La méthode du simplexe multicritère

Sanjy Andriamiseza[1], Arthur Gontier[2]
Encadrants : Anthony Przybylski[3], Xavier Gandibleux[4]

12 août 2020

Introduction

Etat actuel : Beaucoup d'algorithmes d'approximation, méthode simplexe multicritère quasiment inexistante sur le terrain

Objectif : Tenter de fournir un solver linéaire multiobjectif de résolution exacte à intégrer dans VOpt Solver

Approche : On réutilisera ce qui existe déjà pour viser toujours la performance

On considère les MOLP en forme standard comme suit :

$$\begin{array}{llll} \text{Min} & C^T x & & \\ \text{s.t.} & Ax & = & \text{b} \\ & x & \geq & 0 \end{array}$$

C et A sont des matrices, x et b sont des vecteurs.

Cas MOLP :

$\bar{c}^1$	1	0	0	2	0	0	2
$\bar{c}^2$	-1	0	2	0	0	0	0
$\bar{c}^3$	1	0	-1	0	0	0	0
x_2	1	1	0	1	0	0	1
x_5	-1	0	0	-1	1	0	1
x_6	2	0	1	1	0	1	5

Similaire à un tableau simplexe comme on le sait mais on aura plusieurs fonctions objectifs pour décrire l'espace objectif.

Rappels de définition

<u>**Solution efficace :**</u>
Une solution est efficace si son image par z dans l'espace objectif est un point non dominé.

<u>**Point non dominé :**</u>
y^1 non dominé est un point réalisable telle qu'il n'existe pas un y qui le domine.

<u>**Base efficace :**</u>
Une base est efficace si la solution qui lui correspond est efficace.

On ne va plus paramétriser !

Mais scalariser reste intéressant puisque toute solution optimale obtenue par une somme pondérée du problème MOLP est une solution efficace (et inversement).

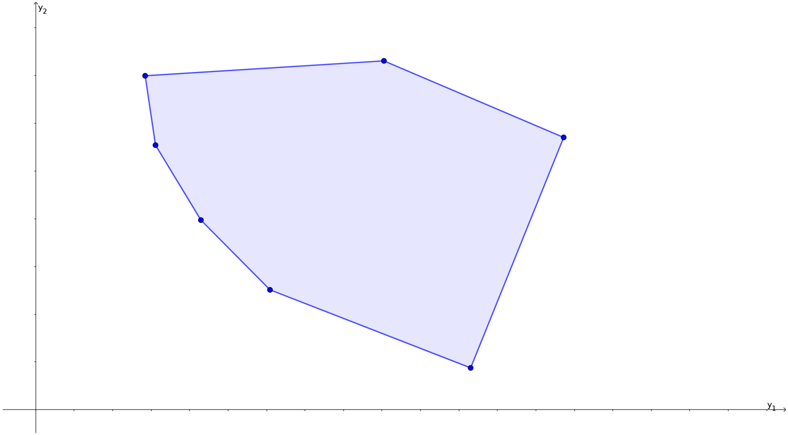
Déjà explicitée auparavant par Charly et Guillaume dans le cadre du simplexe paramétrique en variable bornée.

Phase 1 :
Le MOLP est-il impossible ?

Phase 2 :
Existe t-il au moins une solution efficace ?

Phase 3 :
Quel est l'ensemble des solutions efficaces ?

Supposons un MOLP et son ensemble de points réalisables.

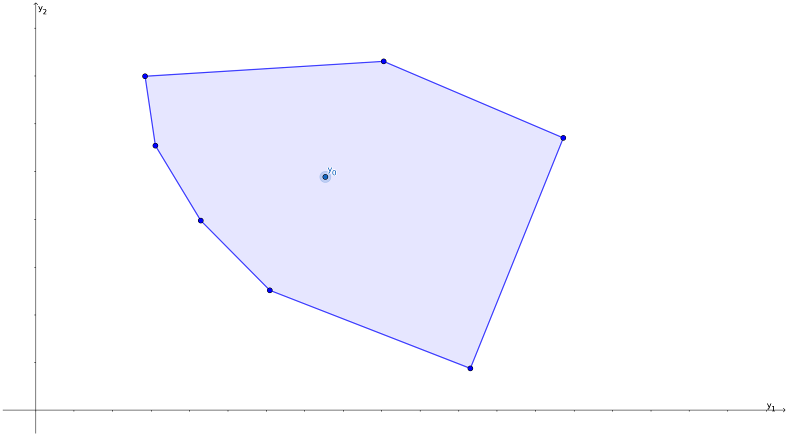


Phase 1 similaire à phase 1 du simplexe mono-objectif

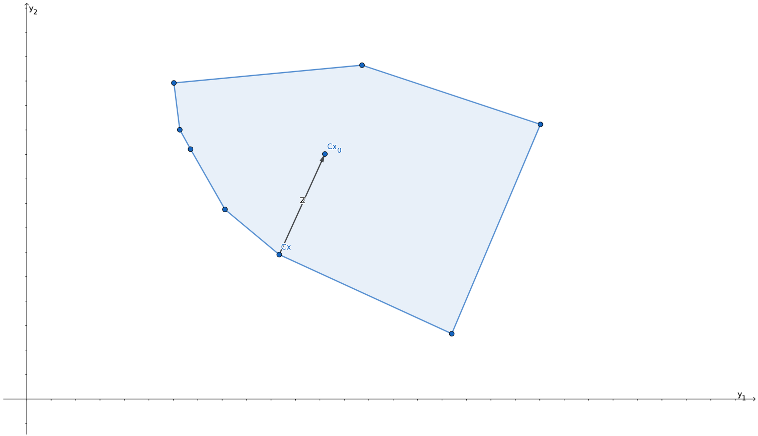
Problème, manipulation par le solver ?

On se permet donc d'annuler les fonctions objectifs pour récupérer une solution réalisable. $min\{0 : Ax = b; x \geq 0\}$

En résolvant un LP où on annule les fonctions objectifs on récupère une première solution admissible x_0 et son image y_0.



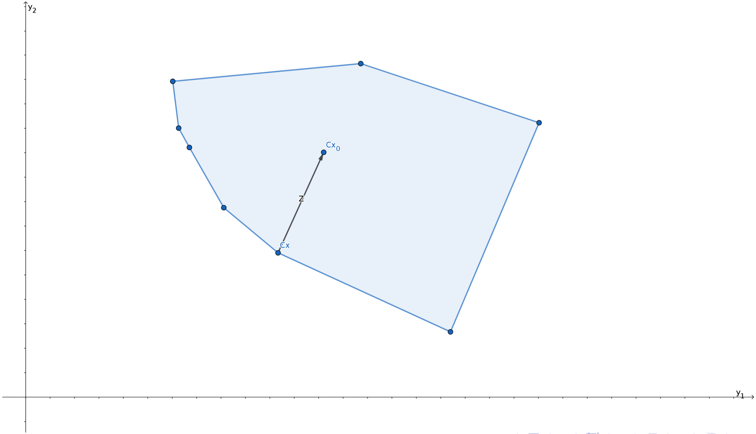
La phase 2 se déroule en 2 étapes : d'abord on va vouloir vérifier l'existence de solutions efficaces. Pour cela on utilise la méthode de benson. $\max\{e^T z, Ax = b; Cx + Iz = Cx^0; x, z \geqq 0\ \}$



Ou plutôt son dual :

$$\underline{min}\{u^T b + w^T C x_0 : u^T A^T + w^T C^T \geq 0; w \geq 1\}$$

qui nous permet de récupérer un poids w strictement positif.

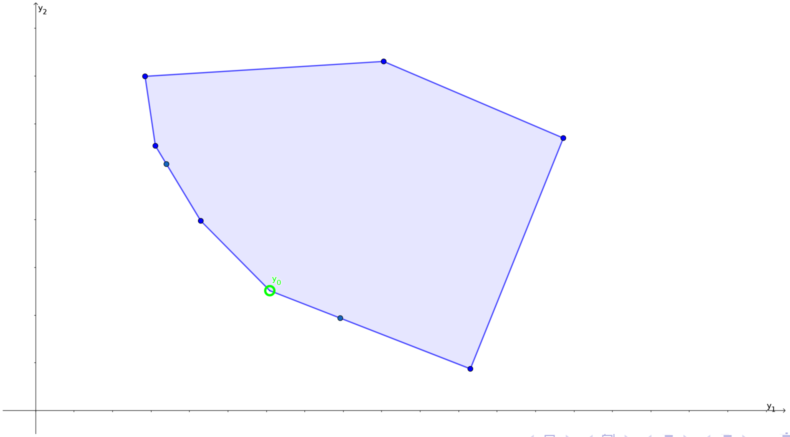


Or on sait que toute solution optimale d'une scalarisation d'un MOLP est une solution efficace !

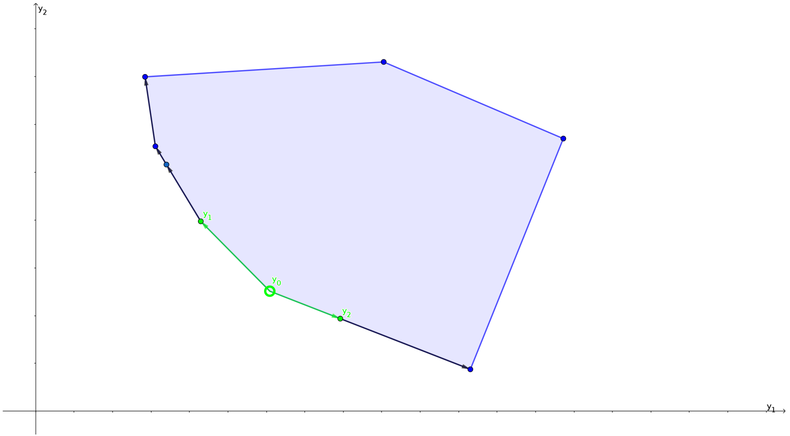
On peut donc simplement résoudre le LP avec les poids trouvés

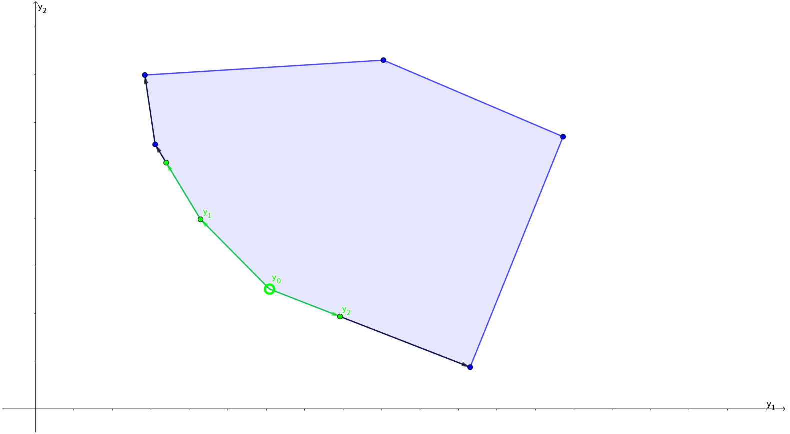
$$min\{w^T Cx : Ax \geq b; x \geq 0\}$$

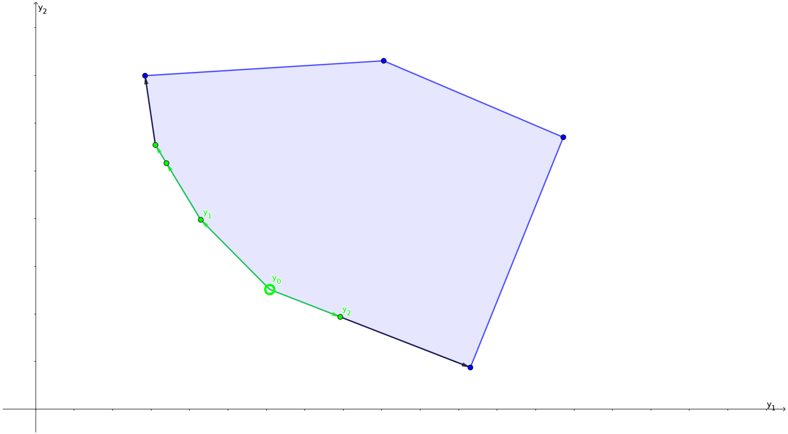
Ainsi on récupère une première solution efficace x_0 et son image.

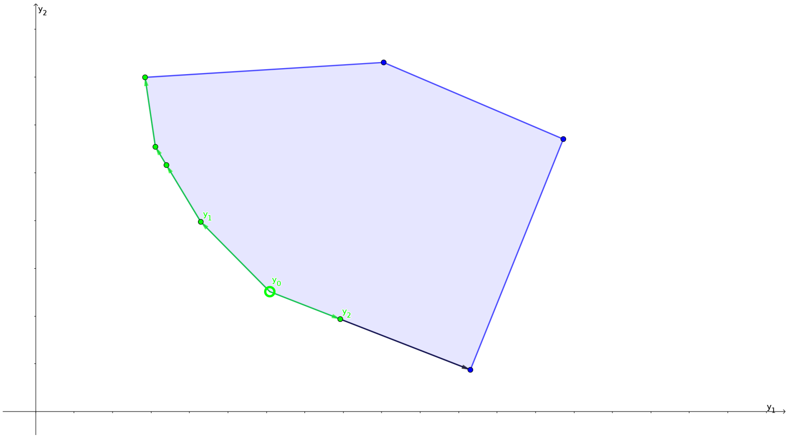


Une fois la première base efficace trouvée, on peut commencer le déplacement de solutions de bases en solutions de bases.

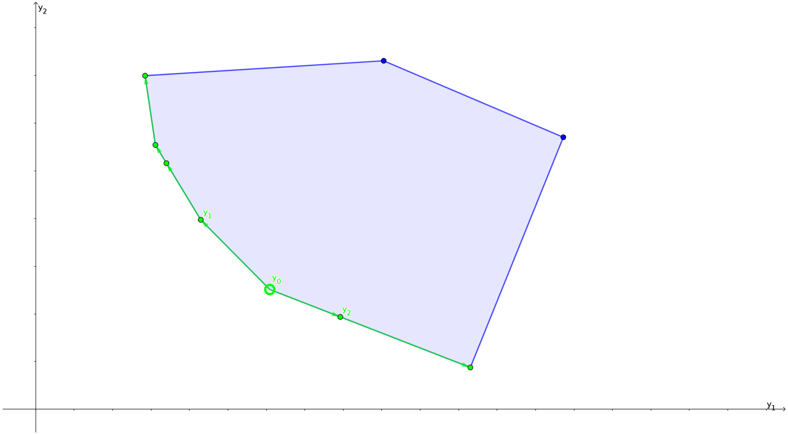



y_2
y_1
y_0
y_1
y_2

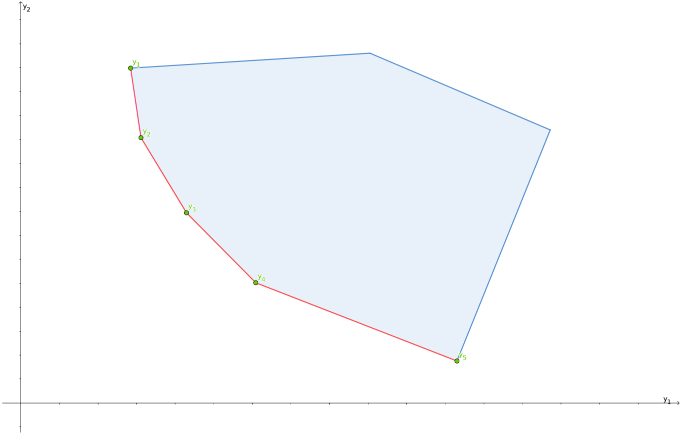

y2
y1
y0
y1
y2



y_2
y_1
y_0
y_1
y_2


y2
y1
y0
y1
y2

On obtient au final l'ensemble des points non dominés qui décrivent la frontière de pareto.



- Pivot efficace $\Longrightarrow$ Programme auxiliaire
- Mécanisme d'exploration $\Longrightarrow$ double boucle
- L'énumération en pratique $\Longrightarrow$ Forme révisé LU et Structures

$\bar{c}^1$	1	0	0	2	0	0	2
$\bar{c}^2$	-1	0	2	0	0	0	0
$\bar{c}^3$	1	0	-1	0	0	0	0
x_2	1	1	0	1	0	0	1
x_5	-1	0	0	-1	1	0	1
x_6	2	0	1	1	0	1	5

$\Longrightarrow$ $R - r^j$: A partir des coûts réduits

$\Longrightarrow$ c_{aux} : Somme des colonne de la matrice $R - r^j$.

On note le problème auxiliaire :

$$min\{c_{aux}^T(y+\delta) : Ry - r^j\delta \geq 0; y, \delta \geq 0\}$$

Pour chaque variable entrante x_j

---

tant que *il reste des bases* **faire**
    pour *chaque variable entrante* x_j **faire**
        $min\{c_{aux}^T(y+\delta) : Ry - r^j\delta \geq 0; y, \delta \leq 0\}$
        si *le* PL_{aux} *est borné,* x_j *est efficace* **alors**
            si *une variable* x_s *peut sortir* **alors**
                $L \leftarrow$ On ajoute la nouvelle base si elle n'est pas déjà vue
            fin
        fin
    fin
fin

Incorporation de la forme révisé LU pour calculer :

- Matrice R
- Solution en base et Variable sortante

tant que *il reste des bases* **faire**
  $L, U \leftarrow$ DécompLU(A_B)
  $R \leftarrow$ CoûtsRéduits(C,L,U)
  $X_B \leftarrow$ Sol$_B$(b,L,U)
  pour *chaque variable entrante* x_j **faire**
    $min\{c_{aux}^T(y+\delta) : Ry - r^j\delta \geq 0; y, \delta \leq 0\}$
    si *le* PL_{aux} *est borné,* x_j *est efficace* **alors**
      $x_s \leftarrow$ VarSortante(A,L,U)
      si *une variable* x_s *peut sortir* **alors**
        $L \leftarrow$ On ajoute la nouvelle base si elle n'est pas déjà vue
      fin
    fin
  fin
fin

Structure des bases et listes de bases :

- Approche ensembliste
- Approche tableau $\mathcal{O}(|B|^2 \times |L|)$
- Approche tableau trié $\mathcal{O}(|B| \times (|L| + log|B|))$
- (Approche matrice inverse, Imprécisions numériques)

Complexité la plus efficace ?

Garde-t-on l'exploration en largeur ?

Instances générées :

- 70% impossibles
- 25% échouent la phase 2
- Sinon rayons infini très fréquents

Générateur et Parser

On peut résoudre des instances de MOLP publiques et disponibles en ligne (format .mop)
http ://moplib.zib.de/

```
julia> main(1)
-----EHRGOTT-----
-P1-
X0 = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
-P2-
u = [2.0, 0.0, 0.0] w = [1.0, 1.0, 1.0]
-P3-
B = [2, 5, 6]
    x1:x2 ==> Nouvelle base
    x3:x6 ==> Nouvelle base
    x4:x2 ==> Inefficient variable
B = [1, 5, 6]
    x2:x1 ==> Base déja vue
    x3:x6 ==> Inefficient variable
    x4:x1 ==> Inefficient variable
B = [2, 5, 3]
    x1:x2 ==> Inefficient variable
    x4:x2 ==> Inefficient variable
    x6:x3 ==> Base déja vue


Bases : Array{Int64,1}[[2, 5, 6], [1, 5, 6], [2, 5, 3]]
Sol associées : Array{Float64,1}[[0.0, 1.0, 0.0, 0.0, 1.0, 5.0], [1.0, 0.0, 0.0,
 0.0, 2.0, 3.0], [0.0, 1.0, 5.0, 0.0, 1.0, 0.0]]
C Trivial

julia> 
```

Conclusion et Perspectives

C'est un sujet très intéressant mais on aimerait aller plus loin car il reste des pistes d'améliorations :

- Chercher un parcourt plus intelligent pour explorer les bases ;
- Recoder décomp LU ; systèmes triangulaires ;
- Jeter jump pour le Prog aux ;
- Incorporer la méthode à V-Opt pour une utilisation plus facile.

Merci pour votre attention !