R Codes for Chapter-4
Simultaneous Inference

1. Simultaneous Confidence Intervals for model parameters

a) Bonferroni CIs

Here we can use "confint (model)" with the adjusted level (or the adjusted alpha). Since there are only two parameters, the adjusted level is "`1-alpha/2`".

<u>Ex.</u> Plastic Hardness (continued….)

```
> alpha=0.05
> confint(SLR,level=1-alpha/2)
                 1.25 %    98.75 %
(Intercept) 161.932014 175.267986
Time          1.807526   2.261224
```

We also can use a function to calculate these intervals for any given model and a value of alpha.

```
> BonCI<-function(model, alpha)
+ {
+   X=confint(SLR,level=1-alpha/2)
+   return(X)
+ }
```

Now we can use the function for Plastic Hardness example with family confidence limit 99% (i.e. alpha=0.1) as follows

```
> BonCI(SLR,alpha=0.1)
                2.5 %    97.5 %
(Intercept) 162.9013 174.29875
Time          1.8405   2.22825
```

2. Simultaneous Estimation of Mean Responses

a) Working-Hotelling Procedure

There is not a direct R command to calculate Working-Hotteling Bounds, but we can calculate those as follows. First degrees of freedom of the residuals should be extracted from the fitted regression model, calculate "W" in the formula manually. The fitted values for the x-levels we consider can be calculated using the "predict" command. Here make sure to use "se.fit" as an argument in "predict" to have the standard deviation of each the fitted values. Then lower and upper Working-Hotteling Bounds can be calculated manually.

<u>Ex.</u> Plastic Hardness (continued….)

```
> df=df.residual(SLR)
> alpha=0.05
> W=sqrt( 2 * qf(1 - alpha, 2, df) )
> newx=data.frame(Time = c(18, 20, 22)) # Creating a data frame with the new
values of the predictor.
> pre    <- predict(SLR, newx, se.fit = TRUE)
```

```
> lwr = pre$fit - W * pre$se.fit
> upr = pre$fit + W * pre$se.fit
>
> CI=cbind('X'=newx,'Fit Val'=pre$fit,lwr,upr)
> CI
  Time  Fit Val      lwr      upr
1   18 205.2187 201.9024 208.5351
2   20 209.2875 206.3213 212.2537
3   22 213.3562 210.6940 216.0185
```

So the Working-Hotteling confidence intervals for the mean responses at Time =18, Time=20 and Time=22, with family confidence coefficient 95% are (201.9024, 208.5351), (206.3213, 212.2537), (210.6940, 216.0185) respectively.

We also can write an R function for this as follows.

```
> WH_CI <- function(model, newdata, alpha )
+ {
+   df    <- nrow(model.frame(model)) - length(coef(model))  # 23
+   W     <- sqrt( 2 * qf(1 - alpha, 2, df) )                # 2.2580
+   ci    <- predict(model, newdata, se.fit = TRUE)
+   x <- cbind(
+     'x'   = newdata,
+     'SE'   = ci$se.fit,
+     'fit' = ci$fit,
+     'lwr' = ci$fit - W * ci$se.fit,
+     'upr' = ci$fit + W * ci$se.fit)
+   return(x)
+ }
```

Using the function:

```
> new <- data.frame(Time = c(18, 20, 22))
> WH_CI(SLR, new,0.05)
  Time       SE      fit      lwr      upr
1   18 1.212760 205.2187 201.9024 208.5351
2   20 1.084726 209.2875 206.3213 212.2537
3   22 0.973571 213.3562 210.6940 216.0185


> WH_CI(SLR, data.frame(Time = c(19, 29, 23)),0.1)
  Time        SE      fit      lwr      upr
1   19 1.1469687 207.2531 204.5748 209.9315
2   29 0.8135441 227.5969 225.6971 229.4966
3   23 0.9262608 215.3906 213.2277 217.5536
```

b) Bonferroni Procedure

As in the above Bonferroni correction, we require level = (1 - alpha/g). For clarity, part of the output includes se.fit (set to true), which are the standard errors and residual.scale, which is equal to the positive square root of the MSE.

```
> alpha=0.05
> newx=data.frame(Time = c(18, 20, 22))
> g=nrow(newx) #number of parameters
> predict(SLR, newx, int = "confidence", level = (1 - alpha/g), se.fit = TRUE
)
```

```
$fit
       fit      lwr      upr
1 205.2187 201.9228 208.5147
2 209.2875 206.3395 212.2355
3 213.3562 210.7103 216.0022

$se.fit
       1        2        3
1.212760 1.084726 0.973571

$df
[1] 14

$residual.scale
[1] 3.234027
```

So the Bonferroni confidence intervals for the mean responses at Time =18, Time=20 and Time=22, with family confidence coefficient 95% are (201.9228, 208.5147), (206.3395, 212.2355), (210.7103, 216.0022) respectively.

A function can also be used as follows,

```
> BCIMR=function(model, newdata, alpha )
+  {
+
+   g=nrow(newdata)
+   CI=predict(model, newdata, int="c",level =(1-alpha/g), se.fit = TRUE)
+   return(CI)
+   }
```

Using the function for the plastic hardness example with family confidence coefficient 90%

```
> BCIMR(SLR, data.frame(Time = c(19, 29, 23)),0.1)
$fit
       fit      lwr      upr
1 207.2531 204.5465 209.9598
2 227.5969 225.6771 229.5167
3 215.3906 213.2048 217.5764

$se.fit
        1         2         3
1.1469687 0.8135441 0.9262608

$df
[1] 14

$residual.scale
[1] 3.234027
```

With family confidence coefficient 99%

```
> BCIMR(SLR, data.frame(Time = c(19, 29, 23,25)),0.01)
$fit
       fit      lwr      upr
1 207.2531 203.0385 211.4678
2 227.5969 224.6074 230.5863
3 215.3906 211.9870 218.7943
4 219.4594 216.3258 222.5930
```

```
$se.fit
        1         2         3         4 
1.1469687 0.8135441 0.9262608 0.8527733 

$df
[1] 14

$residual.scale
[1] 3.234027
```

3. Simultaneous Prediction Intervals for New Observations

The Scheffe and Bonferroni Procedures

Here the calculations are almost same as the previous section and so the only the function is given. Note that this function has an additional argument to choose the method (Scheffe or Bonferroni) we use.

```
> PI_SoB <- function(model, newdata, type = c("B", "S"), alpha)
+ {
+   g  <- nrow(newdata)
+   CI <- predict(model, newdata, se.fit = TRUE)
+   if(match.arg(type) == "B"){
+               M=  qt(1 - alpha / (2*g), model$df)} # B = (4.9a)
+         else{
+               M= sqrt( g * qf( 1 - alpha, g, model$df))} # S = (4.8a)
+
+   spred <- sqrt( CI$residual.scale^2 + (CI$se.fit)^2 )  # (2.38)
+   x <- data.frame(
+     "x"     = newdata,
+     "spred" = spred,
+     "fit"   = CI$fit,
+     "lower" = CI$fit - M * spred,
+     "upper" = CI$fit + M * spred)
+
+   return(x)
+ }
```

The function can be used with type=”S” or using “S” for the second argument for the Sheffe method or with type=”B” or using “B” for the second argument for the Bonferroni method.

Fothr Sheffe method:

```
> PI_SoB(SLR, data.frame(Time = c(19, 29, 23)), type = "S",0.05)
  Time    spred      fit    lower    upper
1   19 3.431394 207.2531 196.3849 218.1213
2   29 3.334784 227.5969 217.0347 238.1591
3   23 3.364058 215.3906 204.7357 226.0455
```

For the Bonferroni method:

```
> PI_SoB(SLR, data.frame(Time = c(19, 29, 23,25)), type = "B",0.01)
  Time    spred      fit    lower    upper
1   19 3.431394 207.2531 194.6441 219.8621
2   29 3.334784 227.5969 215.3429 239.8509
3   23 3.364058 215.3906 203.0291 227.7522
4   25 3.344570 219.4594 207.1694 231.7493
```

Regression through Origin

Since Regression through the origin is just a special case of the simple linear Regression model, the command "lm" can be used as "lm(y ~ 0 + x)" or alternatively "lm(y ~ x - 1)".

Ex. Warehouse data (in Chapter- 4) (X: Work Units Performed, Y: Variable Labor Cost)

```
> WHdata<-read.table("http://www.stat.ufl.edu/~rrandles/sta4210/Rclassnotes/d
ata/textdatasets/KutnerData/Chapter%20%204%20Data%20Sets/CH04TA02.txt ", head
er= FALSE , sep="")
> WHdata
    V1  V2
1   20 114
2  196 921
3  115 560
4   50 245
5  122 575
6  100 475
7   33 138
8  154 727
9   80 375
10 147 670
11 182 828
12 160 762
```

The Regression (trough the origin) modal,

```
> fit <- lm(Y ~ 0 + X)
> fit

Call:
lm(formula = Y ~ 0 + X)

Coefficients:
     X
0.1216
```

So the fitted regression model is Y=0.1216 X.

```
> summary(fit)

Call:
lm(formula = Y ~ 0 + X)

Residuals:
    Min      1Q  Median      3Q     Max
-3.0276 -0.2737  0.1077  0.4754  2.1820

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
X 0.121643   0.002637   46.13   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7257 on 119 degrees of freedom
```

```
Multiple R-squared:  0.947,    Adjusted R-squared:  0.9466
F-statistic:  2128 on 1 and 119 DF,  p-value: < 2.2e-16
```

ANOVA table for the fitted model:

```
> anova(fit)
Analysis of Variance Table

Response: Y
           Df  Sum Sq Mean Sq F value    Pr(>F)    
X           1 1120.71 1120.71  2128.1 < 2.2e-16 ***
Residuals 119   62.67    0.53                      
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Confidence interval for β_1:

```
> confint(fit,level=0.99)
      0.5 %    99.5 %
X 0.1147401 0.1285458
```