Multivariate Statistical Methods - Lab 02

Maximilian Pfundstein (maxpf364), Hector Plata (hecpl268), Aashana Nijhawan(aasni448), Lakshidaa Saigiridharan (laksa656)

2020-01-28

Contents

Focusing on the multivariate normal distribution, we will study methods for estimating, testing hypotheses about and comparing mean vectors. These methods are the multivariate generalizations of the univariate methods.

1 Test of Outliers

Consider again the data set from the `T1-9.dat` file, National track records for women. In the first assignment we studied different distance measures between an observation and the sample average vector. The most common multivariate residual is the Mahalanobis distance and we computed this distance for all observations.

1.1 Chi-Squared Approximation

The Mahalanobis distance is approximately chi–square distributed, if the data comes from a multivariate normal distribution and the number of observations is large. Use this chi–square approximation for testing each observation at the 0.1 percent significance level and conclude which countries can be regarded as outliers. Should you use a multiple–testing correction procedure? Compare the results with and without one. Why is (or maybe is not) 0.1 percent a sensible significance level for this task?

Answer: First we import, name and look at the track times.

```r
track_times = read.table("data/T1-9.dat")
colnames(track_times) = c("country", "100m", "200m", "400m",
                   "800m", "1500m", "3000m", "marathon")
head(track_times)
```

```
##   country  100m  200m  400m 800m 1500m 3000m marathon
## 1     ARG 11.57 22.94 52.50 2.05  4.25  9.19   150.32
## 2     AUS 11.12 22.23 48.63 1.98  4.02  8.63   143.51
## 3     AUT 11.15 22.70 50.62 1.94  4.05  8.78   154.35
## 4     BEL 11.14 22.48 51.45 1.97  4.08  8.82   143.05
## 5     BER 11.46 23.05 53.30 2.07  4.29  9.81   174.18
## 6     BRA 11.17 22.60 50.62 1.97  4.17  9.04   147.41
```

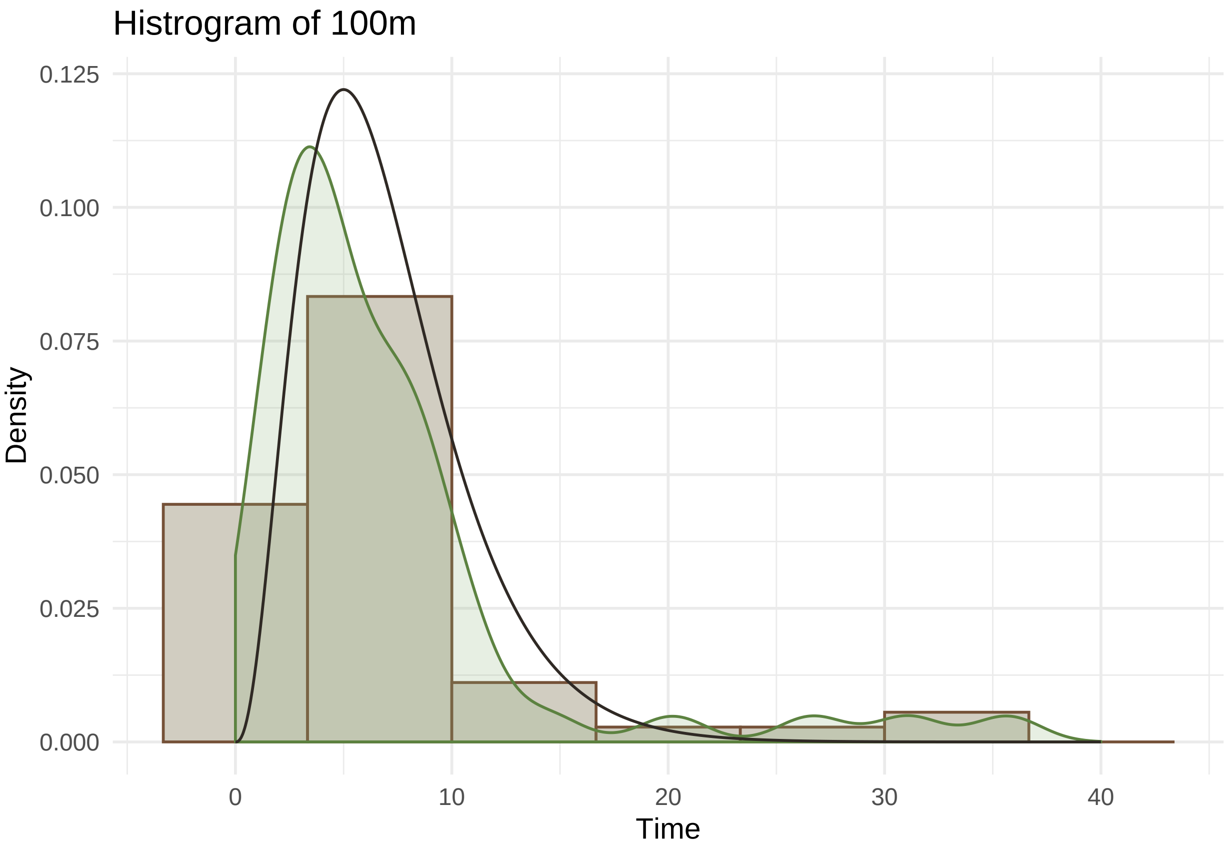
We reimport our function written in the previous lab for computing the Mahalanobis Distance as it is actually more convenient than the built-in function in R.

```
sample_variance = function(X) {

  X = as.matrix(X)

  identity = diag(nrow(X))
  one_n = matrix(1, nrow=nrow(X), ncol=1)

  inter =  identity - 1/nrow(X) * (one_n %*% t(one_n))

  return(1/nrow(X) * (t(X) %*% inter %*% X))
}

mahalanobis_distance = function(X) {
  X = as.matrix(X)

  V = sample_variance(X)
  ident = matrix(1, nrow=nrow(X), ncol=nrow(X))
  mu = 1/nrow(X) * (t(ident) %*% X)

  X_centered = X - mu

  return(diag(X_centered %*% solve(V) %*% t(X_centered)))
}
```

We set the degrees of freedom for the $\chi^2(\nu)$-distribution, which corresponds to the amount of features. We also calculate the Mahalanobis Distances. They should follow a $\chi^2(\nu)$-distribution with 7 degrees of freedom.

```
nu = ncol(track_times) - 1
D = mahalanobis_distance(track_times[,2:8])
```

The following plot shows the histogram of the Mahalanobis Distances with the respective density. The real $\chi^2(\nu)$-distribution with 7 degrees of freedom is outlined in black.



We define $\alpha = 0.001$ and we check for each observation if it lies within the $1 - \alpha$ percentile of the $\chi^2(\nu)$-distribution with 7 degrees of freedom. Finally we check which countries are the outliers. Our findings match the results from the previous lab.

Below is the test without any multiple test correction. In this case the countries KORN, PNG and SAM are regarded as outliers.

```
alpha = 0.001

outlier_indeces = 1 - pchisq(D, nu) < alpha

track_times$country[outlier_indeces]
```

```
## [1] KORN PNG  SAM
## 54 Levels: ARG AUS AUT BEL BER BRA CAN CHI CHN COK COL CRC CZE DEN ... USA
```

Below is the test results with the correction for multiple tests. In this case the only country that is an outlier is SAM.

```
alpha = 0.001 / nrow(track_times)

outlier_indeces = 1 - pchisq(D, nu) < alpha

track_times$country[outlier_indeces]
```

```
## [1] SAM
## 54 Levels: ARG AUS AUT BEL BER BRA CAN CHI CHN COK COL CRC CZE DEN ... USA
```

Multiple test correction should be used because we are doing multiple test at the same time, so the original significance level α should be adjusted to reflect that we are comparing multiple countries.

In this case the significance level $\alpha = 0.1\%$ it depends on what is to be considered an outlier. In this case it seems reasonable since we want countries were its distance is way off from the other ones. So 0.01% seems reasonable. But this also depends on the application.

1.2 Different Outlier Reasoning

One outlier is North Korea. This country is not an outlier with the Euclidean distance. Try to explain these seemingly contradictory result.

The contradictory results arise from the fact that the euclidean distance doesn't normalize the data give the variations on the observations (variance and covariance) while the Mahalanobis distance does consider these variations. This is why we are getting contradictory results.

2 Test, Confidence Region and Confidence Intervals for a Mean Vector

Look at the bird data in file `T5-12.dat` and solve Exercise 5.20 of *Johnson, Wichern*. Do not use any extra R package or built–in test but code all required matrix calculations. You MAY NOT use loops!

2.1 Confidence Ellipse

```r
library(ellipse)
```

```
##
## Attaching package: 'ellipse'

## The following object is masked from 'package:car':
##
##     ellipse

## The following object is masked from 'package:graphics':
##
##     pairs
```

```r
birddata = read.table("T5-12.DAT")
# Male hook-billed kites
n = nrow(birddata)
p = dim(birddata)[2]
mu = data.frame("mu1" = 190, "mu2" = 275)
meanVec =NULL
x_bar = NULL
meanMu = NULL
# x - x_bar
x_bar = data.frame("X1" = mean(birddata$V1), "X2" = mean(birddata$V2))

sample_var = function(obs){
  X = as.matrix(obs)
  IdentityMat = diag(nrow(obs))
  one_n = matrix(1,nrow(X),nrow(X))
  # sample var formula from notes
  mix = (IdentityMat - (1/nrow(X)) *(one_n))
  sample_variance = (1/(nrow(X) - 1)) * (t(X) %*% mix %*% X)
  return(sample_variance)
}

S = sample_var(birddata)
S =  matrix(S, ncol=2, nrow=2)
eigenValVec = eigen(S)
meanMu = as.matrix(x_bar - mu)

# To check if mu is on the confidence region we need to check if t^2 is < F_{n,n-p}

Tsq = n * (meanMu) %*% solve(S) %*% t(meanMu)

F_val =  qf(0.95,df1=p, df=n-p) * (p*(n-1)/(n-p)) #5% significance level

cat("Is T^2 less than F_val? =",Tsq<F_val)
```

```
## Is T^2 less than F_val? = TRUE
```

```
cat("\nThus, we do not reject the NULL Hypothesis(H_0)")
```

```
##
## Thus, we do not reject the NULL Hypothesis(H_0)
```

```
# We conclude that it is in the region as  Tsq is less than F_val

axes_len = function(lam,n,p){
  sqrt(lam)* sqrt((p*(n-1))/ (n*(n-p)) * qf(0.95,df1=p, df2=(n-p)))
}

l1=axes_len(lam=eigenValVec$values[1],n=n,p=p)
l2=axes_len(lam=eigenValVec$values[2],n=n,p=p)

# plot(birddata)
# lines(ellipse(mu, S, npoints = 200))
# dataEllipse(birddata,levels=.95)
# length(c(mu$mu1,mu$mu2))
```

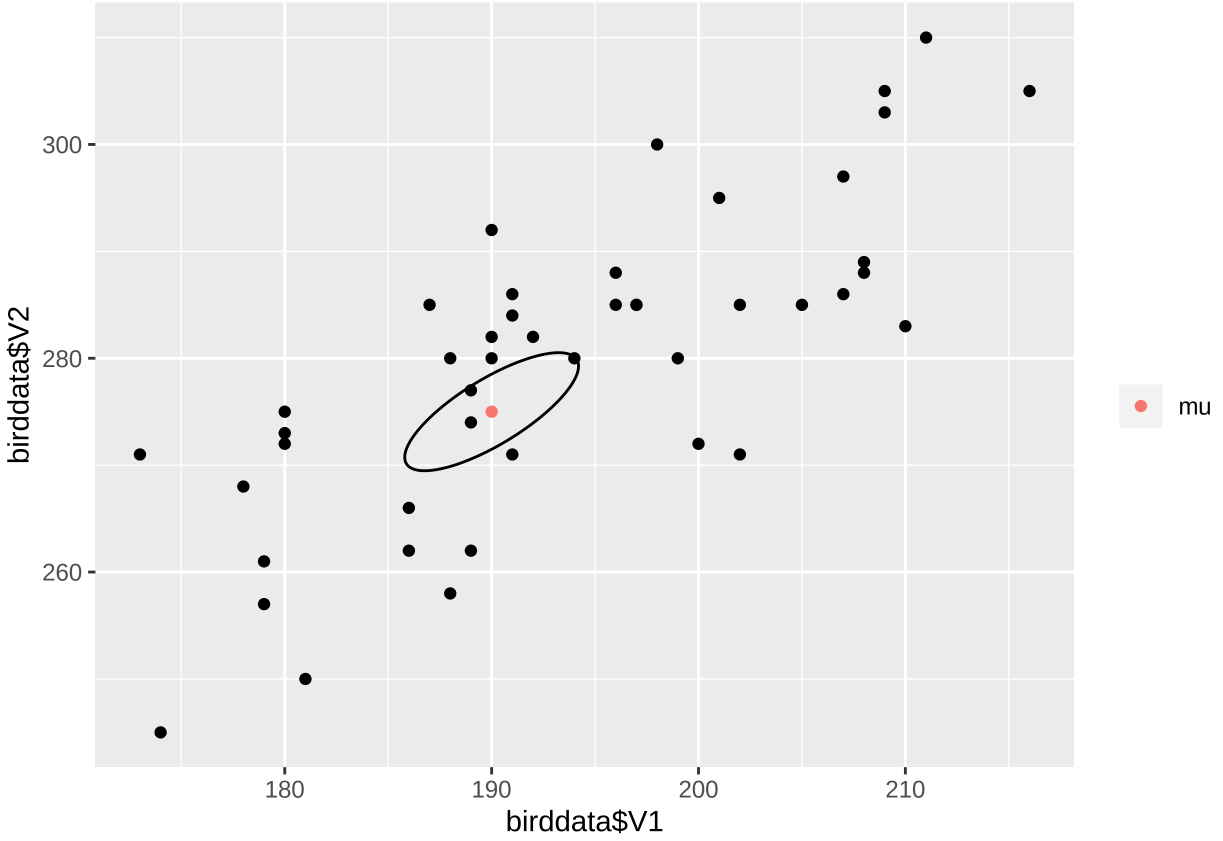
Are these plaussible values for the mean tail length and wing length for the female birds? **Answer** It is quite plaussible as female hook-billed vary a lot in size just like the male hook-billed. Sometimes the juveniles are bigger than the size than they are supposed to be. And thus the mean size of tail length and wing length could be same for the female hook-billed kites.

```
rotate = function(a,b,c){
  res = list()

  if (b==0 & a >= c){res$angle = 0}
  else if (b==0 & a<c) { res$angle = (pi/2)}
  else{ res$angle = atan2(eigenValVec$values[1]-a, b)}

  res$lam1 = (a+c)/2 + sqrt(((a-c)/2)^2 + b^2)
  res$lam2 = (a+c)/2 - sqrt(((a-c)/2)^2 + b^2)

  return(res)
}
# S = matrix(c(4,-2,-2,4),byrow = T, nrow = 2)
```

```r
res = rotate(S[1,1],S[1,2],S[2,2])

f95 = qf(0.95, p, n - p) * ((p * (n - 1)) / (n * (n - p)))
chi95<-qchisq(.95,df=2)
axes95 <- sqrt(eigenValVec$values) * sqrt(f95)


library(ggforce)
```

```
## Warning: package 'ggforce' was built under R version 3.5.2
```

```r
ggplot()+
  geom_point(aes(birddata$V1, birddata$V2))+
  geom_ellipse(aes(x0=mu$mu1,y0=mu$mu2, a=axes95[1], b=axes95[2], angle = res$angle )) +
  geom_point(aes(x=mu$mu1, y=mu$mu2, color="mu")) +
  labs(colour="")
```



We draw the same conclusion as before. The new values are plausible since they lie inside the ellipse (red dot).

2.2 Simultaneous T2-intervals

```r
# Simultaneous T2-intervals for the component means as shadows of the confidence ellipse on the axes

fsqrt = function(i){
  return(sqrt(p*(n-1)/ ((n-p)) * qf(0.95,df1=p, df=n-p)) * sqrt(S[i,i]/n))
}

CI = NULL
CI$Lower = (x_bar$X1) - fsqrt(1)
CI$Upper =(x_bar$X1) + fsqrt(1)
CI = as.data.frame(CI)
temp = c ((x_bar$X2) - fsqrt(2), (x_bar$X2) + fsqrt(2))
CI = rbind(CI, temp)
rownames(CI) = c("Mu_1_tail","Mu_2_wing")
# kableExtra::kable(CI)
CI
```

```
##               Lower    Upper
## Mu_1_tail 189.4217 197.8227
## Mu_2_wing 274.2564 285.2992
```

```
# Bonferroni's Confidence intervals
set.seed(12345)
ben_alpha = 0.95
Ber_CI = NULL
Ber_CI$Lower = x_bar$X1 - abs(qt(0.05/2*p, df=n-1))  * sqrt(S[1,1]/n)
Ber_CI$Upper= x_bar$X1 + abs(qt(0.05/2*p, df=n-1) ) * sqrt(S[1,1]/n)
Ber_CI = as.data.frame(Ber_CI)
temp = c(x_bar$X2 - abs(qt(0.05/2*p, df=n-1) ) * sqrt(S[2,2]/n),
         x_bar$X2 + abs(qt(0.05/2*p, df=n-1) ) * sqrt(S[2,2]/n))

Ber_CI = rbind(Ber_CI, temp)
rownames(Ber_CI) = c("Mu_1_tail", "Mu_2_wing")
Ber_CI
```

```
##               Lower    Upper
## Mu_1_tail 190.8705 196.3740
## Mu_2_wing 276.1607 283.3949
```

Tˆ2 simultaneous CI is slightly wider than Bonferroni Intervals. Bonferroni method provides shorter intervals when m = p. Because they are easy to apply and provide the relatively short confi- dence intervals needed for inference.

source: Applied multivariate Statistical Analysis - Pearson edition 2014

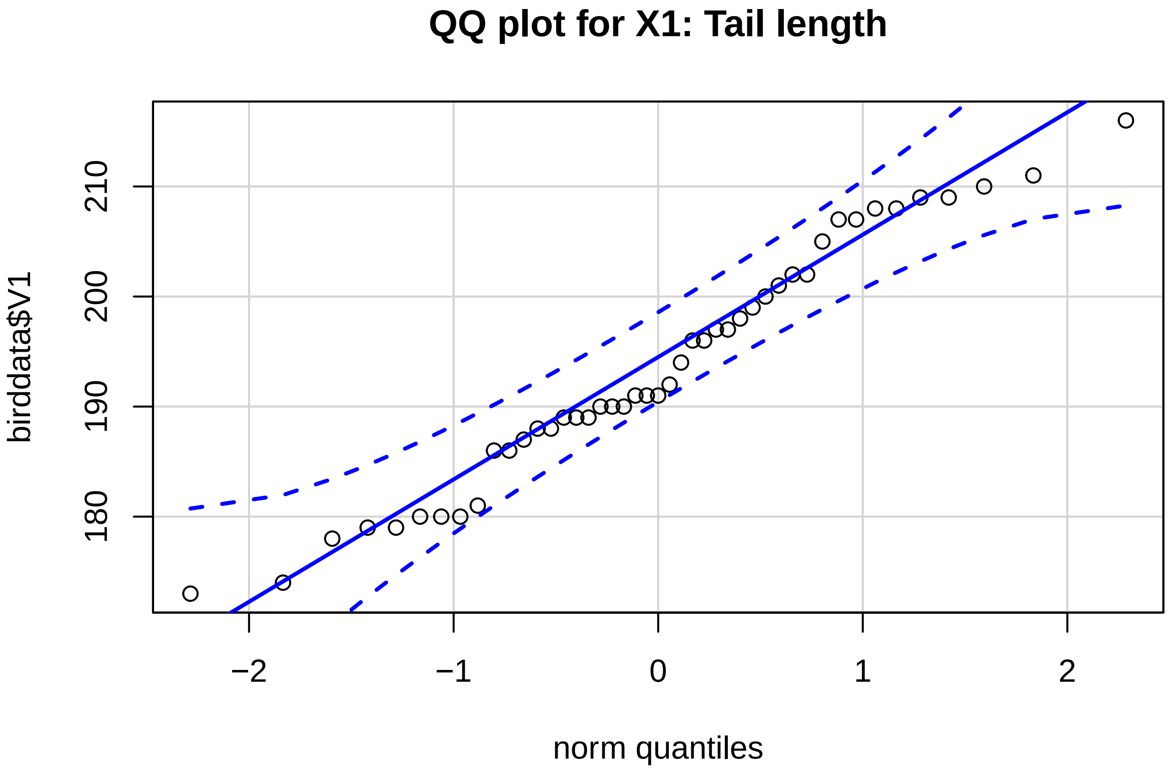
2.3 Q-Q plots and scatter plots

The data doesn't look normal from the QQ-plots.

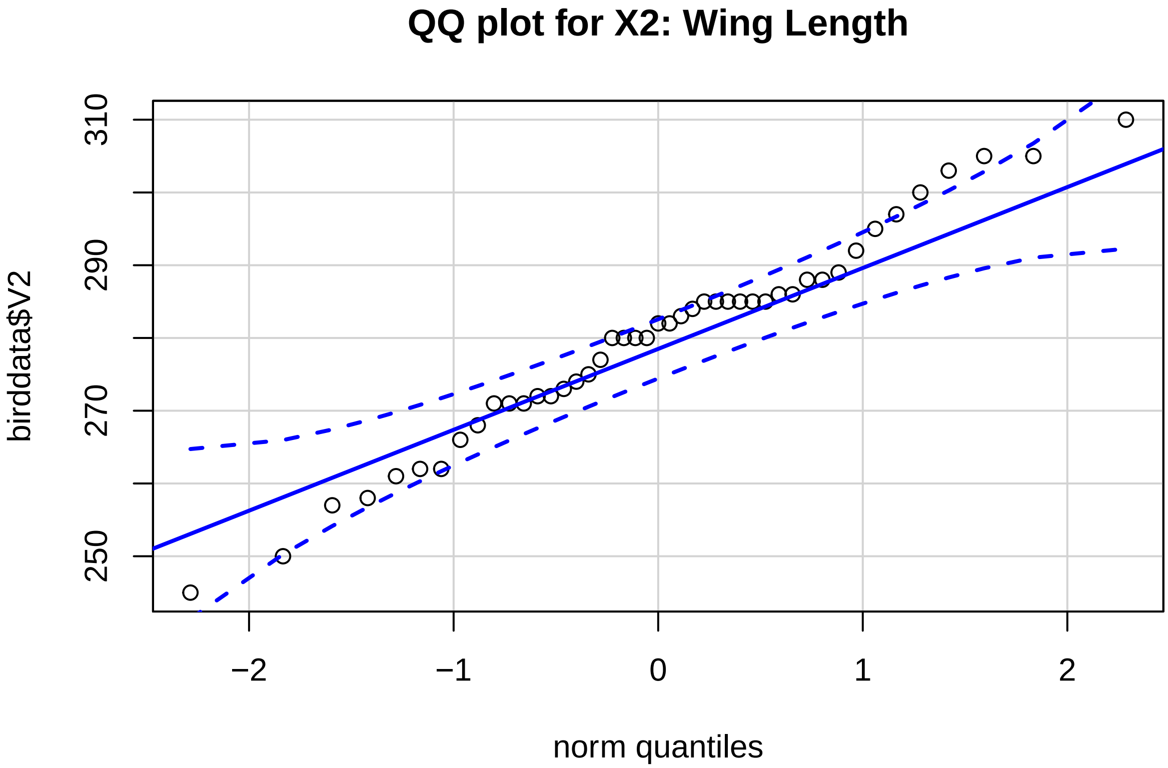
```
library(CARS)
```

```
## Warning: package 'CARS' was built under R version 3.5.2
```

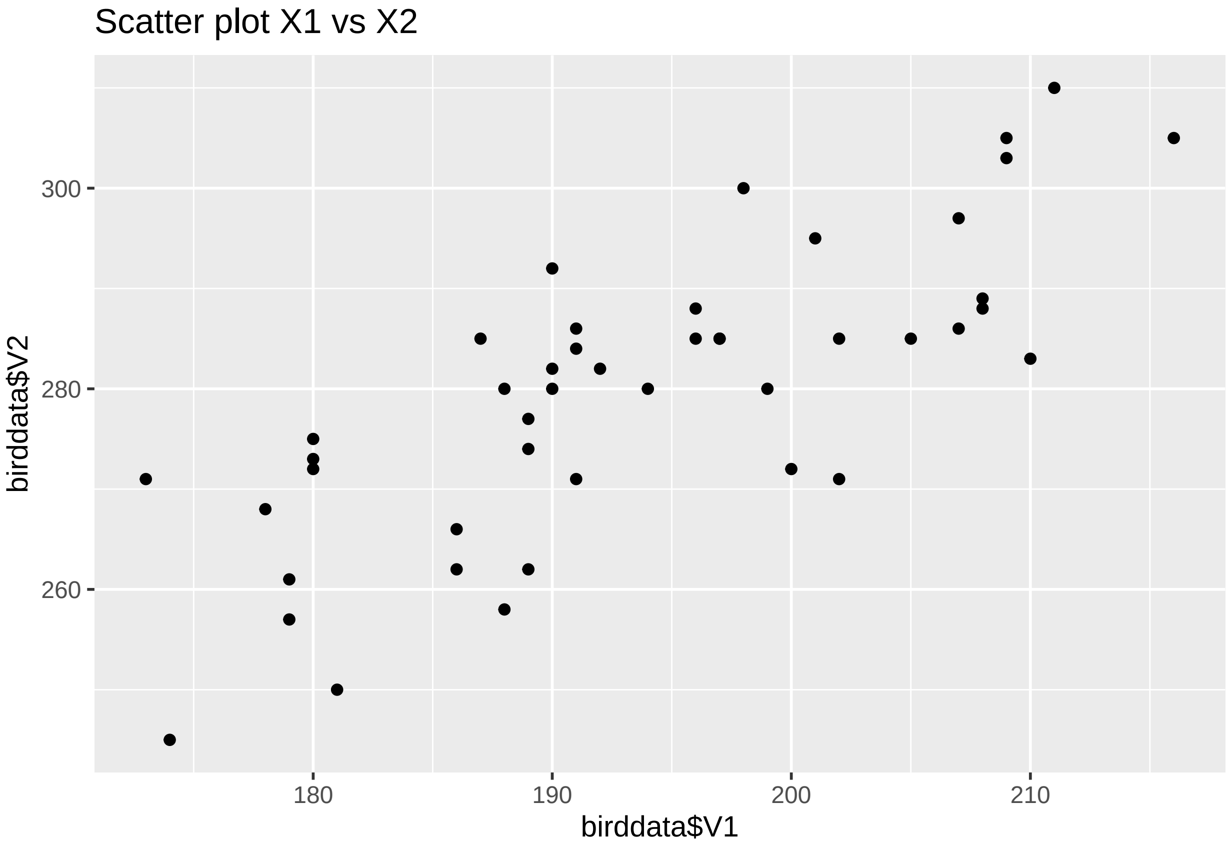
```
qqPlot(birddata$V1, main ="QQ plot for X1: Tail length",id=F  )
```



```
qqPlot(birddata$V2, main="QQ plot for X2: Wing Length",id=F )
```



```
Scat = ggplot()+
  geom_point(aes(x=birddata$V1,y=birddata$V2))+
  ggtitle(label = "Scatter plot X1 vs X2")

plot(Scat)
```



3 Comparison of Mean Vectors (one–way MANOVA)

We will look at a data set on Egyptian skull measurements (published in 1905 and now in `heplots` R package as the object `Skulls`). Here observations are made from five epochs and on each object the maximum breadth (mb), basibregmatic height (bh), basialiveolar length (bl) and nasal height (nh) were measured.

3.1 Exploring the Data

Explore the data first and present plots that you find informative.

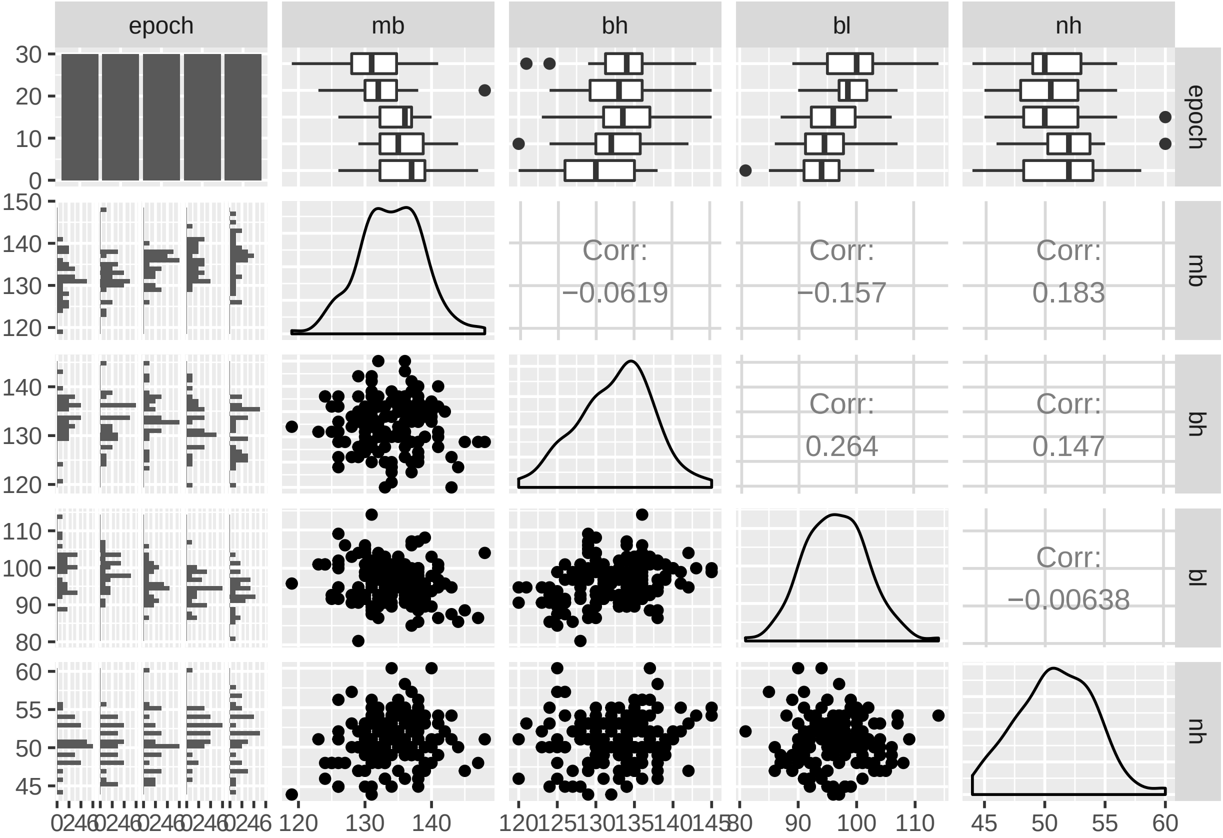
Answer: We first take a glimpse at the data.

```
head(Skulls)
```

```
##     epoch  mb  bh  bl nh
## 1 c4000BC 131 138  89 49
## 2 c4000BC 125 131  92 48
## 3 c4000BC 131 132  99 50
## 4 c4000BC 119 132  96 44
## 5 c4000BC 136 143 100 54
## 6 c4000BC 138 137  89 56
```

Next, we will look at the

```
ggpairs(Skulls)
```



Looking at the different distributions of our features we see that all of them are clustered around a mean. `mb` and `bl` seem to be symmetrical, whereas `bh` and `nh` seem not. Looking at the scatterplots and the corresponding correlations we seeh that aprt from `bh` with `mb` and `nh` and `bl` all of them have a slight correlation. We also see that we have the same amount of epochs, so they're "distributed" normally. The epochs are:

```
unique(Skulls$epoch)
```

```
## [1] c4000BC c3300BC c1850BC c200BC  cAD150
## Levels: c4000BC < c3300BC < c1850BC < c200BC < cAD150
```

3.2 Differing of Mean Vectors

Now we are interested whether there are differences between the epochs. Do the mean vectors differ? Study this question and justify your conclusions.

Task: The mean vectors do defer except nasal height. All other means are different between the epochs with a significante level between of 5 percent.

```
res = manova(cbind(mb, bh, bl, nh) ~ epoch, Skulls)
res
```

```
## Call:
```

```
##    manova(cbind(mb, bh, bl, nh) ~ epoch, Skulls)
##
## Terms:
##                    epoch Residuals
## resp 1           502.827  3061.067
## resp 2           229.907  3405.267
## resp 3           803.293  3505.967
## resp 4            61.200  1472.133
## Deg. of Freedom        4       145
##
## Residual standard errors: 4.59465 4.846091 4.917223 3.186321
## Estimated effects are balanced
```

```r
summary.aov(res)
```

```
##  Response mb :
##              Df  Sum Sq Mean Sq F value    Pr(>F)
## epoch         4  502.83 125.707  5.9546 0.0001826 ***
## Residuals   145 3061.07  21.111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response bh :
##              Df Sum Sq Mean Sq F value  Pr(>F)
## epoch         4  229.9  57.477  2.4474 0.04897 *
## Residuals   145 3405.3  23.485
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response bl :
##              Df Sum Sq Mean Sq F value    Pr(>F)
## epoch         4  803.3 200.823  8.3057 4.636e-06 ***
## Residuals   145 3506.0  24.179
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response nh :
##              Df Sum Sq Mean Sq F value Pr(>F)
## epoch         4   61.2  15.300   1.507 0.2032
## Residuals   145 1472.1  10.153
```

3.3 Confidence Intervals

If the means differ between epochs compute and report simultaneous confidence intervals. Inspect the residuals whether they have mean 0 and if they deviate from normality (graphically).

Tip: It might be helpful for you to read Exercise 6.24 of *Johnson, Wichern*. The function `manova()` can be useful for this question and the residuals can be found in the `$res` field.

Answer: The means differ, so we calculate the confidence intervals for all groups and feature combinations. A row in teh result (combination, feature, group A and group B) is signinficant with $\alpha = 0.05$ if the confidence interval does **not** cover 0.

```r
# k and l are the groups
# i is the feature
```

```r
ci_custom = function(data, p, g, n, W, l, k, i, alpha = 0.05) {

  # Define groups
  group_k = unique(data$epoch)[k]
  group_l = unique(data$epoch)[l]
  group_data_k = data[data$epoch == group_k,]
  group_data_l = data[data$epoch == group_l,]

  n_k = nrow(group_data_k)
  n_l = nrow(group_data_l)

  x_i = mean(group_data_k[,i+1]) - mean(group_data_l[,i+1])
  t_crit = qt(alpha/(p*g*(g-1)), df = (n-g))

  var_i = sqrt(diag(W)[i]/(n-g) * (1/n_k + 1/n_l))
  abs_i = abs(t_crit * var_i)

  res = c(x_i - abs_i, x_i + abs_i)

  return(res)
}

scite = function(data, i) {

  # Static Parameters
  p = ncol(data) - 1
  g = length(unique(Skulls$epoch))
  n = nrow(data)

  # Calculation of W

  W = 0

  for (group in unique(Skulls$epoch)) {

    group_data = Skulls[Skulls$epoch == group,]
    S = sample_variance(group_data[,2:5])
    W = W + (nrow(group_data) - 1) * S
  }

  df = data.frame()

  for (k in 2:length(unique(Skulls$epoch))) {
    for (l in 1:max((k-1), 1)) {
      row = c(i, k, l, ci_custom(data=data, p=p, g=g, n, W=W, l=l, k=k, i=i))
      df = rbind(df, row)
    }
  }

  colnames(df) = c("i", "k", "l", "lower", "upper")

  #res = ci_custom(data=data, p=p, g=g, n, W=W, l=1, k=3, i=2)

  return(df)
}
```

3.3.1 Variable i=1 (mb)

```
df = scite(Skulls, 1)
df
```

```
##    i k l       lower    upper
## 1  1 2 1 -2.83963613 4.839636
## 2  1 3 1 -0.73963613 6.939636
## 3  1 3 2 -1.73963613 5.939636
## 4  1 4 1  0.29369721 7.972969
## 5  1 4 2 -0.70630279 6.972969
## 6  1 4 3 -2.80630279 4.872969
## 7  1 5 1  0.96036387 8.639636
## 8  1 5 2 -0.03963613 7.639636
## 9  1 5 3 -2.13963613 5.539636
## 10 1 5 4 -3.17296946 4.506303
```

3.3.2 Variable i=2 (bh)

```
df = scite(Skulls, 2)
df
```

```
##    i k l     lower     upper
## 1  2 2 1 -4.949760 3.1497596
## 2  2 3 1 -3.849760 4.2497596
## 3  2 3 2 -2.949760 5.1497596
## 4  2 4 1 -5.349760 2.7497596
## 5  2 4 2 -4.449760 3.6497596
## 6  2 4 3 -5.549760 2.5497596
## 7  2 5 1 -7.316426 0.7830929
## 8  2 5 2 -6.416426 1.6830929
## 9  2 5 3 -7.516426 0.5830929
## 10 2 5 4 -6.016426 2.0830929
```

3.3.3 Variable i=3 (bl)

```
df = scite(Skulls, 3)
df
```

```
##    i k l     lower      upper
## 1  3 2 1 -4.209203  4.0092027
## 2  3 3 1 -7.242536  0.9758694
## 3  3 3 2 -7.142536  1.0758694
## 4  3 4 1 -8.742536 -0.5241306
## 5  3 4 2 -8.642536 -0.4241306
## 6  3 4 3 -5.609203  2.6092027
## 7  3 5 1 -9.775869 -1.5574640
## 8  3 5 2 -9.675869 -1.4574640
## 9  3 5 3 -6.642536  1.5758694
## 10 3 5 4 -5.142536  3.0758694
```

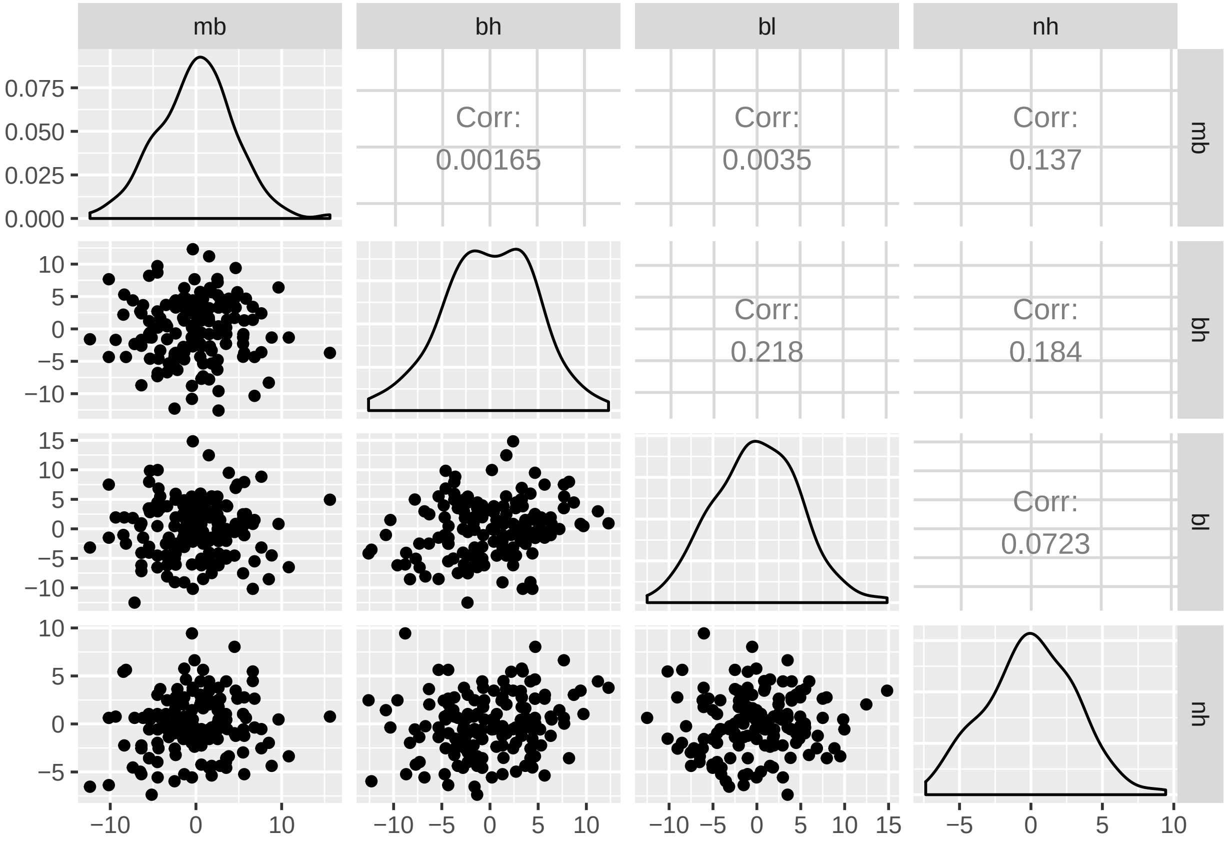
3.3.4 Variable i=4 (nh)

```
df = scite(Skulls, 4)
df
```

```
##    i k l      lower    upper
## 1  4 2 1 -2.9627307 2.362731
## 2  4 3 1 -2.6293974 2.696064
## 3  4 3 2 -2.3293974 2.996064
## 4  4 4 1 -1.2293974 4.096064
## 5  4 4 2 -0.9293974 4.396064
## 6  4 4 3 -1.2627307 4.062731
## 7  4 5 1 -1.8293974 3.496064
## 8  4 5 2 -1.5293974 3.796064
## 9  4 5 3 -1.8627307 3.462731
## 10 4 5 4 -3.2627307 2.062731
```

Looking at the residuals we think that most of them look (almost) normal.

```
ggpairs(data.frame(res$residuals))
```



4 Source Code

```r
library(viridis)
library(ggplot2)
library(heplots)
library(GGally)
knitr::opts_chunk$set(echo = TRUE)

track_times = read.table("data/T1-9.dat")
colnames(track_times) = c("country", "100m", "200m", "400m",
                   "800m", "1500m", "3000m", "marathon")
head(track_times)


sample_variance = function(X) {

  X = as.matrix(X)

  identity = diag(nrow(X))
  one_n = matrix(1, nrow=nrow(X), ncol=1)

  inter =  identity - 1/nrow(X) * (one_n %*% t(one_n))

  return(1/nrow(X) * (t(X) %*% inter %*% X))
}

mahalanobis_distance = function(X) {
  X = as.matrix(X)

  V = sample_variance(X)
  ident = matrix(1, nrow=nrow(X), ncol=nrow(X))
  mu = 1/nrow(X) * (t(ident) %*% X)

  X_centered = X - mu

  return(diag(X_centered %*% solve(V) %*% t(X_centered)))
}


nu = ncol(track_times) - 1


nu = ncol(track_times) - 1
D = mahalanobis_distance(track_times[,2:8])


val = seq(0, 40, 0.01)
chi_sq_7 = dchisq(val, nu)

ggplot() +
  geom_histogram(aes(x = D, y=..density..),
               color = "#755138", fill = "#D1CDC1",
               bins = sqrt(nrow(track_times))) +
  geom_density(aes(x = D, y=..density..),
               color="#5C8240", fill="#8AB077", alpha = 0.2) +
  geom_line(aes(x = val, y = chi_sq_7), color = "#2F2924") +
  labs(title = "Histrogram of 100m",
       y = "Density",
       x = "Time", color = "Legend") +
  scale_color_viridis(discrete=FALSE) +
  theme_minimal()


alpha = 0.001

outlier_indeces = 1 - pchisq(D, nu) < alpha

track_times$country[outlier_indeces]


alpha = 0.001 / nrow(track_times)

outlier_indeces = 1 - pchisq(D, nu) < alpha

track_times$country[outlier_indeces]

library(ellipse)
birddata = read.table("T5-12.DAT")
# Male hook-billed kites
n = nrow(birddata)
p = dim(birddata)[2]
mu = data.frame("mu1" = 190, "mu2" = 275)
meanVec =NULL
x_bar = NULL
meanMu = NULL
# x - x_bar
x_bar = data.frame("X1" = mean(birddata$V1), "X2" = mean(birddata$V2))

sample_var = function(obs){
  X = as.matrix(obs)
  IdentityMat = diag(nrow(obs))
  one_n = matrix(1,nrow(X),nrow(X))
  # sample var formula from notes
  mix = (IdentityMat - (1/nrow(X)) *(one_n))

  sample_variance = (1/(nrow(X) - 1)) * (t(X) %*% mix %*% X)
  return(sample_variance)
}

S = sample_var(birddata)
S =  matrix(S, ncol=2, nrow=2)
eigenValVec = eigen(S)
meanMu = as.matrix(x_bar - mu)

# To check if mu is on the confidence region we need to check if t^2 is < F_{n,n-p}
```

```
Tsq = n * (meanMu) %*% solve(S) %*% t(meanMu)

F_val =  qf(0.95,df1=p, df=n-p) * (p*(n-1)/(n-p)) #5% significance level

cat("Is T^2 less than F_val? =",Tsq<F_val)
cat("\nThus, we do not reject the NULL Hypothesis(H_0)")
# We conclude that it is in the region as  Tsq is less than F_val

axes_len = function(lam,n,p){
  sqrt(lam)* sqrt((p*(n-1))/ (n*(n-p)) * qf(0.95,df1=p, df2=(n-p)))
}

l1=axes_len(lam=eigenValVec$values[1],n=n,p=p)
l2=axes_len(lam=eigenValVec$values[2],n=n,p=p)

# plot(birddata)
# lines(ellipse(mu, S, npoints = 200))
# dataEllipse(birddata,levels=.95)
# length(c(mu$mu1,mu$mu2))



rotate = function(a,b,c){
  res = list()

  if (b==0 & a >= c){res$angle = 0}
  else if (b==0 & a<c) { res$angle = (pi/2)}
  else{ res$angle = atan2(eigenValVec$values[1]-a, b)}

  res$lam1 = (a+c)/2 + sqrt(((a-c)/2)^2 + b^2)
  res$lam2 = (a+c)/2 - sqrt(((a-c)/2)^2 + b^2)

  return(res)
}
# S = matrix(c(4,-2,-2,4),byrow = T, nrow = 2)
res = rotate(S[1,1],S[1,2],S[2,2])

f95 = qf(0.95, p, n - p) * ((p * (n - 1)) / (n * (n - p)))
chi95<-qchisq(.95,df=2)
axes95 <- sqrt(eigenValVec$values) * sqrt(f95)


library(ggforce)
ggplot()+
  geom_point(aes(birddata$V1, birddata$V2))+
  geom_ellipse(aes(x0=mu$mu1,y0=mu$mu2, a=axes95[1], b=axes95[2], angle = res$angle )) +
  geom_point(aes(x=mu$mu1, y=mu$mu2, color="mu")) +
  labs(colour="")

# Simultaneous T2-intervals for the component means as shadows of the confidence ellipse on the axes

fsqrt = function(i){
  return(sqrt(p*(n-1)/ ((n-p)) * qf(0.95,df1=p, df=n-p)) * sqrt(S[i,i]/n))
```

```
}

CI = NULL
CI$Lower = (x_bar$X1) - fsqrt(1)
CI$Upper =(x_bar$X1) + fsqrt(1)
CI = as.data.frame(CI)
temp = c ((x_bar$X2) - fsqrt(2), (x_bar$X2) + fsqrt(2))
CI = rbind(CI, temp)
rownames(CI) = c("Mu_1_tail","Mu_2_wing")
# kableExtra::kable(CI)
CI
# Bonferroni's Confidence intervals
set.seed(12345)
ben_alpha = 0.95
Ber_CI = NULL
Ber_CI$Lower = x_bar$X1 - abs(qt(0.05/2*p, df=n-1))  * sqrt(S[1,1]/n)
Ber_CI$Upper= x_bar$X1 + abs(qt(0.05/2*p, df=n-1) ) * sqrt(S[1,1]/n)
Ber_CI = as.data.frame(Ber_CI)
temp = c(x_bar$X2 - abs(qt(0.05/2*p, df=n-1) ) * sqrt(S[2,2]/n),
         x_bar$X2 + abs(qt(0.05/2*p, df=n-1) ) * sqrt(S[2,2]/n))

Ber_CI = rbind(Ber_CI, temp)
rownames(Ber_CI) = c("Mu_1_tail", "Mu_2_wing")
Ber_CI

library(CARS)

qqPlot(birddata$V1, main ="QQ plot for X1: Tail length",id=F  )
qqPlot(birddata$V2, main="QQ plot for X2: Wing Length",id=F )
Scat = ggplot()+
  geom_point(aes(x=birddata$V1,y=birddata$V2))+
  ggtitle(label = "Scatter plot X1 vs X2")

plot(Scat)

head(Skulls)
ggpairs(Skulls)
unique(Skulls$epoch)

res = manova(cbind(mb, bh, bl, nh) ~ epoch, Skulls)
res


summary.aov(res)


# k and l are the groups
# i is the feature

ci_custom = function(data, p, g, n, W, l, k, i, alpha = 0.05) {

  # Define groups
  group_k = unique(data$epoch)[k]
  group_l = unique(data$epoch)[l]
  group_data_k = data[data$epoch == group_k,]
  group_data_l = data[data$epoch == group_l,]

  n_k = nrow(group_data_k)
  n_l = nrow(group_data_l)

  x_i = mean(group_data_k[,i+1]) - mean(group_data_l[,i+1])
  t_crit = qt(alpha/(p*g*(g-1)), df = (n-g))

  var_i = sqrt(diag(W)[i]/(n-g) * (1/n_k + 1/n_l))
  abs_i = abs(t_crit * var_i)

  res = c(x_i - abs_i, x_i + abs_i)

  return(res)
}

scite = function(data, i) {

  # Static Parameters
  p = ncol(data) - 1
  g = length(unique(Skulls$epoch))
  n = nrow(data)

  # Calculation of W

  W = 0

  for (group in unique(Skulls$epoch)) {

    group_data = Skulls[Skulls$epoch == group,]
    S = sample_variance(group_data[,2:5])
    W = W + (nrow(group_data) - 1) * S
  }

  df = data.frame()

  for (k in 2:length(unique(Skulls$epoch))) {
    for (l in 1:max((k-1), 1)) {
      row = c(i, k, l, ci_custom(data=data, p=p, g=g, n, W=W, l=l, k=k, i=i))
      df = rbind(df, row)
    }
  }

  colnames(df) = c("i", "k", "l", "lower", "upper")

  #res = ci_custom(data=data, p=p, g=g, n, W=W, l=1, k=3, i=2)

  return(df)
}
```

```r
df = scite(Skulls, 1)
df


df = scite(Skulls, 2)
df


df = scite(Skulls, 3)
df


df = scite(Skulls, 4)
df

ggpairs(data.frame(res$residuals))
```