l'algorithme du simplexe

Algorithme efficace pour résoudre des problèmes de programmation linéaire

(ou de trouver qu'il n'y a pas de solution)

Beaucoup utilisé et réussit sur les problèmes "réels"

Théoriquement trop lent pour des problèmes très bizarres

Basé sur deux idées importantes:

- solution itérative parcourt très peu des sommets

- calcul astucieux de chaque itération

L'idée de base (géométrique)

- Choisir une solution réalisable de base (sommet du polytope)

- Si la fonction objective a une valeur supérieure à un sommet voisin, aller à ce sommet

- Répéter (tant que possible...)

Pourquoi ça marche?

Chaque arête parcourue augmente la fonction objective: donc pas de boucles

Le nombre de sommets est fini: donc terminaison garantie

Pas de maximaux locaux qui ne sont pas de maximaux globaux: donc l'algorithme ne peut se terminer qu'à une solution optimale (s'il y en a)

Comment est-ce que ça s'arrête?

Aucun sommet voisin avec une valeur supérieure de z

On a trouvé la valeur optimale (si elle existe)

C'est quoi, un sommet voisin?

- Le sommet actuel est déterminé par n équations

- Une arête du polytope qui passe par ce sommet est déterminée par $n-1$ parmi elles

- Un voisin est donnée par les $n-1$ et une nouvelle équation

- Un ensemble de $n-1$ des équations actuelles et une nouvelle peut échouer (aucune solution ou plusieurs solutions ou pas réalisable ou valeur pas améliorée)

- Sur une arête il n'y a que deux solutions de base réalisables

Un très petit exemple de parcours itératif calculé de façon naïve

Deux variables x_1 et x_2
Deux contraintes explicites: $3x_1 + 2x_2 \ \leq \ 30$
$-2x_1 + x_2 \ \leq \ 8$
les 2 contraintes implicites: $x_1, \ x_2 \ \geq \ 0$
$z \ = \ -3x_1 + 2x_2$ à maximiser.

La solution de base $x_1 = x_2 = 0$ est réalisable avec $z = 0$. Les solutions voisines sont obtenues en supprimant une des équations $x_1 = 0$ ou $x_2 = 0$ et en ajoutant une des équations
$3x_1 + 2x_2 \ = \ 30$ ou $-2x_1 + x_2 \ = \ 8$

Donc quatre possibilités:

- $x_1 = 0,\ 3x_1 + 2x_2 = 30$: $x_2 = 15$ n'est pas réalisable.

- $x_1 = 0,\ -2x_1 + x_2 = 8$: $x_2 = 8$ réalisable avec $z = 16$.

- $x_2 = 0,\ 3x_1 + 2x = 30$: $x_1 = 10$ réalisable avec $z = -30$.

- $x_2 = 0,\ -2x_1 + x_2 = 8$: $x_1 = -4$ n'est pas réalisable.

On continue de la solution:
$x_1 = 0, \quad -2x_1 + x_2 \ = \ 8, \qquad z = 16.$

Supprimer une des équations:
$x_1 = 0, \quad -2x_1 + x_2 \ = \ 8$
Ajouter une équation choisie parmi
$x_2 = 0, \quad 3x_1 + 2x \ = \ 30$

En fait on ne trouve qu'une nouvelle solution:
$3x_1 + 2x_2 \ = \ 30, \quad -2x_1 + x_2 \ = \ 8$:
$x_1 = 2$, $x_2 = 12$ réalisable avec $z = 18$, donc une amélioration.

Et ainsi de suite (mais pour ce petit exemple il n'y a plus de possibilités d'amélioration).

Impressionnant?

Non.

On a fait 3 itérations et pour chaque itération on calcule 4 voisins; donc 12 sommets calculés sur les 6 qui existent!

Mais pour un exemple de taille raisonnable c'est mieux:
10 variables et 10 contraintes explicites: 184756 sommets
Si l'algorithme prend 40 itérations (un peu pessimiste), on n'en calcule que $40 \times 10 \times 10 = 4000$.
(Et on va voir que le vrai algorithme fait encore moins de calcul)

Plus la taille du problème est élevée, plus cette méthode gagne par rapport à la méthode très naïve de calculer toutes les solutions de base.

Le “vrai” algorithme du simplexe de Dantzig

Calcul beaucoup plus vite d’un bon voisin

On considère d’abord le cas où $(0, 0, \cdots, 0)$ est une solution de base réalisable et l’algorithme part de là:

On va supprimer une seule des équations $x_i = 0$; donc cette variable x_i va devenir positive et les autres resteront nulles. Donc, il faut choisir une variable à coefficient positif en z.

En choisir une (peut-être mais pas forcément celle au coefficient le plus élevé?)

Chaque contrainte avec un coefficient $a_{i,j} > 0$ donne une borne supérieure sur la nouvelle valeur de x_i;

Choisir la plus petite de ces bornes; c’est la contrainte qu’il faut rajouter.

Comment continuer?

Une astuce algébrique permet de continuer à faire les calculs de la même façon efficace

Transformer le problème en “forme standard”.

C’est la forme où les seules contraintes sont que chaque variable est positive ou nulle et des contraintes dans la forme d’équations. La transformation s’effectue en ajoutant une nouvelle variable “variable d’écart” pour chaque contrainte d’inéquation:

Par exemple $3x_1 + 2x_2 \leq 30$ devient $3x_1 + 2x_2 + x_3 = 30$ où x_3 est la variable d’écart, contrainte d’être positive ou nulle.

- Une solution de base est représentée par m variables (variables de base) qui sont permises d’être non-nulles

- Les variables de base (et la fonction objective) sont données par des combinaisons linéaires des variables hors base

- On choisit une variable qui va entrer dans la base (coefficient positif en z); c'est-à-dire qu'on a choisi une arête

- On choisit une variable qui va sortir de la base (qui minimise la distance parcourue avant de violer la contrainte ≥ 0)

- On recalcule tous les coefficients

Le même exemple

$$
\begin{array}{rrrrl}
3x_1 & +2x_2 & +x_3 & & = 30 \\
-2x_1 & +x_2 & & +x_4 & = 8 \\
-3x_1 & +2x_2 & & & = (z)
\end{array}
$$

$$
\begin{array}{rrrrl}
7x_1 & & +x_3 & -2x_4 & = 14 \\
-2x_1 & +x_2 & & +x_4 & = 8 \\
x_1 & & & -2x_4 & = (z) - 16
\end{array}
$$

$$
\begin{array}{rrrrl}
x_1 & & +x_3/7 & -2x_4/7 & = 2 \\
& +x_2 & +2x_3/7 & +3x_4/7 & = 12 \\
& & -x_3/7 & -12x_4/7 & = (z) - 18
\end{array}
$$

En pratique on garde les coefficients dans un tableau et supprime les colonnes connues ne contenir qu'un élément non nul etc. mais ce ne sont que des détails de programmation.