线性规划求解污水处理问题

一、摘要

本问题研究在河流边污水处理的模型中如何在满足污水质量标准的情况下实现最低成本。通过改变每个污水处理厂的处理情况实现最低成本，采用线性规划模型，利用 python 编程求解，得出最低成本的结果。

关键词：线性规划

二、问题重述

若干工厂的污水经排污口流入某江，各口有污水处理站。上游江水流量、污水质量浓度、国家标准规定的水的污染浓度及各个工厂的污水流量和污水质量浓度均为已知。污水处理费用与污水处理前后的质量浓度差和污水流量成正比，使每单位流量的污水下降一个质量浓度单位需要的处理费用(称处理系数)为已知。处理后的污水与江水混合流到下一个排污口之前，自然状态下的江水也会使污水质量浓度降低一个比例系数(称自净系数)，该系数可以估计。试确定各污水处理站出口的污水质量浓度，使在符合国家标准规定的条件下总的处理贵用最小。

三、问题分析

问题的本质在于求解满足一定限制条件下多个同种类变量对于线性和的影响，使得总量最小，因此采用线性规划模型，对污水质量密度进行建模。

四、符号说明

符号	说明
n	工厂数
k	第 k 个工厂
Q_k	第 k 段江面的流量
Q	Q_k的近似值

C_k	第 k 段江面的污水质量浓度
$\triangle Q_k$	工厂 k 和处理站 k 的污水流量
u_k	工厂 k 流出的污水质量浓度
v_k	处理站 k 流出的污水质量浓度
r_k	处理站 k 的处理系数
b_k	第 k 段江面的自净系数
C_0	国家标准规定的水的污染质量浓度

五、模型建立

$\triangle Q_k$ 通常远小于 Q_k，因此将 Q_k 视为常数 Q。

Q，C_0，C_1，$\triangle Q_k$，u_k，r_k，b_k 已知，v_k 为决策变量。

在 Q_k 简化为常数 Q 的情况下，处理站 k 的污水与江水混合后的污水质量浓度为 $D_k = C_k + \frac{\Delta Q_k}{Q} v_k$，第 k 段江面自净后的污水质量浓度为 $C_{k+1} = b_k D_k$，处理站 k 的处理费用为 $r_k \Delta Q_k (u_k - v_k)$，于是在江面上所有地段的水污染达到国家标准的条件下,使总费用最小的模型为

$$\min\ T = \sum_{k+1}^{n} r_k \Delta Q_k (u_k - v_k)$$

$$\text{s.t.}\ D_k = C_k + \frac{\Delta Q_k}{Q} v_k$$

$$C_{k+1} = b_k D_k$$

$$D_k \le C_0$$

$$v_k \le u_k$$

六、模型求解

具体问题中，$Q = 1000 \times 10^{12}$ L/min，$C_1 = 0.8$ mg/L，$C_0 = 1$ mg/L，$\triangle Q_1 = \triangle Q_2 = \triangle Q_3 = 5 \times 10^{12}$ L/min，$u_1 = 100$ mg/L，$u_2 = 60$ mg/L，$u_3 = 50$ mg/L，$r_1 = r_2$

$= r_3 = 1$ 万元/（(10^{12} L/min）x（mg/L)），$b_1 = 0.9$，$b_2 = 0.6$.

（1）代入得 $v_1 = 40$，$v_2 = 20$，$v_3 = 50$（mg/L），工厂 3 无需处理污水，最小总费用 500 万元。

（2）将模型中的$D_k \le C_0$改为$C_k \le C_0$,求解得 $v_1 = 62.2$，$v_2 = 60$，$v_3 = 50$（mg/L），工厂 2、3 无需处理污水，最小总费用 189 万元。

七、模型评价

1. 优点：

非常适合本题目，且能够使用 python 进行求解，方便简单。

2. 缺点：

对数据精准度的要求较高。

八、附录代码

```python
import pulp as lp

# 定义问题类型为最小化线性规划
prob = lp.LpProblem("Linear Programming Problem",
lp.LpMinimize)

# 定义变量
Q = 1000 * 10**12 # L/min
C1 = 0.8 # mg/L
C0 = 1 # mg/L
delta_Q = [5 * 10**12, 5 * 10**12, 5 * 10**12] # L/min
u = [100, 60, 50] # mg/L
r = [1, 1, 1] # 万元/((10^12 L/min)*(mg/L))
b = [0.9, 0.6]

D = [lp.LpVariable(f"D{k}", lowBound=0) for k in range(1, 4)]
v = [lp.LpVariable(f"v{k}", lowBound=0) for k in range(1, 4)]

# 定义目标函数
objective = lp.lpSum([r[k-1] * delta_Q[k-1] * (u[k-1] - v[k-
1]) for k in range(2, 4)])
prob += objective
```

```python
# 添加约束条件
prob += D[0] <= C0
for k in range(1, 3):
    prob += D[k] <= b[k-1] * D[k-1]
    prob += D[k] == C1 + delta_Q[k] / Q * v[k]
    prob += v[k] <= u[k]

# 求解线性规划问题
prob.solve()

# 打印结果
print("Optimal Solution:")
for k in range(1, 4):
    print(f"D{k}: {D[k-1].value()}, v{k}: {v[k-1].value()}")
print(f"Minimum Cost: {lp.value(prob.objective)}")
```