Tutorial - Week 5

Please read the related material and attempt these questions before attending your allocated tutorial. Solutions are released on Friday 4pm.

Question 1

We continue the analysis started in last week's tutorial where we were looking at the properties of the Hotelling T^2 statistic and testing means. First we are interested to understand what happens as p becomes large and how it compares to some other alternative methods.

(a) One of the problems when using the T^2 test statistic is that, as p becomes larger compared to the sample size n, the sample covariance matrix $\mathbb{S}$ becomes increasingly singular. A singular matrix is one that does not have an inverse and this causes problems as the T^2 test statistic contains the expression $\mathbb{S}^{-1}$. As a matrix is singular if and only if its determinant is zero, we can study $\det(\mathbb{S})$ as $p \to n$ to understand how quickly $\mathbb{S}$ becomes singular as p increases.
Perform a simulation experiment to show the behaviour of $\det(\mathbb{S})$ as $p \to n = 100$. That is, write a function that, given values of p and $n = 100$, will sample n observations from a multivariate normal distribution $N_p(0, I_p)$, calculate the sample covariance $\mathbb{S}$ of the data, then calculate the determinant of the sample covariance matrix $\det(\mathbb{S})$. The function should perform this simulation `nsims` number of times for a given p, calculate the mean and standard error of the simulation.

Solution: We use the library `mvnfast` to do fast multivariate normal sampling, using other libraries is fine too.

```
library(mvnfast)
```

We write a function that given values of p and $n = 100$ will sample data from a multivariate normal distribution $N_p(0, I_p)$, calculate the sample covariance of the data, then calculate the determinant of the sample covariance matrix. The function does this simulation `nsims` number of times and calculates the mean and standard error of the simulation.

```
det.sample.covar = Vectorize(function(p, n=100, nsims=30) {
  mu = rep(0, p)
  Sigma = diag(rep(1,p))
  dets = replicate(nsims,{
    S = cov(rmvn(n, mu, Sigma))
    det(S)
  })
  c(mean(dets), sqrt(var(dets)/nsims)) # mean and standard error
}, "p")
```

We test our function for a list of $p \in (3, 4)$.

```
det.sample.covar(c(3,4))
```

```
##           [,1]      [,2]
## [1,] 0.98512514 0.83515330
```

```
## [2,] 0.03979757 0.04729509
```

(b) Illustrate the behaviour of $\det(\mathbb{S})$ for $2 < p < n = 100$ by using an error bar plot. You can use the function `errbar` in the package `Hmisc` to do this.

Solution: As an optional step, I write a function that generates a non-uniform sequence of points to sample $2 \leq p \leq 100$, choosing more values near 1 and near 100. This is because the behavior of `det(S)` is the most interesting near these values.

```
nseq = function(from=1, to=100, length.out=20, s=1.5) {
  x = (2 * seq(2, length.out, length.out=length.out)-length.out-1)/(length.out-1)
  as.integer((1+tanh(s*x)/tanh(s))*0.5*(to-from+1)+(from-1))
}
```

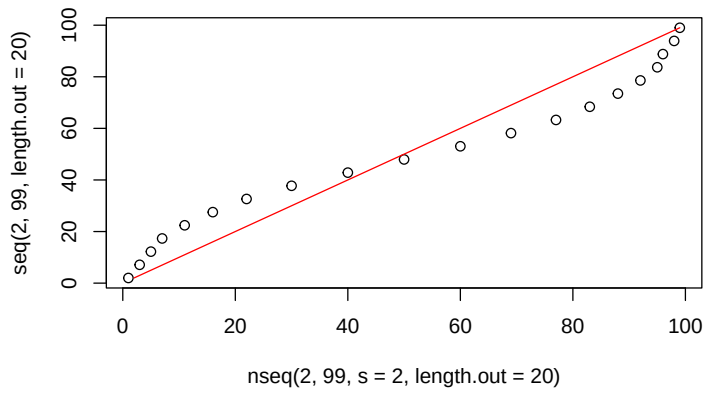
We test the function.

```
nseq(2, 99)
```

```
##  [1]  2  4  7 11 15 21 27 34 42 50 58 65 72 79 84 88 92 95 97 99
```

We can see the number sampling is not uniform. A uniform sampling would give a straight line. The parameter s can be changed to increase or decrease the intensity of the sampling.

```
plot(nseq(2, 99, s=2, length.out=20), seq(2, 99, length.out=20))
lines(seq(2, 99, length.out=20), seq(2, 99, length.out=20), col='red')
```



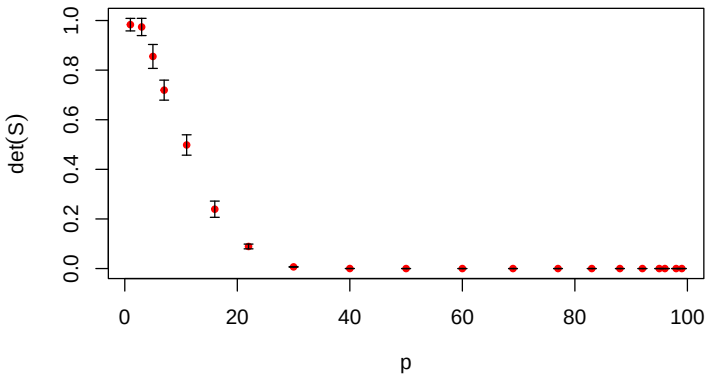
We use the package `Hmisc` for the function `errbar` the creates an error bar plot.

```
#install.packages("Hmisc")
library(Hmisc)
```

```
##
## Attaching package: 'Hmisc'
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

We now plot the sample determinants as p increases using error bars to show the mean and standard error. We see that det($\mathbb{S}$) decays quite quickly to zero. This means that $\mathbb{S}$ becomes quite quickly non-invertible for larger p.

```
ps = nseq(2, 99, s=2)
vs = det.sample.covar(ps, nsims=30)

errbar(ps, vs[1,], vs[1,]+vs[2,], vs[1,]-vs[2,], ylab=expression(det(S)), xlab=expression(p), pch=20, col=
```



(c) Now that we have identified there could be a problem with $\mathbb{S}^{-1}$ as p becomes large, let's consider what happens to the Hotelling T^2 test. First, similar to last week's tutorial, write a function that calculates the power of a Hotelling T^2 test. Take the case $n_1 = n_2 = 50$ with $\mu_2 - \mu_1 = (\delta, \ldots, \delta)'$ with varying $0 \leq \delta \leq 0.5$. However, this time assume that the (population) covariance has an "AR(1) structure" given by

$$\Sigma = \begin{pmatrix} 1 & \rho & \cdots & \rho^p \\ \rho & 1 & & \vdots \\ \vdots & & \ddots & \vdots \\ \rho^p & \rho^{p-1} & \cdots & 1 \end{pmatrix}, \qquad \text{(AR1)}$$

for $\rho = 0.5$. This can be generated succintly in R using the code

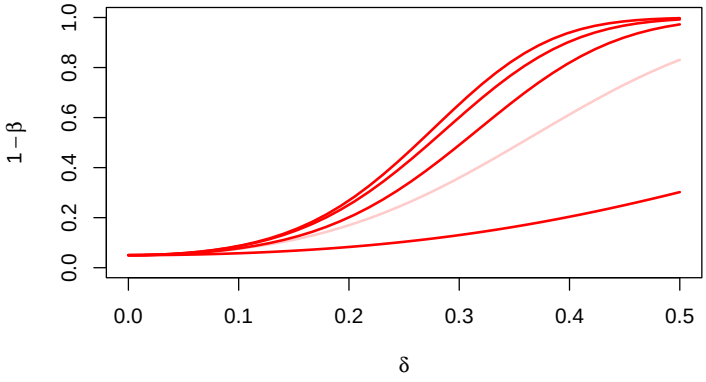
```
rho = 0.5
Sigma = rho^abs(outer(1:p, 1:p, "-"))
```

Plot the power curve for $0 \leq \delta \leq 0.5$, for the cases $p \in \{5, 25, 50, 75, 95\}$. What do you observe?

Solution:

```
power.hotellingT2 = Vectorize(function(delta, n1, n2, Sigma, alpha=0.05) {
  p = nrow(Sigma)
  d = rep(delta, p)
  Delta = (n1 * n2) / (n1 + n2) * t(d) %*% solve(Sigma) %*% d
  x = qf(1 - alpha, p, n1 + n2 - p - 1)
  1 - pf(x, p, n1 + n2 - p - 1, Delta)
}, "delta")
```

```
n1 = n2 = 50
n = n1 + n2

rho = 0.5
alpha = 0.05
deltas = seq(0, 0.5, length.out=40)

# do plot without plotting to set ranges of plot and labels
p = 5
Sigma = rho^abs(outer(1:p, 1:p, "-")) # AR1 covariance
maxdet = det(Sigma)

plot(deltas, power.hotellingT2(deltas, n1, n2, Sigma), type='n',
     ylab=expression(1-beta), xlab=expression(delta),
     ylim=c(0,1))

col = function(x, h=1) hsv(h, s=0.2+0.8*(1-det(x)/maxdet))

p = 5
Sigma = rho^abs(outer(1:p, 1:p, "-")) # AR1 covariance
lines(deltas, power.hotellingT2(deltas, n1, n2, Sigma), type='l', ylab=NA, lwd=2, col=col(Sigma))

p = 25
Sigma = rho^abs(outer(1:p, 1:p, "-")) # AR1 covariance
lines(deltas, power.hotellingT2(deltas, n1, n2, Sigma), type='l', ylab=NA, lwd=2, col=col(Sigma))

p = 50
Sigma = rho^abs(outer(1:p, 1:p, "-")) # AR1 covariance
lines(deltas, power.hotellingT2(deltas, n1, n2, Sigma), type='l', ylab=NA, lwd=2, col=col(Sigma))

p = 75
Sigma = rho^abs(outer(1:p, 1:p, "-")) # AR1 covariance
lines(deltas, power.hotellingT2(deltas, n1, n2, Sigma), type='l', ylab=NA, lwd=2, col=col(Sigma))
p = 95
Sigma = rho^abs(outer(1:p, 1:p, "-")) # AR1 covariance
lines(deltas, power.hotellingT2(deltas, n1, n2, Sigma), type='l', ylab=NA, lwd=2, col=col(Sigma))
```



(d) Consider the one-sample version of Hotelling's T^2 test that uses the χ^2 distribution for testing

$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad \mu \neq \mu_0,$$

where the test statistic is given by $T^2 = n(\bar{\mathbb{x}} - \mu_0)'\mathbb{S}^{-1}(\bar{\mathbb{x}} - \mu_0)$ and the distribution of $T^2 \sim \chi^2_p$ where n is the number of p-dimensional observations $\mathbb{x}_1, \ldots, \mathbb{x}_n$ with sample mean $\bar{\mathbb{x}}$; see **[A]**[1]. Take the case $\mu_0 = (0, 0, \ldots, 0)'$. Write a function to perform this test for a given data matrix $\mathbb{X}$.

Solution: We implement the one-sample version of Hotelling using the χ-squared distribution.

```
hotelling.onesample = function(x, mu0=rep(0,ncol(x))) {
  n = nrow(x)
  p = ncol(x)
  bar.x = colMeans(x)
  S = cov(x)
  S.inv = solve(S)
  T2 = n * t(bar.x - mu0) %*% S.inv %*% (bar.x - mu0)
  pchisq(q = T2, df = p, lower.tail = FALSE)
}
```

(e) Use you one-sample version of Hotelling's T^2 test and perform a simulation study to understand whether the *size* of the test achieves the advertised α. Consider this for p varying between 2 and 50 and $\alpha = 0.05$. What does this show?

Solution: We evaluate whether the "size" of the test achieves the advertised α. To do this, we generate data with $\theta \in \Theta_0$ and then calculate the proportion of rejections of the null hypothesis $\theta \in \Theta_0$. In our case,

$$H_0 : \theta \in \Theta_0 \quad \Leftrightarrow \quad H_0 : \mu = \mu_0 \text{ where } \mu_0 = (0, 0, \ldots, 0)^T.$$

[1] See the "Motivation" section at **[A]**.

We should expect that the proportion is $\approx \alpha$ for our choice of $\alpha = 0.05$.

```
library(mvnfast)
```

We write a function to perform this simulation experiment.

```
sim.size.hotelling = function(p, n=100, alpha=0.05, rho=0.1, nsims=5000) {
  mu = rep(0, p)
  Sigma = rho ^ abs(outer(1:p, 1:p, "-")) # AR1 covariance
  p.values = replicate(nsims, {
    X = rmvn(n, mu, Sigma)
    hotelling.onesample(X, mu)
  })
  mean(p.values < alpha) # proportion of observed Type I errors
}
```

For $p = 2$, this seems to be working well.

```
sim.size.hotelling(2, n=1000, rho=0.05, nsims=10000)
```

```
## [1] 0.0523
```

We now write a parallel version.

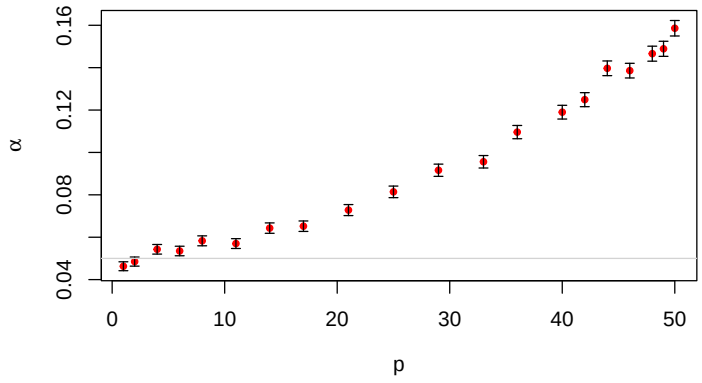
```
library(future.apply)
```

```
## Loading required package: future
```

```
psim.size.hotelling = Vectorize(function(p, n=500, alpha=0.05, rho=0.0, nsims=10000) {
  mu = rep(0, p)
  Sigma = rho ^ abs(outer(1:p, 1:p, "-")) # AR1 covariance
  p.values = future_replicate(nsims, {
    X = rmvn(n, mu, Sigma)
    hotelling.onesample(X, mu)
  })
  below = p.values <= alpha
  c(mean(below), sqrt(var(below)/nsims)) # mean and standard error
}, "p")
```

And we use it to generate a plot.

```
plan(multisession, workers = 36)

alpha = 0.05
ps = nseq(2, 50)
vs = psim.size.hotelling(ps, alpha=alpha)

errbar(ps, vs[1,], vs[1,]+vs[2,], vs[1,]-vs[2,], ylab=expression(alpha), xlab=expression(p), pch=20, col='
par(new=TRUE)
abline(h=alpha, col='lightgray')
```



We see that the size quite quickly exceeds the advertised α.

Question 2

Sometimes we might want to test whether two samples have the same amount of variation in the data. When your data is univariate (i.e., $p = 1$), this leads to a hypothesis testing problem concerning variances. That is, you have a random sample $Y_1, Y_2, \ldots, Y_n$ from a (univariate) normal distribution with unknown mean μ and unknown variance σ^2. Then, you are interesting in testing $H_0 : \sigma^2 = \sigma_0^2$ for some fixed value σ_0^2 versus the alternative hypothesis. This was generalised to multivariate data and the two-sample case by Box **[B]**, where the hypotheses become

$$H_0 : \Sigma_1 = \Sigma_2 \quad \text{vs.} \quad H_1 : \Sigma_1 \neq \Sigma_2,$$

for two p-dimensional samples of size n_1 and n_2, respectively. To perform a test that seperates these two hypotheses, we calculate a statistic called *Box's M-test* which is given by

$$M := \frac{|\mathbb{S}_1|^{N_1/2}|\mathbb{S}_2|^{N_2/2}}{|\mathbb{S}_{\text{pl}}|^{N/2}},$$

where for $i \in \{1, 2\}$, $N_i = n_i - 1$, $\mathbb{S}_i$ is the sample covariance matrix of the ith sample, and $\mathbb{S}_{\text{pl}}$ is the pooled sample covariance matrix given by

$$\mathbb{S}_{\text{pl}} = \frac{N_1\mathbb{S}_1 + N_2\mathbb{S}_2}{N},$$

where $N := N_1 + N_2 = (n_1 + n_2 - 2)$. Box gave a χ^2-approximation for the distribution of M which first requires calculating

$$c_1 := \left[\frac{1}{N_1} + \frac{1}{N_2} - \frac{1}{N}\right]\left[\frac{2p^2 + 2p - 1}{6(p+1)}\right],$$

then $u = -2(1 - c_1)\log(M)$ is approximately $\chi^2_{\frac{1}{2}p(p+1)}$-distributed. The null hypothesis H_0 is then rejected if $u > \chi^2_\alpha$ for your desired size α (e.g., $\alpha = 0.05$).

(a) Take $\Sigma_1 = \Sigma_2 = I_p$, $n_1 = 250$, $n_2 = 250$, $p = 3$, nsims=5000 and simulate the distribution of $-2(1 - c_1)\log(M)$ under the assumption that sample 1 is generated from $N_p(0, \Sigma_1)$ and sample 2 from $N_p(0, \Sigma_2)$. As the logarithm of a determinant may be numerically unstable, you may use the following R function to compute $\log(\det(x))$:

```
logdet = function(x) as.numeric(determinant(x, logarithm=TRUE)$modulus)
```

Compare the empirical distribution obtained from your simulation to the χ^2 distribution.

Solution:

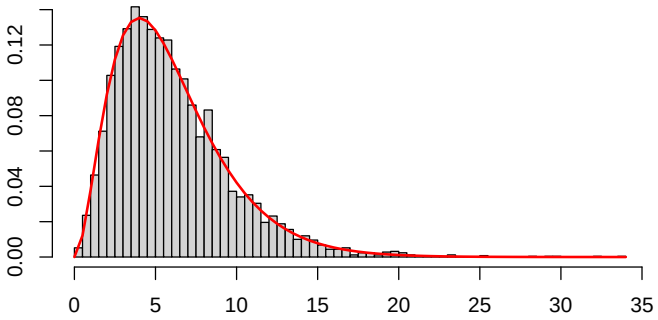
```
library(mvnfast)
library(future.apply)
```

```
logdet = function(x) as.numeric(determinant(x, logarithm=TRUE)$modulus)
```

```
psim.logM = function(Sigma1, Sigma2, n1=250, n2=250, nsims=5000) {
  p = ncol(Sigma1)
  n = n1 + n2
  mu = rep(0, p)
  future_replicate(nsims, {
    X1 = rmvn(n1, mu, Sigma1)
    S1 = cov(X1)
    X2 = rmvn(n2, mu, Sigma2)
    S2 = cov(X2)
    Spl = ((n1 - 1) * S1 + (n2 - 1) * S2) / (n - 2)
    (n1-1)/2 * (logdet(S1) - logdet(Spl)) + (n2-1)/2 * (logdet(S2) - logdet(Spl))
  })
}
```

```
plan(multisession, workers = 10)

p = 3
n1 = 250
n2 = 250
n = n1+n2

rho = 0.0
Sigma1 = rho ^ abs(outer(1:p, 1:p, "-")) # AR1 covariance

rho = 0.0
Sigma2 = rho ^ abs(outer(1:p, 1:p, "-")) # AR1 covariance

values = psim.logM(Sigma1, Sigma2)

c1 = (1 / (n1 - 1) + 1 / (n2 - 1) + 1 / (n - 2)) * ((2 * p ^ 2 + 3 * p - 1) / (6 * (p + 1)))
B = -2*(1-c1)*values
```

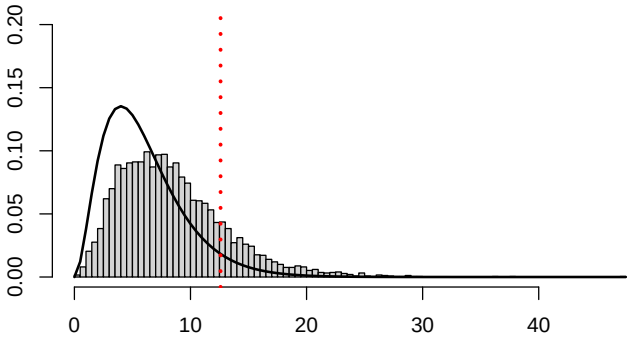
```
hh = hist(B, "FD", probability = TRUE, xlab=NA, ylab=NA, main=NA)
x = hh$breaks
lines(x, dchisq(x, df=0.5*p*(p+1)), lwd=2, col="red")
```



(b) Now consider what happens to the distribution if you redo the simulation with Σ_2 having the form of (AR1) with $\rho = 0.1$. What does this mean in relation to the hypothesis testing problem?

Solution: If ρ in Σ_2 increases slightly so that $\Sigma_1 \neq \Sigma_2$, we see mass in the density shifting to the right. This puts more weight in the tail of the distribution and we can use this to reject H_0 by checking if the Box M-statistic $B > \chi^2_\alpha$ where α is our size.

```
alpha = 0.05

p = 3
n1 = 250
n2 = 250
n = n1+n2

rho = 0.0
Sigma1 = rho ^ abs(outer(1:p, 1:p, "-")) # AR1 covariance

rho = 0.1
Sigma2 = rho ^ abs(outer(1:p, 1:p, "-")) # AR1 covariance

values = psim.logM(Sigma1, Sigma2)

c1 = (1 / (n1 - 1) + 1 / (n2 - 1) + 1 / (n - 2)) * ((2 * p ^ 2 + 3 * p - 1) / (6 * (p + 1)))
B = -2*(1-c1)*values # Box M statistics

hh = hist(B, 100, probability = TRUE, xlab=NA, ylab=NA, main=NA, ylim=c(0,1.4*max(hh$density)))
x = hh$breaks
lines(x, dchisq(x, df=0.5*p*(p+1)), lwd=2, col="black")

abline(v=qchisq(alpha, df=0.5*p*(p+1), lower.tail=FALSE), col="red", lwd=3, lty=3)
```



(c) Consider the `tibetskull` dataset from last week and perform a hypothesis testing problem to determine if the two samples have different (population) covariances.

Solution:

```
source('tibetskull.dat')

Sample = as.numeric(Tibet[,"Type"])
Tibet = Tibet[,!(colnames(Tibet) %in% c('Type'))]
```

We start by breaking up the `Tibet` data into two samples (based on `Sample`).

```
Tibet1 = Tibet[Sample==1,]
Tibet2 = Tibet[Sample==2,]
```

We extract the dimensions.

```
n1 = nrow(Tibet1)
n2 = nrow(Tibet2)
n  = n1+n2
p  = ncol(Tibet1)
```

```
S1=var(Tibet1)
S1
```

```
##            Length   Breadth   Height   Fheight Fbreadth
## Length   45.52941 25.222426 12.39062 22.154412 27.97243
## Breadth  25.22243 57.805147 11.87500  7.519301 48.05515
## Height   12.39062 11.875000 36.09375 -0.312500  1.40625
## Fheight  22.15441  7.519301 -0.31250 20.935662 16.76930
## Fbreadth 27.97243 48.055147  1.40625 16.769301 66.21140
```

```
S2=var(Tibet2)
S2
```

```
##             Length     Breadth     Height    Fheight  Fbreadth
## Length   74.423810  -9.5226190  22.736905 17.7940476 11.125000
## Breadth  -9.522619  37.3523810 -11.263095  0.7047619  9.464286
## Height   22.736905 -11.2630952  36.316667 10.7238095  7.196429
## Fheight  17.794048   0.7047619  10.723810 15.3023810  8.660714
## Fbreadth 11.125000   9.4642857   7.196429  8.6607143 17.964286
```

```
Spl = ((n1 - 1) * cov(Tibet1) + (n2 - 1) * cov(Tibet2)) / (n - 2)
Spl
```

```
##             Length   Breadth    Height   Fheight  Fbreadth
## Length   59.013464  9.008072 17.218889 20.119575 20.110294
## Breadth   9.008072 48.260523  1.077222  4.339183 30.046078
## Height   17.218889  1.077222 36.197778  4.837778  4.108333
## Fheight  20.119575  4.339183  4.837778 18.306797 12.985294
## Fbreadth 20.110294 30.046078  4.108333 12.985294 43.696078
```

```
logM = (n1-1)/2 * (logdet(S1) - logdet(Spl)) + (n2-1)/2 * (logdet(S2) - logdet(Spl))
logM
```

```
## [1] -11.18565
```

```
c1 = (1 / (n1 - 1) + 1 / (n2 - 1) + 1 / (n - 2)) * ((2 * p ^ 2 + 3 * p - 1) / (6 * (p + 1)))
B = -2*(1-c1)*logM # Box M statistics
B
```

```
## [1] 15.7191
```

```
pchisq(B, df=0.5*p*(p+1), lower.tail=FALSE)
```

```
## [1] 0.4009622
```

The probability is not below α, we cannot reject the null hypothesis.

References

[A] https://en.wikipedia.org/wiki/Hotelling%27s_T-squared_distribution

[B] Box (1949). A General Distribution Theory for a Class of Likelihood Criteria. Biometrica 36 (3-4), 317 – 346.