CVXPY 是一种用于凸优化问题的 Python 嵌入式建模语言。它允许您以遵循数学的自然方式表达您的问题，而不是求解器要求的限制性标准形式。

CVXPY 的求解状态

求解状态	含义
OPTIMAL	最优解
INFEASIBLE	不可行
UNBOUNDED	无界
OPTIMAL_INACCURATE	不精确
INFEASIBLE_INACCURATE	不精确
UNBOUNDED_INACCURATE	不精确

CVXPY 的变量类型

- 变量可以是标量、向量以及矩阵
- cvxpy中可以做常数使用的用：
  - - NumPy ndarrays
  - - NumPy matrices
  - - SciPy sparse matrices

CVXPY 的约束可以使用 ==, <=， >= ， 不能使用< ， >。也不能使用 0 <= x <= 1 or x == y == 2。

parameters可以理解为参数求解问题里的一个常数，可以是标量、向量、矩阵。在没有求解问题前（xxx.solve()），其允许你改变其值。

例 1

$$\min z = 160x_{11} + 130x_{12} + 220x_{13} + 170x_{14} + 140x_{21} + 130x_{22} + 190x_{23} + 150x_{24} + 190x_{31} + 200x_{32} + 230x_{33}$$

$$s.t. \begin{cases} x_{11} + x_{12} + x_{13} + x_{14} = 50 \\ x_{21} + x_{22} + x_{33} + x_{24} = 60 \\ x_{31} + x_{32} + x_{33} = 50 \\ 30 \le x_{11} + x_{21} + x_{31} \le 80 \\ 70 \le x_{12} + x_{22} + x_{32} \le 140 \\ 10 \le x_{13} + x_{23} + x_{33} \le 30 \\ 10 \le x_{14} + x_{24} \le 50 \\ x_{ij} \le 0 \quad ((i,\ j) = (1,\ 1),\ \ldots,\ (3,\ 3)) \end{cases}$$

```python
c = np.array([160, 130, 220, 170, 140, 130, 190, 150, 190, 200, 230])

left = np.array(
    [
        [1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0],
        [0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0],
        [0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1],
        [0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0],
    ]
)
```

```python
right_min = np.array([30, 70, 10, 10])
right_max = np.array([80, 140, 30, 50])
x = cp.Variable(11)
obj = cp.Minimize(c @ x)
con = [
    x >= 0,
    left @ x <= right_max,
    left @ x >= right_min,
    cp.sum(x[0:4]) == 50,
    cp.sum(x[4:8]) == 60,
    cp.sum(x[8:11]) == 50,
]
prob = cp.Problem(obj, con)
prob.solve(solver="COPT")
print(f"最优结果: {prob.value}")
print(f"参数取值: {x.value}")
```

例 2

$$
c(x)=\begin{cases}10x \quad (0\le x\le 500)\\ 1000+8x \quad (500\le x\le 1000)\\ 3000+6x \quad (1000\le x\le 1500)\end{cases}
$$

$$
\max z = 4.8x_{11}+5.6x_{12}+4.8x_{21}+5.6x_{22}-c(x)
$$

$$
s.t.\begin{cases}x_{11}+x_{12}\le 500+x\\ x_{21}+x_{22}\le 1000\\ x\le 1500\\ -x_{11}+x_{21}\le 0\\ -2x_{12}+3x_{22}\le 0\\ x_{11},\ x_{12},\ x_{21},\ x_{22},\quad x\ge 0\\ z_1\le y_1,\quad z_2\le y_1+y_2,\quad z_3\le y_2+y_3,\quad z_4\le y_3\\ z_1+z_2+z_3+z_4=1,\quad z_1,\ z_2,\ z_3,\ z_4\ge 0\\ y_1+y_2+y_3=1,\quad y_1,\ y_2,\ y_3=0,\ 1\\ x=500z_2+1000z_3+1500z_4\\ c(x)=5000z_2+9000z_3+12000z_4\end{cases}
$$

```python
coef_x = np.array([4.8, 5.6, 4.8, 5.6])  # 输入目标函数 x 对应系数
coef_cx = np.array([0, 5000, 9000, 12000])  # 输入用 在表示 cx 的系数
coef_buy_x = np.array([0, 500, 1000, 1500])  # 输入用 z 表示 x 的系数
left = np.array([[0, 0, 1, 1], [-1, 0, 1, 0], [0, -2, 0, 3]])  # 输入约束条件系数
right = np.array([1000, 0, 0])  # 输入约束条件上下值
x = cp.Variable(4)  # 创建决策变量 x
y = cp.Variable(3, integer=True)  # 创建 0-1 变量 y
z = cp.Variable(4)  # 创建变量    z
obj = cp.Maximize(coef_x @ x - coef_cx @ z)  # 构造目标函数
con = np.array(
    [
        cp.sum(x[:2]) <= 500 + cp.sum(coef_buy_x @ z),
        left @ x <= right,
        sum(coef_buy_x @ z) <= 1500,
        x >= 0,
        z[0] <= y[0],
```

```python
        z[1] <= y[0] + y[1],
        z[2] <= y[1] + y[2],
        z[3] <= y[2],
        cp.sum(z[:]) == 1,
        z >= 0,
        cp.sum(y[:]) == 1,
        y >= 0,
        y <= 1,
    ],
)
prob = cp.Problem(obj, con)
prob.solve(solver="COPT")
print(f"最优结果: {prob.value}")
print(f"参数取值: {x.value}")
```

```
最优结果: 5000.0
参数取值: [  -0. 1500.    0. 1000.]
```

例三

多目标规划:

(1) $minz = \sum_{i=1}^{9} x_i$

(2) $maxw = 5x_1 + 4x_2 + 4x_3 + 3x_4 + 4x_5 + 3x_6 + 2x_7 + 2x_8 + 3x_9$

(3) $miny = 0.7z - 0.3w = -0.8x_1 - 0.5x_2 - 0.5x_3 - 0.2x_4 - 0.5x_5 - 0.2x_6 + 0.1x_7 + 0.1x_8 - 0.2x_9$

(4) $x_1 + x_2 + x_3 + x_4 + x_5 \geqslant 2$

(5) $x_3 + x_5 + x_6 + x_8 + x_9 \geqslant 3$

(6) $x_4 + x_6 + x_7 + x_9 \geqslant 2$

(7) $2x_3 - x_1 - x_2 \leqslant 0$

(8) $x_4 - x_7 \leqslant 0$

(9) $2x_5 - x_1 - x_2 \leqslant 0$

(10) $x_6 - x_7 \leqslant 0$

(11) $x_8 - x_5 \leqslant 0$

(12) $2x_9 - x_1 - x_2 \leqslant 0$

(13) $x_i = 0, 1, i = 1, 2, 3..., 8, 9$