Méthodes et modélisation pour l'optimisation

M1 informatique, 2018–2019

04 — L'algorithme du simplexe (suite)

UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

INSTITUT D'ÉLECTRONIQUE ET D'INFORMATIQUE GASPARD-MONGE

La semaine dernière

$$\begin{aligned} \max\ & x_1 + x_2 \\ -x_1 + x_2 &\leq 1 \\ x_1 &\leq 3 \\ x_2 &\leq 2 \\ x_1, x_2 &\geq 0 \end{aligned}$$

programme initial

$$\begin{aligned} \max\ & x_1 + x_2 \\ -x_1 + x_2 + x_3 &= 1 \\ x_1 + x_4 &= 3 \\ x_2 + x_5 &= 2 \\ x_1, x_2, x_3, x_4, x_5 &\geq 0 \end{aligned}$$

forme équationnelle

$$\begin{array}{rcl} x_3 &=& 1 + x_1 - x_2 \\ x_4 &=& 3 - x_1 \\ x_5 &=& 2 - x_2 \\ \hline z &=& 0 + x_1 + x_2 \end{array}$$

tableau

La semaine dernière

$$\begin{aligned} x_3 &= 1 + x_1 - x_2 \\ x_4 &= 3 - x_1 \\ x_5 &= 2 - x_2 \\ \hline z &= 0 + x_1 + x_2 \end{aligned}$$

base {3, 4, 5} solution (0, 0, 1, 3, 2)

x2 entre, x3 sort

pivot

$$\begin{aligned} x_2 &= 1 + x_1 - x_3 \\ x_4 &= 3 - x_1 \\ x_5 &= 1 - x_1 + x_3 \\ \hline z &= 1 + 2 \cdot x_1 - x_3 \end{aligned}$$

base {2, 4, 5} solution (0, 1, 0, 3, 1)

x1 entre, x5 sort

$$\begin{aligned} x_1 &= 1 + x_3 - x_5 \\ x_2 &= 2 - x_5 \\ x_4 &= 2 - x_3 + x_5 \\ \hline z &= 3 + x_3 - 2 \cdot x_5 \end{aligned}$$

base {1, 2, 4} solution (1, 2, 0, 2, 0)

x3 entre, x4 sort

$$\begin{aligned} x_1 &= 3 - x_4 \\ x_2 &= 2 - x_5 \\ x_3 &= 2 - x_4 + x_5 \\ \hline z &= 5 - x_4 - x_5 \end{aligned}$$

base {1, 2, 3} solution (3, 2, 2, 0, 0)

La semaine dernière

$$\begin{aligned} x_1 &= 3 - x_4 \\ x_2 &= 2 - x_5 \\ x_3 &= 2 - x_4 + x_5 \\ \hline z &= 5 - x_4 - x_5 \end{aligned}$$

base {1, 2, 3}   solution (3, 2, 2, 0, 0)

coeffs ≤ 0 — terminaison

$$\begin{aligned} &\max\ x_1 + x_2 \\ -x_1 + x_2 + x_3 &= 1 \\ x_1 + x_4 &= 3 \\ x_2 + x_5 &= 2 \\ x_1, x_2, x_3, x_4, x_5 &\geq 0 \end{aligned}$$

vérifier que la solution est admissible !

$$\begin{aligned} &\max\ x_1 + x_2 \\ -x_1 + x_2 &\leq 1 \\ x_1 &\leq 3 \\ x_2 &\leq 2 \\ x_1, x_2 &\geq 0 \end{aligned}$$

solution au programme initial : $(x_1, x_2) = (3, 2)$
valeur de la fonction objectif = 5

Un autre exemple

$$\begin{aligned} \max\ & x_1 \\ x_1 - x_2 &\le 1 \\ -x_1 + x_2 &\le 2 \\ x_1, x_2 &\ge 0 \end{aligned}$$

programme initial

$$\begin{aligned} \max\ & x_1 \\ x_1 - x_2 + x_3 &= 1 \\ -x_1 + x_2 + x_4 &= 2 \\ x_1, x_2, x_3, x_4 &\ge 0 \end{aligned}$$

forme équationnelle

$$\begin{array}{rcl} x_3 &=& 1 - x_1 + x_2 \\ x_4 &=& 2 + x_1 - x_2 \\ \hline z &=& 0 + x_1 \end{array}$$

tableau

$$\begin{array}{rcl} x_1 &=& 1 + x_2 - x_3 \\ x_4 &=& 3 - x_3 \\ \hline z &=& 1 + x_2 - x_3 \end{array}$$

?

Un autre exemple

- il n'y a pas d'équation contraignante — on peut augmenter x_2 par une valeur $t \geq 0$ quelconque
- pour tout $t \geq 0$, la solution suivante est admissible :

$$(x_1, x_2, x_3, x_4) = (1, 0, 0, 3) + t \cdot (1, 1, 0, 0)$$

- la valeur de la fonction objectif = $1+t$

$$\begin{array}{rcl} x_1 &=& 1 + x_2 - x_3 \\ x_4 &=& 3 - x_3 \\ \hline z &=& 1 + x_2 - x_3 \end{array}$$

Un autre exemple

$$\begin{aligned} \max\ & x_1 \\ x_1 - x_2 &\leq 1 \\ -x_1 + x_2 &\leq 2 \\ x_1, x_2 &\geq 0 \end{aligned}$$

$$\begin{array}{lllll} x_1 & = 1 & + x_2 & - x_3 \\ x_4 & = 3 & & - x_3 \\ \hline z & = 1 & + x_2 & - x_3 \end{array}$$

- solution : $(x_1, x_2, x_3, x_4) = (1, 0, 0, 3)+t\cdot(1, 1, 0, 0)$
- solution au programme initial : $(x_1, x_2) = (1, 0)+t\cdot(1, 1)$
- la valeur de la fonction objectif = $1+t$
- la fonction objectif est non bornée !

Trouver une base initiale

- Comment trouver une solution basique admissible pour démarrer l'algorithme ?

$$\begin{array}{rrrrrrrl} & & & & & \max & x_1 + x_2 & \\ -x_1 & + x_2 & + \boxed{x_3} & & & = & 1 & \\ x_1 & & & + x_4 & & = & 3 & \\ & x_2 & & & + x_5 & = & 2 & \\ & & & & x_1, x_2, x_3, x_4, x_5 & \geq & 0 & \end{array}$$

$$\begin{array}{rl} \max & x_1 + 2 \cdot x_2 \\ x_1 + 3 \cdot x_2 + x_3 & = 4 \\ 2 \cdot x_2 + x_3 & = 2 \\ x_1, x_2, x_3 & \geq 0 \end{array}$$

facile

moins facile ?

Trouver une base initiale

- S'il y a une variable d'écart par ligne et les constantes sont non-négatives, alors c'est trivial
- Sinon, il est parfois possible d'en deviner une
- Mais en général, ce problème est "aussi difficile" que le problème d'optimisation

Combien de bases possibles dans une instance ayant 2n variables et n équations ?

Réponse : le nombre de combinaisons de n parmi 2n (colonnes) ≈ 4^n

Trouver une base initiale

$$\max\ x_1 + 2 \cdot x_2$$

$$x_1 + 3 \cdot x_2 + x_3 = 4$$
$$2 \cdot x_2 + x_3 = 2$$
$$x_1,\ x_2,\ x_3 \geq 0$$

$$\min\ x_4 + x_5$$

$$x_1 + 3 \cdot x_2 + x_3 + x_4 = 4$$
$$2 \cdot x_2 + x_3 + x_5 = 2$$
$$x_1,\ x_2,\ x_3,\ x_4,\ x_5 \geq 0$$

- Supposons qu'on essaye la solution non-admissible

$$x_1 = x_2 = x_3 = 0$$

- Alors, les côtés gauches des équations sont plus petits que les côtés droits
- On veut minimiser les différences

$$x_4 = 4 - (x_1 + 3 \cdot x_2 + x_3)$$
$$x_5 = 2 - (2 \cdot x_2 - x_3)$$

Trouver une base initiale

$$\max\ x_1 + 2 \cdot x_2$$

$$\begin{aligned} x_1 + 3 \cdot x_2 + x_3 &= 4 \\ 2 \cdot x_2 + x_3 &= 2 \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

$$\max\ - x_4 - x_5$$

$$\begin{aligned} x_1 + 3 \cdot x_2 + x_3 + x_4 \quad &= 4 \\ 2 \cdot x_2 + x_3 \quad + x_5 &= 2 \\ x_1, x_2, x_3, x_4, x_5 &\geq 0 \end{aligned}$$

programme auxiliaire

- Soit $(x_1, x_2, x_3, x_4, x_5)$ une solution optimale au programme auxiliaire
  - si l'optimum est 0, alors (x_1, x_2, x_3) est une solution basique admissible au programme initial
  - sinon, le programme initial n'a pas de solution basique admissible

Trouver une base initiale

$$\max \; -x_4 - x_5$$

$$\begin{aligned} x_1 + 3\cdot x_2 + x_3 + x_4 \quad &= 4 \\ 2\cdot x_2 + x_3 \quad + x_5 &= 2 \\ x_1, x_2, x_3, x_4, x_5 &\geq 0 \end{aligned}$$

$$\begin{aligned} x_4 &= 4 - x_1 - 3\cdot x_2 - x_3 \\ x_5 &= 2 \quad\;\; - 2\cdot x_2 - x_3 \\ \hline z &= -6 + x_1 + 5\cdot x_2 + 2\cdot x_3 \end{aligned}$$

base {4, 5} solution (0, 0, 0, 4, 2)

x_1 entre, x_4 sort

$$\begin{aligned} x_1 &= 4 - 3\cdot x_2 - x_3 - x_4 \\ x_5 &= 2 - 2\cdot x_2 - x_3 \\ \hline z &= -2 + 2\cdot x_2 + x_3 - x_4 \end{aligned}$$

base {1, 5} solution (4, 0, 0, 0, 2)

x_2 entre, x_5 sort

$$\begin{aligned} x_1 &= 1 + x_3/2 - x_4 \\ x_2 &= 1 - x_3/2 \quad - x_5/2 \\ \hline z &= 0 \quad - x_4 - x_5 \end{aligned}$$

base {1, 2} solution (1, 1, 0, 0, 0)

Trouver une base initiale

$$\max\ x_1 + 2\cdot x_2$$

$$\begin{aligned} \boxed{x_1 + 3\cdot x_2} + x_3 &= 4 \\ \boxed{2\cdot x_2} + x_3 &= 2 \\ x_1,\ x_2,\ x_3 &\geq 0 \end{aligned}$$

$$\begin{aligned} x_1 &= 1 + x_3/2 - x_4 \\ x_2 &= 1 - x_3/2 - x_5/2 \\ \hline z &= 0 - x_4 - x_5 \end{aligned}$$

base {1, 2}   solution (1, 1, 0, 0, 0)

- $(x_1, x_2, x_3, x_4, x_5) = (1, 1, 0, 0, 0)$
  est une solution optimale au programme auxiliaire
- l'optimum est 0, donc $(x_1, x_2, x_3) = (1, 1, 0)$
  est une solution basique admissible au programme initial
  qui correspond à la base {1, 2}

Trouver une base initiale

$$\begin{aligned} \max\ & c^T x \\ Ax &= b \\ x &\geq 0 \end{aligned}$$

programme initial

$$\begin{aligned} \max\ & -x_{n+1}-x_{n+2}-\ldots-x_{n+m} \\ A'x' &= b \\ x' &\geq 0 \end{aligned}$$

programme auxiliaire

- Plus généralement, supposer que le côté droit de chaque équation est ≥ 0, c-à-d, $b_i \geq 0$, sinon multiplier par -1
- Ajouter une nouvelle variable par équation : $x_{n+1}, x_{n+2}, \ldots, x_{n+m}$
- Soit $A' = (A \mid I_m)$, où I_m est la matrice identité et $x' = (x_1, \ldots, x_n, x_{n+1}, \ldots, x_{n+m})$
- $(0, \ldots, 0, b_1, \ldots, b_m)$ est une solution basique admissible
- Si $(x_1, \ldots, x_n, x_{n+1}, \ldots, x_{n+m})$ est une solution optimale au programme auxiliaire avec la valeur 0, alors $(x_1,\ldots,x_n)$ est une solution basique admissible au programme initial. Sinon, le programme initial n'a pas de solution

Hypothèses

- On suppose que $Ax = b$ a des solutions
  - sinon, $Ax = b$, $x \geq 0$ n'a pas de solution non plus
  - peut être détecté par l'élimination de Gauss
- On suppose que les lignes de la matrice A sont indépendantes
  - sinon, au moins une des équations est redondante et peut être éliminée sans changer les solutions à $Ax = b$
  - peut être détecté par l'élimination de Gauss

L’algorithme du simplexe

Initialisation

1. Convertir le programme en ***forme équationnelle***.
2. Si Ax = b n’a pas de solution, alors
   - terminer : ``pas de solution”
3. Faire en sorte que les lignes de A soient ***indépendantes***.
4. Trouver une solution basique initiale (résoudre le ***programme auxiliaire***).
   - s’il n’y en a pas, terminer : ``pas de solution”
5. Démarrer la boucle principale dans la solution basique initiale.

L'algorithme du simplexe

Boucle principale

1. Exprimer le tableau correspondant à la solution basique admissible.
2. Si tous les coefficients de z dans le tableau sont ≤ 0, alors
   - terminer : la solution optimale correspond au dernier tableau
3. Choisir une variable hors base avec coefficient > 0 (***la variable entre***)
4. Choisir une variable basique ayant une équation parmi les plus contraignantes (***la variable sort***)
5. S'il n'y a pas d'équation contraignante, alors
   - terminer : ``optimum non borné"
6. GOTO 1

Questions

- Comment choisir la variable qui entre et celle qui sort ?
- L'algorithme du simplexe termine-t-il toujours ?
- L'algorithme du simplexe est-il efficace ?

Règle de pivot (qui entre, qui sort ?)

- Le plus grand coefficient entre (la règle de Dantzig)
- La plus grande augmentation de z
- Le côté le plus raide (“steepest edge”)
- Le plus petit indice entre et sort (la règle de Bland)

- *Le plus petit indice* ***sort*** *(moins important)*

La règle de Dantzig

Notre choix pour les exercices !

- Le plus grand coefficient entre

$$\begin{aligned} x_4 &= 4 - x_1 - 3\cdot x_2 - x_3 \\ x_5 &= 2 - 2\cdot x_2 - x_3 \\ \hline z &= -6 + x_1 + 5\cdot x_2 + 2\cdot x_3 \end{aligned}$$

x_2 entre

La plus grande augmentation

- La plus grande augmentation de z
  - $x_1 \rightarrow 4$, z augmente par 4
  - $x_2 \rightarrow 1$, z augmente par 5
  - $x_3 \rightarrow 2$, z augmente par 4

$$
\begin{array}{l}
x_4 = 4 - x_1 - 3 \cdot x_2 - x_3 \\
x_5 = 2 - 2 \cdot x_2 - x_3 \\
\hline
z = -6 + x_1 + 5 \cdot x_2 + 2 \cdot x_3
\end{array}
$$

x_2 entre

Le côté le plus raide

Le champion en pratique !

- La plus grande augmentation de $z/||x_{new}-x_{old}||$
  - x_1 entre → aug. $z/||x_{new}-x_{old}|| = 4/\sqrt{17}$
  - x_2 entre → aug. $z/||x_{new}-x_{old}|| = 5/\sqrt{14}$
  - x_3 entre → aug. $z/||x_{new}-x_{old}|| = 4/\sqrt{12}$

$$
\begin{array}{l}
x_4 = 4 - x_1 - 3 \cdot x_2 - x_3 \\
x_5 = 2 - 2 \cdot x_2 - x_3 \\
\hline
z = -6 + x_1 + 5 \cdot x_2 + 2 \cdot x_3
\end{array}
$$

x_2 entre

La règle de Bland

- Le candidat du plus petit indice entre et sort

$$\begin{array}{rcl} x_4 &=& 4 - x_1 - 3 \cdot x_2 - x_3 \\ x_5 &=& 2 - 2 \cdot x_2 - x_3 \\ \hline z &=& -6 + x_1 + 5 \cdot x_2 + 2 \cdot x_3 \end{array}$$

x_1 entre

Dégénérescence

- Il est possible que l'inégalité la plus contraignante ne permette pas d'augmenter la variable entrant
- Cela s'appelle dégénérescence

$$\begin{array}{rcl} x_3 & = & x_1 - x_2 \\ x_4 & = & 2 - x_1 \\ \hline z & = & x_2 \end{array}$$

- Dans ce cas, on fait un pivot sans augmentation en espérant que cela permette d'augmenter dans le pivot suivant

Dégénérescence

$$
\begin{array}{rcl}
x_3 & = & x_1 - x_2 \\
x_4 & = & 2 - x_1 \\
\hline
z & = & x_2
\end{array}
$$

x_2 entre

x_3 sort

$$
\begin{array}{rcl}
x_2 & = & x_1 - x_3 \\
x_4 & = & 2 - x_1 \\
\hline
z & = & x_1 - x_3
\end{array}
$$

x_1 entre

x_4 sort

$$
\begin{array}{rcl}
x_1 & = & 2 - x_4 \\
x_2 & = & 2 - x_3 - x_4 \\
\hline
z & = & 2 - x_3 - x_4
\end{array}
$$

la fonction objectif a augmenté
et on a trouvé l'optimum

Terminaison

- Dans certains cas, il est possible d'entrer dans un cycle de dégénérescence sans augmentation.
- Exemple :
  - variables : x1, x2, x3, x4, base initiale : {x3, x4}
  - x1 entre, x3 sort, nouvelle base {x1, x4}
  - x2 entre, x4 sort, nouvelle base {x1, x2}
  - x3 entre, x1 sort, nouvelle base {x2, x3}
  - x4 entre, x2 sort, nouvelle base {x3, x4}
  - …
- Dans ce cas, *l'algorithme ne termine pas*.

Terminaison

- Comment éviter ou gérer un potentiel cycle de dégénérescence ?

1. Utiliser la règle de Bland, mais celle-là est très lente (le candidat du plus petit indice entre et sort)
2. Certains solveurs peuvent détecter un cycle et changer pour une autre règle temporairement pour le casser
3. Modifier légèrement la fonction objectif : une petite perturbation des coefficients bien choisie

Efficacité

- Le nombre de pivots est exponentiel dans le pire des cas
- En pratique, le pire des cas ne se produit pas, l'algorithme du simplexe est considéré efficace
- Problème théorique important : trouver une règle de pivot qui garantit un nombre polynomial de pivots dans le pire des cas
- Il y a d'autres algorithmes efficaces :
  - La méthode de l'ellipsoïde : polynomiale, lente en pratique
  - Des méthodes de point intérieur : polynomiales, efficaces en pratique