HAI722I — DM

Ivan Lejeune

11 décembre 2025

Table des matières

Instructions

Ce devoir est à rendre avant le 12 décembre 2025 à 12h, soit par mail à l'adresse : rodolphe.giroudeau@lirmm.fr, soit en déposant votre devoir durant le cours

1 Partie théorique

Exercice 1 Algorithmes pour la programmation linéaire.
Considérons la formulation suivante :

$$P_\beta = \begin{cases} \max z = 5x_1 + 2x_2 \\ 6x_1 + x_2 \geq 6 \\ 4x_1 + 4x_2 \geq 12 \\ x_1 + 2x_2 \geq 4 \\ x_i \geq 0, \quad \forall i \in \{1,2\} \end{cases}$$

1. Résoudre le problème P_β par la méthode du big M.
2. Résoudre le problème P_β par la méthode à deux phases.
3. **Difficile :**
   (a) Résoudre le problème P_β par la méthode dual-simplexe.
   (b) Soit le programme linéaire P_θ

$$P_\theta = \begin{cases} \max z = x_1 + 3x_2 \\ x_1 + x_2 \geq 3 \\ x_1 - 2x_2 \geq 5 \\ -2x_1 + x_2 \leq 5 \\ x_i \geq 0, \quad \forall i \in \{1,2\} \end{cases}$$

Résoudre le problème P_θ par la méthode dual-simplexe.

Solution.

1. Commençons par poser le problème sous forme standard :

$$P_\beta = \begin{cases} \max z = 5x_1 + 2x_2 + 0x_3 + 0x_4 + 0x_5 - M \cdot (y_1 + y_2 + y_3) \\ 6x_1 + x_2 - x_3 + y_1 = 6 \\ 4x_1 + 4x_2 - x_4 + y_2 = 12 \\ x_1 + 2x_2 - x_5 + y_3 = 4 \\ x_i \geq 0, y_j \geq 0, \quad \forall i \in \{1,\dots,5\}, \forall j \in \{1,2,3\}. \end{cases}$$

Ensuite on construit notre tableau du simplexe :

		c	5	2	0	0	0	$-M$	$-M$	$-M$
c^J	variables de base		x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3
$-M$	$x_1^1 = y_1$	6	6	1	-1	0	0	1	0	0
$-M$	$x_2^1 = y_2$	12	4	4	0	-1	0	0	1	0
$-M$	$x_3^1 = y_3$	4	1	2	0	0	-1	0	0	1
	$z(x)$	$-22M$	$-11M-5$	$-7M-2$	M	M	M	0	0	0

et on déroule l'algorithme :

- on rentre x_1,

- on sort y_1 car $1 < 3 < 4$,

$\downarrow$

			c	5	2	0	0	0	$-M$	$-M$	$-M$
c^J	variables de base			x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3
5	$x_1^2 = x_1$	1		1	$\frac{1}{6}$	$-\frac{1}{6}$	0	0	$\frac{1}{6}$	0	0
$-M$	$x_2^2 = y_2$	8		0	$\frac{10}{3}$	$\frac{4}{6}$	-1	0	$-\frac{4}{6}$	1	0
$-M$	$x_3^2 = y_3$	3		0	$\frac{11}{6}$	$\frac{1}{6}$	0	-1	$-\frac{1}{6}$	0	1
	$z(x)$	$-11M+5$		0	$-\frac{31}{6}M - \frac{7}{12}$	$-\frac{5}{6}M - \frac{5}{6}$	M	M	$\frac{11}{6}M + \frac{5}{6}$	0	0

- on rentre x_2,
- on sort y_3 car $\frac{11}{2} < 6 < \frac{80}{3}$,

$\downarrow$

			c	5	2	0	0	0	$-M$	$-M$	$-M$
c^J	variables de base			x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3
5	$x_1^3 = x_1$	$\frac{8}{11}$		1	0	$-\frac{2}{11}$	0	$\frac{1}{11}$	$\frac{2}{11}$	0	$-\frac{1}{11}$
$-M$	$x_2^3 = y_2$	$\frac{28}{11}$		0	0	$\frac{4}{11}$	-1	$\frac{20}{11}$	$-\frac{4}{11}$	1	$-\frac{20}{11}$
2	$x_3^3 = x_2$	$\frac{18}{11}$		0	1	$\frac{1}{11}$	0	$-\frac{6}{11}$	$-\frac{1}{11}$	0	$\frac{6}{11}$
	$z(x)$	$-\frac{28}{11}M + \frac{76}{11}$		0	0	$-\frac{4}{11}M - \frac{8}{11}$	$-M$	$-\frac{20}{11}M - \frac{7}{11}$	$\frac{15}{11}M + \frac{8}{11}$	0	$\frac{31}{11}M + \frac{7}{11}$

- on rentre x_5,
- on sort y_2 car $\frac{7}{5} < 8$ (l'autre rapport étant négatif on le compte pas),

$\downarrow$

			c	5	2	0	0	0	$-M$	$-M$	$-M$
c^J	variables de base			x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3
5	$x_1^4 = x_1$	$\frac{3}{5}$		1	0	$-\frac{1}{5}$	$\frac{1}{20}$	0	$\frac{1}{5}$	$-\frac{1}{20}$	0
0	$x_2^4 = x_5$	$\frac{7}{5}$		0	0	$\frac{1}{5}$	$-\frac{11}{20}$	1	$-\frac{1}{5}$	$\frac{11}{20}$	-1
2	$x_3^4 = x_2$	$\frac{12}{5}$		0	1	$\frac{1}{5}$	$-\frac{3}{10}$	0	$-\frac{1}{5}$	$\frac{3}{10}$	0
	$z(x)$	$\frac{39}{5}$		0	0	$-\frac{3}{5}$	$-\frac{7}{20}$	0	$M + \frac{3}{5}$	$M + \frac{7}{20}$	M

A ce stade, il est clair que les variables artificielles ne pourront plus entrer en base. Elles ont un coût dépendant de M et toutes les autres variables ont un coût indépendant de M. On peut donc les retirer et procéder sur le tableau réduit suivant (c'est surtout pour des raisons de clarté) :

			c	5	2	0	0	0
c^J	variables de base			x_1	x_2	x_3	x_4	x_5
5	$x_1^4 = x_1$	$\frac{3}{5}$		1	0	$-\frac{1}{5}$	$\frac{1}{20}$	0
0	$x_2^4 = x_5$	$\frac{7}{5}$		0	0	$\frac{1}{5}$	$-\frac{11}{20}$	1
2	$x_3^4 = x_2$	$\frac{12}{5}$		0	1	$\frac{1}{5}$	$-\frac{3}{10}$	0
	$z(x)$	$\frac{39}{5}$		0	0	$-\frac{3}{5}$	$-\frac{7}{20}$	0

- on rentre x_3,
- on sort x_5,

$\downarrow$

			c	5	2	0	0	0
c^J	variables de base			x_1	x_2	x_3	x_4	x_5
5	$x_1^5 = x_1$	2		1	0	0	$-\frac{1}{2}$	1
0	$x_2^5 = x_3$	7		0	0	1	$-\frac{11}{4}$	5
2	$x_3^5 = x_2$	1		0	1	0	$\frac{1}{4}$	-1
	$z(x)$	12		0	0	0	-2	3

Il ne reste probablement qu'une étape, faisons-la :

- on rentre x_4,
- on sort x_2, c'est la seule valeur positive,

$\downarrow$

			c	5	2	0	0	0
c^J	variables de base			x_1	x_2	x_3	x_4	x_5
5	$x_1^6 = x_1$	4		1	2	0	0	-1
0	$x_2^6 = x_3$	18		0	11	1	0	-6
0	$x_3^6 = x_4$	4		0	4	0	1	-4
	$z(x)$	20		0	8	0	0	-5

Dommage, on a encore un point pivot à faire rentrer (x_5 a valeur négative) donc on aimerait continuer de dérouler l'algorithme mais on a pas de critère pour trouver quelle variable sortir de base (en effet, toutes les valeurs dans la colonne du pivot sont négatives).

On conclut donc que l'espace des solutions est non borné. C'est-à-dire qu'il n'existe pas « un couple maximum » car on peut toujours en trouver un plus grand. Donc il n'y a pas un unique couple (x_1, x_2) qui maximise z.

2. Résolvons maintenant le problème P_θ par la méthode à deux phases.

Commençons par modifier le problème comme il faut pour la phase 1 :

$$P'_\theta = \begin{cases} \max z' = 0x_1 + 0x_2 + 0x_3 + 0x_4 + 0x_5 - (y_1 + y_2 + y_3) \\ 6x_1 + x_2 - x_3 + y_1 = 6 \\ 4x_1 + 4x_2 - x_4 + y_2 = 12 \\ x_1 + 2x_2 - x_5 + y_3 = 4 \\ x_i \geq 0, y_j \geq 0, \quad \forall i \in \{1, \ldots, 5\}, \forall j \in \{1, 2, 3\}. \end{cases}$$

Ensuite on construit notre tableau du simplexe :

		c	0	0	0	0	0	−1	−1	−1
c^J	variables de base		x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3
−1	$x_1^1 = y_1$	6	6	1	−1	0	0	1	0	0
−1	$x_2^1 = y_2$	12	4	4	0	−1	0	0	1	0
−1	$x_3^1 = y_3$	4	1	2	0	0	−1	0	0	1
	$z'(x)$	−22	−11	−7	1	1	1	0	0	0

et on déroule l'algorithme :

- on rentre x_1,
- on sort y_1 car $1 < 3 < 4$,

↓

		c	0	0	0	0	0	−1	−1	−1
c^J	variables de base		x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3
0	$x_1^2 = x_1$	1	1	$\frac{1}{6}$	$-\frac{1}{6}$	0	0	$\frac{1}{6}$	0	0
−1	$x_2^2 = y_2$	8	0	$\frac{10}{3}$	$\frac{4}{6}$	−1	0	$-\frac{4}{6}$	1	0
−1	$x_3^2 = y_3$	3	0	$\frac{11}{6}$	$\frac{1}{6}$	0	−1	$-\frac{1}{6}$	0	1
	$z'(x)$	−11	0	$-\frac{31}{6}$	$-\frac{5}{6}$	1	1	$\frac{11}{6}$	0	0

- on rentre x_2,
- on sort y_3,

↓

		c	0	0	0	0	0	−1	−1	−1
c^J	variables de base		x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3
0	$x_1^3 = x_1$	$\frac{8}{11}$	1	0	$-\frac{2}{11}$	0	$\frac{1}{11}$	$\frac{2}{11}$	0	$-\frac{1}{11}$
−1	$x_2^3 = y_2$	$\frac{28}{11}$	0	0	$\frac{4}{11}$	−1	$\frac{20}{11}$	$-\frac{4}{11}$	1	$-\frac{20}{11}$
0	$x_3^3 = x_2$	$\frac{18}{11}$	0	1	$\frac{1}{11}$	0	$-\frac{6}{11}$	$-\frac{1}{11}$	0	$\frac{6}{11}$
	$z'(x)$	$-\frac{28}{11}$	0	0	$-\frac{4}{11}$	−1	$-\frac{20}{11}$	$\frac{15}{11}$	0	$\frac{31}{11}$

- on rentre x_5,
- on sort y_2,

↓

			c	0	0	0	0	0	−1	−1	−1
c^J	variables de base			x_1	x_2	x_3	x_4	x_5	y_1	y_2	y_3
0	$x_1^4 = x_1$	$\frac{3}{5}$		1	0	$-\frac{1}{5}$	$\frac{1}{20}$	0	$\frac{1}{5}$	$-\frac{1}{20}$	0
0	$x_2^4 = x_5$	$\frac{7}{5}$		0	0	$\frac{1}{5}$	$-\frac{11}{20}$	1	$-\frac{1}{5}$	$\frac{11}{20}$	−1
0	$x_3^4 = x_2$	$\frac{12}{5}$		0	1	$\frac{1}{5}$	$-\frac{3}{10}$	0	$-\frac{1}{5}$	$\frac{3}{10}$	0
	$z'(x)$	0		0	0	0	0	0	1	1	1

A ce stade, toutes les variables artificielles sont sorties de base et la valeur optimale de z' est nulle. On peut donc passer à la phase 2 en retirant les variables artificielles du tableau avec la fonction objectif originale :

			c	5	2	0	0	0
c^J	variables de base			x_1	x_2	x_3	x_4	x_5
5	$x_1^4 = x_1$	$\frac{3}{5}$		1	0	$-\frac{1}{5}$	$\frac{1}{20}$	0
0	$x_2^4 = x_5$	$\frac{7}{5}$		0	0	$\frac{1}{5}$	$-\frac{11}{20}$	1
2	$x_3^4 = x_2$	$\frac{12}{5}$		0	1	$\frac{1}{5}$	$-\frac{3}{10}$	0
	$z(x)$	$\frac{39}{5}$		0	0	$-\frac{3}{5}$	$-\frac{7}{20}$	0

- on rentre x_3,
- on sort x_5,

↓

			c	5	2	0	0	0
c^J	variables de base			x_1	x_2	x_3	x_4	x_5
5	$x_1^5 = x_1$	2		1	0	0	$-\frac{1}{2}$	1
0	$x_2^5 = x_3$	7		0	0	1	$-\frac{11}{4}$	5
2	$x_3^5 = x_2$	1		0	1	0	$\frac{1}{4}$	−1
	$z(x)$	12		0	0	0	−2	3

- on rentre x_4,
- on sort x_2, c'est la seule valeur positive,

↓

			c	5	2	0	0	0
c^J	variables de base			x_1	x_2	x_3	x_4	x_5
5	$x_1^6 = x_1$	4		1	2	0	0	−1
0	$x_2^6 = x_3$	18		0	11	1	0	−6
0	$x_3^6 = x_4$	4		0	4	0	1	−4
	$z(x)$	20		0	8	0	0	−5

Il ne resterait plus qu'à faire rentrer x_5 pour obtenir la solution optimale mais on a aucune variable qui correspond à un critère d'entrée, elles sont toutes negatives. Donc x_5 ne peut pas rentrer en base et le problème est non borné.

Exercice 2 Dualité.

Considérez le programme linéaire le plus général envisageable donné ci-dessous :

$$\begin{cases} \min z = c_1 x_1 + c_2 x_2 \\ A_{11} x_1 + A_{12} x_2 \leq b_1 \\ A_{21} x_1 + A_{22} x_2 = b_2 \\ x_i \geq 0, \quad \forall i \in \{1, 2\} \end{cases}$$

où A est une matrice $(m_1 + m_2) \times (n_1 + n_2)$ et $c, x \in \mathbb{R}^{n_1+n_2}$ et $b \in \mathbb{R}^{m_1+m_2}$.
Caractériser le dual.

Solution.
Pour chaque contrainte de type $\leq$ on associe une variable duale $y_i \geq 0$ et pour chaque contrainte de type $=$ on associe une variable duale y_j pas forcément positive. Le dual est donc :

$$\begin{cases} \max w = b_1 y_1 + b_2 y_2 \\ A_{11} y_1 + A_{21} y_2 \geq c_1 \\ A_{12} y_1 + A_{22} y_2 = c_2 \\ y_1 \geq 0 \end{cases}$$

Au départ on avait m_1 contraintes de type $\leq$ et m_2 contraintes de type $=$, donc le dual a $m_1 + m_2$ variables (c'est le cas car b_1 (respectivement b_2) est un vecteur de taille m_1 (respectivement m_2)). De plus, on avait $n_1 + n_2$ variables dans le primal, donc le dual a $n_1 + n_2$ contraintes (c'est le cas car A est une matrice $(m_1 + m_2) \times (n_1 + n_2)$, donc c est un vecteur de taille $n_1 + n_2$).

Exercice 3 Ensemble convexe.
Soit C_1 et C_2 deux convexes de $\mathbb{R}^{m+n}$. Montrer que l'ensemble

$$C = \{(x, y_1 + y_2) \mid x \in \mathbb{R}^m, y_1 \in \mathbb{R}^n, y_2 \in \mathbb{R}^n, (x, y_1) \in C_1, (x, y_2) \in C_2\}$$

est également convexe.

Solution.
Soient (x^1, y^1) et (x^2, y^2) deux points de C. Par définition de C, il existe $y_1^1, y_2^1, y_1^2, y_2^2$ tels que

$$(x^1, y_1^1) \in C_1, \quad (x^1, y_2^1) \in C_2, \quad (x^2, y_1^2) \in C_1, \quad (x^2, y_2^2) \in C_2,$$

et

$$y^1 = y_1^1 + y_2^1, \quad y^2 = y_1^2 + y_2^2.$$

Soit $\lambda \in [0, 1]$. Considérons le point

$$(x^\lambda, y^\lambda) = \lambda(x^1, y^1) + (1 - \lambda)(x^2, y^2).$$

On a

$$\begin{aligned} x^\lambda &= \lambda x^1 + (1 - \lambda) x^2, \\ y^\lambda &= \lambda y^1 + (1 - \lambda) y^2 \\ &= \lambda(y_1^1 + y_2^1) + (1 - \lambda)(y_1^2 + y_2^2) \\ &= (\lambda y_1^1 + (1 - \lambda) y_1^2) + (\lambda y_2^1 + (1 - \lambda) y_2^2). \end{aligned}$$

Notons $y_1^\lambda = \lambda y_1^1 + (1 - \lambda) y_1^2$ et $y_2^\lambda = \lambda y_2^1 + (1 - \lambda) y_2^2$. Par convexité de C_1 et C_2, on a

$$(x^\lambda, y_1^\lambda) \in C_1, \quad (x^\lambda, y_2^\lambda) \in C_2.$$

Ainsi, par définition de C, on a $(x^\lambda, y^\lambda) \in C$. Donc C est convexe.

Exercice 4 Modélisation et dualité.
Considérons un problème d'affectation avec m jobs et n travailleurs ($n \geq m$). Chaque job doit être affecté à exactement un travailleur. Soit p_{ij} le rendement obtenu si on affecte le job i au travailleur j, où $i \in \{1, \ldots, m\}$ et $j \in \{1, \ldots, n\}$. On cherche une affectation qui maximise le rendement total.

1. Donner le programme linéaire.
2. Donner la formulation du dual de ce problème.

Solution.

1. On introduit les variables x_{ij} qui valent 1 si le job i est affecté au travailleur j, et 0 sinon. Le programme linéaire s'écrit alors :

$$P_0 = \begin{cases} \max z = \sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij} x_{ij} & (1) \\ \sum_{j=1}^{n} x_{ij} = 1, \quad \forall i \in \{1, \ldots, m\} & (2) \\ \sum_{i=1}^{m} x_{ij} \leq 1, \quad \forall j \in \{1, \ldots, n\} & (3) \\ x_{ij} \in \{0, 1\}, \quad \forall i, j & (4) \end{cases}$$

   où les contraintes correspondent à :

   (1) on maximise le rendement total
   (2) chaque job est affecté (à un travailleur)
   (3) chaque travailleur a au plus un job
   (4) les jobs ne sont pas partiellement affectés

2. Pour obtenir la formulation du dual, on introduit les variables duales u_i (une par job) et v_j (une par travailleur). Le dual s'écrit alors :

$$D_0 = \begin{cases} \min w = \sum_{i=1}^{m} u_i + \sum_{j=1}^{n} v_j & (5) \\ u_i + v_j \geq p_{ij}, \quad \forall i, j & (6) \\ v_j \geq 0, \quad \forall j & (7) \end{cases}$$

   où les contraintes correspondent à :

   (5) on minimise la somme des coûts
   (6) les coûts doivent couvrir les rendements
   (7) les variables sont positives

   c'est une autre manière intéressante de voir le problème.

Exercice 5 Programmation linéaire : Farkas.
Considérons le programme linéaire suivant, qui dépend de $\varepsilon \in \mathbb{R}$:

$$\begin{cases} \min z = 4x_1 - 2x_2 \\ x_2 \leq 3 \\ \varepsilon x_1 + (2 - \varepsilon) x_2 \leq 4 \\ x_i \geq 0, \quad \forall i \in \{1, 2\} \end{cases}$$

1. Montrer que le problème est réalisable $\forall \varepsilon \in \mathbb{R}$.
2. Pour quelles valeurs de ε la valeur optimale est-elle non bornée ?

Solution.

1. On remarque tout de suite que le point $x = (0, 0)$ satisfait les contraintes du programme linéaire pour toute valeur de ε. Donc le problème est réalisable pour tout $\varepsilon \in \mathbb{R}$.
2. Pour étudier la borne de la valeur optimale, on peut regarder le dual du programme linéaire :

$$\begin{cases} \max w = 3y_1 + 4y_2 \\ \varepsilon y_2 \geq 4 \\ y_1 + (2 - \varepsilon)y_2 \geq -2 \\ y_i \geq 0 \end{cases}$$

D'après le théorème de dualité de la programmation linéaire (4.4.2, n. 3), si le dual est irréalisable alors le primal est non borné. Étudions donc l'irréalisabilité du dual :

- Si $\varepsilon \leq 0$, la première contrainte du dual ne peut pas être satisfaite car $y_2 \geq 0$ et donc le dual est irréalisable. Donc le primal est non borné.
- Si $\varepsilon > 0$, la première contrainte du dual impose $y_2 \geq \frac{4}{\varepsilon} > 0$. En remplaçant dans la deuxième contrainte on obtient :

$$y_1 + (2 - \varepsilon)\frac{4}{\varepsilon} \geq -2 \iff y_1 \geq -2 - \frac{8}{\varepsilon} + 4 = 2 - \frac{8}{\varepsilon}.$$

Comme $\varepsilon > 0$, on a $2 - \frac{8}{\varepsilon} < 2$. Donc on peut toujours choisir y_1 suffisamment grand pour satisfaire cette contrainte. Donc le dual est réalisable et le primal est donc borné.

En conclusion, la valeur optimale est non bornée pour $\varepsilon \leq 0$.

Exercice 6 Résolution numérique.
Résoudre le programme linéaire suivant par la méthode Primal-Dual :

$$\text{Primal} = \begin{cases} \min z(x_1, x_2, x_3) = 2x_1 + x_2 + 2x_3 + 8x_4 \\ 2x_1 - x_2 + 3x_3 - 2x_4 = 3 \\ -x_1 + 3x_2 - 4x_3 = 1 \\ x_i \geq 0, \quad \forall i \in \{1, 2, 3, 4\} \end{cases}$$

Solution.

2 Partie pratique

Exercice 7 Résolution d'un programme linéaire avec Julia.

1. Dans un premier temps nous allons regarder quelques commandes de base en Julia.
   (a) Pour lancer Julia :

```
julia
```

   (b) Introduction à la syntaxe. La syntaxe est assez simple :

```
a = 5
b = 67.67e42
```

```
c = a + 3 * b ^ 4
m = "Bonjour"
d = sin(atan(3))
toi = "Rosa"
println("Bonjour, $toi !")
e1 = [12; 34; 45]
e2 = [12 34 45]
d = true
f = [12.0; 34; "Youhou !"]
B = [1 2; 1 3; 4 5]
```

(c) les fonctions sont bien définies :

```
function add(a, b)
    return a + b
end

f(x) = 42 * x^3

add(4, 2)
f(a)
```

Renseignements obligatoires sur la syntaxe de Julia peuvent être trouvés sur les pages suivantes :

- `https://julialang.org/`
- `https://zestedesavoir.com/articles/78/a-la-decouverte-de-julia/`

(d) Le package `JuMP` et le solveur `Cbe JuMP` sont des langages de modélisation pour l'optimisation mathématique intégré dans Julia. Pour l'utiliser il faut d'abord ajouter le package associé :

```
using Pkg
Pkg.add("JuMP")
Pkg.add("Cbc")
```

Julia est un langage de modélisation et ne possède pas un solveur de programmation mathématique intégré. `CPLEX` est un solveur de programmation mathématique de haute performance, puissant pour la programmation linéaire, la programmation en nombres entiers et la programmation quadratique. Mais le solveur `CPLEX` n'est pas gratuit et requiert une license (facile à obtenir pour les étudiants mais le système est bridé). Nou sutiliserons donc plutôt `Cbc` qui est un solveur de programmation mathématique open-source moins puissant intégré dans Julia. Dans la suite, nous présentons les commandes pour utiliser `Cbc`. Pour utiliser un solveur à partir de Julia il faut ajouter le package associé.

(e) Création du modèle :
Utiliser les commandes présentées dans cette section pour définir dans Julia un modèle associé au programme linéaire P_1. Les variables x_1 et x_2 représentent respectivement la quantité de deux types différents de yaourt produits. La fonction objectif maximise le profit de la production :

```
Z = max 4x + 5y
2x+y <= 800
x+2y <= 700
y <= 300
x,y >= 0
```

(f) Packages :

```
using JuMP
using Cbc
```

(g) Définition du modèle :

```
m = Model(Cbc.Optimizer)
```

(h) Définition des variables, exemples de définition de variables :

```
@variable(m, x) # No bounds
@variable(m, x >= lb) # Lower bound only (note: 'lb <= x' is not valid)
@variable(m, lb <= x <= ub) # Both lower and upper bounds
@variable(m, x == fixedval) # Fixed variable
@variable(m, x[1:M, 1:N]) # Un vecteur de variables
@variable(m, x[1:5], Bin) # Un vecteur de variabes binaires
```

Ajouter à votre modèle les contraintes de P_1.

(i) Pour résoudre le modèle, il faut exécuter :

```
optimize!(m)
```

La valeur de la solution optimale du modèle, le temps d'exécution et la solutoin optimale, peuvent etre récupérés en utilisant les commandes suivantes :

```
Objective value(m)
    solve time(m)
    value.(x)
    value(x[i])
```

Voici un exemple, en supposant que x est le vecteur de variables, profit et weight sont des vecteurs de coefficients :

```
println("Objective is: ", objective_value(m))
println("Solve time = ", solve_time(m))
println("Solution is:")
for i = 1:5
    print("x[$i] = ", value(x[i]))
    println(", p[$i]/w[$i] = ", profit[i]/weight[i])
end
```

Nous pouvons à tout moment visualiser le modèle en utilisant la commande suivante :

```
print(m)
```

Nous pouvons aussi écrire la totalité du modèle dans un fichier nommé `ex1.jl` et exéctuer la commande suivante :

```
include("ex1.jl")
```

(j) Lecture de fichier :
La commande suivante place le contenu du fichier dans un tableau :

```
using DelimitedFiles
data = readdlm("data.txt")
```

On peut ensuite extraire une partie du tableau. Par exemple, la commande suivante place les lignes 5 à 10 de la seconde colonne de data dans le vecteur `cost` :

```
cost = data[5:10, 2]
```

(k) Pour afficher le nombre de noeuds dans un arbre :

```
XXX.get(model, XXX.NodeCount())
```

2. Sur le problème de **Bin Packing Problem**.
On considère le problème P_1 et sa variante P_2 : $P_1 : x_{ij} \in \{0,1\}$ avec $x_{ij} = 1$ si et seulement si l'objet i est placé dans la boîte j et $y_j \in \{0,1\}$ avec $y_j = 1$ si et seulement si la boîte j contient au moins un objet. On a n objets et m boîtes.

$$PNLE_{\text{bin packing}} = \begin{cases} \min z = \sum_{j=1}^{m} y_j \\ \sum_{j=1}^{m} x_{ij} = 1, \quad \forall i \in \{1,\dots,n\} \\ \sum_{i=1}^{n} a_i x_{ij} \leq W, \quad \forall j \in \{1,\dots,m\} \\ x_{ij} \leq y_j, \quad \forall i,j \\ x_{ij} \in \{0,1\}, \quad \forall i,j \\ y_j \in \{0,1\}, \quad \forall j \end{cases}$$

Noter que l'on peut combiner les deux contraintes $\sum_{i=1}^{n} a_i x_{ij} \leq W$ et $x_{ij} \leq y_j$ en une seule contrainte :

$$\sum_{i=1}^{n} a_i x_{ij} \leq W y_j, \quad \forall j \in \{1,\dots,m\}$$

c'est ce qu'on utilise dans P_2

(a) Considérer les instances Bin Packing Problem se trouvant à l'adresse `https://site.unibo.it/operations-research/en/research/bpplib-a-bin-packing-problem-library` (prendre Falkenaeur et Scholl).
A noter que le format est le suivant :
- n : nombre d'objets
- c : capacité des boîtes
- w_j : poids de l'objet j pour $j \in \{1,\dots,n\}$

Comparer les valeurs des relaxations linéaires des deux modèles précédents pour les instances précédentes. Comparer le temps de calcul pour obtenir les solutions optimales.

(b) Ajouter des coupes pour accélérer les temps de calcul. Reprendre les tests avec les jeux de données.

Solution.

Exercice 8 Résolution du pire cas : problème Klee-Minty.
Soit le problème linéaire suivant :

$$\begin{cases} \max z = \sum_{j=1}^{n} 10^{n-j} x_j \\ 2\sum_{j=1}^{i-1} 10^{i-j} x_j + x_i \leq 100^{i-1}, \quad \forall i \in \{1,\dots,n\} \\ x_j \geq 0, \quad \forall j \in \{1,\dots,n\} \end{cases}$$

1. Résoudre pour $n = 1,2,3$ avec la méthode des tableaux.
2. Utiliser Julia pour résoudre le problème avec $n \geq 4$. Pour cela vous procéderez à des tests. Soit T la borne de temps maximum autorisé (par exemple 60 secondes).
   (a) Faire varier $n \in \{5,10,15,200,30,50,100,150,200,250,\dots\}$ tout en respectant la borne T.
   (b) Afficher les temps de calcul en fonction de n.
   (c) Vérifier que les temps de calcul sibt similaires à $O(2^n)$

Solution.

Exercice 9 Formulation : trajets d'un convoyeur dans un entrepôt.
Dans un grand entrepôt, un convoyeur électrique est utilisé pour acheminer les arrivages (sous forme de palettes) de la zone d'entrée jusqu'à leur place propre de stockage dans l'entrepôt. Ayant placé une palette dans le stock, le convoyeur peut ensuite aller puiser dans le stock une palette d'un article demandé pour la transporter à la zone de sortie. Il retourne ensuite à la zone d'entrée et recommence avec une nouvelle palette à placer dans le stock.
Appelons a_i (avec $i = 1, \ldots, m$) le nombre de palettes à transporter de la zone d'entrée à la zone i de stockage et b_j (avec $j = 1, \ldots, n$) le nombre de palettes à transporter de la zone j de stockage à la zone de sortie. Soit enfin t_{ij} le temps nécessaire pour aller de la zone i à la zone j (où $a_i > 0$ et $b_j > 0$). Nous pouvons admettre que $\{i \mid a_i > 0\} \cap \{j \mid b_j > 0\} = \varnothing$ et que $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$ (sans restriction de généralité).
Déterminer dans quel ordre les palettes doivent être transportées dans l'entrepôt de manière à minimiser le temps total de transport.

Solution.