Homework 04

Joseph Blubaugh
jblubau1@tamu.edu

STAT 636-720

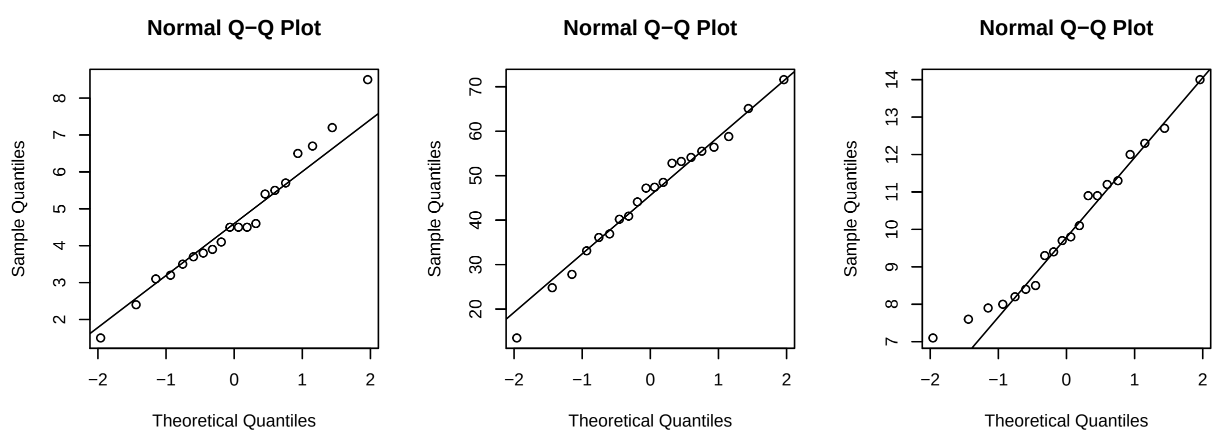
1)

a) All 3 variables appear to be normally distributed based on how the points follow close to the normal probability line. The shapiro-wilks test also supports the claim that all 3 variables are normally distributed

```
sweat = read.table("T5-1.DAT", quote="\"", comment.char="")
colnames(sweat) = c("SR", "NA", "K")
head(sweat)
```

```
   SR   NA    K
1 3.7 48.5  9.3
2 5.7 65.1  8.0
3 3.8 47.2 10.9
4 3.2 53.2 12.0
5 3.1 55.5  9.7
6 4.6 36.1  7.9
```

```
par(mfrow = c(1, 3))
qqnorm(sweat$SR); qqline(sweat$SR)
qqnorm(sweat$`NA`); qqline(sweat$`NA`)
qqnorm(sweat$K); qqline(sweat$K)
```




```
shapiro.test(sweat$SR)
```

```
    Shapiro-Wilk normality test

data:  sweat$SR
W = 0.97578, p-value = 0.8689
```

```r
shapiro.test(sweat$`NA`)
```

```

	Shapiro-Wilk normality test

data:  sweat$`NA`
W = 0.98584, p-value = 0.9862
```

```r
shapiro.test(sweat$K)
```

```

	Shapiro-Wilk normality test

data:  sweat$K
W = 0.96385, p-value = 0.6233
```

b)

```
## Covariance Matrix
(CV = cov(sweat))
```

```
          SR       NA         K
SR  2.879368  10.0100 -1.809053
NA 10.010000 199.7884 -5.640000
K  -1.809053  -5.6400  3.627658
```

```
## confidence region
(C2 = qchisq(.95, 3))
```

```
[1] 7.814728
```

```
## mu center
colMeans(sweat)
```

```
    SR     NA      K
 4.640 45.400  9.965
```

```
## axis
(eigen.vect = eigen(CV)$vectors)
```

```
            [,1]        [,2]        [,3]
[1,] -0.05084144 -0.57370364  0.81748351
[2,] -0.99828352  0.05302042 -0.02487655
[3,]  0.02907156  0.81734508  0.57541452
```

```
## half lengths
eigen.vals = eigen(CV)$values
C2/sqrt(eigen.vals)
```

```
[1] 0.5519469 3.6710350 6.8503022
```

c) Simultaneous Confidence Interval

```
## Confidence Region
(C2 = (20 - 1) * 3 * qf(.95, 3, 17) / (20 - 3))
```

```
[1] 10.7186
```

```
## Simultaneous
(ci_sim_sr = colMeans(sweat)[1] + c(-1, 1) * sqrt(C2 * CV[1, 1] / 20))
```

```
[1] 3.397768 5.882232
```

```
(ci_sim_na = colMeans(sweat)[2] + c(-1, 1) * sqrt(C2 * CV[2, 2] / 20))
```

```
[1] 35.05241 55.74759
```

```
(ci_sim_k = colMeans(sweat)[3] + c(-1, 1) * sqrt(C2 * CV[3, 3] / 20))
```

```
[1]  8.570664 11.359336
```

d) Bonferroni Confidence Interval

```
## Bonferroni
(ci_bon_sr = colMeans(sweat)[1] + c(1, -1) * qt(.95 / 6, 19) * sqrt(CV[1, 1] / 20))
```

```
[1] 4.249784 5.030216
```

```
(ci_bon_na = colMeans(sweat)[2] + c(1, -1) * qt(.95 / 6, 19) * sqrt(CV[2, 2] / 20))
```

```
[1] 42.14957 48.65043
```

```
(ci_bon_k = colMeans(sweat)[3] + c(1, -1) * qt(.95 / 6, 19) * sqrt(CV[3, 3] / 20))
```

```
[1]  9.527005 10.402995
```

e) Our p-value is .3 which is much greater than the .05 cuttoff so we do not reject the null and conclude that $\mu_0 = [4, 45, 10]$ could be the true mean.

```
library(Hotelling)
library(DescTools)

## hypothesis test
mu_0 = c(4, 45, 10)

## test statistic
T2 = 20 * t(colMeans(sweat) - mu_0) %*% solve(CV) %*% (colMeans(sweat) - mu_0)
((20 - 3) / ((20 - 1) * 3)) * T2
```

```
         [,1]
[1,] 1.304715
```

```
## critical value
qf(.05, 3, 17)
```

```
[1] 0.1151689
```

```
## pvalue
1 - pf((20 - 3) * T2 / ((20 - 1) * 3), 3, 17)
```

```
          [,1]
[1,] 0.3052847
```

```
HotellingsT2Test(x = sweat, mu = mu_0, test = 'f')
```

```

    Hotelling's one sample T2-test

data:  sweat
T.2 = 1.3047, df1 = 3, df2 = 17, p-value = 0.3053
alternative hypothesis: true location is not equal to c(4,45,10)
```

f) Since our test statistic is less than the perimeter distance of our 95% confidence ellipsoid we know that $\mu_0 = [4, 45, 10]$ is inside the region.

```
T2 < C2
```

```
     [,1]
[1,] TRUE
```

g)

```
## Center the sample around the null hypothesis
X_0 = sweat - rep(1, 20) %*% t(colMeans(sweat)) + rep(1, 20) %*% t(mu_0)

## results
A = rep(NA, 500)

## set the seed
set.seed(101)
samps = sample(1:20, size = 500*20, replace = TRUE)
samps = matrix(samps, nrow = 500, byrow = TRUE)

for(i in 1:500) {
  ## Generate bootstrap sample under H_0.
  X_b = X_0[samps[i, ], ]

  ## Compute the variance of the sample
  S_b = cov(X_b)

  ## Calculate the variance
  S_b = (1/19) * t(as.matrix(X_b - colMeans(X_b))) %*% (as.matrix(X_b - colMeans(X_b)))
  S_0 = (1/19) * t(as.matrix(X_b - mu_0)) %*% (as.matrix(X_b - mu_0))

  ## Compute test statistic
  A[i] = ( det(S_b) / det(S_0) )^10
}

## pvalue
length(which(A > 1) == TRUE) / 500
```

```
[1] 0.506
```