Programmation Linéaire (PL)

Alix Danvy

25 octobre 2025

Table des matières

1 Introductio à la PL

1.1 Définition et Composants

Définition 1 Programme Linéaire (PL)

Un programme linéaire est un problème d'optimisation mathématique qui consiste à maximiser ou minimiser une **fonction objectif linéaire** sous un ensemble de **contraintes linéaires** (équations ou inéquations).

Les trois composants essentiels d'un PL sont :

— **Les variables de décision** (x_i) : Ce sont les inconnues du problème, les quantités que l'on cherche à déterminer (ex : quantité de produit à fabriquer, allocation de ressources, etc.). On suppose en général que $x_i \geq 0$.

— **La fonction objectif** (z) : C'est une expression linéaire des variables de décision que l'on cherche à optimiser (maximiser un profit ou minimiser un coût).

$$z = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$$

— **Les contraintes** : Ce sont des équations ou inéquations linéaires qui limitent les valeurs possibles des variables de décision. Elles représentent les limitations réelles du problème (ex : temps machine disponible, budget, demande du marché, etc.).

$$a_{i1} x_1 + a_{i2} x_2 + \cdots + a_{in} x_n \leq b_i \quad (\text{ou } \geq, =)$$

1.2 Exemple de Modélisation

Exemple L'usine de produits

Une entreprise fabrique deux produits, P1 et P2.

— Le profit pour P1 est de 4€, pour P2 il est de 8€.
— La fabrication nécessite du temps sur deux machines, A et B.
— Machine A : 1h pour P1, 1h pour P2. Disponibilité totale : 10h.
— Machine B : 2h pour P1, 6h pour P2. Disponibilité totale : 48h.

Objectif : Maximiser le profit total.

Modélisation

1. **Variables de décision** :

— x_1 : quantité de produit P1 fabriquée.
— x_2 : quantité de produit P2 fabriquée.

2. **Fonction objectif** : Maximiser le profit z.

$$\text{Maximiser} \quad z = 4x_1 + 8x_2$$

3. **Contraintes** :
   — Contrainte Machine A : $1x_1 + 1x_2 \leq 10$
   — Contrainte Machine B : $2x_1 + 6x_2 \leq 48$
   — Contraintes de non-négativité : $x_1 \geq 0, x_2 \geq 0$

1.3 Forme Matricielle d'un PL

Tout programme linéaire peut être écrit de manière compacte sous forme matricielle.
Pour un problème de maximisation (de forme canonique) :

$$\max c^T x \quad \text{sujet à} \quad Ax \leq b \quad \text{et} \quad x \geq 0$$

Où :

— $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ est le vecteur des **variables de décision**.

— $c = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}$ est le vecteur des **coûts** (ou profits).

— $b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$ est le vecteur des **seconds membres** (ressources).

— A est la **matrice des coefficients** des contraintes (m lignes, n colonnes).

Exemple Forme matricielle de "L'usine de produits"

Le PL de l'exemple 1 s'écrit matriciellement :

$$\text{Maximiser} \quad z = \begin{pmatrix} 4 & 8 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$\text{Sujet à} \quad \begin{pmatrix} 1 & 1 \\ 2 & 6 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 10 \\ 48 \end{pmatrix}$$

$$\text{et} \quad \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Avec $c = \begin{pmatrix} 4 \\ 8 \end{pmatrix}$, $A = \begin{pmatrix} 1 & 1 \\ 2 & 6 \end{pmatrix}$, $b = \begin{pmatrix} 10 \\ 48 \end{pmatrix}$ et $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$.

2 Formes Canonique et Standard

2.1 Définition des Formes

Un PL peut être exprimé sous différentes formes. Les deux plus importantes sont les formes canonique et standard.

Définition 2 Forme Canonique

C'est la forme la plus "naturelle" pour la théorie de la dualité. Elle existe en deux versions :

1. **Forme Canonique (Maximisation)** :

$$\max c^T x \quad \text{sujet à} \quad Ax \leq b \quad \text{et} \quad x \geq 0$$

(Un objectif de maximisation avec uniquement des contraintes $\leq$.)

2. **Forme Canonique (Minimisation)** :

$$\min c^T x \quad \text{sujet à} \quad Ax \geq b \quad \text{et} \quad x \geq 0$$

(Un objectif de minimisation avec uniquement des contraintes $\geq$.)

Définition 3 Forme Standard

C'est la forme requise pour l'application de l'algorithme du simplexe.

$$\max \text{ (ou min) } c^T x \quad \text{sujet à} \quad Ax = b \quad \text{et} \quad x \geq 0$$

Elle doit impérativement respecter trois conditions :

1. Toutes les contraintes sont des **égalités**.
2. Tous les seconds membres b_i sont **positifs ou nuls** ($b_i \geq 0$).
3. Toutes les variables x_j sont **positives ou nulles** ($x_j \geq 0$).

Remarques Intuition des formes

— La **forme canonique** est "pure". Elle est parfaite pour la théorie et la construction du dual, car elle oppose directement les ressources (b) aux profits (c).

— La **forme standard** est "prête pour le calcul". L'algorithme du simplexe travaille sur des systèmes d'équations (des points, et non des zones), d'où la nécessité de transformer les inéquations en équations.

2.2 Règles de Conversion

On peut toujours passer d'un PL quelconque à l'une de ces formes en utilisant les transformations suivantes :

Proposition 1 Transformations d'un PL

1. **Objectif** : Changer un 'min' en 'max' (et vice-versa).

$$\min z \iff \max(-z)$$

2. **Sens d'une inégalité** : Multiplier toute la ligne par -1.

$$a_1x_1 + a_2x_2 \geq b \iff -a_1x_1 - a_2x_2 \leq -b$$

3. **Contrainte d'égalité** : Remplacer une égalité par deux inégalités.

$$\cdots = b \iff \begin{cases} \cdots \leq b \\ \cdots \geq b \end{cases}$$

4. **Variable non restreinte** : Si x_i est non restreinte en signe, on la remplace par la différence de deux variables positives.

$$x_i = x_i' - x_i'' \quad \text{avec } x_i' \geq 0, x_i'' \geq 0$$

5. **Variable bornée non nulle** ($x_i \geq l$) : Faire un changement de variable pour se ramener à 0. Si $x_i \geq l$ (où l peut être négatif, ex : -34) : On pose $x_i' = x_i - l \implies x_i' \geq 0$. On substitue $x_i = x_i' + l$ partout dans le PL.
6. **Inégalité → Égalité (pour la Forme Standard)** :
   — Pour une contrainte ≤, on ajoute une **variable d'écart** $e_i \geq 0$.

$$\cdots \leq b \implies \cdots + e_i = b$$

   — Pour une contrainte ≥, on soustrait une **variable d'excédent** $s_i \geq 0$.

$$\cdots \geq b \implies \cdots - s_i = b$$

2.3 Exemple de Conversion

Exemple Conversion d'un PL mixte

Soit le PL suivant :

$$\begin{array}{ll} \text{Minimiser} & z = 5x_1 - 2x_2 \\ \text{S.C. :} & x_1 + 3x_2 = 10 \\ & 2x_1 - x_2 \geq 4 \\ & x_1 \geq 0 \end{array}$$

1. Conversion en Forme Canonique (Maximisation ≤)

— **Objectif** : $\min z \implies \max w = -z = -5x_1 + 2x_2$.
— **Variable** : $x_2 = x_2' - x_2''$ avec $x_2', x_2'' \geq 0$.
— **Contraintes** :
  — $x_1 + 3(x_2' - x_2'') = 10 \implies \begin{cases} x_1 + 3x_2' - 3x_2'' \leq 10 \\ x_1 + 3x_2' - 3x_2'' \geq 10 \implies -x_1 - 3x_2' + 3x_2'' \leq -10 \end{cases}$
  — $2x_1 - (x_2' - x_2'') \geq 4 \implies -2x_1 + x_2' - x_2'' \leq -4$

Résultat Canonique :

$$\begin{array}{ll} \text{Maximiser} & w = -5x_1 + 2x_2' - 2x_2'' \\ \text{S.C. :} & x_1 + 3x_2' - 3x_2'' \leq 10 \\ & -x_1 - 3x_2' + 3x_2'' \leq -10 \\ & -2x_1 + x_2' - x_2'' \leq -4 \\ & x_1, x_2', x_2'' \geq 0 \end{array}$$

2. Conversion en Forme Standard

- **Objectif** : On garde $\min z$.
- **Variable** : $x_2 = x_2' - x_2''$.
- **Contraintes** :
  - $x_1 + 3(x_2' - x_2'') = 10$ (Reste une égalité)
  - $2x_1 - (x_2' - x_2'') \geq 4 \implies 2x_1 - x_2' + x_2'' - s_1 = 4$ (On soustrait $s_1 \geq 0$)

Résultat Standard :

$$\begin{array}{ll} \text{Minimiser} & z = 5x_1 - 2x_2' + 2x_2'' \\ \text{S.C. :} & x_1 + 3x_2' - 3x_2'' = 10 \\ & 2x_1 - x_2' + x_2'' - s_1 = 4 \\ & x_1, x_2', x_2'', s_1 \geq 0 \end{array}$$

3 Dualité en PL

La dualité est un concept central en PL. À chaque PL, appelé **problème primal**, on peut associer un autre PL, appelé **problème dual**. La résolution de l'un fournit des informations directes sur la solution de l'autre.

Remarques Intuition de la dualité

Imaginez le primal comme une entreprise qui cherche à **maximiser son profit** (z) en assemblant des produits (x_i) à partir de ressources limitées (b_j). Le dual représente le point de vue d'un acheteur qui cherche à **minimiser le coût d'achat** (w) de toutes les ressources de l'entreprise. L'acheteur doit fixer un prix y_j pour chaque ressource j. Pour que l'entreprise accepte de vendre ses ressources, le prix offert pour les ressources nécessaires à la fabrication d'un produit i doit être au moins aussi élevé que le profit c_i que l'entreprise réalise sur ce produit.

3.1 Définition et Construction du Dual

La construction du dual suit des règles de transformation "miroir". La forme la plus simple est de partir d'un primal en forme canonique.

Proposition 2 Règles de transformation (Primal ↔ Dual)

Le tableau suivant résume la correspondance symétrique.

Primal (Maximisation)	**Dual (Minimisation)**
Maximiser $z = c^T x$	Minimiser $w = b^T y$
Matrice A	Matrice A^T (transposée)
m contraintes ($\leq b_i$)	m variables ($y_i \geq 0$)
n variables ($x_j \geq 0$)	n contraintes ($\geq c_j$)
Contrainte i de type =	Variable y_i non restreinte (n.r.s)
Variable x_j non restreinte (n.r.s)	Contrainte j de type =

3.2 Exemple de Construction (Primal Canonique)

Exemple Primal Canonique

Soit le **PL Primal** (forme canonique) :

$$\begin{array}{lll} \text{Maximiser} & z = 6x_1 + 4x_2 & \\ \text{S.C. :} & 3x_1 + 9x_2 \leq 81 & (y_1) \\ & 4x_1 + 5x_2 \leq 55 & (y_2) \\ & 2x_1 + 1x_2 \leq 20 & (y_3) \\ & x_1, x_2 \geq 0 & \end{array}$$

En appliquant les règles, on obtient le **PL Dual** :

— **Objectif (min)** : Les coefficients sont les seconds membres du primal (81, 55, 20).

— **Variables** : $m = 3$ contraintes $\implies$ 3 variables duales y_1, y_2, y_3.

— **Contraintes** ($\geq$) : $n = 2$ variables $\implies$ 2 contraintes duales. Les coefficients sont les colonnes de A.

$$\begin{array}{lll} \text{Minimiser} & w = 81y_1 + 55y_2 + 20y_3 & \\ \text{S.C. :} & 3y_1 + 4y_2 + 2y_3 \geq 6 & (\text{associée à } x_1) \\ & 9y_1 + 5y_2 + 1y_3 \geq 4 & (\text{associée à } x_2) \\ & y_1, y_2, y_3 \geq 0 & \end{array}$$

3.3 Théorèmes Fondamentaux

Théorème 1 Dualité Faible

Si x est une solution réalisable pour le primal (max) et y une solution réalisable pour le dual (min), alors la valeur de l'objectif du primal est toujours inférieure ou égale à celle du dual :

$$c^T x \leq b^T y$$

Remarques Corollaire de la dualité faible

Si l'on trouve un couple (x, y) de solutions réalisables tel que $c^T x = b^T y$, alors x est optimale pour le primal et y est optimale pour le dual.

⚠ Théorème Important 2 Dualité Forte

Si le problème primal admet une solution optimale x^*, alors le problème dual admet aussi une solution optimale y^*, et leurs valeurs objectives sont **égales** :

$$z_{opt} = c^T x^* = b^T y^* = w_{opt}$$

Remarques Les 3 cas Primal/Dual

La dualité forte implique qu'une seule des trois situations suivantes peut se produire :

1. Les deux problèmes ont une solution optimale finie (et leurs valeurs sont égales).
2. Un des problèmes est non borné (ex : primal $\to +\infty$), alors l'autre n'est pas réalisable (dual n'a pas de solution).
3. Les deux problèmes ne sont pas réalisables.

3.4 Conditions d'Optimalité (Écarts Complémentaires)

Ce théorème est le plus puissant en pratique. Il lie directement la solution optimale du primal à celle du dual.

⚠ Théorème Important 3 Écarts Complémentaires

Soient x^* une solution réalisable du primal et y^* une solution réalisable du dual. x^* et y^* sont optimales si et seulement si :

— **Écart Primal-Dual (Contraintes)** : Pour chaque variable duale y_i^*,

$$y_i^* \cdot (b_i - \sum_j a_{ij} x_j^*) = 0$$

(Si $y_i^* > 0$, la contrainte primale i est saturée (égalité). Si la contrainte i n'est pas saturée, $y_i^* = 0$.)

— **Écart Dual-Primal (Variables)** : Pour chaque variable primale x_j^*,

$$x_j^* \cdot (\sum_i a_{ij} y_i^* - c_j) = 0$$

(Si $x_j^* > 0$, la contrainte duale j est saturée. Si la contrainte j n'est pas saturée, $x_j^* = 0$.)

3.5 Application : Retrouver la solution du Dual à partir du Primal

Lorsque l'on a résolu le primal avec le simplexe, on obtient **gratuitement** la solution du dual.

Proposition 3 Lecture de la solution duale dans le tableau final

Supposons que le primal (max) a été mis sous forme standard en ajoutant les variables d'écart $e_1, \ldots, e_m$. Dans le **tableau final du simplexe** :

— La valeur optimale de la variable duale y_i^* est égale au **coefficient de la variable d'écart e_i dans la ligne de l'objectif** z.

(Note : selon la convention, si la ligne z est $z + \ldots$, c'est le coefficient. Si c'est $z - \ldots$, c'est l'opposé du coefficient). Dans la convention du simplexe par tableaux où la dernière ligne est $z - c'$, la valeur y_i^* est le coefficient de e_i dans cette ligne.

Exemple Utilisation des écarts complémentaires

Si, après résolution, on trouve $x_1^* = 10, x_2^* = 0$ pour l'Exemple 3.2. $e_1^* = 81 - 3(10) - 9(0) = 51 > 0$ $e_2^* = 55 - 4(10) - 5(0) = 15 > 0$ $e_3^* = 20 - 2(10) - 1(0) = 0$ (saturée)

Les conditions d'optimalité nous disent :

— $e_1^* > 0 \implies y_1^* = 0$

— $e_2^* > 0 \implies y_2^* = 0$

— $x_1^* > 0 \implies$ La 1ère contrainte duale est une égalité :

$$3y_1^* + 4y_2^* + 2y_3^* = 6$$

En remplaçant $y_1 = 0$ et $y_2 = 0$ dans l'égalité, on trouve $2y_3^* = 6 \implies y_3^* = 3$. La solution duale optimale (associée à $x = (10,0)$) serait $y^* = (0,0,3)$. (Note : $x = (10,0)$ n'est pas la solution optimale de ce PL, c'est juste pour illustrer la méthode).

3.6 Dual d'un PL en forme standard

La forme standard est $Ax = b, x \geq 0$. Pour trouver son dual, on utilise les règles mixtes.

Soit le **Primal (standard)** :

$$\begin{array}{ll} \text{Maximiser} & z = c^T x \\ \text{S.C. :} & Ax = b \\ & x \geq 0 \end{array}$$

Le **Dual** est :

$$\begin{array}{ll} \text{Minimiser} & w = b^T y \\ \text{S.C. :} & A^T y \geq c \\ & y \text{ n.r.s} \end{array}$$

Remarques Logique de la transformation

— Les m contraintes primales sont des égalités $\Longrightarrow$ Les m variables duales y_i sont non restreintes.
— Les n variables primales sont ≥ 0 $\Longrightarrow$ Les n contraintes duales sont des inégalités ($\geq c_j$).

4 Résolution Graphique (2 variables)

Pour les problèmes à deux variables (x_1, x_2), on peut trouver la solution de manière visuelle.

4.1 Principe

La méthode consiste à représenter graphiquement l'espace des solutions.

1. **Tracer les contraintes** : Pour chaque contrainte (ex : $x_1 + x_2 \leq 10$), on trace la droite frontière ($x_1 + x_2 = 10$).
2. **Identifier le demi-plan** : On teste un point (ex : (0,0)) pour savoir quel côté de la droite est valide. (Pour (0,0) : $0+0 \leq 10$, c'est vrai, donc le demi-plan contenant l'origine est le bon).
3. **Trouver la région réalisable** : C'est l'intersection de tous les demi-plans valides (y compris $x_1 \geq 0$ et $x_2 \geq 0$). Cette zone est un polygone (ou polyèdre) convexe, potentiellement non borné.

Définition 4 Région Réalisable (ou Admissible)

L'ensemble de tous les points (ici x_1, x_2) qui satisfont **toutes** les contraintes du programme linéaire.

4.2 Propriété Fondamentale

⚠ **Théorème Important 4** Théorème Fondamental de la PL

Si un programme linéaire admet une solution optimale (finie), alors cette solution se trouve nécessairement sur un **sommet** (un "coin") de la région réalisable.
— Si la solution optimale est unique, elle correspond à un seul sommet.
— S'il existe une infinité de solutions optimales, elles se trouvent sur une **arête** (un segment) de la région réalisable, incluant au moins deux sommets.

Remarques Intuition

L'algorithme du simplexe est l'implémentation algébrique de cette idée : il "saute" d'un sommet à un sommet adjacent tout en améliorant la fonction objectif, jusqu'à trouver le meilleur sommet.

4.3 Méthode du Vecteur Normal

C'est la méthode la plus rapide pour trouver le sommet optimal.

1. **Vecteur de l'objectif** : Pour la fonction $z = c_1x_1 + c_2x_2$, on trace le vecteur $\vec{n}_z = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$. Ce vecteur pointe dans la direction où z **augmente** (pour une maximisation).
2. **Droite de l'objectif** : On trace une ligne (ou "droite de niveau") perpendiculaire à $\vec{n}_z$. N'importe laquelle fera l'affaire, par exemple $c_1x_1 + c_2x_2 = 0$.
3. **Trouver l'optimum** : On "pousse" cette droite dans la direction de $\vec{n}_z$ (pour un max) ou dans la direction opposée (pour un min), jusqu'au **dernier point** où elle touche la région réalisable. Ce point est le sommet optimal.

Exemple Vecteur Normal pour l'Exercice du Partiel

$z = 4x_1 + 8x_2$. Le vecteur objectif est $\vec{n}_z = \begin{pmatrix} 4 \\ 8 \end{pmatrix}$. On trace la région réalisable définie par les 3 contraintes. En déplaçant la droite $4x_1 + 8x_2 = k$ dans la direction (4, 8), le dernier sommet touché sera la solution optimale.

Remarques Vecteur normal d'une contrainte

La même logique s'applique aux contraintes. Pour $ax_1 + bx_2 \leq c$, le vecteur $\vec{n} = \begin{pmatrix} a \\ b \end{pmatrix}$ est perpendiculaire à la droite et pointe vers la zone "interdite" (où $ax_1 + bx_2 > c$). La région valide est donc à l'opposé de ce vecteur.

4.4 Identification des cas particuliers

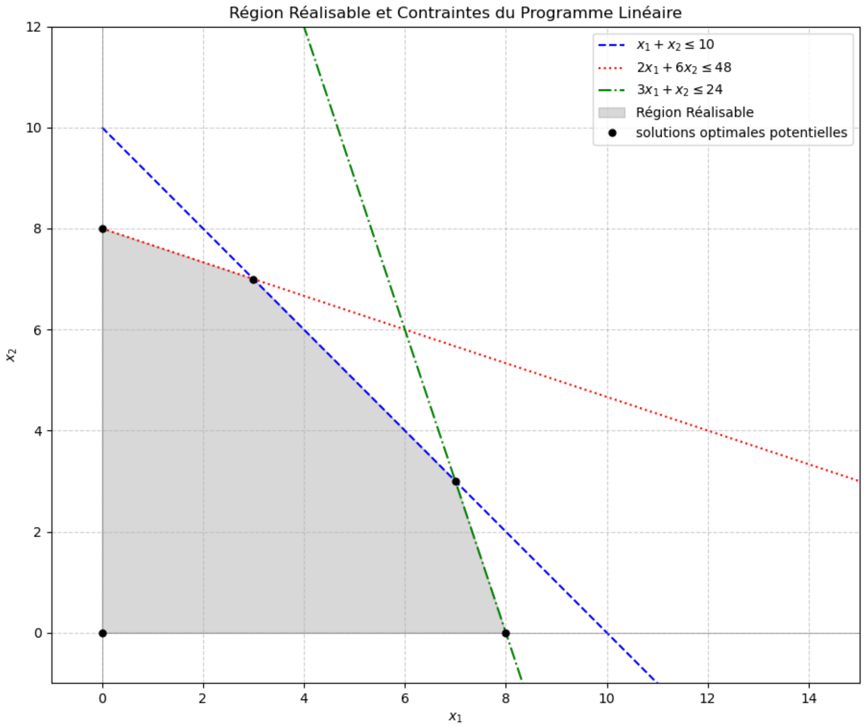
- **Problème non réalisable** : L'intersection des demi-plans est vide. Il n'y a **aucune** région réalisable.
- **Problème non borné** : La région réalisable est "ouverte" et s'étend à l'infini. Si la fonction objectif peut être améliorée indéfiniment en suivant une direction dans cette région (ex : on peut "pousser" la droite de l'objectif à l'infini), le problème est non borné.

4.5 Résolution graphique avec `matplotlib`

Considérons le programme linéaire (PL) suivant :

$$\begin{array}{lrl} \text{Maximiser} & z = 4x_1 + 8x_2 & \\ \text{Sujet à :} & x_1 + x_2 & \leq 10 \\ & 2x_1 + 6x_2 & \leq 48 \\ & 3x_1 + x_2 & \leq 24 \\ & x_1, x_2 & \geq 0 \end{array}$$

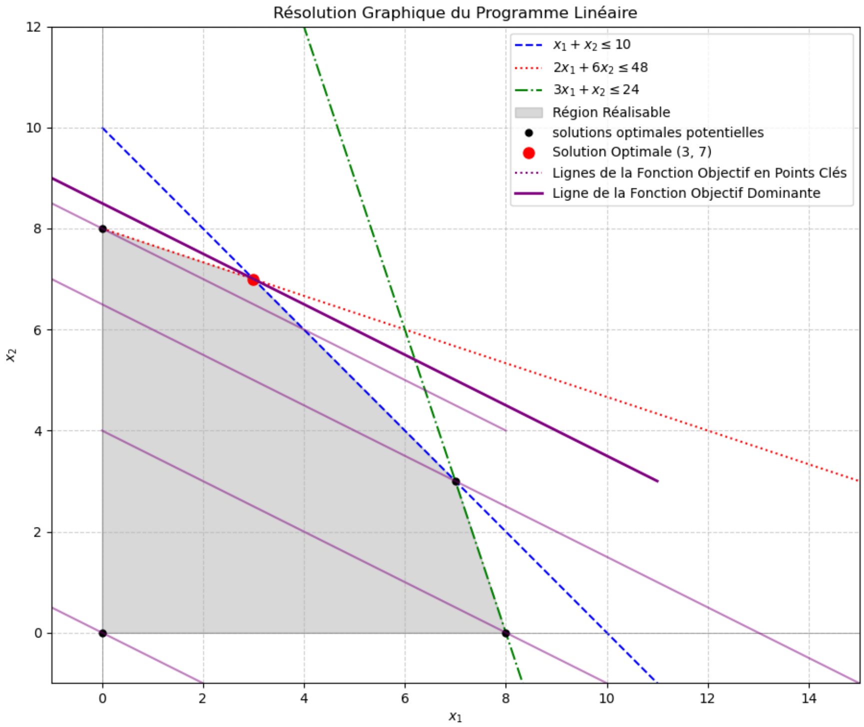
La méthode de résolution graphique consiste d'abord à tracer la **région réalisable**. Celle-ci est définie par l'intersection des demi-plans formés par chaque contrainte, incluant les contraintes de non-négativité ($x_1 \geq 0, x_2 \geq 0$). On obtient ainsi un polygone convexe, visualisé en gris sur la figure ci-dessous.



Ensuite, pour trouver la solution optimale, on trace une droite représentant la fonction objectif.

L'objectif de maximisation revient à trouver la valeur de c la plus élevée possible, ce qui correspond à un "balayage" de la région réalisable par cette droite en la déplaçant parallèlement à elle-même, le plus loin possible de l'origine.

La solution optimale est le **dernier sommet** de la région réalisable que la droite touche lors de ce balayage. Les différentes droites tracées sur la figure suivante illustrent ce processus.



5 L'Algorithme du Simplexe

L'algorithme du simplexe, développé par George Dantzig, est la méthode pivot pour résoudre les programmes linéaires. Il explore les sommets de la région réalisable de manière efficace pour trouver l'optimum.

5.1 Vocabulaire Clé

Définition 5 Base et Solution de Base

- **Base** : Dans un système de m équations (contraintes) et n variables ($n > m$), une *base* est un sous-ensemble de m variables.
- **Variables de base** : Les m variables choisies pour être dans la base.
- **Variables hors base** : Les $n - m$ autres variables, qui sont fixées à 0.
- **Solution de base** : La solution obtenue en résolvant le système d'équations après avoir fixé les variables hors base à 0.
- **Solution de base réalisable** : Une solution de base où toutes les variables (de base et hors base) sont ≥ 0. Chaque solution de base réalisable correspond à un sommet de la région réalisable.

Remarques Cas sur les dimensions n (variables) et m (contraintes)

Les définitions de *base* et *hors base* supposent un système "large" ($n > m$) où l'on a des degrés de liberté.

- **Si** $n = m$ (autant de variables que de contraintes) : Le système $Ax = b$ (en forme standard) a typiquement une seule solution (ou aucune). Il n'y a pas de variables hors base, ni de "choix" à faire. Ce n'est pas un problème pour le simplexe, la solution est triviale.
- **Si** $m > n$ (plus de contraintes que de variables) : Le système est *sur-déterminé*. Il y a des contraintes redondantes ou contradictoires. Si le système est réalisable, un prétraitement révélera que $\text{rang}(A) = m' \leq n$. On se ramène alors à un système de m' équations indépendantes.
- **Dualité** : Si un primal (en forme standard) a $n > m$, son dual (non restreint) aura m variables et n contraintes ($m < n$), et vice-versa.

L'algorithme du simplexe s'applique au cas standard où l'on a n variables (incluant les écarts) et m contraintes, avec $n > m$.

Définition 6 Terminologie de l'Itération

- **Variable entrante** : La variable hors base que l'on choisit de faire "entrer" dans la base (c'est-à-dire de la rendre non nulle) car elle améliore la fonction objectif.
- **Variable sortante** : La variable de base qui est contrainte de "sortir" de la base (passer à 0) pour que la nouvelle solution reste réalisable.
- **Pivot** : L'opération de réécriture du système (dictionnaire ou tableau) qui consiste à échanger les rôles de la variable entrante et de la variable sortante.

5.2 L'Algorithme (Étapes clés pour une maximisation)

Proposition 4 Algorithme du Simplexe

1. **Initialisation** : Mettre le PL sous forme standard. Trouver une première solution de base réalisable (un premier dictionnaire/tableau). Si les contraintes sont toutes $\leq$, les variables d'écart forment la base initiale.
2. **Test d'optimalité** : Regarder la ligne de l'objectif (z).
   — **STOP (Optimum atteint)** : Si tous les coefficients des variables hors base sont **négatifs ou nuls** (≤ 0), la solution actuelle est optimale.
3. **Choix de la variable entrante** : Choisir la variable hors base ayant le **coefficient positif le plus élevé** dans la ligne z. C'est celle qui augmente le plus l'objectif.
4. **Test du ratio (Choix de la variable sortante)** :
   — Pour chaque ligne i (chaque variable de base), calculer le ratio :

$$\text{Ratio}_i = \frac{\text{Valeur de la variable de base (colonne Sol.)}}{\text{Coefficient de la variable entrante sur cette ligne}}$$

   — On ne considère que les ratios où le dénominateur est **strictement positif** (> 0).
   — La **variable sortante** est celle de la ligne correspondant au **ratio minimum**.
5. **Cas non borné** : Si la variable entrante a un coefficient positif dans z, mais que tous ses coefficients dans les lignes de contraintes sont négatifs ou nuls (≤ 0), alors le problème est **non borné**. On peut augmenter cette variable à l'infini sans jamais enfreindre de contrainte. STOP.
6. **Pivotage** : Réécrire le dictionnaire/tableau pour échanger la variable entrante et la sortante. Retourner à l'étape 2.

5.3 Méthode 1 : Résolution par Dictionnaires

La méthode des dictionnaires est une présentation algébrique et littérale de l'algorithme du simplexe. C'est la même logique que les tableaux, mais présentée sous forme de système d'équations.

Définition 7 Dictionnaire

Un dictionnaire est un système d'équations qui exprime :

1. Les **variables de base** en fonction des **variables hors base**.
2. La **fonction objectif** z en fonction des **variables hors base**.

Un dictionnaire est dit *réalisable* si, en fixant toutes les variables hors base à 0, les valeurs résultantes des variables de base sont toutes ≥ 0.

L'algorithme du simplexe consiste à passer d'un dictionnaire réalisable à un autre par une opération de **pivotage** (échange d'une variable de base et hors base), en augmentant la valeur de z.

Exemple Résolution par Dictionnaires

Soit le PL (déjà sous forme standard avec les écarts e_1, e_2, e_3) :

$$\begin{array}{ll} \text{Maximiser} & z = 4x_1 + 8x_2 \\ \text{S.C. :} & x_1 + x_2 + e_1 = 10 \\ & 2x_1 + 6x_2 + e_2 = 48 \\ & 3x_1 + x_2 + e_3 = 24 \end{array}$$

1. Dictionnaire Initial (Base : $\{e_1, e_2, e_3\}$, Hors-base : $\{x_1, x_2\}$)

$$\begin{array}{l} e_1 = 10 - x_1 - x_2 \\ e_2 = 48 - 2x_1 - 6x_2 \\ e_3 = 24 - 3x_1 - x_2 \\ \hline z \ = \quad 0 + 4x_1 + 8x_2 \end{array}$$

Solution actuelle : $x_1 = 0, x_2 = 0 \implies e_1 = 10, e_2 = 48, e_3 = 24$. $z = 0$. Non optimal car z a des coefficients > 0.

2. Itération 1

- **Entrante** : x_2 (coefficient 8, le plus grand).
- **Sortante** (Test du ratio) :
  - $e_1 = 0 \implies 10 - x_2 = 0 \implies x_2 = 10$
  - $e_2 = 0 \implies 48 - 6x_2 = 0 \implies x_2 = 8 \quad \leftarrow$ **Minimum**
  - $e_3 = 0 \implies 24 - x_2 = 0 \implies x_2 = 24$

  e_2 sort. On pivote sur la 2ème ligne : $x_2 = 8 - \frac{1}{3}x_1 - \frac{1}{6}e_2$.

3. Dictionnaire 2 (Base : $\{e_1, x_2, e_3\}$, Hors-base : $\{x_1, e_2\}$) On substitue x_2 dans les autres équations :

$$\begin{array}{l} e_1 = 10 - x_1 - (8 - \frac{1}{3}x_1 - \frac{1}{6}e_2) = 2 - \frac{2}{3}x_1 + \frac{1}{6}e_2 \\ x_2 = 8 - \frac{1}{3}x_1 - \frac{1}{6}e_2 \\ e_3 = 24 - 3x_1 - (8 - \frac{1}{3}x_1 - \frac{1}{6}e_2) = 16 - \frac{8}{3}x_1 + \frac{1}{6}e_2 \\ \hline z \ = \ 4x_1 + 8(8 - \frac{1}{3}x_1 - \frac{1}{6}e_2) = 64 + \frac{4}{3}x_1 - \frac{4}{3}e_2 \end{array}$$

Solution actuelle : $x_1 = 0, e_2 = 0 \implies e_1 = 2, x_2 = 8, e_3 = 16$. $z = 64$. Non optimal car z a un coefficient > 0 $(4/3)$.

4. Itération 2

- **Entrante** : x_1 (coefficient $4/3$).
- **Sortante** (Test du ratio) :
  - $e_1 = 0 \implies 2 - \frac{2}{3}x_1 = 0 \implies x_1 = 3 \quad \leftarrow$ **Minimum**
  - $x_2 = 0 \implies 8 - \frac{1}{3}x_1 = 0 \implies x_1 = 24$
  - $e_3 = 0 \implies 16 - \frac{8}{3}x_1 = 0 \implies x_1 = 6$

  e_1 sort. On pivote sur la 1ère ligne : $x_1 = 3 - \frac{3}{2}e_1 + \frac{1}{4}e_2$.

5. Dictionnaire 3 (Final) (Base : $\{x_1, x_2, e_3\}$, Hors-base : $\{e_1, e_2\}$) On substitue x_1 dans les autres équations :

$$\begin{array}{l} x_1 = 3 - \frac{3}{2}e_1 + \frac{1}{4}e_2 \\ x_2 = 8 - \frac{1}{3}(3 - \frac{3}{2}e_1 + \frac{1}{4}e_2) - \frac{1}{6}e_2 = 7 + \frac{1}{2}e_1 - \frac{1}{4}e_2 \\ e_3 = 16 - \frac{8}{3}(3 - \frac{3}{2}e_1 + \frac{1}{4}e_2) + \frac{1}{6}e_2 = 8 + 4e_1 - \frac{1}{2}e_2 \\ \hline z \ = \ 64 + \frac{4}{3}(3 - \frac{3}{2}e_1 + \frac{1}{4}e_2) - \frac{4}{3}e_2 = 68 - 2e_1 - 1e_2 \end{array}$$

Solution actuelle : $e_1 = 0, e_2 = 0 \implies x_1 = 3, x_2 = 7, e_3 = 8$. $z = 68$. **Optimal** car tous les coefficients de z sont ≤ 0.
La solution optimale est $x_1^* = 3, x_2^* = 7$ avec un profit maximal de $z^* = 68$.

5.4 Présentation 2 : Méthode des Tableaux

Le tableau est une représentation compacte du dictionnaire.

- Les **lignes** correspondent aux variables de base.
- Les **colonnes** correspondent à toutes les variables.
- La **dernière ligne** est la fonction objectif, souvent écrite $z - (c^T x) = 0$.
- **Convention** : Nous écrivons $z - \cdots = 0$. L'optimum (pour un max) est atteint quand tous les coefficients des variables hors base dans cette ligne sont **positifs ou nuls** (≥ 0). Le choix de la variable entrante se porte sur le coefficient le **plus négatif**.

Exemple Résolution par Tableaux)

PL : $\max z = 4x_1 + 8x_2$. Ligne objectif : $z - 4x_1 - 8x_2 = 0$.

1. Tableau Initial

Base	x_1	x_2	e_1	e_2	e_3	Sol	Ratio
e_1	1	1	1	0	0	10	$10/1 = 10$
e_2	2	**6**	0	1	0	48	$48/6 = 8 \leftarrow$
e_3	3	1	0	0	1	24	$24/1 = 24$
z	-4	-8	0	0	0	0	

Choix : Entrante x_2 (coeff. -8, le plus négatif). Sortante e_2 (ratio min 8). Pivot = 6. *Opérations :* $L_2 \leftarrow L_2/6$. $L_1 \leftarrow L_1 - L_2'$. $L_3 \leftarrow L_3 - L_2'$. $L_z \leftarrow L_z + 8L_2'$.

2. Tableau 2

Base	x_1	x_2	e_1	e_2	e_3	Sol	Ratio
e_1	**2/3**	0	1	-1/6	0	2	$2/(2/3) = 3 \leftarrow$
x_2	1/3	1	0	1/6	0	8	$8/(1/3) = 24$
e_3	8/3	0	0	-1/6	1	16	$16/(8/3) = 6$
z	-4/3	0	0	4/3	0	64	

Choix : Entrante x_1 (coeff. -4/3). Sortante e_1 (ratio min 3). Pivot = 2/3. *Opérations :* $L_1 \leftarrow L_1 \cdot (3/2)$. $L_2 \leftarrow L_2 - (1/3)L_1'$. $L_3 \leftarrow L_3 - (8/3)L_1'$. $L_z \leftarrow L_z + (4/3)L_1'$.

3. Tableau 3 (Final)

Base	x_1	x_2	e_1	e_2	e_3	Sol
x_1	1	0	3/2	-1/4	0	3
x_2	0	1	-1/2	1/4	0	7
e_3	0	0	-4	1/2	1	8
z	0	0	2	1	0	68

Tous les coefficients de la ligne z (pour les variables hors base e_1, e_2) sont ≥ 0. La solution est optimale. $x_1^* = 3, x_2^* = 7, z^* = 68$. (Note : on retrouve $y_1^* = 2, y_2^* = 1, y_3^* = 0$ en lisant les coeff. des variables d'écart e_1, e_2, e_3 dans la ligne z).

5.5 Interprétation des solutions

Le tableau final du simplexe permet d'identifier le type de solution :

— **Solution optimale unique** : C'est le cas de l'exemple ci-dessus. Tous les coefficients des variables *hors base* dans la ligne z sont non nuls (strictement positifs dans notre convention de tableau).

— **Solutions optimales multiples** : S'il existe un coefficient **nul** pour une variable *hors base* dans la ligne z du tableau optimal. Cela signifie qu'on peut faire entrer cette variable dans la base (en faire un pivot) sans changer la valeur de z, ce qui nous amène à un autre sommet optimal. L'arête entre ces deux sommets est alors un segment de solutions optimales.

— **Problème non borné** : Comme vu à l'étape 5 de l'algorithme : on choisit une variable entrante (coeff. < 0 dans la ligne z), mais tous les coefficients de sa colonne dans les lignes de contraintes sont négatifs ou nuls (≤ 0). Il n'y a pas de ratio minimum positif, on peut augmenter la variable à l'infini.

5.6 Interprétation des solutions

Le tableau final du simplexe permet d'identifier le type de solution :

— **Solution optimale unique** : C'est le cas de l'exemple ci-dessus. Tous les coefficients des variables *hors base* dans la ligne z sont non nuls (strictement positifs dans notre convention de tableau).
— **Solutions optimales multiples** : S'il existe un coefficient **nul** pour une variable *hors base* dans la ligne z du tableau optimal. Cela signifie qu'on peut faire entrer cette variable dans la base (en faire un pivot) sans changer la valeur de z, ce qui nous amène à un autre sommet optimal. L'arête entre ces deux sommets est alors un segment de solutions optimales.
— **Problème non borné** : Comme vu à l'étape 5 de l'algorithme : on choisit une variable entrante (coeff. < 0 dans la ligne z), mais tous les coefficients de sa colonne dans les lignes de contraintes sont négatifs ou nuls (≤ 0). Il n'y a pas de ratio minimum positif, on peut augmenter la variable à l'infini.

6 Cas Particulier et Initialisation du Simplexe

6.1 Le Problème des contraintes <= ou =

Définition 8 Problème d'Initialisation

L'algorithme du simplexe (tel que vu ci-dessus) démarre avec une solution de base réalisable "évidente" :

— Pour les contraintes $\leq$, on ajoute des **variables d'écart** $e_i \geq 0$.
— Si $b_i \geq 0$, la base $\{e_1, \ldots, e_m\}$ (qui forme une matrice identité) donne la solution de base $x_j = 0, e_i = b_i$, qui est réalisable. C'est le sommet (0,0).

Le problème se pose avec les contraintes $\geq$ ou $=$.

— $2x_1 + x_2 \geq 4 \implies 2x_1 + x_2 - s_1 = 4$. Si on pose $x_1 = x_2 = 0$, on obtient $s_1 = -4$, ce qui n'est **pas réalisable**.
— $x_1 + x_2 = 10$. Si on pose $x_1 = x_2 = 0$, on obtient $0 = 10$, ce qui est impossible.

Nous n'avons pas de base de départ (sommet) évidente.

Remarques Simplexe vs Pivot de Gauss

Trouver une première base réalisable pour $Ax = b$ est un problème en soi. On pourrait tenter de le résoudre par un pivot de Gauss pour trouver une sous-matrice $m \times m$ inversible dont la solution $A_B^{-1}b$ est positive, mais c'est complexe.
Les méthodes suivantes (Deux Phases, Big-M) sont des façons **algorithmiques** d'utiliser le simplexe lui-même pour résoudre ce problème d'initialisation et trouver un premier sommet.

Pour contourner ce problème, on introduit des **variables artificielles** $a_i \geq 0$.

6.2 Méthode des Deux Phases

C'est la méthode la plus robuste. On sépare le problème en deux : 1) Trouver un sommet, 2) Optimiser.

Proposition 5 Principe des Deux Phases

1. **Phase 1 : Trouver une solution réalisable**
   — On ajoute une variable artificielle $a_i \geq 0$ pour *chaque* contrainte $\geq$ ou $=$.
   — On crée un **problème auxiliaire** dont l'objectif est de minimiser la somme de ces variables artificielles :

$$\min w = \sum a_i \quad (\text{ou } \max W' = -\sum a_i)$$

   — On forme le dictionnaire/tableau initial avec les a_i en base. (Attention : il faut substituer les a_i dans l'objectif W' pour qu'il ne dépende que des variables hors base, comme dans l'exemple ci-dessous).
   — On résout ce PL auxiliaire avec le simplexe.
2. **Analyse de la Phase 1** :
   — Si $w_{opt} > 0$ (ou $W'_{opt} < 0$) : Le problème original est **non réalisable** (impossible d'annuler les a_i). **STOP**.
   — Si $w_{opt} = 0$: On a trouvé une solution de base réalisable pour le problème original. On passe à la Phase 2.
3. **Phase 2 : Optimiser le problème original**
   — On repart du dictionnaire/tableau final de la Phase 1.
   — On supprime les colonnes des variables artificielles.
   — On ré-introduit la **fonction objectif originale** z.
   — On exprime z en fonction des variables hors base actuelles (substitution).
   — On continue le simplexe normalement jusqu'à l'optimum.

6.3 Méthode du Big-M

Cette méthode fait les deux phases en une seule fois.

Proposition 6 Principe du Big-M

1. On ajoute les variables artificielles $a_i \geq 0$ comme pour la Phase 1.
2. On crée **un seul** nouvel objectif z' en pénalisant très fortement les a_i dans l'objectif original z.
3. Soit M un nombre "très grand" (symbolique) :
   — Pour un $\max z$, on résout : $\max z' = z - M\sum a_i$
   — Pour un $\min z$, on résout : $\min z' = z + M\sum a_i$
4. On résout ce PL. La pénalité M étant si forte, l'algorithme va "forcer" les a_i à 0 s'il existe une solution réalisable.

Remarques Big-M vs Deux Phases

La méthode des Deux Phases est souvent préférée en implémentation car elle évite les problèmes d'instabilité numérique liés au choix d'un "M" très grand.

6.4 Exemple détaillé (Méthode des Deux Phases par Dictionnaires)

Exemple Initialisation avec contraintes d'égalité

Soit le PL :

$$\begin{array}{ll} \text{Maximiser} & z = 3x_1 + 2x_2 + 5x_3 \\ \text{S.C. :} & (1/2)x_1 + x_2 + (2/3)x_3 = 7.5 \\ & 3x_1 + x_2 + 2x_3 = 10 \\ & x_1, x_2, x_3 \geq 0 \end{array}$$

Phase 1 : Problème Auxiliaire On ajoute $a_1, a_2 \geq 0$.

$$\begin{aligned} a_1 &= 7.5 - 0.5x_1 - x_2 - (2/3)x_3 \\ a_2 &= 10 - 3x_1 - x_2 - 2x_3 \end{aligned}$$

L'objectif est $\min w = a_1 + a_2$, ce qui équivaut à $\max W' = -a_1 - a_2$. On doit exprimer W' en fonction des hors-base $\{x_1, x_2, x_3\}$:

$$W' = -(7.5 - \dots) - (10 - \dots) \implies W' = -17.5 + 3.5x_1 + 2x_2 + (8/3)x_3$$

Dictionnaire 1 (Phase 1)

$$\begin{array}{l} a_1 = 7.5 - 0.5x_1 - x_2 - (2/3)x_3 \\ a_2 = 10 - 3x_1 - x_2 - 2x_3 \\ \hline W' = -17.5 + 3.5x_1 + 2x_2 + (8/3)x_3 \end{array}$$

Itération 1 : x_1 entre, a_2 sort (ratio 10/3).

$$x_1 = 10/3 - 1/3x_2 - 2/3x_3 - 1/3a_2$$

Dictionnaire 2 (Phase 1)

$$\begin{array}{l} a_1 = 35/6 - 5/6x_2 - 1/3x_3 + 1/6a_2 \\ x_1 = 10/3 - 1/3x_2 - 2/3x_3 - 1/3a_2 \\ \hline W' = -35/6 + 5/6x_2 + 1/3x_3 - 7/6a_2 \end{array}$$

Itération 2 : x_2 entre, a_1 sort (ratio 7).

$$x_2 = 7 - 2/5x_3 - 6/5a_1 + 1/5a_2$$

Dictionnaire 3 (Fin Phase 1)

$$\begin{array}{l} x_2 = 7 - 2/5x_3 - 6/5a_1 + 1/5a_2 \\ x_1 = 1 - 8/15x_3 + 2/5a_1 - 2/5a_2 \\ \hline W' = 0 - a_1 - a_2 \end{array}$$

$W'_{opt} = 0$. La solution est réalisable. La base est $\{x_1, x_2\}$.
Phase 2 : Problème Original On repart du dictionnaire de base (en enlevant a_1, a_2) :

$$\begin{aligned} x_2 &= 7 - 2/5x_3 \\ x_1 &= 1 - 8/15x_3 \end{aligned}$$

On ré-introduit $z = 3x_1 + 2x_2 + 5x_3$. On substitue x_1, x_2 :

$$z = 3(1 - 8/15x_3) + 2(7 - 2/5x_3) + 5x_3 \implies z = 17 + (13/5)x_3$$

Dictionnaire 1 (Phase 2)

$$\begin{array}{l} x_2 = 7 - 2/5x_3 \\ x_1 = 1 - 8/15x_3 \\ \hline z \ = 17 + (13/5)x_3 \end{array}$$

Itération 1 : x_3 entre, x_1 sort (ratio $1/(8/15) = 15/8$).

$$x_3 = 15/8 - 15/8x_1$$

Dictionnaire 2 (Fin Phase 2)

$$\begin{array}{l} x_2 = 7 - 2/5(15/8 - 15/8x_1) = 25/4 + 3/4x_1 \\ x_3 = 15/8 - 15/8x_1 \\ \hline z \ = 17 + 13/5(15/8 - 15/8x_1) = 175/8 - 39/8x_1 \end{array}$$

Optimum atteint. Solution : $x_1 = 0$, $x_3 = 15/8$, $x_2 = 25/4$. $z = 175/8$.

7 Analyse Post-Optimale et Sensibilité

Une fois la solution optimale trouvée, il est souvent important de comprendre comment elle réagit à des changements dans les paramètres du problème (coûts, ressources disponibles). C'est l'objet de l'analyse de sensibilité.

7.1 Principe de l'analyse de sensibilité

L'analyse de sensibilité (ou post-optimale) étudie l'effet sur la solution optimale (x^* et z^*) d'une modification :

— des coefficients de la fonction objectif (c_j).
— des seconds membres des contraintes (b_i).
— des coefficients technologiques (a_{ij}).

Cela permet de déterminer la **robustesse** de la solution et les marges de manœuvre possibles.

7.2 Interprétation des sorties `lp_solve -S4`

La commande `lp_solve -S4` fournit une analyse de sensibilité sur les coefficients de la fonction objectif.

Exemple Sortie `lp_solve -S4` d'un PL

Le PL initial est :

```
/* Fonction objectif */
max: +2 x1 +3 x2 +5 x3;

/* Contraintes */
r1: x1 + 7 = 15;
r2: +2 x2 + x3 <= 1;
r3: +2 x1 + x2 <= 20;
```

La sortie `-S4` est :

```
Value of objective function: 21.00000000
```

```
Actual values of the variables:
x1                      8
x2                      0
x3                      1

Actual values of the constraints:
r1                      8
r2                      1
r3                     16

Objective function limits:
                              From         Till    From Value
x1                          -1e+30       1e+30       -1e+30
x2                          -1e+30          10          0.5
x3                             1.5       1e+30       -1e+30
```

Interprétation :

- **Solution optimale** : $z^* = 21$, obtenue pour $x_1^* = 8, x_2^* = 0, x_3^* = 1$. (Vérification : $z^* = 2(8) + 3(0) + 5(1) = 16 + 5 = 21$).
- `Objective function limits (From / Till)` :
  - Pour x_1 (coeff. 2) : L'intervalle $]-\infty, +\infty[$ montre que son coefficient n'a aucun impact sur la solution. Ceci est logique car x_1 est **fixée** à 8 par la contrainte r_1.
  - Pour x_2 (coeff. 3) : Son coefficient peut varier dans $]-\infty, 10]$ sans que la solution optimale $(8, 0, 1)$ ne change.
  - Pour x_3 (coeff. 5) : Son coefficient peut varier dans $[1.5, +\infty[$ sans que la solution optimale $(8, 0, 1)$ ne change.
- `From Value` : Cette colonne est cruciale pour les variables dont la valeur optimale est 0 (variables "hors base"), comme x_2.
  - Elle **prédit la valeur** que prendra x_2 si son coefficient (c_2) sort de l'intervalle de stabilité, la rendant "profitable".
  - Ici (en maximisation), l'intervalle est $]-\infty, 10]$. Si c_2 devient **strictement supérieur** à 10, la solution actuelle n'est plus optimale. x_2 deviendra "profitable", entrera dans la solution, et sa nouvelle valeur sera $x_2^* = 0.5$.

Application (analyse de scénarios) :

- **Scénario 1** : Nouveau coefficient de x_2 est $c_2 = 5$. $5 \in]-\infty, 10]$. Le coefficient reste dans l'intervalle de stabilité. La solution optimale **ne change pas** : $x_1^* = 8, x_2^* = 0, x_3^* = 1$. La nouvelle valeur de l'objectif est $z_{new} = 2(8) + 5(0) + 5(1) = 21$. (La valeur de z ne change pas car $x_2 = 0$).
- **Scénario 2** : Nouveau coefficient de x_2 est $c_2 = 15$. $15 \notin]-\infty, 10]$. Le coefficient est hors de l'intervalle de stabilité. La solution optimale **va changer**. Grâce au `From Value`, on peut prédire que x_2 va devenir non-nulle et prendre la valeur $x_2^* = 0.5$. (Note : la contrainte $r_2 : 2x_2 + x_3 \leq 1$ deviendra alors $2(0.5) + x_3 \leq 1 \Rightarrow 1 + x_3 \leq 1 \Rightarrow x_3 \leq 0$. La nouvelle solution sera donc $x_1^* = 8, x_2^* = 0.5, x_3^* = 0$).
- **Utilité** : L'intervalle `[From, Till]` indique la plage de variation de la marge pour chaque produit sans que le plan de production optimal ne soit modifié. Par exemple, la marge de x_2 peut augmenter de 3 jusqu'à 10 sans qu'il devienne rentable d'en produire (x_2 resterait à 0). Si elle dépasse 10, il faudra en produire 0.5 unité.

7.3 Utilité du pré-traitement (presolve)

Proposition 7 Presolve

Avant de lancer l'algorithme du simplexe, les solveurs comme `lp_solve` appliquent une phase de **prétraitement** (ou *presolve*) pour simplifier le problème.

- **'presolverow'** : Analyse les contraintes (lignes).
  - Élimine les contraintes redondantes.
  - Détecte l'infaisabilité (ex : $x_1 \leq 5$ et $x_1 \geq 10$).
  - Fixe des variables (ex : $x_1 + 7 = 15 \implies x_1 = 8$).
  - Renforce les bornes des variables.
- **'presolvecol'** : Analyse les variables (colonnes), souvent après 'presolverow'.
  - Supprime les variables fixées (ex : si $x_1 = 8$ a été trouvé, on substitue 8 partout et on enlève x_1 du problème).
  - Élimine des variables en utilisant les contraintes d'égalité.

Le but est de réduire la taille du problème (nombre de variables et de contraintes) avant d'appliquer l'algorithme coûteux du simplexe.

Exemple Presolve sur le PL précédent

- 'lp_solve -presolverow to_solve.lp ...' : La contrainte 'r1 : x1 + 7 = 15' est transformée en 'x1 = 8'. Le solveur substitue $x_1 = 8$ dans les autres contraintes et l'objectif.
- 'lp_solve -presolvecol to_solve_row.lp ...' : La variable x_1 étant maintenant fixée, elle est retirée du problème. Le PL soumis au simplexe ne contiendra plus que x_2 et x_3.

8 Bonus : Résoudre avec des Variables de Décision Entières (PLNE / MILP)

Cette section introduit un nouveau type de problème où les solutions "entre-deux" (comme fabriquer 2.5 produits) ne sont pas autorisées.

8.1 Introduction à la Programmation en Nombres Entiers (PLNE)

Définition 9 Programmation en Nombres Entiers (PLNE)

- **PLNE (Programmation en Nombres Entiers)** ou *ILP (Integer Linear Programming)* : Un PL où **toutes** les variables de décision doivent être des entiers ($x_i \in \mathbb{Z}$ ou $x_i \in \mathbb{N}$).
- **PLNE Mixte** ou *MILP (Mixed-Integer Linear Programming)* : Un PL où **certaines** variables sont entières et d'autres peuvent être continues.

Remarques L'illusion de l'arrondi

Pourquoi ne pas simplement résoudre le PL (appelé **relaxation continue**) et arrondir la solution à l'entier le plus proche ?
Cette approche **ne fonctionne pas** car la solution arrondie peut être :

1. **Non réalisable** : Elle ne respecte pas une ou plusieurs contraintes.
2. **Sous-optimale** : Elle est réalisable, mais ce n'est pas la meilleure solution *entière* possible. La solution optimale entière peut être loin de la solution continue.

Voir figure 8.3 pour une illustration graphique.

Remarques Complexité

- La **Programmation Linéaire (PL)** est considérée comme "efficace". Elle appartient à la classe de complexité **P**, ce qui signifie qu'il existe des algorithmes (comme la **méthode des ellispoïdes**) capables de trouver la solution optimale *exacte* en **temps polynomial**. L'algorithme du simplexe, bien qu'ayant une complexité exponentielle dans le pire des cas, reste extrêmement performant en pratique.
- La **Programmation en Nombres Entiers (PLNE)** est **NP-difficile**. L'ajout de contraintes d'intégrité transforme un problème "continu" (PL) en un problème "combinatoire" (PLNE) fondamentalement plus difficile. Il n'existe pas d'algorithme connu pour le résoudre en temps polynomial, et le temps de résolution peut croître de manière exponentielle avec la taille du problème.

8.2 Les Variables Binaires en PLNE

C'est un type courant de variable entière, utilisé pour la modélisation logique.

Exemple Modéliser des variables binaires

On utilise une variable binaire $x_i \in [\![0,1]\!] = \{0,1\}$. Exemple : "Faut-il construire l'usine i ?"

$$x_i = \begin{cases} 1 & \text{si on construit l'usine } i \\ 0 & \text{sinon} \end{cases}$$

Exemple Modéliser des contraintes logiques

Soient x_A et x_B des variables binaires représentant deux décisions.

— **Sélection (K parmi N)** : "Choisir au plus 3 projets parmi 5 ($x_1 \dots x_5$)"

$$\sum_{i=1}^{5} x_i \le 3$$

— **Dépendance (Si... alors...) (Implication)** : "Si on fait le projet A ($x_A = 1$), alors on *doit* faire le projet B ($x_B = 1$)"

$$x_A \le x_B$$

(Vérification : Si $x_A = 1$, x_B doit être 1. Si $x_A = 0$, x_B peut être 0 ou 1.)

— **Exclusion mutuelle (Au plus un)** : "On ne peut pas faire A et B en même temps" (Logiquement A NAND B)

$$x_A + x_B \le 1$$

— **OU Exclusif (XOR)** : "On doit faire A ou B, mais *pas* les deux" (Logiquement A XOR B)

$$x_A + x_B = 1$$

Proposition 8 Modéliser des contraintes disjonctives ("OU") - Le "Big-M"

Comment modéliser "Contrainte 1 OU Contrainte 2 doit être vraie" ? (Ex : $f(x) \le 0$ OU $g(x) \le 0$).

1. On introduit une variable binaire $y \in \{0, 1\}$.
2. On choisit une constante M "très grande" (Big-M), qui doit être supérieure à la valeur maximale que $f(x)$ et $g(x)$ peuvent prendre dans la région réalisable.
3. On écrit le système :

$$f(x) \le M \cdot y$$

$$g(x) \le M \cdot (1 - y)$$

Analyse :

— Si l'optimiseur choisit $y = 0$: La 1ère contrainte devient $f(x) \le 0$ (active) et la 2nde devient $g(x) \le M$ (désactivée, car toujours vraie).

— Si l'optimiseur choisit $y = 1$: La 1ère contrainte devient $f(x) \le M$ (désactivée) et la 2nde devient $g(x) \le 0$ (active).

8.3 Méthodes de Résolution

Comment trouver la solution optimale entière si le Simplexe (qui ne garantit qu'un optimum continu) ne marche pas ?

Proposition 9 Algorithme de Branch and Bound

C'est l'algorithme fondamental pour la PLNE. Le principe est "Diviser pour régner" en explorant intelligemment l'arbre des solutions possibles.
Étapes clés :

1. **Racine (Relaxation)** : Résoudre le PL initial en **relaxant les contraintes d'intégrité** (c'est-à-dire en traitant $x_i \in \mathbb{N}$ comme $x_i \geq 0$). Soit z_R sa valeur. Si la solution x^* est déjà entièrement entière, STOP : c'est l'optimum.
2. **Séparation (Branch)** : Si la solution x^* a une variable non-entière (ex : $x_i = 2.7$), on crée deux nouveaux sous-problèmes (deux "branches") qui partitionnent l'espace des solutions :
   — **Problème A** = Problème initial + Contrainte $x_i \leq \lfloor 2.7 \rfloor \implies x_i \leq 2$
   — **Problème B** = Problème initial + Contrainte $x_i \geq \lceil 2.7 \rceil \implies x_i \geq 3$
3. **Évaluation (Bound)** : On résout la relaxation continue de ces deux sous-problèmes. La solution z_{relax} d'un nœud est une **borne supérieure** (pour un max) de la meilleure solution *entière* qu'on pourra trouver dans cette branche.
4. **Élagage (Prune)** : On arrête d'explorer une branche (on la "coupe") si :
   — **Irréalisable** : Le sous-problème n'a pas de solution.
   — **Dominée par la borne** : Sa borne (ex : $z_{relax} = 8.5$) est moins bonne qu'une solution *entière* déjà trouvée ailleurs (ex : $z_{entiere} = 9$). On appelle cette meilleure solution entière le *meilleur candidat*.
   — **Solution entière** : Le sous-problème donne une solution entièrement entière. On la compare au *meilleur candidat* et on la garde si elle est meilleure.
5. On continue d'explorer les nœuds non-élagués jusqu'à ce que tout l'arbre soit exploré ou élagué.

TODO : donner une exemple qu'on peut faire à la main avec 2 variables

Exemple Résolution par Branch and Bound

Soit le Problème de Programmation Linéaire en Nombres Entiers (PLNE) suivant :

$$\begin{array}{ll} \text{Maximiser} & z = 5x_1 + 8x_2 \\ \text{S.C. :} & x_1 + x_2 \leq 6 \\ & 5x_1 + 9x_2 \leq 45 \\ & x_1, x_2 \in \mathbb{N} \end{array}$$

Nous allons résoudre ce problème en explorant un arbre de décision. La meilleure solution entière trouvée sera notre **meilleur candidat** (ou *incumbent*), sa valeur est la **borne inférieure** (z_{inf}). Initialement, $z_{inf} = -\infty$.

Nœud 0 (Racine - P_0)

On résout la **relaxation continue** (en ignorant $\in \mathbb{N}$).
— La région réalisable est définie par les coins $(0,0)$, $(6,0)$, $(0,5)$ et l'intersection de $x_1 + x_2 = 6$ et $5x_1 + 9x_2 = 45$.
— Intersection :

$$\begin{aligned} x_1 = 6 - x_2 \quad & \implies 5(6 - x_2) + 9x_2 = 45 \\ & \implies 30 - 5x_2 + 9x_2 = 45 \\ & \implies 4x_2 = 15 \implies x_2 = 3.75 \end{aligned}$$

— $x_1 = 6 - 3.75 = 2.25$.
— La solution optimale continue est $x^* = (2.25, 3.75)$.
— Valeur : $z_R = 5(2.25) + 8(3.75) = 11.25 + 30 = 41.25$.

Résultat : $z_R = 41.25$. Cette valeur est notre **borne supérieure** (z_{sup}). La solution n'est pas entière.

Séparation : On choisit une variable non-entière, par exemple $x_2 = 3.75$. On crée deux branches :
— **Nœud 1 (P_1)** : P_0 + contrainte $x_2 \leq \lfloor 3.75 \rfloor \implies x_2 \leq 3$.
— **Nœud 2 (P_2)** : P_0 + contrainte $x_2 \geq \lceil 3.75 \rceil \implies x_2 \geq 4$.

Nœud 1 (Évaluation - P_1)

On résout la relaxation de P_1 (Problème P_0 avec $x_2 \leq 3$).
— La nouvelle région est coupée par $x_2 \leq 3$. L'optimum (qui était à $x_2 = 3.75$) va "glisser" le long d'une contrainte.
— Le nouveau sommet optimal est à l'intersection de $x_1 + x_2 = 6$ et $x_2 = 3$, ce qui donne $x^* = (3,3)$.
— Vérification : $(3,3)$ est réalisable : $3+3 = 6 \leq 6$ et $5(3)+9(3) = 15+27 = 42 \leq 45$.
— Valeur : $z = 5(3) + 8(3) = 15 + 24 = 39$.

Résultat : Solution **entière** trouvée ! $x^* = (3,3)$ avec $z = 39$.
Mise à jour : $z_{inf} = 39$. On garde $(3,3)$ comme meilleur candidat.
Élagage (Prune) : Ce nœud est terminal car il a donné une solution entière.

Nœud 2 (Évaluation - P_2)

On résout la relaxation de P_2 (Problème P_0 avec $x_2 \geq 4$).
— La nouvelle région est coupée par $x_2 \geq 4$.
— Le nouveau sommet optimal est à l'intersection de $x_2 = 4$ et $x_1 + x_2 = 6$, ce qui donne $x^* = (2,4)$.
— Vérification : $(2,4)$ est réalisable : $2+4 = 6 \leq 6$ et $5(2)+9(4) = 10+36 = 46 \not\leq 45$.)
— $(2,4)$ n'est pas réalisable. L'optimum doit être sur l'autre contrainte.
— Essayons l'intersection de $x_2 = 4$ et $5x_1 + 9x_2 = 45$:

$$\begin{aligned} 5x_1 + 9(4) = 45 \quad & \implies 5x_1 = 9 \\ & \implies x_1 = 1.8 \end{aligned}$$

— La solution est $x^* = (1.8, 4)$.
— Valeur : $z_R = 5(1.8) + 8(4) = 9 + 32 = 41$.

Résultat : $z_R = 41$. Cette valeur est la borne supérieure de cette branche.
Analyse : $z_R = 41$ est **supérieur** à notre meilleur candidat $z_{inf} = 39$. Il est donc *possible* qu'une solution entière meilleure que 39 existe dans cette branche.
Séparation : La solution $x^* = (1.8, 4)$ n'est pas entière. On branche sur $x_1 = 1.8$.
— **Nœud 3 (P_3)** : P_2 + contrainte $x_1 \leq 1$.
— **Nœud 4 (P_4)** : P_2 + contrainte $x_1 \geq 2$.

Nœud 3 (Évaluation - P_3)

On résout la relaxation de P_3 (P_0 avec $x_2 \geq 4$ et $x_1 \leq 1$).
— On cherche l'optimum dans la région $x_1 \leq 1, x_2 \geq 4$.
— Le meilleur point est $x_1 = 1, x_2 =?$.

— Contraintes :

$$\begin{aligned} 1 + x_2 \leq 6 & \implies x_2 \leq 5 \\ 5(1) + 9x_2 \leq 45 & \implies 9x_2 \leq 40 \\ & \implies x_2 \leq 4.44... \end{aligned}$$

— La région est $x_1 \leq 1$ et $4 \leq x_2 \leq 4.44...$
— L'optimum est au coin $x^* = (1, 40/9 \approx 4.44)$.
— Valeur : $z_R = 5(1) + 8(40/9) = 5 + 320/9 = (45 + 320)/9 = 365/9 \approx 40.55$.

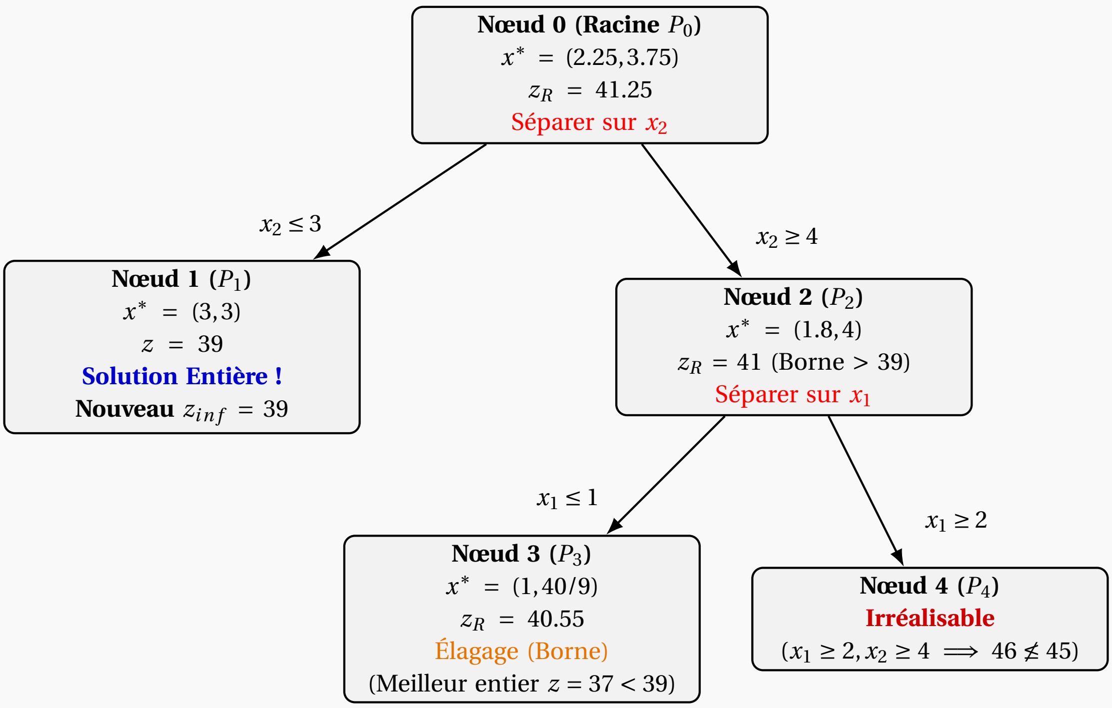
Résultat : $z_R \approx 40.55$. C'est $> z_{inf} = 39$. On doit continuer... mais on peut aussi élaguer par borne si on est malin.
Élagage (Prune) : On peut aussi chercher une solution entière simple. Essayons $(1,4)$. $z = 5(1) + 8(4) = 37$. C'est moins bon que 39. Aucune autre solution entière n'existe dans P_3 (car $x_1 = 0$ ou $x_1 = 1$ et $x_2 = 4$). L'optimum de la relaxation (40.55) est le max possible, mais il n'y a pas d'entier > 39 ici. On peut élaguer.

Nœud 4 (Évaluation - P_4)

On résout la relaxation de P_4 (P_0 avec $x_2 \geq 4$ et $x_1 \geq 2$).
— Contraintes : $x_1 \geq 2, x_2 \geq 4$.
— $x_1 + x_2 \leq 6 \implies 2 + 4 = 6 \leq 6$. Le seul point possible est $(2,4)$.
— $5x_1 + 9x_2 \leq 45 \implies 5(2) + 9(4) = 10 + 36 = 46 \not\leq 45$.

Résultat : Problème **Irréalisable**. **Élagage (Prune) :** Ce nœud est terminal.

Arbre de Branch and Bound



Conclusion

L'arbre a été entièrement exploré (toutes les feuilles sont élaguées). La meilleure solution entière trouvée (le meilleur *incumbent*) est la solution optimale.
Solution Optimale PLNE : $x_1^* = 3, x_2^* = 3$ avec $z^* = 39$.

Remarques Visualisation de l'algorithme Branch and Bound

Pour mieux comprendre le déroulement de l'algorithme, les graphiques suivants décomposent chaque étape de l'exemple.

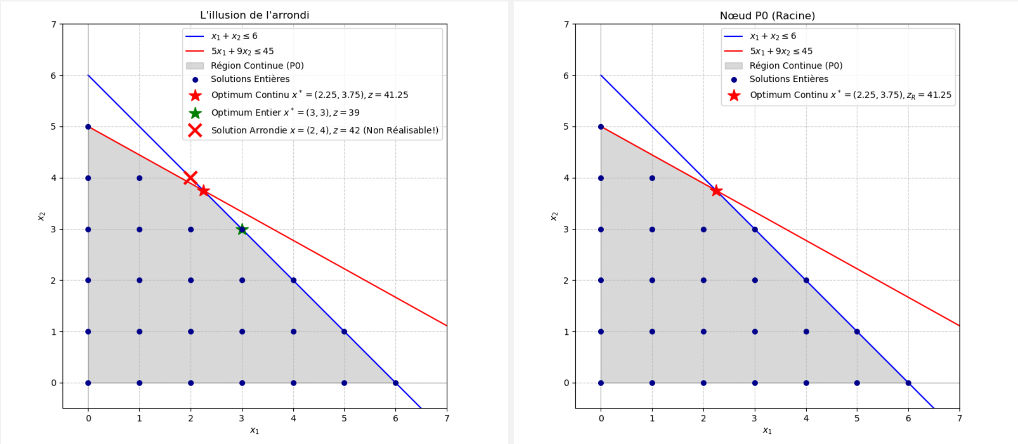


FIGURE 1 – L'illusion de l'arrondi (gauche) montre que l'optimum continu (⋆ rouge) arrondi au plus proche (croix rouge) est non réalisable. L'optimum entier (⋆ verte) est ailleurs. Le Nœud P0 (droite) montre la relaxation continue avec les solutions entières candidates (points bleus).

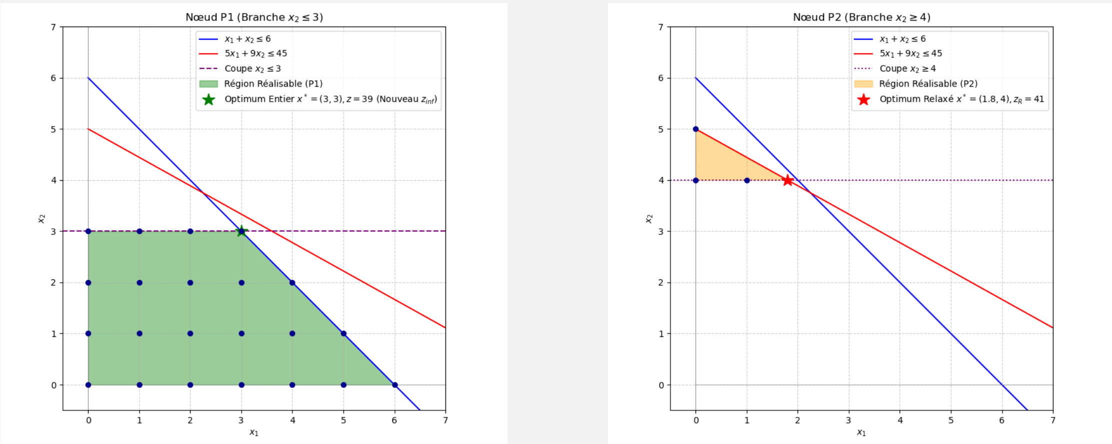


FIGURE 2 – Les séparations de P0 : P1 (gauche) trouve une solution entière ($z = 39$), devient la borne inférieure. P2 (droite) a un optimum continu ($z_R = 41$) supérieur à notre borne. On doit la séparer.

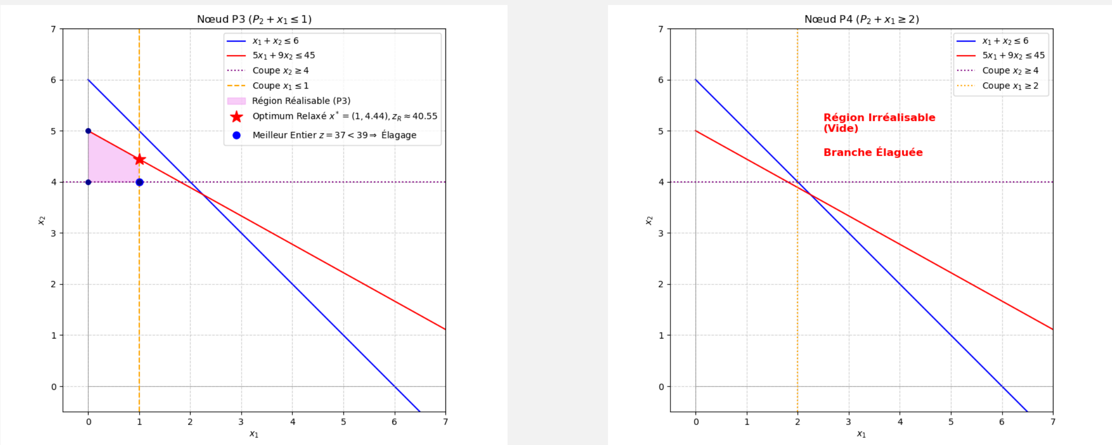


FIGURE 3 – Les séparations de P2 : P3 (gauche) a un optimum $z_R \approx 40.55$, mais sa meilleure solution entière est $z = 37 < 39$. Donc on élague. P4 (droite) est irréalisable, donc on termine.

Remarques Remarque sur l'exemple

Notez que l'optimum continu était $z = 41.25$ à $(2.25, 3.75)$. Si on avait simplement **arrondi** au plus proche, on aurait testé $(2,4)$. Le point $(2,4)$ donne $z = 5(2) + 8(4) = 42$. Mais $(2,4)$ est **non réalisable** (il viole $5x_1 + 9x_2 \leq 45$).
L'arrondi "intelligent" le plus proche et réalisable aurait pu être $(3,3)$ ($z = 39$) ou $(1,4)$ (qui est réalisable, $z = 5(1) + 8(4) = 37$).
Branch and Bound est une méthode qui garantit de trouver le véritable optimum entier $(3,3)$ sans se tromper.

Remarques Méthode des Plans Coupants et Hybrides

Une autre approche consiste à ajouter "intelligemment" de nouvelles contraintes (des "coupes") au PL.
Ces contraintes sont conçues pour réduire la région réalisable continue **sans** éliminer aucune des solutions entières valides.
L'objectif est de "rapprocher" le sommet optimal de la relaxation continue d'un sommet entier (exemple : Coupes de Gomory).
De plus, des approches hybrides combinant Branch and Bound avec l'ajout de plans coupants **permettent** d'obtenir des résultats encore plus efficaces. C'est ce qu'on appelle la méthode **Branch and Cut**, utilisée par la majorité des solveurs PLNE modernes.

9 Bonus : Preuves des propriétés et théorèmes !

TODO : mettre au propre des preuves, avec des explications intuitives, sur les théorèmes précédents.

10 Bonus : Méthode des points intérieurs

Un jour viendra ou je compléterai cette partie avec une run et ou des illustrations

Remarques Méthode des Points Intérieurs (MPI)

— **Principe :** Contrairement à la méthode du simplexe qui explore les sommets (les bords) du polyèdre de solutions réalisables, la MPI traverse l'intérieur de cet ensemble (d'où son nom) en suivant une courbe appelée **chemin central**. Elle utilise une fonction de **barrière logarithmique** pour pénaliser les itérations qui s'approchent trop des contraintes.

— **Nature de la Solution :** C'est une méthode **itérative**. Elle produit une séquence de points convergeant vers la solution optimale *exacte* en PL. En pratique, elle s'arrête lorsqu'une solution **approximative** atteint une précision suffisante, mais des techniques de **croisement (crossover)** peuvent la ramener à un sommet pour obtenir la solution exacte.

— **Complexité Théorique :** La MPI a été le premier algorithme à résoudre la PL en **temps polynomial** (comme la méthode de l'ellipsoïde), confirmant ainsi que la PL est bien dans la classe **P**. Sa complexité est souvent de l'ordre de $O(n^3 L)$ ou $O(n^{3.5} L)$, où n est le nombre de variables et L la taille du codage de l'entrée.

— **Quand l'utiliser ?**

1. **Très Grands Problèmes :** Elle est souvent plus rapide que le simplexe pour les problèmes de PL de très grande taille ou ayant un grand nombre de contraintes, en particulier ceux ayant une structure **creuse**.
2. **Pire Cas :** Elle garantit une complexité polynomiale (contrairement au simplexe).
3. **Optimisation Convexe :** La MPI est facilement généralisable et est la méthode de choix pour des problèmes d'optimisation plus complexes que la PL, comme la **Programmation Quadratique Convexe**.

11 Références

[1] Abdelkader Ouali (2025) Cours de M2 sur la Programmation Linéaire et la Programmation par Contraintes, Université de Caen.

[2] Wikipedia

[3] Salim Haddadi (2021) *Programmation linéaire - Une approche mathématique et algorithmique*