ^i est la transposée de la $i$ème ligne de B^{-1}. Comme $B^{-1}A = [I\ 0]$ est entière, l'hypothèse assure que x est entier comme il fallait le montrer. ■

Remarque 7.35 Le point (d) de la démonstration précédente montre que le calcul de la forme normale de Hermite permet un calcul effectif d'une solution entière (ou permet de prouver qu'il n'existe pas de telles solutions). •

7.5.3 Caractérisation des polyèdres entiers

Nous avons établi en section 2.3 l'intégrité des points extrêmes de polyèdres de la forme $P = \{x \in \mathbb{R}^n;\ h' \leqslant Ax \leqslant h;\ g' \leqslant x \leqslant g\}$ où A est totalement unimodulaire et les bornes sont entières (ou infinies). Par des variantes simples des démonstrations, on prouverait de même que $P_I = P$, et aussi que pour tout c entier, le dual du problème $\mathrm{Min}\{c \cdot x;\ \ h' \leqslant Ax \leqslant h;\ g' \leqslant x \leqslant g\}$ a, si c est entier, et si l'ensemble de linéalité du problème dual est réduit à 0, des solutions entières (en effet la dualité échange coûts et bornes, et transpose les matrices, et la transposée d'une matrice totalement unimodulaire l'est aussi). Nous étudions maintenant le cas de matrices quelconques.

Définition 7.36 Soit P un polyèdre. Alors :
(i) Soit F une face de P. On note I_F l'*ensemble des indices des contraintes saturées en tout* $x \in F$, défini par $I_F = \{1 \leqslant i \leqslant p;\ (Ax)_i = 0,\ \text{pour tout } x \in F\}$.
(ii) On dit qu'un hyperplan $H = \{x \in \mathbb{R}^n;\ c \cdot x = \beta\}$ est un *hyperplan d'appui* si $H \cap P \neq \emptyset$ et soit $P \subset \{x \in \mathbb{R}^n;\ c \cdot x \leqslant \beta\}$, soit $P \subset \{x \in \mathbb{R}^n;\ c \cdot x \geqslant \beta\}$.

Lemme 7.37 *Un polyèdre rationnel P est entier ssi chaque hyperplan d'appui, à coefficients rationnels, contient des vecteurs entiers.*

Démonstration. L'intersection d'un hyperplan d'appui $\{x \in \mathbb{R}^n;\ c \cdot x = \beta\}$, où $c \neq 0$ et β sont rationnels, et du polyèdre est une face F, sur laquelle $\pm c \cdot x$ atteint son maximum sur P. Si P est entier, ce maximum est également atteint sur un point entier de $F \cap P$, ce qui montre que la condition est nécessaire.

Montrons qu'elle est suffisante. Comme chaque face contient une face minimale [3] il suffit de montrer que si tout hyperplan d'appui d'une face minimale à coefficients rationnels contient des vecteurs entiers, alors P

3. Ces faces minimales, si P ne contient pas de droites, sont les points extrêmes du polyèdre. Si $\mathrm{lin}(P) \neq \{0\}$, les faces minimales sont de la forme $x + \mathrm{lin}(P)$, où x parcourt l'ensemble des points extrêmes de $P \cap (\mathrm{lin}(P)^\perp)$.

est entier. Soit F une face minimale de P. Alors $F = \{x \in \mathbb{R}^n;\ (Ax)_i = b_i,\ i \in I_F\}$ (l'inclusion est évidente, et si l'inclusion inverse n'était pas satisfaite, on pourrait construire une face plus petite en intersectant avec une ou plusieurs ligne de A hors de I_F). On peut supposer A et b entiers (les multiplier par le PPCM des dénominateurs de leurs composantes). Si F ne contient pas de points entiers, d'après la proposition 7.34, il existe $\lambda \in \mathbb{Q}^p$, à support sur I_F, tel que $c := A^\top\lambda$ est entier, et que $\beta := b \cdot \lambda$ ne l'est pas. Changeant si nécessaire λ_i en $\lambda_i + q$, avec $q \in \mathbb{N}$ assez grand, on peut supposer que $\lambda_i > 0$, pour tout $i \in I_F$ (le caractère entier de $A^\top\lambda$ et non entier de $b \cdot \lambda$ est invariant par cette transformation). Posons $H := \{x;\ c \cdot x = \beta\}$, et montrons que $F = P \cap H$. En effet, les conditions d'optimalité du problème $\mathrm{Min}_x\{-c \cdot x;\ Ax \leqslant b\}$ sont vérifiées en F et non hors de F, comme on le vérifie facilement grâce à la positivité stricte des λ_i, pour $i \in I_F$. Donc H est un hyperplan d'appui de P. Puisque β n'est pas entier, H ne contient pas de vecteurs entiers, en contradiction avec l'hypothèse. ■

Corollaire 7.38 *Soit $P = \{x;\ Ax \leqslant b\}$ un polyèdre rationnel. Alors* $\min\{c \cdot x;\ x \in P\}$ *a une solution entière, pour tout c pour lequel le minimum est fini, ssi ce minimum est entier pour c entier.*

Démonstration. La condition est évidemment nécessaire. Montrons qu'elle est suffisante. Soit $H := \{x;\ c \cdot x = \beta\}$ un hyperplan d'appui de P, avec c entier et $P \subset \{x;\ c \cdot x \geqslant \beta\}$. Alors $\beta = \min\{c \cdot x;\ Ax \leqslant b\}$ est entier par hypothèse. D'après le lemme 7.37 (dans lequel il est équivalent d'imposer c entier) il suffit de vérifier que cet hyperplan contient des points entiers. D'après l'identité de Bezout (lemme 7.12), $\beta = \gamma\alpha$, où le PGCD α des composantes de c est tel que $\alpha = c \cdot d$, pour un certain $d \in \mathbb{Z}^n$; donc $\gamma d \in H$ ce qui permet de conclure. ■

7.5.4 Intégrité duale totale et base de Hilbert

La discussion des méthodes de coupe conduit à introduire un cas particulier important de paramétrage de polyèdre.

Définition 7.39 On dit qu'un système rationnel $Ax \leqslant b$ est *totalement dual entier* (TDI) si, pour tout $c \in \mathbb{Z}^n$ tel que $\min\{c \cdot x;\ Ax \leqslant b\}$ est fini, le dual $\mathrm{Max}\{-b \cdot \lambda;\ c + A^\top\lambda = 0;\ \lambda \geqslant 0\}$ a une solution entière. Si de plus aucun sous système de $Ax \leqslant b$ (obtenu en supprimant certaines lignes) n'est TDI, on dit que $Ax \leqslant b$ est *TDI minimum.*

Nous avons vu que, si le polyèdre $P = \{x \in \mathbb{R}^n;\ Ax \leqslant b\}$ admet des points extrémaux, si A est totalement unimodulaire, et si b est entier, alors $Ax \leqslant b$ est TDI.

Corollaire 7.40 *Soit $Ax \leqslant b$ un système TDI. Si b est entier, et si c entier est tel que le problème* $\mathrm{Min}_x\{c \cdot x;\ Ax \leqslant b\}$ *a une valeur finie, alors il existe une solution entière. Autrement dit, $P = \{x;\ Ax \leqslant b\}$ est entier.*

Démonstration. Puisque (A, b) est TDI, le dual a, pour tout c entier tel que la valeur est finie, une solution entière, donc la valeur est entière. On obtient avec le corollaire 7.38 l'existence d'une solution entière. ■

L'exemple suivant montre que la TDI est une propriété de la paramétrisation du polyèdre et non du polyèdre lui-même.

Exemple 7.41 Les systèmes suivants, dans lesquels $x \in \mathbb{R}^2$:

$$\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} x \leqslant 0; \quad \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & -1 \end{pmatrix} x \leqslant 0; \tag{7.49}$$

représentent le même polyèdre, qui est un cône ; les seules valeurs finies de minimum de critères linéaires sur cet ensemble sont donc nulles. Montrons que le second est TDI et non le premier. En effet, dans le premier cas, les conditions duales

$$c_1 + \lambda_1 + \lambda_2 = 0; \quad c_2 + \lambda_1 - \lambda_2 = 0; \quad \lambda \geqslant 0 \tag{7.50}$$

donnent $\lambda_1 = -\frac{1}{2}(c_1+c_2)$, $\lambda_2 = \frac{1}{2}(c_2-c_1)$. Pour $c = -(1,2)^\top$ ceci a une seule solution positive, non entière. Ces calculs nous indiquent aussi que la valeur du problème de minimisation de $c \cdot x$ sous la contrainte (7.49) est nulle ssi $c_1 + c_2 \leqslant 0$ et $c_2 \geqslant c_1$. Pour la seconde paramétrisation,

$$c_1 + \lambda_1 + \lambda_2 + \lambda_3 = 0; \quad c_2 + \lambda_1 - \lambda_3 = 0; \quad \lambda \geqslant 0 \tag{7.51}$$

a, pour c entier tel que $c_1 + c_2 \leqslant 0$ et $c_2 \geqslant c_1$, la solution $\lambda_1 = -\frac{1}{2}(c_1 + \lambda_2 + c_2)$, $\lambda_3 = \frac{1}{2}(c_2 - c_1 - \lambda_2)$. Si $c_2 \leqslant 0$ on peut prendre $\lambda = (-c_2, c_2 - c_1, 0)$ et sinon $\lambda = (0, -c_2 - c_1, -2c_1)$.

Définition 7.42 On dit que la famille $\{x^1, \ldots, x^q\}$ de $\mathbb{R}^n$ est une *base de Hilbert* du cône $C \subset \mathbb{R}^n$, si tout élément entier de C est combinaison linéaire positive de $\{x^1, \ldots, x^q\}$ à coefficients entiers.

Lemme 7.43 *Un cône polyédral rationnel a une base de Hilbert.*

Démonstration. Soit C un cône polyédral ; on sait (lemme 2.18) qu'il a un générateur fini $\{y^1, \ldots, y^m\}$ rationnel, qu'on peut supposer entier. Soient

$$C_1 := \left\{ \sum_{i=1}^m \beta_i y^i; \quad \beta \in [0,1]^m \right\}; \quad C_2 := \left\{ \sum_{i=1}^m \gamma_i y^i; \quad \gamma \in \mathbb{N}^m \right\}. \tag{7.52}$$

Si $z \in C$, soit $z = \sum_{i=1}^m \alpha_i y^i$, avec $\alpha \in \mathbb{R}_+^m$, notant γ le vecteur des parties entières de α, et posant $z^2 := \sum_{i=1}^m \gamma_i y^i$ et $z^1 := z - z^2$, il vient $z^i \in C_i$, pour $i = 1, 2$, ce qui prouve que $C = C_1 + C_2$.

Il en résulte qu'une base de Hilbert de C est donnée par la réunion de $\{y^1, \ldots, y^m\}$ et de l'ensemble (fini) des éléments de C_1 à coordonnées entières. ■

Proposition 7.44 *Le système rationnel $Ax \leqslant b$ est TDI ssi, pour chaque face F du polyèdre $P := \{x;\ Ax \leqslant b\}$, les lignes de A saturées sur F forment une base de Hilbert (du cône convexe qu'elles engendrent).*

Démonstration. Si $Ax \leqslant b$ est TDI, et si F est une face de P, soit c un vecteur entier, combinaison linéaire positive de $\{a^i,\ i \in I_F\}$ de coefficients notés μ. Le programme linéaire $\mathrm{Min}\{-c \cdot x;\ Ax \leqslant b\}$ est tel que μ est réalisable pour le dual $\mathrm{Max}_\mu\{-\mu \cdot b;\ c = A^\top \mu,\ \mu \geqslant 0\}$ et réalise la condition d'écarts complémentaires avec tout point de F ; donc tout $x \in F$ est solution. Puisque $Ax \leqslant b$ est TDI, il existe une solution duale λ entière. Les composantes non nulles de λ correspondent à des contraintes actives de F, en raison de la complémentarité avec les points de F. On a donc $-c + \sum_{i \in I_F} \lambda_i a^i = 0$, ce qui montre que les $\{a^i,\ i \in I_F\}$ forment une base de Hilbert.

Réciproquement, soient c un vecteur entier tel que $\min\{-c \cdot x;\ Ax \leqslant b\}$ soit atteint, et F une face optimale. Alors c est combinaison linéaire positive des $\{a^i,\ i \in I_F\}$ et par définition d'une base de Hilbert on peut supposer ces coefficients entiers. Or le vecteur de ces coefficients est solution du problème dual, donc $Ax \leqslant b$ est TDI. ■

Théorème 7.45 *Tout polyèdre rationnel P a une paramétrisation TDI de la forme $Ax \leqslant b$, avec A entière. De plus on peut imposer b entier ssi P est entier.*

Démonstration. Pour toute face F de P, notons C_F le cône engendré par $\{a^i,\ i \in I_F\}$ (où a^i est la $i$ème ligne de A), et soit $y^1, \ldots, y^r$ une base de Hilbert de C_F. Etant donné $x^0 \in F$, posons $\beta_i := y^i \cdot x^0$. Alors $y^i \cdot x \leqslant \beta_i$ est réalisée sur P (chaque point de F réalise le maximum sur P de $y^i \cdot x$, car il vérifie les conditions d'optimalité de ce programme linéaire), et l'union de ces inégalités pour toutes les faces caractérise P. La paramétrisation ainsi obtenue est TDI d'après la proposition 7.44.

Si P est entier on peut remplacer b par $\lfloor b \rfloor$ et donc supposer b entier. Réciproquement, si b est entier, alors P l'est d'après le corollaire 7.40. ■

Le résultat suivant sera utilisé dans la section 7.5.5. Il exprime le fait que chaque face d'un système TDI est elle-même TDI.

Lemme 7.46 *Soit $(Ax \leqslant b;\, a\cdot x \leqslant \beta)$ un système TDI. Alors le système $(Ax \leqslant b;\, a\cdot x = \beta)$ est aussi TDI.*

Démonstration. Soient c entier et x^* solution de $\mathrm{Min}\{c\cdot x;\, Ax \leqslant b;\, a\cdot x = \beta\}$. Soit (λ^*, η^*) solution du problème dual $\mathrm{Max}\{-b\cdot\lambda - \beta\eta;\, \lambda \geqslant 0;\, c + A^\top\lambda + \eta a = 0\}$.

Posons $c' := c - Na$, où $N \in \mathbb{N}$, $N + \min(\eta^*) > 0$ est tel que Na est entier. Alors x^* est point réalisable de $\mathrm{Min}\{c'\cdot x;\, Ax \leqslant b;\, a\cdot x \leqslant \beta\}$, dont le dual $\mathrm{Max}\{-b\cdot\lambda' - \beta\eta';\, \lambda' \geqslant 0\, c' + A^\top\lambda' + \eta' a = 0\}$ a comme point réalisable $(\lambda^*, \eta^* + N\mathbf{1})$. Les conditions de complémentarité étant satisfaites, x^* et $(\lambda^*, \eta^* + N\mathbf{1})$ sont solutions primale et duale de ces derniers problèmes.

Comme $\{Ax \leqslant b;\, a\cdot x \leqslant \beta\}$ est TDI, le dual a une solution entière $(\tilde\lambda, \tilde\eta)$; alors $(\tilde\lambda, \tilde\eta - N\mathbf{1})$ est solution entière du premier dual (car c'est un point réalisable, complémentaire à x^*). La conclusion s'ensuit. ■

7.5.5 Finitude du rang de Chvátal

Soit $P = \{x;\, Ax \leqslant b\}$ un polyèdre rationnel, contenu dans le demi espace $H = \{x;\, c\cdot x \leqslant \beta\}$. Alors $\beta \geqslant \max\{c\cdot x;\ Ax \leqslant b\}$, et par dualité on vérifie que c est combinaison linéaire positive des lignes de A. Notons

$$P^{(1)} := \bigcap_{H\supset P} H_I \tag{7.53}$$

l'intersection des clôtures entières des demi-espaces le contenant. Noter que $P^{(1)}$ est indépendant du paramétrage de P. On a $P_I \subset P^{(1)} \subset P$. Itérant le procédé, on construit la *suite de Chvátal* définie par $P^0 := P$ et $P^{(i+1)} := (P^{(i)})^{(1)}$, pour tout $i \in \mathbb{N}$, qui vérifie

$$P_I \subset P^{(i+1)} \subset P^{(i)} \subset \cdots \subset P^{(1)} \subset P. \tag{7.54}$$

Montrons que la suite de Chvátal est composée de polyèdres, et est, à partir d'un certain rang, dit *rang de Chvátal*, égale à P_I.

Proposition 7.47 *Soit P un polyèdre rationnel, de paramétrisation TDI $Ax \leqslant b$, avec A entière. Alors $P^{(1)} = \{x;\, Ax \leqslant \lfloor b\rfloor\}$.*

Démonstration. Par construction, $P^{(1)} \subset \{x;\, Ax \leqslant \lfloor b\rfloor\}$. Montrons la réciproque. Soit $H = \{x;\, c\cdot x \leqslant \beta\}$ un demi espace rationnel contenant P. On peut supposer c entier et irréductible. Alors $H_I = \{x;\, c\cdot x \leqslant \lfloor\beta\rfloor\}$ d'après la proposition 7.13. Or le problème $\mathrm{Min}\{-c\cdot x;\, Ax \leqslant b\}$ a, puisque $Ax \leqslant b$ est TDI, une solution duale λ entière positive. Donc $c = A^\top\lambda$, et il existe $\bar x$ tel que $A\bar x \leqslant b$ et $\lambda\cdot b = \lambda\cdot A\bar x = c\cdot\bar x \leqslant \beta$. Pour tout x tel que $Ax \leqslant \lfloor b\rfloor$, on a donc (noter que $\lambda\cdot\lfloor b\rfloor$ est un entier minorant $\lambda\cdot b$, donc aussi $\lfloor\lambda\cdot b\rfloor$) :

$$c\cdot x = \lambda\cdot Ax \leqslant \lambda\cdot\lfloor b\rfloor \leqslant \lfloor\lambda\cdot b\rfloor = \lfloor c\cdot\bar x\rfloor \leqslant \lfloor\beta\rfloor. \tag{7.55}$$

Donc $\{x;\, Ax \leqslant \lfloor b\rfloor\} \subset H_I$, comme il fallait le montrer. ■

Lemme 7.48 *Soit F une face du polyèdre rationnel P, et $F^{(s)}$ l'élément de la suite de Chvátal pour F, d'indice $s \in \mathbb{N}$. Alors $F^{(s)} = P^{(s)} \cap F$.*

Démonstration. a) Procédons par récurrence. Puisque $P^0 = P$, le résultat est vrai pour $s = 0$. Supposons-le vrai pour $s \in \mathbb{N}$. Si $F^{(s)} = \emptyset$, alors $F^{(s+1)} = \emptyset$. Dans ce cas $P^{(s)} \cap F = \emptyset$; puisque $P^{(s+1)} \subset P^{(s)}$, on a $P^{(s+1)} \cap F = \emptyset$ et la conclusion est donc satisfaite.
b) Reste le cas où $F^{(s)} = P^{(s)} \cap F$ est non vide. Comme F est une face de P, et que $P^{(s)} \subset P$, cela implique que F est une face de $P^{(s)}$. Soient les paramétrisations $P^{(s)} = \{x;\, A^{(s)}x \leqslant b^{(s)}\}$, avec $A^{(s)}$ entière et $A^{(s)}x \leqslant b^{(s)}$ TDI, $F = \{x;\, Ax \leqslant b;\, a\cdot x = \beta\}$, avec $a\cdot x \leqslant \beta$ inégalité satisfaite sur P, a et β entiers. Par hypothèse, $\{A^{(s)}x \leqslant b^{(s)};\, a\cdot x \leqslant \beta\}$ est TDI. Le lemme 7.46 implique qu'il en est de même pour $\{A^{(s)}x \leqslant b^{(s)};\, a\cdot x = \beta\}$. Or β est entier, donc avec la proposition 7.47 :

$$\begin{aligned} P^{(s+1)} \cap F &= \{x;\, A^{(s)}x \leqslant \lfloor b^{(s)}\rfloor;\, a\cdot x = \beta\} \\ &= \{x;\, A^{(s)}x \leqslant \lfloor b^{(s)}\rfloor;\, a\cdot x \leqslant \lfloor\beta\rfloor;\, a\cdot x \geqslant \lceil\beta\rceil\} \\ &= F^{(s+1)}, \end{aligned} \tag{7.56}$$

d'où le résultat. ■

Théorème 7.49 *Soit P un polyèdre rationnel. Alors $P_I = P^{(k)}$ à partir d'un certain $k \in \mathbb{N}$, appelé rang de Chvátal de P.*

Démonstration. a) Procédons par récurrence sur la dimension d de P (la dimension de son enveloppe affine affhull(P)). Si $d = 0$ (P vide) on a $P_I = P^{(0)} = P$ donc le rang de Chvátal vaut 0. Si $d = 1$, P est réduit à un point x^0. Si x^0 est entier, alors $P_I = P^{(0)} = P$ (le rang vaut 0) ; sinon, la proposition 7.34 assure l'existence d'une égalité $c \cdot x = \beta$ satisfaite par x^0, avec c entier mais non β (il suffit de prendre $c \cdot x = x_i$ avec x_i^0 non entier). La coupe de Chvátal $c \cdot x \leqslant \lfloor \beta \rfloor$ assure que $P^{(1)} = \emptyset$ (le rang de Chvátal vaut 1).
b) Soit C entière de rang plein telle que affhull(P) $= \{x;\, Cx = d\}$. Si affhull(P) ne contient pas de points entiers, d'après la proposition 7.34, il existe un vecteur λ entier tel que $c := C^\top \lambda$ est entier, et $\gamma := d \cdot \lambda$ ne l'est pas. Donc

$$P^{(1)} \subset \{x;\, c \cdot x \leqslant \lfloor \gamma \rfloor;\, c \cdot x \geqslant \lceil \gamma \rceil\} = \emptyset. \tag{7.57}$$

Si affhull(P) contient des points entiers, par translation d'un de ces points à 0 (transformation qui laisse invariant le rang de Chvátal), on peut supposer que affhull(P) $= \{x;\, Cx = 0\}$. D'après le théorème 7.33, on peut ramener C à la forme normale de Hermite : $CU = [B\ 0]$, par les opérations élémentaires sur les colonnes décrites dans ce théorème. Comme l'hypothèse du théorème (matrice de rang plein) est invariante sous ces transformations élémentaires on peut supposer que $C = [B\ 0]$, donc affhull(P) $= \{0\}_{n-d} \times \mathbb{R}^d$ (où par $\{0\}_{n-d}$ on note le zéro de $\mathbb{R}^{n-d}$). Eliminant les $n - d$ premières variables qui ne jouent aucun rôle, on se ramène au cas où P est de dimension n.
d) Montrons qu'il existe une matrice entière A et des vecteurs rationnels b et b' tels que

$$\text{(i)}\ \ P = \{x \in \mathbb{R}^n;\ Ax \leqslant b\}; \quad \text{(ii)}\ \ P_I = \{x \in \mathbb{R}^n;\ Ax \leqslant b'\}. \tag{7.58}$$

En effet, comme P et P_I sont des polyèdres rationnels ils ont des paramétrisations entières de la forme

$$P = \{x \in \mathbb{R}^n;\ A'x \leqslant \hat{b}'\}; \quad P_I = \{x \in \mathbb{R}^n;\ A''x \leqslant \hat{b}''\}. \tag{7.59}$$

Posons

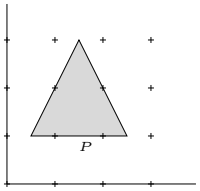
$$A := \begin{pmatrix} A' \\ A'' \end{pmatrix}; \quad b' := \begin{pmatrix} \hat{b}' \\ \hat{b}'' \end{pmatrix}; \tag{7.60}$$

Comme $P_I \subset P$, (7.58)(ii) est satisfaite. Choisissons les premières composantes de b égales à $\hat{b}'$, et pour les suivantes, prenons des constantes positives assez grandes. Justifions ceci en montrant que, pour une contrainte définissant P_I de la forme $c \cdot x \leqslant \beta$, on a $P \subset \{x; c \cdot x \leqslant \beta'\}$ pour un certain β'. Si ce n'est pas le cas, il existe une direction d du cône asymptote de P, telle que $c \cdot d > 0$. Comme P et P_I ont même cône asymptote, ceci est impossible.
e) Soit $H := \{x \in \mathbb{R}^n;\ a \cdot x \leqslant \beta'\}$, où $a \cdot x \leqslant \beta'$ est une des inégalités de $Ax \leqslant b'$. Il suffit de montrer que $P^{(k)} \subset H$ pour k assez grand. Soit $a \cdot x \leqslant \beta$ l'inégalité correspondante dans $Ax \leqslant b$. Si $P^{(k)} \not\subset H$, pour tout $k \in \mathbb{N}$, puisque $P^{(1)} \subset \{x \in \mathbb{R}^n;\ a \cdot x \leqslant \lfloor \beta \rfloor\}$, il existe $r \in \mathbb{N}$ et un entier β'' dans $]\beta', \lfloor \beta \rfloor]$ tels que, pour $k \geqslant r$,

$$P^{(k)} \subset \{x \in \mathbb{R}^n;\ a \cdot x \leqslant \beta''\}; \quad P^{(k)} \not\subset \{x \in \mathbb{R}^n;\ a \cdot x \leqslant \beta'' - 1\}. \tag{7.61}$$

On prend pour β'' le plus petit entier vérifiant ces relations. Posons $F := P^{(r)} \cap \{x \in \mathbb{R}^n;\ a \cdot x = \beta''\}$. Alors $\dim(F) < \dim(P)$ et, puisque $\beta' < \beta''$, F ne contient pas de vecteurs entiers. D'après l'hypothèse de récurrence, $F^{(s)} = \emptyset$ pour $s \in \mathbb{N}$ assez grand. Montrons que F est non vide. En effet soit $\delta := \sup\{a \cdot x;\ x \in P^{(r)}\}$. Si F est vide, alors $\delta < \beta''$ (car la valeur d'un programme linéaire est atteinte), et alors $P^{(r+1)} \subset \{x \in \mathbb{R}^n;\ a \cdot x \leqslant \beta'' - 1\}$, contrairement à l'hypothèse.
f) Puisque F est non vide, c'est une face de $P^{(r)}$. Le lemme 7.48 implique $\emptyset = F^{(s)} = P^{(r+s)} \cap F$. Comme $P^{(r+s)}$ est un polyèdre, il vérifie donc $P^{(r+s)} \subset \{x \in \mathbb{R}^n;\ a \cdot x < \beta'''\}$, avec $\beta''' < \beta''$, et de là $P^{(r+s+1)} \subset \{x \in \mathbb{R}^n;\ a \cdot x \leqslant \beta'' - 1\}$, en contradiction avec (7.61). ■

Exercice 7.50 (Rang de Chvátal) Soit le polyèdre P représenté en grisé :



Dessiner les polyèdres $P^{(1)}$ et $P^{(2)}$. En déduire que le rang de Chvátal de P est égal à 2.

Exercice 7.51 (Plusieurs coupes peuvent être nécessaires). Considérons le programme linéaire en nombres entiers :

$$\text{Max}\, x_2; \quad x \in P \cap \mathbb{Z}^2 \ ,$$

le polyèdre P étant défini par les inégalités

$$\begin{aligned} 0 &\leqslant x_1, x_2 \\ 2kx_1 + x_2 &\leqslant 2k \\ -2kx_1 + x_2 &\leqslant 0 \end{aligned}$$

avec k entier positif. 1) Comparer, en faisant un dessin, la valeur du relâché continu et celle du programme en nombres entiers. 2) Estimer par un raisonnement géométrique le rang de Chvátal de P (on pourra faire le dessin pour $k = 2$ et "généraliser" sans détailler la preuve). 3) Conclure qu'un nombre de coupes de Gomory d'au moins $2k$ est nécessaire pour obtenir l'optimum du problème en nombres entiers.

7.6 Notes

Les coupes de Gomory [Gom58], antérieures à celles de Dantzig [Dan59], restent jusqu'à présent les plus performantes. Les coupes de Chvátal sont introduites dans [Chv73]. On trouvera une vue d'ensemble récente sur les coupes dans Cornuéjols [Cor08]. Voir aussi les livres de Nemhauser et Wolsey [NW99] et Wolsey [Wol98], et pour les aspects théoriques Schrijver [Sch86]. Notons en particulier l'emploi de coupes spécifiques pour les contraintes de flot et de tour. Les coupes peuvent s'obtenir en utilisant des représentations dans des espaces de dimension supérieure : voir Sherali et Adams [SA94], Balas et Perregaard [BP03].

L'usage des seules coupes ne permet généralement pas de résoudre des problèmes de programmation linéaire en nombre entier en un temps raisonnable. En revanche leur combinaison avec les méthodes de séparation se montre efficace. L'approche "couper puis séparer" (cut and branch) consiste à ne pratiquer les coupes qu'avant tout branchement. L'approche "séparer puis couper" consiste à pratiquer les coupes en tout nœud de l'arbre d'exploration. Ces coupes sont valides sur la branche associée au nœud où la coupe a été pratiquée. Linderoth et Ralphs [LR06] analysent les performances des logiciels non commerciaux basés sur ces idées.

Chapitre 8

Décomposition

Les problèmes de recherche opérationnelle comportent souvent un nombre gigantesque de variables, sur lesquelles on peut travailler sans les stocker en mémoire. Un bon exemple est la recherche de plus court chemin, qui n'implique pas le calcul de tous les chemins possibles. Plus généralement, il est souvent possible de résoudre un problème d'optimisation en générant un nombre limité de variables et/ou de contraintes.

Le chapitre commence par un principe de décomposition dû à Benders [Ben62]. L'idée est de partitionner les variables, et pour celles intervenant de manière linéaire, d'exprimer leur contribution avec le programme linéaire dual. On génère les points ou directions extrêmes du dual au cours de l'algorithme. Les applications principales sont la programmation linéaire mixte (variables entières et continues) et l'optimisation linéaire stochastique.

Le chapitre se poursuit avec la présentation des algorithmes de génération de colonnes et le cas particulier de la méthode de Dantzig-Wolfe [DW59]. On détaille l'application à la classe importante des problèmes multiflot.

Nous terminons avec l'exposé des problèmes à recours multiple, qui s'interprètent comme une généralisation (partielle) de la méthode de Benders.

8.1 Principe de décomposition de Benders

8.1.1 Méthode de Benders

Considérons un problème d'optimisation ayant la structure suivante :

$$\underset{x\in X, y\geqslant 0}{\text{Min}}\ f(x) + d\cdot y; \quad Ay = b + Mx. \tag{$LPRS$}$$

Ici X est une partie non vide de $\mathbb{R}^n$, f est une fonction $\mathbb{R}^n \to \mathbb{R}$, $d \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, et A et M sont des matrices de taille $m \times p$ et $m \times n$ resp. Précisons tout de suite les exemples les plus importants de telles situations.

Exemple 8.1 (i) Tout programme linéaire peut être décomposé de la sorte par une partition quelconque des variables. Cela correspond au cas où X est un polyèdre,
(ii) Le cas où X est l'intersection de $\mathbb{Z}^n$ avec un polyèdre, et f est linéaire, correspond à la classe des *programmes linéaires mixtes*. Il s'agit de problèmes dont le coût est linéaire, et qui comportent un nombre fini de contraintes linéaires, et de plus des contraintes d'intégrité pour certaines variables.

Remarque 8.2 On peut interpréter x et y comme une décision initiale et une décision finale, appelée *recours* dans ce contexte, tenant compte de la décision initiale. Dans certains exemples de problèmes avec recours, y représente un ensemble de décisions prises à plusieurs instants successifs. •

Pour x fixé, y est solution du programme linéaire

$$\underset{y \geqslant 0}{\text{Min}}\ d \cdot y; \quad Ay = z, \tag{LPS_z}$$

avec $z := b + Mx$. Le lagrangien du problème (LPS_z) est $d \cdot y + \lambda \cdot (Ay - z)$, et le problème dual a pour expression

$$\underset{\lambda}{\text{Max}}\ -z \cdot \lambda; \quad d + A^\top \lambda \geqslant 0. \tag{LDS_z}$$

L'ensemble réalisable du dual est indépendant de z ; il est naturel de le supposer non vide (sinon, la valeur duale étant $-\infty$, le problème (LPS_z) a pour valeur $\pm\infty$, donc n'a pas de solutions, ce qui implique que le problème (LPRS) n'a pas non plus de solution). Alors (LPS_z) et (LDS_z) ont la même valeur notée $v(z)$, et $v(z) > -\infty$ pour tout z. Réécrivons $(LPRS)$ comme un problème de minimisation en x seul :

$$\underset{x \in X}{\text{Min}}\ f(x) + v(b + Mx). \tag{$LPRS^*$}$$

Le lemme 2.20 permet de décomposer $F(LDS_z)$ en somme d'un polytope Q et d'un cône C. Il en résulte une expression de $v(\cdot)$ de la forme

$$v(z) = \begin{cases} \max\{-z \cdot \lambda^i,\ i \in I\} & \text{si } z \cdot \lambda^j \geqslant 0, \text{ pour tout } j \in J, \\ +\infty & \text{sinon,} \end{cases} \tag{8.1}$$

où les λ^i, $i \in I$ et λ^j, $j \in J$ sont les points extrêmes de Q et des représentants des directions extrêmes de C, respectivement. Reportant cette expression dans $(LPRS^*)$, on obtient le problème équivalent

$$\underset{x \in X}{\text{Min}}\ f(x) + \max\{-(b + Mx) \cdot \lambda^i,\ i \in I\}; \quad \lambda^j \cdot (b + Mx) \geqslant 0, \ \ j \in J. \tag{$LPRS'$}$$

Ce problème peut se réécrire sous la forme

$$\begin{array}{lll} \underset{x \in X, \alpha}{\text{Min}}\ f(x) + \alpha; & -\lambda^i \cdot (b + Mx) & \leqslant \alpha, \quad i \in I, \\ & -\lambda^j \cdot (b + Mx) & \leqslant 0, \quad j \in J. \end{array} \tag{$LPRS''$}$$

La contrainte indicée par $i \in I$ est dite *coupe d'optimalité*. Celle indicée par $j \in J$ est dite *coupe de réalisabilité*. En effet la première modifie la valeur obtenue pour x fixé, tandis que la seconde réduit l'ensemble des x admissibles.

La formulation $(LPRS'')$ élimine la variable y en la remplaçant par des contraintes linéaires portant sur la variable x et la variable auxiliaire scalaire α. Notons cependant que les points et directions extrêmes d'un polyèdre sont souvent très nombreux, de sorte qu'il est exclu de les calculer tous. Cette formulation n'est donc pas directement utilisable. Ceci amène à faire l'hypothèse suivante de *génération de point ou direction extrême* :

$$\left\{ \begin{array}{l} \text{On dispose d'un algorithme efficace tel que :} \\ \text{Pour une valeur donnée de } z, \text{ la résolution de } (LDS_z) \text{ fournit :} \\ \text{(i) si } (LPS_z) \text{ est non réalisable, un } \lambda^j,\ j \in J, \text{ tel que } \lambda^j \cdot z < 0, \\ \text{(ii) si } (LPS_z) \text{ est réalisable, un } \lambda^i,\ i \in I, \text{ tel que } -\lambda^i \cdot z = v(z). \end{array} \right. \tag{HG}$$

Notons que cette hypothèse est satisfaite si on résout (LDS_z) par un algorithme de type simplexe. On est donc amené à manipuler des approximations du problème $(LPRS'')$ du type

$$\begin{array}{lll} \underset{x \in X, \alpha}{\text{Min}}\ f(x) + \alpha; & -\lambda^i \cdot (b + Mx) & \leqslant \alpha, \quad i \in I_k, \\ & -\lambda^j \cdot (b + Mx) & \leqslant 0, \quad j \in J_k, \end{array} \tag{$LPRS_k$}$$

avec $I_k \subset I$ et $J_k \subset J$. La résolution d'un tel problème fournit un point x^k. On résout ensuite le problème (LDS_z) avec $z = b + Mx^k$. Ceci permet, soit d'augmenter les ensembles I_k et J_k, soit de démontrer l'optimalité de x^k. Plus précisément, notons $\text{extr}(Q)$, $\text{extr}(C)$ l'ensemble des points et directions extrêmes de Q et C, resp. L'algorithme peut s'écrire comme suit :

Algorithme 8.3 (Benders)
init Calculer $x^0 \in \operatorname{argmin}\{f(x); x \in X\}$; $I_0 := \emptyset$, $J_0 := \emptyset$, $k := 0$.
do $k := k + 1$.
phase rétrograde : Calculer $z^k := b + Mx^{k-1}$.
If (LPS_{z^k}) réalisable :
Calculer $i \in I$ et $\lambda^i \in \text{extr}(Q)$ tels que $v(z) = -\lambda^i \cdot z^k$.
Si $i \in I_{k-1}$, **stop** : x^{k-1} est optimal.
$I_k := I_{k-1} \cup \{i\}$.
Else (LPS_{z^k}) non réalisable : Calculer $j \in J$ et $\lambda^j \in \text{extr}(C)$ tels que $\lambda^j \cdot z^k < 0$.
$J_k := J_{k-1} \cup \{j\}$.
End if
phase avant : Calculer x^k solution de $(LPRS_k)$.
Si $(LPRS_k)$ non réalisable, **stop** : $(LPRS)$ est non réalisable.
end do

Remarque 8.4 (i) L'énoncé de l'algorithme suppose qu'il est possible de résoudre les problèmes $(LPRS_k)$ de manière exacte. Cependant, dans les premières itérations, il peut être suffisant de les résoudre de manière approchée. L'essentiel est de générer de nouvelles coupes d'optimalité ou de réalisabilité pour enrichir l'information.
(ii) Si (LPS_{z^k}) est réalisable, on peut arrêter l'algorithme dès que le nouveau multiplicateur λ^i est tel que $\lambda^i \cdot z^k = \lambda^{i'} \cdot z^k$, pour un certain $i' \in I_{k-1}$.
(iii) Il faut disposer d'un algorithme efficace pour résoudre le problème $(LPRS_k)$; c'est bien le cas si f est linéaire, et si X est soit un polyèdre, soit l'ensemble des points entiers d'un polyèdre, comme on l'a vu dans les chapitres précédents. •

Proposition 8.5 *La méthode de Benders s'arrête après un nombre fini d'itérations, soit en prouvant que le problème est non réalisable, soit en fournissant la solution du problème.*

Démonstration. Les ensembles I et J étant finis, l'ajout de coupes ne peut intervenir qu'un nombre fini de fois, d'où la conclusion. ■

Remarque 8.6 Il est difficile de donner une estimation du nombre d'itérations nécessitées par l'algorithme. Ceci n'empêche pas que ce nombre reste raisonnable dans de nombreuses applications. •

8.1.2 Application à la programmation linéaire

Il s'agit du cas où f est linéaire et X est un polyèdre. On peut supposer le problème de la forme (noter le changement de notation)

$$\underset{x \geqslant 0, y \geqslant 0}{\text{Min}} \; c \cdot x + d \cdot y; \quad A_0 x = b^0; \quad A^1 y = b^1 + Mx. \tag{$LPRS$}$$

La méthode de Benders conduit alors à résoudre une suite de programmes linéaires, chacun s'obtenant en ajoutant au précédent une contrainte linéaire d'inégalité. Montrons que l'algorithme du simplexe dual (étudié en section 6.2.4) est bien adapté à cette situation. Pour ceci il est commode d'utiliser un formalisme général. Soient les programmes linéaires, identiques à l'ajout près d'une contrainte linéaire scalaire :

$$\underset{x \geqslant 0}{\text{Min}} \; c \cdot x; \quad Ax = b, \tag{P}$$

$$\underset{x \geqslant 0}{\text{Min}} \; c \cdot x; \quad Ax = b; \;\; a \cdot x = \beta. \tag{$\hat{P}$}$$

La matrice A est de taille $p \times n$, et $a \in \mathbb{R}^n$. Ces programmes ont pour duaux respectifs, appelant λ et η les multiplicateurs associés aux contraintes $Ax - b = 0$ et $a \cdot x = \beta$ ($\lambda \in \mathbb{R}^p$ et $a \in \mathbb{R}$) :

$$\underset{\lambda}{\text{Max}} \; -b \cdot \lambda; \quad c + A^\top \lambda \geqslant 0, \tag{D}$$

$$\underset{\lambda,\eta}{\text{Max}} \quad -b\cdot\lambda - \beta\,\eta; \quad c + A^\top\lambda + \eta\,a \geqslant 0. \tag{$\hat{D}$}$$

L'ajout d'une contrainte linéaire revient donc pour le dual à introduire une variable (un degré de liberté) supplémentaire. Une solution basique de (D) fournit donc un point réalisable basique de $(\hat{D})$, pour lequel $\eta = 0$. Ce point permet une initialisation efficace de la méthode du simplexe dual, présentée en section 6.2.4.

8.1.3 Application à la programmation linéaire mixte

Il s'agit du cas où f est linéaire et X est l'ensemble des points entiers d'un polyèdre, soit

$$f(x) = c\cdot x; \quad X = \{x\in\mathbb{Z}^n;\ \ A^0x \leqslant b^0\}. \tag{8.2}$$

L'algorithme de Benders nécessite la résolution d'une suite de problèmes linéaires mixtes (n variables entières et une variable continue) de la forme suivante (on explicite $(LPRS_k)$) :

$$\begin{array}{lll} \underset{x\in\mathbb{Z}^n,\alpha}{\text{Min}}\ c\cdot x+\alpha; & A^0x \leqslant b^0; & \\ & -\lambda^i\cdot(b+Mx) & \leqslant \alpha, \quad i\in I_k, \\ & -\lambda^j\cdot(b+Mx) & \leqslant 0, \quad j\in J_k. \end{array} \tag{PLM_k}$$

Les algorithmes de résolution de ces problèmes sont étudiés dans les chapitres 5 et 7.

8.1.4 Application à l'optimisation linéaire stochastique

Il s'agit du cas particulier dans lequel y est une collection de vecteurs $(y^\omega)_{\omega\in\Omega}$; l'indice ω s'interprète comme un aléa, et on suppose ici Ω fini. On se donne aussi des probabilités p_ω de réalisation de l'événement ω (donc $p_\omega \geqslant 0$ et $\sum_{\omega\in\Omega} p_\omega = 1$), des coûts d^ω, et des contraintes linéaires dépendant de ω. Au total si on minimise en espérance, le problème est de la forme

$$\underset{x\in X, y\geqslant 0}{\text{Min}}\ f(x) + \sum_{\omega\in\Omega} p_\omega\, d^\omega\cdot y^\omega; \quad A^\omega y^\omega = b^\omega + M^\omega x, \quad \text{pour tout} \quad \omega\in\Omega. \tag{$LPRSS$}$$

Autrement dit, la décision initiale x doit être prise indépendamment de l'aléa, tandis que le recours y^ω dépend de la décision initiale et de l'aléa. Le problème de calcul du recours se sépare donc pour chaque $\omega\in\Omega$:

$$\underset{y^\omega\geqslant 0}{\text{Min}}\ d^\omega\cdot y^\omega; \quad A^\omega y^\omega = z^\omega, \tag{$LPSS_z$}$$

avec $z^\omega := b^\omega + M^\omega x$. Appelons $v^\omega(z^\omega)$ la valeur du problème ci-dessus. On peut réécrire $(LPRSS)$ sous la forme

$$\underset{x\in X}{\text{Min}}\, f(x) + \sum_{\omega\in\Omega} p_\omega\, v^\omega(b^\omega + M^\omega x). \tag{$LPRSS'$}$$

Ceci fait apparaître la valeur du recours sous forme de somme de valeurs dépendant de l'aléa. On obtient ainsi une variante de l'algorithme de Benders dans laquelle chaque fonction v^ω (plutôt que leur somme) est approchée par des minorantes affines et des coupes de réalisabilité. A chaque itération de cette variante, on ajoute donc (quand c'est possible) une coupe d'optimalité ou de réalisabilité pour chaque fonction v^ω.

8.2 Génération de colonnes et méthode de Dantzig-Wolfe

8.2.1 Exemple des problèmes multiflots

Un problème de multiflots modélise des réseaux dans lesquels circulent divers types de marchandises ; chacune obéit à la loi de Kirchhoff. L'interaction entre les différentes marchandises est due à la contrainte de capacité sur les arcs. Le problème du routage dans les réseaux de télécommunication se modélise de manière similaire.

Notons $(\mathcal{N}, \mathcal{A})$ le graphe, et soit K l'ensemble (fini) des marchandises. Les lois de Kirchhoff s'écrivent

$$b_i^k + \sum_{j;(j,i)\in\mathcal{A}} x_{ji}^k = \sum_{j;(i,j)\in\mathcal{A}} x_{ij}^k, \quad i \in \mathcal{N}, \quad k \in K. \tag{8.3}$$

Pour simplifier l'exposé on supposera que chaque bien k a un nœud source sk et un nœud puits pk, de sorte que

$$b_i^k = 0, \text{ pour tout } i \notin \{sk, pk\}; \quad b_{sk}^k = \bar{b}^k > 0; \quad b_{pk}^k = -\bar{b}^k < 0. \tag{8.4}$$

Remarque 8.7 Dans le cas général, on peut se ramener au cadre de sources et puits uniques en posant $\bar{b}^k := \sum_{i\in N} \max(b_i^k, 0)$, et en introduisant deux nœuds supplémentaires (les mêmes pour tous les biens) la source s et le puits p. On impose alors (8.4), ainsi que les contraintes d'injection de $\bar{b}^k$ en s et d'extraction de $\bar{b}^k$ en p. On retrouve le format (8.3)-(8.4), avec en plus des contraintes sur les flots dans les arcs reliant source et puits aux autres arcs, ce qui ne change pas de manière essentielle l'approche. •

Soit une *contrainte de capacité* des arcs du type

$$\sum_{k\in K} x_a^k \leqslant u_a, \quad \text{pour tout } a \in \mathcal{A}, \tag{8.5}$$

avec $0 < u_a < +\infty$, pour tout $a \in \mathcal{A}$. Le problème de minimisation de la somme du coût des flots, sous contrainte de capacité, s'écrit

$$\operatorname*{Min}_{x\geqslant 0} \sum_{k\in K} \sum_{a\in\mathcal{A}} c_a^k x_a^k; \quad \text{tel que (8.3)-(8.5).} \tag{MF}$$

Remarque 8.8 Les problèmes multiflots avec injections et capacités entières n'ont pas nécessairement de solutions entières. Ils sont donc profondément différents des problèmes de flots. Soit par exemple un graphe à trois sommets, et trois arcs $(1,2)$, $(2,3)$, et $(3,1)$. On a trois types de marchandises. Les injections (extractions) ont lieu aux sommets 1, 2, et 3 (resp. 3, 1, et 2). On maximise la masse totale transportée, sous contrainte de capacité unitaire des arcs. Moyennant la création d'arcs de retour, ce problème est du type (MF) avec $b = 0$. Les solutions entières consiste à transporter une unité d'une des marchandises. La solution fractionnaire est de transporter une demi unité de chaque marchandise.

•

Remarque 8.9 Nous avons mis les biens transportés à l'échelle de manière à ce qu'ils aient tous le même encombrement. On peut généraliser le problème en introduisant des encombrements variables suivant les arcs, des capacités tenant compte de poids différents suivant les arcs et les marchandises, etc. L'approche développée dans la suite s'étend sans difficultés à ces cas. •

On suppose les coûts c^k positifs. Si les contrainte de capacité étaient absentes, ce problème se découplerait en $|K|$ problèmes de plus court chemin. Plus généralement, on peut s'attendre à ce que le support d'un flot x^k (l'ensemble des arcs sur lesquels x^k est strictement positif) soit souvent petit ; pensons par exemple au routage de communications ; celles entre Rennes et Brest ne passeront pas par Strasbourg. Ceci pousse à introduire des solutions passant par un nombre limité de chemins. Dans un premier temps, montrons qu'on peut se limiter à des flots sans circuits.

Proposition 8.10 *On suppose les coûts c^k positifs. Si le problème est réalisable, il a une solution x telle que, pour tout $k \in K$, le support de x^k ne comporte aucun circuit, et x^k est combinaison linéaire positive de chemins élémentaires allant de la source au puits.*

Démonstration. L'ensemble réalisable est compact ; l'infimum du critère linéaire est donc atteint, ce qui assure l'existence d'une solution à (MF).

Si $x \in F(MF)$, et $k \in K$, le support du flot x^k est l'ensemble des arcs sur lesquels le flot n'est pas nul. Notons $\nu_k(x)$ son cardinal, et $\nu(x) := \sum_{k\in K} \nu_k(x)$. Soit $\bar{x}$ une solution de (MF), minimisant $\nu(\cdot)$ sur

$S(MF)$. Montrons que $\bar{x}$ ne comporte pas de circuit. Dans le cas contraire, soit $k \in K$ tel que le support de $\bar{x}^k$ comprenne un circuit. Notons $\alpha > 0$ le flot minimum de $\bar{x}^k$ sur les arcs du circuit. Diminuant $\bar{x}^k$ de $\alpha > 0$ sur les arcs du circuit, on obtient un nouveau flot, de coût inférieur ou égal (car $c^k \geqslant 0$) et avec un support strictement plus petit, ce qui contredit la définition de $\bar{x}$. Nous avons montré que $\bar{x}$ ne comporte pas de circuit. La proposition 3.10, qui assure que chaque flot se décompose en somme de combinaisons convexes de produits de chemins par la quantité à transporter, et de combinaisons linéaires positives de circuits, permet de conclure. ■

Notons I_k l'ensemble des chemins élémentaires du nœud source au nœud puits du flot k, et $\pi^{k\ell}$ les vecteurs représentant les chemins élémentaires de la source au puits : $\pi_a^{k\ell}$ vaut 1 ou 0 suivant que le chemin passe par l'arc a ou non. On peut paramétrer chaque flot x^k sans circuit sous la forme

$$x^k = \sum_{\ell \in I_k} \alpha_{k\ell}\, \pi^{k\ell}; \quad \alpha_{k\ell} \geqslant 0; \quad \sum_{\ell \in I_k} \alpha_{k\ell} = \bar{b}^k, \text{ pour tout } k \in K, \tag{8.6}$$

et si on note les coûts associés aux chemins élémentaires par

$$\hat{c}_{k\ell} := c \cdot \pi^{k\ell} = \sum_{a \in \mathcal{A}} c_a \pi_a^{k\ell}, \tag{8.7}$$

on obtient alors la *formulation en chemins* du problème multiflots (MF) :

$$\begin{array}{ll} \text{Min}_{\alpha \geqslant 0}\ \hat{c} \cdot \alpha; & \displaystyle\sum_{\ell \in I_k} \alpha_{k\ell} = \bar{b}^k, \quad \text{pour tout } k \in K; \\ & \displaystyle\sum_{k \in K} \sum_{\ell \in I_k} \alpha_{k\ell} \pi_a^{k\ell} \leqslant u_a, \quad \text{pour tout } \ a \in \mathcal{A}. \end{array} \tag{MFC}$$

L'inconvénient de cette paramétrisation est que le nombre de chemins élémentaires est immense, de sorte qu'il est exclu de les calculer tous. Au contraire nous désirons opérer sur un petit nombre de chemins susceptibles d'être présents à la solution. Montrons dans la section suivante comment traiter ce type de problème dans un cadre plus général.

8.2.2 Génération de colonnes

Motivation Considérons un programme linéaire sous forme standard

$$\underset{x \geqslant 0}{\text{Min}}\ c \cdot x; \quad Ax = b, \tag{PL}$$

avec A de taille $p \times n$. Supposons n très grand : on veut éviter de manipuler explicitement des vecteurs de $\mathbb{R}^n$ et l'ensemble de la matrice A. Nous savons (corollaire 6.14) que le problème a une solution basique, donc avec au plus p composantes non nulles. L'algorithme du simplexe (section 6.2) permet bien de limiter une grande partie des calculs à des vecteurs de $\mathbb{R}^n$ avec au plus p composantes non nulles, mais nécessite le calcul du gradient réduit, de dimension $n - p$; si la variable entrante est celle de gradient réduit minimal, on peut se poser le problème de calculer la variable entrante sans expliciter le gradient réduit. Soit (A_B, A_N) la partition de A en colonnes basiques et non basiques. Lors d'une itération de l'algorithme du simplexe, une fois calculée l'estimation $\lambda = -A_B^{-\top} c_B$, on sait que le gradient réduit vaut $g = c_N + A_N^\top \lambda$. Le problème de sélection de la variable entrante (suivant la règle de Dantzig de choix de la composante de gradient réduit minimal) s'écrit donc, en notant A_i la $i$ème colonne de A :

$$\underset{i \in N}{\text{Min}}\, (c_i + \lambda \cdot A_i)\,. \tag{8.8}$$

L'attaque directe de ce problème en variables entières est difficile en général, mais se réduit comme nous le verrons à un calcul de plus courts chemins pour un problème multiflot. Présentons une variante basée sur la résolution de sous problèmes avec gestion de variables actives.

Gestion de variables actives L'idée est simplement de résoudre des problèmes du type

$$\underset{x\geqslant 0}{\text{Min}}\, c\cdot x;\quad Ax=b;\quad x_i=0,\ \ i\notin I. \tag{PL_I}$$

Ici $I\subset\{1,\dots,n\}$ est un ensemble de cardinal au moins p, aussi petit que possible pour faciliter la résolution de (PL_I). A une solution x^I de (PL_I) sont associés des multiplicateurs $\lambda\in\mathbb{R}^p$ et $s\in\mathbb{R}^n$ tels que

$$c+A^\top\lambda=s;\quad s_i\geqslant 0,\quad s_i x_i^I=0,\ \ \text{pour tout}\ \ i\in I. \tag{8.9}$$

Si (B,N) est la partition en variables de base et hors base, alors $s_B=0$ et le gradient réduit est s_N. Une fois résolu (PL_I), il faut résoudre le problème (8.8) pour déterminer la variable entrante. Comme $c_i+\lambda\cdot A_i=0$ pour tout $i\in B$ (par définition de λ) il revient au même de résoudre le problème

$$\underset{1\leqslant i\leqslant n}{\text{Min}}\ (c_i+\lambda\cdot A_i) \tag{8.10}$$

pour trouver la variable de gradient réduit (égal ici à la restriction de s à N) minimum, ou plus généralement (comme on le verra pour les multiflots) un ensemble J de variables de gradient réduit négatif ; on fait alors $I:=I\cup J$ et on itère jusqu'à obtenir un gradient réduit positif. L'avantage de cette variante est de résoudre moins souvent les problèmes de type (8.8) ou (8.10).

Contraintes couplantes Les problèmes de multiflots sont un cas particulier du format suivant :

$$\underset{x\geqslant 0}{\text{Min}}\, c\cdot x;\quad A^k x^k=b^k,\ k\in K;\quad A^0x^0+\sum_{k\in K}B^kx^k\leqslant b^0. \tag{8.11}$$

L'absence de la contrainte couplante $A^0x+\sum_{k\in K}B^kx^k\leqslant b^0$ permettrait le découplage pour chaque $k\in K$ (dans le cas des problèmes de multiflot, le couplage est dû aux contraintes de capacité). Notons λ^k, $k\in K$, et μ les multiplicateurs associés aux contraintes linéaires ; le multiplicateur associé aux contraintes de bornes est

$$s^0=c^0+(A^0)^\top\mu;\quad s^k=c^k+(A^k)^\top\lambda^k+(B^k)^\top\mu. \tag{8.12}$$

La recherche de variables de coût réduit négatif se découple donc pour les variables x^0 et x^k, $k\in K$. On peut donc lancer une procédure de génération de colonne pour chaque marchandise.

8.2.3 Génération des chemins pour un problème multiflot

Nous appliquons le formalisme précédent au problème (MFC) de fin de la section 8.2.1 avec coûts c^k positifs. On retrouve un cas particulier du format (8.11) dans lequel chaque A^k est un vecteur ligne ne comportant que des 1, b^k vaut $\bar b^k$, et la contrainte couplante est celle de capacité, qu'on peut réécrire comme une contrainte d'égalité en introduisant la variable d'écart x^0 (capacités résiduelles) et la matrice identité A^0. Notant η_k, $k\in K$ (scalaires) et $\mu\in\mathbb{R}^{|\mathcal{A}|}$ les multiplicateurs aux contraintes linéaires, il vient

$$s^{k\ell}=\hat c_{k\ell}+\eta_k+\pi^{k\ell}\cdot\mu=\eta_k+\pi^{k\ell}\cdot(c^k+\mu),\quad k\in K,\ \ell\in I_k, \tag{8.13}$$

et $s^0=\mu$, donc $\mu\geqslant 0$. Le test d'optimalité nécessite la minimisation de cette quantité sur tout les chemins possibles. Ceci ne dépend pas de η_k, et s'interprète comme un *problème de plus court chemin* de la source au puits, chaque arc $a\in\mathcal{A}$ ayant pour longueur *positive* $c_a^k+\mu_a$. Notons J_k^q l'ensemble des chemins permis à l'itération q de l'algorithme de génération de colonne ; lors de cette itération, on résout le problème

$$\begin{aligned}\underset{\alpha\geqslant 0}{\text{Min}}\ \hat c\cdot\alpha;\quad &\alpha_{k\ell}=0,\ \ell\notin J_k^q;\quad \sum_{\ell\in I_k}\alpha_{k\ell}=\bar b^k,\ k\in K;\\ &\sum_{k\in K}\sum_{\ell\in I_k}\alpha_{k\ell}\,\pi_a^{k\ell}\leqslant u_a,\ \ \text{pour tout}\ \ a\in\mathcal{A}.\end{aligned} \tag{MFC_q}$$

Nous obtenons l'algorithme suivant :

Algorithme 8.11 (Génération de chemins)
begin genpath
init : $J_k^1 \subset I_k$, pour tout $k \in K$; $q := 1$.
while $q = 1$ ou $J^q \neq J^{q-1}$
Résoudre (MFC_q) ; calculer μ^q, multiplicateur associé aux contraintes de capacité.
$J_k^{q+1} := J_k^q \cup \{\ell_k\}$, où ℓ_k est le plus court chemin avec les poids $c_a^k + \mu_a^q$, pour tout $k \in K$.
$q := q + 1$
end while
end genpath

Proposition 8.12 *Supposons (MFC) et (MFC_1) réalisables. Alors l'algorithme 8.11 se termine en fournissant une solution de (MFC).*

Démonstration. A chaque itération, au moins un des J_k^q augmente strictement. Comme l'ensemble des chemins possibles est fini, le nombre des itérations l'est aussi. ■

Remarque 8.13 Une difficulté est d'assurer la réalisabilité du premier problème (MFC_1). On peut toujours ajouter des capacités supplémentaires de prix très élevés, qui seront nulles à l'optimum. •

8.2.4 Méthode de Dantzig-Wolfe

La génération de colonnes (section 8.2.2) contient un cas particulier important, introduit par Dantzig et Wolfe [DW59] en 1959. Considérons un problème du type

$$\operatorname*{Min}_x c \cdot x; \quad x \in P \cap P', \tag{DW}$$

avec $P = \{x \geqslant 0;\ Ax = b\}$, et $P' = \{x;\ A^0 x \leqslant b^0\}$. L'idée est d'utiliser la décomposition de P donnée dans le lemme 2.20 : P est somme d'un polytope Q et d'un cône C, Q est enveloppe convexe de ses points extrêmes (en nombre fini), et C est généré par un nombre fini de directions extrêmes. Tout $x \in P$ peut donc s'écrire sous la forme

$$x = \sum_{i \in \mathcal{I}} \alpha_i x^i + \sum_{j \in \mathcal{J}} \beta_i y^j; \quad \alpha, \beta \geqslant 0, \quad \sum_{i \in \mathcal{I}} \alpha_i = 1. \tag{8.14}$$

On peut donc reparamétrer le problème (DW) par (α, β) ; on obtient

$$\operatorname*{Min}_{(\alpha,\beta) \geqslant 0} \sum_{i \in \mathcal{I}} \alpha_i\, c \cdot x^i + \sum_{j \in \mathcal{J}} \beta_i\, c \cdot y^j; \quad \sum_{i \in \mathcal{I}} \alpha_i A^0 x^i + \sum_{j \in \mathcal{J}} \beta_i A^0 y^j \leqslant b^0; \quad \sum_{i \in \mathcal{I}} \alpha_i = 1. \tag{DW'}$$

Cette formulation n'est pas directement utilisable car le nombre de points extrêmes de Q et directions extrêmes de C peut être très élevé, et le calcul de l'ensemble de ces points est coûteux. D'un autre côté, supposons connus un nombre limité de points et directions extrêmes, indicés par $I \subset \mathcal{I}$ et $J \subset \mathcal{J}$, et soit le problème correspondant

$$\operatorname*{Min}_{(\alpha,\beta) \geqslant 0} \sum_{i \in I} \alpha_i c \cdot x^i + \sum_{j \in J} \beta_i c \cdot y^j; \quad \sum_{i \in I} \alpha_i A^0 x^i + \sum_{j \in J} \beta_i A^0 y^j \leqslant b^0; \quad \sum_{i \in I} \alpha_i = 1. \tag{DW'_{IJ}}$$

Ce problème revient à minimiser le coût sur un ensemble plus petit. Posant $I' := \mathcal{I} \setminus I$, $J' := \mathcal{J} \setminus J$, on peut aussi l'écrire sous la forme

$$\begin{array}{l} \operatorname*{Min}_{(\alpha,\beta)} \sum_{i \in \mathcal{I}} \alpha_i c \cdot x^i + \sum_{j \in \mathcal{J}} \beta_i c \cdot y^j; \quad \sum_{i \in \mathcal{I}} \alpha_i A^0 x^i + \sum_{j \in \mathcal{J}} \beta_i A^0 y^j \leqslant b^0; \quad \sum_{i \in \mathcal{I}} \alpha_i = 1, \\ \qquad \alpha_i \geqslant 0;\ i \in I; \alpha_i = 0,\ i \in I'; \\ \qquad \beta_j \geqslant 0;\ j \in J; \beta_j = 0,\ J \in J'. \end{array} \tag{DW''_{IJ}}$$

Notons $\gamma \in \mathbb{R}$ le multiplicateur associé à la contrainte $\sum_{i\in\mathcal{I}} \alpha_i = 1$, μ celui associé à la contrainte linéaire d'inégalité, et $\eta \in \mathbb{R}^{|\mathcal{I}|+|\mathcal{J}|}$ celui associé aux contraintes de positivité ou nullité suivant les indices. La condition d'optimalité se compose de la condition duale (minimisation du lagrangien)

$$\left\{ \begin{array}{rcll} \eta_i & = & c\cdot x^i + \gamma + \mu\cdot A^0 x^i, & i\in\mathcal{I}, \\ \eta_j & = & c\cdot y^j + \mu\cdot A^0 y^j, & j\in\mathcal{J}, \end{array} \right. \tag{8.15}$$

ainsi que des conditions de signe et complémentarité

$$\left\{ \begin{array}{llllll} \eta_I \geqslant 0; & \alpha_I \geqslant 0; & \eta_i\alpha_i = 0, & i\in I; & \alpha_i = 0, & i\in I', \\ \eta_J \geqslant 0; & \beta_J \geqslant 0; & \eta_j\beta_j = 0, & j\in J; & \beta_j = 0, & j\in J'. \end{array} \right. \tag{8.16}$$

Noter que la résolution effective n'implique que les indices dans I et J ; les valeurs de η sur I' et J' se déduisent de (8.15). La théorie de la programmation linéaire implique le lemme suivant.

Lemme 8.14 *Soit (α,β) solution du problème (DW'_{IJ}). Alors (α,β) est solution du problème (DW') si $\eta_i \geqslant 0$ et $\eta_j \geqslant 0$, pour tout $i\in I'$ et $j\in J'$.*

Démonstration. En effet, si la condition est satisfaite les conditions d'optimalité de (DW) (réalisabilité primale et duale, et complémentarité) le sont. ■

On voit que l'idée de Dantzig et Wolfe est une génération de colonnes particulière. On obtient l'algorithme de Dantzig-Wolfe en spécialisant l'algorithme 8.11 à la formulation (DW''_{IJ}).

Remarque 8.15 Puisqu'un programme linéaire est le dual de son dual, ses variables d'optimisation s'interprètent comme les multiplicateurs de Lagrange des contraintes duales. La génération de colonne s'interprète donc comme une génération de contraintes pour le dual. •

Exercice 8.16 (Découpe de rouleaux de papier) Cet exercice reprend l'analyse de P.C. Gilmore et R.E. Gomory [GG61]. Un grossiste en papier doit découper un lot de "grands" rouleaux de largeur W pour produire b_i rouleaux de même longueur et de largeur (plus petite) w_i, $i = 1$ à m. On cherche la découpe la plus économique, qui minimise le nombre de "grands" rouleaux utilisés.

Appellons "format de découpe" une découpe possible d'un grand rouleau. Soit J l'ensemble des formats de découpe. Un format est donc un vecteur $y \in \mathbb{N}^m$, où y_i représente le nombre de rouleaux de largeur w_i, tel que $\sum_{i=1}^m w_i y_i \leqslant W$. On note a_{ij} le nombre de rouleaux de largeur w_i que le format $j \in J$ contient. Naturellement il faut que

$$\sum_{i=1}^m w_i a_{ij} \leqslant W, \quad \text{pour tout } \ j\in J. \tag{8.17}$$

1. Soit x_j le nombre de rouleaux découpés suivant le format $j \in J$. On relaxe la contrainte d'intégrité sur x. Montrer que le problème se formule comme

$$\operatorname*{Min}_{x\geqslant 0} \sum_{j\in J} x_j; \quad b_i - \sum_{j\in J} a_{ij}x_j \leqslant 0, \quad i = 1,\dots,m. \tag{LP}$$

2. Ecrire le problème dual de (LP), noté (LD). On notera λ le multiplicateur de Lagrange associé à la contrainte linéaire, et s celui associé à la contrainte de positivité de x.

3. On cherche à éviter de manipuler explicitement l'ensemble des formats de découpe (qui peut être très grand). Soit $J_0 \subset J$. On résout le problème (supposé réalisable)

$$\operatorname*{Min}_{x_j\geqslant 0, j\in J_0} \sum_{j\in J_0} x_j; \ \ b_i - \sum_{j\in J_0} a_{ij}x_j \leqslant 0, \quad i = 1,\dots,m. \tag{LP_{J_0}}$$

Ecrire le problème dual de (LP_{J_0}), noté (LD_{J_0}). On notera encore λ, s les multiplicateurs de Lagrange.

4. Notons J_1 le complémentaire de J_0 dans J. Soit x^0 solution de (LP_{J_0}) et (s^0, λ^0) solution de (LD_{J_0}). Posons $\hat{s}_j := 1 - \sum_{i=1}^m \lambda_i^0 a_{ij}$, pour tout $j \in J$. Montrer que, si $\hat{s} \geqslant 0$, alors $\hat{x}$ défini par

$$\hat{x}_j = x_j^0 \text{ si } j \in J_0, \ 0 \text{ si } j \in J_1,$$

est solution de (LP).

5. Montrer que la coordonnée la plus négative de $\hat{s}$ s'obtient en résolvant le problème de sac à dos à variables entières (pas nécessairement $\{0,1\}$) :

$$\underset{y \in \mathbb{N}^m}{\text{Max}} \sum_{i=1}^m \lambda_i^0 y_i; \quad \sum_{i=1}^m w_i y_i \leqslant W. \tag{8.18}$$

6. En déduire un algorithme de résolution de (LP), partant de la solution de (LP_{J_0}).

Exercice 8.17 (Stockage optimal) Soient $i = 1, \ldots, n$ des silos de stockage de blé de coût c_i, capacité k_i, et coût unitaire de stockage d_i. Toutes ces constantes sont strictement positives. De plus on suppose que $d_1 < \cdots < d_n$. On doit décider quels silos acheter, et quelles quantités y_i mettre dans le silo i de manière à assurer un stockage total $S > 0$ suffisant :

$$\sum_{i=1}^n y_i \geqslant S. \tag{8.19}$$

1. Modéliser le problème.

2. Formuler le programme linéaire d'optimisation du recours.

3. Ecrire le problème dual de ce dernier, et vérifier que primal et dual ont même valeur.

4. Formuler un problème équivalent au problème dual, en éliminant le multiplicateur de Lagrange associé à la contrainte (8.19).

5. Notons P^* l'ensemble réalisable du problème dual. Calculer les directions extrémales du cône asymptotique de P^*.

6. Quelles coupes de réalisabilité l'algorithme va-t-il activer ?

7. Calculer les points extrémaux de P^*.

8.3 Optimisation avec recours multiples

Considérons un programme linéaire ayant la structure suivante :

$$\underset{x \geqslant 0}{\text{Min}} \sum_{t \in \mathcal{T}} c_t \cdot x_t; \quad A_t x_t = b_t + M_t x_{t-1}, \quad t \in \mathcal{T}, \tag{LP}$$

avec $\mathcal{T} := \{1, \ldots, T\}$, et x_0 donné. On peut voir $t \in \mathcal{T}$ comme un temps et x_t la décision prise au temps t. En raison de son caractère dynamique, il peut arriver que ce programme linéaire soit de si grande taille qu'un algorithme non spécialisé ne peut le résoudre. On peut voir ce problème comme un cas particulier de la formulation de type Benders, en partitionnant les variables de décision en x_1 et $y = (x_2, \ldots, x_T)$. Nous allons montrer le lien entre théorie de la dualité linéaire et programmation dynamique, dans l'esprit de la méthode de Benders. Les fonctions valeur sont définies de la manière suivante :

$$v_T(z) \quad := \quad \inf_{x_T \geqslant 0} \{c_T \cdot x_T; \quad A_T x_T = z\}, \tag{8.20}$$

et pour $\tau = 0, \ldots, T-1$:

$$v_\tau(z) := \min_{x \geqslant 0} \left\{ \sum_{t=\tau}^{\top} c_t \cdot x_t;\ A_t x_t = b_t + M_t x_{t-1},\ t = \tau+1, \ldots, T;\ A_\tau x_\tau = z \right\}.$$

On obtient sans difficulté le principe de programmation dynamique

$$v_t(z) := \inf_{x_t \geqslant 0} \{c_t \cdot x_t + v_{t+1}(b_{t+1} + M_{t+1} x_t);\ A_t x_t = z\}, \quad t = 1, \ldots, T-1; \tag{8.21}$$

Comme v_T est la valeur d'un programme linéaire, examinons le problème dual :

$$\operatorname*{Max}_z\ -z \cdot \lambda;\ c_T + (A_T)^\top \lambda \geqslant 0. \tag{8.22}$$

L'ensemble des points réalisables du dual ne dépend pas de z ; on peut donc supposer que cet ensemble est non vide (sinon (LP) n'est pas réalisable). Alors primal et dual ont même valeur :

$$v_T(z) = \sup\{-z \cdot \lambda;\ c_T + (A_T)^\top \lambda \geqslant 0\}. \tag{8.23}$$

Utilisant la décomposition de Minkowski-Weyl et les point et directions extrêmes associés, on peut écrire la fonction valeur finale sous la forme

$$v_T(z) = \begin{cases} \max\{-z \cdot \lambda^i,\ i \in I\} & \text{si } z \cdot \lambda^j \geqslant 0, \text{ pour tout } i \in J, \\ +\infty & \text{sinon.} \end{cases} \tag{8.24}$$

On supposera encore que la résolution de (8.22) pour différentes valeurs de z fournit à chaque fois un élément z^i, $i \in I$, ou z^j, $i \in J$, et on est donc amené à manipuler des *minorantes* de v_T du type

$$w_T(z) = \begin{cases} \max\{-z \cdot \lambda^i,\ i \in \hat{I}\} & \text{si } z \cdot \lambda^j \geqslant 0, \text{ pour tout } i \in \hat{J} \\ +\infty & \text{sinon,} \end{cases} \tag{8.25}$$

avec $\hat{I} \subset I$ et $\hat{J} \subset J$. En raison du principe de programmation dynamique (8.21), ceci induit une *minoration* de v_{T-1} de la forme suivante :

$$\hat{v}_{T-1}(z) = \inf_{x \geqslant 0} \{c_{T-1} \cdot x + w_T(b_T + M_T x);\quad A_{T-1} x = z\}. \tag{8.26}$$

Compte-tenu de l'expression de $w_T(\cdot)$, le calcul de $\hat{v}_{T-1}(z)$ se ramène à la résolution d'un programme linéaire :

$$\begin{array}{lll} \hat{v}_{T-1}(z) = \inf\limits_{x \geqslant 0, \alpha} \quad c_{T-1} \cdot x + \alpha; & A_{T-1} x & = z; \\ & -(b_T + M_T x) \cdot \lambda^i & \leqslant \alpha,\ i \in \hat{I}; \\ & -(b_T + M_T x) \cdot \lambda^j & \leqslant 0,\ j \in \hat{J}. \end{array} \tag{8.27}$$

Étendons le principe pour tous les instants. Supposons donné pour $t \in \mathcal{T}$ une minorante de v_t de la forme

$$w_t(z) = \begin{cases} \max\{-z \cdot \lambda^i,\ i \in \hat{I}_t\} & \text{si } z \cdot \lambda^j \geqslant 0, \text{ pour tout } i \in \hat{J}_t, \\ +\infty & \text{sinon.} \end{cases} \tag{8.28}$$

Pour initialiser cet ensemble de minorantes, on peut prendre une constante négative assez grande. Fixons $t < T$ et $z \in \mathbb{R}^m$. En s'appuyant sur w_{t+1}, on peut calculer une nouvelle minorante affine de v_t exacte au point z, en s'appuyant sur le problème :

$$\hat{v}_t(z) = \inf_{x \geqslant 0} \{c_t \cdot x + w_{t+1}(b_{t+1} + M_{t+1} x);\quad A_t x = z\}. \tag{8.29}$$

En effet, puisque $w_{t+1} \leqslant v_{t+1}$, une minorante affine de la fonction $\hat{v}$ définie ci-dessus sera bien minorante affine de v_t. L'expression de cette minorante affine sera obtenue en reprenant la méthode de Benders présentée au début du chapitre.

Ceci permet la mise en œuvre des deux opérations fondamentales, la *phase avant* (prise de décision par calcul d'une trajectoire basée sur les minorantes w_t), qui s'opère pour t variant de 1 à T, et *phase arrière* (amélioration de ces minorantes) dans laquelle t varie de T à 1. Exposons un algorithme de principe dans le cas où $\mathrm{dom}(v_t) = \mathbb{R}^n$ pour tout t, et donc également $\mathrm{dom}(w_t) = \mathbb{R}^n$ pour tout t :

Algorithme 8.18 (Programmation dynamique duale, cas réalisable)
init : Fournir des minorantes w_t de v_t, $t = 1, \dots, T-1$, et fixer $\varepsilon > 0$.
do Phase avant (primale) : Pour $t = 0, \dots, T-1$, calculer successivement la trajectoire "sous optimale" x_t solution de (cf (8.29)) :

$$\underset{x \geqslant 0}{\text{Min}}\ c_t \cdot x + w_{t+1}(b_{t+1} + M_{t+1}x); \qquad A_t x = b_t + M_t x_{t-1}. \tag{8.30}$$

Phase arrière (duale) : Rétropropagation de l'information pour affiner les approximations $w_t(x)$. Poser $h_T := b_T + M_T x_{T-1}$, évaluer $v_T(h_T)$ et calculer une minorante affine de v_T, exacte en h_T, soit $v_T(h_T) + q_T \cdot (z - h_T)$. Faire

$$w_T(z) := \max(w_T(z), v_T(h_T) + q_T \cdot (z - h_T)). \tag{8.31}$$

Pour t allant de $T-1$ à 1, poser $h_t := b_t + M_t x_{t-1}$, évaluer $\hat{v}_t(h_t)$, calculer une minorante affine de v_t, soit $v_t(h_t) + q_t \cdot (z - h_t)$, et faire

$$w_t(z) := \max(w_t(z), \hat{v}_t(h_t) + q_t \cdot (z - h_t)). \tag{8.32}$$

test Si $\sum_{t \in \mathcal{T}} c_t \cdot x_t - w_1(h_1 + M_1 x^0) \leqslant \varepsilon$, arrêt.
end do

La phase avant (arrière) fournit un majorant (minorant) de la valeur optimale. Le test d'arrêt porte sur la différence entre majorant et minorant.

Remarque 8.19 (i) Comme le problème de calcul de x_t est un programme linéaire, la nouvelle minorante affine (dans la phase rétrograde) est fournie par une solution du problème dual, tout comme dans la méthode de Benders.
(ii) Le coût effectivement obtenu lors de la phase avant constitue un majorant de la valeur du problème ; comme l'algorithme inclut aussi une minorante, on obtient un test d'arrêt garantissant une optimalité à $\varepsilon > 0$ près en imposant que la différence entre majorant et minorant ne dépasse pas ε.
(iii) Les solutions duales sont usuellement des point extrêmes du polytope des points admissibles pour le dual. Après un nombre fini d'itérations la fonction $\hat{v}_{T-1}$ reste donc invariante. Par récurrence on déduit qu'il en est de même pour les fonctions $\hat{v}_t$, pour tout t. Les trajectoires calculées sont donc identiques à partir d'une certaine itération. Nécessairement $\hat{v}_t$ et v_t ont même valeur le long de cette trajectoire, qui est donc solution du problème de départ. •

Cas de points non réalisables Si $\text{dom}(v_t) \neq \mathbb{R}^n$, il est possible que le calcul de trajectoire sous optimale ne puisse être mené jusqu'au temps T. Il s'agit du cas où pour un certain t, ayant calculé x_{t-1}, il apparaît que $z_t := b_t + M_t x_{t-1}$ est tel que $z_t \notin \text{dom}(w_t)$. Le programme dual de (8.30) a donc une direction extrémale fournissant une coupe (ici encore, comme dans la méthode de Benders). Il faut ajouter cette coupe à la définition de w_t, et reprendre le calcul de x_{t-1} (ce qui revient à reprendre le calcul de trajectoire en reculant d'un cran).

8.4 Notes

La décomposition de Benders [Ben62] a été proposée d'abord pour les problèmes linéaires mixtes (contrainte d'intégrité sur une partie des variables). Voir aussi Geoffrion [Geo72]. La décomposition de Dantzig et Wolfe est issue de [DW59]. Une application importante de ces méthodes est le cas dynamique et stochastique (dit programmation stochastique), voir l'ouvrage de référence Ruszczynski et Shapiro [RS03] une application importante étant la production et le stockage d'énergie et les calculs d'investissement. En particulier l'approche de la section 8.3 se généralise au cas stochastique, voir Pereira et Pinto [PP91].

Chapitre 9

Optimisation sous contraintes d'inégalités matricielles, ou "SDP"

9.1 Introduction : le problème SDP

Rappelons que le problème de programmation linéaire standard consiste à maximiser une forme linéaire sur l'intersection du cône positif de $\mathbb{R}^n$ avec un nombre fini d'hyperplans, soit

$$\underset{x \in K}{\text{Min}} \; c \cdot x; \;\; Ax = b \tag{9.1}$$

avec

$$K = \mathbb{R}^n_+ := \{x \in \mathbb{R}^n \mid x_i \geqslant 0, \; i = 1, \ldots, n\}$$

et $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$.

Certains algorithmes de résolution de programmes linéaires, comme les très efficaces méthodes de points intérieurs présentées dans le chapitre 10, s'étendent à d'autres cônes, tels que le cône des matrices symétriques positives, que nous examinons maintenant.

Le cône des matrices symétriques positives, $\mathcal{S}_n^+$ Nous considérons le cas où l'inconnue x n'est pas un vecteur mais est une matrice symétrique. Notons $\mathcal{S}_n$ l'espace vectoriel des matrices symétriques

$$\mathcal{S}_n := \{X \in \mathbb{R}^{n \times n} \mid X = X^\top\} \; .$$

Un cône qui intervient souvent est le cône $\mathcal{S}_n^+ \subset \mathcal{S}_n$ formé des matrices symétriques positives de dimension n, soit

$$\mathcal{S}_n^+ = \{X \in \mathcal{S}_n \mid X \succcurlyeq 0\} \; .$$

Nous désignons ici par $\succcurlyeq$ l'ordre usuel des matrices symétriques, parfois appelé *ordre de Loewner*, qui n'est autre que l'ordre des formes quadratiques associées :

$$X \preccurlyeq Y \iff u^\top X u \leqslant u^\top Y u, \; \forall u \in \mathbb{R}^n \; .$$

Le cône $\mathcal{S}_n^+$ est fermé : en effet, pour tout u, l'ensemble des $X \in \mathcal{S}_n$ telles que $u^\top X u \geqslant 0$ est la préimage du fermé $[0, +\infty[$ par l'application continue $X \mapsto u^\top X u$, et $\mathcal{S}_n^+$ est l'intersection de ces préimages lorsque u parcourt $\mathbb{R}^n$.

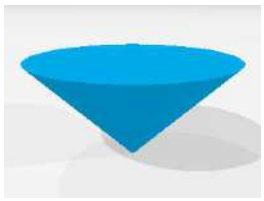
On peut aussi écrire la contrainte $X \succcurlyeq 0$ sous la forme

$$\lambda_{\min}(X) \geqslant 0 \; ,$$

où $\lambda_{\min}$ désigne la plus petite valeur propre d'une matrice symétrique. Appelons *cône de Lorentz* de dimension n, l'ensemble de $\mathbb{R}^n$ défini par

$$x_n \geqslant \sqrt{x_1^2 + \cdots + x_{n-1}^2}. \tag{9.2}$$

En dimension 3, le cône de Lorentz a la forme d'un "cornet de glace" :



Quand $n = 2$, on peut identifier l'ensemble des matrices symétriques positives au cône de Lorentz de dimension 3, en écrivant une matrice symétrique X comme combinaison linéaire des *matrices de Pauli*

$$X = tI + y_1 \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} + y_2 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad t, y_1, y_2 \in \mathbb{R} \ , \tag{9.3}$$

et constater (Exercice) que la contrainte $\lambda_{\min}(X) \geqslant 0$ est équivalente à

$$t \geqslant \sqrt{y_1^2 + y_2^2} \ .$$

Noter que pour $n \geqslant 3$, le cône $\mathcal{S}_n^+$ ne coïncide plus avec un cône de Lorentz.

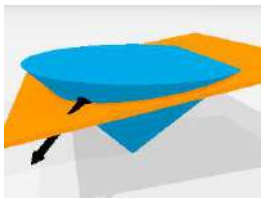
Formulation primale du problème SDP L'analogue du problème de programmation linéaire standard consiste à minimiser une forme linéaire sur l'intersection du cône $\mathcal{S}_n^+$ avec un nombre fini d'hyperplans. Afin de l'expliciter, définissons, pour toutes matrices A et B de même taille, le produit scalaire suivant dit *de Frobenius*

$$\langle A, B \rangle = \mathrm{trace}(AB^\top) = \sum_{i,j} A_{ij} B_{ij} \ .$$

Le problème linéaire sous contrainte de positivité matricielle, appelé problème SDP (pour "semi-definite program"[1]), s'écrit :

$$\underset{X \in \mathcal{S}_n^+}{\mathrm{Min}} \langle C, X \rangle; \ \ \langle A_i, X \rangle = b_i, \ i = 1, \ldots, m \ , \tag{9.4}$$

pour des matrices $C, A_1, \ldots, A_m \in \mathcal{S}_n$ et des scalaires $b_1, \ldots, b_m \in \mathbb{R}$ donnés. Un exemple de problème SDP, pour $n = 2$, est illustré sur le dessin suivant : l'ensemble des points admissibles est l'ellipse obtenue comme intersection du cône $\mathcal{S}_2^+$ avec un plan $\langle A_1, X \rangle = b_1$, l'opposé de la direction de C est représentée par une flèche, dont l'origine est le point de minimum de la fonction $X \mapsto \langle C, X \rangle$:



1. Mettons ici en garde le lecteur sur les pièges de traduction : en français, une matrice symétrique X est *positive* si $\lambda_{\min}(X) \geqslant 0$ et *définie positive* si $\lambda_{\min}(X) > 0$, alors qu'en anglais, X est "positive" si $\lambda_{\min}(X) > 0$ et "positive semi-definite" si $\lambda_{\min}(X) \geqslant 0$.

Remarque 9.1 (SDP par blocs) On peut aussi considérer un problème d'optimisation dont les variables sont plusieurs matrices positives $X_1 \in \mathcal{S}_{n_1}^+, \ldots, X_p \in \mathcal{S}_{n_p}^+$, avec $n_1, \ldots, n_p \geqslant 1$, soit

$$\underset{X_1 \in \mathcal{S}_{n_1}^+, \ldots, X_p \in \mathcal{S}_{n_p}^+}{\text{Min}} \langle C_1, X_1 \rangle + \cdots + \langle C_p, X_p \rangle; \quad \sum_{1 \leqslant j \leqslant p} \langle A_{ij}, X_j \rangle = b_i, \ i = 1, \ldots, m', \tag{9.5}$$

avec $C_i \in \mathcal{S}_{n_i}$, $A_{ij} \in \mathcal{S}_{n_j}$, et $b_i \in \mathbb{R}$.

Le problème (9.5) se ramène à (9.4). Notons en effet $X := \text{diag}(X_1, \ldots, X_p)$ la matrice bloc diagonale formée des blocs $X_1, \ldots, X_p$. La contrainte $X \succcurlyeq 0$ est équivalente à $X_i \succcurlyeq 0$, pour $1 \leqslant i \leqslant p$. D'autre part, la contrainte "X bloc diagonale" peut s'écrire $\langle X, E_{ij} \rangle = 0$, si i et j appartiennent à des blocs différents, en notant E_{ij} la matrice avec un coefficient 1 en position (i, j), et des 0 ailleurs. En tenant compte du caractère symétrique de X, on a $\langle X, E_{ij} \rangle = 0$ si et seulement si $\langle X, E_{ij} + E_{ji} \rangle = 0$, cette dernière contrainte ayant la forme requise dans un SDP. En outre, les contraintes affines apparaissant dans (9.5) peuvent s'écrire comme des contraintes affines en la matrice X. Ceci permet d'écrire le problème (9.5) sous la forme (9.4).

Cet argument montre en particulier que le problème linéaire standard (9.1) est un cas particulier du problème SDP (9.4).

On retiendra de ce qui précède la règle suivante : plusieurs inégalités de type $X_i \succcurlyeq 0$ peuvent se ramener à une seule inégalité de type $X \succcurlyeq 0$ en considérant la matrice X bloc diagonale formée à partir des X_i.

Notons qu'en pratique, il est plus efficace de traiter des SDP "par blocs" (ce qui permet d'exploiter la structure creuse). Dans un souci de simplicité, nous limitons cependant l'exposé au problème standard (9.4). •

Forme duale du problème SDP Une autre formulation du problème SDP est la suivante :

$$\underset{y \in \mathbb{R}^m}{\text{Max}} \, b \cdot y \, ; \qquad y_1 A_1 + \cdots + y_m A_m \preccurlyeq C \tag{9.6}$$

avec $b \in \mathbb{R}^m$ et $A_1, \ldots, A_m, C \in \mathcal{S}_n$. La contrainte

$$y_1 A_1 + \cdots + y_p A_m \preccurlyeq C$$

satisfaite par tout point admissible y est qualifiée d'*inégalité linéaire matricielle*, ou "*LMI*".

Nous verrons dans la section 9.2 que (9.6) est le *problème dual* de (9.4), au sens de la dualité générale en optimisation convexe, décrite dans la section 2.5.4.

Remarque 9.2 Malgré les apparences, le problème primal (9.4) et son dual (9.6) sont de même nature. En effet, notons $\mathcal{E} = \{(k, l) \mid 1 \leqslant k \leqslant l \leqslant n\}$, introduisons $y = (y_{kl}) \in \mathbb{R}^{\mathcal{E}}$, et posons $F_{kk} = E_{kk}$ et $F_{kl} = E_{kl} + E_{lk}$ pour $k \neq l$, (la matrice de base E_{kl} est définie comme dans la remarque 9.1) de sorte que toute matrice de $\mathcal{S}_n$ peut s'écrire sous la forme

$$X = \sum_{(k,l) \in \mathcal{E}} y_{kl} F_{kl} \ .$$

Le problème (9.4) se réécrit comme suit :

$$\underset{y \in \mathbb{R}^{\mathcal{E}}}{\text{Min}} \sum_{k \leqslant l} \langle C, F_{kl} \rangle y_{kl}; \ - \sum_{(k,l) \in \mathcal{E}} y_{kl} F_{kl} \preccurlyeq 0, \ \sum_{(k,l) \in \mathcal{E}} y_{kl} \langle A_i, F_{kl} \rangle \leqslant \pm b_i, i = 1, \ldots, m.$$

En posant $\mathcal{B} = (-\langle C, F_{kl} \rangle)_{(k,l) \in \mathcal{E}}$, $\mathcal{D}_{kl} = \text{diag}(\langle A_1, F_{kl} \rangle, \ldots, \langle A_m, F_{kl} \rangle)$, ainsi que $\mathcal{A}_{kl} = \text{diag}(-F_{kl}, \mathcal{D}_{kl}, -\mathcal{D}_{kl})$ et $\mathcal{C} = \text{diag}(0, b_1, \ldots, b_m, -b_1, \ldots, -b_m)$, on voit que (9.4) est équivalent à

$$\underset{y \in \mathbb{R}^{\mathcal{E}}}{\text{Max}} -\mathcal{B} \cdot y; \qquad \sum_{(k,l) \in \mathcal{E}} y_{kl} \mathcal{A}_{kl} \preccurlyeq \mathcal{C} \ ,$$

qui est bien de la forme (9.6). En guise d'exercice, nous laissons le lecteur se convaincre, que, réciproquement, le problème (9.6) peut se ramener à un problème de la forme (9.4). •

Exercice 9.3 (Complément de Schur) Soit $A \in \mathcal{S}_p$ une matrice définie positive, et

$$M = \begin{pmatrix} A & B \\ B^\top & C \end{pmatrix} \ ,$$

avec $C \in \mathcal{S}_q$ et $B \in \mathbb{R}^{p\times q}$. Montrer que

$$M \succcurlyeq 0 \iff C - B^\top A^{-1} B \succcurlyeq 0 \ .$$

(La matrice $C - B^\top A^{-1} B$ est appelée *complément de Schur*.)

Application. Soient $F \in \mathbb{R}^{r\times n}$, $g \in \mathbb{R}^n$ et $c \in \mathbb{R}$. Montrer que la contrainte convexe quadratique

$$x^\top F^\top F x \leqslant g^\top x + c \tag{9.7}$$

est équivalente à la contrainte "LMI" :

$$\begin{pmatrix} I & Fx \\ x^\top F^\top & g^\top x + c \end{pmatrix} \succcurlyeq 0 \tag{9.8}$$

Fonction barrière pour le problème SDP Les méthodes de points intérieur pour le problème de programmation linéaire standard, qui font l'objet du chapitre 10, reposent sur l'emploi de la fonction strictement convexe, dite *potentiel logarithmique*,

$$\pi(x) = -\sum_{1\leqslant i\leqslant n} \log x_i \ .$$

Les méthodes de points intérieurs, et notamment les résultats de complexité polynomiale, se généralisent au problème SDP (9.4), en prenant le potentiel :

$$\pi(X) = -\log\det X \ .$$

Pour minimiser une fonction convexe $f(X)$ sous la contrainte $X \succcurlyeq 0$, on considère, pour $\mu > 0$, la fonction

$$f_\mu(X) := f(X) + \mu\pi(X) \ .$$

Comme π est strictement convexe sur l'ensemble des matrices symétriques définies positives, le point de minimum X_μ de f_μ, s'il existe est unique. L'idée est d'approcher X_μ pour $\mu > 0$ assez petit, ce qui fournit une approximation du point de minimum de f sur $\mathcal{S}_n^+$. Nous traiterons cette méthode en détail dans le chapitre 10, dans le cas plus simple où lequel le cône $\mathcal{S}_n^+$ est remplacé par le cône standard $\mathbb{R}_+^n$ de $\mathbb{R}^n$. Nous renvoyons le lecteur à [WSV00] pour le cas du cône $\mathcal{S}_n^+$, les preuves étant trop techniques pour être traitées ici. Limitons nous à vérifier la stricte convexité du potentiel $-\log\det X$.

Proposition 9.4 *Si X et Y sont des matrices définies positives, on a*

$$\det\frac{1}{2}(X+Y) \geqslant \sqrt{\det(X)\det(Y)} \ ,$$

et l'égalité a lieu si et seulement si $X = Y$.

Démonstration. Comme une matrice symétrique diagonalise dans une base orthonormale, on peut écrire $X = UDU^\top$ où U est une matrice orthogonale (i.e. $UU^\top = U^\top U = I$) et $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ est la matrice diagonale formée des valeurs propres de X. Posons $\sqrt{D} := \mathrm{diag}(\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_n})$. L'inégalité à prouver pour X et Y est équivalente à la même inégalité pour les matrices :

$$X' = \sqrt{D}^{-1} U^\top X U \sqrt{D}^{-1} = I \text{ et } Y' = \sqrt{D}^{-1} U^\top Y U \sqrt{D}^{-1} \ .$$

On peut donc supposer que $X = I$. On peut écrire $Y = VD'V^\top$ avec V orthogonale et D' diagonale. Quitte à considérer $X' = V^\top XV = I$ et $Y' = V^\top YV$ au lieu de X et Y, on peut supposer que $X = I$ et que Y est diagonale, soit $Y = \mathrm{diag}(\mu_1, \ldots, \mu_n)$. Dans ce cas, l'inégalité demandée est conséquence de

$$\frac{1}{2}(1 + \mu_i) \geqslant \sqrt{\mu_i}$$

ce qui est vrai car la moyenne arithmétique majore la moyenne géométrique. Le cas d'égalité est laissé au lecteur. ■

Corollaire 9.5 *La fonction $X \mapsto -\log\det X$ est strictement convexe sur l'ensemble des matrices symétriques définies positives.* ■

9.2 Dualité SDP

9.2.1 Dualité faible

Associons au problème SDP sous forme primale (9.4) le lagrangien suivant :

$$L(X; Y, \lambda) = \langle C, X\rangle - \langle X, Y\rangle + \sum_{1\leqslant i\leqslant m} \lambda_i(b_i - \langle A_i, X\rangle) \ ,$$

qui est fonction des variables $X, Y \in \mathcal{S}_n$ et $\lambda \in \mathbb{R}^m$. Ce choix de lagrangien est justifié par la proposition :

Proposition 9.6 *Le problème SDP sous forme primale* (9.4) *est équivalent au problème :*

$$\operatorname*{Min}_{X\in\mathcal{S}_n} \sup_{Y\in\mathcal{S}_n^+, \lambda\in\mathbb{R}^m} L(X; Y, \lambda) \tag{9.9}$$

Afin de démontrer cette proposition, rappelons la notion de cône polaire positif, introduite en section 2.5.5, noté dans ce chapitre par une étoile afin d'éviter la confusion avec l'ensemble des matrices positives :

$$K^* = \{y \in \mathbb{R}^k \mid \forall x \in K, \ \ x \cdot y \geqslant 0\} \ .$$

Le cône $K = \mathbb{R}^n_+$ est un exemple de cône auto-polaire : $(\mathbb{R}^n_+)^* = \mathbb{R}^n_+$. Il en est de même du cône des matrices symétriques positives.

Lemme 9.7 *On a $\mathcal{S}_n^+ = (\mathcal{S}_n^+)^*$.*

Afin d'établir ce lemme, rappelons deux résultats d'algèbre linéaire, que nous énonçons sous une forme permettant de les réexploiter dans la suite.

Lemme 9.8 *Une matrice $X \in \mathcal{S}_n$ peut s'écrire*

$$X = \sum_{1\leqslant k\leqslant n} \lambda_k v^k (v^k)^\top \ ,$$

où les λ^k sont les valeurs propres de X, et où les v^k forment une famille orthonormale, telle que v^k soit vecteur propre de X pour la valeur propre λ^k.

Démonstration. Le théorème spectral affirme qu'une matrice symétrique diagonalise par un changement de base orthonormal, ce qui s'écrit $X = UDU^\top$ avec $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_k)$, et $UU^\top = U^\top U = I$. En prenant pour v^k la k-ième colonne de U, on a la décomposition annoncée. ■

Si $X \in \mathcal{S}_n$, nous notons $\mathrm{Im}\, X = \{Xu \mid u \in \mathbb{R}^n\}$ et $\mathrm{Ker}\, X = \{u \mid Xu = 0\}$.

Lemme 9.9 *Toute matrice $X \in \mathcal{S}_n^+$ de rang r peut s'écrire :*

$$X = \sum_{1 \leqslant k \leqslant r} v^k (v^k)^\top, \tag{9.10}$$

où $v^1, \ldots, v^r$ forment une famille orthogonale de vecteurs non-nuls de $\mathbb{R}^n$. D'autre part, si X est de la forme (9.10)*, pour une famille quelconque de vecteurs $v^1, \ldots, v^r \in \mathbb{R}^n$, et si V désigne l'espace vectoriel engendré par $v^1, \ldots, v^r$, on a $\operatorname{Im} X = V$, et $\operatorname{Ker} X = V^\perp$. En particulier, $\operatorname{rg}(X) = \dim V \leqslant r$.*

Démonstration. Soit $X \in \mathcal{S}_n^+$. D'après le lemme 9.8, on peut écrire

$$X = \sum_{1 \leqslant k \leqslant n} \lambda_k u^k (u^k)^\top \ ,$$

avec λ_k valeur propre de X et u^k vecteur propre associé. On a $\lambda_k \geqslant 0$ car $X \succcurlyeq 0$. Quitte à réordonner les valeurs propres, on peut supposer $\lambda_1, \ldots, \lambda_r > 0$ et $\lambda_{r+1} = \cdots = \lambda_n = 0$. En posant $v^k = \sqrt{\lambda_k} u^k$, on a bien une représentation de X sous la forme (9.10), les vecteurs non-nuls $v^1, \ldots, v^r$ formant une famille orthogonale.

Supposons maintenant X de la forme (9.10), et soit V l'espace vectoriel engendré par $v^1, \ldots, v^r$. Tout vecteur y de l'image de X est de la forme $y = Xz$, donc $y = \sum_{1 \leqslant k \leqslant r} v^k (v^k \cdot z) \in V$, donc $\operatorname{Im} X \subset V$. Tout vecteur y du noyau de X vérifie $0 = y^\top X y = \sum_{1 \leqslant k \leqslant r} (v^k \cdot y)^2$, donc $\operatorname{Ker} X \subset V^\top$. Notons s le rang de X et t la dimension de V. De $\operatorname{Im} X \subset V$, on déduit $s \leqslant t$. De $\operatorname{Ker} X \subset V^\top$, on déduit $n - s \leqslant n - t$. Donc $s = t$, $\operatorname{Im} X = V$, et $\operatorname{Ker} X = V^\top$. ■

Démonstration du lemme 9.7. Soit $X \in (\mathcal{S}_n^+)^*$. Alors, pour tout $u \in \mathbb{R}^n$, $u^\top X u = \langle X, uu^\top \rangle \geqslant 0$, donc $X \in \mathcal{S}_n^+$. Ainsi, $(\mathcal{S}_n^+)^* \subset \mathcal{S}_n^+$. Soit $X \in \mathcal{S}_n^+$. D'après le lemme 9.9, tout $Y \in \mathcal{S}_n^+$ peut s'écrire comme somme d'un nombre fini de matrices de la forme $u^\top u$, avec $u \in \mathbb{R}^n$. Comme $\langle X, u^\top u \rangle = u^\top X u \geqslant 0$, on déduit que $\langle X, Y \rangle \geqslant 0$ pour tout $Y \in \mathcal{S}_n^+$, et donc $\mathcal{S}_n^+ \subset (\mathcal{S}_n^+)^*$. ■

Corollaire 9.10 *Pour toute matrice $X \in \mathcal{S}_n$, on a*

$$\inf_{Y \in \mathcal{S}_n^+} \langle X, Y \rangle = \begin{cases} 0 & \text{si } X \in \mathcal{S}_n^+ \\ -\infty & \text{sinon.} \end{cases} \tag{9.11}$$

Démonstration. Si $X \in \mathcal{S}_n^+$, il résulte du lemme 9.7 que $\langle X, Y \rangle \geqslant 0$ pour tout $Y \in \mathcal{S}_n^+$. Si $X \in \mathcal{S}_n \setminus \mathcal{S}_n^+$, il résulte à nouveau du lemme 9.7 qu'il existe $Z \in \mathcal{S}_n^+$ telle que $\langle X, Z \rangle < 0$. Donc

$$\inf_{Y \in \mathcal{S}_n^+} \langle X, Y \rangle \leqslant \inf_{t > 0} \langle X, tZ \rangle = \lim_{t \to \infty} t \langle X, Z \rangle = -\infty \ .$$

■

Démonstration de la Proposition 9.6. Cela résulte aussitôt du corollaire 9.10. ■

Le problème dual de (9.4) s'obtient formellement en commutant le sup et l'inf dans (9.9), soit

$$\begin{aligned} \operatorname*{Max}_{Y \in \mathcal{S}_n^+, \lambda \in \mathbb{R}^m} \inf_{X \in \mathcal{S}_n} L(X; Y, \lambda) &= \operatorname*{Max}_{Y \in \mathcal{S}_n^+, \lambda \in \mathbb{R}^m} \inf_{X \in \mathcal{S}_n} \lambda \cdot b + \langle C - Y - \sum_{1 \leqslant i \leqslant m} \lambda_i A_i, X \rangle \\ &= \operatorname*{Max}_{Y \in \mathcal{S}_n^+, \lambda \in \mathbb{R}^m} \lambda \cdot b \ ; \qquad C - Y - \sum_{1 \leqslant i \leqslant m} \lambda_i A_i = 0 \\ &= \operatorname*{Max}_{\lambda \in \mathbb{R}^m} \lambda \cdot b \ ; \qquad C \succcurlyeq \sum_{1 \leqslant i \leqslant m} \lambda_i A_i \ . \end{aligned}$$

On reconnaît la forme (9.6) du problème SDP, ce qui justifie le nom de "formulation duale" pour (9.6). Comme le sup des infs est inférieur à l'inf des sups (lemme 2.41), on a :

Proposition 9.11 (Dualité faible SDP). *La valeur du problème primal* (9.4) *SDP majore celle du problème dual* (9.6) *associé.* ■

9.2.2 Dualité forte

En programmation linéaire, primal et dual ont toujours même valeur, sauf dans le cas dégénéré où les ensembles admissibles du primal et du dual sont tous deux vides. En outre, un programme linéaire dont la valeur est finie a toujours une solution optimale. Il n'en est pas de même pour les SDP.

Exercice 9.12 (Absence de solution) Considérons le SDP sous forme primale

$$\operatorname*{Min}_{X\in\mathcal{S}_2^+} X_{11}\ ; \qquad X_{12}=1\ . \tag{9.12}$$

Montrer que la valeur de ce problème est 0, mais qu'il n'existe pas de solution optimale. Expliciter le problème dual : montrer que sa valeur est également 0 et qu'il admet une solution optimale.

Exercice 9.13 (Saut de dualité) Considérons le SDP sous forme primale

$$\operatorname*{Min}_{X\in\mathcal{S}_2^+} -X_{12}-X_{22}\ ; \qquad X_{11}+X_{22}+2X_{12}=0\ . \tag{9.13}$$

Montrer que la valeur ce problème est 0 et que le problème dual n'admet aucun point admissible, de sorte que la valeur du dual est $-\infty$.

Afin d'énoncer un résultat de dualité forte, écrivons la contrainte du SDP dual (9.6) sous la forme

$$\sum_{1\leqslant i\leqslant m} y_iA_i+Z=C;\quad Z\in\mathcal{S}_n^+. \tag{9.14}$$

Théorème 9.14 (Dualité forte pour les SDP) *Supposons l'existence de $\bar{y}\in\mathbb{R}^m$ ainsi que d'une matrice définie positive $\bar{Z}$ solution de* (9.14)*. Alors la valeur du primal* (9.4) *et la valeur du dual* (9.6) *coïncident, et si le primal est réalisable, il admet une solution optimale.*

Démonstration. Soit d la valeur du dual. Comme le point $\bar{y}$ est admissible pour le dual, on a $d>-\infty$. En raison de l'inégalité de dualité faible, le seul cas à considérer est celui où d est finie. Il faut montrer que le système suivant a une solution :

$$\text{Il existe}\quad X\in\mathcal{S}_n^+;\quad \langle C,X\rangle=d;\quad \langle A_i,X\rangle=b_i,\ i=1,\dots,m. \tag{9.15}$$

Pour $X\in\mathcal{S}_n$, notons $\mathcal{B}X:=\langle C,X\rangle,\langle A_1,X\rangle,\dots,\langle A_m,X\rangle$, et posons $g:=(d,b_1,\dots,b_m)^\top$ et $\mathcal{K}:=\mathcal{B}\mathcal{S}_n^+$. La relation (9.15) équivaut à $g\in\mathcal{K}$. Si ce n'est pas le cas, appliquons le lemme 9.15 énoncé ci-dessous aux matrices $B_1=A_1,\dots,B_m=A_m,B_{m+1}=C$. L'existence de $\bar{y},\bar{Z}$ solution de (9.14) avec $\bar{Z}$ défini positif entraîne que l'hypothèse du lemme 9.15 est vérifiée. Le cône $\mathcal{K}$ est donc fermé. Il est évidemment convexe. Comme $g\notin\mathcal{K}$, on peut séparer strictement g de $\mathcal{K}$, et ainsi trouver $\alpha\in\mathbb{R}$ et $\beta\in\mathbb{R}^m$ tels que

$$\alpha d+\beta\cdot b<0$$

et

$$\langle\alpha C+\beta_1A_1+\cdots+\beta_mA_m,X\rangle\geqslant 0,\qquad\forall X\in\mathcal{S}_n^+\ .$$

En vertu de $(\mathcal{S}_n^+)^*=\mathcal{S}_n$, cette dernière condition se réécrit

$$\alpha C+\beta_1A_1+\cdots+\beta_mA_m\succcurlyeq 0\ .$$

Quitte à diviser β par $|\alpha|$ si $\alpha\neq 0$, on peut supposer que $\alpha\in\{0,\pm1\}$.

Si $\alpha=0$, on a $b\cdot\beta<0$, et comme pour tout scalaire $t\geqslant 0$, $y^t:=\bar{y}-t\beta$ est un point admissible du dual, avec $b\cdot y^t\to+\infty$ quand $t\to+\infty$, la valeur du dual est $d=+\infty$, une contradiction.

Si $\alpha=1$, $y:=-\beta$ est un point admissible du dual, de valeur $b\cdot y>d$, ce qui contredit le fait que d est la valeur du dual.

Reste à considérer le cas où $\alpha = -1$, de sorte que $\beta \cdot b < d$. Choisissons un point admissible du dual y tel que $\beta \cdot b < y \cdot b < d$. Observons que pour tout $t \geqslant 0$, $y^t := \bar{y} + t(y - \beta)$ est encore un point admissible du dual. On a $b \cdot (y - \beta) > 0$, et donc $b \cdot y^t \to +\infty$ quand $t \to +\infty$, ce qui montre que $d = +\infty$, une contradiction. La conclusion s'ensuit.

■

Lemme 9.15 *Soient $B_1, \ldots, B_p \in \mathcal{S}_n$, et soit $\mathcal{B}$ l'opérateur linéaire de $\mathcal{S}_n$ dans $\mathbb{R}^p$ associant à X le vecteur de coordonnées $\langle B_i, X \rangle$, $i = 1, \ldots, p$. S'il existe un vecteur z tel que la matrice $\sum_{1 \leqslant i \leqslant p} z_i B_i$ soit définie positive, alors, l'image de $\mathcal{S}_n^+$ par $\mathcal{B}$ est fermée.*

Démonstration. Soit X^k une suite de matrices de $\mathcal{S}_n^+$ telle que $\mathcal{B}X^k$ converge vers un vecteur b. On a $z \cdot \mathcal{B}X^k = \sum_{1 \leqslant i \leqslant p} \langle z_i B_i, X^k \rangle = \langle F, X^k \rangle$, où $F := \sum_{1 \leqslant i \leqslant p} z_i B_i$ est définie positive. Notons λ_{kl}, pour $l = 1, \ldots, n$, les valeurs propres de X^k, et soit $(u^{kl})_{1 \leqslant k \leqslant n}$ la famille orthonormée de vecteurs propres correspondants, de sorte que, d'après le lemme 9.8,

$$X^k = \sum_{1 \leqslant l \leqslant n} \lambda_{kl} u^{kl} (u^{kl})^\top \ . \tag{9.16}$$

On a

$$\langle F, X^k \rangle = \sum_{1 \leqslant l \leqslant n} \lambda_{kl} (u^{kl})^\top F u^{kl} \geqslant \sum_{1 \leqslant l \leqslant n} \lambda_{kl} \lambda_{\min}(F) \ . \tag{9.17}$$

où $\lambda_{\min}(F) > 0$ désigne la plus petite valeur propre de F, car $u^\top F u \geqslant \lambda_{\min} \|u\|^2$, pour tout vecteur u, et $\|u^{kl}\| = 1$. Comme X^k est positive, les λ_{kl} sont positifs. Comme $\mathcal{B}X^k$ converge vers b, $\langle F, X^k \rangle = z \cdot \mathcal{B}X^k$ converge. On déduit de (9.17) que les valeurs propres λ_{kl} sont majorées. Les vecteurs u^{kl}, qui sont unitaires, sont bornés indépendamment de k. Il résulte de (9.16) que la suite X^k est bornée. Quitte à prendre une sous suite, on peut supposer que X^k converge vers une matrice X, qui appartient à $\mathcal{S}_n^+$ car $\mathcal{S}_n^+$ est fermé. Par continuité de $\mathcal{B}$, on a $b = \mathcal{B}X$, ce qui montre que $\mathcal{B}(\mathcal{S}_n^+)$ est fermé. ■

Remarque 9.16 Le théorème 9.14 est un cas particulier du corollaire 2.80. La formulation "primale" de ce dernier correspond à la formulation "duale" du théorème 9.14. On peut également appliquer le corollaire 2.81, qui sous une hypothèse primale assure l'absence de saut de dualité et l'existence de solutions duales. •

Exercice 9.17 (Réécriture basée sur le cône de Lorentz) On va montrer comment certains problèmes non linéaires peuvent se réécrire comme des problèmes linéaires (avec critère et contraintes linéaires) et contraintes de cône de Lorentz.

1. *Contrainte de carré de norme.* Soient $w \in \mathbb{R}^n$, α et β scalaires. Montrer que

$$\left\{\alpha \geqslant 0,\ \beta \geqslant 0,\ \|w\|^2 \leqslant \alpha\beta\right\} \ \Leftrightarrow\ \alpha + \beta \geqslant \left\| \begin{pmatrix} 2w \\ \alpha - \beta \end{pmatrix} \right\|. \tag{9.18}$$

2. *Critères inverses.* Réécrire linéairement avec cône de Lorentz le problème

$$\begin{array}{ll} \underset{x \in \mathbb{R}^n}{\text{Min}} & \displaystyle\sum_{i=1}^{p} 1/(a_i \cdot x + b_i); \\ & a_i \cdot x + b_i > 0, \quad i = 1, \ldots, p, \\ & c_i \cdot x + d_i \geqslant 0, \quad i = 1, \ldots, q, \end{array} \tag{9.19}$$

avec les a_i, c_i, vecteurs de $\mathbb{R}^n$, et b, d vecteurs de $\mathbb{R}^p$ et $\mathbb{R}^q$.

3. *Approximation uniforme en échelle logarithmique.* Réécrire linéairement avec cône de Lorentz le problème d'approximation uniforme, en échelle logarithmique :

$$\operatorname*{Min}_{x\in\mathbb{R}^n} \max |\log(a_i \cdot x) - \log(b_i)| \tag{9.20}$$

avec $b_i > 0$ pour tout i, et la contrainte implicite $a_i \cdot x > 0$ pour tout i.

4. *Puissances fractionnaires I.* Soient ℓ un entier strictement positif. Réécrire linéairement avec cône de Lorentz la relation

$$x \in \mathbb{R}^{2^\ell}_+; \quad t \in \mathbb{R}; \quad t \leqslant (x_1 x_2 \cdots x_{2^\ell})^{1/2^\ell} . \tag{9.21}$$

5. *Puissances fractionnaires II.* Réécrire linéairement avec cône de Lorentz la relation

$$x \in \mathbb{R}^n_+; \quad t \in \mathbb{R}_+; \quad t \leqslant x_1^{\pi_1} x_2^{\pi_2} \cdots x_n^{\pi_n}. \tag{9.22}$$

On suppose les exposant de la forme $\pi_i = p_i/p$, avec p_i entier strictement positifs et p entier, $p \geqslant \sum_i p_i$.

Exercice 9.18 Maximisation de forme quadratique sous contrainte de boîte). Soit A une matrice symétrique réelle, et soit

$$B_n = \{x \in \mathbb{R}^n \mid \|x\|_\infty \leqslant 1\}$$

l'hypercube unité. On s'intéresse au problème d'optimisation :

$$\operatorname{Max} x^\top A x; \;\; x \in B_n \;\; .$$

Montrer que la valeur de ce problème est majorée par la valeur du problème SDP

$$\operatorname{Min} \sum_{1\leqslant i\leqslant n} \mu_i; \; \mu \in (\mathbb{R}_+)^n, \; A - \operatorname{diag}(\mu) \preccurlyeq 0 \;\; .$$

9.3 Quelques problèmes combinatoires où interviennent des SDP

9.3.1 Problème de la coupe maximale

Reprenons le problème de la coupe maximale, décrit dans l'exemple 1.22. Étant donné un graphe non-orienté $(\mathcal{V}, \mathcal{E})$, chaque arête $\{i, j\} \in \mathcal{E}$ étant munie d'une capacité c_{ij}, on cherche une bi-partition non-triviale :

$$\mathcal{V} = I^- \cup I^+, \; I^- \cap I^+ = \emptyset, \; I^- \neq \emptyset, \; I^+ \neq \emptyset \tag{9.23}$$

maximisant

$$c(I^-, I^+) := \sum_{\{i,j\}\in\mathcal{E}, \; i\in I^-, j\in I^+} c_{ij} \;\; .$$

On dit que l'ensemble des arêtes reliant I^- et I^+ est une *coupe*, et que $c(I^-, I^+)$ est la capacité de la coupe.

Remarque 9.19 Le cas particulier où $c_{ij} \leqslant 0$, pour tout $\{i, j\} \in \mathcal{E}$, est facile : on se ramène à *minimiser* la capacité d'une coupe dans un graphe dont les arêtes sont munies de capacités $-c_{ij} \geqslant 0$. On retrouve ainsi le problème de la coupe minimale (globale), qui, comme nous l'avons vu dans l'exercice 5.2, peut être résolu notamment au moyen d'algorithmes de flots. Nous renvoyons à nouveau à [CGK+96] pour un état de l'art. •

Nous nous intéressons ici au problème général de la coupe maximale (qui est NP-difficile). Pour l'aborder, associons à toute partition (9.23) le vecteur de signe $x \in \{\pm 1\}^{\mathcal{E}}$,

$$x_i = \pm 1 \text{ si } i \in I^\pm \;\; .$$

Posons $c_{ij} = 0$ si $\{i,j\} \notin \mathcal{E}$. La capacité de la coupe (I^-, I^+) s'écrit :

$$c(I^-, I^+) = \sum_{i<j} c_{ij} \frac{1 - x_i x_j}{2} = \frac{1}{4} \sum_{i,j} c_{ij}(1 - x_i x_j) \tag{9.24}$$

$$= \frac{1}{4}\Big(\sum_{ij} c_{ij} x_i x_i - \sum_{ij} c_{ij} x_i x_j \Big) = \frac{1}{4} x^\top L x \tag{9.25}$$

où L désigne la matrice telle que

$$L_{ii} = \sum_j c_{ij}, \qquad L_{ij} = -c_{ij} \ .$$

La matrice $-L$ est appelée *Laplacien* du graphe $\mathcal{G}$ (relativement aux capacités c_{ij}). Nous venons de montrer :

Proposition 9.20 *Le problème de la coupe maximale est équivalent à :*

$$\max_{x \in \{\pm 1\}^\nu} x^\top L x \ ,$$

où $-L$ est le Laplacien associé au graphe. ■

Ceci suggère d'étudier plus généralement, pour toute matrice $C \in \mathcal{S}^n$, le problème :

$$\max_{x \in \{\pm 1\}^n} x^\top C x \ . \tag{9.26}$$

La proposition suivante formule la généralisation (9.26) du problème de la coupe maximale comme un problème SDP, sous contrainte de rang. En oubliant la contrainte de rang, on obtient un majorant calculable par une méthode de points intérieurs :

Proposition 9.21 Relaxation SDP du problème de la coupe maximale). *Toute matrice $C \in \mathcal{S}_n$ vérifie*

$$\max_{x \in \{\pm 1\}^n} x^\top C x = \max_{X \in \mathcal{S}_n^+} \langle C, X \rangle \qquad X_{ii} = 1 \ \textit{pour tout } i, \ \ \mathrm{rg}(X) = 1 \ . \tag{9.27}$$

En particulier, on a la majoration :

$$\max_{x \in \{\pm 1\}^n} x^\top C x \leqslant \sup_{X \in \mathcal{S}_n^+} \langle C, X \rangle \qquad X_{ii} = 1 \ \textit{pour tout } i. \tag{9.28}$$

Le problème à droite de (9.28) est appelé *relaxation SDP* du problème (9.26). Afin d'établir cette proposition, énonçons le cas particulier du lemme 9.9 :

Corollaire 9.22 *Les matrices de $\mathcal{S}_n^+$ qui sont de rang 1 sont précisément les matrices de la forme $xx^\top$, où x est un vecteur non-nul de $\mathbb{R}^n$.* ■

Démonstration de la Proposition 9.21. Associons à tout $x \in \{\pm 1\}^n$ la matrice positive de rang 1 :

$$X = xx^\top \ .$$

Observons que $X_{ii} = 1$, pour tout i. Comme

$$x^\top C x = \langle C, X \rangle \ ,$$

on a aussitôt

$$\max_{x \in \{\pm 1\}^n} x^\top C x \leqslant \max_{X \in \mathcal{S}_n^+} \langle C, X \rangle \qquad X_{ii} = 1, \forall i, \ \ \mathrm{rg}(X) = 1 \ .$$

Réciproquement, si X est positive de rang 1, d'après le Corollaire 9.22, on peut écrire $X = xx^\top$, avec $x \in \mathbb{R}^n$. Si $X_{ii} = 1$, on a $x_i^2 = 1$, et donc $x_i = \pm 1$, ce qui démontre l'égalité (9.27). L'inégalité (9.28) en découle aussitôt. ■

Exercice 9.23 Montrer que les directions extrêmes du cône $\mathcal{S}_n^+$ sont engendrées par les matrices de la forme $xx^\top$, où x est un vecteur non-nul de $\mathbb{R}^n$.

Remarque 9.24 Le terme Laplacien fait peut être justifié en considérant par exemple le cycle d'ordre n, et des capacités unitaires. Dans ce cas,

$$-L = \begin{pmatrix} -2 & 1 & & & & 1 \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ 1 & & & & 1 & -2 \end{pmatrix}$$

est la matrice de discrétisation de l'opérateur $u \mapsto u''$ (avec conditions aux limites cycliques). Plus généralement, pour un graphe de type grille de dimension k, on retrouve la discrétisation du Laplacien en dimension k. •

9.3.2 Clique, ensembles stables, et capacité ϑ de Lovasz

Soit $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ un graphe non-orienté. Nous avons déjà rencontré (section 7.3.1) les notions de clique et d'ensemble indépendant ou stable, que nous reprenons maintenant. Rappelons qu'une *clique* de $\mathcal{G}$ est un sous-ensemble $\mathcal{C}$ de sommets de $\mathcal{G}$ tel que deux sommets quelconques de $\mathcal{C}$ sont reliés par un arc de $\mathcal{G}$:

$$i \neq j, \;\; i,j \in \mathcal{C} \implies \{i,j\} \in \mathcal{E} \;\; .$$

On note $\omega(\mathcal{G})$ le cardinal maximal d'une clique.

Un sous-ensemble *indépendant* ou *stable* est un sous-ensemble $\mathcal{S} \subset \mathcal{E}$ tel que

$$i \neq j, \;\; i,j \in \mathcal{S} \implies \{i,j\} \notin \mathcal{E} \;\; .$$

On appelle *nombre de stabilité*, noté $\alpha(\mathcal{G})$ le cardinal maximal d'un ensemble indépendant. Notons $\bar{\mathcal{G}} = (\mathcal{V}, \bar{\mathcal{E}})$ le *graphe complémentaire* de $\mathcal{G}$, tel que $\bar{\mathcal{E}} = \{\{i,j\} \mid i,j \in \mathcal{V}, i \neq j, \{i,j\} \notin \mathcal{E}\}$. Comme les ensembles indépendants de $\mathcal{G}$ sont les cliques de $\bar{\mathcal{G}}$, on a

$$\alpha(\mathcal{G}) = \omega(\bar{\mathcal{G}}) \;\; . \tag{9.29}$$

Le problème du calcul de $\alpha(\mathcal{G})$ peut se voir comme un cas très particulier de mesure de degré de parallélisme. Imaginons un atelier avec un ensemble de tâches $\mathcal{V}$, et relions deux tâches i et j par une arête si i et j sont dépendantes, ce qui signifie qu'elles utilisent une ressource commune (machines, palettes, personnels, etc.). Dans ce cas, $\alpha(\mathcal{G})$ mesure le nombre maximal de tâches qui peuvent être exécutées simultanément dans l'atelier.

On appelle *capacité de Lovasz* la quantité :

$$\vartheta(\mathcal{G}) := \inf_{A \in \mathcal{S}_n} \left\{ \lambda_{\max}(A); \quad A_{ii} = 1, \forall i \in \mathcal{V}, \; A_{ij} = 1, \; \forall \{i,j\} \in \bar{\mathcal{E}} \right\} \;\; , \tag{9.30}$$

où $\lambda_{\max}(A)$ désigne la plus grande valeur propre d'une matrice symétrique A. Le problème (9.30) est bien de type SDP. En introduisant une variable supplémentaire scalaire, t, on a en effet :

$$\vartheta(\mathcal{G}) = \inf_{A \in \mathcal{S}_n, t \in \mathbb{R}} \left\{ t; \quad tI - A \succcurlyeq 0, \;\; A_{ii} = 1, \forall i \in \mathcal{V}, \; A_{ij} = 1, \; \forall \{i,j\} \in \bar{\mathcal{E}} \right\} \;\; , \tag{9.31}$$

où I désigne la matrice identité. Ce problème est de la forme (9.6), la collection des variables y_k étant ici constituée des variables A_{ij}, pour $\{i,j\} \in \mathcal{E}$, ainsi que de la variable scalaire t.

Théorème 9.25 *Pour tout graphe non-orienté $\mathcal{G}$, on a*

$$\alpha(\mathcal{G}) \leqslant \vartheta(\mathcal{G}) \ . \tag{9.32}$$

Afin de démontrer ce résultat, faisons l'observation élémentaire :

Lemme 9.26 *Si $A \in \mathcal{S}_p$ est une sous-matrice principale d'une matrice $B \in \mathcal{S}_n$, on a*

$$\lambda_{\max}(A) \leqslant \lambda_{\max}(B) \ .$$

Démonstration. Pour toute matrice symétrique C, on a

$$\lambda_{\max}(C) = \sup_{u \neq 0} \frac{u^\top C u}{\|u\|^2} \ .$$

On peut supposer que la matrice A est formée des p premières lignes et colonnes de B. On a

$$\begin{aligned} \lambda_{\max}(A) &= \sup_{v \in \mathbb{R}^p \setminus \{0\}} \frac{v^\top A v}{\|v\|^2} = \sup_{v \in \mathbb{R}^p \setminus \{0\}} \frac{\begin{pmatrix} v \\ 0 \end{pmatrix}^\top C \begin{pmatrix} v \\ 0 \end{pmatrix}}{\|v\|^2} \\ &\leqslant \sup_{u \in \mathbb{R}^n \setminus \{0\}} \frac{u^\top C u}{\|u\|^2} = \lambda_{\max}(C) \ . \end{aligned}$$

■

Démonstration du Théorème 9.25. Si $\mathcal{S}$ est un ensemble indépendant, alors, pour toute matrice A vérifiant les contraintes (9.30), la sous-matrice B de A obtenue en se restreignant aux lignes et colonnes de $\mathcal{S}$ a toutes ses composantes égales à 1. Les valeurs propres de B sont $|\mathcal{S}|$ (de multiplicité 1), et 0 (de multiplicité $|\mathcal{S}|-1$). Donc $\lambda_{\max}(B) = \max(0, |\mathcal{S}|) = |\mathcal{S}|$. D'après le lemme 9.26, $\lambda_{\max}(B) \leqslant \lambda_{\max}(A)$, et donc $|\mathcal{S}| \leqslant \lambda_{\max}(A)$. Il en résulte que $\alpha(\mathcal{G}) \leqslant \theta(\mathcal{G})$. ■

Exemple 9.27 Soit $\mathcal{G}$ la chaîne à trois sommets :

Sa capacité de Lovasz est donnée par :

$$\vartheta(\mathcal{G}) = \inf_{x_{12}, x_{23} \in \mathbb{R}} \lambda_{\max}(A(x_{12}, x_{23})) \qquad \text{avec } A(x_{12}, x_{23}) := \begin{pmatrix} 1 & x_{12} & 1 \\ x_{12} & 1 & x_{23} \\ 1 & x_{23} & 1 \end{pmatrix}$$

Cette chaîne admet un ensemble indépendant de cardinal 2 (formé des extrémités de la chaîne). D'après le théorème 9.25, on a $\vartheta(\mathcal{G}) \geqslant 2$. En prenant $x_{12} = x_{23} = 0$, on obtient une matrice $A(x)$ telle que $\lambda_{\max}(A(x)) = 2$. Donc $\vartheta(\mathcal{G}) = 2$.

9.3.3 Nombre chromatique d'un graphe

Soit $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ un graphe non-orienté. Le *nombre chromatique* de $\mathcal{G}$, noté $\chi(G)$, est le nombre minimal de couleurs nécessaires pour colorier tous les sommets de $\mathcal{G}$, de telle sorte que deux sommets adjacents n'aient pas la même couleur.

Le problème de calcul du nombre chromatique peut être vu comme une version très purifiée du problème d'*allocation de fréquence* : une couleur représente une fréquence, les sommets représentent des émetteurs, la contrainte de coloriage exprime que deux émetteurs voisins ont des fréquence différentes (en pratique on demande plutôt que des émetteurs voisins aient des fréquences suffisamment éloignées).

Pour colorier une clique, il faut autant de couleurs que la clique a de sommets, et donc, pour tout graphe $\mathcal{G}$,

$$\omega(\mathcal{G}) \leqslant \chi(\mathcal{G}) \ . \tag{9.33}$$

Nous allons améliorer cette minoration de $\chi(\mathcal{G})$.

Théorème 9.28 *Pour tout graphe non-orienté $\mathcal{G}$, on a*

$$\vartheta(\mathcal{G}) \leqslant \chi(\bar{\mathcal{G}}) \ . \tag{9.34}$$

Avant de démontrer ce théorème, énonçons le corollaire obtenu à l'aide du théorème 9.25 et de l'identité (9.29).

Corollaire 9.29 (du sandwich) *Pour tout graphe non-orienté $\mathcal{G}$, on a*

$$\omega(\mathcal{G}) \leqslant \vartheta(\bar{\mathcal{G}}) \leqslant \chi(\mathcal{G}) \ . \tag{9.35}$$

L'intérêt de ces inégalités est que $\vartheta(\bar{\mathcal{G}})$ est facilement calculable, alors que $\omega(\mathcal{G})$ et $\chi(\mathcal{G})$ ne le sont probablement pas (calculer $\omega(G)$ et calculer $\chi(\mathcal{G})$ sont des problèmes NP-durs).

Afin de montrer le théorème 9.28, donnons une formulation duale de la capacité ϑ de Lovasz.

Proposition 9.30 *On a*

$$\vartheta(\mathcal{G}) = \max_{X \in \mathcal{S}_n^+} \Big\{ \sum_{i,j \in \mathcal{V}} X_{ij} \ ; \qquad X_{ij} = 0, \ \textit{pour tout} \ i,j \in \mathcal{E}, \qquad \sum_{i \in \mathcal{V}} X_{ii} = 1 \Big\} \tag{9.36}$$

Démonstration. Soit E_{ij} la matrice de base définie dans la remarque 9.1. Nous posons $F_{ij} = E_{ij} + E_{ji}$ pour $j \neq i$. En faisant apparaître les degrés de liberté $y_{kl} := X_{kl}$, avec $(k,l) \in \mathcal{E}$, dans la formulation (9.31), on obtient :

$$-\vartheta(\mathcal{G}) = \max_{y \in \mathbb{R}^{\mathcal{E}}, t \in \mathbb{R}} \Big\{ -t \ ; \qquad \sum_{(i,j) \in \bar{\mathcal{E}}} F_{ij} + \sum_{i \in \mathcal{V}} E_{ii} + \sum_{(i,j) \in \mathcal{E}} y_{ij} F_{ij} - tI \preccurlyeq 0 \Big\} \ . \tag{9.37}$$

Ce problème est le problème dual du problème de minimisation ci-dessous :

$$\operatorname*{Min}_{X \in \mathcal{S}_n^+} - \sum_{(i,j) \in \bar{\mathcal{E}}} X_{ij} - \sum_{i \in \mathcal{V}} X_{ii} \ ; \quad X_{ij} = 0 \text{ pour tout } \ (i,j) \in \mathcal{E}, \quad \sum_{i \in \mathcal{V}} X_{ii} = 1. \tag{9.38}$$

Prenons un $y \in \mathbb{R}^{\mathcal{E}}$ quelconque. Pour t assez grand, (y,t) est un point admissible de (9.37) tel que la variable d'écart Z solution de

$$\sum_{(i,j) \in \mathcal{E}} y_{ij} F_{ij} - tI + Z = - \sum_{(i,j) \in \bar{\mathcal{E}}} F_{ij} \ ,$$

soit définie positive. En raison du théorème de dualité forte 9.14, les problèmes (9.37) et (9.38) ont même valeur. L'égalité (9.36) en résulte. ■

Démonstration du théorème 9.28. Notons n le nombre de sommets. Soit X un point admissible de la formulation (9.36). D'après le Lemme 9.9, on peut écrire $X = VV^\top$, avec $V \in \mathbb{R}^{n \times r}$, et en notant z^i la ligne i de V, il vient $X_{ij} = z^i \cdot z^j$. La valeur du critère vaut donc

$$\sum_{ij} X_{ij} = \|z^1 + \cdots + z^n\|^2 \ . \tag{9.39}$$

Considérons maintenant une coloration de $\bar{\mathcal{G}}$ avec k couleurs, c'est-à-dire, une application γ de $\mathcal{V}$ dans $\{1, \ldots, k\}$ telle que

$$(i,j) \in \bar{\mathcal{E}} \implies \gamma(i) \neq \gamma(j) \ .$$

Si les sommets i et j sont de la même couleur (i.e., $\gamma(i) = \gamma(j)$), alors i et j ne peuvent être reliés par une arête de $\bar{\mathcal{E}}$, donc $(i,j) \in \mathcal{E}$, donc $z^i \cdot z^j = X_{ij} = 0$. On a montré :

$$\gamma(i) = \gamma(j) \implies z^i \cdot z^j = 0 \ . \tag{9.40}$$

D'autre part,

$$\sum_{i \in \mathcal{V}} X_{ii} = \sum_{i \in \mathcal{V}} \|z^i\|^2 = 1 \ .$$

Pour toute couleur $c \in \{1, \ldots, k\}$, considérons la somme des vecteurs associés à des sommets de couleur c :

$$w^c = \sum_{i \in \mathcal{V},\ \gamma(i)=c} z^i \ .$$

En utilisant (9.40), on obtient

$$\sum_{1 \leqslant c \leqslant k} \|w^c\|^2 = \sum_{i \in \mathcal{V}} \|z^i\|^2 = 1 \ .$$

Comme $\sum_{1 \leqslant c \leqslant k} w^c = \sum_{i \in \mathcal{V}} z^i$, en utilisant (9.39) et l'inégalité triangulaire, on obtient :

$$\sum_{ij} X_{ij} = \| \sum_{1 \leqslant c \leqslant k} w^c \|^2 \leqslant \left(\sum_{1 \leqslant c \leqslant k} \|w^c\| \right)^2 \ .$$

Soit $t_c := \|w^c\|$. On a $\sum_{1 \leqslant c \leqslant k} t_c^2 = 1$, et $\sum_{ij} X_{ij} \leqslant (\sum_{1 \leqslant c \leqslant k} t_c)^2$. Un simple exercice d'optimisation montre que

$$\sqrt{k} = \max \Big\{ \sum_{1 \leqslant c \leqslant k} t_c \ ; \qquad \sum_{1 \leqslant c \leqslant k} t_c^2 = 1 \Big\}$$

(en termes géométriques, la boule ℓ_2 de rayon 1 est incluse dans la boule ℓ_1 de rayon $\sqrt{k}$). On a donc $\sum_{ij} X_{ij} \leqslant k$. Ceci étant vrai pour tout point admissible X de la formulation (9.36), on a $\vartheta(\mathcal{G}) \leqslant k$. Cette inégalité étant vraie pour toute coloration γ de $\bar{\mathcal{G}}$ en k couleurs, on a $\vartheta(\mathcal{G}) \leqslant \chi(\bar{\mathcal{G}})$. ■

Exercice 9.31 (Coupe maximale) On considère la généralisation du problème de la coupe maximale :

$$\max_{x \in \{\pm 1\}^n} x^\top C x \ , \tag{9.41}$$

avec $C = (C_{ij})$, et sa relaxation SDP

$$\max_{X \in \mathcal{S}_n^+, X_{ii}=1, \forall i} \langle C, X \rangle \ , \tag{9.42}$$

1. Montrer que cette relaxation SDP peut s'écrire :

$$\text{Max} \sum_{ij} C_{ij} v^i \cdot v^j \ ; \qquad \|v^i\|^2 = 1, \forall i, \ v^1, \ldots, v^n \in \mathbb{R}^n \ .$$

Si v désigne un point admissible de ce problème, une manière de fabriquer une coupe est de prendre un vecteur h aléatoire, tiré selon une distribution Gaussienne standard, et de poser :

$$x_i = \operatorname{sgn} v^i \cdot h \ .$$

La coupe associée est formée des arcs reliant des sommets de signe opposé. La valeur de cette coupe est

$$H := \sum_{i,j} C_{ij} (\operatorname{sgn} v^i \cdot h)(\operatorname{sgn} v^j \cdot h) \ .$$

2. Montrer que

$$\Pr(\mathrm{sgn}(v^i \cdot h) \neq \mathrm{sgn}(v^j \cdot h)) = \frac{1}{\pi} \arccos(v^i \cdot v^j) \ .$$

(On admettra que la projection d'un vecteur aléatoire Gaussien standard sur un plan est encore un vecteur aléatoire Gaussien standard).

3. En déduire que l'espérance de H vaut :

$$E(H) = \frac{2}{\pi} \sum_{ij} C_{ij} \arcsin(v^i \cdot v^j) \ .$$

4. On définit $\arcsin(X) := (\arcsin(X_{ij}))$. Montrer que (9.41) équivaut à

$$\max_{X \in \mathcal{S}_n^+, X_{ii}=1, \forall i} \frac{2}{\pi} \langle C, \arcsin X \rangle \ , \tag{9.43}$$

5. Montrer que si A et B sont symétriques positives, alors, le produit coefficient par coefficient de A est B est encore une matrice positive. En notant que

$$\arcsin(X) = X + \frac{1}{2}\frac{X^{\odot 3}}{3} + \frac{1 \times 3}{2 \times 4}\frac{X^{\odot 5}}{5} + \cdots$$

où $X^{\odot k} := (X_{ij}^k)$, déduire de la question précédente que si $C \in \mathcal{S}_n^+$, la valeur de (9.41) est minorée par $2/\pi$ fois la valeur de (9.42). Ce corollaire est dû à Nesterov.

9.4 Optimisation polynomiale et problèmes de moments

Cette section étudie les problèmes d'optimisation à données polynomiales. On montre comment relaxer le problème par des contraintes de positivité des matrices de moments. Cette relaxation s'interprète comme un certain type de décomposition du critère en somme pondérée de carrés de polynômes.

9.4.1 Optimisation polynomiale

Appliquons maintenant l'optimisation SDP à la résolution de problèmes de calcul du minimum global de problèmes de la forme

$$\operatorname*{Min}_{x \in \mathbb{R}^n} f(x); \ \ g_i(x) \geqslant 0, \ \ i = 1, \ldots, r, \tag{P}$$

où les fonctions f et g_i sont des polynômes à n variables, à coefficients réels. On note f^* la valeur du problème, et K l'ensemble admissible :

$$K := \{x \in \mathbb{R}^n; \ \ g_i(x) \geqslant 0, \ \ i = 1, \ldots, r\}. \tag{9.44}$$

Soit $\mathcal{P}(K)$ l'ensemble des mesures de probabilité à support dans K, ayant des moments de tous ordres. Introduisons le problème relaxé

$$\operatorname*{Min}_{\mu \in \mathcal{P}(K)} \int_K f(x) \mathrm{d}\mu(x), \tag{PR}$$

qui s'interprète comme la minimisation (par rapport à la loi de probabilité) de l'espérance de $f(x)$.

Proposition 9.32 *Les problèmes (P) et (PR) ont même valeur, et $\mu \in S(PR)$ ssi $\mathrm{supp}(\mu) \subset S(P)$.*

Démonstration. Associant à tout $x \in K$ la mesure de Dirac δ_x concentrée sur x, on obtient $\mathrm{val}(PR) \leqslant \inf_{x\in K} f(x) = f^*$. Réciproquement, si $\mu \in \mathcal{P}(K)$, on a

$$f^* \leqslant \inf\{f(x);\ x \in \mathrm{supp}(\mu)\} \leqslant \int_K f(x)\mathrm{d}\mu(x), \tag{9.45}$$

d'où $f^* \leqslant \mathrm{val}(PR)$, et finalement $f^* = \mathrm{val}(PR)$. De plus les égalités de (9.45) sont des égalités ssi $f(x)$ est constant sur $\mathrm{supp}(\mu)$, et vaut f^* d'où la conclusion. ■

Le problème (PR) est convexe, mais de dimension infinie et donc à première vue inexploitable. Nous allons voir comment lui associer des relaxations de dimension finie.

9.4.2 Relaxation grâce aux matrices de moments

Au multi-indice $\alpha \in \mathbb{N}^n$, de *taille* $|\alpha| := \sum_{i=1}^n \alpha_i$, est associé le monôme $x^\alpha = \Pi_{i=1}^n x_i^{\alpha_i}$. On appellera α l'*exposant* du monôme, et $|\alpha|$ son *degré*. Le degré d'un polynôme est le plus grand degré des monômes le composant. Notons

$$S_N := \{\alpha \in \mathbb{N}^n;\ |\alpha| \leqslant N\}$$

l'ensemble des multi-indices de taille au plus N. Un polynôme q de degré au plus N est de la forme

$$q(x) = \sum_{\alpha\in S_N} q_\alpha x^\alpha,$$

où q_α est le *coefficient* du monôme x^α. Notons $y_\alpha(\mu) := \int_{\mathbb{R}^n} x^\alpha \mathrm{d}\mu(x)$ le *moment* d'*ordre* α associé à la mesure de probabilité $\mu \in \mathcal{P}(\mathbb{R}^n)$. On notera $y_N(\mu)$ le vecteur[2] des moments d'ordre au plus N. Soit $MM_N(K)$ l'ensemble des moments d'ordre au plus N de mesures de $\mathcal{P}(K)$:

$$MM_N(K) := \{y_N(\mu);\ \ \mu \in \mathcal{P}(K)\}.$$

Notons p_α les coefficients des monômes de f. Le problème (PR) peut se réécrire sous la forme de problèmes de moments optimaux :

$$\underset{y\in MM_N(K)}{\mathrm{Min}} \sum_{\alpha\in S_N} p_\alpha y_\alpha, \tag{9.46}$$

où $N \geqslant \deg(f)$ (a priori $N = \deg(f)$ suffit, mais l'analyse qui suit fait intervenir les moments d'ordre plus élevés). L'ensemble $MM_N(K)$ est convexe, car c'est l'image de l'ensemble convexe $\mathcal{P}(K)$ par une application linéaire. Le problème (9.46) est donc convexe et de dimension finie. La difficulté est que ses contraintes ne sont pas explicites.

Montrons qu'on peut relaxer ces contraintes en introduisant la *matrice de moment* d'ordre $N \in \mathbb{N}$, et notée $M^N = M^N(y)$, où $y \in MM_{2N}(K)$, définie de la manière suivante : ses indices (α, β) parcourent S_N et

$$M^N_{\alpha,\beta}(y) := y_{\alpha+\beta}, \quad \text{pour tout } \ \alpha, \beta \in S_N. \tag{9.47}$$

Ceci définit une application linéaire de $MM_{2N}(K)$ vers l'espace des matrices symétriques d'indices dans S_N.

Lemme 9.33 *Pour tout $N \in \mathbb{N}$, et $y \in MM_{2N}(K)$, la matrice $M^N(y)$ est semi-définie positive.*

Démonstration. Pour tout $v = \{v_\alpha;\ \alpha \in S_N\}$, identifié au polynôme $v(x) = \sum_{|\alpha|\leqslant N} v_\alpha x^\alpha$, on a

$$v^\top M(y) v = \sum_{(\alpha,\beta)\in S_N\times S_N} v_\alpha v_\beta y_{\alpha+\beta} = \sum_{\substack{(\alpha,\beta)\in S_N\times S_N \\ \alpha+\beta=\gamma}} v_\alpha v_\beta y_\gamma = \int_{\mathbb{R}^n} v(x)^2 \mathrm{d}\mu(x) \geqslant 0. \tag{9.48}$$

2. Il n'est pas nécessaire à ce stade de préciser l'ordre choisi sur les monômes. Aux vecteurs de la forme $(y_\beta)_{\beta\in J}$ sont associés les matrices $(M_{\alpha\beta})_{\alpha\in I,\beta\in J}$. Le produit matrice vecteur est défini par $(My)_\alpha = \sum_{\beta\in J} M_{\alpha\beta} y_\beta$.

La dernière égalité est basée sur le fait que si $w(x) = v(x)^2$, alors $w_\gamma = \sum\{v_\alpha v_\beta;\ \alpha+\beta=\gamma\}$.

■

Notons Y_N l'espace vectoriel des vecteurs indicés par les indices de S_N. Le lemme suggère immédiatement la relaxation suivante du problème de moments (9.46) : remplacer la contrainte $y \in MM_N(K)$ par $y \in Y_{2N}$, tel que $M^N(y) \succcurlyeq 0$, et, notant que le moment d'ordre zéro vaut 1, résoudre le problème

$$\operatorname*{Min}_{y\in Y_{2N}} \sum_{|\alpha|\leqslant \deg(f)} p_\alpha y_\alpha; \quad y_0 = 1 : \quad M^N(y) \succcurlyeq 0. \tag{9.49}$$

Pour que le critère ait un sens, il faut que $2N \geqslant \deg(f)$. Cependant la relaxation (9.49) a un grave défaut : elle ne tient pas compte des contraintes $g_i(x) \geqslant 0$, $i = 1$ à r, et n'est donc qu'une relaxation du problème de minimisation du critère sur l'espace entier.

Étendons donc le lemme 9.33 de manière à prendre en compte les contraintes. Soient q un polynôme de degré m et μ une mesure de moments finis jusqu'à l'ordre $2N+m$. Notons $q\mu$ la mesure qui à une fonction polynomiale g associe $\int_{\mathbb{R}^n} g(x)q(x)\mathrm{d}\mu(x)$. Si y est le vecteurs des moments de μ, définissons $qy \in Y_{2N}$ comme le vecteur des moments de $q\mu$. Cela a un sens, car ce dernier ne dépend que de q et y, et a pour expression alors

$$(qy)_\eta := \sum_{|\gamma|\leqslant m} q_\gamma y_{\eta+\gamma}; \quad |\alpha| \leqslant 2N. \tag{9.50}$$

Cette relation s'étend à tout $y \in Y_{2N+m}$. La *matrice de moments pondérée* dite aussi *matrice de localisation* de degré $N \in \mathbb{N}$ avec poids q, est définie comme $M^N(qy)$. C'est donc la matrice de moments de la mesure $q(x)\mathrm{d}\mu(x)$ si y est le vecteur de moments associé.

Lemme 9.34 *Soient q un polynôme de degré m, et $N \in \mathbb{N}$. Si la mesure de probabilité μ est à support dans l'ensemble $\{x \in \mathbb{R}^n;\ q(x) \geqslant 0\}$, et si $y \in Y_{2N+m}$ est le vecteur de moments associé, alors la matrice $M^N(qy)$ est semi-définie positive.*

Démonstration. Soit $v = \{v_\alpha;\ \alpha \in S_N\}$, vecteur des coefficients du polynôme $v(x)$. Alors

$$v^\top M^N(qy)v = \sum_{\alpha,\beta} v_\alpha v_\beta (qy)_{\alpha+\beta} = \sum_{\alpha,\beta,\gamma} v_\alpha v_\beta q_\gamma y_{\alpha+\beta+\gamma} = \int_{\mathbb{R}^n} v(x)^2 q(x)\mathrm{d}\mu(x) \tag{9.51}$$

est positif (avec dans les sommes ci-dessus α, β dans S_N et γ dans S_m), d'où la conclusion. ■

On pose $g_0 = 1$ de sorte que $g_0 y = y$. Alors $M^N(g_0 y) = M^N(y)$. La discussion précédente conduit à une famille de relaxations du problème de moments, indicée par $N := (N_0, \ldots, N_r)$:

$$\operatorname*{Min}_{y\in S_{\bar N}} \sum_{|\alpha|\leqslant \deg(f)} p_\alpha y_\alpha; \quad y_0 = 1; \quad M^{N_i}(g_i y) \succcurlyeq 0, \ \ i = 0, \ldots, r, \tag{Q_N}$$

où

$$\bar N := \max_i \{2N_i + \deg(g_i)\}.$$

On dira que $N \to +\infty$ si $\min_i N_i \to +\infty$. Pour que le critère ait un sens on est amené à supposer que $\bar N \geqslant \deg(f)$.

Lemme 9.35 *Munissons $\mathbb{N}^n$ de l'ordre naturel : $N' \geqslant N$ si $N' - N \geqslant 0$. Si $\bar N \geqslant \deg(f)$, alors $N \to \mathrm{val}(Q_N)$ est croissante, et $\lim_{N\to+\infty} \mathrm{val}(Q_N) \leqslant \mathrm{val}(P)$.*

Démonstration. Soit $N \in \mathbb{N}^{r+1}$ tel que $\bar N \geqslant \deg(f)$. Soient $m \in \mathbb{N}$, $m > 1$, et $q \in \mathbb{P}$. On passe de $M^m(qy)$ à $M^{m-1}(qy)$ en supprimant la dernière ligne et colonne de $M^m(qy)$. Cette opération conserve la semi définie positivité de la matrice. Tout y réalisable pour $(Q_{N'})$, $N' \geqslant N$, a donc une restriction $\{y_\alpha,\ |\alpha| \leqslant \bar N\}$ réalisable pour (Q_N). Ceci prouve que $N \to \mathrm{val}(Q_N)$ est croissante. D'après le lemme 9.34, si $x \in F(P)$, les moments associés $\{y_\alpha = x^\alpha,\ |\alpha| \leqslant \bar N\}$ sont dans $F(Q_N)$. La conclusion s'ensuit. ■

9.4.3 Décomposition en somme pondérée de carrés

Dualisation de la relaxation

Nous allons expliciter le problème dual de (Q_N), et montrer qu'il s'interprète comme un problème de décomposition de f comme somme pondérée de carrés de polynômes, et d'une constante la plus élevée possible. Le problème (Q_N) est de type SDP linéaire (minimisation d'un critère linéaire sous contrainte de positivité de matrices, fonctions affines des paramètres d'optimisation). C'est donc un problème convexe. Son lagrangien a pour expression

$$L(y,\lambda,\mathbf{X}) := \sum_{|\alpha|\leqslant \deg(f)} p_\alpha y_\alpha + \lambda(1-y_0) - \sum_{i=0}^{r} \langle X^i, M^{N_i}(g_i y)\rangle, \tag{9.52}$$

avec $\lambda \in \mathbb{R}$ et $\mathbf{X} = (X^1,\ldots,X^r)$, $X^i \succcurlyeq 0$, $i = 0$ à r, matrices symétriques indicée par les multiindices de taille au plus $|S_{N_i}|$.

Pour calculer le problème dual il est nécessaire d'expliciter la transposée de $y \to M^{N_i}(g_i y)$, qui découle des expressions ci-dessous. Soit $X = (X_{\alpha,\beta};\ \alpha,\beta \in S_N)$ symétrique. Alors

$$\left\{ \begin{array}{lll} \langle X, M^m(y)\rangle & = & \displaystyle\sum_{|\delta|\leqslant 2m} y_\delta \left(\sum_{\alpha+\beta=\delta} X_{\alpha,\beta} \right); \\ \langle X, M^m(qy)\rangle & = & \displaystyle\sum_{|\delta|\leqslant 2m+\deg(q)} y_\delta \left(\sum_{\alpha+\beta+\gamma=\delta} q_\gamma X_{\alpha,\beta} \right). \end{array} \right. \tag{9.53}$$

Le problème dual, obtenu en minimisant le lagrangien par rapport à la variable primale y, a donc pour expression

$$\left\{ \begin{array}{lll} \underset{\lambda,\mathbf{X}}{\mathrm{Max}}\, \lambda; & p_0 & = \lambda + \sum_{i=0}^r (g_i)_0 X^i_{00};\ \ X^i \succcurlyeq 0,\ i = 0,\ldots,r, \\ & p_\delta & = \sum_{i=0}^r \sum_{\alpha+\beta+\gamma=\delta} (g_i)_\gamma X^i_{\alpha,\beta}, \quad 0 < |\delta| \leqslant 2\bar{N}. \end{array} \right. \tag{DQ_N}$$

Si $y \in F(Q_N)$ et $(\lambda, X) \in F(DQ_N)$, d'après l'expression de (DQ_N), la différence des critères primal et dual est, comme on s'y attend, égale à la somme des écarts complémentaires :

$$\Delta := \sum_{\delta\in S_{\bar{N}}} p_\delta y_\delta - \lambda = \sum_{\delta\in S_{\bar{N}}} \sum_{i=0}^{r} y_\delta \sum_{\alpha+\beta+\gamma=\delta} (g_i)_\gamma X^i_{\alpha,\beta} = \sum_{i=0}^{r} \langle X^i, M^{N_i}(g_i y)\rangle. \tag{9.54}$$

Lemme 9.36 (i) *Si l'hypothèse suivante de qualification est satisfaite :*

$$\textit{Il existe } y \in F(Q_N) \textit{ tel que } M^{N_i}(g_i y) \succ 0,\ i = 1 \textit{ à } r, \tag{9.55}$$

alors $\mathrm{val}(Q_N) = \mathrm{val}(DQ_N)$*, et si cette quantité est finie,* $S(DQ_N)$ *est non vide et borné.*

Démonstration. C'est une conséquence directe du corollaire 2.80. ■

Remarque 9.37 On peut en particulier chercher si un vecteur de moments d'une mesure de probabilité μ à support sur K satisfait (9.55). En raison de (9.51), la condition de qualification est satisfaite ssi

$$\int_{\mathbb{R}^n} v(x)^2 g_i(x) \mathrm{d}\mu(x) > 0, \text{ pour tout polynôme } v \text{ non nul de degré au plus } N. \tag{9.56}$$

L'hypothèse de qualification est donc satisfaite si K est d'intérieur non vide. En effet, il contient alors une boule de la forme $B(x_0, r)$ avec $r > 0$, et il suffit de prendre μ uniforme sur $B(x_0, r)$.

Sommes pondérées de carrés de polynômes

Nous dirons que le polynôme f est décomposé comme sommes de carrés de polynômes pondérées par les g_i, si la relation suivante est satisfaite :

$$f(x) = \mu + \sum_{i=0}^{r} g_i(x) \sum_{j \in J_i} q^{ij}(x)^2, \quad \text{pour tout } x \in \mathbb{R}^n. \tag{9.57}$$

La décomposition a pour paramètres une constante μ, les ensembles finis J_i, et les polynômes q^{ij}, de degré au plus d_i, $i = 0$ à r. On a alors $f(x) \geqslant \mu$ sur K, et donc $\mu \leqslant f^*$. On peut alors se poser le problème de trouver la *meilleure constante* μ, sous contrainte de degré maximum des q^{ij}, ce qui conduit au problème de maximisation paramétré par $D := (d_0, \ldots, d_r)$:

$$\operatorname*{Max}_{\mu,q} \mu \quad \text{tel que (9.57) est satisfait ; } \deg(q^{ij}) \leqslant d_i, \quad i = 0, \ldots, r. \tag{DEC_d}$$

Proposition 9.38 (i) *L'ensemble réalisable des points (λ, X) de $F(DQ_N)$ est l'image des points admissibles (μ, q) de $F(DEC_d)$, avec $d_i = |S_{N_i}|$, $i = 0$ à r, par la relation*

$$\text{(a)} \quad \lambda = \mu; \qquad \text{(b)} \quad X^i = \sum_{j \in J_i} q^{ij} (q^{ij})^\top, \quad i = 0, \ldots, r. \tag{9.58}$$

(ii) *Quand $d_i = |S_{N_i}|$, on a $\operatorname{val}(DQ_N) \leqslant \operatorname{val}(DEC_d)$, avec égalité si K est d'intérieur non vide.*
(iii) *Supposons K d'intérieur non vide, ainsi que l'existence de polynômes $\bar{q}^{ij}$, avec $j \in J_i$ (ensembles finis), de degré d_i, tels que*

$$f(x) = f^* + \sum_{i=0}^{r} g_i(x) \sum_{j \in J_i} \bar{q}^{ij}(x)^2, \tag{9.59}$$

pour tout $x \in \mathbb{R}^n$. Alors $\operatorname{val}(P) = \operatorname{val}(Q_N) = \operatorname{val}(DQ_N)$, quand $d_i \leqslant |S_{N_i}|$, pour tout $i = 0$ à r.

Démonstration. (i) A tout $(\mu, q) \in F(DEC_d)$ on peut associer (λ, X) par (9.58), et $(\lambda, q) \in F(DEC_d)$ implique $(\lambda, X) \in F(DQ_N)$. En effet, $X^i \succcurlyeq 0$ pour tout i, et on obtient les contraintes d'égalité de (DQ_N) en explicitant l'égalité des coefficients de chaque monôme dans (9.57). Comme $\lambda = \mu$ on en déduit que $\operatorname{val}(DQ_N) \leqslant \operatorname{val}(DEC_d)$.
Réciproquement, soit $(\lambda, X) \in F(DQ_N)$. Les X^i étant semi définis positifs, ils sont de la forme donnée par (9.58)(b) avec $\deg(q_{ij}) \leqslant |S_{N_i}|$. Avec (9.54), et utilisant $\langle xx^\top, A \rangle = x^\top A x$, pour $y \in F(Q_N)$, il vient

$$\Delta = \sum_{i=0}^{r} \sum_{j \in J_i} (q^{ij})^\top M^{N_i}(g_i y) q^{ij}. \tag{9.60}$$

Prenant pour y les moments associés à la mesure de Dirac δ_x, où $x \in K$, et utilisant (9.51), nous vérifions que l'égalité dans (9.57) est satisfaite pour tout $x \in K$, et donc si K est d'intérieur non vide, pour tout $x \in \mathbb{R}^n$. ceci signifie que $(\lambda, q) \in F(DEC_d)$, et donc $\operatorname{val}(DEC_d) \leqslant \operatorname{val}(DQ_N)$. ces deux valeurs sont donc égales si K est d'intérieur non vide.
(ii) et (iii) Conséquence directe du point (i). ■

L'hypothèse de la proposition 9.38(iii) n'est malheureusement pas toujours satisfaite quand $n > 1$. Nous allons montrer dans la prochaine section que $\operatorname{val}(Q_N) \to \operatorname{val}(P)$ dans certaines situations auxquelles on peut se ramener.

Existence de décompositions en sommes de carrés

La théorie est basée sur un résultat (admis car de démonstration difficile) dû à Putinar [Put93] ; Abrégeons *somme de carrés* en SDC. Définissons l'ensemble suivant de polynômes :

$$S(K) := \left\{ \rho_0(x) + \sum_{i=1}^{r} \rho_i(x) g_i(x); \quad \rho_i(x) \ \text{ SDC }, \ i = 0, \ldots, r \right\}. \tag{9.61}$$

Théorème 9.39 *Supposons K compact, ainsi que l'existence de $\hat{\rho} \in S(K)$ tel que $\hat{\rho}^{-1}(\mathbb{R}_+)$ est compact. Alors tout polynôme strictement positif sur K appartient à $S(K)$.*

Montrons comment exploiter ce résultat dans le cas de la minimisation sur $\mathbb{R}^n$ d'un polynôme $f(x)$ (l'extension au cas avec contraintes ne présente pas de difficultés). Supposons connu un nombre $a > 0$ tel que p atteint son minimum en un point x^* avec $\|x^*\| \leqslant a$ (norme euclidienne). Posons

$$\theta(x) := a^2 - \sum_{j=1}^{n} x_j^2; \quad K_a := \{x \in \mathbb{R}^n;\ \theta(x) \geqslant 0\}, \tag{9.62}$$

et étudions le problème avec contrainte $\mathrm{Min}_x\{f(x);\ x \in K_a\}$. Une famille de problèmes relaxés associés est, pour tout $N = (N_0, N_1)$ tel que $N_0 \geqslant \deg(f)$:

$$\min_y \sum_{\alpha \in S_N} p_\alpha y_\alpha; \quad y_0 = 1; \quad M^{N_0}(y) \succcurlyeq 0;\ \ M^{N_1}(\theta y) \succcurlyeq 0. \tag{Q_N^a}$$

Le problème dual (DQ_N) a ici pour expression

$$\begin{aligned} \max_{\lambda, X, Z} \lambda; \quad p_0 &= \lambda + X_{00} + a^2 Z_{00};\ \ X, Z \succcurlyeq 0; \\ p_\alpha &= \sum_{\beta+\gamma=\alpha} \left(X_{\beta,\gamma} + a^2 Z_{\beta,\gamma}\right) - \sum_{i=1}^{n} \sum_{\beta+\gamma+2e_i=\alpha} Z_{\beta,\gamma}, \\ &\quad \alpha \neq 0,\ |\alpha| \leqslant \bar{N}. \end{aligned} \tag{DQ_N^a}$$

Théorème 9.40 *Soit f de degré $2m$, atteignant son minimum sur $\mathbb{R}^n$ en un point x^* tel que $\|x^*\| \leqslant a$. Alors* (i) *Pour N assez grand, (Q_N^a) est qualifié, et $\mathrm{val}(Q_N^a) \uparrow f^*$ quand $N \to \infty$.*
(ii) *On a $\min(Q_N^a) = f^*$ ssi il existe des polynômes q_i de degré $d_i \leqslant |S_{N_i}|$, $i = 1$ à r, et des polynômes t_j de degré au plus $d_0 \leqslant |S_{N_0}|$ tels que*

$$f(x) - f^* = \sum_i q_i(x)^2 + \theta(x) \sum_j t_j(x)^2. \tag{9.63}$$

Démonstration. (i) Pour tout $\varepsilon > 0$, le polynôme $f_\varepsilon(x) := f(x) - f^* + \varepsilon$ est strictement positif sur K. Appliquant le théorème 9.39, avec $\hat{\rho}(x) = \theta(x)$ (pour lequel $\rho_0(x) = 0$ et $\rho_1(x) = 1$), on vérifie que $f_\varepsilon(x) \in S(K)$. Lui associant les X^i définies par (9.58), on obtient pour N assez grand un élément admissible de $F(DQ_N)$, avec $\lambda \geqslant f^* - \varepsilon$, et donc $\mathrm{val}(DQ_N) \geqslant f^* - \varepsilon$.
(ii) On applique la proposition 9.38. ■

Variantes creuses

Notons $J_i \subset \{1, \ldots, n\}$ l'ensemble des variables apparaissant dans l'expression de la contrainte $g_i(x) \geqslant 0$, $i = 1$ à r. Pour réduire l'encombrement mémoire et la taille des calculs, on peut penser à remplacer la relaxation $M^{N_i}(g_i y) \succcurlyeq 0$ par $M_i^{N_i}(g_i y) \succcurlyeq 0$, où la matrice $M_i^{N_i}(g_i y)$ est la restriction de $M^{N_i}(g_i y)$ aux lignes et colonnes n'incluant que des indices dans I_i.

Plus généralement, donnons-nous une collection I_k de parties de $\{1, \ldots, n\}$ telles que (i) pour tout $i = 1$ à r, il existe k tel que $J_i \subset I_k$, (ii) on peut décomposer le critère en $f = f_1 + \cdots + f_p$, où les f_i sont des polynômes, faisant intervenir les variables d'indice $J_i' \subset \{1, \ldots, n\}$, et pour chaque $i = 1$ à p, il existe k tel que $J_i' \subset I_k$, et (iii) la relation suivante est satisfaite :

$$I_{k+1}\,(\cap \cup_{j \leqslant k} I_j) \subset I_s, \quad \text{pour un certain } \ s \leqslant k. \tag{9.64}$$

On peut alors étendre les résultats de convergence de la section précédente. Voir Lasserre [Las06].

Contre exemple à la décomposition en somme de carrés

On ne peut pas décomposer tous les polynômes positifs de $\mathbb{R}^n$ en somme de carrés.
Le premier contre exemple est dû à Motzkin, c'est le polynôme

$$f(x,y,z) = x^4y^2 + x^2y^4 + z^6 - x^2y^2z^2. \tag{9.65}$$

Détaillons une variante à deux variables, due à Berg :

$$p(x,y) = x^4y^2 + x^2y^4 + 1 - x^2y^2 = 1 + x^2y^2(x^2 + y^2 - 1). \tag{9.66}$$

Lemme 9.41 *Le polynôme donné par* (9.66) *est positif, mais ce n'est pas une somme de carrés.*

Démonstration. Si $x^2 + y^2 \geqslant 1$ alors $p(x,y) \geqslant 1$. Si $x^2 + y^2 < 1$, de $(x \pm y)^2 \geqslant 0$ on déduit que $|xy| \leqslant x^2 + y^2 < 1$, donc $p(x,y) > 1 + (x^2 + y^2 - 1) \geqslant 0$. Le polynôme est donc positif.
Supposons l'existence de polynômes q_i, $i = 1$ à n, tels que

$$p(x,y) = \sum_{i=1}^{n} q_i(x,y)^2. \tag{9.67}$$

De $p(x,0) = p(0,y) = 1$ on déduit que $q_i(x,0)$ et $q_i(0,y)$ doivent être constants. On peut donc écrire $q_i(x,y) = a_i + xyh_i(x,y)$, et de plus les polynômes h_i doivent être de degré au plus 1. Avec (9.67) il vient

$$p(x,y) = \sum_{i=1}^{n} a_i^2 + 2xy \sum_{i=1}^{n} a_i h_i(x,y) + x^2y^2 \sum_{i=1}^{n} h_i(x,y)^2. \tag{9.68}$$

De là, comparant les termes de chaque degré

$$\sum_{i=1}^{n} a_i^2 = 1; \quad xy \sum_{i=1}^{n} a_i h_i(x,y) = 0; \quad x^2y^2 \sum_{i=1}^{n} h_i(x,y)^2 = x^2y^2(x^2 + y^2 - 1). \tag{9.69}$$

En conséquence $x^2+y^2-1 = \sum_{i=1}^{n} h_i(x,y)^2$, ce qui est impossible (une somme de carrés a un terme constant positif). ■

9.5 Notes

L'ouvrage introductif [BTN01] présente de nombreuses applications de la programmation semi-définie, allant des sciences de l'ingénieur à la combinatoire. On pourra aussi se reporter à [WSV00, Hel00, BTN01, Lov, Bar02, GM12] pour approfondir.

Plusieurs solveurs SDP sont librement disponibles, comme la librairie csdp [3]. Si l'on dispose de Matlab (payant), on peut utiliser d'autres solveurs comme SeDuMi [4] associés éventuellement à la boîte à outils YALMIP [5] (qui fournit une interface de haut niveau effectuant "l'assemblage" des matrices préalable à l'appel de solveurs).

Dans la section 9.4 nous avons suivi l'approche de Lasserre [Las02]. Une démonstration basée sur des outils élémentaires, du résultat de Putinar [Put93], a été donnée par M. Schweighofer [Sch05]. L'extension aux versions creuses est introduite dans Waki et al. [WKKM09]. Sa convergence est établie par Lasserre [Las06].

3. https ://projects.coin-or.org/Csdp/
4. http ://sedumi.mcmaster.ca/
5. http ://control.ee.ethz.ch/ joloef/wiki/pmwiki.php

Chapitre 10

Algorithmes de points intérieurs

Ce chapitre donne une brève introduction aux principes de base des algorithmes de points intérieurs pour la programmation linéaire et quadratique convexe. Ces algorithmes forment une suite de points vérifiant strictement les contraintes d'inégalité (d'où leur nom). La classe d'algorithmes prédicteur-correcteur qui est étudiée a la propriété d'avoir la meilleure estimation connue (y compris dans la classe des programmes linéaires) du nombre d'itérations (nécessaire pour atteindre une précision donnée).

10.1 Pénalisation logarithmique et trajectoire centrale

Considérons le problème

$$\operatorname*{Min}_x f(x)\ ;\ Ax = b,\ x \geqslant 0, \tag{P}$$

avec $f : \mathbb{R}^n \to \mathbb{R}$ convexe et de classe C^1, et A matrice $p \times n$ de rang p. L'exemple principal est celui du programme quadratique convexe sous forme standard, pour lequel, H étant une matrice symétrique $n \times n$, semi définie positive :

$$f(x) = c \cdot x + \tfrac{1}{2} x^\top H x; \quad c \in \mathbb{R}^n. \tag{10.1}$$

Quand $H = 0$, on retrouve le programme linéaire sous forme standard. Rappelons que nous notons par $F(P) := \{x \in \mathbb{R}^n_+ ; Ax = b\}$ l'ensemble des points réalisables du problème (P).

On dit que $x \in \mathbb{R}^n$ est *strictement positif* si $x_i > 0$, $i = 1$ à n, et on écrit $x > 0$. Notons l'ensemble des *vecteurs strictement positifs* de $\mathbb{R}^n$ et des *points intérieurs réalisables* du problème (P) par, respectivement,

$$\mathbb{R}^n_{++} := \{x \in \mathbb{R}^n_+ ; x > 0\}; \quad F_{++}(P) = F(P) \cap \mathbb{R}^n_{++} = \{x \in \mathbb{R}^n_{++};\ Ax = b\}. \tag{10.2}$$

Le *potentiel logarithmique* est la fonction strictement convexe, de domaine $\mathbb{R}^n_{++}$, définie par

$$\pi(x) := -\sum_{i=1}^n \log x_i.$$

Le problème avec pénalisation logarithmique associé à (P), de paramètre $\mu > 0$, est

$$\operatorname*{Min}_x f_\mu(x) := f(x) + \mu\pi(x)\ ;\ Ax = b,\ x > 0. \tag{P_μ}$$

Lemme 10.1 *Si $F_{++}(P)$ est borné et non vide, (P_μ) possède une solution unique x^μ.*

Démonstration. a) Existence. Puisque $F_{++}(P)$ est borné et non vide, une suite minimisante x^n pour (P_μ) existe et est bornée. Extrayant une sous-suite si nécessaire, on peut supposer que x^n a une limite notée x^μ. Comme $F(P)$ est fermé et contient $F_{++}(P)$, on a $x^\mu \in F(P)$. Mais puisque x^n est minimisante, on a pour n

assez grand $f(x^n) + \mu\pi(x^n) \leqslant f(x^0) + \mu\pi(x^0) + 1$. Combinant avec $f(x^n) \to f(x^\mu)$, on déduit que $\pi(x^n)$ est bornée supérieurement. Si une composante de x^μ est nulle, on vérifie facilement que $\pi(x^n) \to +\infty$, ce qui est impossible, donc $x^\mu \in F_{++}(P)$. Par continuité de f_μ sur $F_{++}(P)$, il vient $f_\mu(x^\mu) = \lim_n f_\mu(x^n) = \mathrm{val}(P_\mu)$. Donc $x^\mu \in S(P_\mu)$.
b) Unicité. Elle résulte de la stricte convexité de f_μ. ■

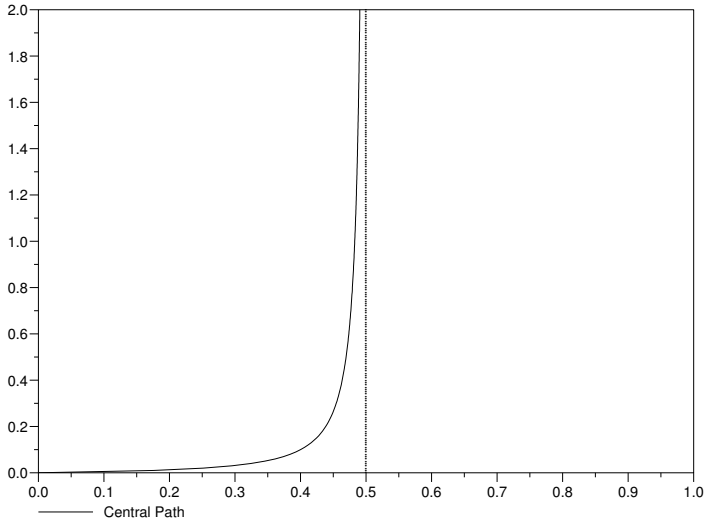


FIGURE 10.1 – Une trajectoire centrale avec virage serré

Puisque (P_μ) est un problème de minimisation d'une fonction convexe et C^1 sous contraintes linéaires, x est solution de (P_μ) ssi il existe $\lambda \in \mathbb{R}^p$ tel que (x, λ) est solution du système d'optimalité du premier ordre

$$\begin{cases} \nabla f(x) - \mu x^{-1} + A^\top \lambda = 0 \\ Ax = b,\ x > 0 \end{cases} \tag{10.3}$$

avec $x^{-1} := (x_1^{-1}, \cdots, x_n^{-1})^\top$. Posons $s := \mu x^{-1}$, et notons $\mathbf{1}$ le vecteur de $\mathbb{R}^n$ de composantes égales à 1. On utilise dans ce chapitre le produit de vecteurs composante par composante, par exemple $(xs)_i = x_i s_i$, pour $i = 1$ à n. Alors $xs = \mu\mathbf{1}$, et le système d'optimalité équivaut à

$$\begin{cases} xs = \mu\mathbf{1} \\ \nabla f(x) + A^\top \lambda = s \\ Ax = b \\ x \geqslant 0,\ s \geqslant 0. \end{cases} \tag{10.4}$$

Quand $\mu = 0$, les relations (10.4) se réduisent au système d'optimalité de (P). Comme $\pi(x)$ est strictement convexe, (P_μ) a au plus une solution x^μ. L'ensemble des solutions de (10.4) est de la forme $(x^\mu, s^\mu = \mu(x^\mu)^{-1}, \lambda^\mu)$; l'unicité de x^μ implique celle de s^μ. Comme A est de rang p, donc surjective, $A^\top$ est injective ce qui implique l'unicité de λ^μ.

On appelle trajectoire centrale primale (resp. primale-duale, duale), l'ensemble des x^μ (resp. (x^μ, s^μ), s^μ) solution optimale (resp. primale-duale, duale) de (10.4).

On représente en figure 10.1 la trajectoire centrale primale du problème de minimisation de $x_2 + \varepsilon x_1$ sous les contraintes $x_1 \in [0, 1]$ et $x_2 \geqslant 0$, pour $\varepsilon = 0$ (demi-droite $x_1 = 1/2$ et $x_2 \geqslant 0$) et pour $\varepsilon = 0.04$. Quand

$\varepsilon > 0$, la trajectoire centrale primale converge vers le point $(0,0)$. En conséquence, quand $\varepsilon > 0$ est petit, il y a un virage serré près du point $(\frac{1}{2}, 0)$. Nous verrons que, malgré l'existence de ces virages serrés, il existe des algorithmes efficaces de suivi de la trajectoire centrale.

Le lemme ci-dessous permet de préciser dans quelle mesure une solution de (10.4) donne une solution approchée du problème (P).

Lemme 10.2 *Si (x,s) vérifie (10.4), alors $f(x) \leqslant \mathrm{val}(P) + n\mu$.*

Démonstration. L'application $x' \mapsto L(x', \lambda, s) := f(x) + \lambda^\top (Ax - b) - s \cdot x$ est convexe, de gradient nul au point x puisque $\nabla f(x) + A^\top \lambda = s$. Donc si $x' \in F(P)$, utilisant $Ax = Ax' = b$, il vient

$$f(x) - s \cdot x = L(x, \lambda, s) \leqslant L(x', \lambda, s) = f(x') - s \cdot x' \leqslant f(x').$$

Minimisant sur $x' \in F(P)$, il vient $f(x) \leqslant \mathrm{val}(P) + s \cdot x$. Or $s \cdot x = n\mu$, d'où la conclusion. ■

Remarque 10.3 Dans le cas quadratique convexe : $f(x) = c \cdot x + \frac{1}{2} x \cdot Hx$, (10.4) s'écrit

$$\left\{ \begin{array}{ll} xs & = \mu \mathbf{1} \\ c + Hx + A^\top \lambda & = c \\ Ax & = b \\ x & \geqslant 0,\ s \geqslant 0. \end{array} \right. \tag{10.5}$$

•

Définissons la mesure de proximité de la trajectoire centrale :

$$\delta(x, s, \mu) := \left\| \frac{xs}{\mu} - \mathbf{1} \right\|.$$

où $\| \ \|$ est la norme euclidienne. La trajectoire centrale est donc l'ensemble des points (x,s) réalisables pour (P), de proximité nulle pour un certain μ. On appelle *petit voisinage* et on note $\mathcal{V}_\alpha$, où $\alpha \in]0,1[$, l'ensemble des points (x, s, λ, μ) tels que $(x, s, \lambda) \in F_{++}(P)$ est réalisable, $\mu > 0$ et $\delta(x, s, \mu) \leqslant \alpha$.

10.2 La méthode prédicteur-correcteur

Dans la suite nous allons nous intéresser à une classe d'algorithmes de points intérieurs dans le cas d'un critère quadratique convexe. Le système d'optimalité est la réunion des relations linéaires

$$\left\{ \begin{array}{l} xs = \mu \mathbf{1}, \\ c + Hx + A^\top \lambda = s, \quad Ax = b, \end{array} \right. \tag{10.6}$$

avec les inégalités $x \geqslant 0$ et $s \geqslant 0$. Le point (x, s, λ) est dit (primal-dual) réalisable si $x \geqslant 0$ et $s \geqslant 0$, $Ax = b$ et $c + Hx + A^\top \lambda = s$.

Soit (x, s, λ) réalisable. Appliquons la méthode de Newton au système (10.6) (d'inconnues (x, s, λ) et paramétré par μ). Nous obtenons la direction de *centrage* noté (u^c, v^c, η^c) solution de

$$\left\{ \begin{array}{lcl} su^c + xv^c & = & \mu \mathbf{1} - xs, \\ Hu^c + A^\top \eta^c & = & v^c, \quad Au^c = 0. \end{array} \right. \tag{10.7}$$

Notons que $u^c \cdot v^c = u^c \cdot Hu^c \geqslant 0$. Le point obtenu après centrage est

$$x^c := x + u^c; \quad s^c := s + v^c; \quad \lambda^c := \lambda + \eta^c; \quad \mu^c := \mu. \tag{10.8}$$

La *direction affine* (u^a, v^a, η^a) est défini de manière similaire, mais en "visant" une valeur nulle de μ. Elle est donc solution de

$$\left\{ \begin{array}{lcl} su^a + xv^a & = & -xs, \\ Hu^a + A^\top \eta^a & = & v^a, \quad Au^a = 0. \end{array} \right. \tag{10.9}$$

On a $u^a \cdot v^a = u^a \cdot Hu^a \geqslant 0$. Pour le déplacement affine il est nécessaire de faire intervenir un pas $\theta \in]0,1[$. Le nouveau point (qui inclut une mise à jour de μ) est

$$x^a := x + \theta u^a; \quad s^a := s + \theta v^a; \quad \lambda^a := \lambda + \theta \eta^a; \quad \mu^a := (1-\theta)\mu. \tag{10.10}$$

L'algorithme prédicteur-correcteur ci-dessous est le premier algorithme à convergence polynomiale dont on a montré la convergence quadratique. Il alterne les déplacements de centrage, et les pas affines dans lesquels θ est le plus grand pas permettant de rester dans le voisinage de taille $\alpha \in]0,1[$ fixe. Autrement dit :

Algorithme 10.4 (pc : Prédicteur Correcteur)

1. Initialisation : choisir $\alpha \in]0, \frac{1}{2}]$ et $\varepsilon > 0$; fournir $(x^0, s^0, \lambda^0, \mu^0) \in \mathcal{V}_\alpha$; $k \leftarrow 0$.
2. $(x, s, \lambda, \mu) \leftarrow (x^k, s^k, \lambda^k, \mu^k)$
3. Calcul du point après centrage $(x^c, s^c, \lambda^c, \mu)$; faire $(x, s, \lambda, \mu) \leftarrow (x^c, s^c, \lambda^c, \mu)$.
4. Déplacement affine : θ plus grand élément de $]0,1[$ tel que $(x^a, s^a, \lambda^a, \mu^a)$ donné par (10.10) est dans le petit voisinage de taille α ; $(x, s, \lambda, \mu) \leftarrow (x^a, s^a, \lambda^a, \mu^a)$.
5. Si $\mu^k \leqslant \varepsilon$, arrêt. Sinon, $k \leftarrow k+1$; $(x^k, s^k, \lambda^k, \mu^k) \leftarrow (x, s, \lambda, \mu)$; aller en 2.

Notons que chaque itération comporte la résolution de deux systèmes linéaires dont les matrices sont différentes ; en effet la direction affine est calculée au point obtenu après centrage. La résolution de ces systèmes linéaires est discutée en section 10.5.

10.3 Analyse de l'algorithme prédicteur-correcteur

Le théorème suivant énonce la complexité de l'algorithme ainsi que sa propriété de convergence quadratique.

Théorème 10.5 (i) *Soit* $\bar{L} := \log(\mu^0/\varepsilon)$. *Alors l'algorithme* **PC** *s'arrête après* $O(\sqrt{n}\bar{L})$ *itérations.*
(ii) *Si* (P) *a une solution strictement complémentaire (telle que* $\bar{x} + \bar{s} > 0$*), et* $\varepsilon = 0$*, alors la suite (infinie)* $\{w^k\}$ *vérifie* $\mu_{k+1} = O((\mu_k)^2)$.

Remarque 10.6 (i) Ce théorème a une caractéristique remarquable : la borne sur le nombre d'itérations ne dépend des données du problème (en dehors de la dimension) que par l'intermédiaire des paramètres μ_0 et ε.
(ii) En pratique μ_0/ε sera de l'ordre de 10^q, $q \leqslant 100$ et donc $\bar{L} \leqslant 100 \log 10$. La borne du nombre d'itérations est donc essentiellement $O(\sqrt{n})$.
(iii) On peut montrer que l'algorithme ainsi obtenu est polynomial, au sens où en choisissant ε assez petit, un algorithme (polynomial) de purification permet de calculer une solution exacte du problème ; voir [BGLS06, Ch. 25]).
(iv) En pratique le nombre d'itérations est remarquablement faible, surtout dans la variante du *grand voisinage* discuté en section 10.4.
(v) Cependant, pour des exemples inhabituels dans lesquels les coefficients du programme linéaire ont un nombre de bits exponentiel en la dimension, $\bar{L}$ devient exponentiel en la dimension. Cette explosion de l'estimation de complexité est reliée au fait que le chemin central peut faire dans ce cas un nombre exponentiel de "virages" [ABGJ14]. Rappelons que l'existence d'un algorithme fortement polynomial en programmation linéaire (ce qui exige notamment que le nombre d'opérations arithmétiques soit indépendant du nombre de bits des coefficients) est un problème ouvert difficile (9-ième problème de Smale). •

Nous donnons dans la suite la démonstration de l'estimation de complexité, et l'idée de la démonstration de la convergence quadratique.

10.3.1 Proximité après centrage

On peut donner un résultat de convergence quadratique du pas de centrage, indépendant des données du problème.

Lemme 10.7 *Notons $w := (x, s, \lambda, \mu)$, et w^c le point obtenu après centrage. Si $w \in \mathcal{V}_\alpha$, avec $\alpha \in]0, 1[$, alors*

$$\delta(w^c) \leqslant \frac{1}{\sqrt{8}} \frac{\delta(w)^2}{1 - \delta(w)}. \tag{10.11}$$

En particulier si $\delta(w) \leqslant 1/2$, alors $\delta(w^c) \leqslant \frac{1}{2}\delta(w)$.

La démonstration est basée sur le lemme suivant, utile à plusieurs reprises :

Lemme 10.8 *(Lemme de Mizuno). Soient u et v dans $\mathbb{R}^n$ tels que $u \cdot v \geqslant 0$. Alors*

$$\|uv\| \leqslant \frac{1}{\sqrt{8}} \|u + v\|^2.$$

Démonstration. Soient β et γ dans $\mathbb{R}$. Alors $\beta\gamma = \frac{1}{4}(\beta + \gamma)^2 - \frac{1}{4}(\beta - \gamma)^2 \leqslant \frac{1}{4}(\beta + \gamma)^2$. D'autre part

$$\|uv\|^2 = \sum_{i=1}^n (u_i v_i)^2 \leqslant \left(\sum_{u_i v_i > 0} u_i v_i \right)^2 + \left(\sum_{u_i v_i < 0} u_i v_i \right)^2.$$

Puisque $u \cdot v \geqslant 0$, on a $\left| \sum_{u_i v_i < 0} u_i v_i \right| \leqslant \sum_{u_i v_i > 0} u_i v_i$ et donc

$$\|uv\|^2 \quad = \quad 2 \left(\sum_{u_i v_i > 0} u_i v_i \right)^2 \leqslant 2 \left(\sum_{u_i v_i > 0} \frac{1}{4}(u_i + v_i)^2 \right)^2 \leqslant \frac{1}{8}(\|u + v\|^2)^2$$

d'où le résultat. ■

Démonstration du lemme 10.7 Notons $\delta := \|\frac{xs}{\mu} - \mathbf{1}\|$. La mesure de proximité après centrage est, en raison de (10.7)

$$\delta^c := \left\| \frac{x^c s^c}{\mu} - \mathbf{1} \right\| = \left\| \frac{xs + su^c + xv^c + u^c v^c}{\mu} - \mathbf{1} \right\| = \frac{\|u^c v^c\|}{\mu}.$$

Divisons (composante par composante) l'équation $su^c + xv^c = \mu\mathbf{1} - xs$ par le vecteur $\sqrt{xs}$ (de composantes $\sqrt{x_i s_i}$, $i = 1$ à n). Il vient

$$\sqrt{\frac{s}{x}} u^c + \sqrt{\frac{x}{s}} v^c = \frac{\mu}{\sqrt{xs}} - \sqrt{xs},$$

donc avec le lemme de Mizuno (nous savons que $u^c \cdot v^c \geqslant 0$)

$$\|u^c v^c\| = \left\| \left(\sqrt{\frac{s}{x}} u^c \right) \left(\sqrt{\frac{x}{s}} v^c \right) \right\| \leqslant \frac{1}{\sqrt{8}} \left\| \frac{\mu}{\sqrt{xs}} - \sqrt{xs} \right\|^2.$$

En conséquence

$$\delta^c \leqslant \frac{1}{\sqrt{8}} \left\| \sqrt{\frac{\mu}{xs}} - \sqrt{\frac{xs}{\mu}} \right\|^2 \leqslant \frac{1}{\sqrt{8}} \left\| \sqrt{\frac{\mu}{xs}} \right\|_\infty^2 \left\| \mathbf{1} - \frac{xs}{\mu} \right\|^2 = \frac{1}{\sqrt{8}} \left\| \frac{\mu}{xs} \right\|_\infty \delta^2.$$

Or $\left\| \frac{xs}{\mu} - \mathbf{1} \right\| = \delta$ implique $\frac{x_i s_i}{\mu} \geqslant 1 - \delta(w)$, d'où $\frac{\mu}{x_i s_i} \leqslant \frac{1}{1 - \delta(w)}$. Combinant avec l'inégalité ci-dessus, on obtient (10.11), et donc la conclusion du lemme. ■

10.3.2 Analyse du pas affine

Il s'agit de déterminer une valeur aussi élevée que possible de la valeur de θ lors du pas affine. Notons δ^a la mesure de proximité du point affine $(x^a, s^a, \lambda^a, \mu^a)$. La condition à satisfaire est $\delta^a \leqslant \alpha$. Notons que, compte-tenu de (10.9) :

$$\delta^a = \left\| \frac{(x+\theta u^a)(s+\theta v^a)}{(1-\theta)\mu} - \mathbf{1} \right\| = \left\| \frac{xs}{\mu} - \mathbf{1} + \theta^2 \frac{u^a v^a}{(1-\theta)\mu} \right\|. \tag{10.12}$$

Lemme 10.9 *Le pas affine vérifie* $\theta \geqslant \sqrt{\dfrac{\alpha}{3n}}$.

Démonstration. D'après (10.12), le pas θ sera accepté s'il satisfait

$$\alpha \geqslant \left\| \frac{xs}{\mu} - \mathbf{1} \right\| + \frac{\theta^2}{1-\theta} \frac{\|u^a v^a\|}{\mu},$$

et comme $\|\frac{xs}{\mu} - \mathbf{1}\| \leqslant \frac{1}{2}\alpha$ d'après le lemme 10.7, également si

$$\tfrac{1}{2}\alpha \geqslant \frac{\theta^2}{1-\theta} \frac{\|u^a v^a\|}{\mu}. \tag{10.13}$$

Admettons pour l'instant la relation

$$\frac{\|u^a v^a\|}{\mu} \leqslant n. \tag{10.14}$$

Alors le pas θ sera accepté s'il vérifie $\frac{\theta^2}{1-\theta} n \leqslant \frac{\alpha}{2}$. Si $\theta \leqslant \dfrac{1}{3}$ alors $\dfrac{1}{1-\theta} \leqslant \dfrac{3}{2}$. Il suffit donc de satisfaire $\frac{3}{2}(\theta\sqrt{n})^2 \leqslant \frac{\alpha}{2}$ avec $\theta \in]0, \dfrac{1}{3}[$, inégalité qui est satisfaite pour $\theta = \sqrt{\dfrac{\alpha}{3n}}$.

Pour démontrer (10.14), appliquons le lemme de Mizuno à l'égalité $su^a + xv^a = -xs$, après la mise à l'échelle obtenue en divisant l'équation précédente par $\sqrt{xs}$. Nous savons que $u^c \cdot v^c \geqslant 0$. Il vient

$$\sqrt{\frac{s}{x}} u^a + \sqrt{\frac{x}{s}} v^a = -\sqrt{xs},$$

et donc

$$\|u^a v^a\| = \left\| \left(\sqrt{\frac{s}{x}} u^a \right) \left(\sqrt{\frac{x}{s}} v^a \right) \right\| \leqslant \frac{1}{\sqrt{8}} \|\sqrt{xs}\|^2 = \frac{1}{\sqrt{8}} \sum_{i=1}^{n} x_i s_i.$$

Or $\left\| \dfrac{xs}{\mu} - \mathbf{1} \right\| \leqslant \dfrac{\alpha}{2} \leqslant \dfrac{1}{4}$, donc $x_i s_i \leqslant \dfrac{5}{4}\mu$ et donc $\|u^a v^a\| \leqslant \frac{5/4}{\sqrt{8}} n\mu \leqslant n\mu$, d'où (10.14). ■

10.3.3 Synthèse de l'estimation de complexité

Nous avons vérifié que le pas de centralisation réduit d'un facteur au moins $\frac{1}{2}$ la mesure de proximité, ce qui permet de vérifier que le pas affine vaut au moins $\sqrt{\dfrac{\alpha}{3n}}$. D'après la formule de mise à jour de μ, on a

$$\mu_k \leqslant \left(1 - \sqrt{\frac{\alpha}{3n}}\right)^k \mu_0.$$

On aura $\mu_k \leqslant \varepsilon$ dès que

$$k \log(1 - \sqrt{\frac{\alpha}{3n}}) + \log \mu_0 \leqslant \log \varepsilon,$$

soit

$$k|\log(1-\sqrt{\frac{\alpha}{3n}})| \geqslant \log\frac{\mu_0}{\varepsilon} = \bar{L}.$$

Or $|\log(1-\beta)| \geqslant \beta$ si $\beta \in [0,1[$ donc $k = O(\sqrt{n}\bar{L})$ est (à l'arrondi à l'entier supérieur près) une estimation supérieure du nombre d'itérations de l'algorithme.

10.3.4 Analyse de la convergence quadratique

Nous donnons seulement le schéma. Puisque $\mu_{k+1} = (1-\theta_k)\mu_k$ il faut vérifier que $\theta_k = 1 - O(\mu_k)$ pour obtenir $\mu_{k+1} = O((\mu_k)^2)$. D'après la démonstration du lemme 10.9, (10.13) est une condition suffisante d'acceptation du pas. *A fortiori*, puisque $\theta \leqslant 1$, le pas sera accepté si $\|u^a v^a\|/\mu \leqslant \frac{1}{2}(1-\theta)\alpha$. Sous l'hypothèse de complémentarité stricte, on peut vérifier (voir [BGLS06, Partie IV]) que $u^a = O(\mu)$, $v^a = O(\mu)$ et donc $\|u^a v^a\|/\mu = O(\mu)$, d'où $(1-\theta_k) = O(\mu_k)$, comme il fallait le vérifier.

10.4 Algorithme de grands voisinages

Décrivons brièvement dans cette section une variante beaucoup plus efficace en pratique (mais dont l'estimation du nombre d'itérations est moins bonne !) que l'algorithme de petits voisinages. Le *grand voisinage* de la trajectoire centrale est défini, pour $\varepsilon > 0$, comme

$$\mathcal{N}_\epsilon := \left\{(x,s,\mu);\quad (x,s) \in F(P);\quad \mu > 0;\quad \varepsilon\mathbf{1} \leqslant \frac{xs}{\mu} \leqslant \varepsilon^{-1}\mathbf{1}\right\}. \tag{10.15}$$

L'algorithme de grand voisinage ressemble à celui du petit voisinage ; dans les deux cas on utilise les directions de centrage et affines. Le pas affine est le plus grand permettant de rester dans le voisinage considéré. Par contre, dans la variante du grand voisinage, il faut faire intervenir un pas également pour la direction de centrage (car le pas unité ne ramène pas nécessairement à l'intérieur du grand voisinage).

Algorithme 10.10 (PCG : Prédicteur Correcteur, Grand Voisinage)

1. Initialisation : choisir $\varepsilon \in]0, \frac{1}{2}]$, $\varepsilon > 0$, $(x^0, s^0, \lambda^0, \mu^0) \in \mathcal{V}_\alpha$; $k \leftarrow 0$.
2. Centrage : Choix de $\theta^c \in]0,1]$ tel que le point centré $(x,s,\lambda) \leftarrow (x,s,\lambda) + \theta^c(u^c, v^c, \eta^c)$ soit dans l'intérieur de $\mathcal{N}_\epsilon$.
3. Déplacement affine : θ plus grand élément de $]0,1[$ tel que $(x^a, s^a, \lambda^a, \mu^a)$ donné par (10.10) est dans $\mathcal{N}_\epsilon$; $(x,s,\lambda,\mu) \leftarrow (x^a, s^a, \lambda^a, \mu^a)$.
4. Si $\mu^{k+1} \leqslant \varepsilon$, arrêt. Sinon, $k \leftarrow k+1$; $w \leftarrow w^k$; aller en 2).

Sous certaines conditions sur le choix de θ^c, on démontre (voir [BGLS06, Partie IV]) que l'algorithme s'arrête après au plus $O(n\bar{L})$ itérations, et que sa convergence est quadratique.

La figure 10.2 montre un comportement typique de la convergence de l'algorithme de grands voisinages pour des données creuses générées aléatoirement avec un facteur de remplissage de 1%. L'algorithme est codé en Scilab. Le problème test a 10000 inconnues et 100 contraintes linéaires. L'exécution ne demande que quelques secondes sur un PC. La convergence est obtenue en une trentaine d'itérations. Le tracé du logarithme décimal de μ en fonction des itérations montre une décroissance rapide suivie d'une phase de "plateau", avant la convergence finale rapide.

10.5 Aspects pratiques

Les deux étapes non triviales de l'algorithme sont le calcul des directions de centrage et affine, et la recherche linéaire. Les directions sont solution de (10.7) et (10.9). Dans le cas d'un problème quadratique,

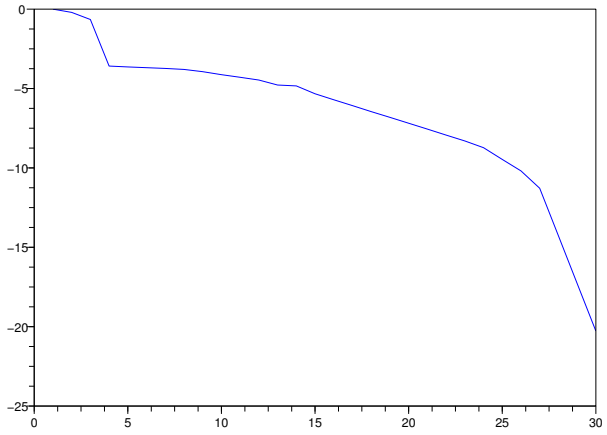


FIGURE 10.2 – Logarithme décimal de μ en fonction des itérations

résoudre ces systèmes revient à calculer (u, v, η), solution de

$$\left\{ \begin{array}{l} su + xv = \gamma\mu\mathbf{1} - xs, \\ Au = 0, \quad Hu + A^\top \eta = v, \end{array} \right. \tag{10.16}$$

avec $\gamma = 1$ pour la direction de centrage, et 0 pour la direction affine. Posons $d := \sqrt{x/s}$, $D := \mathrm{diag}(d)$ (matrice diagonale, dont la diagonale égale d). Après la *mise à l'échelle* $\bar{u} = d^{-1}u$, $\bar{v} := dv$, on obtient le système

$$\left\{ \begin{array}{l} \bar{u} + \bar{v} = \bar{f}, \\ \bar{A}\bar{u} = 0, \quad \bar{H}\bar{u} + \bar{A}^\top \eta - \bar{v} = 0, \end{array} \right. \tag{10.17}$$

avec $\bar{f} := (\gamma\mu\mathbf{1} - xs)/\sqrt{xs}$, $\bar{A} := AD$ et $\bar{H} := DHD$. Eliminant $\bar{v}$, il vient

$$\left\{ \begin{array}{l} (I + \bar{H})\bar{u} + \bar{A}^\top \eta = \bar{f}, \\ \bar{A}\bar{u} = 0. \end{array} \right. \tag{10.18}$$

Eliminant maintenant $\bar{u}$, on obtient le système réduit

$$\bar{A}(I + \bar{H})^{-1}\bar{A}^\top \eta = \bar{A}(I + \bar{H})^{-1}\bar{f}. \tag{10.19}$$

On peut donc résoudre le système symétrique (mais non défini positif) (10.18). Ce dernier s'interprète comme le système d'optimalité du problème quadratique convexe

$$\min_{\bar{u}} \bar{f}^\top \bar{u} + \frac{1}{2}\bar{u}^\top (I + \bar{H})\bar{u}; \quad \bar{A}\bar{u} = 0.$$

On peut aussi résoudre le système réduit (10.19). Si les données A et H sont creuses, alors le système (10.18) est creux. Il n'en est pas toujours de même pour le système réduit, et ceci pour deux raisons. Tout d'abord, $(I + \bar{H})^{-1}$ peut être pleine ; cependant, dans le cas de l'optimisation linéaire, et plus généralement si H est diagonale, cette matrice est diagonale. Alors la matrice du système réduit a le même creux que $AA^\top$. D'autre part, il suffit que A ait au moins une colonne pleine pour que $AA^\top$ soit pleine.

La matrice $A(I + \bar{H})^{-1}A^\top$ est définie positive si A est de rang plein. Dans ce cas on peut la factoriser par la méthode de Cholesky. La factorisation étant stable quel que soit l'ordre des pivots, on peut choisir cet ordre de manière à minimiser le remplissage du facteur de Cholesky, par une factorisation "symbolique" qui ne sera effectuée qu'une seule fois.

10.6 Compléments sur les algorithmes de points intérieurs

10.6.1 Complémentarité linéaire monotone

On peut voir l'optimisation quadratique convexe comme un cas particulier de *problème de complémentarité linéaire*, défini par l'ensemble de relations ci-dessous :

$$\left\{\begin{array}{l} xs = 0 \\ Qx + Rs = h \\ x \geqslant 0,\ s \geqslant 0 \end{array}\right. \qquad (LCP)$$

avec Q et R matrices $n \times n$ et $h \in \mathbb{R}^n$. On trouvera dans [CPS92] des exemples de problème de complémentarité linéaire. Le problème est dit *monotone* si

$$Qu + Rv = 0 \Rightarrow u \cdot v \geqslant 0.$$

On définit l'ensemble réalisable de (LCP) comme

$$F(LCP) := \{x \in \mathbb{R}^n_+,\ s \in \mathbb{R}^n_+ \ ;\ Qx + Rs = h\}.$$

Exemple 10.11 Reprenons les conditions d'optimalité du problème (P) énoncé au début du chapitre. On peut éliminer λ du système d'optimalité (10.4) en notant que la seconde relation équivaut à $\nabla f(x) - s \in \mathcal{R}(A^\top)$. Or $\mathcal{R}(A^\top) = \mathcal{N}(A)^\perp$. Puisque A est de rang p, $\mathcal{N}(A)$ a pour dimension $n - p$. Soit B une matrice $n \times (n-p)$ dont les vecteurs colonnes forment une base de $\mathcal{N}(A)$; alors

$$\nabla f(x) - s \in \mathcal{R}(A^\top) \Leftrightarrow B^\top(\nabla f(x) - s) = 0,$$

et donc (10.4) équivaut à

$$\left\{\begin{array}{l} xs = \mu \mathbf{1} \\ B^\top(\nabla f(x) - s) = 0 \\ Ax = b \\ x \geqslant 0,\ s \geqslant 0 \end{array}\right. \qquad (10.20)$$

Lorsque $\mu = 0$, et si f est quadratique convexe, le problème (10.5) est bien un cas particulier de problème de complémentarité linéaire monotone. En effet, il inclut n relations d'égalité, et par ailleurs, la relation homogène $B^\top v = B^\top H u$; $Au = 0$ équivaut à $(v - Hu) \perp \mathcal{N}(A)$; $Au = 0$, donc implique $0 = u \cdot (v - Hu)$, soit $u \cdot v = u \cdot Hu \geqslant 0$, d'où la monotonie.

Lemme 10.12 *L'ensemble des solutions de (LCP) monotone est convexe.*

Démonstration. Soient $(x^\sharp, s^\sharp)$ et $(x^\flat, s^\flat)$ deux solutions, et $\alpha \in (0,1)$. Posons $(x,s) := \alpha(x^\sharp, s^\sharp) + (1-\alpha)(x^\flat, s^\flat)$. Il faut monter que $(x,s) \in S(LCP)$. On obtient facilement $Qx + Rs = h$, $x \geqslant 0$ et $s \geqslant 0$. Reste à montrer $xs = 0$.

De $(x^\sharp - x^\flat) + R(s^\sharp - s^\flat) = 0$, on déduit que $(x^\sharp - x^\flat) \cdot (s^\sharp - s^\flat) \geqslant 0$, ou encore $(x^\sharp) \cdot s^\flat + (x^\flat) \cdot s^\sharp \leqslant (x^\sharp) \cdot s^\sharp + (x^\flat) \cdot s^\flat$. Puisque $(x^\sharp, s^\sharp)$ et $(x^\flat, s^\flat)$ sont solutions de (LCP), le membre de droite est nul. Le membre de gauche étant une somme de termes positifs, tous les termes sont nuls ; donc $x^\sharp s^\flat = x^\flat s^\sharp = 0$. Développant alors xs en utilisant les expressions de x et s, on obtient bien $xs = 0$. ■

La *trajectoire centrale* pour (LCP) est définie comme l'ensemble des (x,s) tels que pour un certain $\mu > 0$

$$\left\{\begin{array}{l} xs = \mu \mathbf{1} \\ Qx + Rs = h \\ x \geqslant 0,\ s \geqslant 0 \end{array}\right.$$

Il est facile de vérifier que cette définition étend celle donnée précédemment pour le problème d'optimisation convexe.

On étend aisément au problème monotone (LCP) les algorithmes prédicteur correcteur, en définissant les solutions de centrage et affines comme solution de (comparer à (10.16))

$$\begin{cases} su + xv = \gamma\mu\mathbf{1} - xs, \\ Qu + Rv = 0, \end{cases} \tag{10.21}$$

avec $\gamma = 1$ pour la direction de centrage, et 0 pour la direction affine. Il existe une variante de l'algorithme qui permet d'obtenir une convergence surlinéaire (c'est à dire $\mu_{k+1}/\mu_k \to 0$) même en l'absence de complémentarité stricte ; voir [BGLS06, Partie IV].

10.7 Notes

Khachiyan [Kha79] a énoncé le premier algorithme polynomial pour la programmation linéaire. La résolution approchée d'une succession de problèmes (P_μ), quand $\mu \to 0$ fournit donc un moyen de résoudre (P) de façon approchée. L'idée date des années 50. Karmarkar [Kar84] a montré en 1984 que cette approche permet d'énoncer des algorithmes polynomiaux et efficaces pour les problèmes de grande taille. Son algorithme, de complexité plus réduite que ceux alors connus, s'appuye sur une transformation projective et un potentiel logarithmique. Les familles prédicteur-correcteur sont très employées dans le cadre des "grands voisinages", avec corrections multiples du déplacement. De nombreux ouvrages présentent les différents familles d'algorithmes de points intérieurs : voir par exemple [BGLS06, Partie IV], [NN94, RTV97, Sai95, Ter96].

Annexe A

Algorithme glouton pour le problème de l'arbre couvrant de coût minimum

Afin d'être complet, nous présentons brièvement dans cette annexe l'algorithme de Kruskal pour le calcul de l'arbre couvrant de coût minimum, dont nous avons eu besoin afin de présenter une relaxation classique du voyageur de commerce (borne du 1-arbre, section 5.1.4).

A.1 Généralités sur les méthodes gloutonnes

On dit qu'un algorithme pour minimiser un critère est *glouton* ("greedy", en anglais), s'il construit une solution admissible en se ramenant à une suite de décisions, que l'on prend à chaque fois au mieux en fonction d'un critère local, en ne remettant jamais en question les décisions précédentes. Lorsque la solution admissible ainsi obtenue est sous-optimale, on parle d'*heuristique gloutonne*. Par exemple, si l'on a un certain nombre de colis de tailles variées à mettre dans des containers, et si l'on veut minimiser le nombre de containers utilisés (c'est une version du problème dit de "packing"), on peut imaginer une méthode consistant à trier les colis en fonction du volume, et à rentrer les colis dans les containers en commençant par les plus gros : c'est là un exemple typique d'heuristique gloutonne. L'intérêt des heuristiques gloutonnes est d'être souvent très simples à implémenter. Leur défaut est évidemment leur myopie, ainsi que la difficulté à évaluer l'écart à l'optimum. Il est cependant des classes particulières de problèmes pour lesquels une méthode gloutonne fournit une solution optimale, incluant le problème de l'arbre couvrant de poids minimum, dont nous détaillons maintenant la résolution.

A.2 Algorithme de Kruskal pour le problème de l'arbre couvrant de coût minimum

Rappelons qu'une *forêt* est un graphe (non-orienté) sans circuit. Un *arbre* est une forêt connexe. On dit qu'un sous-graphe $\mathcal{G}'$ couvre un graphe $\mathcal{G}$ si chaque sommet de $\mathcal{G}$ est extrémité d'au moins une arête de $\mathcal{G}'$.

Étant donné un graphe (non-orienté) connexe $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, dont les arêtes sont munies d'une fonction coût $\mathcal{E} \to \mathcal{R}$, $\{i, j\} \mapsto c_{ij}$, le problème de l'*arbre couvrant de coût minimum*, déjà défini et motivé dans l'Exemple 1.18, consiste à trouver un arbre couvrant $\mathcal{T}$ dont le coût

$$\sum_{\{i,j\} \in \mathcal{T}} c_{ij} \ ,$$

est minimum.

L'algorithme de Kruskal construit une suite de forêts. On part de la forêt comprenant tous les sommets et aucune arête. A chaque étape, on choisit de rajouter à la forêt, parmi toutes les arêtes dont l'adjonction

ne crée pas de circuit, celle dont le coût est minimum. L'algorithme termine quand la forêt est un arbre couvrant.

Théorème A.1 *Si $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ est un graphe (non-orienté) connexe, muni d'une fonction coût arbitraire $c : \mathcal{E} \to \mathbb{R}$, l'algorithme de Kruskal fournit un arbre couvrant de coût minimum.*

Afin de montrer le Théorème A.1, nous énonçons une propriété très élémentaire des arbres couvrants, dont la vérification est laissée au lecteur en guise d'exercice.

Lemme A.2 (Lemme d'échange) *Si $\mathcal{T}$ est un arbre couvrant d'un graphe $\mathcal{G}$, et si i et j sont deux sommets de $\mathcal{G}$, il existe un unique chemin de $\mathcal{T}$ reliant i à j. En outre, si $\{i, j\}$ est une arête de $\mathcal{G}$ qui n'appartient pas à $\mathcal{T}$, on obtient encore un arbre couvrant si l'on remplace une arête quelconque du chemin de $\mathcal{T}$ reliant i à j par $\{i, j\}$.*

Nous pouvons maintenant énoncer la condition d'optimalité.

Lemme A.3 *Soit $\mathcal{G}$ et c comme dans le Théorème A.1, et soit $\mathcal{T}$ un arbre couvrant $\mathcal{G}$. Les assertions suivantes sont équivalentes.*

1. *$\mathcal{T}$ est de coût minimum ;*
2. *pour chaque arête $\{i, j\}$ qui n'est pas dans $\mathcal{T}$, l'unique chemin de $\mathcal{T}$ reliant i et j est formé d'arêtes dont chacune est de coût inférieur ou égal à c_{ij} ;*
3. *pour chaque arête $\{r, s\}$ de $\mathcal{T}$, pour toute composante connexe C du graphe obtenu en retirant $\{r, s\}$ de $\mathcal{T}$, et pour toute arête $\{i, j\}$ dont une extrémité et une seule est dans C, $c_{rs} \leqslant c_{ij}$.*

Démonstration. L'implication (non 2)⇒(non 1) résulte aussitôt du lemme d'échange : si $\{i, j\}$ est une arête qui n'est pas dans $\mathcal{T}$, et si $\{r, s\}$ est une arête de l'unique chemin de $\mathcal{T}$ reliant i et j, telle que $c_{ij} < c_{rs}$, on obtient en mettant $\{i, j\}$ à la place de $\{r, s\}$ dans $\mathcal{T}$ un nouvel arbre couvrant de coût strictement inférieur à celui de $\mathcal{T}$.

Nous montrons maintenant (non 3)⇒(non 2). Remarquons tout d'abord que le graphe obtenu en retirant $\{r, s\}$ de $\mathcal{T}$ a exactement deux composantes connexes, C et $\overline{C} = \mathcal{V} \setminus C$. Supposons qu'on ait une arête $\{i, j\}$ avec $i \in C$ et $j \in \overline{C}$, telle que $c_{ij} < c_{rs}$. L'unique chemin reliant reliant i à j dans $\mathcal{T}$ contient nécessairement $\{r, s\}$, ce qui montre que la condition (2) n'est pas satisfaite.

Nous montrons enfin (3)⇒(1). Soit $\mathcal{T}$ un arbre vérifiant (3), et soit $\mathcal{T}'$ un arbre de coût optimal, que l'on peut choisir tel que le nombre d'arêtes en commun entre $\mathcal{T}$ et $\mathcal{T}'$ soit maximal. On va montrer que $\mathcal{T} = \mathcal{T}'$. Dans le cas contraire, on peut trouver une arête $\{r, s\}$ dans $\mathcal{T}$ et pas dans $\mathcal{T}'$. Le graphe obtenu en rajoutant $\{r, s\}$ à $\mathcal{T}'$ contient un circuit, $\mathcal{C}$, contenant l'arête $\{r, s\}$. Considérons maintenant les deux composantes connexes C et $\overline{C}$ du graphe obtenu en enlevant $\{r, s\}$ à $\mathcal{T}$. Comme le circuit $\mathcal{C}$ contient déjà une arête reliant C à $\overline{C}$, à savoir $\{r, s\}$, il doit nécessairement contenir une autre arête, $\{i, j\}$, reliant C et $\overline{C}$. Le graphe $\mathcal{T}''$ obtenu en remplaçant $\{i, j\}$ par $\{r, s\}$ dans $\mathcal{T}'$ est encore un arbre couvrant, d'après le lemme d'échange, et la condition (3) montre que le coût de $\mathcal{T}''$ est inférieur ou égal à celui de $\mathcal{T}'$. Ainsi, $\mathcal{T}''$ est encore un arbre couvrant de coût minimum, et comme $\mathcal{T}''$ a en commun avec $\mathcal{T}$ une arête de plus que $\mathcal{T}'$, on contredit l'hypothèse de maximalité dans la définition de $\mathcal{T}'$, On a montré $\mathcal{T}' = \mathcal{T}$. ■

Démonstration du Théorème A.1. Il est immédiat que l'algorithme de Kruskal, appliqué à un graphe connexe, termine avec un arbre couvrant. Cet arbre vérifie l'assertion 2 du Lemme A.3. ■

Remarque A.4 Il est possible d'implémenter l'algorithme de Kruskal en temps $O(|\mathcal{E}||\log \mathcal{E}|)$ où $|\mathcal{E}|$ est le nombre d'arêtes du graphe, voir [CCPS98]. •

Remarque A.5 L'optimalité de méthodes gloutonnes est toujours le signe de fortes propriétés de structure. Par exemple, le lecteur a déjà rencontré un algorithme glouton en algèbre linéaire : on peut voir le problème consistant à fabriquer une base d'un espace vectoriel de dimension finie E comme un problème d'optimisation,

consistant à maximiser le cardinal d'une famille libre de E. L'algorithme qui consiste à partir d'une famille vide, et à chaque étape, à rajouter dans la famille un vecteur quelconque de E qui n'est pas combinaison linéaire des vecteurs déjà dans la famille, est un algorithme glouton (qui fournit une solution optimale, c'est-à-dire une base). La preuve de correction de la méthode repose sur un résultat simple d'algèbre linéaire, le lemme d'échange, dont le lecteur aura noté l'analogie avec le Lemme A.2 ci-dessus. Plus généralement, les propriétés qui permettent de montrer que l'algorithme glouton est correct ont été étudiées dans le cadre de la théorie des matroïdes et anti-matroïdes, voir notamment [CCPS98]. •

Bibliographie

[AB09] S. Arora and B. Barak. *Computational complexity : a modern approach*. Cambridge University Press, Cambridge, 2009.

[Aba86] J. Abadie. Generalized reduced gradient and global Newton methods. In *Optimization and related fields (Erice, 1984)*, volume 1190 of *Lecture Notes in Math.*, pages 1–20. Springer, Berlin, 1986.

[ABCC06] D.L. Applegate, R.E. Bixby, V. Chvátal, and W.J. Cook. *The traveling salesman problem*. Princeton Series in Applied Mathematics. Princeton University Press, Princeton, NJ, 2006. A computational study.

[ABG13] M. Akian, R. Bapat, and S. Gaubert. Max-plus algebras. In L. Hogben, editor, *Handbook of Linear Algebra (Discrete Mathematics and Its Applications)*, volume 39. Chapman & Hall/CRC, 2013. Chapter 25, Second Edition (First edition, 2006).

[ABGJ14] X. Allamigeon, P. Benchimol, S. Gaubert, and M. Joswig. Long and winding central paths, 2014. arXiv :1405.4161.

[AHU58] K.J. Arrow, L. Hurwicz, and H. Uzawa. *Studies in linear and non-linear programming*. With contributions by H. B. Chenery, S. M. Johnson, S. Karlin, T. Marschak, R. M. Solow. Stanford Mathematical Studies in the Social Sciences, vol. II. Stanford University Press, Stanford, 1958.

[AMO93] R.K. Ahuja, T.L Magnanti, and J.B. Orlin. *Network flows*. Prentice Hall, 1993.

[Bac03] N. Bacaer. Modèles mathématiques pour l'optimisation des rotations. *Comptes Rendus de l'Académie d'Agriculture de France*, (3) :52, 2003.

[Bar02] A. Barvinok. *A course in Convexity*. AMS, 2002.

[BCOQ92] F. Baccelli, G. Cohen, G.J. Olsder, and J.P. Quadrat. *Synchronization and Linearity*. Wiley, 1992.

[Bel58] R. Bellman. On a routing problem. *Quart. Appl. Math.*, 16, 1958.

[Bel61] R. Bellman. *Dynamic programming*. Princeton University Press, Princeton, 1961.

[Ben62] J.F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4 :238–252, 1962.

[Ber70] C. Berge. *Graphes et hypergraphes*. Dunod, Paris, 1970. Monographies Universitaires de Mathématiques, No. 37.

[Ber01] D.P. Bertsekas. *Dynamic Programming and Optimal Control (2nd edition), Vol I & II*. Athena Scientific, Belmont, Mass., 2000,2001.

[BFG+00] R.E. Bixby, M. Fenelon, Z. Gu, E. Rothberg, and R. Wunderling. MIP : theory and practice—closing the gap. In *System modelling and optimization (Cambridge, 1999)*, pages 19–49. Kluwer Acad. Publ., Boston, MA, 2000.

[BGLS06] J.F. Bonnans, J. Ch. Gilbert, C. Lemaréchal, and C. Sagastizábal. *Numerical Optimization : theoretical and numerical aspects*. Universitext. Springer-Verlag, Berlin, 2006. second edition.

[BL00] J.M. Borwein and A.S. Lewis. *Convex analysis and nonlinear optimization*. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC, 3. Springer-Verlag, New York, 2000.

[Bla77] R.G. Bland. New finite pivoting rules for the simplex method. *Mathematics of Operations Research*, 2(2) :103–107, 1977.

[Bon06] J.F. Bonnans. *Optimisation Continue*. Dunod, Paris, 2006.

[Bor91] V. S. Borkar. *Topics in controlled Markov chains*, volume 240 of *Pitman Research Notes in Mathematics Series*. Longman Scientific & Technical, Harlow ; copublished in the United States with John Wiley & Sons, Inc., New York, 1991.

[BP03] E. Balas and M. Perregaard. A precise correspondence between lift-and-project cuts, simple disjunctive cuts, and mixed integer Gomory cuts for 0-1 programming. *Math. Program.*, 94(2-3, Ser. B) :221–245, 2003. The Aussois 2000 Workshop in Combinatorial Optimization.

[BS00] J.F. Bonnans and A. Shapiro. *Perturbation analysis of optimization problems*. Springer-Verlag, New York, 2000.

[BTN01] A. Ben-Tal and A. Nemirovski. *Lectures on modern convex optimization*. MPS/SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001. Analysis, algorithms, and engineering applications.

[CCPS98] J. Cook, W.H. Cunningham, W.R. Pulleybank, and A. Schrijver. *Combinatorial optimization*. Wiley, 1998.

[CGK+96] C.S. Chekuri, A.V. Goldberg, D.R. Karger, M. S. Levine, and C. Stein. Experimental study of minimum cut algorithms. Technical Report 96-132, NEC Research Institute, Inc., 1996.

[CHS01] R. Cori, G. Hanrot, and J.-M. Steyaert. *Conception et analyse d'algorithmes, seconde partie : complexité*. École Polytechnique, 2001. Cours de majeure d'informatique.

[Chv73] V. Chvátal. Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete Math.*, 4 :305–337, 1973.

[Chv83] V. Chvátal. *Linear programming*. Freeman and Co., 1983.

[Cob65] A. Cobham. The intrinsic computational difficulty of functions. In *Proc. Logic, Methodology, and Philosophy of Science II*. North Holland, 1965.

[Coo71] S. Cook. The complexity of theorem proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158, 1971.

[Coo00] S. Cook. The P versus NP problem, 2000. Official Problem Description, http ://www.claymath.org.

[Cor08] G. Cornuéjols. Valid inequalities for mixed integer linear programming. *Mathematical Programming Ser. B*, 112(1) :3–44, 2008.

[CPS92] R.W. Cottle, J.-S. Pang, and R.E. Stone. *The Linear Complementarity Problem*. Academic Press, New York, 1992.

[Dan59] G.B. Dantzig. Note on solving linear programs in integers. *Naval Res. Logist. Quart*, 6 :75–76, 1959.

[Dan63] G.B. Dantzig. *Linear Programming and extensions*. Princeton University Press, Princeton, 1963.

[DW59] G.B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8, 1959.

[Edm65] J. Edmonds. Paths, trees, and flowers. *Canadian J. Math.*, 17 :449–467, 1965.

[ET76] I. Ekeland and R. Temam. *Convex analysis and variational problems*, volume 1 of *Studies in Mathematics and its Applications*. North-Holland, Amsterdam, 1976. French edition : Analyse convexe et problèmes variationnels, Dunod, Paris, 1974.

[FF56] L.R. Ford and D.R. Fulkerson. Maximal flow through a network. *Canadian J. Math.*, 8 :399–404, 1956.

[FG92] J.J.H. Forrest and D. Goldfarb. Steepest-edge simplex algorithms for linear programming. *Mathematical Programming, Ser. A*, 57 :341–374, 1992.

[FV97] J. Filar and K. Vrieze. *Competitive Markov decision processes*. Springer-Verlag, New York, 1997.

[Geo72] A.M. Geoffrion. Generalized Benders decomposition. *Journal of Optimization Theory and Applications*, 10 :237–260, 1972.

[GG61] P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting-stock problem. *Operations Research*, 9 :849–859, 1961.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and intractability*. Freeman, 1979.

[GLS93] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, second edition, 1993.

[GM09] M. Gondran and M. Minoux. *Graphes et algorithmes*. Collection EDF R&D. Editions Tec & Doc, Paris, fourth edition, 2009.

[GM12] B. Gärtner and J. Matoušek. *Approximation algorithms and semidefinite programming*. Springer, Heidelberg, 2012.

[Gom58] R.E. Gomory. Outline of an algorithm for integer solutions to linear programs. *Bull. Amer. Math. Soc.*, 64 :275–278, 1958.

[Hel00] C. Helmberg. Semidefinite programming for combinatorial optimization. Report 00-34, ZIB, 2000. Habilitationsscrhift.

[HL96] J.B. Hiriart-Urruty and C. Lemaréchal. *Convex analysis and minimization algorithms*. Springer-Verlag, Berlin, 1996.

[HLL99] O. Hernández-Lerma and J.-B. Lasserre. *Further topics on discrete-time Markov control processes*, volume 42 of *Applications of Mathematics (New York)*. Springer-Verlag, New York, 1999.

[Kar84] N. Karmarkar. A new polynomial time algorithm for linear programming. *Combinatorica*, 4 :373–395, 1984.

[Kha79] L.G. Khachiyan. A polynomial algorithm in linear programming. *Doklady Adad. Nauk SSSR*, 244 :1093–1096, 1979. Trad. anglaise : Soviet Math. Doklady 20(1979), 191-194.

[KT06] J. Kleinberg and E. Tardos. *Algorithm Design*. Pearson Education, Inc., 2006.

[KV02] B. Korte and J. Vygen. *Combinatorial optimization*. Springer, 2002.

[Las02] J.B. Lasserre. Semidefinite programming vs. LP relaxations for polynomial programming. *Mathematics of Operations Research*, 27 :347–360, 2002.

[Las06] J.B. Lasserre. Convergent SDP-relaxations for polynomial optimization with sparsity. In *Mathematical software—ICMS 2006*, volume 4151 of *Lecture Notes in Comput. Sci.*, pages 263–272. Springer, Berlin, 2006.

[Lee04] J. Lee. *A first course in combinatorial optimization*. Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge, 2004.

[Lem03] C. Lemaréchal. The omnipresence of Lagrange. *4OR*, 1(1) :7–25, 2003.

[Lev73] L. Levin. Universal'nye perebornye zadachi. *Problemy Peredachi Informatsii*, 9(3) :265–266, 1973.

[Lov] L. Lovász. Semidefinite programs and combinatorial optimization. Lecture notes, http ://www.research.microsoft.com/˜lovasz.

[LR06] J.T. Linderoth and T.K. Ralphs. Noncommercial software for mixed-integer linear programming. In J. Karlof, editor, *Integer programming : Theory and Practice*, Oper. Res. Ser., pages 253–303. CRC Press, Boca Raton, FL, 2006.

[Mar57] H.M. Markowitz. The elimination form of the inverse and its application to linear programming. *Management Science. Journal of the Institute of Management Science. Application and Theory Series*, 3 :255–269, 1957.

[MG07] J. Matousek and B. Gärtner. *Understanding and Using Linear Programming*. Springer, 2007.

[NN94] Y. Nesterov and A. Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13 of *SIAM Studies in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1994.

[NS03] A. Neyman and S. Sorin, editors. *Stochastic games and applications*, volume 570 of *NATO Science Series C : Mathematical and Physical Sciences*, Dordrecht, 2003. Kluwer Academic Publishers.

[NW99] G. Nemhauser and L. Wolsey. *Integer and combinatorial optimization*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons Inc., New York, 1999. Reprint of the 1988 original, A Wiley-Interscience Publication.

[Pap95] C.H. Papadimitriou. *Computational complexity*. Addison Wesley, 1995.

[PP91] M.V.F. Pereira and L. M.V.G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52(2, Ser. B) :359–375, 1991.

[Put93] M. Putinar. Positive polynomials on compact semi-algebraic sets. *Indiana Univ. Mathematics J.*, 42 :969–984, 1993.

[Rei82] J.K. Reid. A sparsity-exploiting variant of the Bartels-Golub decomposition for linear programming bases. *Math. Programming*, 24 :55–69, 1982.

[Roc70] R.T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, New Jersey, 1970.

[RS03] A. Ruszczynski and A. Shapiro, editors. *Stochastic Programming*, volume 10 of *Handbook in Operations Research and Management*. Elsevier, Amsterdam, 2003.

[RTV97] C. Roos, T. Terlaky, and J.-Ph. Vial. *Theory and algorithms for linear optimization*. John Wiley & Sons Ltd., Chichester, 1997. An interior point approach.

[SA94] H.D. Sherali and W.P. Adams. A hierarchy of relaxations and convex hull characterizations for mixed-integer zero-one programming problems. *Discrete Applied Mathematics*, 52(1) :83–106, 1994.

[Sai95] R. Saigal. *Linear Programming : A Modern Integrated Analysis*. Kluwer Academic Publishers, Boston, 1995.

[Sch86] A. Schrijver. *Theory of linear and integer programming*. Wiley, 1986.

[Sch03] A. Schrijver. *Combinatorial optimization. Polyhedra and efficiency. Vol. A, B, C*, volume 24 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2003.

[Sch05] M. Schweighofer. Optimization of polynomials on compact semialgebraic sets. *SIAM J. Optimization*, 15 :805–825, 2005.

[Ter96] T. Terlaky, editor. *Interior Point Methods of Mathematical Programming*. Kluwer Academic Publishers, Boston, 1996.

[TZ93] T. Terlaky and S.Z. Zhang. Pivot rules for linear programming : a survey on recent theoretical developments. *Ann. Oper. Res.*, 46/47(1-4) :203–233, 1993. Degeneracy in optimization problems.

[vNM44] J. von Neumann and O. Morgenstern. *Theory of games and economic behavior*. Princeton University Press, Princeton, New Jersey, 1944.

[WKKM09] H. Waki, S. Kim, M. Kojima, and M. Muramatsu. Algorithm 883 : sparsePOP—a sparse semidefinite programming relaxation of polynomial optimization problems. *ACM Trans. Math. Software*, 35(2) :Art. 15, 13, 2009.

[Wol63] P. Wolfe. *Recent Advances in Mathematical Programming*, chapter Methods of nonlinear programming, pages 67–86. McGraw-Hill, 1963.

[Wol98] L.A. Wolsey. *Integer programming*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons Inc., New York, 1998.

[WSV00] H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors. *Handbook of semidefinite programming*. Kluwer Academic Publishers, Boston, MA, 2000.

Liste des notations

- $|I|$: cardinal d'un ensemble I.
- $C^o = \{y \in \mathbb{R}^n;\ y \cdot x \leqslant 1,\ \text{pour tout}\ \ x \in C\}$: ensemble polaire d'un sous-ensemble convexe C de $\mathbb{R}^n$.
- $\mathrm{conv}(X)$: enveloppe convexe d'un ensemble X (c'est-à-dire, ensemble des combinaisons convexes d'un nombre fini d'éléments de X).
- $\mathrm{cone}(X)$: cône convexe engendré par un ensemble X (c'est-à-dire, ensemble des combinaisons linéaires à coefficients positifs d'un nombre fini d'éléments de X).
- $\mathrm{épi}\, f = \{(x, y) \mid\ y \geqslant f(x)\}$: épigraphe d'une fonction f.
- $F(P)$: ensemble des points admissibles d'un problème d'optimisation (P).
- $\lambda_{\max}(A)$: valeur propre maximale d'une matrice symétrique réelle A.
- $\lambda_{\min}(A)$: valeur propre minimale d'une matrice symétrique réelle A.
- $K^* = \{y \mid\ y \cdot x \geqslant 0,\ \ \forall x \in K\}$: cône dual d'un cône K.
- $\mathrm{Max}_{x \in X}\, f(x)$: définition formelle d'un problème de maximisation (cette écriture ne suppose pas que le maximum est atteint).
- $\mathrm{Min}_{x \in X}\, f(x)$: définition formelle d'un problème de minimisation (cette écriture ne suppose pas que le minimum est atteint).
- $\mathcal{N}(A)$ ou $\mathrm{Ker}\, A$: noyau à droite d'une matrice A, soit $\mathcal{N}(A) = \{x \mid\ Ax = 0\}$.
- $(\mathcal{N}, \mathcal{A})$: graphe orienté d'ensemble de nœuds $\mathcal{N}$ et d'ensemble d'arcs $\mathcal{A}$. Un arc $(i, j) \in \mathcal{A}$ peut être muni de diverses valuations : coût c_{ij}, distance d_{ij}, temps t_{ij}, capacité u_{ij}, etc.
- $\pi(x)$: potentiel logarithmique ($\pi(x) = -\sum_{1 \leqslant i \leqslant n} \log x_i$ dans le cas du cône positif de $\mathbb{R}^n$).
- $\mathcal{R}(A)$ ou $\mathrm{Im}\, A$: image de l'application linéaire $x \mapsto Ax$.
- $S(P)$: ensemble des solutions optimales d'un problème d'optimisation (P).
- $\mathcal{S}_n$: espace vectoriel des matrices symétriques réelles de taille $n \times n$.
- $\mathcal{S}_n^+$: cône des matrices symétriques réelles positives de taille $n \times n$.
- $\cdot^\top$: transposé d'un vecteur (par exemple : $x^\top$).
- $u(I^-, I^+)$: capacité de la coupe associée à une bi-partition en deux ensembles, I^- et I^+, des nœuds d'un graphe (chaque arc (i, j) est supposé muni d'une capacité u_{ij}).
- $\mathrm{val}(P)$: valeur d'un problème d'optimisation (P).
- $(\mathcal{V}, \mathcal{E})$: graphe non-orienté d'ensemble de sommets $\mathcal{V}$ et d'ensemble d'arêtes $\mathcal{E}$. Une arête $\{i, j\} \in \mathcal{A}$ peut être munie de diverses valuations. On notera par exemple $c_{\{i,j\}}$ ou c_{ij} le coût d'une arête $\{i, j\}$, étant entendu que $c_{ji} = c_{ij}$.

Index