Charles Hesketh

Test 1

Problem 1:

1.a) From class we know we have

$$y_{ij} = \eta + a_j + \tau_i + \varepsilon_{ij}$$

to model the result of the keyboards used when

$i = 1$ or 2 for keyboard

$j = 1, 2, 3, 4, 5, 6$ for each manuscript

$\varepsilon_{ij} \sim N(0,1)$ for error.

We want to let $\tau = \tau_2 - \tau_1$ be the estimator using the data provided and we will estimate $\hat{\tau} = \bar{y}_{2.} - \bar{y}_{1.}$. This implies

$$\hat{\tau} = \frac{\sum_{j=1}^{6} y_{2j}}{6} - \frac{\sum_{j=1}^{6} y_{ij}}{6}$$

$$= (\tau_2 - \tau_1) + \Sigma(\cancel{a_j - a_j})^{0} + \Sigma(\cancel{\eta - \eta})^{0} - \sum_j \frac{\varepsilon_{2j} - \varepsilon_{1j}}{6}$$

$$= (\tau_2 - \tau_1) - \sum_j \frac{\varepsilon_{2j} - \varepsilon_{1j}}{6}$$

Now to test our estimator $\hat{\tau}$,

$$E(\hat{\tau}) = E(\tau_2 - \tau_1) - E\left(\cancel{\frac{\sum_j \varepsilon_{2j} - \varepsilon_{1j}}{6}}\right)^{0 \text{ by } N(0,1)} \Rightarrow E(\hat{\tau}) = \tau_2 - \tau_1$$

Thus $\hat{\tau}$ is an unbiased estimator.

b) Using the learning effect we can update our model to be

$$y_{ij} = \eta + a_i + \tau_j + \delta_{ij} \ell_j + \varepsilon_{ij}$$

with $\delta_{ij} = \begin{cases} 1 & \text{if keyboard is used second} \\ 0 & \text{o.w.} \end{cases}$ Thus for AB, AB, AB, BA, BA, BA we have (with $\ell_1 = \dots = \ell_6$),

$$\hat{\tau} = (\tau_2 - \tau_1) + \frac{\Sigma \ell_j(\delta_{2j} - \delta_{1j})}{6} + \frac{\sum_{j=1}^{6}(\varepsilon_{2j} + \varepsilon_{1j})\ell}{6}$$

$$= (\tau_2 - \tau_1) + \ell\frac{(\cancel{3-3})^{0}}{6} + \sum_j \frac{(\varepsilon_{2j} + \varepsilon_{1j})}{6}$$

$\Rightarrow E(\hat{\tau}) = \tau_2 - \tau_1$ for this sequence, but not all.

The problem is that in real world aplications it is not probible to have $\ell_1 = \dots = \ell_6$. Thus each ℓ_j should be weighted by difficulty. This could possibly be done by have a separate group use the same keyboard and type up each M.S. We could use the average time of each person typing each M.S divided by the grand mean to give us our weights, then run the above calculations.

This would yeild the equation

$$E(\hat{\tau}) = \tau_2 - \tau_1 + \frac{\sum_j \ell_j (\delta_{2j} + \delta_{1j})}{6}, \quad \text{not all } \ell_j \text{ equal.}$$

Where the second term wont go to zero under most real world weights.

Problem 2:

a) For Bonferroni our confidence intervals are:
AB: (-0.8002,0.4401)
AC: (-0.2402,1.0001)
AD: (-0.1802,1.0601)
BC: (-0.0602,1.1802)
BD: (-0.0002,1.2402)
CD: (-0.5602,0.6802)

For Tukey our confidence intervals are:
AB: (-0.7698,0.4098)
AC: (-0.2098,0.9698)
AD: (-0.1498,1.0298)
BC: (-0.0298,1.1498)
BD: (-0.0302,1.2098)
CD: (-0.5298,0.6498)

Tukey has a shorter interval then Bonferroni for the confidence interval of 95% given the data.

Problem 3:

a) This experiment is designed as a one way layout with:
Factors=1
Levels=4
Sample Size=30
b) The p-value for this experiment, with probability of 8.98 and degrees of freedom of 3 and 26, is 0.00059, suggesting that there is a statistical difference in at least one of the treatments.
c) Using the Tukey method we get the following comparisons:

AB=0.437
AC=4.454
AD=2.616
BC=4.153
BD=2.276
CD=1.497

d) Using the contrast provided we obtain a p-value of 0.0188 which is above the 0.01 level we were testing, thus the data does not support a statistical difference between the brand name drugs and the generic drugs.

Problem 4:

I first restructure the data so that we had a variable column and a value column. This allowed us to create a linear model to then calculate metrics. From here we used Bonferroni and Tukey's method to calculate the confidence intervals. This resulted in maximum confidence intervals of:

AB=(0.8585,2.4675)

AC=(-0.5205,1.0885)

BC=(-2.1835,-0.5745)

With a p value of 1.107e-14. Therefor, the data suggest there is a significant difference between the weights of the 3 machines.

Appendix: Code

```
library(reshape2)

setwd("C:\\Users\\User\\Desktop\\School\\Math_531T\\Exam1")



#####################################################

#1)
```

```
#####################################################

#2)

data<-read.table("http://www2.isye.gatech.edu/%7Ejeffwu/book/data/pulp.dat", h=T)
```

```
Y = as.matrix(data)
n = rep(nrow(Y),ncol(Y))
k=ncol(Y)
N=sum(n)
alpha = 0.05

Yidot = apply(Y, 2, mean) # sample mean for each treatment
avgY = mean(Yidot) # grand mean

kprime <- choose(k,2)

kprime = choose(k,2)
res = sum((Y-t(Yidot%*%t(rep(1,5))))^2)/(N-k)
b=qt(1-(alpha/(2*kprime)), N-k)*sqrt(2/5*res)

t=qtukey(1-alpha,k,N-k)/sqrt(2)*sqrt(2/5*res)

stats=matrix(0,ncol(Y)-1,ncol(Y))

for(j in 1:ncol(Y)-1)
{
  for(i in j:ncol(Y))
  {
    stats[j,i]=Yidot[i]-Yidot[j]
  }
}
```

```
uppBoundB=stats
uppBoundB[1,2:4]=stats[1,2:4]+b
uppBoundB[2,3:4]=stats[2,3:4]+b
uppBoundB[3,4]=stats[3,4]+b

lowBoundB=stats
lowBoundB[1,2:4]=stats[1,2:4]-b
lowBoundB[2,3:4]=stats[2,3:4]-b
lowBoundB[3,4]=stats[3,4]-b

uppBoundT=stats
uppBoundT[1,2:4]=stats[1,2:4]+t
uppBoundT[2,3:4]=stats[2,3:4]+t
uppBoundT[3,4]=stats[3,4]+t

lowBoundT=stats
lowBoundT[1,2:4]=stats[1,2:4]-t
lowBoundT[2,3:4]=stats[2,3:4]-t
lowBoundT[3,4]=stats[3,4]-t




######################################################
#3)
```

```
#b)

f=21.47/2.39

pf(f,3,26,lower=F)*2

#c)

yA=66.1

yB=65.75

yC=62.63

yD=63.85

k=4

Nk=26

alpha = 0.01

a=1/7

b=1/8

c=1/9

d=1/6

AB=abs(yA-yB)/(sqrt(2.39*(a+b)))

AC=abs(yA-yC)/(sqrt(2.39*(a+c)))

AD=abs(yA-yD)/(sqrt(2.39*(a+d)))

BC=abs(yB-yC)/(sqrt(2.39*(b+c)))

BD=abs(yB-yD)/(sqrt(2.39*(b+d)))
```

```
CD=abs(yC-yD)/(sqrt(2.39*(c+d)))

qtukey(1-alpha,k,Nk)/sqrt(2)

AB
AC
AD
BC
BD
CD

#D)
con=1/2*(yA+yB)-1/2*(yC+yD)
con2=c(1/2,1/2,-1/2,-1/2)
contrasts(as.factor(con2),as.factor(c(1,0,-1,0)))

AB=abs(yA-yB)/(sqrt(2.39*(a+b)))
AC=abs(yA-yC)/(sqrt(2.39*(a+c)))
AD=abs(yA-yD)/(sqrt(2.39*(a+d)))
BC=abs(yB-yC)/(sqrt(2.39*(b+c)))
BD=abs(yB-yD)/(sqrt(2.39*(b+d)))
CD=abs(yC-yD)/(sqrt(2.39*(c+d)))

SE=sqrt(2.39*(1/4*a+1/4*b+1/4*c+1/4*d))

f=(1/2*yA+1/2*yB-1/2*yC-1/2*yD)/SE

pf(f,3,26,lower=F)*2
```

```
####################################################
#4)
data=read.csv("cement.csv")
head(data)

data.m = melt(data)

g=lm(value~variable, data = data.m)
anova(g)

Y = as.matrix(data)
n = rep(nrow(Y),ncol(Y))
k=ncol(Y)
N=sum(n)
alpha = 0.05

Yidot = apply(Y, 2, mean) # sample mean for each treatment
avgY = mean(Yidot) # grand mean

kprime <- choose(k,2)

kprime = choose(k,2)
res = sum((Y-t(Yidot%*%t(rep(1,nrow(Y)))))^2)/(N-k)
b=qt(1-(alpha/(2*kprime)), N-k)*sqrt(2/5*res)

t=qtukey(1-alpha,k,N-k)/sqrt(2)*sqrt(2/5*res)

stats=matrix(0,ncol(Y)-1,ncol(Y))
```

```
for(j in 1:ncol(Y)-1)
{
  for(i in j:ncol(Y))
  {
    stats[j,i]=Yidot[i]-Yidot[j]
  }
}



uppBoundB=stats
uppBoundB[1,2:3]=stats[1,2:3]+b
uppBoundB[2,3]=stats[2,3]+b

lowBoundB=stats
lowBoundB[1,2:3]=stats[1,2:3]-b
lowBoundB[2,3]=stats[2,3]-b

uppBoundT=stats
uppBoundT[1,2:3]=stats[1,2:3]+t
uppBoundT[2,3]=stats[2,3]+t

lowBoundT=stats
lowBoundT[1,2:3]=stats[1,2:3]-t
lowBoundT[2,3]=stats[2,3]-t
```