# 习题4

4.1

(1) 试检验 Ho:μ=μo=(4,50,10)',H1:μ≠μo

Python代码：

import numpy as np

import pandas as pd

from scipy.stats import f

from scipy.stats import t

# 写入数据

raw\_data = np.array([[3.7,48.5,9.3],[5.7,65.1,8.0],[3.8,47.2,10.9],

[3.2,53.2,12.0],[3.1,55.5,9.7],[4.6,36.1,7.9],

[2.4,24.8,14.0],[7.2,33.1,7.6],[6.7,47.4,8.5],

[5.4,54.1,11.3],[3.9,36.9,12.7],[4.5,58.8,12.3],

[3.5,27.8,9.8],[4.5,40.2,8.4],[1.5,13.5,10.1],

[8.5,56.4,7.1],[4.5,71.6,8.2],[6.5,52.8,10.9],

[4.1,44.1,11.2],[5.5,40.9,9.4]])

data = pd.DataFrame(raw\_data,columns=['x1','x2','x3'],index=np.arange(20)+1)

miu\_0 = pd.Series([4,50,10],index=['x1','x2','x3'])

x\_ba = data.mean() #计算样本均值

s = data.cov() #计算样本协方差矩阵

n = len(data['x1']) #计算样本量

p = len(data.columns) #计算数据维度

#计算霍特林统计量

T\_2 = n \* np.array((x\_ba-miu\_0)).T @ np.linalg.inv(s) @ np.array((x\_ba-miu\_0))

f\_value = (n-p)/(p\*(n-1)) \* T\_2 #计算相应的F分布的分位数

p\_value = f.sf(f\_value,p,n-p) #计算相对应的P值

print('计算出来的p值：P-value=',p\_value)



程序运行结果如图所示，计算出来的p值大于0.05，故我们暂时不能拒绝原假设，暂时认为μ=μo。

(2) 试求μ的0.95置信区间

Python代码：

# 计算在置信度0.05，自由度为(p,n-1)霍特林统计量的值

T2 = (p\*(n-1)) / (n-p) \* f.ppf(0.95,p,n-p)

T2



程序运行结果如图所示，因此μ在0.05的置信度条件下的置信椭球为：

n \* np.array((x\_ba-miu\_0)).T @ np.linalg.inv(s) @ np.array((x\_ba-miu\_0)) <= T2。

(3) 试求μ1、μ2、μ3的0.95联合T2置信区间和0.95邦弗伦尼联合置信区间，并对这两种区间进行比较。

Python代码：

# 联合T2置信区间

μ1\_up = miu\_0[0] + np.sqrt(T2) \* np.sqrt(s.iloc[0][0]/n)

μ1\_down = miu\_0[0] - np.sqrt(T2) \* np.sqrt(s.iloc[0][0]/n)

μ1 = (μ1\_down,μ1\_up)

μ2\_up = miu\_0[1] + np.sqrt(T2) \* np.sqrt(s.iloc[1][1]/n)

μ2\_down = miu\_0[1] - np.sqrt(T2) \* np.sqrt(s.iloc[1][1]/n)

μ2 = (μ2\_down,μ2\_up)

μ3\_up = miu\_0[2] + np.sqrt(T2) \* np.sqrt(s.iloc[2][2]/n)

μ3\_down = miu\_0[2] - np.sqrt(T2) \* np.sqrt(s.iloc[2][2]/n)

μ3 = (μ3\_down,μ3\_up)

print("μ1联合T2置信区间:",μ1)

print("μ2联合T2置信区间:",μ2)

print("μ3联合T2置信区间:",μ3)

# 邦弗伦尼联合置信区间

#先计算t分位数

t\_value = t.ppf(1-0.05/(2\*p),n-1)

#计算联合置信区间

μ1\_up = miu\_0[0] + t\_value \* np.sqrt(s.iloc[0][0]/n)

μ1\_down = miu\_0[0] - t\_value \* np.sqrt(s.iloc[0][0]/n)

μ1 = (μ1\_down,μ1\_up)

μ2\_up = miu\_0[1] + t\_value \* np.sqrt(s.iloc[1][1]/n)

μ2\_down = miu\_0[1] - t\_value \* np.sqrt(s.iloc[1][1]/n)

μ2 = (μ2\_down,μ2\_up)

μ3\_up = miu\_0[2] + t\_value \* np.sqrt(s.iloc[2][2]/n)

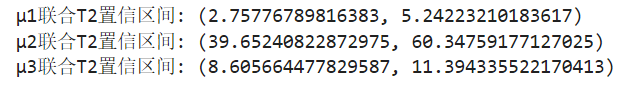
μ3\_down = miu\_0[2] - t\_value \* np.sqrt(s.iloc[2][2]/n)

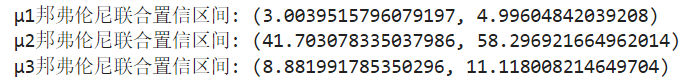
μ3 = (μ3\_down,μ3\_up)

print("μ1邦弗伦尼联合置信区间:",μ1)

print("μ2邦弗伦尼联合置信区间:",μ2)

print("μ3邦弗伦尼联合置信区间:",μ3)T2





程序运行结果如图所示，综上可知，邦弗伦尼联合置信区间更小。

4.2

R代码：

y <- read.csv('data.csv') # 读取数据

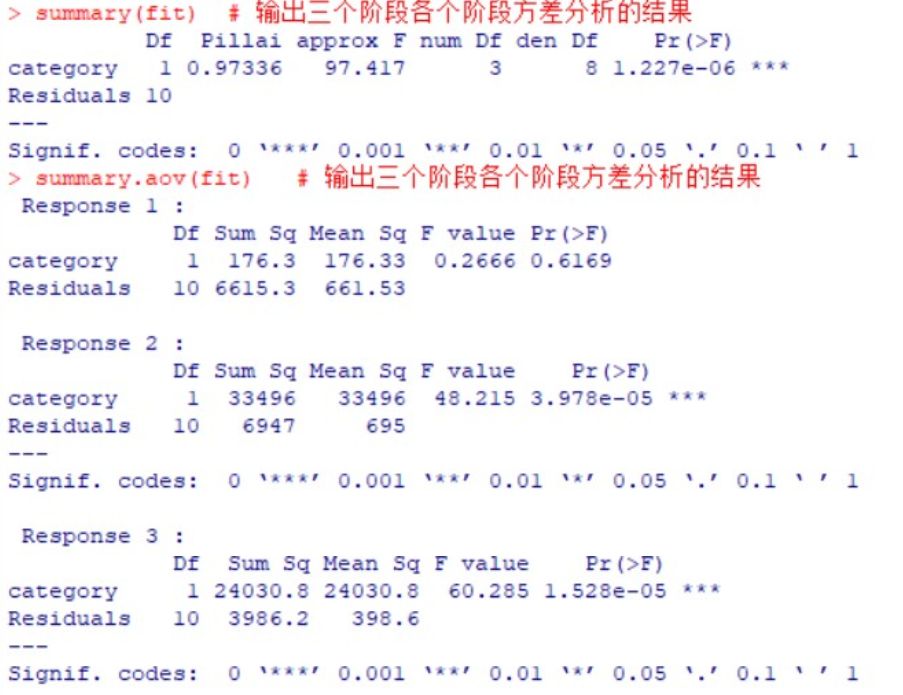
category = y$catagory # 将解释变量归为一类

y = cbind(y$x1,y$x2,y$x3) # 将响应变量归为一类

fit <- manova(y~category) # 多元方差分析模型

summary(fit) # 输出三个阶段方差分析的结果

summary.aov(fit) # 输出三个阶段各个阶段方差分析的结果



程序运行结果如图所示,三个阶段总的p值小于0.05，认为甲乙两种轮胎耐用性显著不同。分析三个阶段各个阶段的检验可以得出结论，第二和第三阶段对拒绝原假设起到了很大的作用。

4.7

Python代码：

import numpy as np

import pandas as pd

import pylab as plt

#写入甲的数据

jia = np.array([[11,18,15,18,15],[33,27,31,21,17],

[20,28,27,23,19],[18,26,18,18,9],

[22,23,22,16,10],[20.8,24.4,22.6,19.2,14.0]])

jia\_data = pd.DataFrame(jia,columns=['I1','I2','I3','I4','I5'],

index=['1','2','3','4','5','Mean'])

#写入乙的数据

yi = np.array([[18,17,20,18,18],[31,24,31,26,20],

[14,16,17,20,17],[25,24,31,26,28],

[36,28,24,26,29],[24.8,21.8,24.6,23.2,20.4]])

yi\_data = pd.DataFrame(yi,columns=['I1','I2','I3','I4','I5'],

index=['1','2','3','4','5','Mean'])

# 画出轮廓分析图

plt.figure()

plt.plot(jia\_data.ix['Mean'],marker='x',color='b',label='A')

plt.plot(yi\_data.ix['Mean'],marker='o',color='r',label='B')

plt.ylabel('Sample Mean')

plt.xlabel('Variable')

plt.title('A and B sample plots')

plt.legend(loc='best')

plt.show()

# 设置相应参数

n1 = 5

n2 = 5

k = 4

C = np.array([[1,-1,0,0,0],[0,1,-1,0,0],

[0,0,1,-1,0],[0,0,0,1,-1]])

C = pd.DataFrame(C)

#计算样本均值之差

M = jia\_data.ix['Mean'] - yi\_data.ix['Mean']

# 计算联合协方差

S1 = jia\_data.iloc[0:5,:].cov()

S2 = yi\_data.iloc[0:5,:].cov()

Sp = ((n1-1)\*S1+(n2-1)\*S2) / (n1+n2-2)

# 计算C(X\_ba\_jia - X\_ba\_yi)

A = np.array(C) @ np.array(M)

# 计算CSpC'

B = np.array(C) @ np.array(Sp) @ np.array(C).T

# 对上一步求逆

D = np.linalg.inv(B)

#计算霍特林统计量

T2 = ((n1\*n2)/(n1+n2)) \* np.array(M).T @ np.array(C).T @ D @ A

# 计算相应的F统计量

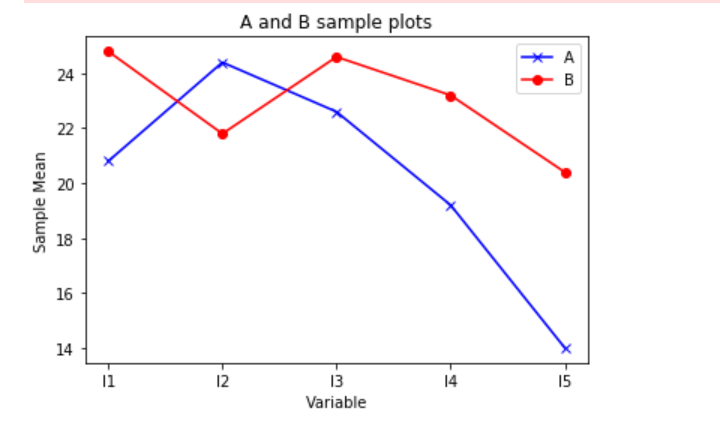
F = (n1+n2-k-1)/(k\*(n1+n2-2)) \* T2

#计算相应的p值

from scipy.stats import f

p\_value = f.sf(F,k,n1)

p\_value





程序运行结果如图所示,根据其计算出来的P值0.04683小于显著性水平0.05，拒绝原假设，认为甲、乙两种品牌的产品每个指标间的差异有显著不同。

4.10

R代码：

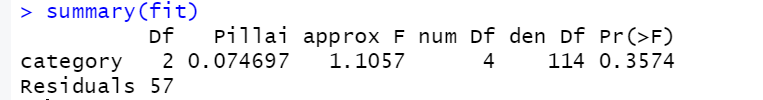
data = read.csv("4.10.csv") # 读取数据

category = data$catagory # 将解释变量归为一类

y = cbind(data$L,data$R) # 将响应变量归为一类

fit <- manova(y~category) # 多元方差分析模型

summary(fit) # 输出方差分析的结果



程序运行结果如图所示,假设检验的p值为0.3574大于给定的显著性水平，暂时不能拒绝原假设，也就是暂时认为三部分犯人耳朵长度无明显差异。