In [1]:
```
import numpy as np
import pandas as pd
import scipy.stats as stats
from matplotlib import pyplot as plt
import seaborn as sns
```

6.19

Trong giai đoạn đầu của một nghiên cứu về chi phí vận chuyển sữa từ trang trại đến nhà máy sữa, một cuộc khảo sát được thực hiện đối với các công ty tham gia vận chuyển. Dữ liệu chi phí trên $X_1 = fuel$, $X_2 = repair$ và $X_3 = capital$, các phép đo trên cơ sở mỗi dặm, được trình bày trong bảng 6.10 trang 345 cho $n_1 = 36$ xe chạy bằng xăng và $n_2 = 23$ xe chạy bằng dầu diesel.

In [2]:
```
path_19 = 'T6-10.txt'
data_19 = pd.read_table(path_19, delim_whitespace=True, header=None)
data_19.columns=['x1(fuel)', 'x2(repair)', 'x3(capital)', 'type_truck']

gasoline_df = data_19[data_19['type_truck']=='gasoline'].iloc[:,:-1]
diesel_df = data_19[data_19['type_truck']=='diesel'].iloc[:,:-1]

print('>> Gasoline trunks data: \n', gasoline_df.head(5))
print('\n>> Diesel trunks data: \n', diesel_df.head(5))

n1, p = gasoline_df.shape
n2 = diesel_df.shape[0]

print('\n>> Number of samples: n1 = {} and n2 = {}'.format(n1,n2))
```

```
>> Gasoline trunks data:
    x1(fuel)  x2(repair)  x3(capital)
0     16.44       12.43        11.23
1      7.19        2.70         3.92
2      9.92        1.35         9.75
3      4.24        5.78         7.78
4     11.20        5.05        10.67

>> Diesel trunks data:
     x1(fuel)  x2(repair)  x3(capital)
36      8.50       12.26         9.11
37      7.42        5.13        17.15
38     10.28        3.32        11.23
39     10.16       14.72         5.99
40     12.79        4.17        29.28

>> Number of samples: n1 = 36 and n2 = 23
```

(a) Kiểm định sự chênh lệch giữa các vectơ chi phí trung bình với mức ý nghĩa $\alpha = 0.01$.

In [3]:
```
x1_mean = gasoline_df.mean(axis=0)
S1 = np.cov(gasoline_df.T)

x2_mean = diesel_df.mean(axis=0)
S2 = np.cov(diesel_df.T)

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

print('>> Summary statistics: \n')
print('* x1_mean = \n', np.array(x1_mean).reshape(-1,1))
print('\n* x2_mean = \n', np.array(x2_mean).reshape(-1,1))
print('\n* S1 = \n', S1)
print('\n* S2 = \n', S2)
print('\n* S_pooled = \n', S_pooled)
```

```
>> Summary statistics:

* x1_mean =
 [[12.21861111]
 [ 8.1125    ]
 [ 9.59027778]]

* x2_mean =
 [[10.10565217]
 [10.76217391]
 [18.16782609]]

* S1 =
 [[23.01336087 12.366395    2.90660897]
 [12.366395   17.54411071  4.77308214]
 [ 2.90660897  4.77308214 13.96333421]]

* S2 =
 [[ 4.3623166   0.75988715  2.36209921]
 [ 0.75988715 25.85123597  7.68573221]
 [ 2.36209921  7.68573221 46.6543996 ]]

* S_pooled =
 [[15.81471221  7.88669022  2.69644731]
 [ 7.88669022 20.75036958  5.89726287]
 [ 2.69644731  5.89726287 26.5809384 ]]
```

Phát biểu giả thuyết:

$$H_0 : \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2 = 0 \quad \text{và} \quad H_1 : \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2 \neq 0$$

Giả sử H_0 đúng, ta có thống kê

$$T^2 = (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)' \left[\left(\frac{1}{n_1} + \frac{1}{n_2} \right) \mathbf{S}_{\text{pooled}} \right]^{-1} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2) \sim \frac{(n_1 + n_2 - 2)p}{n_1 + n_2 - p - 1} F_{p, n_1 + n_2 - p - 1}$$

Với mẫu thực nghiệm, ta có giá trị thống kê như sau

In [4]:
```
T2 = np.transpose(x1_mean - x2_mean).dot(np.linalg.inv((1/n1+1/n2)*S_pooled).dot(x1_mean - x2_mean))
print('>> Giá trị thống kê: ',T2)
```

```
>> Giá trị thống kê:  50.91278683116499
```

Với mức ý nghĩa $\alpha = 0.01$, ta có

In [5]:
```
alpha = 0.01

f = stats.f.ppf(q=1-alpha, dfn=p, dfd=n1+n2-p-1)
F = (((n1+n2-2)*p)/(n1+n2-p-1))*f

F
```

Out[5]:
```
12.9309599794576
```

Vì 50.91278683116499 > 12.9309599794576 nên ta bác bỏ giả thuyết H_0 với mức ý nghĩa $\alpha = 0.01$. Do đó với mức ý nghĩa 0.01, có sự chênh lệch giữa các vectơ chi phí trung bình giữa các xe chạy bằng gasoline và xe chạy bằng dầu diesel.

(b) Nếu giả thuyết các vectơ chi phí bằng nhau bị bác bỏ ở câu (a), tìm tổ hợp tuyến tính của các thành phần trung bình ảnh hưởng lớn nhất đến việc bác bỏ.

Tổ hợp tuyến tính quan trọng nhất của các thành phần trung bình dẫn đến việc bác bỏ giả thuyết H_0 có vectơ hệ số như sau

$$\hat{\mathbf{a}} \propto \mathbf{S}_{\text{pooled}}^{-1} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)$$

In [6]:
```
a_hat = np.linalg.inv(S_pooled).dot(x1_mean - x2_mean)
np.array(a_hat).reshape(-1,1)
```

Out[6]:
```
array([[ 0.25474521],
       [-0.13390356],
       [-0.31882963]])
```

Ta có thể thấy rằng, sự chênh lệch giữa việc tiêu thụ nhiên liệu giữa những xe chạy bằng xăng và xe chạy bằng dầu diesel có ảnh hưởng mạnh nhất đến việc bác bỏ giả thuyết H_0.

(c) Xây dựng các khoảng tin cậy đồng thời 99% cho các cặp thành phần trung bình. Chi phí (nếu có) nào có sự chênh lệch?

In [7]:
```
lcb = lambda i: (x1_mean[i] - x2_mean[i]) - np.sqrt(F)*np.sqrt((1/n1+1/n2)*S_pooled[i][i])
ucb = lambda i: (x1_mean[i] - x2_mean[i]) + np.sqrt(F)*np.sqrt((1/n1+1/n2)*S_pooled[i][i])

# 95% simultaneous confidence intervals for individual means
print("99% simultaneous confidence intervals for mean components are: \n")
IC = []
for i in range(p):
    ic = [lcb(i), ucb(i)]
    print(">> {}:\t{}\n".format(data_19.columns[i], ic))
    IC.append(ic)
```

```
99% simultaneous confidence intervals for mean components are:

>> x1(fuel):    [-1.704346120977224, 5.9302639953733625]

>> x2(repair):  [-7.022267760770974, 1.7229199346840192]

>> x3(capital): [-13.526479046383113, -3.6286175719743694]
```

Với mức ý nghĩa 0.01, ta thấy rằng khoảng tin cậy đồng thời 99% cho thành phần trung bình thứ 3 (capital) không chứa 0, do đó chi phí cho phần vốn bỏ ra có sự chênh lệch giữa xe chạy bằng xăng và xe chạy bằng dầu diesel.

(d) Nhận xét về tính hợp lý của các giả thuyết được sử dụng trong bài phân tích của bạn. Lưu ý rằng, các quan trắc 9 và 21 đối với xe chạy bằng xăng được cho là ngoại lai. Lặp lại câu (a) với việc xoá đi các quan trắc này. Nhận xét kết quả.

In [8]:
```
gasoline_df_without_outliers = gasoline_df.drop([9,21], axis=0)
n1 = gasoline_df_without_outliers.shape[0]
print('>> Number of samples for gasoline after remove outliers: ', n1)
```

```
>> Number of samples for gasoline after remove outliers:  34
```

In [9]:
```
x1_mean = gasoline_df.mean(axis=0)
S1 = np.cov(gasoline_df.T)

print('>> Summary statistics after romove outliers: \n')
print('* x1_mean = \n', np.array(x1_mean).reshape(-1,1))
print('\n* x2_mean = \n', np.array(x2_mean).reshape(-1,1))
print('\n* S1 = \n', S1)
print('\n* S2 = \n', S2)
```

```
>> Summary statistics after romove outliers:

* x1_mean =
 [[12.21861111]
 [ 8.1125    ]
 [ 9.59027778]]

* x2_mean =
 [[10.10565217]
 [10.76217391]
 [18.16782609]]

* S1 =
 [[23.01336087 12.366395    2.90660897]
 [12.366395   17.54411071  4.77308214]
 [ 2.90660897  4.77308214 13.96333421]]

* S2 =
 [[ 4.3623166   0.75988715  2.36209921]
 [ 0.75988715 25.85123597  7.68573221]
 [ 2.36209921  7.68573221 46.6543996 ]]
```

Phát biểu giả thuyết:

$$H_0 : \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2 = 0 \quad \text{và} \quad H_1 : \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2 \neq 0$$

Giả sử $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2$ và giả thuyết H_0 đúng. Vì $\mathbf{S}_1$ và $\mathbf{S}_2$ khá khác nhau nên việc gộp chúng lại không hợp lý. Tuy nhiên, bằng cách sử dụng lý thuyết cỡ mẫu lớn $(n_1 = 34, n_2 = 23)$ và theo kết quả 6.4 ta có thống kê

$$(\overline{\mathbf{x}}_1 - \overline{\mathbf{x}}_2)' \left(\frac{1}{n_1}\mathbf{S}_1 + \frac{1}{n_2}\mathbf{S}_2 \right)^{-1} (\overline{\mathbf{x}}_1 - \overline{\mathbf{x}}_2) \sim \chi^2_p$$

In [10]:
```
T2 = np.transpose(x1_mean - x2_mean).dot(np.linalg.inv(1/n1*S1+1/n2*S2).dot(x1_mean - x2_mean))
print('>> Giá trị thống kê: ',T2)
```

```
>> Giá trị thống kê:  42.524812258635734
```

In [11]:
```
alpha = 0.01

chisq = stats.chi2.ppf(1-alpha, p)

chisq
```

Out[11]:
```
11.344866730144373
```

Vì 42.524812258635734 > 11.344866730144373 nên ta bác bỏ giả thuyết H_0 với mức ý nghĩa $\alpha = 0.01$. Do đó với mức ý nghĩa 0.01, có sự chênh lệch giữa các vectơ chi phí trung bình giữa các xe chạy bằng gasoline và xe chạy bằng dầu diesel.

Như vậy, kết quả trên tương đồng với kết quả ở câu (a).

6.22

Các nhà nghiên cứu thích thú việc đánh giá chức năng của phổi trong một tổng thể những người không có bệnh lý được yêu cầu chạy trên một máy chạy bộ tới khi kiệt sức. Những mẫu không khí được sưu tập ở một khoảng xác định và thành phần thể khí được phân tích. Kết quả trên 4 phép đo lượng tiêu thụ oxy của 25 nam và 25 nữ được cho bởi bảng 6.12 ở trang 348.

In [12]:
```
#Import dataset
path = 'T6-12.txt'
f = open(path, "r")
data=f.readlines()
male_data=list()
female_data=list()
for txt in data:
    if 'female' in txt:
        female_data.append(txt[:-7])
    else:
        male_data.append(txt[:-6])

male_df=pd.DataFrame(np.loadtxt(male_data))
female_df=pd.DataFrame(np.loadtxt(female_data))
```

(a) Tìm sự chênh lệch về giới tính bằng cách kiểm định tính bằng nhau của trung bình từng group. Sử dụng $\alpha = .05$. Nếu bác bỏ H_0 : $\boldsymbol{\mu_1 - \mu_2 = 0}$, hãy tìm tổ hợp tuyến tính thoả mãn.

In [13]:
```
# vecto trung bình
male_mean = np.array(male_df.apply(np.mean))
print("\nMean vector Male: \n", male_mean)

# Ma trận hiệp phương sai
S_male = np.array(np.cov(male_df.T))
print("\nCovariance matrix Male: \n", S_male)

# vecto trung bình
female_mean = np.array(female_df.apply(np.mean))
print("\nMean vector female: \n", female_mean)

# Ma trận hiệp phương sai
S_female = np.array(np.cov(female_df.T))
print("\nCovariance matrix female: \n", S_female)
```

```
Mean vector Male:
 [ 0.3972  5.3296  3.6876 49.4204]

Covariance matrix Male:
 [[7.12100000e-03 7.00030000e-02 3.14471667e-02 1.50580333e-01]
 [7.00030000e-02 1.14417900e+00 1.47678167e-01 3.43090850e+00]
 [3.14471667e-02 1.47678167e-01 4.55877333e-01 3.30812183e+00]
 [1.50580333e-01 3.43090850e+00 3.30812183e+00 5.52521457e+01]]

Mean vector female:
 [ 0.3136  5.1788  2.3152 38.1548]

Covariance matrix female:
 [[ 9.73233333e-03  1.54087833e-01  4.16800000e-03  2.97570000e-02]
 [ 1.54087833e-01  2.78066100e+00 -3.94476667e-02  1.28069767e+00]
 [ 4.16800000e-03 -3.94476667e-02  1.20509333e-01  1.09814900e+00]
 [ 2.97570000e-02  1.28069767e+00  1.09814900e+00  2.32608260e+01]]
```

In [14]:
```
n_1=n_2=male_df.shape[0]
S_pooled=(n_1-1)/(n_1+n_2-2)*S_male+(n_2-1)/(n_1+n_2-2)*S_female
print(S_pooled)
```

```
[[8.42666667e-03 1.12045417e-01 1.78075833e-02 9.01686667e-02]
 [1.12045417e-01 1.96242000e+00 5.41152500e-02 2.35580308e+00]
 [1.78075833e-02 5.41152500e-02 2.88193333e-01 2.20313542e+00]
 [9.01686667e-02 2.35580308e+00 2.20313542e+00 3.92564858e+01]]
```

In [15]:
```
t_2=np.matmul((male_mean-female_mean).T,np.linalg.inv((1/n_1+1/n_2)*S_pooled))
t_2=np.matmul(t_2,(male_mean-female_mean))
print('Hotelling T^2: ',t_2)
```

```
Hotelling T^2:  96.37322129760133
```

In [16]:
```
p=male_df.shape[1]
f = stats.f.ppf(q=1-0.05, dfn=p, dfd=n_1+n_2-p-1)
c_2=(n_1+n_2-2)*p/(n_1+n_2-p-1)*f
print('Critical Value: ',c_2)
```

```
Critical Value:  11.002620519729316
```

Với Hotelling $T^2 = 96.373 > 11$, ta bác bỏ H_0 với mức ý nghĩa $\alpha = 0.05$. Ta kết luận là với mức ý nghĩa 95% có sự chênh lệch trong chức năng của phổi giữa 2 giới tính nam và nữ.

In [17]:
```
linear_combination=np.matmul(np.linalg.inv(S_pooled),(male_mean-female_mean))
print('linear combination most reponsible: ',linear_combination)
```

```
linear combination most reponsible:  [99.39897799 -6.3759988  -6.2281408   0.79082376]
```

(b) Xây dựng khoảng tin cậy đồng thời 95% cho mỗi $\mu_{1i} - \mu_{2i}, i = 1, 2, 3, 4$. So sánh với khoảng tin cậy Bonferroni

In [18]:
```
for i in range(male_df.shape[1]):
    lowerbound=(male_mean[i]-female_mean[i])-np.sqrt(c_2)*np.sqrt((1/n_1+1/n_2)*S_pooled[i][i])
    upperbound=(male_mean[i]-female_mean[i])+np.sqrt(c_2)*np.sqrt((1/n_1+1/n_2)*S_pooled[i][i])
    print('95% confidence interval for mu_1{}-mu_2{} in range [{}, {}]'.format(i+1,i+1,lowerbound,upperbound))
```

```
95% confidence interval for mu_11-mu_21 in range [-0.0025233606309393586, 0.1697233606309396]
95% confidence interval for mu_12-mu_22 in range [-1.1634834568030512, 1.46508345680305]
95% confidence interval for mu_13-mu_23 in range [0.8687428242955314, 1.8760571757044682]
95% confidence interval for mu_14-mu_24 in range [5.387340280917852, 17.143859719082148]
```

In [19]:
```
t = stats.t.ppf(q=1-(0.05/(2*p)),df=n_1+n_2-2)
for i in range(male_df.shape[1]):
    lowerbound=(male_mean[i]-female_mean[i])-t*np.sqrt((1/n_1+1/n_2)*S_pooled[i][i])
    upperbound=(male_mean[i]-female_mean[i])+t*np.sqrt((1/n_1+1/n_2)*S_pooled[i][i])
    print('Bonferroni confidence interval for mu_1{}-mu_2{} in range [{},{}]'.format(i+1,i+1,lowerbound,upperbound))
```

```
Bonferroni confidence interval for mu_11-mu_21 in range [0.016214838949563673,0.15098516105043658]
Bonferroni confidence interval for mu_12-mu_22 in range [-0.8775296164221205,1.1791296164221192]
Bonferroni confidence interval for mu_13-mu_23 in range [0.978325504411156,1.7664744955888438]
Bonferroni confidence interval for mu_14-mu_24 in range [6.666296454890269,15.86490354510973]
```

Từ các khoảng tin cậy trên, ta thấy khoảng tin cậy 95% Bonferroni nhỏ hơn so với khoảng tin cậy còn lại.

(c) Với bộ dữ liệu bảng 6.12 được thu thập từ những tình nguyện viên đại học, do đó, họ không đại diện cho một mẫu ngẫu nhiên. Suy ra điều gì từ thông tin này.

Do dữ liệu bảng 6.12 chỉ được thu thập từ một lượng tình nguyện viên đại học nhỏ và cũng không thể hiện mẫu ngẫu nhiên nên ta không thể mở rộng các kết quả trên cho tổng thể lớn hơn là tổng thể những người đang ở độ tuổi sinh viên đại học.

6.23

Xây dựng one-way MANOVA bằng cách sử dụng các phép đo chiều rộng từ bộ dữ liệu iris trong bảng 11.5. Xây dựng các khoảng tin cậy đồng thời 95% cho các chênh lệch giữa các thành phần của vectơ trung bình đối với 2 phản hồi cho mỗi cặp tổng thể. Nhận xét về tính hợp lý cho giả thuyết $\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \mathbf{\Sigma}_3$.

In [20]:

```
path_23 = 'T11-5.txt'
data_23 = pd.read_table(path_23, delim_whitespace=True, header=None)
data_23.columns=['Sepal_Length', 'Sepal_Width', 'Petal_Length', 'Petal_Width', 'Species']
data_23 = data_23.drop(['Sepal_Length', 'Petal_Length'], axis=1)
print(data_23)
setosa_df = data_23[data_23['Species']==1].iloc[:,:-1]
versicolor_df = data_23[data_23['Species']==2].iloc[:,:-1]
virginica_df = data_23[data_23['Species']==3].iloc[:,:-1]

print('>> Iris setosa data: \n', setosa_df.head(5))
print('\n>> Iris versicolor data: \n', versicolor_df.head(5))
print('\n>> Iris virginica data: \n', virginica_df.head(5))

g = len(np.unique(data_23['Species']))
n1, p = setosa_df.shape
n2, n3 = versicolor_df.shape[0], virginica_df.shape[0]

print('\n>> Number of samples: n1 = {}, n2 = {} and n3 = {}'.format(n1,n2,n3))
print('\n>> Number of dimentions: p = {}'.format(p))
print('\n>> Number of populations: g = {}'.format(g))
```

```
     Sepal_Width  Petal_Width  Species
0            3.5          0.2        1
1            3.0          0.2        1
2            3.2          0.2        1
3            3.1          0.2        1
4            3.6          0.2        1
..           ...          ...      ...
145          3.0          2.3        3
146          2.5          1.9        3
147          3.0          2.0        3
148          3.4          2.3        3
149          3.0          1.8        3

[150 rows x 3 columns]
>> Iris setosa data:
    Sepal_Width  Petal_Width
0          3.5          0.2
1          3.0          0.2
2          3.2          0.2
3          3.1          0.2
4          3.6          0.2

>> Iris versicolor data:
     Sepal_Width  Petal_Width
50          3.2          1.4
51          3.2          1.5
52          3.1          1.5
53          2.3          1.3
54          2.8          1.5

>> Iris virginica data:
      Sepal_Width  Petal_Width
100          3.3          2.5
101          2.7          1.9
102          3.0          2.1
103          2.9          1.8
104          3.0          2.2

>> Number of samples: n1 = 50, n2 = 50 and n3 = 50

>> Number of dimentions: p = 2

>> Number of populations: g = 3
```

In [21]:

```
class OneWayMANOVA:
    def __init__(self):
        self.list_n = []
        self.n = None
        self.g = None
        self.p = None
        self.mean = None
        self.list_means = []
        self.list_cov = []
        self.B = None
        self.W = None
        self.B_Plus_W = None

    def _calc_init_values(self, X, y):
        self.g = len(np.unique(y))
        self.p = X.shape[1]
        self.mean = np.array(np.mean(X,axis=0)).reshape(-1,1)
        for i in np.unique(y):
            data = X[y==i]
            self.list_n.append(data.shape[0])
            self.list_means.append(np.array(np.mean(data, axis=0)).reshape(-1,1))
            self.list_cov.append(np.array(np.cov(data.T)))
        self.n = np.sum(self.list_n)

    def fit(self, X, y):
        self._calc_init_values(X, y)

        B = np.zeros((self.p, self.p))
        W = np.zeros((self.p, self.p))
        for i in range(self.g):

            B += self.list_n[i]*(self.list_means[i] - self.mean).dot((self.list_means[i] - self.mean).T)
            data = X[y==np.unique(y)[i]].to_numpy()
            W += (self.list_n[i]-1)*(self.list_cov[i])

        self.B = (B, self.g-1)
        self.W = (W, self.n-self.g)
        self.B_Plus_W = (B+W, self.n-1)

    def table(self):
        print('>> MANOVA Table: \n')
        print('*'*50)
        print('> Treatment: \nB = \n{},\t d.f. = {}'.format(self.B[0], self.B[1]))
        print('\n> Residual: \nW = \n{},\t d.f. = {}'.format(self.W[0], self.W[1]))
        print('\n> Total: \nB+W = \n{},\t d.f. = {}'.format(self.B_Plus_W[0], self.B_Plus_W[1]))
        print('*'*50)
```

```
    def simultaneous_confidence_intervals(self, alpha):
        import scipy.stats as stats
        t = stats.t.ppf(q=1 - (alpha/(self.p*self.g*(self.g-1))), df=np.sum(self.n - self.g))
        lcb = lambda i, k, l: (self.list_means[k][i]-self.list_means[l][i]) - t*np.sqrt(self.W[0][i][i]/(self.n-self.g)
*(1/self.list_n[k]+1/self.list_n[l]))
        ucb = lambda i, k, l: (self.list_means[k][i]-self.list_means[l][i]) + t*np.sqrt(self.W[0][i][i]/(self.n-self.g)
*(1/self.list_n[k]+1/self.list_n[l]))

        for i in range(self.p):
            for k in range(self.g):
                for l in range(0, k):
                    print('> 95% Confidence interval of tau_{}{} - tau_{}{}: \n[{}, {}]\n'.format(k,i,l,i,lcb(i,k,l),uc
b(i,k,l)))
```

In [22]:

```
X = data_23.drop(['Species'], axis=1)
y = data_23['Species']
maov = OneWayMANOVA()
maov.fit(X, y)
maov.table()
```

```
>> MANOVA Table:

************************************************
> Treatment:
B =
[[ 11.34493333 -22.93266667]
 [-22.93266667  80.41333333]],   d.f. = 2

> Residual:
W =
[[16.962   4.8084]
 [ 4.8084  6.1566]],     d.f. = 147

> Total:
B+W =
[[ 28.30693333 -18.12426667]
 [-18.12426667  86.56993333]],   d.f. = 149
************************************************
```

In [23]:
```
alpha = 0.05
maov.simultaneous_confidence_intervals(alpha)
```

```
> 95% Confidence interval of tau_10 - tau_00:
[[-0.83969436], [-0.47630564]]

> 95% Confidence interval of tau_20 - tau_00:
[[-0.63569436], [-0.27230564]]

> 95% Confidence interval of tau_20 - tau_10:
[[0.02230564], [0.38569436]]

> 95% Confidence interval of tau_11 - tau_01:
[[0.97053548], [1.18946452]]

> 95% Confidence interval of tau_21 - tau_01:
[[1.67053548], [1.88946452]]

> 95% Confidence interval of tau_21 - tau_11:
[[0.59053548], [0.80946452]]
```

Phát biểu giả thuyết

$$\begin{cases} H_0 : \tau_1 = \tau_2 = \tau_3 = 0 \\ H_1 : \exists i \in \{1, 2, 3\} : \tau_i \neq 0 \end{cases}$$

Giả sử giả thuyết H_0 đúng, ta sử dụng thống kê Wilk's lambda để kiểm định giả thuyết trên. Khi đó ta có giá trị thống kê

$$\Lambda^* = \frac{|\mathbf{W}|}{|\mathbf{B} + \mathbf{W}|}$$

In [24]:
```
Lambda = np.linalg.det(maov.W[0])/np.linalg.det(maov.B_Plus_W[0])
Lambda
```

Out[24]:
```
0.03831573747619264
```

Ta có giá trị thống kê của Λ^* là 0.03831573747619264.

Do $p = 2$ và $g = 3 \geq 2$ nên ta xét thống kê

$$\left(\frac{n - g - 1}{g - 1}\right)\left(\frac{1 - \sqrt{\Lambda^*}}{\sqrt{\Lambda^*}}\right) \sim F_{2(g-1),2(n-g-1)}$$

In [25]:
```
value = ((maov.n - maov.g - 1)/(maov.g - 1))*((1 - np.sqrt(Lambda))/np.sqrt(Lambda))
value
```

Out[25]:
```
299.9359632698892
```

Với mức ý nghĩa $\alpha = 0.05$ ta có

In [26]:
```
f = stats.f.ppf(q=1-alpha,dfn=2*(maov.g-1),dfd=2*(maov.n-maov.g-1))
f
```

Out[26]:
```
2.40256219045279
```

Vì $299.9359632698892 > 2.40256219045279$ nên ta bác bỏ H_0 với mức ý nghĩa 0.05. Điều đó cho thấy rằng có sự chênh lệch độ rộng ở 3 loài hoa.

6.24

Các nhà nghiên cứu đã nhận định rằng một thay đổi trong kích thước hộp sọ theo thời gian là bằng chứng cho một giao phối giữa tổng thể bản địa và tổng thể di cư. 4 phép đo được tạo bởi hộp sọ đàn ông Ai Cập trong 3 thời kì khác nhau: thời kì 1 là 4000 năm trước công nguyên, thời kì 2 là 3300 năm trước công nguyên và thời kì 3 là 1850 năm trước công nguyên. Bộ dữ liệu được thể hiện ở bảng 6.13 trang 349. Hãy xây dựng MANOVA 1 chiều của bộ dữ liệu hộp sọ người Ai Cập. Sử dụng $\alpha = .05$. Xây dựng khoảng tin cậy đồng thời 95% để xác định thành phần trung bình nào khác nhau trong các tổng thể được thể hiện bởi 3 thời kì. Giả định MANOVA thông thường có thực tế với những dữ liệu này không ? Giải thích

In [27]:
```
#Import dữ liệu
path = 'T6-13.txt'
data = pd.DataFrame(np.loadtxt(path))
print(data.head(11))
```

```
        0      1      2     3    4
0   131.0  138.0   89.0  49.0  1.0
1   125.0  131.0   92.0  48.0  1.0
2   131.0  132.0   99.0  50.0  1.0
3   119.0  132.0   96.0  44.0  1.0
4   136.0  143.0  100.0  54.0  1.0
5   138.0  137.0   89.0  56.0  1.0
6   139.0  130.0  108.0  48.0  1.0
7   125.0  136.0   93.0  48.0  1.0
8   131.0  134.0  102.0  51.0  1.0
9   134.0  134.0   99.0  51.0  1.0
10  129.0  138.0   95.0  50.0  1.0
```

In [28]:

```
list_mean=list()
list_data=list()
list_cov=list()

#Trung bình và ma trận hiệp phương sai của hộp sọ trong thời kì 1
data_1=data[data.iloc[:,-1]==1]
data_1=data_1.drop([4],axis=1)
list_data.append(data_1)
list_cov.append(np.cov(data_1.T))

x_1_mean=np.mean(data_1,axis=0)
list_mean.append(x_1_mean.to_numpy().reshape(-1,1))

#Trung bình và ma trận hiệp phương sai của hộp sọ trong thời kì 2
data_2=data[data.iloc[:,-1]==2]
data_2=data_2.drop([4],axis=1)
list_data.append(data_2)
list_cov.append(np.cov(data_2.T))

x_2_mean=np.mean(data_2,axis=0)
list_mean.append(x_2_mean.to_numpy().reshape(-1,1))

#Trung bình và ma trận hiệp phương sai của hộp sọ trong thời kì 3
data_3=data[data.iloc[:,-1]==3]
data_3=data_3.drop([4],axis=1)
list_data.append(data_3)
list_cov.append(np.cov(data_3.T))

x_3_mean=np.mean(data_3,axis=0)
list_mean.append(x_3_mean.to_numpy().reshape(-1,1))

data_all=data.copy()
data_all=data_all.drop([4],axis=1)
x_all_mean=np.mean(data_all,axis=0).to_numpy().reshape(-1,1)
```

In [29]:
```
#Tính B
B=np.zeros([x_1_mean.shape[0],x_1_mean.shape[0]])
for i in range(0,3):
    B+=list_data[i].shape[0]*np.matmul((list_mean[i]-x_all_mean),(list_mean[i]-x_all_mean).T)

print(B)
```

```
[[ 150.2          20.3        -161.83333333    5.03333333]
 [  20.3          20.6         -38.73333333    6.43333333]
 [-161.83333333  -38.73333333  190.28888889  -10.85555556]
 [   5.03333333    6.43333333  -10.85555556    2.02222222]]
```

In [30]:
```
#Tính W
W=np.zeros([x_1_mean.shape[0],x_1_mean.shape[0]])
for i in range(0,2):
    W+=(list_data[i].shape[0]-1)*list_cov[i]

print(W)
```

```
[[1433.93333333  149.7         151.43333333  263.56666667]
 [ 149.7        1205.5          74.6         174.5       ]
 [ 151.43333333   74.6        1552.03333333  -50.13333333]
 [ 263.56666667  174.5         -50.13333333  474.83333333]]
```

In [31]:
```
#Tính hệ số wilk lambda
wilk_lambda=(np.linalg.det(W))/(np.linalg.det(B+W))
print(wilk_lambda)
```

```
0.7730783604911553
```

In [32]:
```
#Tính giá trị
p=4
value=((data.shape[0]-p-2)/p)*((1-np.sqrt(wilk_lambda))/np.sqrt(wilk_lambda))
print('Calculated value: ',value)
```

```
Calculated value:  2.8840260538008735
```

In [33]:
```
#Tính giá trị tới hạn
f=stats.f.ppf(q=1-0.05,dfn=2*p,dfd=2*(data.shape[0]-p-2))
print('Critical value: ',f)
```

```
Critical value:  1.9938838709988889
```

Ta nhận thấy 2.884 > 1.99388. Nghĩa là: $(\frac{\sum n_l - p - 2}{p})(\frac{1-\sqrt{\Lambda*}}{\sqrt{\Lambda*}}) > F_{2p,2(\sum n_l - p - 2)}(\alpha)$. Nên ta bác bỏ H_0 với mức ý nghĩa 95%. Nghĩa là có sự thay đổi trong kích thước hộp sọ ở 3 thời kì

In [34]:
```
#Tính khoảng tin cậy 95%
t=stats.t.ppf(q=1-0.05/(p*3*2),df=data.shape[0]-3)
for i in range(0,4):
    for l in range(0,3):
        for k in range(l+1,3):
            lower=list_mean[k][i]-list_mean[l][i]-t*np.sqrt(1/(data.shape[0]-3)*(1/list_data[k].shape[0]+1/list_data[l]
.shape[0])*W[i][i])
            upper=list_mean[k][i]-list_mean[l][i]+t*np.sqrt(1/(data.shape[0]-3)*(1/list_data[k].shape[0]+1/list_data[l]
.shape[0])*W[i][i])
            print('95% Confidence interval of tau_{}{} - tau_{}{}: [{}, {}]'.format(k,i,l,i,lower,upper))
```

```
95% Confidence interval of tau_10 - tau_00: [[-2.08494115], [4.08494115]]
95% Confidence interval of tau_20 - tau_00: [[0.01505885], [6.18494115]]
95% Confidence interval of tau_20 - tau_10: [[-0.98494115], [5.18494115]]
95% Confidence interval of tau_11 - tau_01: [[-3.72856401], [1.92856401]]
95% Confidence interval of tau_21 - tau_01: [[-2.62856401], [3.02856401]]
95% Confidence interval of tau_21 - tau_11: [[-1.72856401], [3.92856401]]
95% Confidence interval of tau_12 - tau_02: [[-3.30946708], [3.10946708]]
95% Confidence interval of tau_22 - tau_02: [[-6.34280042], [0.07613375]]
95% Confidence interval of tau_22 - tau_12: [[-6.24280042], [0.17613375]]
95% Confidence interval of tau_13 - tau_03: [[-2.07522336], [1.47522336]]
95% Confidence interval of tau_23 - tau_03: [[-1.74189002], [1.80855669]]
95% Confidence interval of tau_23 - tau_13: [[-1.44189002], [2.10855669]]
```

Ta nhận thấy có khoảng tin cậy 95% ở biến 0 là độ rộng tối đa của hộp sọ không có chứa 0. Đây chính là thành phần có trung bình mà khác nhau trong các tổng thể qua các thời kì. Để kiểm tra giả định MANOVA ta sẽ dùng kiểm định Box để kiểm định tính bằng nhau của các ma trận hiệp phương sai với mức ý nghĩa 5%

In [35]:
```
#Box Test cho kiểm định tính bằng nhau của ma trận hiệp phương sai với mức ý nghĩa 95%
n_1,n_2,n_3=data_1.shape[0],data_2.shape[0],data_3.shape[0]
p=4
g=3
u=(1/(n_1-1)+1/(n_2-1)+1/(n_3-1)-1/(n_1+n_2+n_3-3))*((2*p*p+3*p-1)/(6*(p+1)*(g-1)))
S_pooled=1/(n_1+n_2+n_3-3)*((n_1-1)*list_cov[0]+(n_2-1)*list_cov[1]+(n_3-1)*list_cov[2])
C=(1-u)*((n_1+n_2+n_3-3)*np.log(np.linalg.det(S_pooled))-(n_1-1)*np.log(np.linalg.det(list_cov[0]))-(n_2-1)*np.log(np.l
inalg.det(list_cov[1]))-(n_3-1)*np.log(np.linalg.det(list_cov[2])))
print('C value: ',C)
chi=stats.chi2.ppf(q=1-0.05,df=p*(p+1)*(g-1)/2)
print('Critical value chi square: ',chi)
```

```
C value:  21.04843638691599
Critical value chi square:  31.410432844230918
```

Ta thấy $C = 21.04 < 31.414 = \chi^2_{p(p+1)(g-1)/2}(\alpha)$ Nên ta không đủ cơ sở bác bỏ H_0. Do đó, với mức ý nghĩa $\alpha = .05$, ta kết luận các ma trận hiệp phương sai của các tổng thể là bằng nhau. Do đó, giả đinh MANOVA thông thường là phù hợp với bộ dữ liệu này

6.28

Hai loài ruồi cắn-biting flies (chi Leptoconops) giống nhau về hình thái, đến nỗi trong nhiều năm chúng được cho là một. Các sai khác về mặt sinh học như tỷ lệ giới tính của emerging flies và biting flies được tìm thấy. Dữ liệu phân loại được liệt kê trong bảng 6.15 trang 352 và trên trang www.prenhall.com/statistics có cho thấy sự khác biệt nào giữa hai loài *L.carteri* và *L.torrens* không? Kiểm định tính bằng nhau giữa các vectơ trung bình tổng thể với mức ý nghĩa $\alpha = 0.05$. Nếu giả thuyết các vectơ trung bình bằng nhau bị bác bỏ, hãy xác định các thành phần của vectơ trung bình (hoặc các tổ hợp tuyến tình của các thành phần của vectơ trung bình) ảnh hưởng nhiều nhất đến việc bác bỏ H_0. Biện minh cho việc sử dụng các phương pháp lý thuyết thông thường cho bộ dữ liệu này.

In [36]:
```
path_28 = 'T6-15.txt'
data_28 = pd.read_table(path_28, delim_whitespace=True, header=None)
data_28.columns=['Wing_Length', 'WingWidth', 'Third_Palp_Length', 'Third_Palp_Width', 'Fourth_Palp_Length', 'Length_of_
Antennal_Segment12', 'Length_of_Antennal_Segment13', 'Species']
data_28
```

Out[36]:

	Wing_Length	WingWidth	Third_Palp_Length	Third_Palp_Width	Fourth_Palp_Length	Length_of_Antennal_Segment12	Length_of_Antennal_Segment13	Species
0	85	41	31	13	25	9	8	0
1	87	38	32	14	22	13	13	0
2	94	44	36	15	27	8	9	0
3	92	43	32	17	28	9	9	0
4	96	43	35	14	26	10	10	0
...	...	...	...	...	...	...	...	...
65	101	47	38	14	37	11	11	1
66	103	47	40	15	32	11	11	1
67	99	43	37	14	23	11	10	1
68	105	50	40	16	33	12	11	1
69	99	47	39	14	34	7	7	1

70 rows × 8 columns

In [37]:
```
L_torrens_df = data_28[data_28['Species']==0].iloc[:,:-1]
n1, p = L_torrens_df.shape
L_carteri_df = data_28[data_28['Species']==1].iloc[:,:-1]
n2 = L_torrens_df.shape[0]

sample_mean = pd.DataFrame({'L.torrens': np.array(L_torrens_df.mean()), 'L.carteri': np.array(L_carteri_df.mean())})
sample_mean['Difference'] = sample_mean['L.torrens'] - sample_mean['L.carteri']

print('>> Sample Mean for L.torrens and L.carteri: ')
sample_mean
```

```
>> Sample Mean for L.torrens and L.carteri:
```

Out[37]:

	L.torrens	L.carteri	Difference
0	96.457143	99.342857	-2.885714
1	42.914286	43.742857	-0.828571
2	35.371429	39.314286	-3.942857
3	14.514286	14.657143	-0.142857
4	25.628571	30.000000	-4.371429
5	9.571429	9.657143	-0.085714
6	9.714286	9.371429	0.342857

In [38]:
```
S1 = np.cov(L_torrens_df.T)
S2 = np.cov(L_carteri_df.T)

S_pooled = (n1-1)/(n1+n2-2)*S1 + (n2-1)/(n1+n2-2)*S2
print('>> Pooled Sample Covariance Matrix: ')
S_pooled
```

```
>> Pooled Sample Covariance Matrix:
```

Out[38]:
```
array([[36.00840336, 14.59495798,  6.07773109,  3.67478992,  9.57268908,
         2.42605042,  2.6487395 ],
       [14.59495798, 16.63865546,  2.76386555,  2.99201681,  6.1012605 ,
         1.05336134,  0.93361345],
       [ 6.07773109,  2.76386555,  6.43697479,  0.69243697,  1.61512605,
         0.21092437,  0.6710084 ],
       [ 3.67478992,  2.99201681,  0.69243697,  3.03865546,  2.40714286,
         0.27352941,  0.22941176],
       [ 9.57268908,  6.1012605 ,  1.61512605,  2.40714286, 13.76722689,
         0.56512605,  0.63655462],
       [ 2.42605042,  1.05336134,  0.21092437,  0.27352941,  0.56512605,
         1.21260504,  0.91428571],
       [ 2.6487395 ,  0.93361345,  0.6710084 ,  0.22941176,  0.63655462,
         0.91428571,  0.98991597]])
```

Phát biểu giả thuyết:

$$H_0 : \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2 = 0 \quad \text{và} \quad H_1 : \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2 \neq 0$$

Giả sử H_0 đúng, ta có thống kê

$$T^2 = (\overline{\mathbf{x}}_1 - \overline{\mathbf{x}}_2)' \left[\left(\frac{1}{n_1} + \frac{1}{n_2} \right) \mathbf{S}_{\text{pooled}} \right]^{-1} (\overline{\mathbf{x}}_1 - \overline{\mathbf{x}}_2) \sim \frac{(n_1 + n_2 - 2)p}{n_1 + n_2 - p - 1} F_{p, n_1 + n_2 - p - 1}$$

Với mẫu thực nghiệm, ta có giá trị thống kê như sau

In [39]:
```
T2 = np.transpose(sample_mean['Difference']).dot(np.linalg.inv((1/n1+1/n2)*S_pooled).dot(sample_mean['Difference']))
print('>> Giá trị thống kê: ',T2)
```

```
>> Giá trị thống kê:  106.13481343310384
```

Với mức ý nghĩa $\alpha = 0.05$, ta có

In [40]:
```
alpha = 0.05

f = stats.f.ppf(q=1-alpha, dfn=p, dfd=n1+n2-p-1)
F = (((n1+n2-2)*p)/(n1+n2-p-1))*f

F
```

Out[40]:
```
16.593314751671375
```

Vì 106.13481343310384 > 16.593314751671375 nên ta bác bỏ giả thuyết H_0 với mức ý nghĩa $\alpha = 0.05$. Do đó với mức ý nghĩa 0.05, có sự khác biệt giữa hai loài.

Tổ hợp tuyến tính quan trọng nhất của các thành phần trung bình dẫn đến việc bác bỏ giả thuyết H_0 có vectơ hệ số như sau

$$\hat{\mathbf{a}} \propto \mathbf{S}_{\mathbf{pooled}}^{-1}(\overline{\mathbf{x}}_1 - \overline{\mathbf{x}}_2)$$

In [41]:
```
a_hat = np.linalg.inv(S_pooled).dot(sample_mean['Difference'])
np.array(a_hat).reshape(-1,1)
```

Out[41]:
```
array([[ 0.00623561],
       [ 0.15059417],
       [-0.85382218],
       [ 0.26757305],
       [-0.38278705],
       [-2.18730695],
       [ 2.97072661]])
```

Ta có thể thấy rằng, sự khác biệt giữa hai loài L.torrens và L.carteri được thể hiện nổi bật ở thành phần *Length of Antennal Segment 13*.

In [42]:
```
lcb = lambda i: (sample_mean['Difference'][i]) - np.sqrt(F)*np.sqrt((1/n1+1/n2)*S_pooled[i][i])
ucb = lambda i: (sample_mean['Difference'][i]) + np.sqrt(F)*np.sqrt((1/n1+1/n2)*S_pooled[i][i])

# 95% simultaneous confidence intervals for individual means
print("95% simultaneous confidence intervals for mean components are: \n")
IC = []
for i in range(p):
    ic = [lcb(i), ucb(i)]
    print(">> {}:\n{}\n".format(data_28.columns[i], ic))
    IC.append(ic)
```

```
95% simultaneous confidence intervals for mean components are:

>> Wing_Length:
[-8.72889722791249, 2.957468656483919]

>> WingWidth:
[-4.80054788605201, 3.143405028909152]

>> Third_Palp_Length:
[-6.413377605520244, -1.4723366801940418]

>> Third_Palp_Width:
[-1.8402731034264352, 1.5545588177121505]

>> Fourth_Palp_Length:
[-7.984452070272665, -0.7584050725844755]

>> Length_of_Antennal_Segment12:
[-1.1579918918712138, 0.9865633204426429]

>> Length_of_Antennal_Segment13:
[-0.6259709329409787, 1.311685218655262]
```

Các thành phần *Third Palp Length* và *Fourth Palp Length* có khoảng tin cậy đồng thời 95% không chứa 0. Do đó việc bác bỏ giả thuyết H_0 là hợp lý.

6.30

Bảng 6.16 ở trang 353 chứa thông tin về thành phần trong các quặng khoáng sản của xương. Với 24 đối tượng đầu tiên là bảng 1.8, 1 năm sau sự tham gia của họ vào một chương trình thí nghiệm. So sánh dữ liệu từ hai bảng và xác định xem có mất mát xương không ?

In [43]:
```
path = 'T6-16.txt'
bone_after = pd.DataFrame(np.loadtxt(path))
print(bone_after.head(11))
print(bone_after.shape)
```

```
       0      1      2      3      4      5
0   1.027  1.051  2.268  2.246  0.869  0.964
1   0.857  0.817  1.718  1.710  0.602  0.689
2   0.875  0.880  1.953  1.756  0.765  0.738
3   0.873  0.698  1.668  1.443  0.761  0.698
4   0.811  0.813  1.643  1.661  0.551  0.619
5   0.640  0.734  1.396  1.378  0.753  0.515
6   0.947  0.865  1.851  1.686  0.708  0.787
7   0.886  0.806  1.742  1.815  0.687  0.715
8   0.991  0.923  1.931  1.776  0.844  0.656
9   0.977  0.925  1.933  2.106  0.869  0.789
10  0.825  0.826  1.609  1.651  0.654  0.726
(24, 6)
```

In [44]:
```
path = 'T1-8.txt'
bone_before = pd.DataFrame(np.loadtxt(path))
bone_before=bone_before.iloc[:24,:]
print(bone_before.head(11))
print(bone_before.shape)
```

```
       0      1      2      3      4      5
0   1.103  1.052  2.139  2.238  0.873  0.872
1   0.842  0.859  1.873  1.741  0.590  0.744
2   0.925  0.873  1.887  1.809  0.767  0.713
3   0.857  0.744  1.739  1.547  0.706  0.674
4   0.795  0.809  1.734  1.715  0.549  0.654
5   0.787  0.779  1.509  1.474  0.782  0.571
6   0.933  0.880  1.695  1.656  0.737  0.803
7   0.799  0.851  1.740  1.777  0.618  0.682
8   0.945  0.876  1.811  1.759  0.853  0.777
9   0.921  0.906  1.954  2.009  0.823  0.765
10  0.792  0.825  1.624  1.657  0.686  0.668
(24, 6)
```

(a) Kiểm định với $\alpha = .05$

In [45]:
```
# vecto trung bình
bone_before_mean = np.array(bone_before.apply(np.mean))
print("\nMean vector Mineral Content Before: \n", bone_before_mean)

# Ma trận hiệp phương sai
S_before = np.array(np.cov(bone_before.T))
print("\nCovariance matrix Mineral Content Before: \n", S_before)

# vecto trung bình
bone_after_mean = np.array(bone_after.apply(np.mean))
print("\nMean vector Mineral Content After: \n", bone_after_mean)

# Ma trận hiệp phương sai
S_after = np.array(np.cov(bone_after.T))
print("\nCovariance matrix Male: \n", S_after)
```

```
Mean vector Mineral Content Before:
 [0.84083333 0.81341667 1.78525    1.72925    0.69754167 0.68658333]

Covariance matrix Mineral Content Before:
 [[0.01333728 0.0104502  0.0227467  0.02052635 0.00898649 0.00774223]
 [0.0104502  0.01128712 0.01839067 0.02130185 0.00802337 0.00836762]
 [0.0227467  0.01839067 0.08241089 0.06859602 0.01623964 0.01199907]
 [0.02052635 0.02130185 0.06859602 0.07169037 0.01750647 0.01646554]
 [0.00898649 0.00802337 0.01623964 0.01750647 0.01084435 0.00712376]
 [0.00774223 0.00836762 0.01199907 0.01646554 0.00712376 0.00968625]]

Mean vector Mineral Content After:
 [0.84095833 0.81016667 1.77808333 1.71691667 0.71266667 0.68675   ]

Covariance matrix Male:
 [[0.01551526 0.01116144 0.02949613 0.02415421 0.00897607 0.00928429]
 [0.01116144 0.01175754 0.02348533 0.02357567 0.00868114 0.0096873 ]
 [0.02949613 0.02348533 0.10505521 0.08406183 0.02396133 0.01832702]
 [0.02415421 0.02357567 0.08406183 0.0843433  0.02121584 0.01965733]
 [0.00897607 0.00868114 0.02396133 0.02121584 0.01150754 0.00754317]
 [0.00928429 0.0096873  0.01832702 0.01965733 0.00754317 0.0125462 ]]
```

In [46]:
```
n_1=n_2=24
S_pooled=(n_1-1)/(n_1+n_2-2)*S_before+(n_2-1)/(n_1+n_2-2)*S_after
print(S_pooled)
```

```
[[0.01442627 0.01080582 0.02612141 0.02234028 0.00898128 0.00851326]
 [0.01080582 0.01152233 0.020938   0.02243876 0.00835226 0.00902746]
 [0.02612141 0.020938   0.09373305 0.07632893 0.02010049 0.01516304]
 [0.02234028 0.02243876 0.07632893 0.07801683 0.01936115 0.01806143]
 [0.00898128 0.00835226 0.02010049 0.01936115 0.01117594 0.00733347]
 [0.00851326 0.00902746 0.01516304 0.01806143 0.00733347 0.01111622]]
```

In [47]:
```
t_2=np.matmul((bone_after_mean-bone_before_mean).T,np.linalg.inv((1/n_1+1/n_2)*S_pooled))
t_2=np.matmul(t_2,(bone_after_mean-bone_before_mean))
print('Hotelling T^2: ',t_2)
```

```
Hotelling T^2:  0.8299432210795186
```

In [48]:
```
p=6
f = stats.f.ppf(q=1-0.05, dfn=p, dfd=n_1+n_2-p-1)
c_2=(n_1+n_2-2)*p/(n_1+n_2-p-1)*f
print('Critical Value: ',c_2)
```

```
Critical Value:  15.683337743007508
```

Ta thấy Hotelling 0.829 < 15.683 nên ta không đủ cơ sở để bác bỏ H_0 với mức ý nghĩa 5%. Ta kết luận các thành phần của xương không bị mất mát

(b) Xây dựng khoảng tin cậy đồng thời 95% cho sự chênh lệch trung bình

In [49]:
```
for i in range(6):
    lowerbound=(bone_after_mean[i]-bone_before_mean[i])-np.sqrt(c_2)*np.sqrt((1/n_1+1/n_2)*S_pooled[i][i])
    upperbound=(bone_after_mean[i]-bone_before_mean[i])+np.sqrt(c_2)*np.sqrt((1/n_1+1/n_2)*S_pooled[i][i])
    print('95% confidence interval for mu_1{}-mu_2{} in range [{}, {}]'.format(i+1,i+1,lowerbound,upperbound))
```

```
95% confidence interval for mu_11-mu_21 in range [-0.13718608872058177, 0.13743608872058163]
95% confidence interval for mu_12-mu_22 in range [-0.12596531703856026, 0.11946531703856053]
95% confidence interval for mu_13-mu_23 in range [-0.35717227283569053, 0.34283893950235655]
95% confidence interval for mu_14-mu_24 in range [-0.33165087993423037, 0.30698421326756375]
95% confidence interval for mu_15-mu_25 in range [-0.10573168483474202, 0.13598168483474235]
95% confidence interval for mu_16-mu_26 in range [-0.12036669849968282, 0.12070003183301634]
```

(c) Xây dựng khoảng tin cậy 95% Bonferroni và so sánh khoảng này với khoảng ở phần b

In [50]:
```
t = stats.t.ppf(q=1-(0.05/(2*p)),df=n_1+n_2-2)
for i in range(6):
    lowerbound=(bone_after_mean[i]-bone_before_mean[i])-t*np.sqrt((1/n_1+1/n_2)*S_pooled[i][i])
    upperbound=(bone_after_mean[i]-bone_before_mean[i])+t*np.sqrt((1/n_1+1/n_2)*S_pooled[i][i])
    print('Bonferroni confidence interval for mu_1{}-mu_2{} in range [{},{}]'.format(i+1,i+1,lowerbound,upperbound))
```

```
Bonferroni confidence interval for mu_11-mu_21 in range [-0.09547340556508102,0.09572340556508088]
Bonferroni confidence interval for mu_12-mu_22 in range [-0.08868657148602403,0.0821865714860243]
Bonferroni confidence interval for mu_13-mu_23 in range [-0.2508467525240225,0.23651341919068852]
Bonferroni confidence interval for mu_14-mu_24 in range [-0.23464785001536137,0.20998118334869476]
Bonferroni confidence interval for mu_15-mu_25 in range [-0.06901755891302268,0.09926755891302301]
Bonferroni confidence interval for mu_16-mu_26 in range [-0.08375079138178526,0.08408412471511878]
```

Từ các khoảng tin cậy trên, ta thấy khoảng tin cậy Bonferroni hẹp hơn so với khoảng tin cậy còn lại.

6.33

Tham khảo bài tập 6.32. Dữ liệu trong bảng 6.18 là các phép đo trên các biến: $X_1 =$ phần trăm độ phản xạ phổ ở bước sóng 560 nm (xanh lục), $X_2 =$ phần trăm độ phản xạ phổ ở bước sóng 720 nm (gần hồng ngoại), cho ba loài (sitka spruce [SS], Japanese larch [JL] và lodgepole pine [LP]) của một cây con 1 tuổi được lấy vào 3 thời điểm khác nhau (Julian day 150 [1], Julian day 235 [2] và Julian day 320 [3]) trong mùa sinh trưởng. Tất cả các cây con đều được trồng với mức dinh dưỡng tối ưu.

In [51]:
```
path_33 = 'T6-18.txt'
data_33 = pd.read_table(path_33, delim_whitespace=True, header=None)
data_33.columns=['x1', 'x2', 'Species', 'Time', 'Replication']
data_33
```

Out[51]:

	x1	x2	Species	Time	Replication
0	9.33	19.14	SS	1	1
1	8.74	19.55	SS	1	2
2	9.31	19.24	SS	1	3
3	8.27	16.37	SS	1	4
4	10.22	25.00	SS	2	1
5	10.13	25.32	SS	2	2
6	10.42	27.12	SS	2	3
7	10.62	26.28	SS	2	4
8	15.25	38.89	SS	3	1
9	16.22	36.67	SS	3	2
10	17.24	40.74	SS	3	3
11	12.77	67.50	SS	3	4
12	12.07	33.03	JL	1	1
13	11.03	32.37	JL	1	2
14	12.48	31.31	JL	1	3
15	12.12	33.33	JL	1	4
16	15.38	40.00	JL	2	1
17	14.21	40.48	JL	2	2
18	9.69	33.90	JL	2	3
19	14.35	40.15	JL	2	4
20	38.71	77.14	JL	3	1
21	44.74	78.57	JL	3	2
22	36.67	71.43	JL	3	3
23	37.21	45.00	JL	3	4
24	8.73	23.27	LP	1	1
25	7.94	20.87	LP	1	2
26	8.37	22.16	LP	1	3
27	7.86	21.78	LP	1	4

	x1	x2	Species	Time	Replication
28	8.45	26.32	LP	2	1
29	6.79	22.73	LP	2	2
30	8.34	26.67	LP	2	3
31	7.54	24.87	LP	2	4
32	14.04	44.44	LP	3	1
33	13.51	37.93	LP	3	2
34	13.33	37.93	LP	3	3
35	12.77	60.87	LP	3	4

(a) Thực hiện two-factor MANOVA bằng cách sử dụng dữ liệu trong bảng 6.18. Kiểm định cho species effect, time effect và species-time interation với mức ý nghĩa $\alpha = 0.05$.

In [52]:

```
class TwoWayMANOVA:
    def __init__(self):
        self.g = None
        self.b = None
        self.n = None
        self.p = None
        self.mean = None
        self.list_means = []
        self.list_cov = []
        self.Fac1 = None
        self.Fac2 = None
        self.Int = None
        self.Res = None
        self.Cor = None

    def _calc_init_values(self, X, y1, y2):
        self.g = len(np.unique(y1))
        self.b = len(np.unique(y2))
        _, counts = np.unique(y1, return_counts=True)
        self.n = counts[0]/self.b
        self.p = X.shape[1]
        self.mean = np.array(np.mean(X,axis=0)).reshape(-1,1)
        for i in np.unique(y1):
            data = X[y1==i]
            temp_mean = []
            temp_cov = []
            for j in np.unique(y2):
                data1 = data[y2==j]
                temp_mean.append(np.array(np.mean(data1, axis=0)).reshape(-1,1))
                temp_cov.append(np.array(np.cov(data1.T)))
            self.list_means.append(temp_mean)
            self.list_cov.append(temp_cov)
        self.list_means = np.array(self.list_means)
        self.list_cov = np.array(self.list_cov)

    def fit(self, X, y1, y2):
        self._calc_init_values(X, y1, y2)

        Fac1 = np.zeros((self.p, self.p))
        Fac2 = np.zeros((self.p, self.p))
        Int = np.zeros((self.p, self.p))
        Res = np.zeros((self.p, self.p))
        for l in range(self.g):
            Fac1 += self.b*self.n*(np.mean(self.list_means[l], axis=0) - self.mean).dot((np.mean(self.list_means[l], ax
is=0) - self.mean).T)
            for k in range(self.b):
```

```
                Int += self.n*(self.list_means[l][k] - np.mean(self.list_means[l], axis=0) - np.mean(self.list_means[:,
k], axis=0) + self.mean).dot((self.list_means[l][k] - np.mean(self.list_means[l], axis=0) - np.mean(self.list_means[:,k
], axis=0) + self.mean).T)
                Res += (self.n-1)*self.list_cov[l][k]

        for k in range(self.b):
            Fac2 += self.g*self.n*(np.mean(self.list_means[:,k], axis=0) - self.mean).dot((np.mean(self.list_means[:,k
], axis=0) - self.mean).T)
        self.Fac1 = (Fac1, self.g-1)
        self.Fac2 = (Fac2, self.b-1)
        self.Int = (Int, (self.g-1)*(self.b-1))
        self.Res = (Res, self.g*self.b*(self.n-1))
        self.Cor = (Fac1+Fac2+Int+Res, self.g*self.b*self.n - 1)

    def table(self):
        print('>> MANOVA Table: \n')
        print('*'*50)
        print('> Factor 1: \nSSP_fac1 = \n{},\t d.f. = {}'.format(self.Fac1[0], self.Fac1[1]))
        print('\n> Factor 2: \nSSP_fac2 = \n{},\t d.f. = {}'.format(self.Fac2[0], self.Fac2[1]))
        print('\n> Interaction: \nSSP_int = \n{},\t d.f. = {}'.format(self.Int[0], self.Int[1]))
        print('\n> Residual: \nSSP_res = \n{},\t d.f. = {}'.format(self.Res[0], self.Res[1]))
        print('\n> Total: \nSSP_cor = \n{},\t d.f. = {}'.format(self.Cor[0], self.Cor[1]))
        print('*'*50)
```

In [53]:
```
X = data_33[['x1', 'x2']]
y1 = data_33['Species']
y2 = data_33['Time']
maov = TwoWayMANOVA()
maov.fit(X, y1, y2)
maov.table()
```

```
>> MANOVA Table:

************************************************
> Factor 1:
SSP_fac1 =
[[ 965.18117222 1377.60191389]
 [1377.60191389 2026.85637222]],         d.f. = 2

> Factor 2:
SSP_fac2 =
[[1275.24773889 2644.92736389]
 [2644.92736389 5573.80570556]],         d.f. = 2

> Interaction:
SSP_int =
[[795.80794444 375.96311944]
 [375.96311944 193.54926111]],   d.f. = 4

> Residual:
SSP_res =
[[  76.658775   37.9299  ]
 [  37.9299   1769.642225]],     d.f. = 27.0

> Total:
SSP_cor =
[[3112.89563056 4436.42229722]
 [4436.42229722 9563.85356389]],         d.f. = 35.0
************************************************

<ipython-input-52-23f327f81e92>:28: UserWarning: Boolean Series key will be reindexed to match DataFrame index.
  data1 = data[y2==j]
```

Xét thống kê

$$-\left[gb(n-1)-\frac{p+1-(g-1)(b-1)}{2}\right]\ln\Lambda^* \sim \chi^2_{(g-1)(b-1)p}(\alpha)$$

kiểm định giả thuyết $H_0: \gamma_{11}=\gamma_{12}=\ldots=\gamma_{gb}=\mathbf{0}$, trong đó

$$\Lambda^* = \frac{|SSP_{res}|}{|SSP_{int}+SSP_{res}|}$$

có giá trị là

In [54]:
```
Lambda = np.linalg.det(maov.Res[0])/np.linalg.det(maov.Int[0] + maov.Res[0])
print('Lambda* = ', Lambda)
```

```
Lambda* =  0.08707032115867996
```

Ta có giá trị thống kê và giá trị tới hạn của phân phối chi bình phương là

In [55]:
```
value = -(maov.g*maov.b*(maov.n-1)-(maov.p+1-(maov.g-1)*(maov.b-1))/2)*np.log(Lambda)
print('> Giá trị thống kê: ', value)

alpha = 0.05
chisq = stats.chi2.ppf(1-alpha, (maov.g-1)*(maov.b-1)*maov.p)
print('> Giá trị tại phân vị trên thứ 5 của phân phối chi bình phương: ', chisq)
```

```
> Giá trị thống kê:  67.12857793502033
> Giá trị tại phân vị trên thứ 5 của phân phối chi bình phương:  15.50731305586545
```

Vì $67.12857793502033 > 15.50731305586545$ nên ta bác bỏ H_0 với mức ý nghĩa 5%. Do đó với mức ý nghĩa 5%, không có sự ảnh hưởng qua lại giữa thành phần *Species* và *Time*.

Xét thống kê

$$-\left[gb(n-1)-\frac{p+1-(g-1)}{2}\right]\ln\Lambda_1^* \sim \chi^2_{(g-1)p}(\alpha)$$

kiểm định giả thuyết $H_0: \boldsymbol{\tau}_1 = \boldsymbol{\tau}_2 = \ldots = \boldsymbol{\tau}_g = \mathbf{0}$, trong đó

$$\Lambda_1^* = \frac{|SSP_{res}|}{|SSP_{fac1} + SSP_{res}|}$$

có giá trị là

In [56]:
```
Lambda1 = np.linalg.det(maov.Res[0])/np.linalg.det(maov.Fac1[0] + maov.Res[0])
print('Lambda1* = ', Lambda1)
```

```
Lambda1* =  0.06877382340376705
```

Ta có giá trị thống kê và giá trị tới hạn của phân phối chi bình phương là

In [57]:
```
value = -(maov.g*maov.b*(maov.n-1)-(maov.p+1-(maov.g-1))/2)*np.log(Lambda1)
print('> Giá trị thống kê: ', value)

chisq = stats.chi2.ppf(1-alpha, (maov.g-1)*maov.p)
print('> Giá trị tại phân vị trên thứ 5 của phân phối chi bình phương: ', chisq)
```

```
> Giá trị thống kê:  70.93870012593094
> Giá trị tại phân vị trên thứ 5 của phân phối chi bình phương:  9.487729036781154
```

Vì $70.93870012593094 > 9.487729036781154$ nên ta bác bỏ H_0 với mức ý nghĩa 5%. Do đó với mức ý nghĩa 5%, không có sự ảnh hưởng của *Species*.

Xét thống kê

$$-\left[gb(n-1)-\frac{p+1-(b-1)}{2}\right]\ln\Lambda_1^* \sim \chi^2_{(b-1)p}(\alpha)$$

kiểm định giả thuyết $H_0: \boldsymbol{\beta}_1 = \boldsymbol{\beta}_2 = \ldots = \boldsymbol{\beta}_b = \mathbf{0}$, trong đó

$$\Lambda_2^* = \frac{|SSP_{res}|}{|SSP_{fac2} + SSP_{res}|}$$

có giá trị là

In [58]:
```
Lambda2 = np.linalg.det(maov.Res[0])/np.linalg.det(maov.Fac2[0] + maov.Res[0])
print('Lambda2* = ', Lambda2)
```

```
Lambda2* =  0.049166033502360484
```

Ta có giá trị thống kê và giá trị tới hạn của phân phối chi bình phương là

In [59]:
```
value = -(maov.g*maov.b*(maov.n-1)-(maov.p+1-(maov.b-1))/2)*np.log(Lambda2)
print('> Giá trị thống kê: ', value)

chisq = stats.chi2.ppf(1-alpha, (maov.b-1)*maov.p)
print('> Giá trị tại phân vị trên thứ 5 của phân phối chi bình phương: ', chisq)
```

```
> Giá trị thống kê:  79.8326351515917
> Giá trị tại phân vị trên thứ 5 của phân phối chi bình phương:  9.487729036781154
```

Vì $79.8326351515917 > 9.487729036781154$ nên ta bác bỏ H_0 với mức ý nghĩa 5%. Do đó với mức ý nghĩa 5%, không có sự ảnh hưởng của *Time*.

(b) Bạn có nghĩ rằng các giả thuyết MANOVA thông thường được thoả mãn với bộ dữ liệu này không? Thảo luận với sự tương quan giữa residual analysis và khả năng các quan trắc có quan hệ với nhau theo thời gian.

Các quan trắc có tương quan với nhau theo thời gian. Các quan trắc này không độc lập.

(c) Kiểm lâm đặc biệt quan tâm đến sự tác động giữa các loài và thời gian. Sự tác động có cho thấy một biến mà có cho thấy biến còn lại không? Kiểm tra bằng cách chạy two-factor ANOVA cho mỗi phản hồi trong hai phản hồi.

In [60]:
```
from statsmodels.multivariate.manova import MANOVA
import statsmodels.api as sm
from statsmodels.formula.api import ols


print('Analysis of Variance for 560nm')
reg = ols("x1 ~ Species + Time + Species : Time", data=data_33).fit()
sm.stats.anova_lm(reg, typ=2)
```

```
Analysis of Variance for 560nm
```

Out[60]:

	sum_sq	df	F	PR(>F)
Species	965.181172	2.0	28.490840	1.162524e-07
Time	1016.731837	1.0	60.025091	1.209455e-08
Species:Time	622.829200	2.0	18.385074	6.138805e-06
Residual	508.153421	30.0	NaN	NaN

In [61]:
```
print('Analysis of Variance for 720nm')
reg = ols("x2 ~ Species + Time + Species : Time", data=data_33).fit()
sm.stats.anova_lm(reg, type=2)
```

```
Analysis of Variance for 720nm
```

Out[61]:

	df	sum_sq	mean_sq	F	PR(>F)
Species	2.0	2026.856372	1013.428186	12.506589	1.121548e-04
Time	1.0	4950.466504	4950.466504	61.093082	1.010658e-08
Species:Time	2.0	155.584508	77.792254	0.960024	3.943353e-01
Residual	30.0	2430.946179	81.031539	NaN	NaN

Từ 2 bảng ANOVA trên, ta thấy rằng Interaction giữa Species và Time chỉ cho thấy ở bước sóng 560nm còn trên bước sóng 720nm thì không do p-value của interaction tại bước sóng 560nm nhỏ hơn $\alpha = 0.05$, còn ở bước sóng 720nm thì không.

(d) Bạn có thể nghĩ ra một phương pháp để phân tích những dữ liệu này (hoặc một thiết kế thử nghiệm khác) cho phép tạo ra xu hướng thời gian tiềm năng trong số các phản xạ quang phổ không?

Dữ liệu có thể được phân tích bằng cách sử dụng phương pháp đường tăng trưởng (growth curve) được trình bày trong **phần 6.9**. Dữ liệu có thể được phân tích bằng cách giả sử các loài (species) được *xếp lồng* vào ngày.