STAT636 - Homework 3

Daniel Osorio - dcosorioh@tamu.edu
Department of Veterinary Integrative Biosciences
Texas A&M University

1. Consider the `Auto` data. These represent a two-factor experiment on cars. The two factors are (i) the number of cylinders (4 and 6 were considered) and (ii) origin (three origins were considered). We have 4 cars under each of the $2 \times 3 = 6$ factor combinations. For each car, we have measurements of three weight variables: X_1 = displacement, X_2 = horsepower, and X_3 = acceleration. So, in terms of a two-way MANOVA model, $g = 2$, $b = 3$, and $n = 4$.

```
> auto<- read.csv("Auto_hw.csv")
> auto$cylinders <- as.factor(auto$cylinders)
> auto$origin <- as.factor(auto$origin)
```

(a) Test for a location effect, a variety effect, and a location-variety interaction at $\alpha = 0.05$. Do this using the manova function in R. Overall, what do you conclude about these data?

```
> manovaResult <- manova(cbind(displacement,horsepower,acceleration) ~
+                           origin + cylinders + origin*cylinders,auto)
> manovaResult
Call:
   manova(cbind(displacement, horsepower, acceleration) ~ origin +
    cylinders + origin * cylinders, auto)

Terms:
                  origin cylinders origin:cylinders Residuals
resp 1           9843.25  34808.17          5766.58   4692.00
resp 2            787.00   4082.04          1057.33   6857.25
resp 3             23.24      5.04             5.74    105.91
Deg. of Freedom        2         1                2        18

Residual standard errors: 16.14517 19.51815 2.425673
Estimated effects may be unbalanced
> summary(manovaResult, test = "Wilks")
                 Df   Wilks approx F num Df den Df    Pr(>F)
origin            2 0.20304    6.503      6     32 0.0001468 ***
cylinders         1 0.11520   40.961      3     16 9.813e-08 ***
origin:cylinders  2 0.25986    5.129      6     32 0.0008635 ***
Residuals        18
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

(b) Construct the two-way MANOVA table by computing SSP_{FAC1}, SSP_{FAC2}, SSP_{INT}, SSP_{RES}, and SSP_{COR}. Provide R code that matches the Wilks' statistics computed by manova. Note that your p-values (computed according to the notes) will not match those of manova, because the distributional results we have learned for two-way MANOVA are large-sample approximations. That said, how do your p-values compare to those of manova?

```
> n <- 4
> p <- 3
> g <- 2
> b <- 3
> xBar <- apply(auto[,2:4],2,mean)
> SSPfac1 <- SSPfac2 <- SSPint <- SSPres <- SSPcor <- 0
> for (l in unique(auto$cylinders)){
+   xBar_l <- apply(auto[auto[,1] == l,2:4],2,mean)
+   SSPfac1 <- SSPfac1 + ((b * n * (xBar_l-xBar)^2))
+   for(k in unique(auto$origin)){
+     xBar_k <- apply(auto[auto[,5] == k,2:4],2,mean)
+     xBar_lk <- apply(auto[auto[,1] == l & auto[,5] == k,2:4],2,mean)
+     SSPint <- SSPint + (n * ((xBar_lk - xBar_l - xBar_k + xBar)^2))
+     for(r in seq_len(n)){
+       x <- auto[auto[,1] == l & auto[,5] == k & seq_len(n) == r,2:4]
+       SSPres <- SSPres + ((x - xBar_lk)^2)
+       SSPcor <- SSPcor + ((x - xBar)^2)
+     }
+   }
+ }
> for(k in unique(auto$origin)){
+     xBar_k <- apply(auto[auto[,5] == k,2:4],2,mean)
+     SSPfac2 <- SSPfac2 + (g * n * ((xBar_k - xBar)^2))
+ }
> manovaTable <- cbind(SSPfac1,SSPfac2, SSPint,
+                      t(SSPres), t(SSPcor))
> colnames(manovaTable) <- c("SSP_fac1", "SSP_fac2", "SSP_int",
+                            "SSP_res", "SSP_cor")
> manovaTable
```

```
                SSP_fac1   SSP_fac2     SSP_int SSP_res    SSP_cor
displacement 34808.166667 9843.25000 5766.583333 4692.00 55110.0000
horsepower    4082.041667  787.00000 1057.333333 6857.25 12783.6250
acceleration     5.041667   23.24333    5.743333  105.91   139.9383
```

```
> attach(auto)
> x_bar <- colMeans(auto[, 2:4])
> x_bar_lk <- rbind(
+   colMeans(auto[cylinders == 4 & origin == 1, 2:4]),
+   colMeans(auto[cylinders == 4 & origin == 2, 2:4]),
+   colMeans(auto[cylinders == 4 & origin == 3, 2:4]),
+   colMeans(auto[cylinders == 6 & origin == 1, 2:4]),
+   colMeans(auto[cylinders == 6 & origin == 2, 2:4]),
+   colMeans(auto[cylinders == 6 & origin == 3, 2:4]))
> x_bar_l_dot <- rbind(
+   colMeans(auto[cylinders == 4, 2:4]),
+   colMeans(auto[cylinders == 6, 2:4]))
> x_bar_dot_k <- rbind(
+   colMeans(auto[origin == 1, 2:4]),
+   colMeans(auto[origin == 2, 2:4]),
```

```
+   colMeans(auto[origin == 3, 2:4]))
> SSP_cor <- SSP_fac_1 <- SSP_fac_2 <- SSP_int <-
+   SSP_res <- matrix(0, nrow = p, ncol = p)
> for(l in 1:g) {
+   SSP_fac_1 <- SSP_fac_1 + b * n *
+     t(x_bar_l_dot[l, , drop = FALSE] - x_bar) %*%
+     (x_bar_l_dot[l, , drop = FALSE] - x_bar)
+   SSP_fac_2 <- SSP_fac_2 + g * n *
+     t(x_bar_dot_k[l, , drop = FALSE] - x_bar) %*%
+     (x_bar_dot_k[l, , drop = FALSE] - x_bar)
+   for(k in 1:b) {
+     SSP_int <- SSP_int + n * t(x_bar_lk[(l - 1) * 2 + k, , drop = FALSE] -
+       x_bar_l_dot[l, , drop = FALSE] -
+         x_bar_dot_k[k, , drop = FALSE] + x_bar) %*%
+       (x_bar_lk[(l - 1) * 2 + k, , drop = FALSE] -
+          x_bar_l_dot[l, , drop = FALSE] -
+       x_bar_dot_k[k, , drop = FALSE] + x_bar)
+     for(r in 1:n) {
+       SSP_res <- SSP_res +
+         t(as.matrix(auto[(l - 1) * 2 * n + (k - 1) * n + r, 2:4]) -
+         x_bar_lk[(l - 1) * 2 + k, , drop = FALSE]) %*%
+         (as.matrix(auto[(l - 1) * 2 * n + (k - 1) * n + r, 2:4]) -
+         x_bar_lk[(l - 1) * 2 + k, , drop = FALSE])
+       SSP_cor <- SSP_cor +
+         t(as.matrix(auto[(l - 1) * 2 * n + (k - 1) * n + r, 2:4]) -
+         x_bar) %*% (as.matrix(auto[(l - 1) * 2 * n + (k - 1) *
+                                    n + r, 2:4]) - x_bar)
+     }
+   }
+ }
> # Cylinders
> Lambda <- det(SSP_res) / det(SSP_fac_1 + SSP_res)
> 1 - pf((((g * b * (n - 1) - p + 1) / 2) / ((abs((g - 1) - p) + 1) / 2)) *
+   (1 - Lambda) / Lambda, abs((g - 1) - p) + 1, g * b * (n - 1) - p + 1)

[1] 0.1033753

> # Origin
> Lambda <- det(SSP_res) / det(SSP_fac_2 + SSP_res)
> 1 - pf((((g * b * (n - 1) - p + 1) / 2) / ((abs((b - 1) - p) + 1) / 2)) *
+   (1 - Lambda) / Lambda, abs((b - 1) - p) + 1, g * b * (n - 1) - p + 1)

[1] 0.256252

> # Interaction
> Lambda <- det(SSP_res) / det(SSP_int + SSP_res)
> 1 - pf((((g * b * (n - 1) - p + 1) / 2) /
+           ((abs((g - 1) * (b - 1) - p) + 1) / 2)) *
+   (1 - Lambda) / Lambda, abs((g - 1) * (b - 1) - p) +
+     1, g * b * (n - 1) - p + 1)

[1] 0.0005879751
```

2. Conduct a simulation study to investigate the coverage probabilities of different confidence interval types with multivariate data. Let the sample size be $n = 30$, the number of variables $p = 5$, the

number of simulations $B = 10000$, and

$$\mu' = (0, 0, 0, 0, 0)$$

and

$$\Sigma = \begin{pmatrix} 1.0 & 0.6 & 0.6 & 0.6 & 0.6 \\ 0.6 & 1.0 & 0.6 & 0.6 & 0.6 \\ 0.6 & 0.6 & 1.0 & 0.6 & 0.6 \\ 0.6 & 0.6 & 0.6 & 1.0 & 0.6 \\ 0.6 & 0.6 & 0.6 & 0.6 & 1.0 \end{pmatrix}$$

For each of B times, simulate a dataset of size n from the $N_p(\mu, \Sigma)$ distribution, compute 95% confidence intervals of types one-at-a-time, T^2 simultaneous, and Bonferroni simultaneous, and record whether each interval contains its corresponding population mean component value. Report a 3×5 matrix of estimated coverage probabilities. The rows of your matrix should correspond to the 3 different interval types, and the columns should correspond to the p mean components; be sure to clearly indicate which row goes with which interval type. Comment on the performance of the different interval types.

```
> simmulationFunction <- function(n, p, rho, alpha, B = 1000) {
+   mu <- rep(0, p)
+   Sigma <- matrix(rho, nrow = p, ncol = p); diag(Sigma) <- 1
+   F_crit <- (n - 1) * p * qf(1 - alpha, p, n - p) / (n - p)
+   bon_crit <- qt(1 - alpha / (2 * p), n - 1)
+   cov_Q <- cov_T2 <- cov_bon <- matrix(NA, nrow = B, ncol = p)
+   for(i in seq_len(B)) {
+     X <- MASS::mvrnorm(n, mu, Sigma)
+     x_bar <- colMeans(X)
+     S <- var(X)
+     for(k in 1:p) {
+       ci_Q <- quantile(X[,k],c((alpha/2),(1-(alpha/2))))
+       ci_T2 <- x_bar[k] + c(-1, 1) * sqrt(F_crit * S[k, k] / n)
+       ci_bon <- x_bar[k] + c(-1, 1) * bon_crit * sqrt(S[k, k] / n)
+       cov_Q[i, k] <- (ci_Q[1] <= mu[k]) & (mu[k] <= ci_Q[2])
+       cov_T2[i, k] <- (ci_T2[1] <= mu[k]) & (mu[k] <= ci_T2[2])
+       cov_bon[i, k] <- (ci_bon[1] <= mu[k]) & (mu[k] <= ci_bon[2])
+     }
+   }
+   out <- matrix(data = NA, nrow = 3, ncol = p)
+   rownames(out) <- c("95%:", "T2:", "Bonferroni:")
+   out[1,] <- apply(cov_Q, 2, mean)
+   out[2,] <- apply(cov_T2, 2, mean)
+   out[3,] <- apply(cov_bon, 2, mean)
+   return(out)
+ }
> simmulationFunction(n = 30, p = 5, alpha = 0.05, rho = 0.6, B = 10000)

              [,1]   [,2]   [,3]   [,4]   [,5]
95%:        1.0000 1.0000 1.0000 1.0000 1.0000
T2:         0.9996 0.9994 0.9997 0.9994 0.9996
Bonferroni: 0.9900 0.9896 0.9921 0.9904 0.9905
```