Introduction au Compressed sensing.
Méthode du simplexe

Guillaume Lecué[1]

Résumé

Dans les deux chapitres qui se suivent, nous présentons deux types d'algorithmes pour résoudre des problèmes de programmation linéaires. Le premier, présenté dans ce chapitre, est l'algorithme du simplexe qui est un algorithme itératif de marche sur les sommets du polytope des contraintes. Le deuxième, présenté dans le chapitre suivant, est une méthode de barrière qui s'autorise à traverser le polytope de l'intérieur pour atteindre la solution.

1 Introduction

La procédure de Basis Pursuit peut être réécrite comme un problème de programmation linéaire : étant donnés $A \in \mathbb{R}^{m\times N}$ et $y \in \mathbb{R}^m$, on considère le problème d'optimisation

$$(\hat z^+, \hat z^-) \in \underset{(z^+,z^-)\in\mathbb{R}^{2N}}{\operatorname{argmin}} \sum_{j=1}^N z_j^+ + z_j^- \text{ tel que } [A| -A]\begin{bmatrix} z^+ \\ z^- \end{bmatrix} = y \text{ et } \begin{bmatrix} z^+ \\ z^- \end{bmatrix} \geq 0. \tag{LP}$$

On rappelle que pour tout $x \in \mathbb{R}^N$, on note $x^+ \in \mathbb{R}^N$ et $x^- \in \mathbb{R}^N$ les vecteurs dont les coordonnées sont données pour tout $j = 1, \dots, N$ par

$$(x^+)_j = \max(0, x_j) \text{ et } (x^-)_j = \max(0, -x_j).$$

L'équivalence entre le Basis Pursuit et (LP) est à comprendre de la manière suivante.

Proposition 1.1. *Il y a équivalence entre le problème d'optimisation du Basis Pursuit et (LP) :*

1. *Si $\hat x$ est solution du basis pursuit alors $(\hat x^+, \hat x^-)$ est solution de (LP)*
2. *Si $(\hat z^+, \hat z^-)$ est solution de (LP) alors $\hat z^+ - \hat z^-$ est solution du basis pursuit.*

L'objectif des deux chapitres suivants du cours est de proposer des algorithmes solutionnant le problème (LP). Pour cela, on introduit les formes générales et standard des problème de programmation linéaire. Ensuite, on présente deux types d'algorithmes pour solutionner les problèmes de programmation linéaire.

2 Programmation linéaire

Un problème de programmation linéaire est un problème d'optimisation sous contraintes où la fonction objectif est linéaire et les contraintes sont elles aussi linéaires. C'est donc un problème de la forme

$$\min_{x\in\mathbb{R}^p} c_1x_1 + \cdots + c_px_p \text{ tel que } \begin{cases} a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{ip}x_p \leq b_i, i = 1, \dots, m_1 \\ a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{ip}x_p = b_i, i = m_1 + 1, \dots, m_2. \end{cases} \tag{2.1}$$

1. CNRS, CREST, ENSAE. Bureau E31, 3 avenue Pierre Larousse. 92 245 Malakof. Email : guillaume.lecue@ensae.fr.

La fonction objectif

$$(x_1, \dots, x_n)^\top \in \mathbb{R}^n \longmapsto c_1 x_1 + \dots + c_n x_n$$

est bien linéaire. Il y a $m_1 + m_2$ contraintes linéaires dont m_1 sont des inégalités et m_2 sont des égalités.

Le problème 2.1 est la **forme générale** ou **forme canonique** des problèmes de programmation linéaire. Il est courant de réécrire ce problème sous la **forme standard** :

$$\boxed{\min_{x\in\mathbb{R}^n} \langle c, x\rangle \text{ tel que } Ax = b, x \geq 0.} \tag{2.2}$$

Proposition 2.1. *Il est toujours possible de transformer un problème de programmation linéaire écrit sous forme général en un problème de programmation linéaire sous forme standard.*

Démonstration. Étant donné un problème sous la forme générale (2.1). On considère deux transformations :

1. les contraintes d'inégalité "$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{ip}x_p \leq b_i$" peuvent être transformée en contraintes d'égalité par l'ajout d'une "**slack variable**" (variable molle) u_i par :

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{ip}x_p + u_i = b_i \text{ et } u_i \geq 0$$

2. on transforme toutes les variables x_i par

$$x^+ - x^- = x, x^+ \geq 0 \text{ et } x^- \geq 0.$$

On réécrit alors (2.1) sous la forme (2.2) où

$$x \to \begin{bmatrix} x^+ \\ x^- \\ u \end{bmatrix}, c \to \begin{bmatrix} c^+ \\ -c^- \end{bmatrix}, A \to \left[\begin{array}{c|c|c} A_1 & -A_1 & I_{m_1} \\ \hline A_2 & -A_2 & 0_{m_2} \end{array}\right] \text{ et } b \to b \tag{2.3}$$

où $A_1 = (a_{ij} : 1 \leq i \leq m_1, 1 \leq j \leq p)$ et $A_2 = (a_{ij} : m_1 + 1 \leq i \leq m_2, 1 \leq j \leq p)$

De la même manière que dans la preuve de Proposition 1.1, on vérifie l'équivalence des deux programmes :

1. Si $\hat{x}$ est solution de (2.1) alors il existe $\hat{u} \geq 0$ tel que $(\hat{x}^+, \hat{x}^-, \hat{u})$ est solution de (2.2) pour c, A, b définis dans (2.3).
2. Si $(\hat{x}^+, \hat{x}^-, \hat{u})$ est solution de (2.2) pour c, A, b définis dans (2.3) alors $\hat{x}^+ - \hat{x}^-$ est solution de (2.1).

■

Remarque 2.2. *En pratique, la transformation d'un problème de programmation linéaire sous forme canonique en un problème standard s'effectue par une suite d'opération du style :*

1. $x_1 + 2x_3 \leq 3 \longrightarrow x_1 + 2x_3 + s_1 = 3, s_1 \geq 0$
2. $-2x_1 + x_2 \geq 5 \longrightarrow -2x_1 + x_2 - s_2 = 5, s_2 \geq 0$ *et s_1, s_2 sont appelées les slack variables. Quand une slack variable est nulle la contrainte associée est dite **saturée** ; càd la solution est sur le bord de la contrainte.*
3. *"$x_1 \geq 5$" : ici on pourrait introduire une slack variable mais plutôt que d'augmenter inutilement le nombre de variables, on préférera poser $y_1 = x_1 - 5$ et remplacer $x_1 \to y_1 + 5$ pour $y_1 \geq 0$ partout où x_1 apparaît.*

4. Le variables qui ne sont pas contraintes à être positives sont remplacées par deux variables : $x_1 = x_1^+ - x_1^-$ *où* $x_1^+, x_1^- \geq 0$.

On s'intéresse dans la suite aux problèmes de programmation linéaire écrits sous forme standard. On remarque que le problème (LP) qu'on souhaite résoudre est directement écrit sous forme standard. Un problème de programmation linéaire se définit alors à partir de $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ et $b \in \mathbb{R}^m$.

3 Méthode du simplexe

On considère un problème de programmation linéaire sous forme standard

$$\boxed{\min_{x\in\mathbb{R}^n} \langle c, x\rangle \text{ tel que } Ax = b, x \geq 0}$$

où $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ et $b \in \mathbb{R}^m$.

L'**ensemble des contraintes** aussi appelé **ensemble de faisabilité** du problème est

$$\mathcal{C} = \{x \in \mathbb{R}^n : x \geq 0 \text{ et } Ax = b\}.$$

C'est l'intersection de l'octant supérieur $(\mathbb{R}_+)^n$ et de l'espace affine $\{x : Ax = b\}$. Soit cet ensemble est vide (en particulier, il n'y a pas de solution au problème d'optimisation), soit il n'est pas compact, soit c'est l'enveloppe convexe d'un nombre fini de points. Dans le troisième cas, $\mathcal{C}$ est, par définition, appelé **polytope** ou **simplexe**. Les points d'un polytope dont il est l'enveloppe convexe et qui ne peuvent pas être écrits comme combinaison convexe des autres points sont appelés **sommets** ou **points extrêmes**.

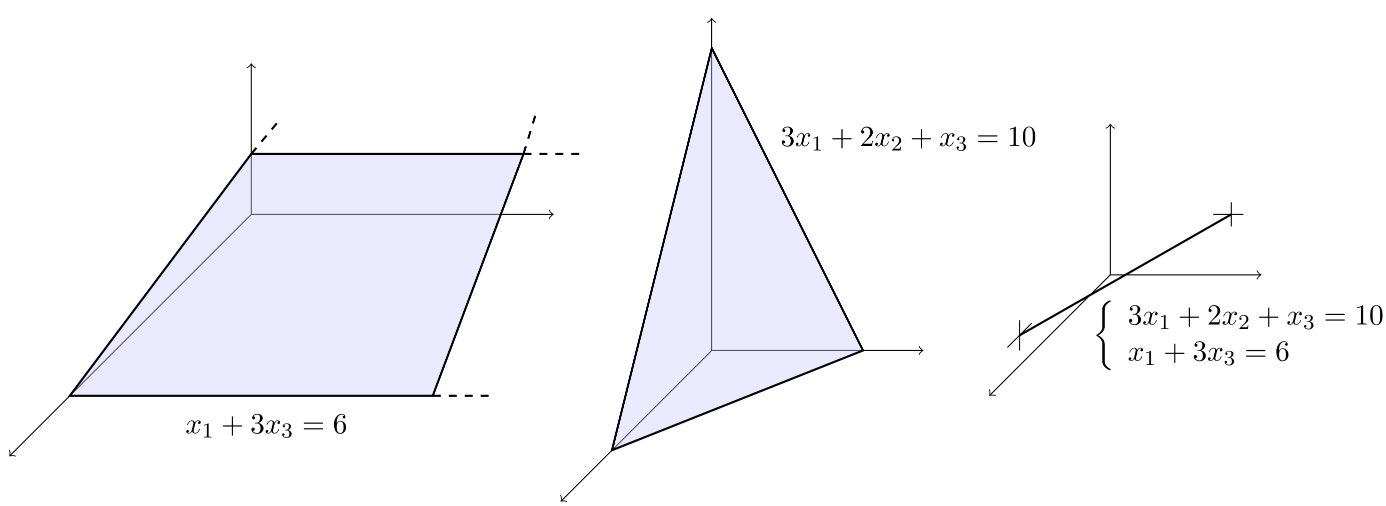


FIGURE 1 – Exemples d'espaces de contraintes non vides

3.1 Deux résultats préliminaires

Proposition 3.1. *Soit* $A \in \mathbb{R}^{m\times n}$ *et* $b \in \mathbb{R}^m$*. On note* $\mathcal{C} = (\mathbb{R}^n_+) \cap \{x \in \mathbb{R}^n : Ax = b\}$*. On suppose que* $\mathcal{C}$ *est non vide et compact. Dans ce cas, au moins un des sommets de l'ensemble des contraintes* $\mathcal{C}$ *est solution du problème (2.2).*

Démonstration. L'ensemble $\mathcal{C}$ des contraintes de (2.2) est un polytope, on peut donc écrire

$$\mathcal{C} = \mathrm{conv}(x_1, \dots, x_p) = \Big\{ \sum_{j=1}^{p} \lambda_j x_j : \lambda \in \Lambda^{(p)} \Big\}$$

où $x_1, \dots, x_p$ sont les points extrêmes de $\mathcal{C}$ et

$$\Lambda^{(p)} = \Big\{ \lambda = (\lambda_1, \dots, \lambda_p)^\top \in \mathbb{R}^p : \lambda_i \geq 0, i = 1, \dots, p \text{ et } \sum_{i=1}^{p} \lambda_i = 1 \Big\}.$$

La fonction objectif est continue. Elle atteint donc son minimum sur l'ensemble compact $\mathcal{C}$. En particulier, (2.2) admet au moins une solution. Soit $\hat{x} \in \mathcal{C}$ tel que $\big\langle c, \hat{x} \big\rangle \leq \big\langle c, x \big\rangle$ pour tout $x \in \mathcal{C}$. On a donc pour tout $\lambda \in \Lambda^{(p)}$,

$$\big\langle c, \hat{x} \big\rangle \leq \big\langle c, \sum \lambda_i x_i \big\rangle = \sum \lambda_i \big\langle c, x_i \big\rangle.$$

En prenant le minimum sur $\lambda \in \Lambda^{(p)}$ dans la dernière inégalité, on voit que $\big\langle c, \hat{x} \big\rangle \leq \min_{1 \leq i \leq p} \big\langle c, x_i \big\rangle$.

Par ailleurs, $\hat{x} \in \mathcal{C}$. Il existe alors $\hat{\lambda} \in \Lambda^{(p)}$ tel que $\hat{x} = \sum \hat{\lambda}_i x_i$. Ainsi, par linéarité,

$$\big\langle c, \hat{x} \big\rangle = \sum_i \hat{\lambda}_i \big\langle c, x_i \big\rangle \geq \min_{1 \leq i \leq p} \big\langle c, x_i \big\rangle.$$

On en déduit que $\min_{1 \leq i \leq p} \big\langle c, x_i \big\rangle = \big\langle c, \hat{x} \big\rangle$ et donc le problème (2.2) admet au moins une solution en un des points extrêmes de $\mathcal{C}$. ■

Une solution au problème (2.2) peut donc être trouvée en identifiant tous les points extrêmes de $\mathcal{C}$ et en testant leur optimalité les uns à la suite des autres. En général, cet algorithme n'est pas réalisable en grande dimension car le nombre de sommets de $\mathcal{C}$ peut être grand.

Néanmoins, le fait que (2.2) admette une solution en un des sommets de $\mathcal{C}$ est central pour l'algorithme du simplexe. En effet, la base de cet algorithme est de trouver un chemin parmi tous les sommets de $\mathcal{C}$ qui mène à une solution de (2.2).

Idée : L'algorithme du simplexe est une méthode itérative qui explore les sommets de $\mathcal{C} = (\mathbb{R}^n_+) \cap \{x \in \mathbb{R}^n : Ax = b\}$ de proche en proche et qui assure la décroissance de la fonction objectif $x \to \big\langle c, x \big\rangle$ à chaque itération. A la fin de l'algorithme du simplexe, un sommet de $\mathcal{C}$ solutionnant (2.2) est rendu.

A chaque étape de l'algorithme du simplexe, càd à un sommet de $\mathcal{C}$, la question centrale est de déterminer quel sommet voisin doit être exploré – idéalement, pour arriver à une solution de (2.2) (cette solution, rendue par l'algorithme du simplexe, est nécessairement un sommet) le plus rapidement possible. C'est précisément à cette question que répond (partiellement) l'algorithme du simplexe.

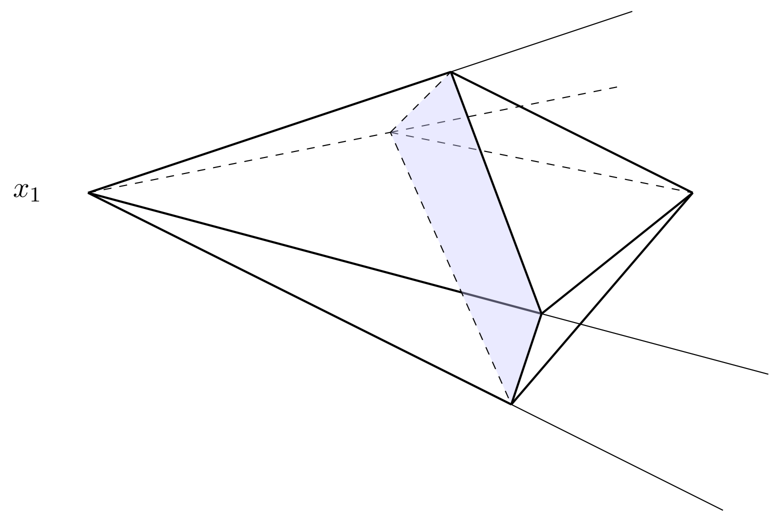
Remarque 3.2. *Il n'y a pas forcément unicité de la solution de (2.2). Si $\mathcal{C}$ est non vide et compact alors il peut très bien arriver qu'une face entière du polytope $\mathcal{C}$ soit solution. Quand $\mathcal{C}$ est non vide et non compact, il se peut que la solution soit à l'infini et donc le minimum de (2.2) est $-\infty$. On verra dans la suite comment l'algorithme du simplexe se comporte dans ces deux cas.*

La première chose qu'on doit s'assurer est qu'étant donné une étape de l'algorithme du simplexe, càd étant donné un sommet de $\mathcal{C}$, si la fonction objective $x \to \big\langle c, x \big\rangle$ n'est pas minimale en ce sommet alors nécessairement un des sommets voisins aura une valeur de la fonction objectif strictement plus petite.

Proposition 3.3. *Soit x_1 un point extrême de $\mathcal{C}$ (qu'on suppose non-vide et compact). Si x_1 n'est pas solution de (2.2) alors pour au moins un des sommets voisins de x_1 la fonction objectif $x \to \langle c, x \rangle$ prend une valeur strictement plus petite qu'en x_1.*

Démonstration. Pour cela, il suffit de montrer que si x_1 est un sommet de $\mathcal{C}$ tel que sur tous les sommets voisins la fonction $\varphi : x \to \langle c, x \rangle$ est plus grande qu'en x_1 alors φ est minimal en x_1.

On écrit $\mathcal{C} = \operatorname{conv}(x_1, \dots, x_p)$ où $x_1, \dots, x_p$ sont les sommets de $\mathcal{C}$. On définit l'ensemble $S_1 \subset \{x_1, \dots, x_p\}$ des sommets voisins de x_1 comme étant le plus petit ensemble de $\{x_1, \dots, x_p\}$ tel que le cône de sommet x_1 engendré par $\operatorname{conv}(S_1)$ contienne $\mathcal{C}$.



Par hypothèse, on a pour tout $x \in S_1$, $\varphi(x) \geq \varphi(x_1)$. On en déduit, par linéarité, de φ que sur toute demie-droite issue de x_1 et contenue dans le cône de sommet x_1 engendré par $\operatorname{conv}(S_1)$ la fonction φ est croissante (x_1 étant vu comme l'origine de la demie-droite). En particulier, si $x \in \{x_2, \dots, x_p\}$ alors le segment $[x_1, x]$ est un sous-ensemble d'une telle demie-droite. Donc φ croît sur ce segment de x_1 à x et donc $\varphi(x) \geq \varphi(x_1)$. Ceci étant vrai pour tout sommet x de $\mathcal{C}$, on en déduit le résultat à l'aide de Proposition 3.1. ■

Proposition 3.3 assure donc que le sommet exploré par l'algorithme du simplexe est soit solution du problème (2.2), soit il y a parmi ces sommets voisins un sommet pour lequel la fonction à minimiser sera strictement plus petite.

3.2 Présentation de l'algorithme du simplexe

L'algorithme du simplexe est un algorithme itératif d'exploration de sommets du polytope des contraintes $\mathcal{C} = (\mathbb{R}_+)^n \cap \{x : Ax = b\}$ qui assure la décroissance de la fonction objectif $\varphi : x \in \mathbb{R}^n \to \langle c, x \rangle$ le long de son chemin.

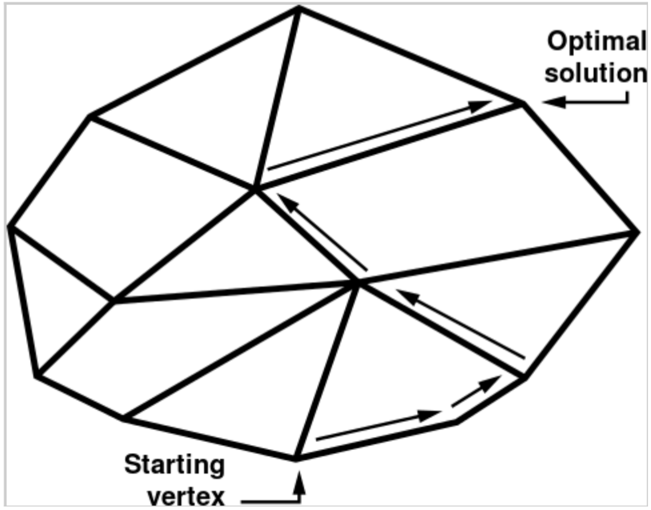


FIGURE 2 – Exploration des sommets du polytope des contraintes par l'algorithme du simplexe.

Il est basé sur deux règles :

1. la première règle est un **critère d'arrêt**
2. la deuxième règle énonce le principe pour **passer d'un sommet à un de ses sommets voisins** assurant la décroissance de φ.

Chaque itération s'effectue par une étape du **pivot de Gauss** (aussi appelé *élimination Gaussienne* ou *méthode de Gauss-Jordan*). En effet, en introduisant la variables $z = \langle c, x \rangle$, on voit qu'une solution de (2.2) sera aussi solution du système d'équations linéaires

$$\begin{cases} z - \langle c, x \rangle & = 0 \\ Ax & = b. \end{cases} \tag{3.1}$$

On va donc à chaque étape trouver une solution positive au système (3.1) par la méthode du pivot de Gauss.

A l'étape suivante, on sera amené à résoudre un autre système d'équations linéaires semblable à celui de (3.1) mais pour d'autres coefficients $c \to c', A \to A', b \to b'$ obtenus par une étape de pivot de Gauss. Ce système est de la forme

$$\begin{cases} z - \langle c', x \rangle & = b'_0 \\ A'x & = b'. \end{cases} \tag{3.2}$$

Ce système apparaît à chaque itération de l'algorithme du simplexe. Il est utile de le représenter sous forme de tableau.

z	$x^\top$	b	
1	$-(c')^\top$	b'_0	ligne 0
0	A'	b'	lignes d'indices $1, \dots, m$

nom des variables

FIGURE 3 – Représentation sous forme de tableau de chaque itération de l'algorithme du simplexe

Si, pour une certaine itération, le vecteur c' n'a que des coefficients positifs ou nuls alors, comme $x \geq 0$, on ne pourra plus faire décroître z et alors la solution au problème sera $z = b'_0$. Ceci correspond au critère d'arrêt de l'algorithme du simplexe et donc à la première règle.

Première règle : *Si toutes les coordonnées du vecteur $-(c')^\top$ de la ligne 0 sont négatifs ou nuls alors une solution à coordonnées positives ou nulles au système (3.2) minimisant z est solution au problème. Dans ce cas, l'algorithme du simplexe s'arrête et renvoie une telle solution. Sinon, une variable associée à un coefficient strictement positif de $-(c')^\top$ est choisie comme nouvelle variable à pivoter.*

L'algorithme du simplexe va donc dérouler ses itérations jusqu'à obtenir la première ligne du tableau n'ayant que des coefficients négatifs ou nuls (sauf pour b'_0).

Comme annoncé précédemment, les itérations de l'algorithme du simplexe sont obtenues par résolution d'un système d'équations linéaires par la méthode du pivot de Gauss. Le choix de la ligne à pivoter défini la deuxième règle. Nous la présentons maintenant sur un exemple.

Exemple 1 : On souhaite résoudre le problème de programmation linéaire suivant :

$$\boxed{\min \left(-x_1 - x_2 : \begin{array}{ll} 2x_1 + x_2 & \leq 4 \\ x_1 + 2x_2 & \leq 3 \\ x_1, x_2 \geq 0 & \end{array} \right)} \tag{Ex1}$$

La première étape consiste à écrire ce problème sous la forme standard. Ici il suffit d'introduire deux slack variables : x_3 et x_4. Le problème est équivalent au problème suivant

$$\min \left(-x_1 - x_2 : \begin{array}{ll} 2x_1 + x_2 + x_3 & = 4 \\ x_1 + 2x_2 + x_4 & = 3 \\ x_1, x_2, x_3, x_4 \geq 0 & \end{array} \right)$$

On écrit le problème standard sous forme de tableau :

z	x_1	x_2	x_3	x_4	b
1	1	1	0	0	0
0	2	1	1	0	4
0	1	2	0	1	3

Une solution positive minimisant z à ce système d'équation est donnée par

$$x_1 = 0, \quad x_2 = 0, \quad x_3 = 4 \text{ et } x_4 = 3$$

pour laquelle on obtient une valeur de la fonction objective $z = 0$. Les solutions positives minimisant z aux tableaux de l'algorithme sont appelées **solution de base faisable (SBF)**. Cette solution est-elle solution du problème (Ex1) ? Non, car il y a des coefficients strictement positifs (associés aux x_i) dans la *ligne* 0 et donc d'après la *Première règle*, on peut encore faire décroître z.

L'étape suivante se décompose en deux phases :

1. choix d'une variable à pivoter : on prend n'importe quelle variable x_i associée à un coefficient strictement positif dans la ligne 0. Par exemple, ici, on choisi x_1 comme variable pivot. Cette variable est appelée **variable entrante**.
2. choix d'une ligne à pivoter : on peut faire pivoter x_1 soit dans la ligne 1 soit dans la ligne 2. Ici, dans l'exemple introductif, on va essayer les deux possibilités pour en déduire une règle (la deuxième règle de la méthode de simplexe) sur le choix de la ligne à pivoter.

Première possibilité : faire pivoter x_1 dans la ligne 1. On applique la méthode du pivot de Gauss sur la variable x_1 dans la première ligne. On obtient alors le tableau suivant :

z	x_1	x_2	x_3	x_4	b
1	0	1/2	−1/2	0	−2
0	1	1/2	1/2	0	2
0	0	3/2	−1/2	1	1

Ce tableau a pour solution de base faisable (SBF)

$$x_2 = 0, \quad x_3 = 0, \quad x_1 = 2 \text{ et } x_4 = 1.$$

Idée : Les SBF sont trouvées rapidement, en prenant $x_i = 0$ quand x_i est associé à un coefficient non nul dans la ligne 0. Ensuite, les valeurs des autres variables sont obtenues grâce aux équations des lignes suivantes $A'x = b'$.

La SBF est-elle une solution au problème (Ex1) ? Non, car il reste encore un coefficient strictement positif dans la ligne 0 (celui de x_2). On relance une itération de l'algorithme. Mais avant ça, on regarde ce qui se passe si on choisit la deuxième ligne comme ligne à pivoter pour x_1.

Deuxième possibilité : faire pivoter x_1 dans la ligne 2. On applique la méthode du pivot de Gauss sur la variable x_1 dans la deuxième ligne. On obtient alors le tableau suivant :

z	x_1	x_2	x_3	x_4	b
1	0	−1	0	−1	−3
0	0	−3	1	−2	−2
0	1	2	0	1	3

Ce tableau a pour solution de base faisable (SBF)

$$x_2 = 0, \quad x_4 = 0, \quad x_3 = -2 \text{ et } x_1 = 3.$$

Cette solution n'est pas faisable : elle n'est pas dans l'ensemble de contrainte imposant $x \geq 0$ (ici $x_3 = -2$). Donc, faire pivoter x_1 dans la deuxième ligne n'est pas la bonne stratégie. On va donc choisir de faire pivoter x_1 dans la première ligne.

Comment pourrions-nous savoir quelle ligne choisir pour faire pivoter la variable entrante avant même d'appliquer la méthode du pivot de Gauss ? Pour cela, on va faire en sorte de ne pas introduire de coefficients négatifs dans le vecteur b' de la dernière colonne du tableau grâce au **test du ratio minimum**. Ce qui constitue la deuxième règle de l'algorithme du simplexe.

Deuxième règle : *Étant donné une variable entrante x_i. On choisi de faire pivoter x_i dans la ligne $\ell \in \{1, \dots, m\}$ du tableau pour l'indice ℓ ayant le plus petit ratio :*

$$\frac{b'_\ell}{a'_{\ell i}} = \frac{\textit{valeur du coefficient de droite}}{\textit{coefficient de la variable entrante de la ligne}}$$

quand le "coefficient de la variable entrante de la ligne" est strictement positif.

Pour le problème (Ex1), ce ratio est

$$\text{pour la ligne 1 : } ratio = \frac{4}{2} \qquad \text{pour la ligne 2 : } ratio = \frac{3}{1}.$$

La ligne qui a donc le plus petit ratio est la ligne 1, c'est donc elle qui est choisie pour faire pivoter la variable entrante x_1.

On reprend l'exemple (Ex1) après avoir fait pivoter x_1 dans la ligne 1. On sait d'après la première règle que la SBF obtenue n'est pas solution de (Ex1) car il y a un coefficient non nul dans la ligne 0. La dernière variable à avoir un coefficient non nul dans la ligne 0 est x_2. C'est cette variable qui est choisie comme variable à pivoter.

Dans quelle ligne faire pivoter la variable entrante x_2 ? Pour cela, on applique la deuxième règle en calculant le ratio pour chaque ligne :

$$\text{pour la ligne 1 : } ratio = \frac{2}{1/2} \qquad \text{pour la ligne 2 : } ratio = \frac{1}{3/2}.$$

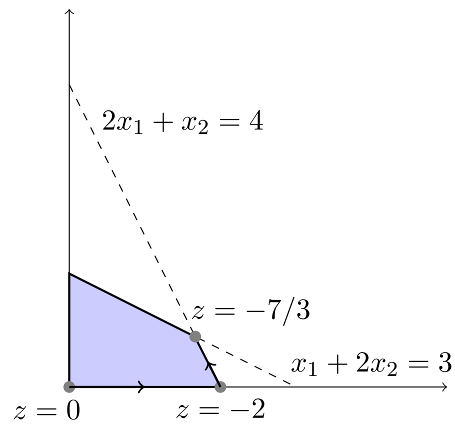
C'est la deuxième ligne qui a le plus petit ratio. C'est donc cette ligne qui est choisie pour faire pivoter x_2. On obtient finalement le tableau suivant pour l'algorithme du simplexe :

z	x_1	x_2	x_3	x_4	b	
1	1	1	0	0	0	pivot variable : x_1
0	2	1	1	0	4	pivot ligne : 1
0	1	2	0	1	3	(ratio test : ligne 1 $\to 4/2$, ligne 2 $\to 3/1$)
1	0	$1/2$	$-1/2$	0	-2	
0	1	$1/2$	$1/2$	0	2	pivot variable : x_2
0	0	$3/2$	$-1/2$	1	1	pivot ligne : 2 (ratio test : ligne 2 $\to 2/(1/2)$, ligne 2 $\to 1/(3/2)$)
1	0	0	$-1/3$	$-1/3$	$-7/3$	
0	1	0	$2/3$	$-1/3$	$5/3$	les coefficients des x_i sont tous négatifs ou nuls
0	0	1	$-1/3$	$2/3$	$2/3$	SBF : $x_3 = x_4 = 0, x_1 = 5/3, x_2 = 2/3$

On s'arrête à la deuxième itération car tous les coefficients des x_i sont négatifs dans la ligne 0. La SBF de cette dernière étape est solution du problème (Ex1) : une solution à (Ex1) est donc $z = -7/3$ atteinte pour $x_1 = 5/3$ et $x_2 = 2/3$. On remarque aussi que les slack variables valent $x_3 = x_4 = 0$. Ce qui signifie que les deux conditions associées sont saturées : càd la solution est atteinte au sommet du polytope des contraintes (du problème (Ex1) et non de sa forme standard) solution du système

$$\begin{cases} 2x_1 + x_2 & = 4 \\ x_1 + 2x_2 & = 3. \end{cases}$$

Le sommet de l'ensemble des contraintes du problème standard associé à (Ex1) où est atteint la solution du problème est $(5/3, 2/3, 0, 0)$. On ne peut pas représenter ce polytope qui vit en dimension 4 néanmoins on peut représenter celui du problème original (Ex1) (qui est sous forme non standard) et pour lequel l'espace des contraintes est un polytope de $\mathbb{R}^2$ (pour lequel les propositions 3.3 et 3.1 sont aussi valides).



Idée : Le choix des variables pivots s'effectue parmi toutes les variables affectées d'un coefficient strictement positif dans la ligne 0*. En général, ce choix n'est pas très important (on pourra vérifier que si* x_2 *est choisi en premier pivot dans l'exemple précédent, l'algorithme prendra un autre chemin sur les sommets mais le nombre d'itérations reste le même ici). Cependant, dans certains cas, bien choisir son pivot peut éviter des phénomènes de cycle. Pour cela certaines règles ont été proposées ; comme par exemple, la règle de Bland qui consiste à choisir la variable entrante d'indice minimal.*

Définition 3.4. *Les variables qui apparaissent dans une seule équation, càd dont la colonne associèe est une colonne de la matrice identité sont appelées* ***variables de base****. Les autres variables sont appelées* ***variables non-basique****.*

En particulier, à chaque étape, la variable entrante (càd la variable qui va être pivotée) est une variable non-basique qui sera transformée en variable basique par le pivot de Gauss. Au cours de ce processus, il y a, en générale, une variable de base qui sera transformée en variable non-basique ; cette variable est appelée **variable sortante**.

Exemple 2 (solutions multiples) : Dans l'exemple précédent, il n'y avait qu'un unique sommet solution. Cependant dans certain cas, il se peut qu'une face entière du polytope des contraintes soit solution. Dans ce cas, l'algorithme du simplexe renvoie bien une solution mais choisir différentes variables pivot lors des itérations peut retourner différentes solutions.

Pour illustrer ce phénomène, on reprend l'exemple précédent en modifiant légèrement la fonction objectif tout en gardant le même espace des contraintes.

On souhaite résoudre le problème de programmation linéaire suivant :

$$\boxed{\min \left(-x_1 - \frac{x_2}{2} : \begin{array}{cc} 2x_1 + x_2 & \leq 4 \\ x_1 + 2x_2 & \leq 3 \\ x_1, x_2 \geq 0 & \end{array} \right)} \tag{Ex2}$$

qu'on réécrit sous forme standard directement au format d'un tableau : On écrit le problème standard sous forme de tableau.

z	x_1	x_2	x_3	x_4	b
1	1	1/2	0	0	0
0	2	1	1	0	4
0	1	2	0	1	3

On choisit x_1 comme variable pivot et la ligne 1 comme ligne pivot (test du ratio donne ligne 1 : 4/2 et ligne 2 : 3/2). On obtient le tableau suivant :

z	x_1	x_2	x_3	x_4	b
1	0	0	$-1/2$	0	-2
0	1	1/2	1/2	0	2
0	0	3/2	$-1/2$	1	1

Tous les coefficients associés aux x_i de la ligne 0 sont négatifs ou nuls alors, d'après la première règle, l'algorithme s'arrête et une SBF fournit une solution au problème. Ici $z = -2$ est atteint pour $x_3 = 0$ et pour x_1, x_2, x_4 vérifiant le système d'équations :

$$\left\{ \begin{array}{l} x_1 + (1/2)x_2 = 2 \\ (3/2)x_2 + x_4 = 1. \end{array} \right.$$

En fait la valeur $z = -2$ est atteinte en tout points de

$$\left\{ \left(\frac{5}{3} + \frac{x_4}{3}, \frac{2}{3} - \frac{2x_4}{3} \right) \in (\mathbb{R}_+)^2 : x_4 \geq 0 \right\}$$

qui décrit une face du polytope des contraintes de (Ex2).

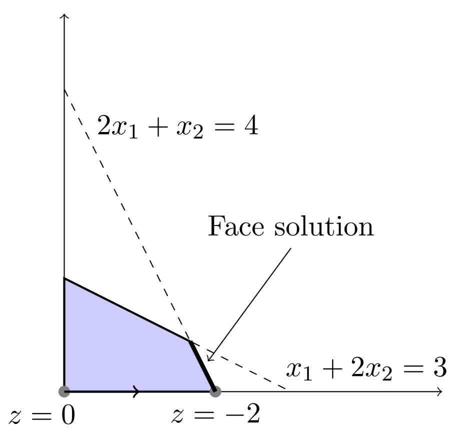


FIGURE 4 – Algorithme du simplexe pour un problème avec solutions multiples.

Exemple 3 (variable dégénérée) : Dans certain cas, une variable de base vaut 0 pour la SBF. Dans ce cas, la fonction objectif stagne. En général, cela ne fait que retarder l'algorithme du simplexe. Cependant, dans certain cas, un phénomène de cycle se produit ; càd l'algorithme alterne entre deux ou plus tableau identique sans jamais sortir de cette boucle. On peut éviter ces cycles par exemple en utilisant la règle de Bland.

L'apparition d'une variable dégénérée peut être due à la présence d'une contrainte redondante càd qu'elle peut être retirée de la liste des contraintes sans changer le problème.

Pour illustrer ce phénomène, on considère le problème de programmation linéaire suivant :

$$\boxed{\min \left(-2x_1 - x_2 : \begin{array}{ll} 3x_1 + x_2 & \leq 6 \\ x_1 - x_2 & \leq 2 \\ x_2 \leq 3 & \\ x_1, x_2 \geq 0 & \end{array} \right)} \tag{Ex3}$$

On écrit ce problème sous forme standard et on lance l'algorithme du simplexe (sous forme de tableaux) :

z	x_1	x_2	x_3	x_4	x_5	b
1	2	1	0	0	0	0
0	3	1	1	0	0	6
0	1	-1	0	1	0	2
0	0	1	0	0	1	3
1	0	3	0	-2	0	-4
0	0	4	1	-3	0	0
0	1	-1	0	1	0	2
0	0	1	0	0	1	3
1	0	0	$-3/4$	$1/4$	0	-4
0	0	$\boxed{1}$	$1/4$	$-3/4$	0	$\boxed{0}$
0	1	0	$1/4$	$1/4$	0	2
0	0	0	$-1/4$	$3/4$	1	3
1	0	0	$-2/3$	0	$-1/3$	-5
0	0	1	0	0	1	3
0	1	0	$1/3$	0	$-1/3$	1
0	0	0	$-1/3$	1	$4/3$	4

pivot variable : x_1
pivot ligne : 2
(ratio test : ligne 1 $\rightarrow 6/3$, ligne 2 $\rightarrow 2/1$)

pivot variable : x_2
pivot ligne : 1
(ratio test : ligne 1 $\rightarrow 0/4$, ligne 3 $\rightarrow 3/1$)

pivot variable : x_4
pivot ligne : 3
(ratio test : ligne 2 $\rightarrow 2/(1/4)$, ligne 3 $\rightarrow 3/(3/4)$)

les coefficients des x_i sont tous négatifs ou nuls dans la ligne 0
SBF : $x_3 = x_5 = 0$, $x_2 = 3$, $x_1 = 1$, $x_4 = 4$

On obtient donc la solution $z = -5$ atteinte en $x_1 = 1$, $x_2 = 3$. Il y a deux slack variables égales à zéro : $x_3 = x_5 = 0$ donc les contraintes associées sont saturées en la solution : $3x_1 + x_2 = 6$ et $x_2 = 3$.

En détaillant les SBF à chaque étape, on se rend compte que la fonction objectif a stagné en $z = -4$ entre l'étape 1 et 2 :

1. Etape 0 : SBF $x_1 = 0, x_2 = 0, x_3 = 6, x_4 = 2, x_5 = 3$ et $z = 0$
2. Etape 1 : SBF $x_1 = 2, x_2 = 0, x_3 = 0, x_4 = 0, x_5 = 3$ et $z = -4$
3. Etape 2 : SBF $x_1 = 2, x_2 = 0, x_3 = 0, x_4 = 0, x_5 = 3$ et $z = -4$
4. Etape 3 : SBF $x_1 = 1, x_2 = 3, x_3 = 0, x_4 = 4, x_5 = 0$ et $z = -5$.

Cela est due à la présence d'une variable dégénérée : $x_2 = 0$ alors que x_2 est une variable de base. Cela n'a fait que retarder l'algorithme sans l'empêcher de converger vers la solution. Dans ce cas, la dégénérescence de x_2 est due à une contrainte redondante : "$x_1 - x_2 \leq 3$" est impliquée par les autres contraintes.

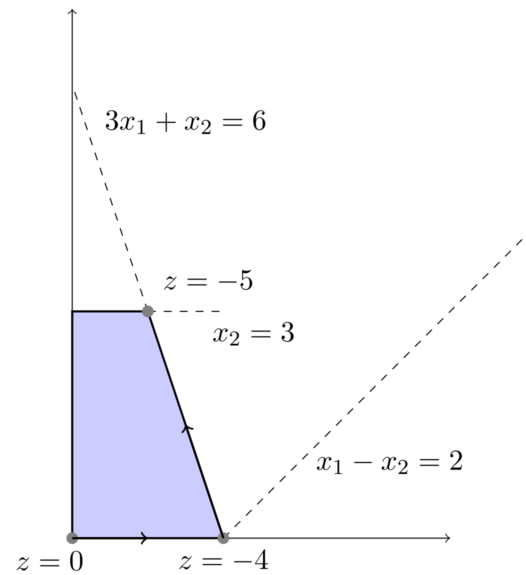


FIGURE 5 – Algorithme du simplexe pour un problème avec une variable dégénérée.

Dans cet exemple, la contrainte "$x_1 - x_2 \leq 2$" est dégénérée. Elle ne fait que ralentir l'algorithme du simplexe dans notre cas. Dans d'autres cas, elle peut être responsable d'un comportement cyclique.

Idée : L'algorithme du simplexe n'assure pas une décroissance stricte de la fonction objectif mais seulement sa décroissance. Il se peut alors que si une variable est dégénérée ou si la fonction objectif est constante sur une face (non réduite à un point) de l'ensemble des contraintes alors l'algorithme va s'enfermer dans un cycle infini. Il existe plusieurs tactiques pour éviter ces phénomènes ; la règle de Bland en est une.

Exemple 4 (solution non bornée) : Dans certain cas, l'ensemble des contraintes est non borné et la fonction objectif peut alors décroître jusqu'à l'infini. Ces situations sont très facilement identifiable par l'algorithme du simplexe.

Pour illustrer ce phénomène, on considère le problème de programmation linéaire suivant :

$$\boxed{\min \left(-2x_1 - x_2 : \begin{array}{ll} -x_1 + x_2 & \leq 1 \\ x_1 - 2x_2 & \leq 2 \\ x_1, x_2 \geq 0 & \end{array} \right)} \tag{Ex4}$$

On écrit ce problème sous forme standard et on lance l'algorithme du simplexe (sous forme de tableaux) :

z	x_1	x_2	x_3	x_4	b
1	2	1	0	0	0
0	-1	1	1	0	1
0	1	-2	0	1	2
1	0	5	0	-2	-4
0	0	$\boxed{-1}$	1	1	3
0	1	$\boxed{-2}$	0	1	2

FIGURE 6 – Comportement de l'algorithme du simplexe pour une solution non bornée

A la première étape, x_1 a été choisi comme variable pivot et la ligne pivot est la deuxième. A l'étape suivante, la seule variable pivot à choisir est x_2 (c'est la seule avec un coefficient positif à la ligne 0). La deuxième règle nous dit alors de choisir la ligne selon le test du ratio. Mais ici *tous les coefficients de x_2 sont négatifs* dans les lignes 1 et 2. Il n'y a donc pas de ratio à calculer. L'idée est que pour ce tableau on peut faire croître x_2 à l'infini tout en restant dans l'ensemble des contraintes et avoir une fonction objective qui tend vers $-\infty$. On est dans le cas d'une solution non-bornée et l'algorithme du simplexe le détecte quand le seul pivot possible est associé à des coefficients tous négatifs dans la colonne du pivot autres que la ligne 0.

En regardant l'ensemble des contraintes pour ce problème, on voit en effet qu'il n'est pas borné.

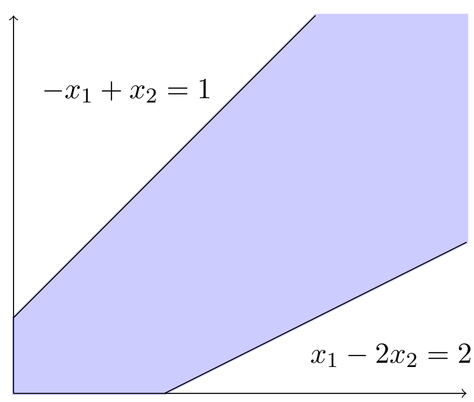


FIGURE 7 – Algorithme du simplexe pour un problème à ensemble de contraintes non bornée.

<u>*Idée*</u> *: Il y a trois situations possibles :*

1. *l'ensemble des contraintes est vide, alors il n'y a pas de SBF.*
2. *l'ensemble des contraintes est non borné et il existe une solution non bornée. Alors la fonction objectif tend vers $-\infty$.*
3. *Il y a une solution finie et dans ce cas soit elle est unique, soit il y a une face entière du polytope des contraintes qui est solution.*

Dans tous les cas, l'algorithme du simplexe identifie la situation et renvoie une solution au problème. Les seuls cas problématiques sont dus aux cycles et dans certains cas pathologiques le nombre d'itération peut être très grand. Mais en pratique, l'algorithme du simplexe est très performant.

3.3 Deux autres exemples pour s'entraîner

Exemple 5 : Résoudre le problème de programmation linéaire suivant par la méthode du simplexe sous forme de tableaux :

$$\boxed{\min\left(-4x_1 - x_2 + x_3 : \begin{array}{rl} x_1 + 3x_3 & \leq 6 \\ 3x_1 + x_2 + 3x_3 & \leq 9 \\ x_1, x_2, x_3 \geq 0 & \end{array}\right)}$$

z	x_1	x_2	x_3	x_4	x_5	b
1	4	1	1	0	0	0
0	1	0	3	1	0	6
0	3	1	3	0	1	9
1	0	$-1/3$	-3	0	$-4/3$	-12
0	0	$-1/3$	2	1	$-1/3$	3
0	1	$1/3$	1	0	$1/3$	3

Donc la solution est $z = -12$ atteinte en $x_1 = 3$, $x_2 = 0$ et $x_3 = 0$.

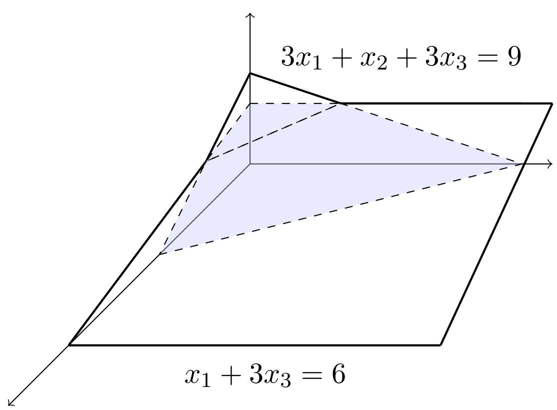


Figure 8 – Simplexe des contraintes du problème sous forme non-standard.

Exemple 6 : Résoudre le problème de programmation linéaire suivant par la méthode du simplexe sous forme de tableaux :

$$\boxed{\min\left(-2x_1 - 3x_2 - 4x_3 : \begin{array}{rl} 3x_1 + 2x_2 + x_3 & \leq 10 \\ 2x_1 + 5x_2 + 3x_3 & \leq 15 \\ x_1, x_2, x_3 \geq 0 & \end{array}\right)}$$

z	x_1	x_2	x_3	x_4	x_5	b
1	2	3	4	0	0	0
0	3	2	1	1	0	10
0	2	5	3	0	1	15
1	$-2/3$	$-11/3$	0	0	$-4/3$	-20
0	$7/3$	$1/3$	0	1	$-1/3$	5
0	$2/3$	$5/3$	1	0	$1/3$	5

Donc la solution est $z = -20$ atteinte en $x_1 = 0$, $x_2 = 0$ et $x_3 = 5$.

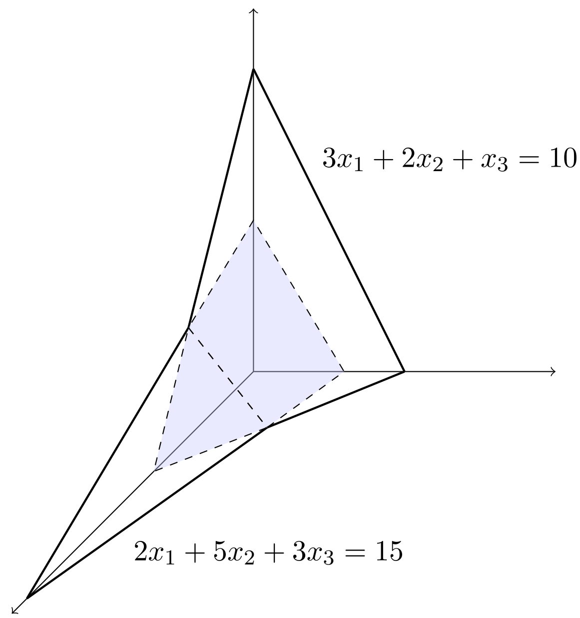


Figure 9 – Simplexe des contraintes du problème sous forme non-standard.