linprog Guide

A standard form of linear programming problem can be formulated as

$$\begin{aligned} &\min_x c^T x \\ &\text{such that} \\ &\quad A_{ub}x \le b_{up} \\ &\quad A_{eq}x = b_{eq} \\ &\quad \ell \le x \le u \end{aligned} \tag{1}$$

where x is a vector of decision variables; c, b_{up}, b_{eq},ℓ and u are vectors; and A_{up} and A_{eq} are matrices.

x: 最终解

fun: 目标函数最优解

status:

```
0: optimization successfully
1: Iteration limit reached
2: problem appears to be infeasible (无解的/不可行的)
3: problem appears to be unbounded (无界的)
4: numerical difficulties encountered
```

In [1]:

```
import numpy as np
from scipy import optimize
```

Q1: (10 points) Read the linprog guide example and complete following questions.

$$\max_{x_1,x_2,x_3,x_4} \quad 29x_1 + 45x_2$$

$$\begin{aligned} &\text{such that} \\ &x_1 - x_2 - 3x_3 && \le 5 \\ &2x_1 - 3_x2 - 7x_3 + 3x_4 && \ge 10 \\ &2x_1 + 8x_2 + x_3 && = 60; \\ &4x_1 + 4x_2 + x_4 && = 60 \\ &0 \le x_1 && \\ &0 \le x_2 && \le 5 \\ &x_3 && \le 0.5 \\ &-3 && \le x_4 \end{aligned}$$

(a) (7 points) Transform the problem to the standard form (1) of linear programming. Specify that c, A_up, A_eq, b_up, b_eq, l and u.

a):

$$c = \begin{bmatrix} -29 \\ -45 \\ 0 \\ 0 \end{bmatrix}$$

$$b_{eq} = \begin{bmatrix} 60 \\ 60 \end{bmatrix}$$

$$l = \begin{bmatrix} 0 \\ 0 \\ -\infty \\ -3 \end{bmatrix}$$

$$A_{up} = \begin{bmatrix} 1 & -1 & -3 & 0 \\ -2 & 3 & 7 & -3 \end{bmatrix}$$

$$b_{up} = \begin{bmatrix} 5 \\ -10 \end{bmatrix}$$

$$u = \begin{bmatrix} +\infty \\ 5 \\ 0.5 \\ +\infty \end{bmatrix}$$

$$A_{eq} = \begin{bmatrix} 2 & 8 & 1 & 0 \\ 4 & 4 & 0 & 1 \end{bmatrix}$$

(b) (3 points) Exclude the python code in the guide.

In [2]:

```python
c = [-29, -45, 0, 0]
A_up =[[1, -1, -3, 0], [-2, 3, 7, -3]]
b_up = [5, -10]
A_eq = [[2, 8, 1, 0], [4, 4, 0, 1]]
b_eq = [60, 60]
bounds = ((0, None), (0, 5), (None, 0.5), (-3, None))
res = optimize.linprog(c, A_ub=A_up, b_ub=b_up, A_eq = A_eq,
                       b_eq = b_eq, bounds=bounds)
print(res)
```

```
     con: array([15.53611342, 16.61287497])
     fun: -370.23225684077204
 message: 'The algorithm terminated successfully and determined that t
he problem is infeasible.'
     nit: 6
   slack: array([ 0.79315139, -1.76308995])
  status: 2
 success: False
       x: array([ 6.60059384,  3.97366745, -0.52664074,  1.09007985])
```

The problem is infeasible.

Q2: (20 points) Consider the following problem

$$\max\ x_1 + 4x_2 + x_3$$
$$\text{subject to}: 2x_1 - 2x_2 + x_3 = 4$$
$$x_1 - x_3 = 1$$
$$x_2 \geq 0, x_3 \geq 0$$

(a) (14 points) Convert the following problem to standard form.

a):

$$\min -x_1 - 4x_2 - x_3$$
$$\textit{subject to}:\ 2x_1 - 2x_2 + x_3 = 4$$
$$x_1 - x_3 = 1$$
$$x_2 \geq 0, x_3 \geq 0$$

$$c = \begin{bmatrix} -1 \\ -4 \\ -1 \end{bmatrix} \qquad l = \begin{bmatrix} -\infty \\ 0 \\ 0 \end{bmatrix}$$

$$A_{eq} = \begin{bmatrix} 2 & -2 & 1 \\ 1 & 0 & -1 \end{bmatrix}$$

$$b_{eq} = \begin{bmatrix} 4 \\ 1 \end{bmatrix} \qquad u = \begin{bmatrix} +\infty \\ +\infty \\ +\infty \end{bmatrix}$$

(b) (6 points) Solve it with linprog

```
In [3]:
```

```
c = [-1, -4, -1]
A_eq = [[2, -2, 1], [1, 0, -1]]
b_eq = [4, 1]
bounds = ((None, None), (0, None), (0, None))
res = optimize.linprog(c, A_eq = A_eq, b_eq = b_eq, bounds=bounds)

print(res)
```

```
     con: array([7.23165894, 2.89105225])
     fun: -608757409861.3794
 message: 'The algorithm terminated successfully and determined that t
he problem is unbounded.'
     nit: 4
   slack: array([], dtype=float64)
  status: 3
 success: False
       x: array([7.60946762e+10, 1.14142014e+11, 7.60946762e+10])
```

The problem is unbounded, answer for this question is infinity.

Q3: (30 points) Follow the steps of in question 1. Solve the following to linear programming pro- gram

$$\min\ x + 2y + 3z$$

$$\text{subject to:}$$

$$2 \le x + y \le 3$$

$$4 \le x + z \le 5$$

$$x \ge 0, y \ge 0, z \ge 0$$

(Hints: you can use addition variables to make the equality constraints.)

(a) (21 points) Convert the following problem to standard form.

a):

$$\min x + 2y + 3z$$

$$\begin{aligned} \textit{subject to}:\quad & x + y \le 3 \\ & -x - y \le -2 \\ & x + z \le 5 \\ & -x - z \le -4 \end{aligned}$$

$$x \ge 0, y \ge 0, z \ge 0$$

$$c = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \qquad b_{up} = \begin{bmatrix} 3 \\ -2 \\ 5 \\ -4 \end{bmatrix}$$

$$A_{up} = \begin{bmatrix} 1 & 1 & 0 \\ -1 & -1 & 0 \\ 1 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix} \qquad l = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$u = \begin{bmatrix} +\infty \\ +\infty \\ +\infty \end{bmatrix}$$

(b) (9 points) Solve it with linprog

In [4]:

```
c = [1, 2, 3]
A_up = [[1, 1, 0], [-1, -1, 0], [1, 0, 1], [-1, 0, -1]]
b_eq = [3, -2, 5, -4]
bounds = ((0, None), (0, None), (0, None))
res = optimize.linprog(c, A_ub = A_up, b_ub = b_eq, bounds=bounds)
print(res)
```

```
     con: array([], dtype=float64)
     fun: 5.999999993369686
 message: 'Optimization terminated successfully.'
     nit: 4
   slack: array([ 5.72031578e-09,  9.99999994e-01,  1.00000001e+00, -
8.55915250e-09])
  status: 0
 success: True
       x: array([2.99999999e+00, 1.90162796e-09, 9.99999999e-01])
```

When x = 3, y = 0 and z = 1, the object function takes the minimum value, which is 6.

Q4: (30 points) A large textile firm has two manufacturing plants, two sources of raw material, and three market centers. The transportation costs between the sources and the plants and between the plants and the markets are as follows

	plant A	plant B
source 1	¥100/ton	¥150/ton
source 2	¥200/ton	¥150/ton

	market 1	market 2	market 3
plant A	¥400/ton	¥200/ton	¥100 /ton
plant B	¥300/ton	¥400 /ton	¥200/ton

Ten tons are available from source 1 and 15 tons from source 2. The three market centers require 8 tons, 14 tons and 3 tons. The plants have unlimited processing capacity.

(a) (20 points) Formulate the problem of finding the shipping patterns from sources to plants to markets that minimizes the total transportation cost.

a):

Let **X_ij, i=s1,s2, j=A,B** represents the quantity transported between source 1,2 and plant A,B；

Let **Y_ij, i=m1,m2,m3, j=A,B** represents the quantity transported between market 1,2,3 and plant A,B;

$$\min 100x_{1A} + 150x_{1B} + 200x_{2A} + 150x_{2B} + 400y_{1A} + 200y_{2A} + 100y_{3A} + 300y_{1B} + 400y_{2B} + 200y_{3B}$$

$$\begin{aligned} \textit{subject to:}\quad x_{1A} + x_{2A} - y_{1A} - y_{2A} - y_{3A} &= 0 \\ x_{1B} + x_{2B} - y_{1B} - y_{2B} - y_{3B} &= 0 \\ x_{1A} + x_{1B} &\le 10 \\ x_{2A} + x_{2B} &\le 15 \\ y_{1A} + y_{1B} &\ge 8 \\ y_{2A} + y_{2B} &\ge 14 \\ y_{3A} + y_{3B} &\ge 3 \end{aligned}$$

(b) (10 points) Solve the problem with linprog.

In [5]:

```python
c = [100, 150, 200, 150, 400, 200, 100, 300, 400, 200]
A_eq = [[1, 0, 1, 0, -1, -1, -1, 0, 0, 0],
        [0, 1, 0, 1, 0, 0, 0, -1, -1, -1,]]
b_eq = [0, 0]
A_up = [[1, 1, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 1, 1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, -1, 0, 0, -1, 0, 0],
        [0, 0, 0, 0, 0, -1, 0, 0, -1, 0],
        [0, 0, 0, 0, 0, 0, -1, 0, 0, -1]]
b_up = [10, 15, -8, -14, -3]
bounds = ((0, None), (0, None), (0, None), (0, None), (0, None),
          (0, None), (0, None), (0, None), (0, None), (0, None))
res = optimize.linprog(c, A_eq = A_eq, b_eq = b_eq,
                       A_ub=A_up, b_ub=b_up,bounds=bounds)
print(res)
```

```
     con: array([1.38975054e-10, 1.38871063e-10])
     fun: 9099.99999889964
 message: 'Optimization terminated successfully.'
     nit: 6
   slack: array([ 1.11819709e-09,  2.08282103e-09, -9.50149293e-10, -
1.71801950e-09,
       -2.55000021e-10])
  status: 0
 success: True
       x: array([1.00000000e+01, 6.09313000e-11, 7.00000000e+00, 8.000
00000e+00,
       1.10353575e-11, 1.40000000e+01, 3.00000000e+00, 8.00000000e+00,
       7.24916267e-11, 1.81980299e-10])
```

Therefore, to minimize the total transportation cost we need:

1. 10 tons are transported from source 1 to plant A,
2. 7 tons are transported from source 2 to plant A,
3. 8 tons are transported from source 2 to plant B,
4. 14 tons are transported from plant A to market 2,
5. 3 tons are transported from plant A to market 3,
6. 8 tons are transported from plant B to market 1.

And the final minimized cost is $9100

b): if we let quantity from A,B larger or equals to the quantity that demands form A,B:

that is, no A_eq and b_eq, change them into A_up and b_up

$$\min 100x_{1A} + 150x_{1B} + 200x_{2A} + 150x_{2B} + 400y_{1A} + 200y_{2A} + 100y_{3A} + 300y_{1B} + 400y_{2B} + 200y_{3B}$$

$$\begin{aligned} subject\ to: \quad x_{1A} + x_{2A} - y_{1A} - y_{2A} - y_{3A} &\geq 0 \\ x_{1B} + x_{2B} - y_{1B} - y_{2B} - y_{3B} &\geq 0 \\ x_{1A} + x_{1B} &\leq 10 \\ x_{2A} + x_{2B} &\leq 15 \\ y_{1A} + y_{1B} &\geq 8 \\ y_{2A} + y_{2B} &\geq 14 \\ y_{3A} + y_{3B} &\geq 3 \\ x_{ij} \geq 0, i = S1, S2; j &= A, B \\ y_{ij} \geq 0, i = M1, M2, M3; j &= A, B \end{aligned}$$

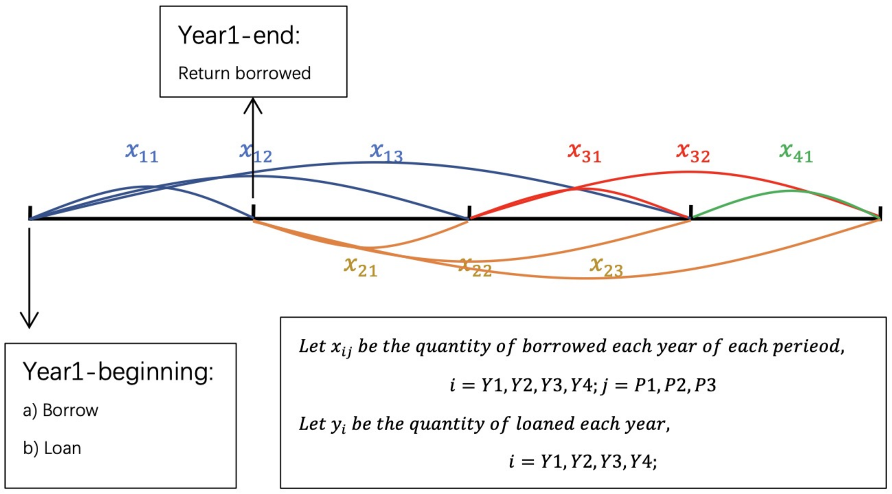
In [6]:

```
c = [100, 150, 200, 150, 400, 200, 100, 300, 400, 200]
# A_eq = [[1, 0, 1, 0, -1, -1, -1, 0, 0, 0],
#         [0, 1, 0, 1, 0, 0, 0, -1, -1, -1,]]
# b_eq = [0, 0]
A_up = [[-1, -1, 0, 0, 1, 1, 1, 0, 0, 0],
        [0, 0, -1, -1, 0, 0, 0, 1, 1, 1],
        [1, 1, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 1, 1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, -1, 0, 0, -1, 0, 0],
        [0, 0, 0, 0, 0, -1, 0, 0, -1, 0],
        [0, 0, 0, 0, 0, 0, -1, 0, 0, -1]]
b_up = [0, 0, 10, 15, -8, -14, -3]
bounds = ((0, None), (0, None), (0, None), (0, None), (0, None),
          (0, None), (0, None), (0, None), (0, None), (0, None))
res = optimize.linprog(c, A_ub=A_up, b_ub=b_up,bounds=bounds)
print(res)
```

```
     con: array([], dtype=float64)
     fun: 9849.999999740576
 message: 'Optimization terminated successfully.'
     nit: 6
   slack: array([-5.06084064e-11, -4.90101293e-11,  2.19662510e-10,
4.76688911e-10,
       -1.90674143e-10, -3.53990615e-10, -5.20690158e-11])
  status: 0
 success: True
       x: array([1.00000000e+01, 7.97896461e-11, 6.87441999e-11, 1.500
00000e+01,
       6.59248186e-12, 1.00000000e+01, 4.28181805e-13, 8.00000000e+00,
       4.00000000e+00, 3.00000000e+00])
```

However, the results ($9850) is greater than the first answer. So we'd better let them just be equal.

Q5: (20 points) A businessman is considering an investment project. The project has a lifetime of four years, with cash flows of -¥100,000, +¥50,000, +¥70,000, +¥30,000 in each of the four years, respectively. At any time he may borrow funds at the rates of 12%, 22%,and 34%(total) for 1, 2, 3 periods, respectively. He may also loan fund at 10% per period. He calculate the present value of a project as the maximum amount of money he would pay now, to another party, for the project, assuming that he has no cash on hand and must borrow and lend to pay the other party and operate the project while maintaining a non-negative cash balance after all debts are paid. Formulate the project valuation problem in a linear programming framework.



Let x_{ij} be the quantity of borrowed each year of each perieod,

$i = Y1, Y2, Y3, Y4; j = P1, P2, P3$

Let y_i be the quantity of loaned each year,

$i = Y1, Y2, Y3, Y4;$

At the beginning of each year, he may choose to borrow (借入) or loan (借出) money

At the end of each year, he needs to return the borrowed money if it's due

Balance(t) = Cash Flow(t) + U(t), Balance each year needs to be non-negative

U(t) represents the operation of borrowed and loaned money, we can write the B(t) for 4 years:

$$B(1) = (x_{11} + x_{12} + x_{13}) + 0.1y_1 - 1.12x_{11} - 100000$$
$$B(2) = (x_{12} + x_{13} + x_{21} + x_{22} + x_{23}) + 0.1y_2 - (1.22x_{12} + 1.12x_{21}) + 50000$$
$$B(3) = (x_{13} + x_{22} + x_{23} + x_{31} + x_{32}) + 0.1y_3 - (1.34x_{13} + 1.22x_{22} + 1.12x_{31}) + 70000$$
$$B(4) = (x_{23} + x_{32} + x_{41}) + 0.1y_4 - (1.34x_{23} + 1.22x_{32} + 1.12x_{41}) + 30000$$

He wants to Maximum B(4), which is to Minimize -B(4)

$$
\begin{aligned}
\min.\ & -(x_{23}+x_{32}+x_{41})-0.1y_4+(1.34x_{23}+1.22x_{32}+1.12x_{41}) \\
s.t.\ & B(i)\geq 0, x_{ij}\geq 0, y_i\geq 0; i=1,2,3,4; j=1,2,3 \\
& y_1\leq x_{11}+x_{12}+x_{13} \\
& y_2\leq B(1)+x_{21}+x_{22}+x_{23} \\
& y_3\leq B(2)+x_{31}+x_{32} \\
& y_4\leq B(3)+x_{41}
\end{aligned}
$$

In [7]:

```
c = [0, 0, 0, 0, 0, 0.34, 0, 0.22, 0.12, 0, 0, 0, -0.1]

A_up = [[0.12, -1, -1, 0, 0, 0, 0, 0, 0, -0.1, 0, 0, 0],
        [0, 0.22, -1, 0.12, -1, -1, 0, 0, 0, 0, -0.1, 0, 0],
        [0, 0, 0.34, 0, 0.22, -1, 0.12, -1, 0, 0, 0, -0.1, 0],
        [0, 0, 0, 0, 0, 0.34, 0, 0.22, 0.12, 0, 0, 0, -0.1],
        [-1,-1,-1,0,0,0,0,0,0,1,0,0,0],
        [0.12,-1,-1,-1,-1,-1,0,0,0,-0.1,1,0,0],
        [0,0.22,-1,0.12,-1,-1,-1,-1,0,0,-0.1,1,0],
        [0,0,0.34,0,0.22,-1,0.12,-1,-1,0,0,-0.1,1]]

b_up = [-100000,50000,70000,30000,0,-100000,50000,70000]

bounds = ((0, None), (0, None), (0, None), (0, None),
          (0, None), (0, None), (0, None), (0, None),
          (0, None), (0, None), (0, None), (0, None), (0, None))
res = optimize.linprog(c, A_ub=A_up, b_ub=b_up,bounds=bounds)
print(res)
```

```
     con: array([], dtype=float64)
     fun: -7299.999929439171
 message: 'Optimization terminated successfully.'
     nit: 9
   slack: array([5.34643128e-04, 2.99999961e+04, 7.29999995e+04, 3.729
99999e+04,
       3.06882995e-03, 2.89805222e-03, 2.84251917e-04, 3.90301429e-0
5])
  status: 0
 success: True
       x: array([8.00445420e-01, 9.09091062e+04, 6.84054108e-05, 3.811
33782e-02,
       3.66694949e-04, 2.03156361e-05, 9.22445666e-04, 3.58766939e-05,
       2.07992567e-04, 9.09099036e+04, 3.61369797e-02, 2.99999968e+04,
       7.29999997e+04])
```

Seems by the end of the year 4, he can make ¥22700 (-¥7300+¥30000), if he

borrows: ¥90909 of 2 period in year 1;

loan: ¥90909 in year 1; ¥30000 in year 2; ¥73000 in year 3