ProblemSet3

Aaron Graybill

3/8/2021

Problem 2.6

a.

$$y_i - \hat{y}_i = y_i - \hat{\beta}_0 + \hat{\beta}_1 x_i = y_i - \overline{y} - \hat{\beta}_1 \overline{x} + \hat{\beta}_1 x_i = (y_i - \overline{y}) - \hat{\beta}_1 (x_i - \overline{x})$$

b.

$$\hat{y}_i - \overline{y} = \hat{\beta}_0 + \hat{\beta}_1 x_i - \overline{y} = \overline{y} - \hat{\beta}_1 \overline{x} + \hat{\beta}_1 x_i - \overline{y} = \hat{\beta}_1 (x_i - \overline{x})$$

c.

$$\begin{aligned}
&\sum_{i=1}^{n} (\hat{y}_i - \overline{y}) (y_i - \hat{y}_i) \\
&\sum_{i=1}^{n} \Big((y_i - \overline{y}) - \hat{\beta}_1 (x_i - \overline{x}) \Big) \Big(\hat{\beta}_1 (x_i - \overline{x}) \Big) \\
&\sum_{i=1}^{n} (y_i - \overline{y}) \hat{\beta}_1 (x_i - \overline{x}) - \hat{\beta}_1^2 (x_i - \overline{x})^2 \\
&\hat{\beta}_1 SXY - \hat{\beta}_1^2 SXX \\
&\frac{SXY}{SXX} SXY - \left(\frac{SXY}{SXX} \right)^2 SXX \\
&\frac{SXY^2 - SXY^2}{SXX} = 0
\end{aligned}$$

Simultaneous Example Problem 1

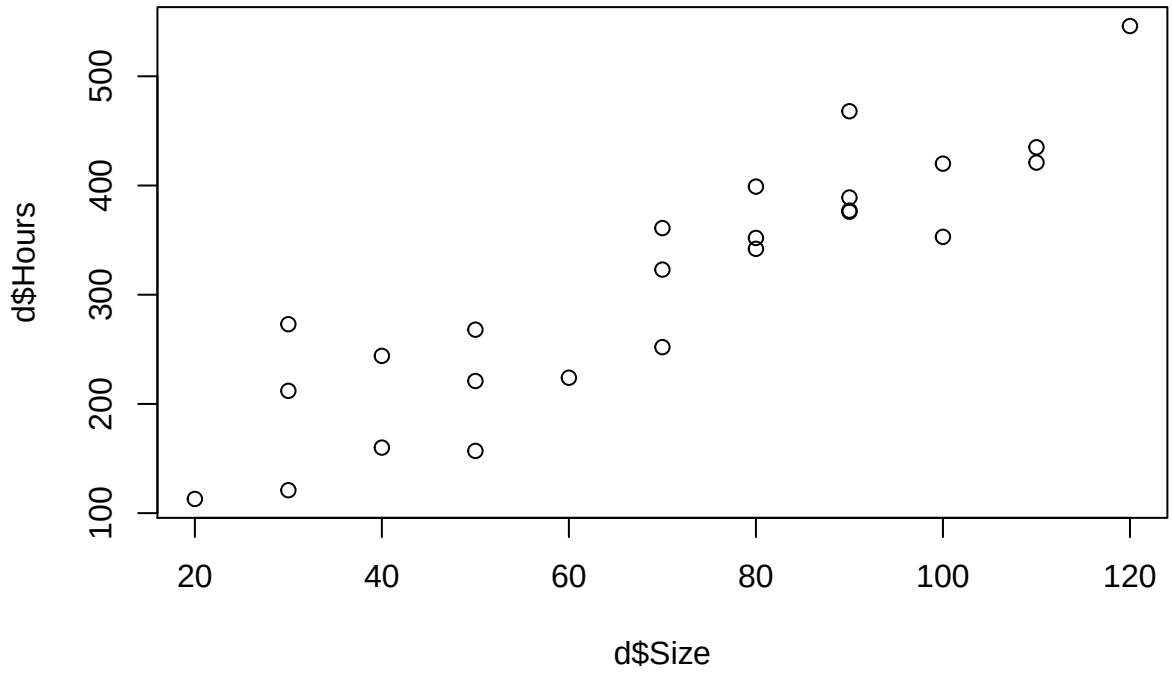
```
rm(list=ls())
library(here)

d <-
  read.csv(here("data","Toluca.csv"))
```

a.

I say cost a lot in this question when I mean hours. I realized this during the last part, and I don't have enough time this week to change it, but I at least wanted to recognize the error made.

The following code plots the Hours against the Size

```r
plot(d$Size,d$Hours)
```



By the plot above a linear model seems okay, although the variance of the errors might be quite high.

b.

We can find the regression line with the following code:

```r
out <- lm(Hours~Size,data=d);summary(out)
```

```
##
## Call:
## lm(formula = Hours ~ Size, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -83.876 -34.088  -5.982  38.826 103.528
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   62.366     26.177   2.382   0.0259 *
## Size           3.570      0.347  10.290 4.45e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 48.82 on 23 degrees of freedom
## Multiple R-squared:  0.8215, Adjusted R-squared:  0.8138
## F-statistic: 105.9 on 1 and 23 DF,  p-value: 4.449e-10
```

c.

I compute the confidence intervals in the following way:

```r
confint(out,"Size",level=.95)
```

```
##         2.5 %   97.5 %
## Size 2.852435 4.287969
```

```
confint(out,"Size",level=.9)
```

```
##           5 %     95 %
## Size 2.975536 4.164868
```

Both confidence intervals are strictly positive, so we expect a positive relationship between the `Size` of the batch and the `Cost` thereof.

d.

However, I find the probability that we observe the data we do given that the slope was actually zero in the following way. I need not actually compute it because the summary output of the regression model has the p-value for a two side test at zero, so simply dividing this by two would give the p-value of the one-tail positive test. The reported p-value is 4.5×10^{-10}, so dividing this by two gives: 2.3×10^{-10}. This is a very small p-value, so we can be rather certain that there is a positive relationship.

e. & f.

We can compute both of these confidence intervals in the following way:

```
prediction_points <-
  data.frame(Size=c(65,100))
predict(out,prediction_points,interval = 'confidence',se.fit=T,level = .9)
```

```
## $fit
##        fit      lwr      upr
## 1 294.4290 277.4315 311.4264
## 2 419.3861 394.9251 443.8470
##
## $se.fit
##         1         2
##  9.917579 14.272328
##
## $df
## [1] 23
##
## $residual.scale
## [1] 48.82331
```

I suppose I should briefly interpret these results. When there are 65 size units, we are 90% certain that the mean value of cost is between \$277.43 and \$311.43. Similarly, when the lot is of size 100, we are 90% confident that the mean cost is between \$419.39 and \$443.85.

g.

This question is asking for a prediction interval at 100 units, we can implement that as such:

```
prediction_points <-
  data.frame(Size=c(100))
predict(out,prediction_points,interval = 'prediction',se.fit=T,level = .9)
```

```
## $fit
##        fit      lwr      upr
## 1 419.3861 332.2072 506.5649
##
```

```
## $se.fit
## [1] 14.27233
##
## $df
## [1] 23
##
## $residual.scale
## [1] 48.82331
```

Given a lot of size 100 units, we can be 90% confident that the corresponding cost of the lot is between \$332.21 and \$506.56.

h.

This question is asking about the R squared, so reading off of the summary table gives that 82.15% of the variation in cost can be explained by the intercept and the variation in `Size`.

```
summary(out)
```

```
##
## Call:
## lm(formula = Hours ~ Size, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -83.876 -34.088  -5.982  38.826 103.528
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   62.366     26.177   2.382   0.0259 *
## Size           3.570      0.347  10.290 4.45e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 48.82 on 23 degrees of freedom
## Multiple R-squared:  0.8215, Adjusted R-squared:  0.8138
## F-statistic: 105.9 on 1 and 23 DF,  p-value: 4.449e-10
```

i.

The Bonferroni method says that if we wish to have a joint confidence level of $\alpha_j = .1$, we need to have individual confidence levels (2) at $\alpha_i = \frac{\alpha_j}{2} = .05$. Therefore we need to compute a 90% confidence interval. That gives:

```
confint(out,level = 1-.1/2)
```

```
##                2.5 %     97.5 %
## (Intercept) 8.213711 116.518006
## Size        2.852435   4.287969
```

j.

The Working-Hotelling procedure has us compute the $\hat{y}^*$ and its standard error and the W statistic. I do that for $30, 65, 100$ as follows:

```
data=data.frame(Size=c(30,65,100))
prediction <-
```

```
  predict(out,data,interval="confidence",se.fit=T,level=.9)

# Compute prediction interval standard error
ses <-
  # I don't think this is right, i may have to add the square residuals??
  sqrt(prediction$se.fit^2)

W <- sqrt(2*qf(1-.1,2,23))

data.frame(
  fit=prediction$fit[,1],
  up=prediction$fit[,1]+W*ses,
  low=prediction$fit[,1]-W*ses)
```

```
##        fit       up      low
## 1 169.4719 207.7897 131.1542
## 2 294.4290 316.8229 272.0351
## 3 419.3861 451.6130 387.1591
```

k.

The Bonferonni method is different (that's why it has a different name), we can implement this as follows:

```
# the percentile here might not be right.
# I have some note saying I don't need the extra 2, but that seems wrong
t=qt(1-.9/(2*2),23)

se <- #this might not be right
  sqrt(48.82^2+prediction$se.fit^2)

data.frame(
  fit=prediction$fit[,1],
  up=prediction$fit[,1]+t*se,
  low=prediction$fit[,1]-t*se)
```

```
##        fit       up      low
## 1 169.4719 209.1929 129.7510
## 2 294.4290 332.7143 256.1437
## 3 419.3861 458.4755 380.2966
```

j.

I think what this question is asking for, though maybe I'm missing something is for us to compare the results of the Scheffé against the Bonferroni, and use the method with the smaller prediction interval.

```
# Setup:
data=data.frame(Size=c(80,100))
prediction <-
  predict(out,data,interval="confidence",se.fit=T,level=.95)

# Bonferroni ----
# the percentile here might not be right.
# I have some note saying I don't need the extra 2, but that seems wrong
t=qt(1-.95/(2*2),23)

se <- #this might not be right
```

```
  sqrt(48.82^2+prediction$se.fit^2)

data.frame(
  fit=prediction$fit[,1],
  up=prediction$fit[,1]+t*se,
  low=prediction$fit[,1]-t*se)
```

```
##        fit       up      low
## 1 347.9820 384.2285 311.7355
## 2 419.3861 456.3267 382.4455
```

```
# Scheffé ----
W=sqrt(2*qf(1-.95,2,23))

se <- #this might not be right
  sqrt(48.82^2+prediction$se.fit^2)

data.frame(
  fit=prediction$fit[,1],
  up=prediction$fit[,1]+W*se,
  low=prediction$fit[,1]-W*se)
```

```
##        fit       up      low
## 1 347.9820 363.9849 331.9792
## 2 419.3861 435.6954 403.0767
```

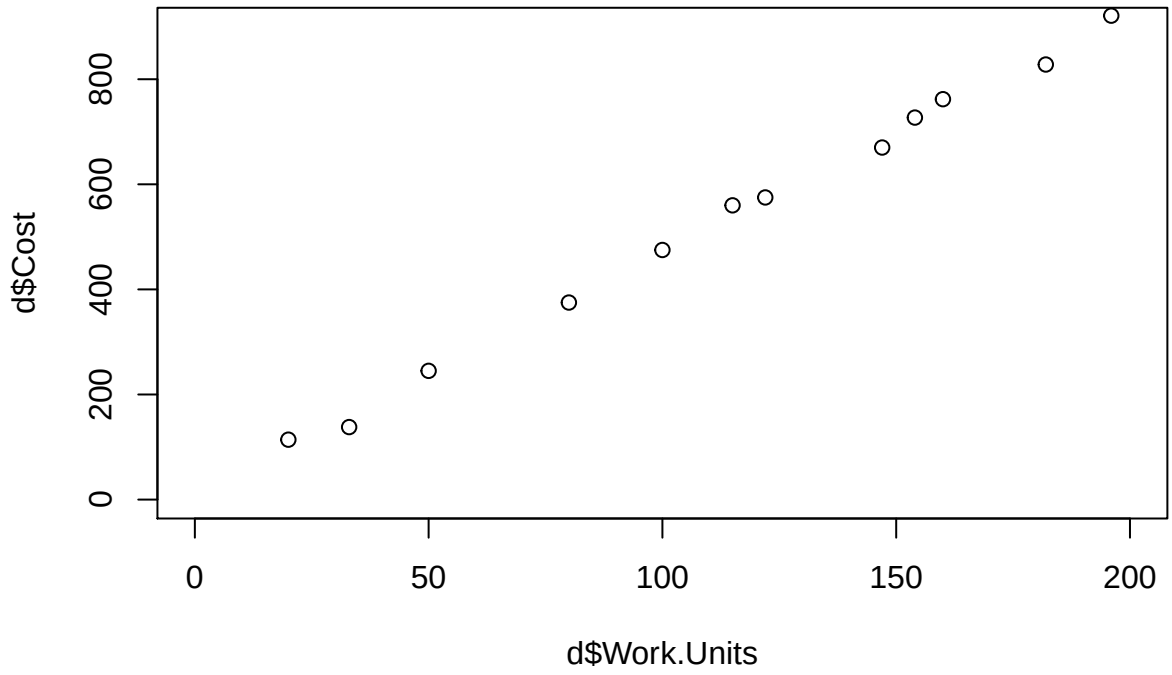
I'm surprised at how big the differences between these two methods are, but for both estimation points, 80 and 100, the Scheffé method produces smaller intervals, so we would select this method to say that the family prediction interval says that we are 95% confident that the true value of hours lies between 331 and 363 and 403 and 436 for 80 and 100 as the size of the batch respectively.

Simultaneous Example Question 2

```
d <- read.csv(here("Data/Charles.csv"))
```

a.

```
plot(d$Work.Units,d$Cost,xlim=c(0,200),ylim = c(0,900))
```



###

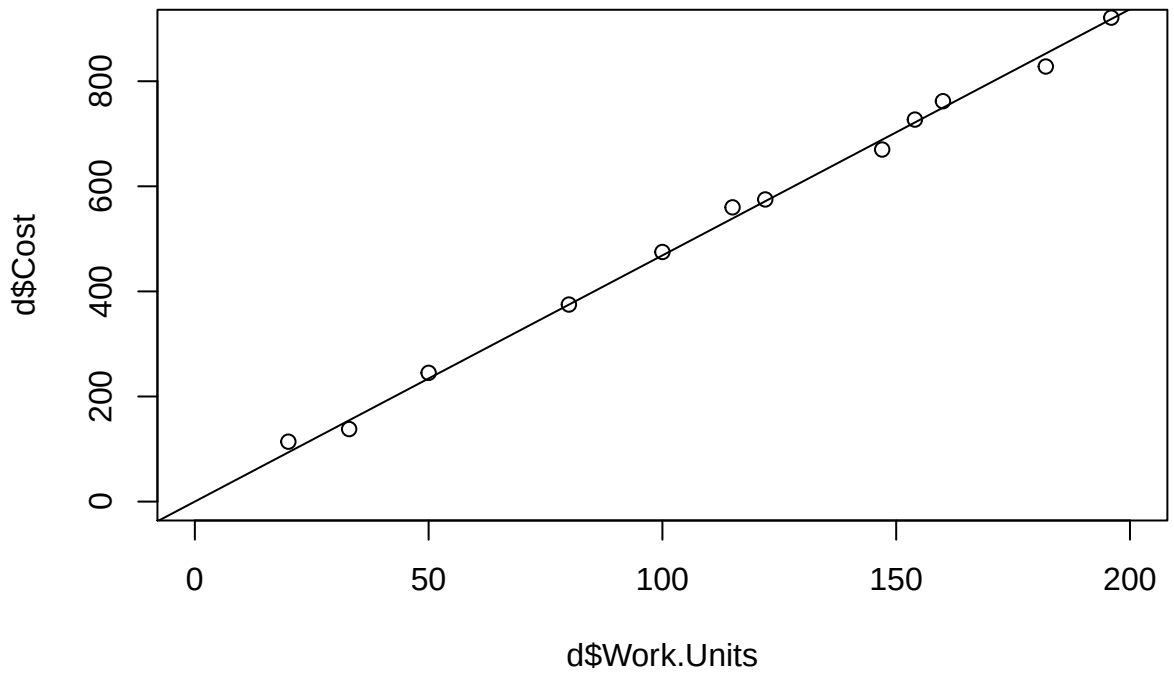
b.

```
out <-
  lm(Cost~Work.Units-1,data=d)
summary(out)
```

```
##
## Call:
## lm(formula = Cost ~ Work.Units - 1, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.720  -4.020   4.432  11.141  21.194
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## Work.Units  4.68527    0.03421     137   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.95 on 11 degrees of freedom
## Multiple R-squared:  0.9994, Adjusted R-squared:  0.9994
## F-statistic: 1.876e+04 on 1 and 11 DF,  p-value: < 2.2e-16
```

c.

```
plot(d$Work.Units,d$Cost,xlim=c(0,200),ylim = c(0,900))
abline(out)
```



Based on the plot above, it is quite clear that the linear model through the origin is appropriate. All of the points seem to fall near the line and they are not off in any systematic way.

d.

We can compute exactly the desired confidence interval with the stock `confint()` command:

```
confint(out)
```

```
##                2.5 %   97.5 %
## Work.Units 4.609989 4.760559
```

e.

And finally, we can compute the confidence interval in the following way.

```
data=data.frame(Work.Units=100)
predict(out,data,level=.9,interval = 'confidence',se.fit=T)
```

```
## $fit
##        fit      lwr      upr
## 1 468.5274 462.3846 474.6702
##
## $se.fit
## [1] 3.420502
##
## $df
## [1] 11
##
## $residual.scale
## [1] 14.94736
```

So we can be 90% confident that mean of the Cost would be between $462 and $474 dollars.