2023-2024学年第二学期《数学模型》

课 程 报 告

姓名 陆奕非 学号 20222429 贡献度 30% 成绩

姓名 李瀚学 学号 20222426 贡献度 40% 成绩

姓名 刘伟涛 学号 20222421 贡献度 30% 成绩

**圈养湖羊的空间利用率**

**1. 问题重述**

**1.1 问题的背景**

湖羊作为国家级绵羊保护品种，具有多种优良特性，其养殖涉及繁殖和育肥两大环节，特别是繁殖多采用自然交配方式，且母羊的孕期、哺乳期等周期对养殖管理有重要影响。需要规模化的圈养养殖场需要根据牲畜的性别和生长阶段分群饲养，以优化空间使用，保障牲畜的健康和安全，同时减少资源浪费。某湖羊养殖场采用标准羊栏设置，并根据羊只的不同阶段和性别规定了羊栏的容纳量，要求不同阶段的羊只原则上不能同栏。养殖场的经营管理者需要制定生产计划，以优化空间利用率。这包括决定何时对基础母羊进行配种、控制繁育期，从而调节对羊栏的需求，确保羊栏的足够使用，同时尽量减少闲置。

**1.2 问题的重述**

问题一：不考虑不确定因素和种羊的淘汰更新，假定自然交配期 20 天，母羊都能受孕，孕期 149 天，每胎产羔 2 只，哺乳期 40 天，羔羊育肥期 210 天，母羊空怀休整期 20 天。该湖羊养殖场现有 112 个标准羊栏，在实现连续生产的条件下，试确定养殖场种公羊与基础母羊的合理数量，并估算年化出栏羊只数量的范围。若该养殖场希望每年出栏不少于1500 只羊，试估算现有标准羊栏数量的缺口。

问题二：在问题 1 的基础上，对 112 个标准羊栏给出具体的生产计划（包括种公羊与基础母羊的配种时机和数量、羊栏的使用方案、年化出栏羊只数量等），使得年化出栏羊只数量最大。

问题三：问题 1 和问题 2 中用到的数据都没有考虑不确定性，然而配种成功率、分娩羔羊的数目和死亡率等都有不确定性，哺乳时间也可以调控，这些都会影响空间需求。现根据经验作以下考虑：

(1) 母羊通过自然交配受孕率为 85%，交配期结束后 30 天可识别出是否成功受孕；

(2) 在自然交配的 20 天中受孕母羊的受孕时间并不确知，而孕期会在 147-150 天内波动，这些因素将影响到预产期范围；

(3) 怀孕母羊分娩时一般每胎产羔 2 只，少部分每胎产羔 1 只或 3 只及以上，目前尚没有实用手段控制或提前得知产羔数。羔羊出生时，有夭折的可能，多羔死亡率高于正常。通常可以按平均每胎产羔 2.2 只、羔羊平均死亡率 3%估算。

(4) 母羊哺乳期过短不利于羔羊后期的生长，通常是羔羊体重达到一定标准后断奶；而哺乳期过长，母羊的身体消耗就越大，早点断奶，有利于早恢复、早发情配种。一种经验做法是将哺乳期控制在 35-45 天内，以 40 天为基准，哺乳期每减少 1 天，羔羊的育肥期增加 2 天；哺乳期每增加 1 天，羔羊的育肥期减少 2 天。除此之外，母羊的空怀休整期可在不少于 18 天的前提下灵活调控。

此外，如有必要，允许分娩日期相差不超过 7 天的哺乳期母羊及所产羔羊同栏，允许断奶日期相差不超过 7 天的育肥期羔羊同栏，允许断奶日期相差不超过 7 天的休整期母羊同栏。为简化问题，不考虑母羊流产、死亡以及羔羊在哺乳期或育肥期夭折和个体发育快慢等情况。请综合考虑可行性和年化出栏羊只数量，制定具体的生产计划，使得整体方案的期望损失最小。

**2. 问题分析**

问题一：由于问题一中已经给出了自然交配期、受孕期、哺乳期等一系列定制，我们可以通过估算大致了解羊场的生产规模，在提供的确定条件下可以估计出基础母羊的数量范围。因为第一问给定了标准羊栏数和每年出栏羊的数量，而当栏位固定时，可以估计每年养羊数量的上限。根据母羊及其羊羔在一个生育周期内占用的栏数乘天数（即“栏天”数），可以估算出每年养羊数量的上限和下限。

问题二：由题目可知母羊数量远大于公羊数量，且公羊的需求量由母羊数量决定，因此先考虑母羊占用栏数，再计算公羊数量和所占栏数。我们固定了每批次母羊和公羊的数量相等以及每批次之间差的天数相等，因此主要确定每批次开始的时间和每批公羊及母羊的数量即可得到生产方案。

问题三：本题涉及的随机因素太多，我们根据经验一和经验三，大致确定经验一和三中考虑的因素并以小推大，推断出其他不确定因素对本模型可能造成的影响。比如经验三中提到了羔羊出生时夭折的可能性。多羔的情况下，由于母羊的照顾能力有限，羔羊的死亡率往往高于正常水平。为了更准确地估计这一影响，我们通常按照平均每胎产羔2.2只和羔羊平均死亡率3%来进行估算。这样的处理既考虑到了母羊分娩的实际情况，也充分考虑了羔羊的生存风险。结合经验一和经验三，我们可以大致推断出其他不确定因素对本模型可能造成的影响。虽然这些影响难以精确量化，但通过合理的假设和估算，我们可以为模型的预测结果提供一定的参考依据。

**3. 模型假设**

**3.1**

（1）种公羊按不交配时计算空间占用情况。

（2）在交配期时所有公羊全部分配与母羊同栏不占用栏数，非交配期时公羊栏数单独计算。（算上限）

（3）所有母羊怀孕生育周期同步，即只有一批，推出标准羊栏可饲养母羊数从而得出出栏羊只数量。（算下限）

**3.2**

1. 公羊母羊连续生产。
2. 不考虑随机性，所有参数固定，即每批次母羊和公羊的数量相等以及每批次之间差的天数相等。
3. 不考虑因自然交配期种公羊占羊栏数量减少的情况。

**3.3**

3.3.1

（1）仅考虑母羊通过自然交配受孕率为 85%。

（2）不考虑羊栏数量不足的情况。

（3）以出栏羊只最多为目标。

3.3.2

1. 仅考虑以按平均每胎产羔 2.2 只、羔羊平均死亡率 3%估算。

**4. 符号说明**

|  |  |  |
| --- | --- | --- |
| 符号 | 含义 | 单位 |
|  | 一个生育周期内一只母羊及其小羊的资源占用量 | 栏天 |
|  | 空怀修整期时长 | 天 |
|  | 交配期时长 | 天 |
|  | 怀孕期时长 | 天 |
|  | 哺乳期时长 | 天 |
|  | 空怀修整期每只母羊的羊栏占用数量 | 个 |
|  | 交配期每只母羊的羊栏占用数量 | 个 |
|  | 怀孕期每只母羊的羊栏占用数量 | 个 |
|  | 哺乳期每只母羊的羊栏占用数量 | 个 |
|  | 每胎的产羔数量 | 只 |
|  | 羊羔育肥期的时长 | 天 |
|  | 羊羔育肥期羊栏占用数量 | 个 |
|  | 母羊总数 | 只 |
|  | 第j批母羊数量 | 只 |
|  | 第j批母羊及其本周期所产小羊在第i天占用的羊栏数量 | 个 |
|  | 各阶段羊只占用羊栏数 | 个 |
|  | 处于生长阶段k的各批次母羊所需羊栏数 | 个 |
|  | 第i天第j批次处于生长阶段k的母羊所需羊栏数量 | 个 |
|  | 公羊个数 | 只 |
|  | 公羊所需羊栏数 | 个 |

**5. 模型的建立和求解**

**5.1 问题一分区块求解**

**5.1.1 羊只数量上限**

在一个生育周期内，一只基础母羊及其小羊的资源占用量为：

母羊生育周期时长为。

由题目已知条件可知母羊一个生育周期为229天（空怀休整期20天+自然交配期20天+怀孕期149天+哺乳期40天），每胎两只小羊，羊羔育肥期210天。再根据每个阶段给出的每栏羊只数量可知一只基础母羊及其小羊所占的资源总量为：

假设种公羊均按不交配时计算空间占用情况。每天占用栏，公羊母羊按1:50计算，则每只母羊对应的公羊在母羊一个生育周期内占用的资源数量为：

则

两者合计占用59.2938栏天。

若有112个标准羊栏，则饲养母羊数量上限为：

因为母羊只数决定出栏羊只数量，因此可以得出每年出栏羊只上限：

**5.1.2羊只数量下限**

我们让所有母羊的怀孕生育周期完全同步，使之空间利用率降低，就可以根据母羊总数估计出羊栏占用峰值，进一步可以推出母羊数量最大值的下限近似值。

当第一批羊羔进入育肥期，第二批羊羔进入哺乳期时，此时占用羊栏最多：

那么112个标准羊栏可饲养母羊的下限为只，同理可得每年出栏羊只下限约为只。

 综上，112个标准羊栏可以饲养母羊数在342到433之间，出栏羊只数在1090到1380之间。

**5.1.3指定产量目标时的羊栏需求量**

由前问求的上限可得，若希望年化出栏羊只数量1500只，至少需要羊栏数122个。

根据下限可得，至多需要羊栏数155个。

**5.2 问题二的建模与求解**

**5.2.1确定生产方案**

在连续生产的条件下公羊数量比母羊数量小很多，而且由题可知种公羊与基础母羊一般按不低于 1:50 的比例配置，因此先考虑母羊占用羊栏数，再计算公羊数量和公羊所占羊栏数。

我们不考虑随机性，且认定每批次母羊和公羊的数量相等以及每批次之间差的天数相等，那么每一批未来占用羊栏空间的情况都是确定的。因此只需要确定每一批次的开始时间以及母羊数量就可以确定一个生产方案。

前面我们计算过母羊一个生育周期为229天，羊羔育肥期为210天，将两者合并成一个大周期，即为一个439天的生产周期。从母羊空怀修整期第一天开始计算，一只母羊及其小羊所占用羊栏数为下图所示。



**5.2.2 规划模型**

 设母羊的生育周期为D，即229天。当决策变量时，表示该批次不组织生产，最后求解结果中非零的构成实际的生产方案。

由于出栏羊数与母羊数量有对应关系，因此可将优化目标设置为母羊只数最大，因为假设的两年内连续生产，我们可以得出如下规划模型。

其中为羊栏总数，C为730×D的系数矩阵。实际上C的每一列由循环轮换得到。

**5.2.3 基于整数辅助变量求解修正后的规划模型**

实际上同一生产周期内不同生长阶段的羊只不能同栏，且羊只占用羊栏数必须为整数。因此我们需要将上述系数矩阵C按羊只生长阶段拆分为，可以得出

引入整数辅助变量，则修正后的整数规划模型为：

上述模型中不同生产批次的相同生长阶段的羊只有可能同栏，但实际并不可行。因此我们引入整数辅助变量矩阵。

在此基础上再加上公羊数量以及其所需的羊栏数。则应满足以下约束条件。

同时当母羊数量相同时，公羊数量越少占用羊栏数越少。因此完整模型为：

**5.3问题三的假设与求解**

基于问题二所建立的模型我们结合了题目中给出的经验一和经验三对于模型的影响，因为直接关联到羊群繁殖的效率和成活率，从而决定了最终出栏羊只的最大潜力。

由经验一可知母羊自然受孕率为85%，我们只需要把系数矩阵C中除去交配期外的其他相关系数做出相应的改变，因为只有受孕期母羊情况与母羊自然受孕率无关。而经验三中考虑到羔羊出生时可能存在的夭折情况，因此我们需要预留一定的冗余空间来应对这种损失，按以平均每胎产羔 2.2 只，羔羊平均死亡率3%估算，在我们的模型中略加修改，在羔羊育肥期中210的系数变成2.2/14再乘上0.97即可得到所需的出栏羊只数量。

**6. 模型评价与误差分析**

**6.1问题一结果检验与误差分析**

问题一中我们建立的模型所得出的上下限太宽，考虑的因素过于极限，会影响出栏羊只数量和标准羊栏数量的缺口的因素考虑的不够多。我们需要在此基础上考虑羊栏只能是整数个，且不同阶段羊只不能同栏，所得出的下限与我们模型中得出的下限可能有出入，可以得出更精确的下限近似值。

**6.2问题二结果检验与误差分析**

问题二中我们建立的模型约束条件也不完整，我们规定了每批次母羊和公羊的数量相等以及每批次之间差的天数相等。而这些固定值在实际中无法做到将羊栏做到最大程度的利用率。因此我们需要保持批次间的间隔时间相同，在此基础上改变每一批母羊的数量，而不是规定每批母羊数量也相同。这样就能得出每批不同母羊个数得出利用率更高的解。

事实上在此基础上仍存在一定的改进空间。倘若总批次也是未知情况，那么该模型规模会变得非常庞大复杂，我们应该先来考虑每批母羊数量的合理取值。因为当每批母羊数量在小范围内波动时，对羊栏的需求量是保持不变的。比如在本题中一批母羊数量在1-6之间时对后续羊栏需求量是相同的。在此基础上优化模型应该可以搜索得到母羊数量之间更为合理的分界点。

在得到所有合理取值之后，对于给定的母羊总数，批次的组合只有有限多个，我们也可以使用动态规划或者用遍历的算法得出所有组合，并检测这些组合能否满足所有约束，对这些进行筛选得出最优方案。

**6.3问题三误差分析**

我们对于第三问中考虑的方面比较片面，在经验一和经验三中也考虑的并不完整。

首先对于经验一来说母羊交配期结束后30天就可以通过观察来判断母羊是否受孕成功，这对于母羊受孕率在模型中的假设还有很大的影响，使模型更加复杂。实验三中每胎羔羊数量不同也会影响到模型的运算和统计，多羔死亡率高需要进一步用到数据分析与决策支持模型，而不选择按每胎2.2只来算。

而经验二在自然交配的20天中受孕母羊的受孕时间并不确知，而孕期会在 147-150 天内波动，这些因素将影响到预产期范围。这些都会影响到母羊生育周期及其羔羊育肥期的天数变化。使模型变数更大，考虑范围更广。

经验四也是对于母羊生育周期的进一步判断分析。由于哺乳期过短不利于羔羊后期的生长，通常是羔羊体重达到一定标准后断奶；而哺乳期过长，母羊的身体消耗就越大，早点断奶，有利于早恢复、早发情配种。除此之外，母羊的空怀休整期可在不少于18天的前提下灵活调控。

**7. 结果检验**

**7.1 结果：**

母羊最多个数为433

母羊最少的个数342.0

与理论值一致

**7.2 结果：**

396.0 [22. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 22. 0. 0. 0. 0.

 0. 0. 0. 0. 0. 0. 0. 0. 22. 0. 0. 0. 0. 0. 0. 0. 0. 0.

 0. 0. 0. 22. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 22. 0.

 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 22. 0. 0. 0. 0. 0. 0.

 0. 0. 0. 0. 0. 0. 22. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.

 0. 22. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 22. 0. 0. 0.

 0. 0. 0. 0. 0. 0. 0. 0. 0. 22. 0. 0. 0. 0. 0. 0. 0. 0.

 0. 0. 0. 0. 22. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 22.

 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 22. 0. 0. 0. 0. 0.

 0. 0. 0. 0. 0. 0. 0. 22. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.

 0. 0. 22. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 22. 0. 0.

 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 22. 0. 0. 0. 0. 0. 0. 0.

 0. 0. 0. 0. 0. 22. 0. 0. 0. 0. 0. 0. 0.]

即母羊最多为396，实现此方案为每批22只母羊，间隔13天。

**7.3（1）结果：**

母羊数460.0

在两年时间内共多余25238.0个围栏

1. **结果：**

母羊数378.0

在两年时间内共多余25304.0个围栏

**分析：**分别添加了经验1，经验3后，母羊的个数也产生了较为明显的变化。因此，在实际问题中，有了不稳定因素的加入，我们很难得出一个绝对的最优解，需要根据现实的情况进行进一步的判断，这样才能使得整体的损失更小。

**8. 程序附录**

**附录1：问题一**

import math

import numpy as np

#初始

natural\_mating=20

pregnancy=149

every\_fetus=2

l\_period=40

f\_period=210

rest=20

total\_field=112

least\_target=1500

#条件

max\_each\_field\_f=14

max\_each\_field\_m=4

max\_natural\_f=14

max\_natural\_m=1

max\_pregnancy=8

max\_l=6

max\_f=14

#最大值

total\_day=natural\_mating+pregnancy+l\_period+rest

e\_r\_f=natural\_mating\*(1/max\_each\_field\_f)+pregnancy\*(1/max\_pregnancy)+l\_period\*(1/max\_l)+rest\*(1/max\_each\_field\_f)+every\_fetus\*f\_period\*(1/max\_f)

e\_r\_m=1/50\*(1/max\_each\_field\_m)\*(total\_day-natural\_mating)

e\_r\_s=e\_r\_f+e\_r\_m

n=365/total\_day

sheep\_max=math.floor(total\_field\*total\_day/e\_r\_s)

print(f"最多个数为{sheep\_max}")

#最小值(第一批羔羊进入育肥期，第二批进入哺育期)

n=2/14+1/6+1/14\*1/4

n\_min=np.floor(112/n)

n\_min

print(f"最少的个数{n\_min}")

图像程序：

import matplotlib.pyplot as plt

# 定义x和y的范围和间隔

x = list(range(0, 410, 10))

y = [1/14 if i <= 40 else (1/8 if i <= 189 else (1/6 if i <= 209 else 1/7)) for i in x]

new\_x = [val + 9 for val in x]

# 绘制图形

plt.step(new\_x, y)

plt.xticks(range(0, 410, 100))

plt.yticks([0.06, 0.08, 0.1, 0.12, 0.14, 0.16, 0.18])

plt.xlabel('X')

plt.ylabel('Y')

plt.title('Graph of Y as a Function of X')

plt.axis([0, 400, 0.06, 0.18])

plt.grid(True)

plt.show()

**附录2：问题二**

import numpy as np

sum\_l =0

current\_max\_sum = None

current\_max\_matrix = None

def fun1(X):

 global current\_max\_sum, current\_max\_matrix

 # 如果X是一维数组，直接计算其和；否则，遍历每一行计算总和

 matrix\_sum = sum(X) if X.ndim == 1 else sum(sum(row) for row in X)

 if current\_max\_sum is None or matrix\_sum > current\_max\_sum:

 current\_max\_sum = matrix\_sum

 current\_max\_matrix = X

 return current\_max\_matrix, current\_max\_sum

def fun2(a3):

 x = np.zeros(229)

 t = np.zeros((750, 229))

 t1 = np.zeros((978, 229))

 # 批次距离dis和个数n

 for dis in range(1,227):

 for n in range(0,441):

 for j in range(0, 229, dis):

 x[j] = n

 t = np.ceil(a3 \* x)

 t1[:750, 0] = t[:, 0]

 for col in range(1, 229):

 t1[col:col+750, col] = t[:, col]

 row\_sums = np.sum(t1, axis=1)

 max\_value = np.max(row\_sums)

 matrix\_sum = sum(x)

 nr\_min=np.ceil(matrix\_sum/50)

 if max\_value>= 112-1/4\*(nr\_min):

 x = np.zeros(229)

 t = np.zeros((750, 229))

 t1 = np.zeros((978, 229))

 else:

 fun1(x)

 x = np.zeros(229)

 t = np.zeros((750, 229))

 t1 = np.zeros((978, 229))

a1 = np.array([1/14, 1/14, 1/8, 1/6, 2/14])#空间利用率矩阵

a3= np.zeros((750,229))

for i in range(0,20):

 a3[i][0]=a1[1]

for i in range(20,169):

 a3[i][0]=a1[2]

for i in range(169,209):

 a3[i][0]=a1[3]

for i in range(209,229):

 a3[i][0]=a1[4]+a1[0]

for i in range(229,249):

 a3[i][0]=a1[4]+a1[1]

for i in range(249,398):

 a3[i][0]=a1[4]+a1[2]

for i in range(398,438):

 a3[i][0]=a1[4]+a1[3]

for i in range(438,439):

 a3[i][0]=a1[3]

for i in range(439,459):

 a3[i][0]=a1[0]+a1[4]

for i in range(459,479):

 a3[i][0]=a1[1]+ a1[4]

for i in range(479,628):

 a3[i][0]=a1[2]+ a1[4]

for i in range(628,649):

 a3[i][0]=a1[3]+ a1[4]

for i in range(649,668):

 a3[i][0]=a1[3]

for i in range(668,688):

 a3[i][0]=a1[0]+ a1[4]

for i in range(688,708):

 a3[i][0]=a1[1]+ a1[4]

for i in range(708,750):

 a3[i][0]=a1[2]+ a1[4]

first\_col = a3[:, 0]

# 将第一列复制到整个矩阵中，每列都是第一列的复制

a3 = np.tile(first\_col, (229, 1)).T

fun2(a3)

print(current\_max\_sum, current\_max\_matrix)

**附录3：问题三**

**（1）**

import numpy as np

sum\_l =0

current\_max\_sum = None

current\_max\_matrix = None

def fun1(X):

 global current\_max\_sum, current\_max\_matrix

 # 如果X是一维数组，直接计算其和；否则，遍历每一行计算总和

 matrix\_sum = sum(X) if X.ndim == 1 else sum(sum(row) for row in X)

 if current\_max\_sum is None or matrix\_sum > current\_max\_sum:

 current\_max\_sum = matrix\_sum

 current\_max\_matrix = X

 return current\_max\_matrix, current\_max\_sum

def fun3(a4):

 global sum\_l

 sum\_l=np.sum(112- a4)

 return 0

def fun2(a3):

 x = np.zeros(229)

 t = np.zeros((750, 229))

 t1 = np.zeros((978, 229))

 # 批次距离dis和个数n

 for dis in range(1,227):

 for n in range(0,441):

 for j in range(0, 229, dis):

 x[j] = n

 t = np.ceil(a3 \* x)

 t1[:750, 0] = t[:, 0]

 for col in range(1, 229):

 t1[col:col+750, col] = t[:, col]

 row\_sums = np.sum(t1, axis=1)

 max\_value = np.max(row\_sums)

 matrix\_sum = sum(x)

 nr\_min=np.ceil(matrix\_sum/50)

 if max\_value>= 112-1/4\*(nr\_min):

 x = np.zeros(229)

 t = np.zeros((750, 229))

 t1 = np.zeros((978, 229))

 else:

 fun1(x)

 fun3(x)

 x = np.zeros(229)

 t = np.zeros((750, 229))

 t1 = np.zeros((978, 229))

a1 = np.array([1/14, 1/14\*0.85, 1/8\*0.85, 1/6\*0.85, 2/14\*0.85])#空间利用率矩阵

a3= np.zeros((750,229))

for i in range(0,20):

 a3[i][0]=a1[1]

for i in range(20,169):

 a3[i][0]=a1[2]

for i in range(169,209):

 a3[i][0]=a1[3]

for i in range(209,229):

 a3[i][0]=a1[4]+a1[0]

for i in range(229,249):

 a3[i][0]=a1[4]+a1[1]

for i in range(249,398):

 a3[i][0]=a1[4]+a1[2]

for i in range(398,438):

 a3[i][0]=a1[4]+a1[3]

for i in range(438,439):

 a3[i][0]=a1[3]

for i in range(439,459):

 a3[i][0]=a1[0]+a1[4]

for i in range(459,479):

 a3[i][0]=a1[1]+ a1[4]

for i in range(479,628):

 a3[i][0]=a1[2]+ a1[4]

for i in range(628,649):

 a3[i][0]=a1[3]+ a1[4]

for i in range(649,668):

 a3[i][0]=a1[3]

for i in range(668,688):

 a3[i][0]=a1[0]+ a1[4]

for i in range(688,708):

 a3[i][0]=a1[1]+ a1[4]

for i in range(708,750):

 a3[i][0]=a1[2]+ a1[4]

first\_col = a3[:, 0]

# 将第一列复制到整个矩阵中，每列都是第一列的复制

a3 = np.tile(first\_col, (229, 1)).T

fun2(a3)

print(current\_max\_sum, current\_max\_matrix)

print(f"在两年时间内共缺{sum\_l}个围栏")

**（2）**

import numpy as np

sum\_l =0

current\_max\_sum = None

current\_max\_matrix = None

def fun1(X):

 global current\_max\_sum, current\_max\_matrix

 # 如果X是一维数组，直接计算其和；否则，遍历每一行计算总和

 matrix\_sum = sum(X) if X.ndim == 1 else sum(sum(row) for row in X)

 if current\_max\_sum is None or matrix\_sum > current\_max\_sum:

 current\_max\_sum = matrix\_sum

 current\_max\_matrix = X

 return current\_max\_matrix, current\_max\_sum

def fun3(a4):

 global sum\_l

 sum\_l=np.sum(112- a4)

 return 0

def fun2(a3):

 x = np.zeros(229)

 t = np.zeros((750, 229))

 t1 = np.zeros((978, 229))

 # 批次距离dis和个数n

 for dis in range(1,227):

 for n in range(0,441):

 for j in range(0, 229, dis):

 x[j] = n

 t = np.ceil(a3 \* x)

 t1[:750, 0] = t[:, 0]

 for col in range(1, 229):

 t1[col:col+750, col] = t[:, col]

 row\_sums = np.sum(t1, axis=1)

 max\_value = np.max(row\_sums)

 matrix\_sum = sum(x)

 nr\_min=np.ceil(matrix\_sum/50)

 if max\_value>= 112-1/4\*(nr\_min):

 x = np.zeros(229)

 t = np.zeros((750, 229))

 t1 = np.zeros((978, 229))

 else:

 fun1(x)

 fun3(x)

 x = np.zeros(229)

 t = np.zeros((750, 229))

 t1 = np.zeros((978, 229))

cunhuolv=0.97

a1 = np.array([1/14, 1/14, 1/8, 1/6, 2.2/14\*cunhuolv])#空间利用率矩阵

a3= np.zeros((750,229))

for i in range(0,20):

 a3[i][0]=a1[1]

for i in range(20,169):

 a3[i][0]=a1[2]

for i in range(169,209):

 a3[i][0]=a1[3]

for i in range(209,229):

 a3[i][0]=a1[4]+a1[0]

for i in range(229,249):

 a3[i][0]=a1[4]+a1[1]

for i in range(249,398):

 a3[i][0]=a1[4]+a1[2]

for i in range(398,438):

 a3[i][0]=a1[4]+a1[3]

for i in range(438,439):

 a3[i][0]=a1[3]

for i in range(439,459):

 a3[i][0]=a1[0]+a1[4]

for i in range(459,479):

 a3[i][0]=a1[1]+ a1[4]

for i in range(479,628):

 a3[i][0]=a1[2]+ a1[4]

for i in range(628,649):

 a3[i][0]=a1[3]+ a1[4]

for i in range(649,668):

 a3[i][0]=a1[3]

for i in range(668,688):

 a3[i][0]=a1[0]+ a1[4]

for i in range(688,708):

 a3[i][0]=a1[1]+ a1[4]

for i in range(708,750):

 a3[i][0]=a1[2]+ a1[4]

first\_col = a3[:, 0]

# 将第一列复制到整个矩阵中，每列都是第一列的复制

a3 = np.tile(first\_col, (229, 1)).T

fun2(a3)

print(current\_max\_sum, current\_max\_matrix)

print(f"在两年时间内共缺{sum\_l}个围栏")

**课程报告成绩评阅表**

|  |  |  |  |
| --- | --- | --- | --- |
| **考查内容** | **具 体 要 求** | **分值** | **评 分** |
| **A** | **B** | **C** | **D** | **E** | **得分** |
| **报告质量** | 逻辑清晰，文字通顺，格式规范，图表清楚，论述充分 | 20 | 20 | 18 | 16 | 14 | ≤12 |  |
| 数学模型理论分析深刻，能够对数学模型编程求解，计算结论严谨合理等 | 50 | 50 | 45 | 40 | 35 | ≤30 |  |
| **答辩表现** | 能够介绍对所选问题的理解和完成的主要工作，表达清楚，重点突出，能正确并流利地回答出老师提出的问题 | 20 | 20 | 18 | 16 | 14 | ≤12 |  |
| **整体评价** | 对模型理解深刻，有分析问题和解决问题的能力，有自己的见解 | 10 | 10 | 9 | 8 | 7 | ≤6 |  |
| **成 绩** |  |

教师签名：

日 期：