例 2.2 线性规划单纯形算法的实现

```
修改参数：d：添加变量将问题变为标准型
"""

import numpy as np


# d 数组第 0 行是 c，最后一列是 b，其余是 A
def pivot():
    l = list(d[0][:-1])  #
    jnum = l.index(max(l))  # 转入编号
    m = []
    for i in range(bn):
        if d[i][jnum] == 0:
            m.append(0.)
        else:
            m.append(d[i][-1] / d[i][jnum])
    inum = m.index(min([x for x in m[1:] if x != 0]))  # 转出下标
    s[inum - 1] = jnum  # 更新基变量
    # 更新 d
    r = d[inum][jnum]
    d[inum] /= r
    for i in [x for x in range(bn) if x != inum]:
        r = d[i][jnum]
        d[i] -= r * d[inum]


def solve():
    flag = True
    while flag:
        if max(list(d[0][:-1])) <= 0:  # 直至所有系数小于等于 0
            flag = False
        else:
            pivot()


def printSol():
    for i in range(cn - 1):
        if i in s:
            print("x" + str(i) + "=%.2f" % d[s.index(i) + 1][-1])
        else:
            print("x" + str(i) + "=0.00")
    print("objective is %.2f" % (-d[0][-1]))
```

```
if __name__ == '__main__':
    d = np.array([[25, 30, 0, 0, 0, 0, 0, 0],
                  [2, 3, 1, 0, 0, 0, 0, 1200],
                  [6, 5, 0, 1, 0, 0, 0, 2112],
                  [3, 4, 0, 0, 1, 0, 0, 1800],
                  [1, 0, 0, 0, 0, 1, 0, 0],
                  [0, 1, 0, 0, 0, 0, 1, 0]
                  ], dtype=np.float)
    (bn, cn) = d.shape
    s = list(range(cn - bn, cn - 1))  # 基变量列表
    solve()
    printSol()
```