基于 xx 的 xx 的研究

摘要

本文研究的是 xx，通过建立 xx 模型和 xx 模型，求解得到 xx，获得 xx。

首先采用 xx 对已知数据进行（插值补全），为模型建立提供数据基础。

针对问题一： 本文对 xx 建立了 xx 模型，描述...（针对...）【重要模型和重要结果加粗处理】

针对问题二： 建立模型 + 模型参数设置 + 方法阐述（中间过程结果可适当分析）+ 所得结果（展示所有要求结果，可不进行分析）

针对问题三： 本题需要 xxx（做什么）。按做法步骤，关键部分展示，最后写出数据结果

针对问题四： 本题提出 xx，在问题 x 的基础上，...。【每个部分用一句话概括】

关键词： **重要研究对象 主要模型 主要求解方法**

1 问题的背景与重述

1.1 问题的背景

研究意义和研究进展

因此探究 xx 问题具有重要的研究价值。

1.2 问题的重述

根据 xx 观察数据，解决下列问题：

（1）用一两句话改写题目内容：主要包括需要完成什么任务

（2）

（3）

（4）

2 问题分析

对于问题一：问题解决的思路展现（讨论时确定的框架-会议记录）

对于问题二：不需要呈现结果

对于问题三：或可增加基于 xx 数据图，我们可以发现...，因而采用...（选择理由可简单阐述）

对于问题四：

3 模型假设

1. 假设（题中假设）
2. 假设所给数据均真实可靠
3. 做题过程中的前提假设

4 符号说明

符号	含义	单位
$S_i(x)$	第 i 段 x 的三次样条插值函数	/
x(T)	温度为 T 时的乙醇转化率	%

5 数据预处理

首先我们对附件 x 中的数据进行观察分析，发现 xx（低质量数据问题），这些都给后续的分析带来不便，因此我们用 xx，令 xx，（对其中的数据进行插值补全）。

处理方法简述

处理后的数据结果展示（图/表）

在（表一）中，xxx，（增加了数据点数量），可以有效地提高问题中模型拟合的准确性。

6 模型的建立与求解

注：模型建立在做完后有足够时间考虑加分项-改进经典模型，创新性；同一问题使用两个或以上合理模型进行求解，需做对比和评价

模型求解-包含算法设计与选择（原理、思想、依据、采用软件理由及名称等），算法步骤（对求解无帮助的计算过程和中间结果不需列出），算法实现；另：1. 命题叙述符合数学命题　2. 要求回答的问题（数值结果，结论）逐项解答，结论明确　3. 列出多组数据进行比较、分析　4. 运用流程图、模式图、数据表灵活展示结果

模型检验-包括结果正确性的分析、检验，模型合理性的分析、检验

6.1 问题一模型的建立与求解

6.1.1 数据可视化

为了更加直观方便地分析 xx 与 xx 之间的关系，我们考虑对 xx 数据进行可视化，画出 xx 如下：

方程拟合与结论分析

注：重点给出从数据中获取的信息，为建模作铺垫

6.1.2 （第一小问）

将一个大问转化成几个小问，按顺序阐述

6.1.2.1 模型的建立

通过图/题目信息获取信息，给出对应模型，需要说明模型思考的来源

对于较复杂问题，可以采用:

Step1

Step2

Step3

6.1.2.2 结果及分析

算法求解、数据展示、数据分析、结果展示、结果分析，不符合预期的异常值分析，得出最终结论

数据结果或表太长可以加在附录当中

6.1.3 （第二小问）

6.1.3.1 模型的建立

6.1.3.2 结果及分析

6.1.4 （补充内容）

展示优势的部分，加上思考延伸的内容，如：缺失值的预测等

6.2 问题二模型的建立与求解

6.2.1 xx 模型的建立

模型的相关阐释，并解释为什么用这种模型（概览）

【三线表】

表 1: 这是一张三线表

姓名	学号	性别
Steve Jobs	001	Male
Bill Gates	002	Female

以上：table 有若干可选参数 [!htbp]

h 代表 here, 将表格排在当前文字位置

t 表示将表格放在下一页的 top (页首)

b 表示将表格放在当前页的 bottom (底部)

! 表示忽略美观因素，尽可能按照参数指定的方式来处理表格浮动位置。

表格将会按照所给参数，依次尝试按照每个参数进行排版，当无法排版时，将会按照下一个参数

【单元格合并】

S_i		事件				max
		50	100	150	200	
策略	50	0	100	200	300	300
	100	100	0	100	200	200
	150	200	100	0	100	200
	200	300	200	100	0	300

【斜线表头】

$\alpha_{i,j}$ 乙 / 甲	β_1	β_2	β_3
α_1	-4	0	-8
α_2	3	2	4
α_3	16	1	-9
α_4	-1	1	7

【混合类型】

需要添加参数 \ diagbox[innerwidth=2cm](参数大小取决于列宽度) 解决

λ_i / S_i		事件				max
		50	100	150	200	
策略	50	0	100	200	300	300
	100	100	0	100	200	200
	150	200	100	0	100	200
	200	300	200	100	0	300

注：若模型阐释有许多前提假设，设立条件，可以一步步细化：

a.

b.

c. 构建模型

公式表达方式【大括号】

$$h_0(x) = \begin{cases} (1+2\frac{x-x_0}{x_1-x_0})(\frac{x-x_1}{x_0-x_1})^2 & x_0 \le x \le x_1 \\ 0 \quad x_1 < x \le x_n \end{cases} \tag{1}$$

$$F^{HLLC} = \begin{cases} F_L & 0 < S_L \\ F_L^* & S_L \le 0 < S_M \\ F_R^* & S_M \le 0 < S_R \\ F_R & S_R \le 0 \end{cases} \tag{2}$$

6.2.2 模型的求解

若可以借助工具直接解出，可以在这部分证明用这个工具的前提条件是否成立

6.2.3 模型的结果与分析

若利用工具，如 SPSS，需解释各数据的实际意义，再得出结论

尽可能展示每一步求解的方法与结果，并分别给出结论，最后整合（可以结合第一问中得出的结论，使文章一脉相承）

6.3 问题三模型的建立与求解

本题的目标 + 总体思路

6.3.1 xx 的（拟合）模型

Step1

Step2

6.3.2 xx 的（优化）模型

利用 xx，我们可以建立 xx（单目标规划模型），来获得...

1）**决策变量：**解释 + 具体符号/公式

2）**目标函数：**

3）**约束条件：**

① xx 约束：

$$33 \le x_1 \le 200 \tag{3}$$

② xx 约束：

$$33 \le x_2 \le 200 \tag{4}$$

综上所述，对 xx 建立优化模型如下：

$$max \quad y = 1.1x_1 + 1.2x_2$$

$$s.t.\begin{cases} 33 \leq x_1 \leq 200 \\ 33 \leq x_2 \leq 200 \end{cases} \tag{5}$$

6.3.3 结果的对比分析

插入表/图
对数字结果进行分析（由上表数据...）

6.3.4 结果的检验

根据上表数据，可以发现...
与 xx 结论相符，从侧面验证了我们结论的正确性。

6.4 问题四的分析解决

注：一般为开放性问题，可以将思考得到的方面作为不同步分展开阐述。
需写出思考来源，如由上文结论，数据分析，文献资料，附件数据等
需加上扩展该部分内容的价值（现实意义等）

7 模型的评价与推广

7.1 模型的优点

1、着力讲文章的创新点与优势，例如：补全数据点增加准确性...
2、本文做了大量的图表来统计分析数据特点，更加直观地...
3、...

7.2 模型的缺点

1、结合模型假设对模型缺点点评，每个缺点写完后可以加改进措施/想法。
2、...

7.3 灵敏度分析

对数据提出的假设做分析，最好结合实际说明为什么可能出现这种情况，对应现实中的结果会如何改变。

不同的模型灵敏度分析的方面不同，共分为决策型模型（优化），动态模型，概率模型，线性回归/时间序列，涉及这些方面时需做相关探究。

7.4 模型的推广

本文主要建立的 xx 模型有哪些优点，有什么价值，适宜推广到哪些领域。以及本文所建立的 xx 模型有哪些优点，适合推广到 xx 等相关问题/研究工作中去。

参考文献

[1] 陈立辉, 苏伟, 蔡川　基于 *latex* 的 *web* 数学公式提取方法研究 [J], 计算机科学，2017(06)

[2] xx

[3] William H ,Press,Saul A ,Teukolsky,William T ,VeterLing ,Brian P.Flannery,*Numerical Recipes 3rd Edition:The Art of Scientific Computing* Cambridge University Press ,New York,2007

附录

附录说明:

1.xx 图

2.xx 表

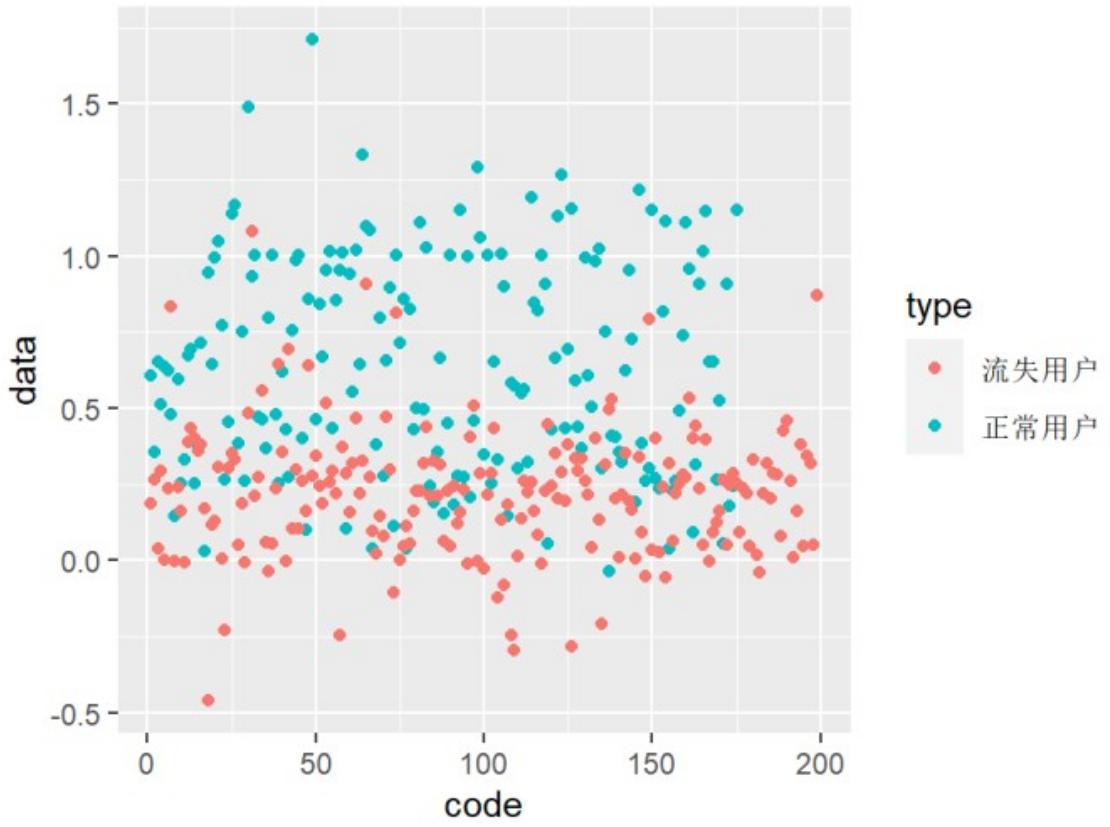
3.xx 程序

4.xx 程序

5.xx 程序

6.xx 程序

附录 1



附录 2

数字	1	2	3	4	5
字母	A	B	C	D	E
天干	甲	乙	丙	丁	戊

附录 3

本部分代码使用的软件是 Python

```
# %%
import pandas as pd
```

```
from sklearn.tree import DecisionTreeClassifier as dtc
import numpy as np
from sklearn.tree import plot_tree
from sklearn.model_selection import RandomizedSearchCV
import seaborn as sns
import matplotlib.pyplot as plt
pd.set_option('display.max_columns',None)

# %%
df=pd.read_csv("C:/Users/Krebs/Desktop/task3_train_data.txt",header=None,sep=' ')

# %%
df_1=pd.read_csv("C:/Users/Krebs/Desktop/data/附件1测试数据",header=None,sep=' ')

# %%
train_x=df.iloc[:,1:101]
train_y=df.iloc[:,101]

# %%
param_grid={
    'min_samples_leaf':np.arange(0.05,0.75,0.05),
    'max_leaf_nodes':range(2,50,1),
    'random_state':range(0,100)
}
rs=RandomizedSearchCV(dtc(),param_grid,n_iter=1000,cv=3,scoring='f1',n_jobs=1)
# %%
rs.fit(X=train_x,y=train_y)
best_para=rs.best_params_

dt=dtc(best_para)

# %%
dt.fit(X=train_x,y=train_y)

# %%
dt.score(X=train_x,y=train_y)

# %%
df[102]=dt.predict(X=train_x)

# %%
df_1[102]=dt.predict(X=df_1.iloc[:,1:101])

# %%
df_1[102].value_counts()

# %%
tp=df[(df[101]==1) & (df[102]==1)].shape[0]
fp=df[(df[101]==0) & (df[102]==1)].shape[0]
pcs=tp/(tp+fp)
pcs

# %%
fn=df[(df[101]==1) & (df[102]==0)].shape[0]
rc=tp/(tp+fn)
fn

# %%
F1=2*pcs*rc/(pcs+rc)
```

```
F1

# %%
plt.figure(figsize=(20,20))
plot_tree(dt)
plt.title('Tree')
plt.savefig('tree.jpg')

# %%
plt.figure(figsize=(10,10))
sns.countplot(x=df[102],hue=df[101])
plt.xlabel('predict')
plt.legend(['0','1'])
plt.savefig('2.jpg')

# %%
t=pd.read_csv("C:/Users/Krebs/Desktop/data/data4_1.data",header=None,sep=' ')
ddf=pd.DataFrame(t[0])
ddf

# %%
for i in range(1,13):
    loc='C:/Users/Krebs/Desktop/data/data4_'+str(i)+'.data'
    t=pd.read_csv(loc,header=None,sep=' ')
    p_y=dt.predict(X=t.iloc[:,1:101])
    ddf['mon_'+str(i)]=p_y
ddf

# %%
ddf.describe()

# %%
grade=[]
for i in range(0,200):
    val=ddf.iloc[i,1:13].values
    if 1 not in val:
        grade+='A'
    else:
        for j in range(11,-1,-1):
            if val[j]==1:
                j+=2
                break
        if j<=3:
            grade+='B'
        elif j<=6:
            grade+='C'
        else:
            grade+='D'
ddf['grade']=grade
ddf

# %%
ddf.rename(columns={0:'id'},inplace=True)

# %%
ans=ddf[ddf['mon_12']==0][['id','grade']]
ans.to_csv('附件5.csv')

# %%
```

```
plt.figure(figsize=(10,10))
sns.countplot(x=df_1[102])
plt.xlabel('predict')
plt.savefig('3.jpg')
```

附录 4

本部分代码使用的软件是 Matlab

```
close all; clear all; clc

results=[0.0000000000 0.1688693098 0.3502202643 0.1798825257 0.2540381791 0.0932452276 0.1497797357
    0.1328928047 0.2408223201 0.0117474302 0.0433186490 0.5491923642 0.2569750367 0.4302496329
    0.0726872247 0.2305433186 0.0007342144 0.6938325991 0.2540381791 0.6284875184 0.0976505140
    0.1108663730 0.0792951542 0.0146842878 0.0778267254 0.0154185022 0.0000000000 0.0000000000 ,
0.1688693098 0.0000000000 0.0051395007 0.0088105727 0.1013215859 0.0000000000 0.0036710720
    0.0044052863 0.1064610866 0.0051395007 0.0014684288 0.1358296623 0.0418502203 0.0058737151
    0.1152716593 0.0022026432 0.0000000000 0.1035242291 0.0256975037 0.0301027900 0.1027900147
    0.0022026432 0.0022026432 0.0007342144 0.0168869310 0.0000000000 0.0000000000 0.0007342144 ,
0.3502202643 0.0051395007 0.0000000000 0.0293685756 0.2892804699 0.0286343612 0.0029368576
    0.1541850220 0.3113069016 0.0029368576 0.0704845815 0.1226138032 0.0110132159 0.1358296623
    0.4676945668 0.0227606461 0.0029368576 0.1314243759 0.1035242291 0.1930983847 0.1424375918
    0.0051395007 0.0007342144 0.0190895742 0.0403817915 0.0007342144 0.0000000000 0.0036710720 ,
0.1798825257 0.0088105727 0.0293685756 0.0000000000 0.3891336270 0.0044052863 0.0227606461
    0.0066079295 0.3223201175 0.0029368576 0.0022026432 0.0572687225 0.0190895742 0.2195301028
    0.1299559471 0.0058737151 0.0007342144 0.1218795888 0.0176211454 0.0234948605 0.0895741557
    0.0198237885 0.0124816446 0.0007342144 0.0359765051 0.0007342144 0.0007342144 0.0022026432 ,
0.2540381791 0.1013215859 0.2892804699 0.3891336270 0.0000000000 0.1262848752 0.1938325991
    0.1600587372 0.1255506608 0.0242290749 0.0690161527 0.4581497797 0.2885462555 0.5102790015
    0.0491923642 0.2422907489 0.0220264317 1.0000000000 0.4177679883 0.5337738620 0.0712187959
    0.2665198238 0.0807635830 0.1013215859 0.0506607930 0.0381791483 0.0007342144 0.0022026432 ,
0.0932452276 0.0000000000 0.0286343612 0.0044052863 0.1262848752 0.0000000000 0.0022026432
    0.0044052863 0.1402349486 0.0000000000 0.0022026432 0.0690161527 0.0022026432 0.0528634361
    0.1049926579 0.0007342144 0.0000000000 0.0616740088 0.0088105727 0.0359765051 0.0712187959
    0.0000000000 0.0036710720 0.0000000000 0.0234948605 0.0000000000 0.0007342144 0.0007342144 ,
0.1497797357 0.0036710720 0.0029368576 0.0227606461 0.1938325991 0.0022026432 0.0000000000
    0.0580029369 0.1930983847 0.0000000000 0.0022026432 0.0668135095 0.0124816446 0.2246696035
    0.0712187959 0.0022026432 0.0000000000 0.1556534508 0.0124816446 0.0301027900 0.0829662261
    0.0029368576 0.0007342144 0.0000000000 0.0168869310 0.0007342144 0.0014684288 0.0022026432 ,
0.1328928047 0.0044052863 0.1541850220 0.0066079295 0.1600587372 0.0044052863 0.0580029369
    0.0000000000 0.1123348018 0.0000000000 0.0000000000 0.0102790015 0.0088105727 0.0205580029
    0.1255506608 0.0477239354 0.0007342144 0.0411160059 0.1101321586 0.1982378855 0.0425844347
    0.0000000000 0.0367107195 0.0036710720 0.0279001468 0.0000000000 0.0000000000 0.0007342144 ,
0.2408223201 0.1064610866 0.3113069016 0.3223201175 0.1255506608 0.1402349486 0.1930983847
    0.1123348018 0.0000000000 0.0007342144 0.0381791483 0.4089574156 0.2679882526 0.7143906021
    0.2444933921 0.1233480176 0.0044052863 0.3803230543 0.4295154185 0.6314243759 0.0763582966
    0.2136563877 0.0506607930 0.0330396476 0.0227606461 0.0374449339 0.0007342144 0.0007342144 ,
0.0117474302 0.0051395007 0.0029368576 0.0029368576 0.0242290749 0.0000000000 0.0000000000
    0.0000000000 0.0007342144 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0058737151
    0.0176211454 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0198237885
    0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 ,
0.0433186490 0.0014684288 0.0704845815 0.0022026432 0.0690161527 0.0022026432 0.0022026432
    0.0000000000 0.0381791483 0.0000000000 0.0000000000 0.0198237885 0.0022026432 0.0462555066
    0.0242290749 0.0014684288 0.0000000000 0.0264317181 0.0293685756 0.0029368576 0.0029368576
    0.0007342144 0.0044052863 0.0000000000 0.0117474302 0.0000000000 0.0000000000 0.0007342144 ,
0.5491923642 0.1358296623 0.1226138032 0.0572687225 0.4581497797 0.0690161527 0.0668135095
    0.0102790015 0.4089574156 0.0000000000 0.0198237885 0.0000000000 0.0264317181 0.0580029369
    0.3120411160 0.1248164464 0.0000000000 0.0455212922 0.0690161527 0.1358296623 0.2239353891
```

```
    0.0293685756 0.0124816446 0.0022026432 0.1424375918 0.0139500734 0.0014684288 0.0044052863 ,
0.2569750367 0.0418502203 0.0110132159 0.0190895742 0.2885462555 0.0022026432 0.0124816446
    0.0088105727 0.2679882526 0.0000000000 0.0022026432 0.0264317181 0.0000000000 0.0704845815
    0.2621145374 0.1387665198 0.0000000000 0.0807635830 0.0506607930 0.0697503671 0.1152716593
    0.0007342144 0.0007342144 0.0029368576 0.0301027900 0.0022026432 0.0000000000 0.0000000000 ,
0.4302496329 0.0058737151 0.1358296623 0.2195301028 0.5102790015 0.0528634361 0.2246696035
    0.0205580029 0.7143906021 0.0058737151 0.0462555066 0.0580029369 0.0704845815 0.0000000000
    0.5704845815 0.0176211454 0.0088105727 0.0917767988 0.1718061674 0.4265785609 0.1732745962
    0.0660792952 0.0242290749 0.0014684288 0.0440528634 0.0058737151 0.0000000000 0.0000000000 ,
0.0726872247 0.1152716593 0.4676945668 0.1299559471 0.0491923642 0.1049926579 0.0712187959
    0.1255506608 0.2444933921 0.0176211454 0.0242290749 0.3120411160 0.2621145374 0.5704845815
    0.0000000000 0.1916299559 0.0014684288 0.5345080764 0.2114537445 0.3024963289 0.1938325991
    0.0976505140 0.0998531571 0.0117474302 0.0403817915 0.0080763583 0.0014684288 0.0014684288 ,
0.2305433186 0.0022026432 0.0227606461 0.0058737151 0.2422907489 0.0007342144 0.0022026432
    0.0477239354 0.1233480176 0.0000000000 0.0014684288 0.1248164464 0.1387665198 0.0176211454
    0.1916299559 0.0000000000 0.0000000000 0.2415565345 0.1350954479 0.0616740088 0.1035242291
    0.0000000000 0.0022026432 0.0301027900 0.0161527166 0.0000000000 0.0000000000 0.0014684288 ,
0.0007342144 0.0000000000 0.0029368576 0.0007342144 0.0220264317 0.0000000000 0.0000000000
    0.0007342144 0.0044052863 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0088105727
    0.0014684288 0.0000000000 0.0000000000 0.0000000000 0.0036710720 0.0000000000 0.0631424376
    0.0000000000 0.0000000000 0.0007342144 0.0000000000 0.0000000000 0.0000000000 0.0000000000 ,
0.6938325991 0.1035242291 0.1314243759 0.1218795888 1.0000000000 0.0616740088 0.1556534508
    0.0411160059 0.3803230543 0.0000000000 0.0264317181 0.0455212922 0.0807635830 0.0917767988
    0.5345080764 0.2415565345 0.0000000000 0.0000000000 0.0881057269 0.4508076358 0.2466960352
    0.0528634361 0.0264317181 0.0051395007 0.1108663730 0.0022026432 0.0000000000 0.0044052863 ,
0.2540381791 0.0256975037 0.1035242291 0.0176211454 0.4177679883 0.0088105727 0.0124816446
    0.1101321586 0.4295154185 0.0000000000 0.0293685756 0.0690161527 0.0506607930 0.1718061674
    0.2114537445 0.1350954479 0.0036710720 0.0881057269 0.0000000000 0.4096916300 0.2841409692
    0.0168869310 0.0220264317 0.0000000000 0.0506607930 0.0000000000 0.0000000000 0.0000000000 ,
0.6284875184 0.0301027900 0.1930983847 0.0234948605 0.5337738620 0.0359765051 0.0301027900
    0.1982378855 0.6314243759 0.0000000000 0.0029368576 0.1358296623 0.0697503671 0.4265785609
    0.3024963289 0.0616740088 0.0000000000 0.4508076358 0.4096916300 0.0000000000 0.2187958884
    0.0256975037 0.0110132159 0.0198237885 0.1417033774 0.0088105727 0.0022026432 0.0029368576 ,
0.0976505140 0.1027900147 0.1424375918 0.0895741557 0.0712187959 0.0712187959 0.0829662261
    0.0425844347 0.0763582966 0.0198237885 0.0029368576 0.2239353891 0.1152716593 0.1732745962
    0.1938325991 0.1035242291 0.0631424376 0.2466960352 0.2841409692 0.2187958884 0.0000000000
    0.0080763583 0.0000000000 0.0051395007 0.0058737151 0.0029368576 0.0007342144 0.0007342144 ,
0.1108663730 0.0022026432 0.0051395007 0.0198237885 0.2665198238 0.0000000000 0.0029368576
    0.0000000000 0.2136563877 0.0000000000 0.0007342144 0.0293685756 0.0007342144 0.0660792952
    0.0976505140 0.0000000000 0.0000000000 0.0528634361 0.0168869310 0.0256975037 0.0080763583
    0.0000000000 0.0022026432 0.0000000000 0.0088105727 0.0000000000 0.0000000000 0.0000000000 ,
0.0792951542 0.0022026432 0.0007342144 0.0124816446 0.0807635830 0.0036710720 0.0007342144
    0.0367107195 0.0506607930 0.0000000000 0.0044052863 0.0124816446 0.0007342144 0.0242290749
    0.0998531571 0.0022026432 0.0000000000 0.0264317181 0.0220264317 0.0110132159 0.0000000000
    0.0022026432 0.0000000000 0.0000000000 0.0073421439 0.0000000000 0.0007342144 0.0000000000 ,
0.0146842878 0.0007342144 0.0190895742 0.0007342144 0.1013215859 0.0000000000 0.0000000000
    0.0036710720 0.0330396476 0.0000000000 0.0000000000 0.0022026432 0.0029368576 0.0014684288
    0.0117474302 0.0301027900 0.0007342144 0.0051395007 0.0000000000 0.0198237885 0.0051395007
    0.0000000000 0.0000000000 0.0000000000 0.0022026432 0.0000000000 0.0000000000 0.0007342144 ,
0.0778267254 0.0168869310 0.0403817915 0.0359765051 0.0506607930 0.0234948605 0.0168869310
    0.0279001468 0.0227606461 0.0000000000 0.0117474302 0.1424375918 0.0301027900 0.0440528634
    0.0403817915 0.0161527166 0.0000000000 0.1108663730 0.0506607930 0.1417033774 0.0058737151
    0.0088105727 0.0073421439 0.0022026432 0.0000000000 0.0051395007 0.0022026432 0.0007342144 ,
0.0154185022 0.0000000000 0.0007342144 0.0007342144 0.0381791483 0.0000000000 0.0007342144
    0.0000000000 0.0374449339 0.0000000000 0.0000000000 0.0139500734 0.0022026432 0.0058737151
    0.0080763583 0.0000000000 0.0000000000 0.0022026432 0.0000000000 0.0088105727 0.0029368576
    0.0000000000 0.0000000000 0.0000000000 0.0051395007 0.0000000000 0.0000000000 0.0000000000 ,
0.0000000000 0.0000000000 0.0000000000 0.0007342144 0.0007342144 0.0007342144 0.0014684288
    0.0000000000 0.0007342144 0.0000000000 0.0000000000 0.0014684288 0.0000000000 0.0000000000
```

```
    0.0014684288 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0022026432 0.0007342144
    0.0000000000 0.0007342144 0.0000000000 0.0022026432 0.0000000000 0.0000000000 0.0000000000 ,
0.0000000000 0.0007342144 0.0036710720 0.0022026432 0.0022026432 0.0007342144 0.0022026432
    0.0007342144 0.0007342144 0.0000000000 0.0007342144 0.0044052863 0.0000000000 0.0000000000
    0.0014684288 0.0014684288 0.0000000000 0.0044052863 0.0000000000 0.0029368576 0.0007342144
    0.0000000000 0.0000000000 0.0007342144 0.0007342144 0.0000000000 0.0000000000 0.0000000000];

bar3(results)

xlabel('字符序号');ylabel('字符序号');zlabel('频率')

%title('A2')
```

附录 5

本部分代码使用的软件是 C++

```
#pragma GCC optimize(3)
#pragma GCC target("avx,sse2,sse3,sse4,mmx")
#pragma GCC optimize("Ofast")
#pragma GCC optimize("inline")
#include<bits/stdc++.h>
using namespace std;

const int N=1e4+10;
const int M=30;
const int INF=1e6;
const double eps=1e-6;
double A[M][M],B[M][M];
double s[M];
int p[M];
struct link
{
  int top=1,fi[N],to[N],la[N],ne[N],cap[N],cur[N];
  double cost[N];
  void clear()
  {
    top=1;
    memset(fi,0,sizeof fi);
    memset(to,0,sizeof to);
    memset(la,0,sizeof la);
    memset(ne,0,sizeof ne);
    memset(cap,0,sizeof cap);
    memset(cur,0,sizeof cur);
    memset(cost,0,sizeof cost);
  }
  void add(int x,int y,int z,double c)
  {
    top++,to[top]=y;cap[top]=z,cost[top]=c;
    if(fi[x]==0)fi[x]=top;else ne[la[x]]=top;
    la[x]=top;
  }
  void adde(int x,int y,int z,int c)
  {
    add(x,y,z,c);
    add(y,x,0,-c);
  }
}L;
```

```
int S,T;
double TOTcost;
int TOTflow,vis[N];
double dis[N];
int ans[N][M];
queue<int> q;
int SPFA()
{
  for(int i=1;i<=T;i++)dis[i]=INF,vis[i]=0;
  dis[T]=0;
  while(!q.empty())q.pop();
  q.push(T);
  while(!q.empty())
  {
    int now=q.front();
    q.pop();vis[now]=0;
    for(int i=L.fi[now];i;i=L.ne[i])
    if(L.cap[i^1]>0)
    if(dis[L.to[i]]>dis[now]+L.cost[i^1])
    {
      dis[L.to[i]]=dis[now]+L.cost[i^1];
      if(!vis[L.to[i]])
        q.push(L.to[i]),vis[L.to[i]]=1;
    }
  }
  return fabs(dis[S]-INF)>eps;
}
int DFS(int now,int maxflow)
{
  vis[now]=1;
  if(now==T)return maxflow;
  int tot=0;
  for(int i=L.cur[now];i;i=L.ne[i],L.cur[now]=i)
  if(L.cap[i]>0&&fabs(dis[now]-dis[L.to[i]]-L.cost[i])<eps&&!vis[L.to[i]])
  {
    int del=DFS(L.to[i],min(maxflow,L.cap[i]));
    tot+=del;TOTcost+=del*L.cost[i];maxflow-=del;
    L.cap[i]-=del;
    L.cap[i^1]+=del;
  }
  return tot;
}
double calc(int* p)
{
  double re=0;
  for(int i=1;i<=28;i++)
  {
    for(int j=1;j<=28;j++)
      re+=s[i]*s[j]*(A[i][j]-B[p[i]][p[j]])*(A[i][j]-B[p[i]][p[j]]);
  }
  return re;
}
int main()
{
  srand(233);
  freopen("A.txt","r",stdin);
  for(int i=1;i<=28;i++)
  {
    for(int j=1;j<=28;j++)
```

```
101       scanf("%lf",&A[i][j]);
102     }
103     for(int i=1;i<=28;i++)
104     {
105       scanf("%lf",&s[i]);
106       s[i]=1.0;
107       p[i]=i;
108     }
109     freopen("B.txt","r",stdin);
110     for(int i=1;i<=28;i++)
111     {
112       for(int j=1;j<=28;j++)
113       scanf("%lf",&B[i][j]);
114     }
115
116     int cas=0;
117     for(int rou=1;rou<=40;rou++)
118     {
119       random_shuffle(p+1,p+1+28);
120       for(int ti=1;ti<=50;ti++)
121       {
122       L.clear();
123       S=2*28+1,T=S+1;
124       for(int i=1;i<=28;i++)
125       {
126         for(int j=1;j<=28;j++)
127         {
128           double C=0;
129           for(int k=1;k<=28;k++)if(i!=k)C+=s[i]*s[k]*(A[i][k]-B[j][p[k]])*(A[i][k]-B[j][p[k]]);
130           L.adde(i,j+28,1,C);
131         }
132         L.adde(S,i,1,0);
133         L.adde(i+28,T,1,0);
134       }
135       TOTcost=0;
136       TOTflow=0;
137       while(SPFA())
138       {
139         for(int i=1;i<=T;i++)vis[i]=0,L.cur[i]=L.fi[i];
140         TOTflow+=DFS(S,INF);
141       }
142       cas++;
143       for(int i=1;i<=28;i++)
144       {
145         for(int j=L.fi[i];j;j=L.ne[j])
146         if(L.cap[j]==0&&(L.to[j]>28)&&(L.to[j]<=28*2))
147         {
148           p[i]=L.to[j]-28;
149         }
150         ans[cas][i]=p[i];
151       }
152     }
153     }
154
155     double mn=1e9;
156     int ans1;
157     for(int i=1;i<=cas;i++)
158     {
159 //      cout<<calc(ans[i])<<endl;
```

```
    if(calc(ans[i])<mn)
    {
      mn=calc(ans[i]);
      ans1=i;
    }
  }
  freopen("plan.txt","w",stdout);
  char out[4][7];
  for(int i=1;i<=26;i++)
  {
    out[(ans[ans1][i]-1)/7][(ans[ans1][i]-1)%7]='a'+i-1;
  }
  out[(ans[ans1][27]-1)/7][(ans[ans1][27]-1)%7]=' ';
  out[(ans[ans1][28]-1)/7][(ans[ans1][28]-1)%7]='-';
  for(int i=0;i<4;i++)
  {
    for(int j=0;j<7;j++)putchar(out[i][j]);
    puts("");
  }
  return 0;
}
```

附录 6

本部分代码使用的软件是 R

```
library(ggplot2)
library(reshape2)
library(ggsignif)
df = read.delim("https://www.bioladder.cn/shiny/zyp/bioladder2/demoData/BoxPlot/boxplot.txt",header
    = T
)
df = melt(df)

ggplot(df,aes(x=variable,y=value,fill=variable))+
  geom_boxplot(alpha = 1,
               outlier.color = "black"
  )+
  theme_bw()+
  theme(
    axis.text.x = element_text(angle = 90,
                               vjust = 0.5
    )
  )+
  geom_signif(
    comparisons=list(c("Sample1","Sample2"),c("Sample3","Sample4")),
    step_increase = 0.1,
    test="t.test",
    map_signif_level=F
  )
```