Intelligence Artificielle 1

Cours 9 et 10 - Prog. Linéaire

Table des matières

Qu'est-ce que la Programmation Linéaire?

La PL est une méthode d'optimisation permettant de déterminer la solution optimale d'un problème formulé avec des équations linéaires.

Cours basé sur les diapositives de Prof. Vincent Leduc.

Source : `https://coin-or.github.io/pulp/main/index.html`

Objectifs du cours sur la PL

- Décrire un PL; le processus d'optimisation et sa formulation mathématique
  - Les variables de décisions
  - La fonction objective
  - Les contraintes
- Résoudre un PL
  - par un graphique
  - avec une librairie Python (Pulp)
- Intégrer un PL dans une application

Fonction linéaire

On appelle fonction linéaire toute fonction pouvant s'écrire sous la forme régulière suivante :

$$f(x) = ax + b \tag{1}$$

On appelle contrainte une fonction linéaire écrite sous la forme d'une équation :

$$-ax + y = b \tag{2}$$

Fonction linéaire

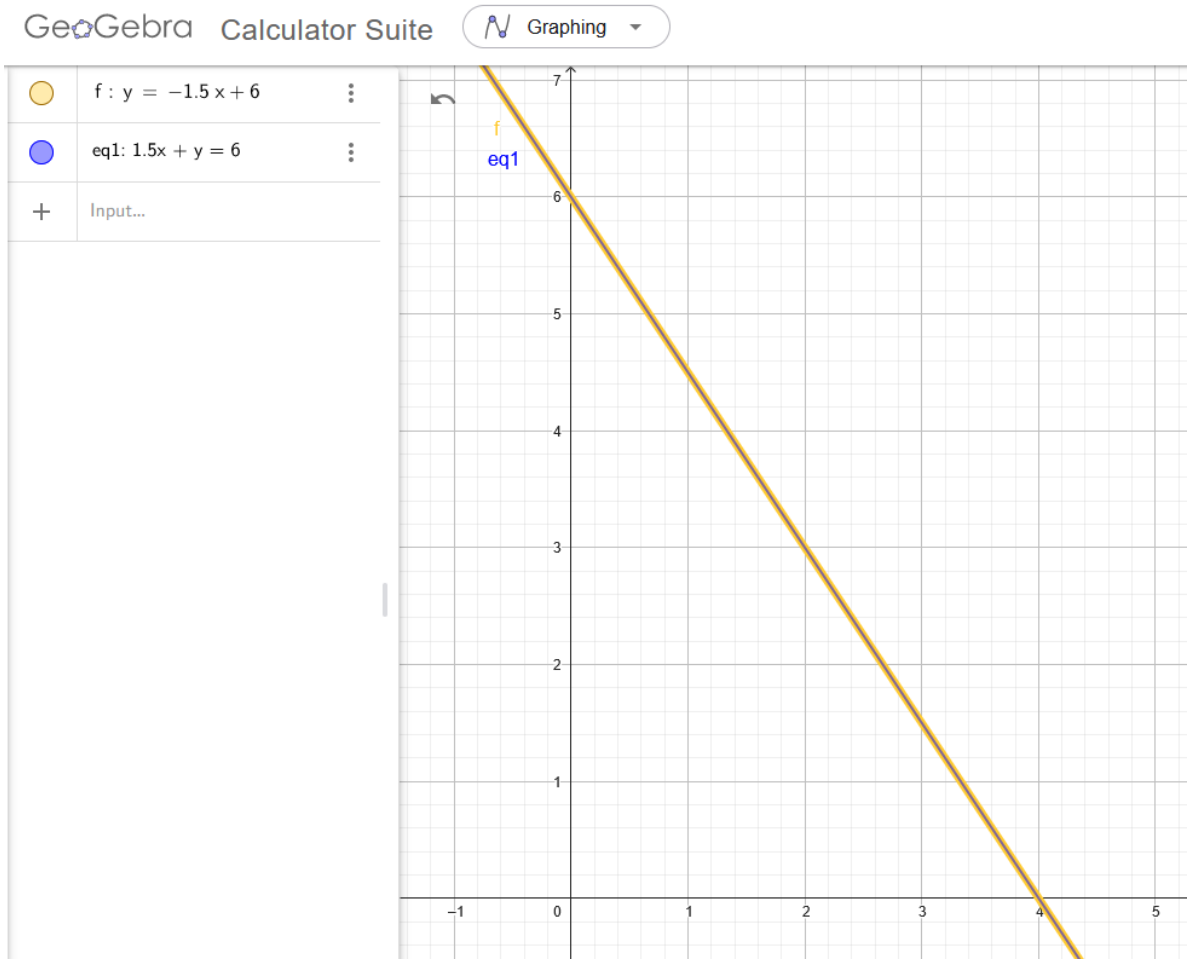


Figure: Représentation graphique de la fonction

Cas d'utilisation de PL

La PL est utilisé dans la gestion d'entreprise pour résoudre des problèmes de planification, de gestion de production, de transport et bien plus encore. D'une manière générale, il s'agit de maximiser les profits en minimisant les coûts.

Cas d'utilisation de PL

Nous verrons un cas dans le contexte de l'IoT; maximiser la précision de mesure de qualité de l'air par des capteurs.

En utilisant la PL, beaucoup de problèmes peuvent être résolus de manière optimale.

Dans ce cas précis, ainsi que d'autres cas où nous ne cherchons pas à prédire, mais bien à avoir une solution optimale parmi les données, **l'intelligence artificielle, les réseaux de neurones, l'apprentissage profond, et les LLM ne sont pas le meilleur choix.**

Bien que la PL garantit une solution optimale (parmi les contraintes), celle-ci peut utiliser beaucoup de ressources.

Processus d'optimisation

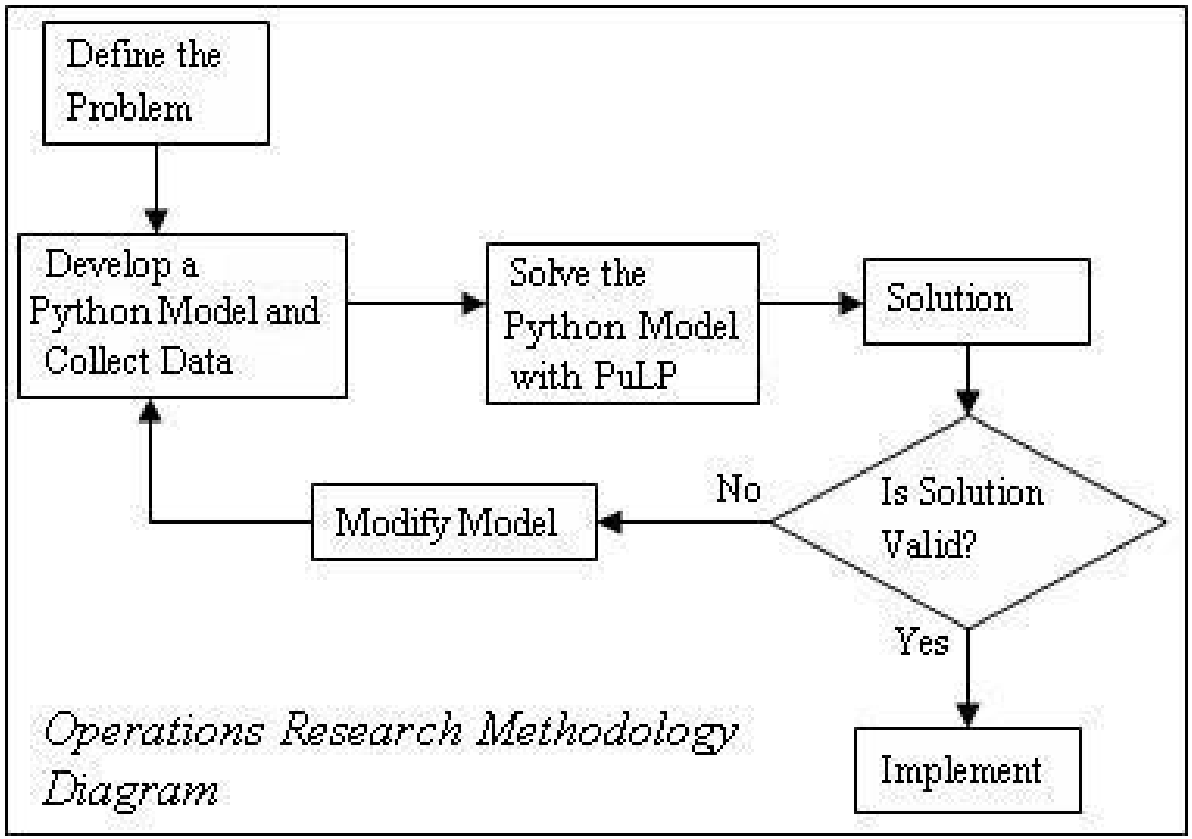


Figure:
`https://coin-or.github.io/pulp/main/the_optimisation_process.html#the-optimisation-process`

Formulation mathématique

La formulation d'un problème à résoudre par Programmation Linéaire est composée de 3 éléments :

- Des variables de décision (unité)
- Une fonction objective linéaire (min/max)
- Des contraintes représentées par des égalités ou inégalités linéaires

Source :
`https://coin-or.github.io/pulp/main/the_optimisation_process.html#formulating-the-mathematical-program`

Variables de décision

Les variables x_j (nombre réel ou entier) sont les variables de décision à déterminer sous contraintes (s.c.) et peuvent être bornées

$$
\begin{aligned}
Max\ (Min) \quad & z = \sum_{j=1}^{n} c_j x_j \\
\text{s.c.} \quad & \sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad i \in I \subseteq \{1, \ldots, m\} \\
& \sum_{j=1}^{n} a_{kj} x_j \geq b_k \quad k \in K \subseteq \{1, \ldots, m\} \\
& \sum_{j=1}^{n} a_{rj} x_j = b_r \quad r \in R \subseteq \{1, \ldots, m\}
\end{aligned}
$$

Les ensembles I, K et R sont disjoints et $I \cup K \cup R = \{1, ..., m\}$.

Fonction objective linéaire (min/max)

La fonction objective est la fonction que l'on veut optimiser en minimisant ou maximisant sa valeur. t.q. c_j une valeur connue dans le problème et x_j une variable recherchée

$$Max/Min\ Z = \sum_{j=i}^{n} c_j x_j \qquad (3)$$

Contraintes (égalités/inégalités linéaires)

La fonction objective est optimisée en déterminant les n valeurs de x_j qui respectent m contraintes linéaires.

$$
\begin{aligned}
Max\ (Min) \quad & z = \sum_{j=1}^{n} c_j x_j \\
\text{s.c.} \quad & \sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad i \in I \subseteq \{1, \ldots, m\} \\
& \sum_{j=1}^{n} a_{kj} x_j \geq b_k \quad k \in K \subseteq \{1, \ldots, m\} \\
& \sum_{j=1}^{n} a_{rj} x_j = b_r \quad r \in R \subseteq \{1, \ldots, m\}
\end{aligned}
$$

Les ensembles I, K et R sont disjoints et $I \cup K \cup R = \{1, ..., m\}$.

Exemple synthaxe simple avec PuLP

Voici un exemple simple avec la syntaxe PuLP pour concrétiser en code un problème de Programmation Linéaire (PL).

$$\min_{0 \leq x \leq 3,\, 0 \leq y \leq 1} z = -4x + y \quad (4)$$

$$x + y \leq 2 \quad (5)$$

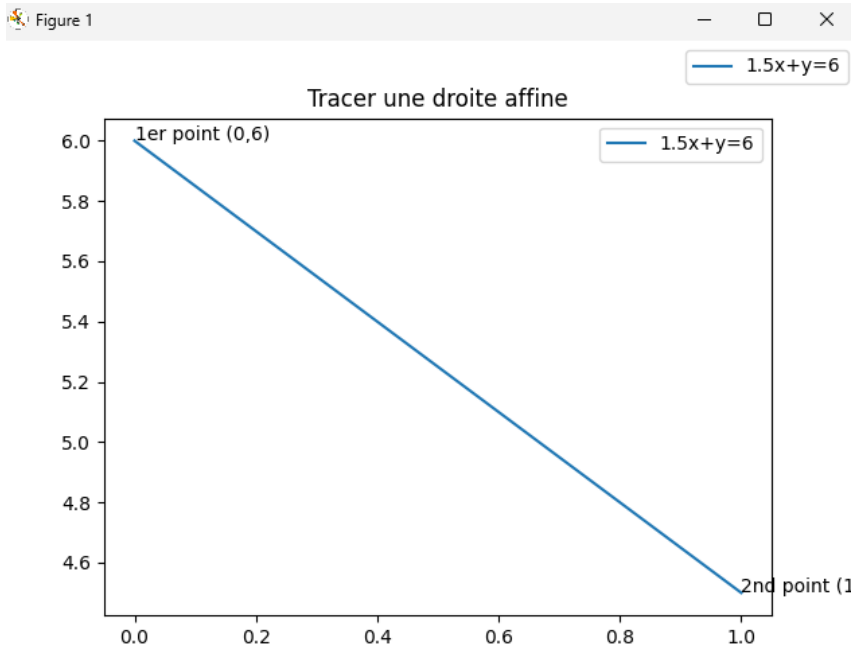
Exemple synthaxe simple avec PuLP

Listing: https://coin-or.github.io/pulp/main/includeme.html

```python
# Steve Levesque, All rights reserved
from pulp import *

# a) Define the LP global variable
prob: LpProblem = LpProblem("Demo1Pulp", LpMinimize)
# b) Create decision variables (x and y)
x: LpVariable = LpVariable("x", 0, 3)
y: LpVariable = LpVariable("y", 0, 1)

# c) Create a constraint (simple/direct form)
prob += x + y <= 2
# d) Create the objective function (simple/direct form)
prob += -4 * x + y

# f) Show the status (Optimal or not)
print(LpStatus[prob.solve(PULP_CBC_CMD(msg=False))])
# h) Show the final value of the optimal objective function
print(f"Optimal objective value (Z) : {prob.objective.value()}")
```

Matplotlib

Il est possible de visualiser un problème de programmation linéaire (PL) graphiquement grâce à Matplotlib.



Exemple synthaxe simple avec PuLP

```
1  import matplotlib.pyplot as plt
2  from typing import List
3
4  x: List[int] = [0, 1]
5  y: List[float] = [6, 4.5]
6
7  # the figure (window) and axe in which we draw the curves
8  fig, ax = plt.subplots()
9
10 ax.plot(x, y, label="1.5x+y=6")  # legend title
11 fig.legend(loc="upper right")    # legend in figure
12 ax.legend(loc="upper right")     # legend in relation towards axe
13
14 plt.title("Tracer une droite affine")
15 plt.text(0, 6, '1er point (0,6)')
16 plt.text(1, 4.5, '2nd point (1,4.5)')
17 plt.show()
```

Matplotlib - Concepts de base

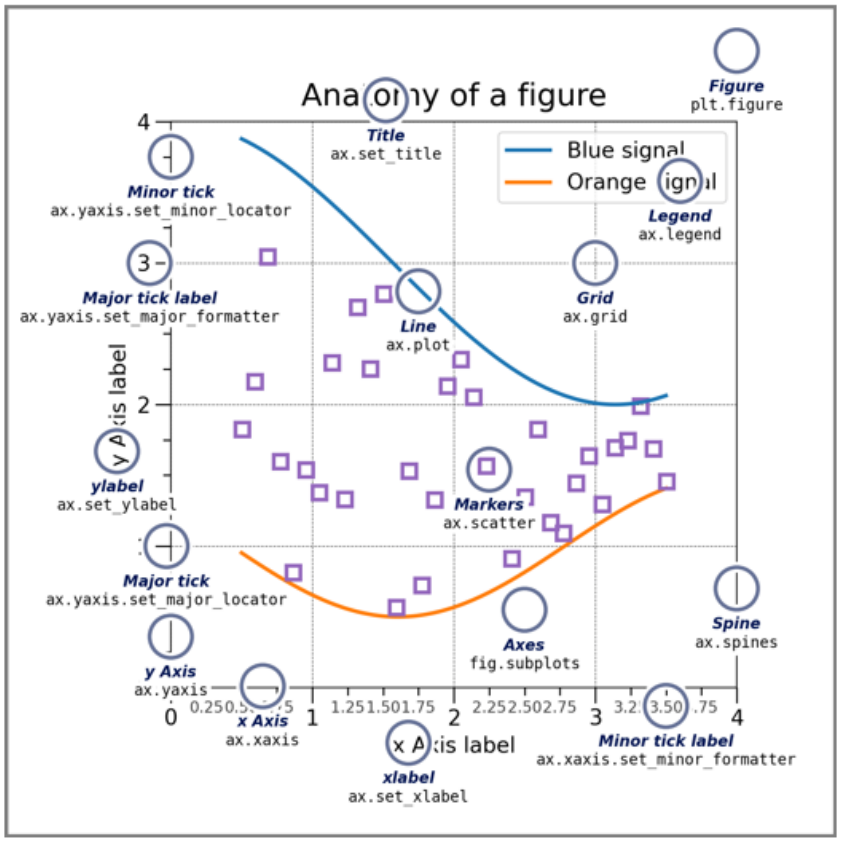


Figure: `https://matplotlib.org/stable/users/explain/quick_start.html`

Résolution graphique

Les étapes à suivre pour la résolution graphique d'un problème de PL :

- Dessiner la zone de faisabilité
  - Bornes des variables de décision
  - Contraintes
- Dessiner une fonction objective arbitraire
- La solution est à l'intersection entre la zone de faisabilité et une droite affine parallèle à la fonction objective arbitraire

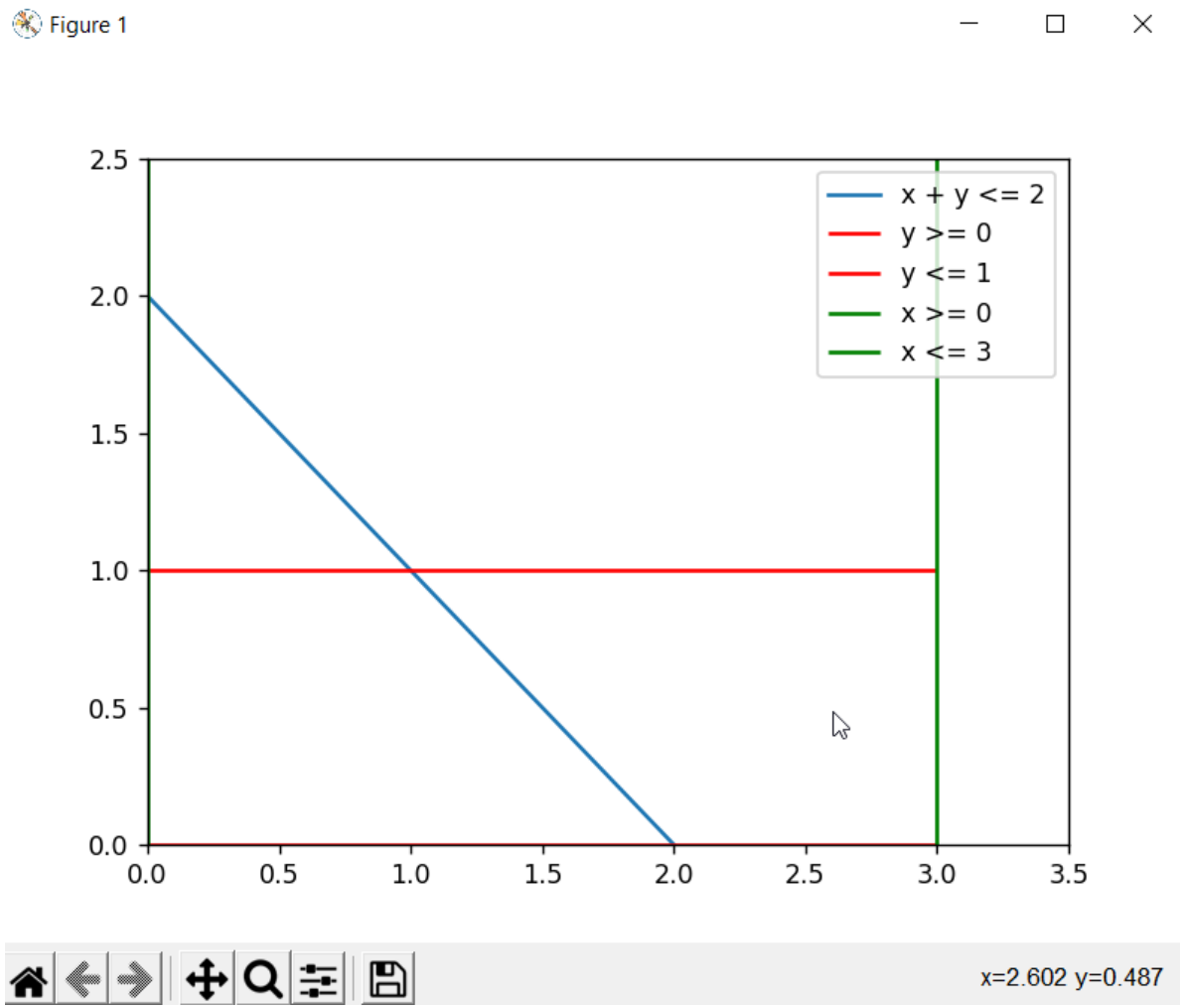
Résolution graphique

Retour sur l'exemple précédent :

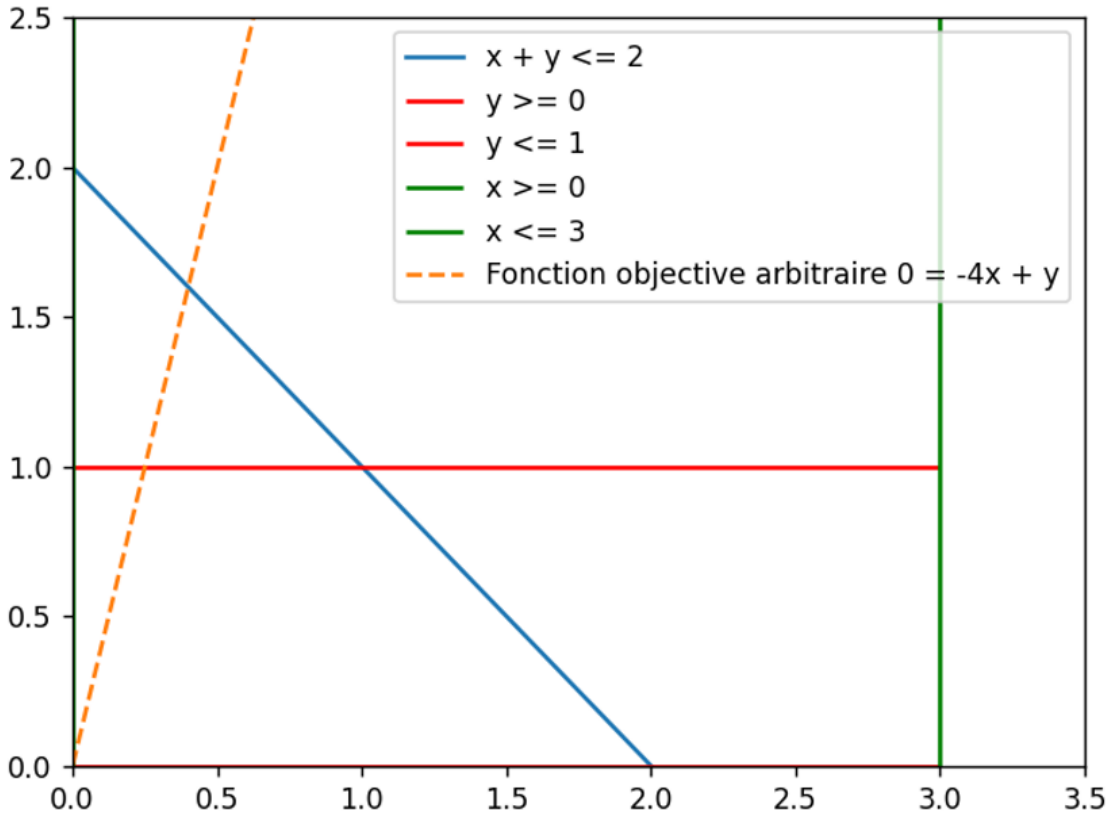
$$\min_{0 \leq x \leq 3,\, 0 \leq y \leq 1} z = -4x + y \quad (6)$$

$$x + y \leq 2 \quad (7)$$

Résolution graphique - Zone de faisabilité

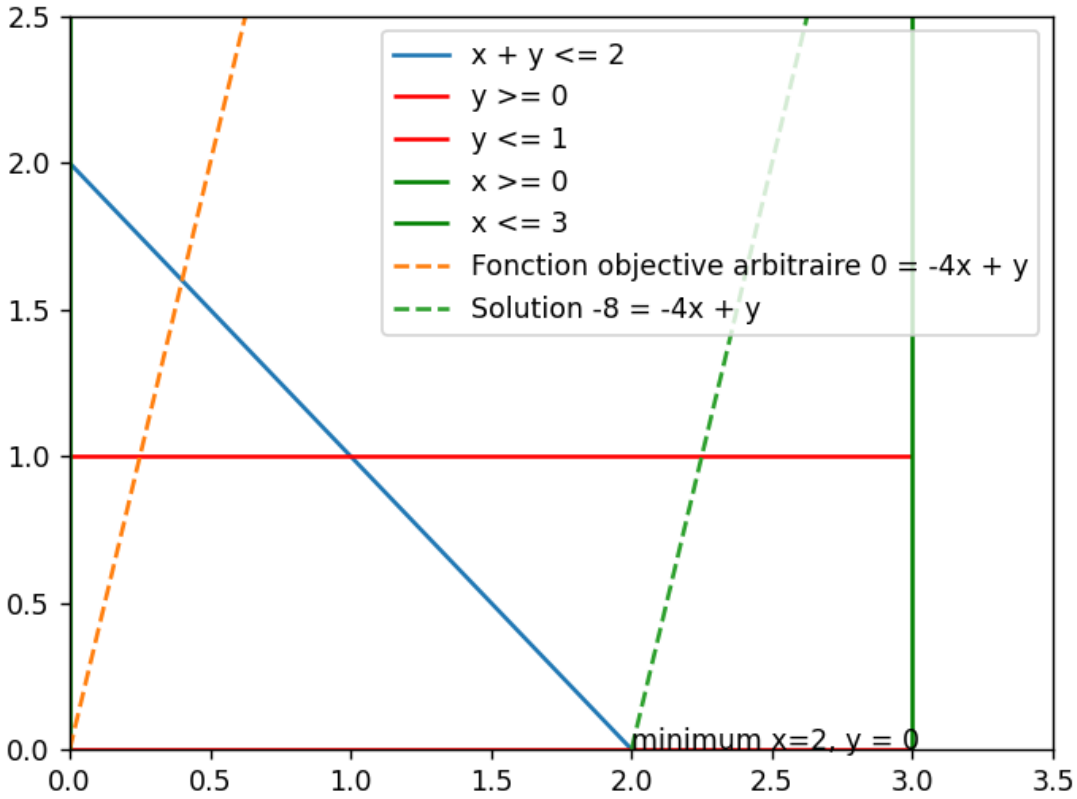


Résolution graphique - Fonction objective



Résolution graphique - Optimum

Pourquoi le minimum n'est pas (0, 0) ? Puisque la valeur optimale objective la plus minimale est -8. Ce n'est pas une question du plus petit x ou y, mais plutôt de Z (-8).



Bibliographie

- https://coin-or.github.io/pulp/main/index.html
- https://matplotlib.org/stable/tutorials/pyplot.html