# -\*- coding: utf-8 -\*-

"""module d'utilitaires pour le TP 1 de GRO comprenant les fonctions

linprog, affiche\_resultat et print\_key

Ecrit par Bruno Pinçon pour le TP 1 de GRO

"""

from pymprog import \*

import numpy as np

import matplotlib.pyplot as plt

from tp import \*

print("merci\_maitre\_antoine (la salope)")

if \_\_name\_\_ == "\_\_main\_\_":

# Exercice 1 (a)

"""

c = np.array([50,40,70,80])

A = np.array([[2,4,8,6],

[10,8,6,10],

[1,1,2,2]])

b = np.array([100,160,20])

stat, F, X = linprog(c, ineq=(A,b), sens = "max")

affiche\_resultat(stat, F, X)

"""

# Exercice 1 (b)

"""

c = np.array([1,1])

A = np.array([[3,2],

[3,4],

[0,1]])

b = np.array([15,21,3])

stat, F, X = linprog(c, ineq=(A,b), sens = "max")

affiche\_resultat(stat, F, X)

"""

# Exercice 1 (c)

"""

c = np.array([1,3])

A = np.array([[-1,-1],

[-1,2],

[-2,1]])

b = np.array([-3,-5,5])

stat, F, X = linprog(c, ineq=(A,b), sens = "max")

affiche\_resultat(stat, F, X)

"""

# Exercice 1 (d)

"""

c = np.array([1,-1])

A = np.array([[1,2],

[2,1],

[-1,-1]])

b = np.array([5,6,-4])

stat, F, X = linprog(c, ineq=(A,b), sens = "max")

affiche\_resultat(stat, F, X)

"""

# Exercice 2 (a)

"""

c = np.array([6,9,8,4])

A = np.array([[4,6,5,4],

[3,6,4,3]])

b = np.array([8,7])

stat, F, X = linprog(c, ineq=(A,b), sens = "max", typevar = bool)

affiche\_resultat(stat, F, X)

"""

# Exercice 2 (b)

"""

c = np.array([6,-9,8,4])

A = np.array([[4,-6,5,-4],

[3,-6,-4,3]])

b = np.array([2,1])

stat, F, X = linprog(c, ineq=(A,b), sens = "max", typevar = bool)

affiche\_resultat(stat, F, X)

"""

# Exercice 3

| # Solution de David CHRUSCIEL # Il y a plus simple mais voilà ce que je propose de la question 1 à 3 de l'exo 3  # Oui, il y a des fonctions qui permettent d'initialiser des tableaux mais flem de chercher # donc j'en ai créer un nouveau def init\_array(l, val=0):  res = []  for i in range(l):  res.append(val)  return res   if \_\_name\_\_ == "\_\_main\_\_":  S = 4 # Capacité  m = 4 # m clés  n = 3 # n fichiers  t = [1, 2, 3] # t de taille n   c\_array = []   # c pour les yj  for j in range(m):  c\_array.append(1)   # c pour les xij  for ij in range(n \* m):  c\_array.append(0)   c = np.array(c\_array)  # Cas avec S = 4, m = 4, n = 3, t = [1, 2, 3],  # c = [y1 , y2 , y3 , y4 , x11, x21, x31, x12, x22, x32, x13, x23, x33, x14, x24, x34]  # c = [1 , 1 , 1 , 1 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ]   A\_array = []   for j in range(m):  tmp = init\_array(m + n \* m)  tmp[j] = -S  for i in range(n):  tmp[m + (j \* n + i)] = t[i]  A\_array.append(tmp)   # Cas avec S = 4, m = 4, n = 3, t = [1, 2, 3],  # A\_array = [  # [y1, y2, y3, y4, x11, x21, x31, x12, x22, x32, x13, x23, x33, x14, x24, x34]  # [y1, y2, y3, y4, x11, x21, x31, x12, x22, x32, x13, x23, x33, x14, x24, x34]  # [y1, y2, y3, y4, x11, x21, x31, x12, x22, x32, x13, x23, x33, x14, x24, x34]  # [y1, y2, y3, y4, x11, x21, x31, x12, x22, x32, x13, x23, x33, x14, x24, x34]  # ]  # A\_array = [  # [-S, 0 , 0 , 0 , t1 , t2 , t3 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ]  # [0 , -S, 0 , 0 , 0 , 0 , 0 , t1 , t2 , t3 , 0 , 0 , 0 , 0 , 0 , 0 ]  # [0 , 0 , -S, 0 , 0 , 0 , 0 , 0 , 0 , 0 , t1 , t2 , t3 , 0 , 0 , 0 ]  # [0 , 0 , 0 , -S, 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , t1 , t2 , t3 ]  # ]   B\_array = []   for i in range(n):  tmp = init\_array(m + n \* m)  for j in range(m):  tmp[m + (j \* n + i)] = 1  B\_array.append(tmp)   # Cas avec S = 4, m = 4, n = 3, t = [1, 2, 3],  # B\_array = [  # [y1, y2, y3, y4, x11, x21, x31, x12, x22, x32, x13, x23, x33, x14, x24, x34]  # [y1, y2, y3, y4, x11, x21, x31, x12, x22, x32, x13, x23, x33, x14, x24, x34]  # [y1, y2, y3, y4, x11, x21, x31, x12, x22, x32, x13, x23, x33, x14, x24, x34]  # ]  # B\_array = [  # [0 , 0 , 0 , 0 , 1 , 0 , 0 , 1 , 0 , 0 , 1 , 0 , 0 , 1 , 0 , 0 ]  # [0 , 0 , 0 , 0 , 0 , 1 , 0 , 0 , 1 , 0 , 0 , 1 , 0 , 0 , 1 , 0 ]  # [0 , 0 , 0 , 0 , 0 , 0 , 1 , 0 , 0 , 1 , 0 , 0 , 1 , 0 , 0 , 1 ]  # ]   # Ax <= b  A = np.array(A\_array)  b = np.array(init\_array(m))    # Bx = d  B = np.array(B\_array)  d = np.array(init\_array(n, 1))   stat, F, X = linprog(c, ineq=(A, b), eq=(B,d), sens="min", typevar=bool)  affiche\_resultat(stat, F, X) |
| --- |