hw3

1a

```r
water = read.table("water.txt",header = TRUE)
diff = water$bottom - water$surface
n = length(diff)
(t.ratio = mean(diff)/(sd(diff)/sqrt(n)))
```

```
## [1] 4.863813
```

```r
(pval = 2*pt(t.ratio,df=n-1,lower.tail = FALSE))
```

```
## [1] 0.0008911155
```

We reject H_0 since the p-value is very small. If $\alpha = 0.05$, then we are 95% confident that the mean zync concentration is different between bottom and surface water. Let's see if we get the same result by using `t.test()`:

```r
t.test(x = water$bottom, y = water$surface, alternative = "two.sided", paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  water$bottom and water$surface
## t = 4.8638, df = 9, p-value = 0.0008911
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.043006 0.117794
## sample estimates:
## mean of the differences
##                  0.0804
```

1b

1) If $E[X_{ij}] \neq \mu_i$ for some j, then $\bar{X}_i$ is a biased estimation for μ_i.
2) If $\sigma_i^2 \neq \sigma^2$ for some i, then S_p is a biased estimation fro σ.
3) If the observations are dependent, like in many time series data, we cannot test the mean difference without accounting for the dependence structure.
4) If the data is not normally distributed, then the Student t distribution become inappropriate for inference.

2a not using aov()

The following solution do not use the built-in aov() function.

```r
#install.packages('HSAUR3')
library(HSAUR3)
```

```
## Loading required package: tools
```

```
##
```

```
## Attaching package: 'HSAUR3'

## The following object is masked _by_ '.GlobalEnv':
##
##     water
```

```r
data("pottery")
pottery = subset(pottery, kiln != 3)

N = nrow(pottery)
g = 4

site1 = subset(pottery, kiln == 1)[,1:9]
site2 = subset(pottery, kiln == 2)[,1:9]
site4 = subset(pottery, kiln == 4)[,1:9]
site5 = subset(pottery, kiln == 5)[,1:9]

pottery = pottery[,1:9]

center_site1 = site1 - matrix(colMeans(site1), nr = nrow(site1),
                              nc = ncol(site1), byrow = TRUE)
center_site2 = site2 - matrix(colMeans(site2), nr = nrow(site2),
                              nc = ncol(site2), byrow = TRUE)
center_site4 = site4 - matrix(colMeans(site4), nr = nrow(site4),
                              nc = ncol(site4), byrow = TRUE)
center_site5 = site5 - matrix(colMeans(site5), nr = nrow(site5),
                              nc = ncol(site5), byrow = TRUE)

n1 = nrow(site1)
n2 = nrow(site2)
n4 = nrow(site4)
n5 = nrow(site5)

site1_col_mean_center = colMeans(site1) - colMeans(pottery)
site2_col_mean_center = colMeans(site2) - colMeans(pottery)
site4_col_mean_center = colMeans(site4) - colMeans(pottery)
site5_col_mean_center = colMeans(site5) - colMeans(pottery)

center_pottery = rbind.data.frame(center_site1,center_site2,center_site4,center_site5)
F = c()
for(i in 1:9){
  SSE = sum(center_pottery[,i]^2)/(N-g)
  SST= n1*site1_col_mean_center[i]^2 + n2*site2_col_mean_center[i]^2 +
    n4*site4_col_mean_center[i]^2 + n5*site5_col_mean_center[i]^2
  SST = SST/(g-1)
  F[i] = SST/SSE
}

alpha = 0.05
cutoff = qf(1-alpha,g-1,N-g)
decision = F >= cutoff
as.numeric(decision)
```

```
## [1] 1 1 1 1 1 1 1 1 0
```

Note that we reject H_0 for Al_2O_3, Fe_2O_3, Mgo, CaO, Na_2O, K_2O, TiO_2, and MnO. But we failed to reject

H_0 for BaO. Hence, we are 95% confident that the chemical concentrations of Al_2O_3, Fe_2O_3, Mgo, CaO, Na_2O, K_2O, TiO_2, and MnO are different among the 4 sites. On the other hand, we are 95% confident that the chemical concentrations of BaO are the same among the 4 sites.

2a using aov()

The following solution makes use of the `aov()` function.

```
# Using aov()
data("pottery")
pottery = subset(pottery, kiln != 3)
result = aov(Al2O3 ~ kiln, data = pottery)
summary(result)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## kiln         3 191.88   63.96      26 2.08e-09 ***
## Residuals   39  95.94    2.46
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
result = aov(Fe2O3 ~ kiln, data = pottery)
summary(result)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## kiln         3 234.66   78.22   154.3 <2e-16 ***
## Residuals   39  19.77    0.51
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
result = aov(MgO ~ kiln, data = pottery)
summary(result)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## kiln         3 114.40   38.13   97.77 <2e-16 ***
## Residuals   39  15.21    0.39
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
result = aov(CaO ~ kiln, data = pottery)
summary(result)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## kiln         3  7.225   2.408    53.5 6.88e-14 ***
## Residuals   39  1.755   0.045
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
result = aov(Na2O ~ kiln, data = pottery)
summary(result)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## kiln         3 0.5886 0.19621   10.46 3.48e-05 ***
## Residuals   39 0.7312 0.01875
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
result = aov(K2O ~ kiln, data = pottery)
summary(result)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)    
## kiln         3 24.183   8.061   81.76 <2e-16 ***
## Residuals   39  3.845   0.099                   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
result = aov(TiO2 ~ kiln, data = pottery)
summary(result)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)    
## kiln         3 0.6528 0.21759   14.66 1.52e-06 ***
## Residuals   39 0.5789 0.01484                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
result = aov(MnO ~ kiln, data = pottery)
summary(result)
```

```
##             Df  Sum Sq  Mean Sq F value   Pr(>F)    
## kiln         3 0.07585 0.025283   52.76 8.56e-14 ***
## Residuals   39 0.01869 0.000479                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
result = aov(BaO ~ kiln, data = pottery)
summary(result)
```

```
##             Df    Sum Sq   Mean Sq F value Pr(>F)
## kiln         3 0.0000128 4.275e-06   0.459  0.712
## Residuals   39 0.0003632 9.313e-06               
```

```r
F
```

```
## [1]  26.0008709 154.3196535  97.7672981  53.5021255  10.4651667  81.7618589
## [7]  14.6584538  52.7562946   0.4590205
```

Observe that we reached the same conclusion when using `aov()` function. Also, the F-statistics are the same.

2b

```r
alpha = 0.05/9
cutoff = qf(1-alpha,g-1,N-g)
decision = F >= cutoff
as.numeric(decision)
```

```
## [1] 1 1 1 1 1 1 1 1 0
```

Same conclusion as in 2a.

2c

```r
alpha = 0.05
pval = pf(F,g-1,N-g,lower.tail = FALSE)
```

```r
pval = sort(pval,decreasing = FALSE)
decision = rep(0,9)
for(i in 1:9){
  if(pval[i] <= i*alpha/9) decision[i] = 1
}
decision
```

```
## [1] 1 1 1 1 1 1 1 1 0
```

Same conclusion as in 2a.