Programmation linéaire : définitions, algorithme du simplexe

Quentin Fortier

October 28, 2021

Programmation linéaire

Définition

Un programme linéaire est un problème qui consiste à maximiser (ou minimiser) une fonction linéaire avec des contraintes linéaires.

Exemple :

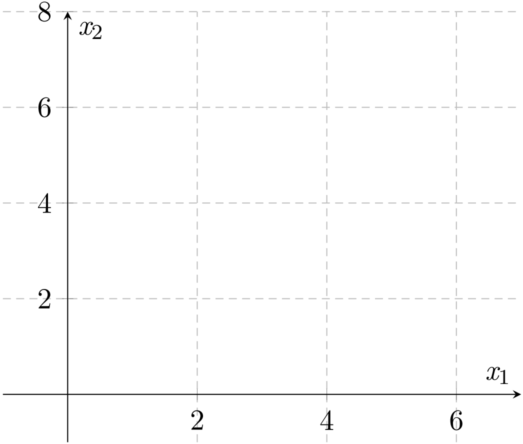
$$\begin{aligned} \max_{x_1, x_2 \in \mathbb{R}} \quad & 4x_1 + 5x_2 \\ \text{t.q} \quad & 2x_1 + x_2 \leq 8 \\ & x_1 + 2x_2 \leq 7 \\ & x_2 \leq 3 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Il faut trouver les valeurs de x_1, x_2 qui maximisent $4x_1 + 5x_2$ en respectant les contraintes.

Programmation linéaire : méthode graphique

$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

$$\begin{aligned} \texttt{t.q} \quad 2x_1 + x_2 &\leq 8 \\ x_1 + 2x_2 &\leq 7 \\ x_2 &\leq 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$

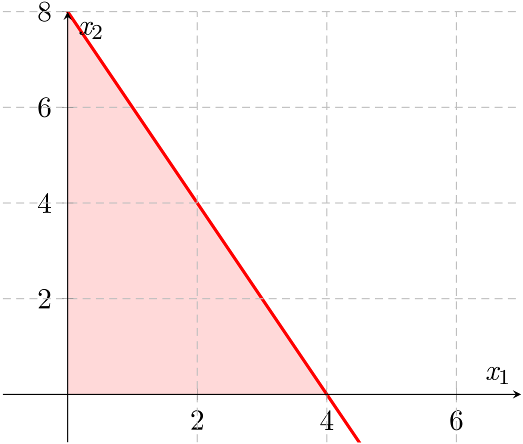


Programmation linéaire : méthode graphique

$$\max_{x_1,x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

$$\text{t.q} \quad 2x_1 + x_2 \leq 8$$

$$x_1, x_2 \geq 0$$



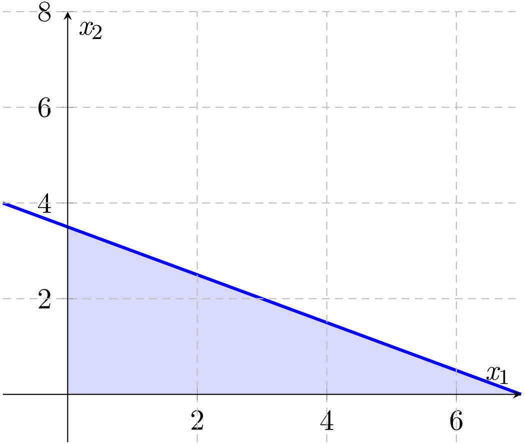
Programmation linéaire : méthode graphique

$$\max_{x_1,x_2\in\mathbb{R}} \quad 4x_1 + 5x_2$$

t.q

$$x_1 + 2x_2 \leq 7$$

$$x_1, x_2 \geq 0$$



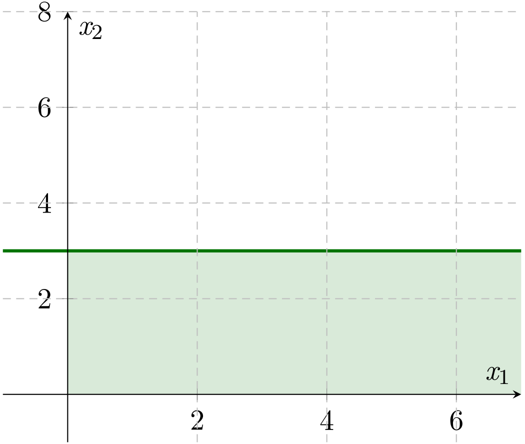
Programmation linéaire : méthode graphique

$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

t.q

$$x_2 \leq 3$$

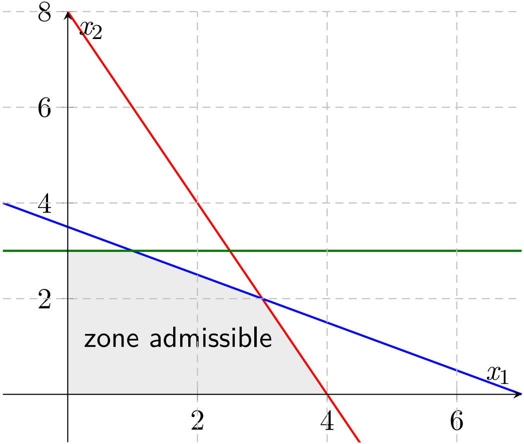
$$x_1, x_2 \geq 0$$



Programmation linéaire : méthode graphique

$$\max_{x_1,x_2\in\mathbb{R}} \quad 4x_1 + 5x_2$$

$$\begin{aligned} \text{t.q} \quad 2x_1 + x_2 &\leq 8 \\ x_1 + 2x_2 &\leq 7 \\ x_2 &\leq 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$

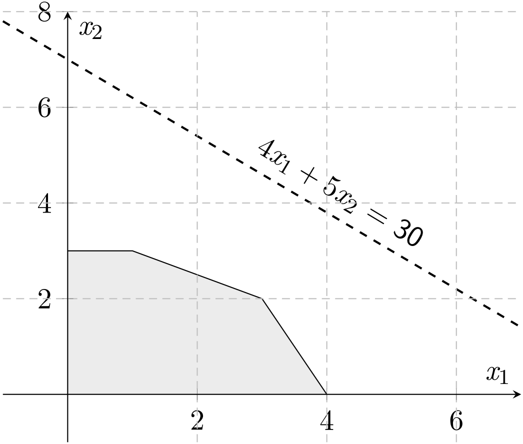


La zone admissible est un **polyhèdre**.

Programmation linéaire : méthode graphique

$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

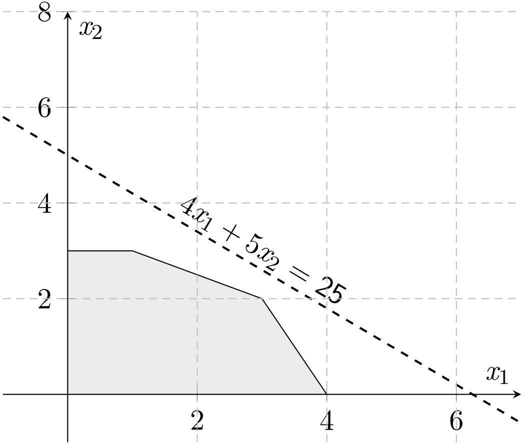
$$\begin{aligned} \texttt{t.q} \quad 2x_1 + x_2 &\leq 8 \\ x_1 + 2x_2 &\leq 7 \\ x_2 &\leq 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$



Programmation linéaire : méthode graphique

$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

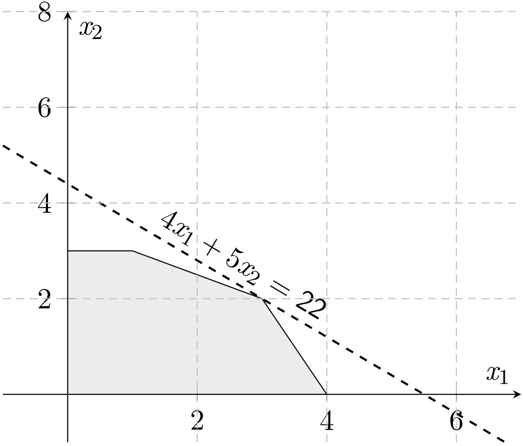
$$\begin{aligned} \texttt{t.q} \quad 2x_1 + x_2 &\leq 8 \\ x_1 + 2x_2 &\leq 7 \\ x_2 &\leq 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$



Programmation linéaire : méthode graphique

$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

$$\begin{aligned} \text{t.q} \quad 2x_1 + x_2 &\leq 8 \\ x_1 + 2x_2 &\leq 7 \\ x_2 &\leq 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$

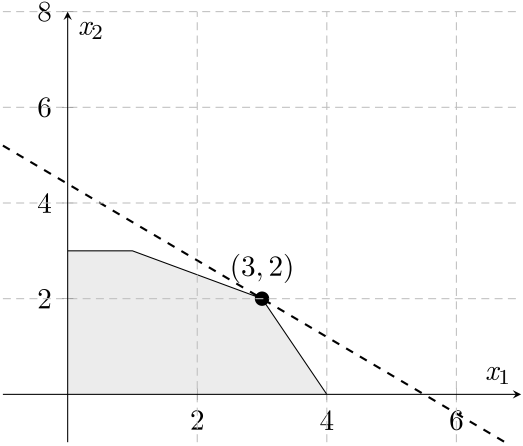


L'optimum est 22, atteint pour $x_1 = 3$ et $x_2 = 2$

Programmation linéaire : méthode graphique

$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

$$\begin{aligned} \text{t.q} \quad 2x_1 + x_2 &\leq 8 \\ x_1 + 2x_2 &\leq 7 \\ x_2 &\leq 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$



L'optimum est 22, atteint pour $x_1 = 3$ et $x_2 = 2$

Programmation linéaire : formulation générale

On peut mettre un programme linéaire sous forme matricielle :

$$\begin{aligned} \max_{x_1,x_2\in\mathbb{R}} \quad & 4x_1 + 5x_2 \\ \text{t.q} \quad & 2x_1 + x_2 \leq 8 \\ & x_1 + 2x_2 \leq 7 \\ & x_2 \leq 3 \\ & x_1, x_2 \geq 0 \end{aligned}$$

$$\begin{aligned} \max_{x\in\mathbb{R}^2} \; & cx \\ \text{t.q} \quad & Ax \leq b \\ & x \geq 0_{\mathbb{R}^2} \end{aligned}$$

avec $c = (4, 5)$, $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, $A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \\ 0 & 1 \end{pmatrix}$, $b = \begin{pmatrix} 8 \\ 7 \\ 3 \end{pmatrix}$

Programmation linéaire : forme canonique et standard

Forme standard

$$\max_{x \in \mathbb{R}^n} cx$$

$$Ax \leq b$$

$$x \geq 0_{\mathbb{R}^n}$$

Forme canonique

$$\max_{x \in \mathbb{R}^n} cx$$

$$Ax = b$$

$$x \geq 0_{\mathbb{R}^n}$$

Programmation linéaire : changer de forme

Changement de forme

- min $\longrightarrow$ max

$$\min z \quad \Longleftrightarrow$$

Programmation linéaire : changer de forme

Changement de forme

- min $\longrightarrow$ max

$$\min z \quad \Longleftrightarrow \quad -\max(-z)$$

- $\leq 0 \longrightarrow \geq 0$

$$x \leq b \quad \Longleftrightarrow$$

Programmation linéaire : changer de forme

Changement de forme

- min $\longrightarrow$ max

$$\min z \quad \Longleftrightarrow \quad -\max(-z)$$

- $\leq 0 \longrightarrow \geq 0$

$$x \leq b \quad \Longleftrightarrow \quad -x \geq b$$

- $\in \mathbb{R} \longrightarrow \geq 0$

$$x \in \mathbb{R} \quad \Longleftrightarrow$$

Programmation linéaire : changer de forme

Changement de forme

- min $\longrightarrow$ max

$$\min z \quad \Longleftrightarrow \quad -\max(-z)$$

- $\leq 0 \longrightarrow \geq 0$

$$x \leq b \quad \Longleftrightarrow \quad -x \geq b$$

- $\in \mathbb{R} \longrightarrow \geq 0$

$$x \in \mathbb{R} \quad \Longleftrightarrow \quad \begin{cases} x = x^+ - x^- \\ x^+, x^- \geq 0 \end{cases}$$

Programmation linéaire : changer de forme

Changement de forme

- égalité $\longrightarrow$ inégalité

$$ax = b \quad \Longleftrightarrow$$

Programmation linéaire : changer de forme

Changement de forme

- égalité $\longrightarrow$ inégalité

$$ax = b \quad \Longleftrightarrow \quad \begin{cases} ax \leq b \\ ax \geq b \end{cases}$$

- inégalité $\longrightarrow$ égalité

$$ax \geq b \quad \Longleftrightarrow$$

Programmation linéaire : changer de forme

Changement de forme

- égalité $\longrightarrow$ inégalité

$$ax = b \quad \Longleftrightarrow \quad \begin{cases} ax \leq b \\ ax \geq b \end{cases}$$

- inégalité $\longrightarrow$ égalité

$$ax \geq b \quad \Longleftrightarrow \quad \begin{cases} ax - s = b \\ s \geq 0 \end{cases}$$

$$ax \leq b \quad \Longleftrightarrow$$

Programmation linéaire : changer de forme

Changement de forme

- égalité $\longrightarrow$ inégalité

$$ax = b \quad \Longleftrightarrow \quad \begin{cases} ax \leq b \\ ax \geq b \end{cases}$$

- inégalité $\longrightarrow$ égalité

$$ax \geq b \quad \Longleftrightarrow \quad \begin{cases} ax - s = b \\ s \geq 0 \end{cases}$$

$$ax \leq b \quad \Longleftrightarrow \quad \begin{cases} ax + s = b \\ s \geq 0 \end{cases}$$

Programmation linéaire : changer de forme

Forme standard

$$\max_{x \in \mathbb{R}^2} cx$$

$$Ax \leq b$$

$$x \geq 0_{\mathbb{R}^2}$$

Forme canonique

$$\max_{x \in \mathbb{R}^2} cx$$

$$Ax = b$$

$$x \geq 0_{\mathbb{R}^2}$$

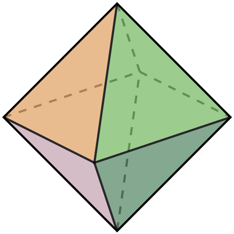
Exercice

Mettre sous forme standard puis canonique le PL suivant :

$$\begin{array}{rrrcl} \min\limits_{x_1, x_2 \in \mathbb{R}} & x_1 & -x_2 & & \\ & 3x_1 & -x_2 & \geq & 2 \\ & x_1 & +2x_2 & \leq & 4 \\ & 4x_1 & -x_2 & = & 8 \\ & x_1 & & \leq & 0 \\ & & x_2 & \geq & 0 \end{array}$$

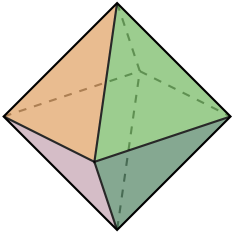
Programmation linéaire : géométrie

L'ensemble des solutions faisables (vérifiant les contraintes) est une intersection de demi-espaces qu'on appelle **polyhèdre**.



Programmation linéaire : géométrie

L'ensemble des solutions faisables (vérifiant les contraintes) est une intersection de demi-espaces qu'on appelle **polyhèdre**.



Les lignes de niveaux de la foncion objectif z sont des **hyperplans affines**.
L'optimum est obtenu en prenant la plus grande ligne de niveau qui intersecte le polyhèdre. D'où :

Théorème

La valeur maximum de la fonction objectif est obtenu en un sommet du polyhèdre des solutions admissibles.

Programmation linéaire : géométrie

Théorème

La valeur maximum de la fonction objectif est obtenu en un sommet du polyhèdre des solutions admissibles.

$$\max_{x \in \mathbb{R}^n} cx$$

$$Ax \leq b$$

$$x \geq 0_{\mathbb{R}^n}$$

Dans un espace de dimension n, l'intersection de k hyperplans linéairement indépendants est de dimension $n - k$.

Programmation linéaire : géométrie

Théorème

La valeur maximum de la fonction objectif est obtenu en un sommet du polyhèdre des solutions admissibles.

$$\max_{x \in \mathbb{R}^n} cx$$

$$Ax \leq b$$

$$x \geq 0_{\mathbb{R}^n}$$

Dans un espace de dimension n, l'intersection de k hyperplans linéairement indépendants est de dimension $n - k$.
Donc les sommets du polyhèdre sont obtenus lorsque n des p contraintes sont *tight*, c'est-à-dire avec $=$ au lieu de $\leq$.

Programmation linéaire : géométrie

$$\max_{x \in \mathbb{R}^n} cx$$

$$Ax \leq b$$

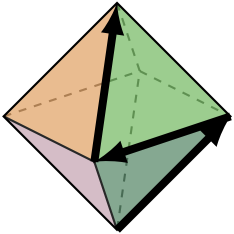
$$x \geq 0_{\mathbb{R}^n}$$

Dans un espace de dimension n, l'intersection de k hyperplans linéairement indépendants est de dimension $n - k$.
Donc les sommets du polyhèdre sont obtenus lorsque n des p contraintes sont *tight*, c'est-à-dire avec $=$ au lieu de $\leq$.

Il serait donc possible d'énumérer les $\binom{p}{n}$ sommets obtenus en fixant des $\leq$ à $=$ et conserver le maximum, mais ceci donne un algo. exponentiel.

Algorithme du simplexe

L'algorithme du simplexe est une méthode très utilisée pour résoudre un programme linéaire.
Il consiste à se déplacer de sommet en sommet sur le polyhèdre des solutions admissible en augmentant à chaque fois la fonction objectif.
Quand ce n'est plus possible de l'améliorer, une solution optimale est trouvée.



Algorithme du simplexe : *slack variables*

On suppose que le programme est donné sous forme standard.
On commence par le mettre sous forme canonique en ajoutant des *slack variables* (variables d'écart).

$$
\begin{aligned}
\max_{x_1, x_2 \in \mathbb{R}} \quad & 4x_1 + 5x_2 \\
2x_1 + x_2 &\leq 8 \\
x_1 + 2x_2 &\leq 7 \\
x_2 &\leq 3 \\
x_1, x_2 &\geq 0
\end{aligned}
$$

$$
\begin{aligned}
\max_{x_1, x_2 \in \mathbb{R}} \quad & 4x_1 + 5x_2 \\
2x_1 + x_2 + e_1 &= 8 \\
x_1 + 2x_2 + e_2 &= 7 \\
x_2 + e_3 &= 3 \\
x_1, x_2, e_1, e_2, e_3 &\geq 0
\end{aligned}
$$

Algorithme du simplexe : *slack variables*

$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

$$\begin{aligned} 2x_1 + x_2 + e_1 &= 8 \\ x_1 + 2x_2 + e_2 &= 7 \\ x_2 + e_3 &= 3 \\ x_1, x_2, e_1, e_2, e_3 &\geq 0 \end{aligned}$$

Les sommets du polyhèdre sont obtenus en rendant 2 contraintes *tight*, c'est-à-dire en mettant 2 variables à 0.

Algorithme du simplexe : *slack variables*

$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

$$\begin{aligned} 2x_1 + x_2 + e_1 &= 8 \\ x_1 + 2x_2 + e_2 &= 7 \\ x_2 + e_3 &= 3 \\ x_1, x_2, e_1, e_2, e_3 &\geq 0 \end{aligned}$$

Les sommets du polyhèdre sont obtenus en rendant 2 contraintes *tight*, c'est-à-dire en mettant 2 variables à 0.
Le plus simple est de choisir $x_1 = 0$ et $x_2 = 0$, ce qui nous donne $e_1 = 8,\ e_2 = 7,\ e_3 = 3$.
Comme toutes les équations sont vérifiées, on obtient une **base admissible** (sommet du polyhèdre) :

$$(x_1, x_2, e_1, e_2, e_3) = (0, 0, 8, 7, 3)$$

Algorithme du simplexe : *slack variables*

$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

$$\begin{aligned} 2x_1 + x_2 + e_1 &= 8 \\ x_1 + 2x_2 + e_2 &= 7 \\ x_2 + e_3 &= 3 \\ x_1, x_2, e_1, e_2, e_3 &\geq 0 \end{aligned}$$

Base admissible :

$$(x_1, x_2, e_1, e_2, e_3) = (0, 0, 8, 7, 3)$$

On dit que e_1, e_2, e_3 sont les variables dans la base et x_1, x_2 les variables hors-base.
Tout au long de l'algorithme, on exprime les variables dans la base en fonction des variables hors-base.

Algorithme du simplexe : *slack variables*

On exprime les variables dans la base en fonction des variables hors-base :

$$
\begin{aligned}
\max \quad & 4x_1 + 5x_2 \\
& e_1 = 8 - 2x_1 - x_2 \\
& e_2 = 7 - x_1 - 2x_2 \\
& e_3 = 3 - x_2 \\
& x_1, x_2, e_1, e_2, e_3 \geq 0
\end{aligned}
$$

On essaie ensuite de faire rentrer une variable hors-base en sortant une variable. Pour faire augmenter la fonction objectif, il faut que ce soit une variable ayant un coefficient positif dans la fonction objectif.

Algorithme du simplexe : itérations

Faisons rentrer x_2. De combien peut-on augmenter x_2 tout en ayant $e_1, e_2, e_3 \geq 0$?

$$\begin{aligned} \max \quad & 4x_1 + 5x_2 \\ & e_1 = 8 - 2x_1 - x_2 \geq 0 \\ & e_2 = 7 - x_1 - 2x_2 \geq 0 \\ & e_3 = 3 - x_2 \geq 0 \\ & x_1, x_2, e_1, e_2, e_3 \geq 0 \end{aligned}$$

Algorithme du simplexe : itérations

Faisons rentrer x_2. De combien peut-on augmenter x_2 tout en ayant $e_1, e_2, e_3 \geq 0$?

$$\begin{aligned}
\max \quad & 4x_1 + 5x_2 \\
& e_1 = 8 - 2x_1 - x_2 \geq 0 \implies x_2 \leq 8 \\
& e_2 = 7 - x_1 - 2x_2 \geq 0 \implies x_2 \leq \frac{7}{2} = 3.5 \\
& e_3 = 3 - x_2 \geq 0 \qquad\qquad \implies x_2 \leq 3 \\
& x_1, x_2, e_1, e_2, e_3 \geq 0
\end{aligned}$$

La plus grande valeur que l'on peut donner à x_2 est 3, ce qui met e_3 à 0 (donc le sort de la base).

Algorithme du simplexe : itérations

On sort e_3 et on rentre $x_2 = 3 - e_3$. La base est maintenant (e_1, e_2, x_2).

$$\begin{aligned}
\max \quad & 4x_1 + 5(3 - e_3) = 15 + 4x_1 - 5e_3 \\
& e_1 = 8 - 2x_1 - (3 - e_3) = 5 - 2x_1 + e_3 \\
& e_2 = 7 - x_1 - 2(3 - e_3) = 1 - x_1 + 2e_3 \\
& x_2 = 3 - e_3 \\
& x_1, x_2, e_1, e_2, e_3 \geq 0
\end{aligned}$$

Algorithme du simplexe : itérations

On sort e_3 et on rentre $x_2 = 3 - e_3$. La base est maintenant (e_1, e_2, x_2).

$$\begin{aligned}
\max \quad & 15 + 4x_1 - 5e_3 \\
e_1 &= 5 - 2x_1 + e_3 \\
e_2 &= 1 - x_1 + 2e_3 \\
x_2 &= 3 - e_3 \\
x_1, x_2&, e_1, e_2, e_3 \geq 0
\end{aligned}$$

Algorithme du simplexe : itérations

On sort e_3 et on rentre $x_2 = 3 - e_3$. La base est maintenant (e_1, e_2, x_2).

$$
\begin{aligned}
\max \quad & 15 + 4x_1 - 5e_3 \\
& e_1 = 5 - 2x_1 + e_3 \\
& e_2 = 1 - x_1 + 2e_3 \\
& x_2 = 3 - e_3 \\
& x_1, x_2, e_1, e_2, e_3 \geq 0
\end{aligned}
$$

On peut faire rentrer x_1 et sortir e_2 (en augmentant x_1 de 1)

$$e_2 = 1 - x_1 + 2e_3 \implies x_1 = 1 - e_2 + 2e_3$$

Algorithme du simplexe : itérations

On sort e_2 et on rentre $x_1 = 1 - e_2 - 2e_3$. La base est maintenant (e_1, x_1, e_2).

$$
\begin{aligned}
\max \quad & 15 + 4(1 - e_2 + 2e_3) - 5e_3 = 19 - 4e_2 + 3e_3 \\
& e_1 = 5 - 2(1 - e_2 + 2e_3) = 3 + 2e_2 - 3e_3 \\
& x_1 = 1 - e_2 + 2e_3 \\
& x_2 = 3 - e_3 \\
& x_1, x_2, e_1, e_2, e_3 \geq 0
\end{aligned}
$$

Algorithme du simplexe : itérations

On sort e_2 et on rentre $x_1 = 1 - e_2 - 2e_3$. La base est maintenant (e_1, x_1, e_2).

$$
\begin{aligned}
\max \quad & 19 - 4e_2 + 3e_3 \\
e_1 &= 3 + 2e_2 - 3e_3 \\
x_1 &= 1 - e_2 + 2e_3 \\
x_2 &= 3 - e_3 \\
x_1, x_2&, e_1, e_2, e_3 \geq 0
\end{aligned}
$$

Algorithme du simplexe : itérations

On sort e_2 et on rentre $x_1 = 1 - e_2 - 2e_3$. La base est maintenant (e_1, x_1, e_2).

$$\begin{aligned} \max \quad & 19 - 4e_2 + 3e_3 \\ e_1 &= 3 + 2e_2 - 3e_3 \\ x_1 &= 1 - e_2 + 2e_3 \\ x_2 &= 3 - e_3 \\ x_1, x_2, & e_1, e_2, e_3 \geq 0 \end{aligned}$$

On peut faire rentrer e_3. $x_1 = 1 - e_2 + 2e_3 \Longrightarrow 2e_3 \geq -1$ ne donne pas de contrainte sur e_3.
La valeur maximum qu'on peut donner à e_3 est 1, et on sort e_1.

Algorithme du simplexe : itérations

On sort e_1 et on rentre $e_3 = 1 + \frac{2}{3}e_2 - \frac{1}{3}e_1$. La base est maintenant (e_3, x_1, x_2).

$$\begin{aligned}
\max \quad & 19 - 4e_2 + 3(1 + \frac{2}{3}e_2 - \frac{1}{3}e_1) = 22 - e_1 - 2e_2 \\
& e_3 = 1 + \frac{2}{3}e_2 - \frac{1}{3}e_1 \\
& x_1 = 1 - e_2 + 2(1 + \frac{2}{3}e_2 - \frac{1}{3}e_1) = 3 - \frac{1}{3}e_2 - \frac{2}{3}e_1 \\
& x_2 = 3 - (1 + \frac{2}{3}e_2 - \frac{1}{3}e_1) = 2 - \frac{2}{3}e_2 + \frac{1}{3}e_1 \\
& x_1, x_2, e_1, e_2, e_3 \geq 0
\end{aligned}$$

Algorithme du simplexe : itérations

La base est maintenant (e_3, x_1, x_2).

$$
\begin{aligned}
\max \quad & 22 - e_1 - 2e_2 \\
e_3 &= 1 + \frac{2}{3}e_2 - \frac{1}{3}e_1 \\
x_1 &= 3 - \frac{1}{3}e_2 - \frac{2}{3}e_1 \\
x_2 &= 2 - \frac{2}{3}e_2 + \frac{1}{3}e_1 \\
& x_1, x_2, e_1, e_2, e_3 \geq 0
\end{aligned}
$$

Algorithme du simplexe : itérations

La base est maintenant (e_3, x_1, x_2).

$$\begin{aligned}
\max \quad & 22 - e_1 - 2e_2 \\
e_3 &= 1 + \frac{2}{3}e_2 - \frac{1}{3}e_1 \\
x_1 &= 3 - \frac{1}{3}e_2 - \frac{2}{3}e_1 \\
x_2 &= 2 - \frac{2}{3}e_2 + \frac{1}{3}e_1 \\
& x_1, x_2, e_1, e_2, e_3 \geq 0
\end{aligned}$$

Il n'est pas possible d'augmenter la fonction objectif donc on a trouvé l'optimal, pour $e_1 = e_2 = 0$ (variables hors-base) et :

$$(e_3, x_1, x_2) = (1, 3, 2)$$

Algorithme du simplexe : itérations

La solution $(e_1, e_2, e_3, x_1, x_2) = (0, 0, 1, 3, 2)$ donne $\boxed{(x_1, x_2) = (3, 2)}$ avec une valeur objectif $\boxed{4x_1 + 5x_2 = 22}$.

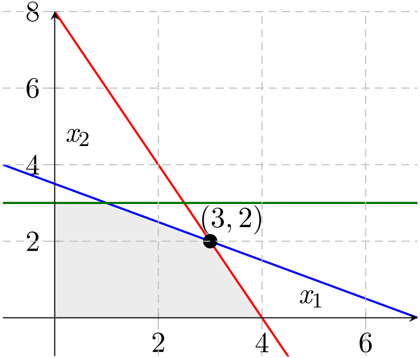
$$\max_{x_1, x_2 \in \mathbb{R}} \quad 4x_1 + 5x_2$$

$$\text{t.q} \quad 2x_1 + x_2 \leq 8$$

$$x_1 + 2x_2 \leq 7$$

$$x_2 \leq 3$$

$$x_1, x_2 \geq 0$$



Algorithme du simplexe : itérations

On considère un boulanger qui possède 2 types de farines : farine de blé et farine de d'avoine.
Il peut produire 2 types de baguettes qui demandent différentes quantités de farine :

	Blé	Avoine	Seigle	Prix
Baguette 1	100g	100g	100g	1,5€
Baguette 2	200g	50g	0g	2€
Stock	10kg	4kg	3kg	

Exercice

Combien le boulanger pourra t-il gagner au maximum ?
Modéliser le problème, et le résoudre avec la méthode du simplexe puis la méthode graphique.

Solveur de programmation linéaire

Il existe beaucoup de solveurs :

- **GLPK** : kit de programmation linéaire open-source, en C GNU
- **COIN-OR** : ensemble de logiciels open-source pour la RO
- **OR-Tools** avec Glop par Google : open-source
- **CPLEX** par IBM : solveur propriétaire
- **Gurobi** : solveur propriétaire, gratuit pour un usage académique
- **XPRESS** : solveur propriétaire
- **Knitro** : solveur propriétaire, surtout pour l'optimisation non-linéaire

Solveur de programmation linéaire

python-mip fait partie de COIN-OR et permet de résoudre simplement un PL en Python :

Exemple avec Binder

Simplexe à 2 phases

Parfois, il n'est pas évident de trouver une première base admissible.

$$\begin{aligned} \max \quad & -x + y \\ x + y &\geq 1 \\ 3x + 2y &= 6 \\ x, y &\geq 0 \end{aligned}$$

Simplexe à 2 phases

On commene par mettre sous forme canonique :

$$\begin{aligned} \max \quad -x + y \\ x + y - e_1 = 1 \\ 3x + 2y = 6 \\ x, y, e_1 \geq 0 \end{aligned}$$

Ici il n'y a pas de première solution évidente.

Question

Comment trouver une première base admissible dans le cas général ?

Simplexe à 2 phases

La méthode des 2 phases consiste à créer une première base admissible évidente en ajoutant des **variables artificielles** :

$$
\begin{aligned}
\max \quad & -x + y \\
x + y - e_1 + \boldsymbol{a_1} &= 1 \\
3x + 2y + \boldsymbol{a_2} &= 6 \\
x, y, e_1, a_1, a_2 &\geq 0
\end{aligned}
$$

Simplexe à 2 phases

La méthode des 2 phases consiste à créer une première base admissible évidente en ajoutant des **variables artificielles** :

$$\begin{aligned} \max \quad & -x + y \\ x + y - e_1 + \boldsymbol{a_1} &= 1 \\ 3x + 2y + \boldsymbol{a_2} &= 6 \\ x, y, e_1, a_1, a_2 &\geq 0 \end{aligned}$$

Alors $(x, y, e_1, a_1, a_2) = (0, 0, 0, 1, 6)$ est une première base admissible.

Simplexe à 2 phases

La méthode des 2 phases consiste à créer une première base admissible évidente en ajoutant des **variables artificielles** :

$$\begin{aligned} \max \quad & -x+y \\ x+y-e_1+\boldsymbol{a_1} &= 1 \\ 3x+2y+\boldsymbol{a_2} &= 6 \\ x, y, e_1, a_1, a_2 &\geq 0 \end{aligned}$$

Alors $(x, y, e_1, a_1, a_2) = (0, 0, 0, 1, 6)$ est une première base admissible. Mais l'optimum de ce nouveau PL n'est pas forcément égal à l'optimum du PL initial...

Simplexe à 2 phases

On change la fonction objectif, pour minimiser la somme des variables artificielles :

$$
\begin{aligned}
\min\ a_1 + a_2 \\
x + y - e_1 + a_1 = 1 \\
3x + 2y + a_2 = 6 \\
x, y, e_1, a_1, a_2 \geq 0
\end{aligned}
$$

On applique l'algorithme du simplexe sur ce nouveau PL.

Simplexe à 2 phases

On change la fonction objectif, pour minimiser la somme des variables artificielles :

$$\begin{gathered} \min \ a_1 + a_2 \\ x + y - e_1 + a_1 = 1 \\ 3x + 2y + a_2 = 6 \\ x, y, e_1, a_1, a_2 \geq 0 \end{gathered}$$

On applique l'algorithme du simplexe sur ce nouveau PL.
Si on trouve un optimum de valeur 0 obtenu en (x, y, e_1, a_1, a_2), on a donc $a_1 = a_2 = 0$ et (x, y, e_1) est une première base admissible pour démarrer un 2ème simplexe.

Simplexe à 2 phases

Exercice

Résoudre complètement le PL suivant avec la méthode à 2 phases.

$$
\begin{aligned}
\max \quad & -x + y \\
x + y - e_1 &= 1 \\
3x + 2y &= 6 \\
x, y, e_1 &\geq 0
\end{aligned}
$$

Méthode du simplexe : résumé

1. Mettre le PL sous forme canonique : $\max z,\ Ax = b$.

Méthode du simplexe : résumé

1. Mettre le PL sous forme canonique : $\max z$, $Ax = b$.
2. Trouver une première base admissible :

Méthode du simplexe : résumé

1. Mettre le PL sous forme canonique : $\max z$, $Ax = b$.
2. Trouver une première base admissible :
    - Dans le cas où le PL était initialement sous forme standard $Ax \leq b$ avec $b \geq 0$, le fait de le transformer sous forme canonique avec des variables d'écart $e = (e_1, ..., e_p)$ donne automatiquement une première base admissible avec $x = 0$ et $e = b$.

Méthode du simplexe : résumé

1. Mettre le PL sous forme canonique : $\max z$, $Ax = b$.
2. Trouver une première base admissible :
   - Dans le cas où le PL était initialement sous forme standard $Ax \leq b$ avec $b \geq 0$, le fait de le transformer sous forme canonique avec des variables d'écart $e = (e_1, ..., e_p)$ donne automatiquement une première base admissible avec $x = 0$ et $e = b$.
   - Sinon, utiliser la 1ère phase de la méthode à 2 phases.
3. Résoudre le PL :

Méthode du simplexe : résumé

1. Mettre le PL sous forme canonique : $\max z,\ Ax = b$.
2. Trouver une première base admissible :
   - Dans le cas où le PL était initialement sous forme standard $Ax \leq b$ avec $b \geq 0$, le fait de le transformer sous forme canonique avec des variables d'écart $e = (e_1, ..., e_p)$ donne automatiquement une première base admissible avec $x = 0$ et $e = b$.
   - Sinon, utiliser la 1ère phase de la méthode à 2 phases.
3. Résoudre le PL :
   - Considérer une variable hors-base x (un **pivot**) de coeff > 0 dans z
   - Échanger x avec une variable y dans la base de façon à mettre y à 0
   - Répéter les 2 opérations précédentes jusqu'à ce que tous les coeffs de z soient ≤ 0

Méthode du simplexe : terminaison

Problème : la méthode du simplexe peut ne pas terminer en cyclant sur les mêmes sommets.

Méthode du simplexe : terminaison

Problème : la méthode du simplexe peut ne pas terminer en cyclant sur les mêmes sommets.

Une solution possible avec la **règle de Bland** : fixer un ordre sur les variables et lorsque l'on a plusieurs choix possibles de pivot, choisir la première dans l'ordre alphabétique.

Méthode du simplexe : complexité

On peut montrer qu'il existe un choix de pivots pour l'algorithme du simplexe qui donne une complexité exponentielle (en le nombre de contraintes).

Méthode du simplexe : complexité

On peut montrer qu'il existe un choix de pivots pour l'algorithme du simplexe qui donne une complexité exponentielle (en le nombre de contraintes).

Par contre, c'est un problème ouvert de savoir s'il existe toujours un choix de pivots qui donne l'optimum en un nombre polynomial d'étapes.

Méthode du simplexe : complexité

On peut montrer qu'il existe un choix de pivots pour l'algorithme du simplexe qui donne une complexité exponentielle (en le nombre de contraintes).

Par contre, c'est un problème ouvert de savoir s'il existe toujours un choix de pivots qui donne l'optimum en un nombre polynomial d'étapes.

Il est démontré que la **méthode de l'ellipsoide** permet de résoudre un PL en complexité polynomial. En pratique, il est plus lent que l'algorithme du simplexe.

Application : flot maximum

Exercice

Soit $\overrightarrow{G} = (V, \overrightarrow{E})$ un graphe orienté avec une capacité c sur les arcs. Mettre le problème de recherche de flot maximum dans $\overrightarrow{G}$ sous forme d'un programme linéaire.

Application : flot maximum

Exercice

Soit $\vec{G} = (V, \vec{E})$ un graphe orienté avec une capacité c sur les arcs. Mettre le problème de recherche de flot maximum dans $\vec{G}$ sous forme d'un programme linéaire.

On définit une variable f_{uv} sur chaque arête $\{u, v\}$.

$$\max \sum_{\{s,v\} \in s^+} f_{sv}$$

$$\sum_{\{u,v\} \in v^-} f_{uv} - \sum_{\{v,w\} \in v^*} f_{vw} = 0, \quad \forall v \in V$$

$$0 \leq f_{uv} \leq c(u, v)$$

Exemple avec Binder