Université Sorbonne Paris Nord
Institut Galilée - Ingénieur 2ème année

Optimisation Linéaire (OL) -G4SIOL-

Cours 3 - Algorithme du simplexe (2)

pierre.fouilhoux@lipn.fr

4 octobre 2020

Questions

L'algorithme du simplexe Phase II présenté précédemment consiste à passer de base en base jusqu'à trouver une base correspondant à une solution optimale.

Plusieurs questions se posent alors :

- La **validité** de l'algorithme :
  - est-ce qu'à chaque itération, il est possible d'exprimer la base en fonction des variables hors-base ?
  - est-ce que ces bases sont à chaque fois associées à des solutions réalisables ?
- La **terminaison** de l'algorithme :
  - est-ce que l'algorithme du simplexe se terminer en un nombre fini d'itérations par une solution optimale ou la détection d'un PL non-borné ?
  - comment détecter les PL de domaine de définition vide ?
- Comment **initialiser** la phase II de l'algorithme par une première base si celle des variables d'écart n'est pas réalisable ?
- Quelle est la **complexité** de l'algorithme du simplexe ?
- Et comment prouver formellement tout cela ?

Visualisation graphique

Complexité

Notations matricielles et écriture en base

Théorèmes d'optimalité et validé du pivot

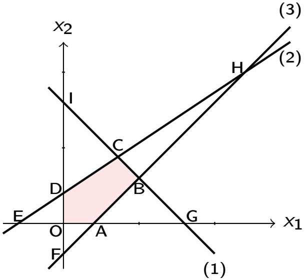
Dégénérescence et cyclage

Terminaison et validité de l'algorithme du simplexe

Visualisation graphique de l'algorithme du simplexe

Considèrons à nouveau le PL

$$\begin{cases} \max\ z = & 6x_1 & + & 5x_2 & & & \\ & x_1 & + & x_2 & \leq & 8 & (1) \\ & -2x_1 & + & 3x_2 & \leq & 6 & (2) \\ & x_1 & - & x_2 & \leq & 2 & (3) \\ & x_1 & & & \geq & 0 & (5) \\ & & & x_2 & \geq & 0 & (4) \end{cases}$$



Sa forme standard est :

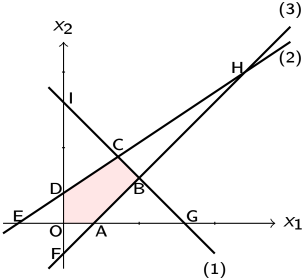
$$\begin{cases} \max\ z = & 6x_1 & + & 5x_2 & & & & \\ & x_1 & + & x_2 & +e_1 & = & 8 & (1) \\ & -2x_1 & + & 3x_2 & +e_2 & = & 6 & (2) \\ & x_1 & - & x_2 & +e_3 & = & 2 & (3) \\ & x_1 & & & & \geq & 0 & (5) \\ & & & x_2 & & \geq & 0 & (4) \\ & & & & e_i & \geq & 0 & i \in \{1,2,3\} \end{cases}$$

Etre à l'intersection de 2 droites parmi (1),...,(5) revient à annuler 2 variables parmis x_1, x_2, x_3, e_1, e_2, e_3 : c'est-à-dire à choisir une base !

Visualisation graphique de l'algorithme du simplexe

Pour la base des variables d'écart $B = \{e_1, e_2, e_3\}$, on a le dictionnaire :

```
e1 =  +   8   -   1   x1 -   1   x2
e2 =  +   6   +   2   x1 -   3   x2
e3 =  +   2   -   1   x1 +   1   x2
 z =  +   0   +   6   x1 +   5   x2
```



Cette base correspond à la solution où les variables hors-base x_1, x_2 et x_3 sont nulles : c'est le point O (0,0)
qui correspond dans la forme standard au vecteur solution $s_0 = (0, 0, 8, 6, 2)$
de valeur $z = 0$.

Il n'est pas optimal
(les coûts dans z des variables d'écarts ne sont pas tous négatifs ou nuls).

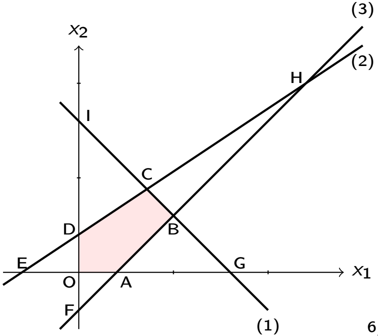
Visualisation graphique de l'algorithme du simplexe

Première itération :
On choisit comme variable entrante x_1 (critère de Dantzig car 6>5).

On maintient $x_2 = 0$ et on augmente x_1 au maximum possible.
$e_1 \geq 0 \Leftrightarrow x_1 \leq 8$
$e_2 \geq 0 \Leftrightarrow x_1 \geq -3$
$e_3 \geq 0 \Leftrightarrow x_1 \leq 2$
Le plus contraignant est e_3 qui va sortir de base.

Dans le dessin, augmenter x_1 en maintenant $x_2 = 0$: c'est se déplacer sur l'axe des abscisses à partir du point O(0,0).

En faisant cela :
- on reste du "bon côté" de la droite (2), c'est-à-dire que e_2 va rester positif (non contraint).
- on se rapproche de (1) et (3) :
cela fait diminuer e_1 et e_3.
- la première droite rencontrée est (3) :
e_3 s'annule avant e_1.



Visualisation graphique de l'algorithme du simplexe

La nouvelle base est $B = \{x_1, e_1, e_2\}$
qui correspond à l'annulation de x_2 et de e_3, c'est-à-dire le point d'intersection de (3) et (4) : le point A(2,0).

On écrit le nouveau dictionnaire

```
e1 =  +    6     -     2     x2 +     1     e3
e2 =  +   10     -     1     x2 -     2     e3
x1 =  +    2     +     1     x2 -     1     e3
 z =  +   12     +    11     x2 -     6     e3
```

La solution est $s_1 = (2, 0, 6, 10, 0)$ de valeur $z = 12$

qui n'est pas optimale car le coefficient de x_2 n'est pas négatif ou nul.

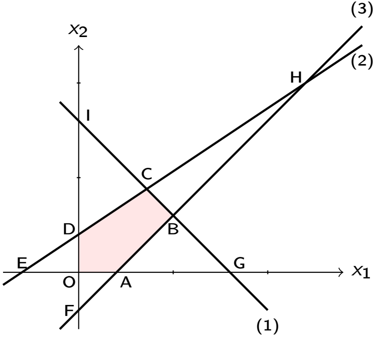
Visualisation graphique de l'algorithme du simplexe

Deuxième itération :
La seule variable entrante possible est x_2
On maintient $e_3 = 0$ et on augmente x_2 au maximum possible.
$e_1 \geq 0 \Leftrightarrow x_2 \leq 3$
$e_2 \geq 0 \Leftrightarrow x_2 \leq 10$
$x_1 \geq 0 \Leftrightarrow x_2 \geq -2$
Le plus contraignant est e_1 qui va sortir de base.

Dans le dessin, augmenter x_2 en maintenant $e_3 = 0$: c'est se déplacer sur la droite (3) à partir du point point A(2,0).

En faisant cela :
- on reste du "bon côté" de la droite (4), c'est-à-dire que x_1 va rester positif (non contraint).
- on se rapproche de (1) et (2) :
cela fait diminuer e_1 et e_2.
- la première droite rencontrée est (1) :
e_1 s'annule avant e_2.

On atteint le point B(5,3).



Visualisation graphique de l'algorithme du simplexe

La nouvelle base est $B = \{x_1, x_2, e_2\}$
qui correspond à l'annulation de e_1 et de e_3, c'est-à-dire le point d'intersection de (1) et (3) : le point B(5,3).

On écrit le nouveau dictionnaire

```
x2 =  +    3    -  1/  2  e1 +  1/  2  e3
e2 =  +    7    +  1/  2  e1 -  5/  2  e3
x1 =  +    5    -  1/  2  e1 -  1/  2  e3
 z =  +   45    - 11/  2  e1 -  1/  2  e3
```

La solution est $s_2 = (5, 3, 6, 0, 0)$ de valeur $z = 45$

qui est optimale car tous les coefficients dans z sont négatifs ou nuls.

Complexité

Plaçons-nous sous les 3 hypothèse suivantes :
- l'algorithme est valide
- l'algorihme se termine (pas de cyclage)
- on connaît une base initiale pour démarrer la phase II

quelle est la complexité de la phase II ?

La vision géométrique précédente témoigne du fait qu'une base est associée à un point extrême du polyèdre de définitions.
L'algorithme du simplexe consiste en fait à passer de base en base réalisable, c'est-à-à-dire d'un point extrême à un autre.

On a vu qu'un point extrême est l'intersection de n inégalités parmi $n + m$: ce qui est un nombre largement exponentiel de points :
contrairement à l'algorithme d'énumération, l'algorithme du simplexe ne considère que les intersections **à l'intérieur** du polyèdre de définitions.

Mais combien peut-il y avoir d'intersections dans un polyèdre de définitions d'un PL ?

Pire-cas

Klee et Minty ont proposé le PL suivant :

$$\begin{array}{lll} \text{Max} & \sum_{j=1}^{n} 10^{n-j} x_j & \\ & \left(2 \sum_{j=1}^{i-1} x_j\right) + x_i \leq 100^{i-1} & \forall i \in \{1, \dots, n\} \\ & x_j \geq 0 & \forall j \in \{1, \dots, n\} \end{array}$$

Ce PL correspond a $2^n - 1$ points extrêmes : tous explorables l'un après l'autre par l'algorithme du simplexe. Ce qui prouve qu'il peut falloir au moins $2^n - 1$ itérations de l'algorithme du simplexe pour arriver à l'optimum dans un pire-cas.

Ainsi L'algorithme du simplexe est un algorithme de complexité exponentielle au sens classique de la complexité pire-cas.

Beaucoup de résulats ont permis de déterminer des règles de choix de variables, de pivot, de prétraitement efficaces permettant d'accélérer l'algorithme.
Néanmoins l'algorithme reste de complexité exponentielle (dans le pire des cas).

Nombre d'itérations constaté

Si l'algorithme du simplexe a une complexité exponentielle pire-cas,
les pire-cas apparaissent peu **"en pratique"** et un nombre faible d'itérations est suffisant !

Il est difficile de définir exactement le terme "en pratique" :
- des PL provenant des problèmes d'optimisation réels où les coefficients sont rationnels
- des coefficients de valeurs éloignées les unes des autres.
- ...

Il est difficile de définir ce "en pratique"... pourtant la réalité quotidienne de l'utilisation de la PL est claire : aucun PL rencontré dans le quotidien d'un optimiseur de l'industrie ne demande à l'algorithme du simplexe un temps dépassant quelques minutes : même pour des PL de 200 000 variables et 200 000 inégalités !

Cette question de la présence faible de pire-cas laisse penser qu'il y existe une **structure combinatoire particulière** dans ces PL "utilisé en pratique".
C'est une question de recherche importante qui passionne à la fois les algorithméticiens, les combinatoriciens et les géométriciens !
Avec une meilleure connaissance de cette structure polyédrale, on en saurait davantage sur la fameuse question "P est il-différent de NP ?" ($P ? = NP$).

Complexité de la Programmation Linéaire

L'algortihme du simplexe n'est pas le seul algorithme capable de résoudre tout programme linéaire (en variable continue).

La complexité de la Programmation Linéaire se définit comme la complexité du meilleur algorithme permettant de résoudre tout PL.

Posée explicitement dans les années 1870, la question de la complexité de la programmation linéaire n'est pas résolue par Georges Dantzig lorsqu'il propose l'algorithme du simplexe en 1947 car Klee et Minty prouve sa complexité expontielle en 1970.

En 1979, Leonid Khatchian s'inspire de la méthode des ellipsoïdes connues dans un autre cadre pour proposer un algorithme polynomial pour la Programmation Linéaire ! La Programmation Linéaire est donc polynomiale !

Mais le degré du polynome de la complexité de la méthode des ellipsoïdes est assez élevé : cette méthode est très lente en pratique : elle est largement battu par l'algorithme du simplexe sur les fameux cas pratiques.

En 1984, Narendra Karmarkar propose la méthode des points intérieurs qui est polynomiale et efficace !

Résolution et solveurs

Un programme linéaire peut être résolu efficacement :
- par la méthode des points intérieurs (qui est polynomial)
- par l'algorithme du simplexe (qui est pourtant exponentiel).

Quelle méthode choisir ?

Depuis 1947, de nombreux travaux ont permis des implémentations efficaces de ces 2 méthodes : chacune progressant peu à peu.

Il existe de nombreux **solveurs de programmation linéaire** :
- des solveurs commerciaux Cplex (IBM), Gurobi, Xpress et même Matlab ou Excel...
- des solveurs universitaires : Lp (COIN-OR), Soplex (univesité ZIB)
- des solveurs libres : Glpk (gnu)

Les meilleurs d'entre eux peuvent résoudre des PL jusqu'à 200 000 variables et 200 000 contraintes en quelques minutes.

Remarque : En revanche, lorsque les variables doivent prendre des valeurs discrètes (par exemple binaires), la Programmation Linéaire en Nombres Entiers (PLNE) est un problème NP-difficile que les solveurs ont du mal à résoudre en général : la plupart des PLNE contenant à peine un millier de lignes et colonnes ne peuvent pas être résolus après plusieurs semaines de calculs.

Hypothèses

Revenons à l'algorithme du simplexe et à sa preuve.

En effet, cet algorithme est à la base de nombreuses méthodes pour des problèmes célèbres (flots, affectations...).

Pour montrer la validité et la terminaison de l'algorithme du simplexe, plaçons-nous sous l'hypothèses suivantes :
- on connaît une base initiale pour démarrer la phase II
- on définira aussi plus tard l'hypothèse de non cyclage
(Ces deux hypothèses seront levées dans le chapitre suivant).

Notations matricielles

On présente ici l'algorithme du simplexe sous forme matriciel dans l'objectif de prouver la validité de l'algorithme.

Considèrons un programme linéaire à n variables et m inégalités non-triviales sous sa forme canonique

$$\left\{\begin{array}{lll} \text{Max} & z = \sum_{j=1}^{n} c_j x_j & \\ \text{s.c.} & & \\ & \sum_{j=1}^{n} a_{ij} x_j \leq b_i & \forall i \in \{1, \ldots, m\} \\ & x_j \geq 0 & \forall j \in \{1, \ldots, n\} \end{array}\right. \leftrightarrow \left\{\begin{array}{ll} \text{Max} & z = c^T x \\ \text{s.c.} & \\ & A\, x \leq b \\ & x \geq 0 \end{array}\right.$$

où

- $x \in \boldsymbol{R}^n$
- $c \in \boldsymbol{R}^n$
- $b \in \boldsymbol{R}^m$
- A est une matrice réelle à m lignes et n colonnes avec :
  a_{ij} son coefficients de i-ème ligne et j-ème colonne.
  On note A^j la j-ème colonne de A.

Notations matricielles

Ecrivons la forme standard

$$\left\{\begin{array}{lllll} \max & \sum_{j=1}^{n} c_j x_j & & & \\ s.c. & \sum_{j=1}^{n} a_{ij} x_j & + x_{n+i} & = b_i & \forall i \in \{1,\ldots,m\} \\ & x_j & & \geq 0 & \forall j \in \{1,\ldots,n\} \\ & & x_{n+i} & \geq 0 & \forall i \in \{1,\ldots,m\} \end{array}\right. \leftrightarrow \left\{\begin{array}{ll} \text{Max} & z = c^T x \\ \text{s.c.} & \\ & [\ A \quad I\ ]\ x \leq b \\ & x \geq 0 \end{array}\right.$$

où

- $x \in \boldsymbol{R}^{n+m}$
- $c \in \boldsymbol{R}^{n+m}$ avec m nouvelles composantes nulles
- $b \in \boldsymbol{R}^{m}$
- $[\ A \quad I\ ]$ désigne une matrice obtenue en ajoutant à A les n colonnes de la matrice identité I

Base candidate

Une base candidate B est un ensemble de m indices de colonnes parmi les $n+m$ colonnes de la matrice $[\ A \quad I\ ]$.

On note alors $B(i)$ l'indice de la i-ème colonne choisie dans$[\ A \quad I\ ]$.

On va également noter B la matrice associée à la base B, c'est-à-dire les m colonnes $B(i)$, $i \in B$.

$$B = \left[A^{B(1)} A^{B(2)} \dots A^{B(m)} \right]$$

On dit que les colonnes $A^{B(1)}, A^{B(2)} \dots A^{B(m)}$ sont les *colonnes de base* et les autres colonnes les *colonnes hors-bases*.
Cela correspond exactement aux variables de base et hors-base de la version algébrique.

Une remarque importante : la base des variables d'écart correspond à la matrice identité !

Base

Ecrivons le PL suivant sous forme standard matricielle.

$$
\begin{array}{llllll}
\text{Min} & 5x_1 & +5x_2 & +3x_3 & \\
 & x_1 & +3x_2 & +x_3 & \leq 3 \\
 & -x_1 & & +3x_3 & \leq 2 \\
 & 2x_1 & -x_2 & +2x_3 & \leq 4 \\
 & 2x_1 & +3x_2 & -x_3 & \leq 2 \\
 & \multicolumn{4}{l}{x_i \geq 0, \forall i \in \{1,2,3\}.}
\end{array}
$$

$n = 3$ et $m = 4$

$$
c = \begin{bmatrix} 5 \\ 5 \\ 3 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
\quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix}
\quad [\, A \quad I \,] = \left[\begin{array}{ccc|cccc} 1 & 3 & 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 3 & 0 & 1 & 0 & 0 \\ 2 & -1 & 2 & 0 & 0 & 1 & 0 \\ 2 & 3 & -1 & 0 & 0 & 0 & 1 \end{array}\right]
\quad b = \begin{bmatrix} 3 \\ 2 \\ 4 \\ 2 \end{bmatrix}
$$

Base primale

On dit qu'une base est *primale* si la matrice B est inversible
i.e. si ses m colonnes sont linéairement indépendantes.

Si B est primale, on va réécrire le PL en utilisant la base B comme "clef" d'écriture.

Posons

- $E^j = B^{-1}A^j \qquad$ pour tout $j \in \{1, \dots, n+m\}$
  le vecteur des coordonnées de la colonne A^j selon la base primale B
- $E^0 = B^{-1}b$
  le vecteur des coordonnées de b sur B.

Ecriture en base

Donc pour une colonne $j \in \{1, \ldots, n+m\}$ donnée, on a

$$A^j = BE^j$$

ou encore

$$A^j = \sum_{l=1}^{m} E_l^j A^{B(l)}$$

Rappelons que $A^{B(l)}$ est la l-ème colonne de B.
On peut donc dire que le réel E_l^j est la l-ème coordonnée de la colonne A^j sur la base primale B
et donc que E^j est le vecteur des coordonnées de la colonne A^j sur la base primale B.

Ecriture en base

Sur l'exemple où $[\ A\ \ I\ ] = \begin{bmatrix} 1 & 3 & 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 3 & 0 & 1 & 0 & 0 \\ 2 & -1 & 2 & 0 & 0 & 1 & 0 \\ 2 & 3 & -1 & 0 & 0 & 0 & 1 \end{bmatrix}$
la base $B = (A^7, A^3, A^5, A^6)$ (en respectant cet ordre)

correspond à la matrice $B = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 3 & 1 & 0 \\ 0 & 2 & 0 & 1 \\ 1 & -1 & 0 & 0 \end{bmatrix}$
C'est une base primale car sa matrice est inversible (q diagonale de produit des termes diagonaux non nuls).

Essayons de déterminer E^1 "à la main" dans la base B ci-dessus :
On peut remarquer que :

- $A^1 = A^4 - A^5 + 2A^6 + 2A^7$ par jeu des vecteurs canoniques A^4, A^5, A^6 et A^7
- et $A^3 = A^4 + 3A^5 + 2A^6 - A^7$ (idem) donc $A^4 = A^3 - 3A^5 - 2A^6 + A^7$
- Donc $A^1 = A^3 - 4A^5 + 3A^7$

On a donc $E^1 = \begin{bmatrix} 3 \\ 1 \\ -4 \\ 0 \end{bmatrix}$.

Ecriture en base

Rappel : vecteur canonique $\varepsilon_i^T = \begin{array}{ccccccc} 1 & & & i & & & m \\ \end{array}$
$\begin{pmatrix} 0 & \dots & 0 & 1 & 0 & \dots & 0 \end{pmatrix}$

Remarque importante : si $B(i) = j$ alors E^j est le vecteur canonique $_i$.
En d'autre terme, si la j-ième colonne de $[\ A \quad I\ ]$ est la i-ième colonne de la base, une écriture en base B de cette j-ième colonne étant une combinaison linéaire : elle ne peut être qu'elle-même : or elle est en i-ème position dans la base.

Exemple : Dans l'exemple précédent, on peut faire le même calcul que dans l'exemple de E^1 pour obtenir E^5 mais on obtiendra

$E^5 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$ car A^5 est la 3ème colonne de $B = (A^7, A^3, A^5, A^6)$ donc $E^5 = \varepsilon_3$.

Base et Hors-Base

Pour une base primale B, on considère également les colonnes hors-bases numérotées $N(1), N(2), \ldots, N(n)$.
On a alors les matrices

$$B = \left[A^{B(1)} A^{B(2)} \ldots A^{B(n)}\right] \qquad \text{et} \qquad N = \left[A^{N(1)} A^{N(2)} \ldots A^{N(n)}\right]$$

On note
$x_B = (x_{B(1)}, x_{B(2)}, \ldots, x_{B(m)})$ le vecteur des variables de base.
$c_B = (c_{B(1)}, c_{B(2)}, \ldots, c_{B(m)})$ le vecteur des coûts des variables de base.
$x_N = (x_{N(1)}, x_{N(2)}, \ldots, x_{N(m)})$ le vecteur des variables hors-base.
$c_N = (c_{N(1)}, c_{N(2)}, \ldots, c_{N(m)})$ le vecteur des coûts des variables hors-base.

Le PL s'écrit alors également

$$\begin{array}{ll} \text{Max} & c_B x_B + c_N x_N \\ & B x_B + N x_N = b \\ & x_B \geq 0 \\ & x_N \geq 0 \end{array}$$

Ecriture dictionnaire

En multipliant les contraintes par B^{-1}, on obtient

$$x_B + B^{-1}Nx_N = B^{-1}b$$

et

$$x_B = B^{-1}b - B^{-1}Nx_N.$$

En utilisant cela dans le PL, on obtient l'écriture dictionnaire vue dans la section "algébrique" :

$$x_B = B^{-1}b - B^{-1}Nx_N$$

avec l'écriture de z selon les variables hors-base

$$z = c_B B^{-1}b + (c_N - c_B B^{-1}N)x_N$$

Coûts réduits (ou marginaux)

Un des résultats de cette écriture est la définition très importante en économie des coûts marginaux.
Les coefficients des variables hors-bases dans cette écritures sont appellées coûts réduits.

$$\bar{c}_N = c_N - c_B B^{-1} N$$

On les appelera aussi "coût marginaux" dans la section Dualité .

Dans l'écriture dictionnaire algébrique de la section précédente, c'est tout simplement les coûts obtenus dans l'écriture de z selon les variables hors-base.

Remarque : Le coût réduit d'une variable de base est donc 0.

Base primale réalisable

Une base primale est dite réalisable si $E^0 = B^{-1}b \geq 0$.

En effet, si B est une base primale réalisable, alors la solution définie par

$$x_B = E^0 \text{ et } x_N = 0$$

définit une solution réalisable du PL appelée *solution de base associée à la BRP B*.
Sa valeur objective correspondante est $z_B = c_B E^0$.

Cas particulier important :
Si $b \geq 0$, la base des variables d'écart est primale réalisable.
En effet, la matrice B est la matrice identité.
Cela revient à dire que dans une écriture sous forme canonique, si toutes les valeurs $b_i \geq 0$, $i = 1, \ldots, m$, alors la base des variables d'écart peut être la base initiale de la phase II de l'algorithme du simplexe.

Deux cas sont donc à identifier :
- Si $b \geq 0$, on peut ainsi initier l'algorithme du simplexe : que l'on appelle *Phase II* .
- Si b possède des valeurs négatives, la base des variables d'écart n'est pas réalisable : il faut alors précéder la phase II d'une phase d'initialisation appelée *Phase I* (voir section correspondante).

Cas d'un PL non-borné

Un cas particulier de base mène à prouver qu'un PL est non-borné.

Lemme

Pour une base B donnée, s'il existe une colonne hors-base j de coût réduit srictement positif et tel que $E^j = B^{-1}A^j \leq 0$, alors le PL est non-borné dans la direction d'optimisation.

Preuve : Pour une telle base B et une telle colonne j, remarquons que la fonction objectif s'écrit $z = c_B B^{-1} b + (c_N - c_B B^{-1} N) x_N$. Soit un réel $M > 0$, et considérons la solution $\tilde{x}$ telle que $\tilde{x}_e = M$, $\tilde{x}_k = 0$ pour tout autre variable hors base k et les valeurs $\tilde{x}_l$ des variables de base calculées par l'écriture dictionnaire. Alors on peut remarquer que $\tilde{x}$ est solution du PL. Ainsi pour tout $M > 0$, il existe une solution $z > M$, ce qui signifie que le PL est non-borné dans la direction d'optimisation. □

Dans l'énoncé algébrique de l'algorithme du simplexe, cela correspond à l'arrêt en cas de solution infinie.

On dit aussi "oralement" que le PL admet une "solution infinie".

Critère d'optimalité

Théorème

Pour une base primale réalisable B et variables hors-base N, si $\bar{c}_N \leq 0$, i.e. si tous les coûts réduits sont négatifs ou nuls, alors la solution de base associée à B est optimale.

Preuve : Considérons une base B^* suivant l'hypothèse. Dans cette base B^*, la fonction objective z est écrite en fonction des variables hors-base, c'est-à-dire

$$z = c_B B^{*-\mathbf{1}} b + \bar{c}_N x_N.$$

Or par hypothèse, $\bar{c}_N \leq 0$ et par définition $x_N \geq 0$.
On obtient donc une borne supérieure

$$z \leq c_B B^{*-\mathbf{1}} b$$

D'autre part, la solution x^* correspondante à B^* correspond exactement à une solution de valeur $c_B B^{*-\mathbf{1}} b$: cette borne supérieure est donc atteinte pour x^* qui est donc optimale. $\square$

Pivot

Deux bases B et B' sont dites voisines si B' est obtenue à partir de B en remplaçant une colonne B^s par une colonne A^e de A,

c'est-à-dire $\forall k = 1, \dots, m \quad B'(k) = \begin{cases} e & \text{si } k = e \\ B(k) & \text{sinon} \end{cases}$

L'opération suivante, dite pivot permet de passer de B à B' :

- Repérons une colonne A^e hors-base telle que E^e contient des composantes strictement positive.
- Repérons une ligne s telle que $E^e_s > 0$.
- La ligne s sert alors de ligne d'échange

$$x_e = \frac{1}{E^e_s}(B^{-1}b)_s - \frac{1}{E^e_s} x_{B(s)} - \sum_{j \in N \setminus \{s\}} \frac{E^j_s}{E^e_s} x_j$$

- Dans les autres lignes ($k \neq s$), on substitue x_e par l'expression précédente

$$x_{B(k)} = (B^{-1}b)_k - \frac{E^e_k}{E^e_s}(B^{-1}b)_s - \frac{E^e_k}{E^e_s} x_{B(s)} - \sum_{j \in N \setminus \{s\}} \left(E^j_k - \frac{E^e_k E^j_s}{E^e_s} \right) x_j$$

Pivot

La solution x' associée à la base B' est alors donnée par :

$$x'_N = 0$$
$$x'_e = \frac{1}{E_s^e}(B^{-1}b)_s$$
$$x'_{B(k)} = (B^{-1}b)_k - \frac{E_k^e}{E_s^e}(B^{-1}b)_s$$ pour les colonnes de la base B' autre que e.

La question est donc de savoir si la base B' est primale et réalisable.

Pivot

Lemme

Si les deux conditions suivantes sont vérifiées :
i) $E_s^j > 0$
ii) $(B^{-1}b)_k - \frac{E_k^e}{E_s^e}(B^{-1}b)_s \geq 0$, *pour tout* $k \neq s$
Alors la base B' *obtenu par pivot est primale réalisable.*

Preuve : Remarquons que, par les hypothèses i) et ii), la solution x' associée à la base B est bien positive ou nulle. Donc x' limitée aux variables initiales vérifient bien le PL. Donc si B' est primale, B' est réalisable.
Pour prouver que B' est primale, c'est-à-dire que B' est inversible, nous allons prouver que $B'x = b$ admet une unique solution. On peut déjà remarquer que la restriction x'_B de x' à B est solution de $B'x = b$, donc ce système admet bien une solution. Soit $\tilde{x}_B$ une solution de $B'x = b$. Posons alors $\tilde{x}_N = 0$. Considérons alors le système

$$Bx_B + Nx_N = b \text{ et } x_N = 0.$$

Par l'écriture dictionnaire vue plus haut, on peut constater que ce système mène à une unique solution, qui est précisement x'_B. Donc ce système et par conséquent $B'x = b$ ont une unique solution. □

Validité du pivot

On peut alors énoncer le résultat fondamental pour l'algorithme du simplexe.

Théorème

Si on peut choisir :
- une colonne entrante e de coût réduit $\bar{c}_e > 0$
- une colonne sortante s telle que

$$\frac{1}{E_s^e}(B^{-1}b)_s = \min\left\{(B^{-1}b)_k - \frac{E_k^e}{E_s^e}(B^{-1}b)_s\right\}$$

Alors B' est une base primale réalisable et $z_{B'} \geq z_B$.

Si de plus, B' est telle que $B'^{-1}b$ ne possède aucune composante nulle, alors $z_{B'} > z_B$.

Validité du pivot

Preuve : En choisissant une des colonnes e et s ainsi définie, par le théorème précédent, la base B' obtenue par pivot est une primale réalisable. En effet, en prenant le min de toutes les formules ii), on vérifie bien les hypothèses du théorème précédent. Posons z_B (resp. $z_{B'}$) la valeur associée à B (resp. B'). Or on a par pivot

$$z_{B'} = z_B + (c_e - c_B B^{*-1} A^e) x'_e$$

$$\text{Donc } z_{B'} = z_B + \frac{1}{E_s^e}(B^{-1}b)_s(c_e - c_B B^{*-1} A^e).$$

Or par hypothèse, $\frac{1}{E_s^e}(B^{-1}b)_s \geq 0$ et $(c_e - c_B B^{*-1} A^e) > 0$. Donc $z_{B'} \geq z_B$.
Si de plus $\frac{1}{E_s^e}(B^{-1}b)_s\ 0$, on a $z_{B'} > z_B$. $\square$

Schéma de l'algorithme

On obtient alors l'algorithme :

Schéma de l'algorithme du simplexe Phase II :

Initialisation par une base réalisable B

si $b \geq$: Pour $i = 1, \ldots, m$, $B(i) \leftarrow n + i$

sinon utiliser la phase I (voir cours suivant)

Répéter

Calculer x solution de base associée à B
Calculer N matrice des variables hors-base pour B
Si $(c_N - c_B B^{-1} N) \leq 0$ alors Retourner(x)
Choix d'une colonne A^e de N telle que $c_e - c_B B^{-1} A^e > 0$
Si $E^e = B^{-1} A^e \leq 0$ alors Retourner ("PL non-borné")
Calculer s tel que

$$\frac{1}{E_s^e}(B^{-1}b)_s = \min \left\{ (B^{-1}b)_k - \frac{E_k^e}{E_s^e}(B^{-1}b)_s \right\}$$

$B(s) \leftarrow e$

Fin Répéter

Choix d'une variable sortante et dégénérescence

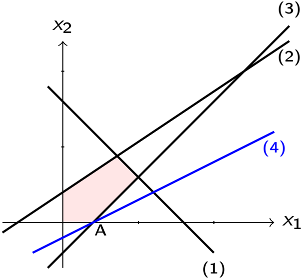
On remarque que lorsqu'à une itération passant de la matrice B à la matrice B',
s'il y a annulation d'un coefficient de $B'^{-1}b$,
- plusieurs choix sont possibles pour la variable sortante
- la valeur de la fonction objective z reste inchangée

On appelle une base dégénérée une base B
telle que $B^{-1}b$ possède au moins une composante nulle.

Pivot dégénéré

Ajoutons une inégalité au PL de l'exemple

$$\begin{cases} \max\ z = & 6x_1 & + & 5x_2 & & & \\ & x_1 & + & x_2 & \leq & 8 & (1) \\ & -2x_1 & + & 3x_2 & \leq & 6 & (2) \\ & x_1 & - & x_2 & \leq & 2 & (3) \\ & x_1 & - & 2x_2 & \leq & 2 & (4) \\ & x_1 & & & \geq & 0 & (Ox_2) \\ & & & x_2 & \geq & 0 & (Ox_1) \end{cases}$$



La droite (4) correspondant à la nouvelle inégalité :
- passe comme (1) et (Ox_1) par le point A(2,0)
- ne coupe aucun point de l'espace de définition.

Si on note e_4 sa variable d'écart, on voit que, dans les 3 cas de bases :
- $B_1 = \{x_1, e_2, e_3, e_4\}$, hors-base : $\{x_2, e_1\}$
- $B_2 = \{x_1, e_1, e_2, e_3\}$, hors-base : $\{x_2, e_4\}$
- $B_3 = \{x_1, x_2, e_1, e_3\}$, hors-base : $\{e_1, e_4\}$

l'annulation des variables hors-base correspond au même point A(2,0) et à la même solution $s_1 = (2, 0, 6, 10, 0)$ et donc à la même valeur $z = 12$.

Pivot dégénéré

Une itération du simplexe sur une des trois bases peut mener à passer sur une autre de ces bases : l'itération sera dite dégénérée et le coût de z n'augmentera pas : on parle d'itération "à vide".

Souvent, après avoir explorer un lot de bases dégénérées correspondant à une même solution, l'algorithme se poursuit avec une solution différente sur un nouveau point.

Il peut se produire un cyclage où l'on revient à la base d'origine et ainsi à l'infini !

Heureusement, plusieurs améliorations ont été mises au point pour éviter ce problème du cyclage (voir chapitre suivant).

Terminaison

Les deux hypothèses faites au-début de cette preuve peuvent alors se réécrire :

- hypothèse d'initialisation : $b \geq 0$
- non-dégénérescence : chaque base B rencontré possède au moins une composante de $B^{-1}b$ non nulle

Remarque essentielle : Le nombre de bases primales réalisables est fini (car majoré par le nombre de combinaisons de m colonnes parmi $n + m$ colonnes).

Terminaison

Théorème

Sous les hypothèse précédentes, l'algorithme Phase II se termine.

Preuve : La boucle Répéter de l'algorithme consiste à passer de base en base.
Or si aucune base n'est dégénérée, à chaque itération la valeur de la fonction objective augmente strictement.
Ainsi, l'algorithme parcourt uniquement des bases distinctes.
Or il existe un nombre fini de bases possibles (borné par C^n_{m+n}).
Donc la boucle Répéter crée une suite finie de bases distinctes de valeurs z croissante.
Donc la boucle Répéter a un nombre fini d'itérations. $\square$

Validité

Théorème

Sous les hypothèse précédentes, l'algorithme Phase II est valide.

Preuve : Notons B^* la dernière base de la boule Répéter (il en existe une car l'algorithme se termine).
Trois cas sont possibles :

- soit pour B^*, le coût réduit est négatif ou nul pour chaque colonne et la base rencontrée est donc optimale par le théorème du critère d'optimalité.
- soit pour B^* le coût réduit $\bar{c}_e$ d'une des colonnes est positif avec une colonne $E^e = B^{-1}A^e \leq 0$, dans ce cas, on prouve que le PL est non-borné par le théorème correspondant.
- soit pour B^* le coût réduit $\bar{c}_e$ d'une des colonnes est positif avec une colonne $E^e = B^{-1}A^e$ possédant un coefficient strictement positif. Dans c cas, par le théorème de validité du pivot et sous l'hypothèse qu'aucune des bases n'est dégénérée : on peut réaliser une opération de pivot qui produit une base de solution associée strictement supérieure à z^*. Ce qui contredit l'existence de ce cas.

Ainsi, l'algorithme se termine soit en prouvant que le PL est non-borné, soit en fournissant une base optimale. □

Prochain cours

Dans le cours suivant, pour terminer à la fois la preuve et obtenir un algorithme complet, on s'intéressera aux deux cas :

Initialisation : utilisation de la phase I d'initialisation de l'algorithme du simplexe.

Dégénérescence : si au cours de l'algorithme une base est dégénérée, l'algorithme doit prendre en compte la présence de cyclage.