Assignment 8

Chase Baggett

November 9, 2017

8.1

```
model_1 <- lm(lifeExpF ~ 1,data=UN11)
model_2 <- lm(lifeExpF ~ group,data=UN11)
anova(model_1,model_2)
```

```
## Analysis of Variance Table
##
## Model 1: lifeExpF ~ 1
## Model 2: lifeExpF ~ group
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1    198 20293.2
## 2    196  7730.2  2     12563 159.27 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

8.2

One model is not a constrained version of the other model. We can do a comparison in this manner between 6.7 or 6.8 and 6.9 or 6.10 because it is possible to constrain the other model to assume a coefficient is equal to zero, but we cannot constrain 6.7 or 6.8 to be equal to each other.

We can express the t-test as the mean of the sample minus the hypothesized value, over the the standard error expressed as the sample standard deviation divided by the square root of the sample size.

$$t = \frac{\bar{y} - y}{sd/\sqrt{n}}$$

We can also epxress the F test in terms of the standard deviation, sample size, and sample mean.

$$F = \frac{n(\bar{y} - y)^2}{sd^2}$$

If we express them this way, it becomes apparent that $F = t^2$.

8.3

```
model <- lm(lifeExpF ~ group + log(ppgdp) + group:log(ppgdp), data=UN11)
summary(model)
```

```
##
## Call:
## lm(formula = lifeExpF ~ group + log(ppgdp) + group:log(ppgdp),
##     data = UN11)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.634  -2.089   0.301   2.255  14.489
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              59.2137    15.2203   3.890 0.000138 ***
## groupother              -11.1731    15.5948  -0.716 0.474572
## groupafrica             -22.9848    15.7838  -1.456 0.146954
## log(ppgdp)                2.2425     1.4664   1.529 0.127844
## groupother:log(ppgdp)     0.9294     1.5177   0.612 0.540986
## groupafrica:log(ppgdp)    1.0950     1.5785   0.694 0.488703
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.129 on 193 degrees of freedom
## Multiple R-squared:  0.7498, Adjusted R-squared:  0.7433
## F-statistic: 115.7 on 5 and 193 DF,  p-value: < 2.2e-16
```

8.4

First, we fit all of the models.

```
model_1 <- lm(Y ~ X1 + I(X1^2) + X2 + I(X2^2) + X1:X2, data = cakes)
model_2 <- lm(formula = Y ~ X1 + I(X1^2) + X2 + I(X2^2), data = cakes)
model_3 <- lm(formula = Y ~ X1 + X2 + I(X2^2) + X1:X2, data = cakes)
model_4 <- lm(formula = Y ~ X2 + I(X2^2), data = cakes)
```

First Test We reject the null hypothesis due to the .005 P-Value.

```
anova(model_2,model_1)
```

```
## Analysis of Variance Table
##
## Model 1: Y ~ X1 + I(X1^2) + X2 + I(X2^2)
## Model 2: Y ~ X1 + I(X1^2) + X2 + I(X2^2) + X1:X2
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1      9 4.2430
## 2      8 1.4707  1    2.7722 15.079 0.004654 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Second Test We reject the null hypothesis due to the .004 P-Value.

```
anova(model_3,model_1)
```

```
## Analysis of Variance Table
##
## Model 1: Y ~ X1 + X2 + I(X2^2) + X1:X2
## Model 2: Y ~ X1 + I(X1^2) + X2 + I(X2^2) + X1:X2
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1      9 4.3785
## 2      8 1.4707  1    2.9077 15.816 0.004079 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Third Test We reject the null hypothesis due to the .0006 P-Value.

```
anova(model_4,model_1)
```

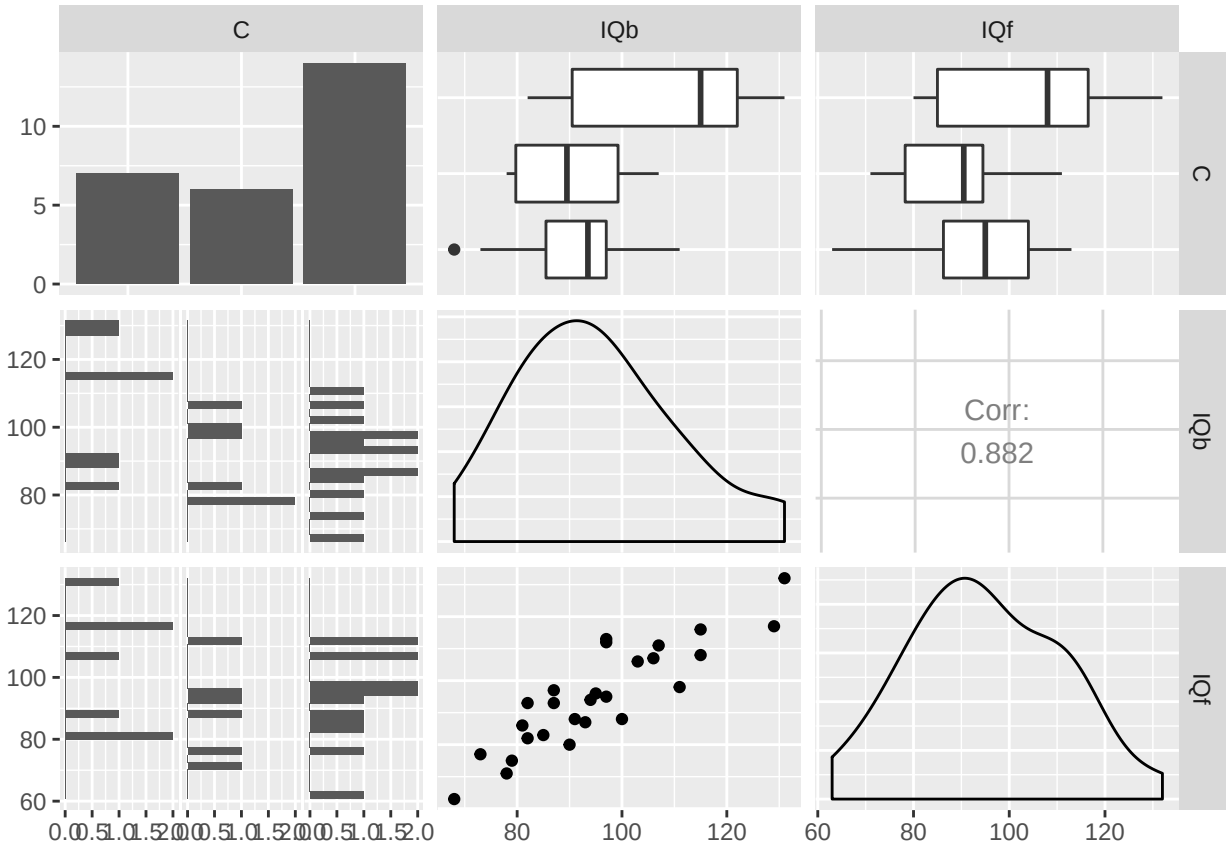
```
## Analysis of Variance Table
##
## Model 1: Y ~ X2 + I(X2^2)
## Model 2: Y ~ X1 + I(X1^2) + X2 + I(X2^2) + X1:X2
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1     11 11.4739
## 2      8  1.4707  3    10.003 18.137 0.0006293 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

8.5

We start with an exploration of the data, and we see that IQb and IQf both appear roughly normally distributed. We see a correlation between IQf and IQb almost immediately.

```
ggpairs(twins)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Now we fit a simple model of IQf and IQb.

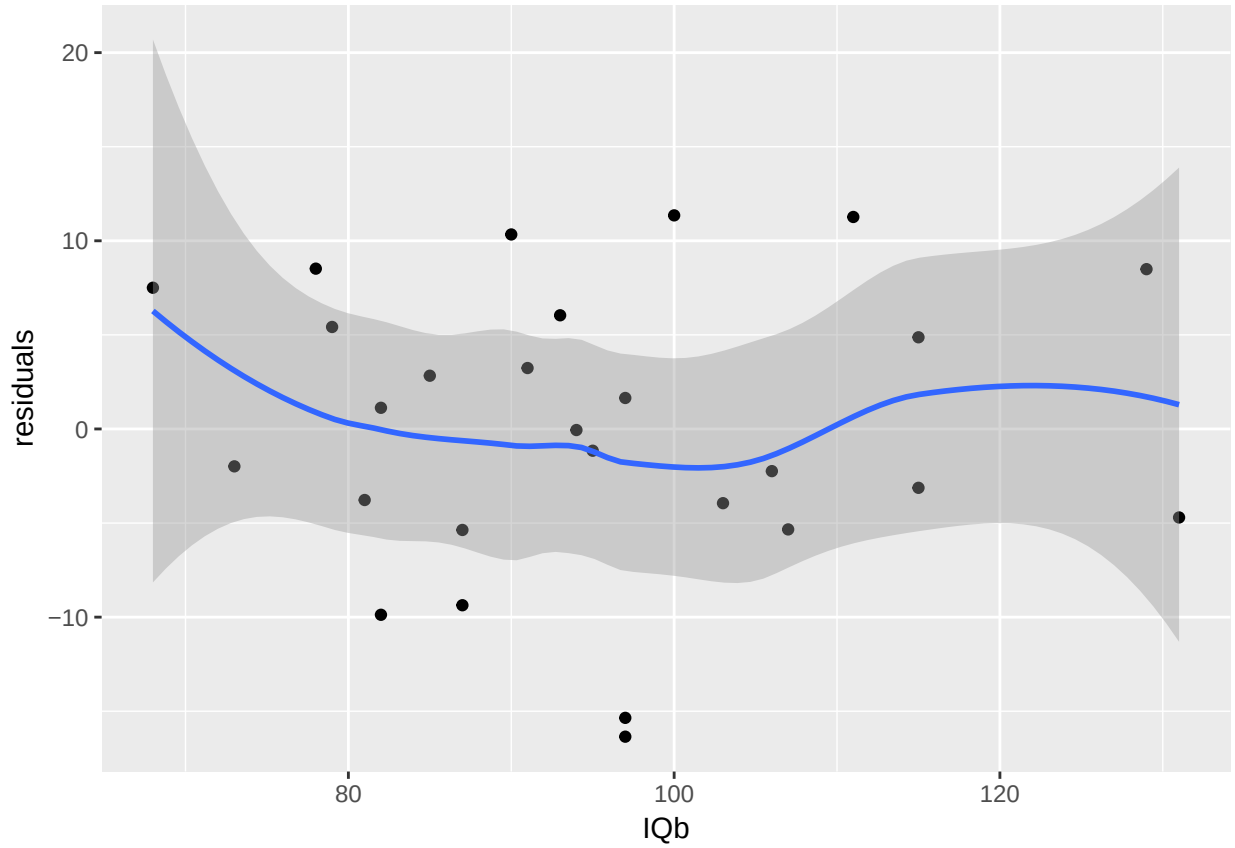
```
model <- lm(IQf ~ IQb,data=twins)
summary(model)
```

```
##
## Call:
## lm(formula = IQf ~ IQb, data = twins)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.3512  -5.7311   0.0574   4.3244  16.3531
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.20760    9.29990   0.990    0.332
## IQb          0.90144    0.09633   9.358  1.2e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.729 on 25 degrees of freedom
## Multiple R-squared:  0.7779, Adjusted R-squared:  0.769
## F-statistic: 87.56 on 1 and 25 DF,  p-value: 1.204e-09
```

Now extract the residuals and look at the variance. It appears to be fairly constant, so we try a test for non-constant variance.

```
twins$residuals <- predict(model) - twins$IQf
ggplot(twins,aes(x=IQb,y=residuals)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'loess'
```




```
ncvTest(model)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 0.03830386    Df = 1     p = 0.8448342
```

Now, we fit the model with C as a factor, and it appears class is not significant. IQb is very significant, and the model has an R-Square of .8 which suggests a large amount of the variation in IQf is explained by IQb.

```
model <- lm(IQf ~ IQb + C,data=twins)
summary(model)
```

```
##
## Call:
## lm(formula = IQf ~ IQb + C, data = twins)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.8235  -5.2366  -0.1111   4.4755  13.6978
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.6076    11.8551  -0.051    0.960
## IQb           0.9658     0.1069   9.031 5.05e-09 ***
## CC2           2.0353     4.5908   0.443    0.662
## CC3           6.2264     3.9171   1.590    0.126
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 7.571 on 23 degrees of freedom
## Multiple R-squared:  0.8039, Adjusted R-squared:  0.7784
## F-statistic: 31.44 on 3 and 23 DF,  p-value: 2.604e-08
```

We can use an F test as well to verify that IQb is significant, and we see we can reject the null nypothesis that there is no significance.

```
nh <- lm(IQf ~ 1,data=twins)
ah <- lm(IQf ~ IQb,data=twins)
anova(ah,nh)
```

```
## Analysis of Variance Table
##
## Model 1: IQf ~ IQb
## Model 2: IQf ~ 1
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     25 1493.5
## 2     26 6724.7 -1   -5231.1 87.563 1.204e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Now we move onto C.

Let's test the model with and without C and see if we can reject the null hypothesis that class is not significant.

We cannot reject the null hypothesis using an F test.

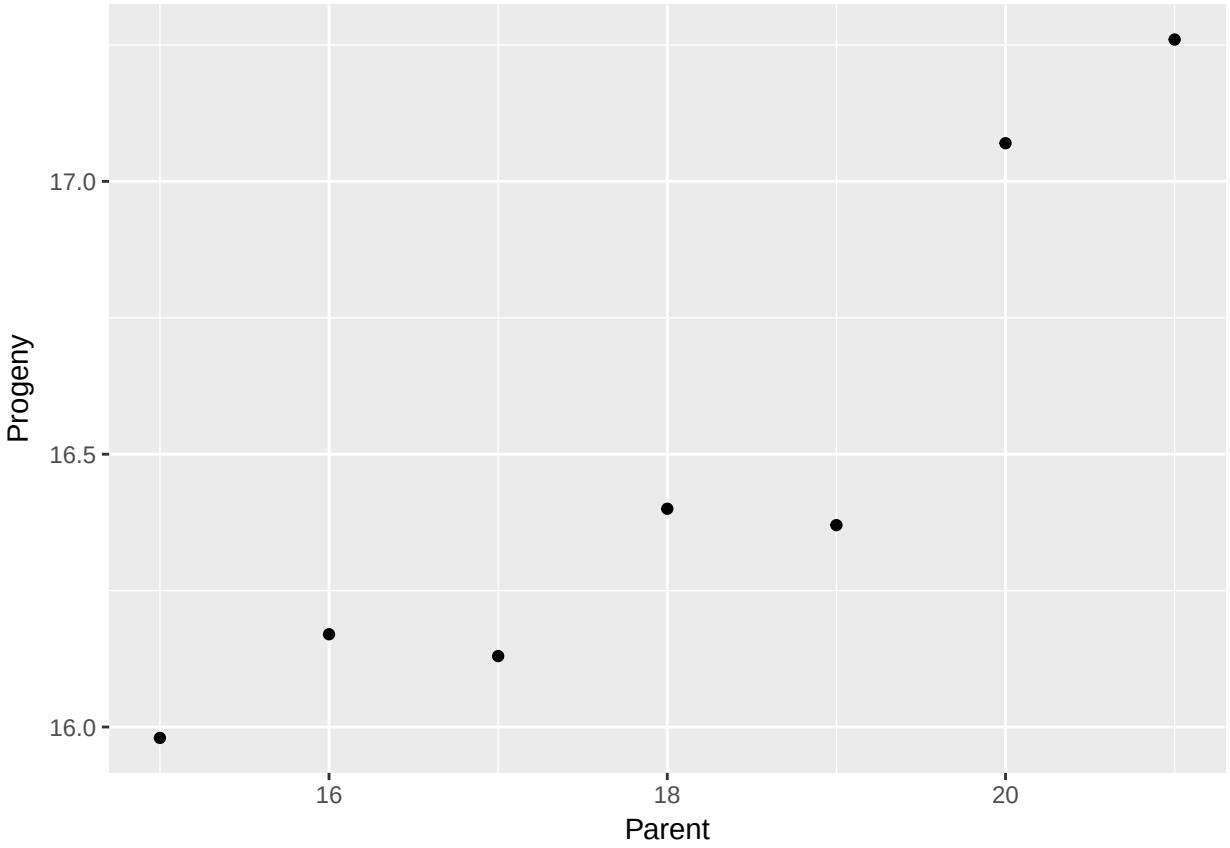
```
nh <- lm(IQf ~ IQb,data=twins)
ah <- lm(IQf ~ IQb + C,data=twins)
anova(ah,nh)
```

```
## Analysis of Variance Table
##
## Model 1: IQf ~ IQb + C
## Model 2: IQf ~ IQb
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     23 1318.4
## 2     25 1493.5 -2   -175.13 1.5276 0.2383
```

8.6

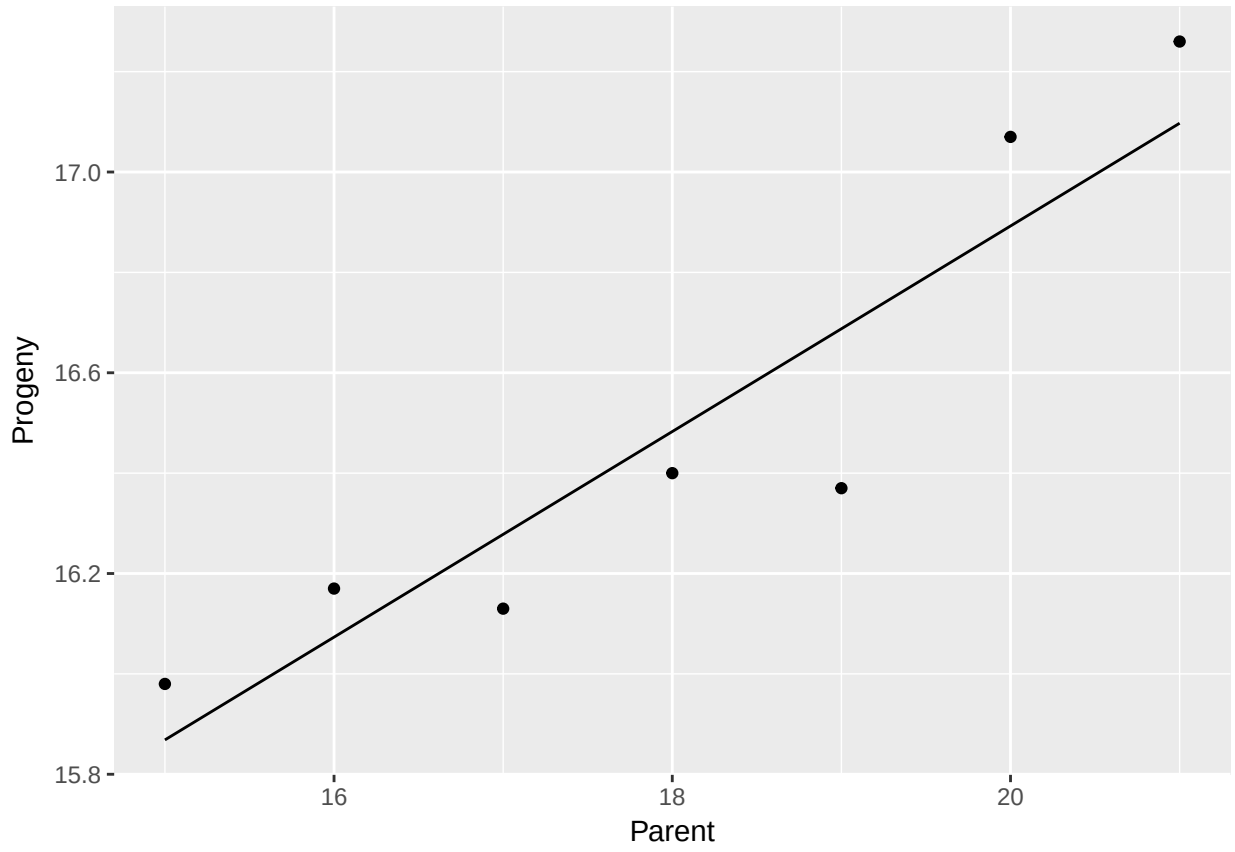
7.7.1

```
ggplot(galtonpeas,aes(x=Parent,y=Progeny)) + geom_point()
```



7.7.2

```
weighted_regression = lm(formula = Progeny ~ Parent, data = galtonpeas, weights = 1/SD^2)
ggplot(galtonpeas,aes(x=Parent,y=Progeny)) +
    geom_point() +
    geom_line(aes(y=predict(weighted_regression)))
```



7.7.3

The intercept would be increased due to the decrease in the slope, and the variance would be increased due to the biased selection.

8.6.1

Without weights, we are estimating the mean of a group of peas of an unknown size, not the size of an individual pea.

```
unweighted_regression = lm(formula = Progeny ~ Parent, data = galtonpeas)
```

8.7

```
model_1 <- lm(Y ~ X1 + X2 + I(X1^2) + I(X2^2) + X1:X2, data=cakes)
param.names <- paste("b", 0:5, sep="")
x1.max <- "(b2*b5-2*b1*b4)/(4*b3*b4-b5^2)"
deltaMethod(model_1, x1.max,parameterNames=param.names)
```

```
##                                           Estimate        SE    2.5 %
## (b2 * b5 - 2 * b1 * b4)/(4 * b3 * b4 - b5^2) 35.82766 0.4330974 34.97881
##                                             97.5 %
```

```
## (b2 * b5 - 2 * b1 * b4)/(4 * b3 * b4 - b5^2) 36.67652
```

8.8

First let's fit the model and use the built in package.

```
model <- lm(Y ~ .,data=sniffer)
ncvTest(model)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 4.802652    Df = 1     p = 0.02841596
```

Now we do the math manually and make return the p-value of the fitted model, which matches both the book and the test above.

```
score_test_fitted_values <- function(model){
  model_summary <-summary(model)
  rss <- residuals(model_summary, type="pearson")
  S.sq <-df.residual(model)*(model_summary$sigma)^2/sum(!is.na(rss))
  U <-(rss^2)/S.sq
  model_U <- lm(U~fitted.values(model))
  df<-1
  SS<-anova(model_U)$"Sum Sq"
  SS_2<-sum(SS)-SS[length(SS)]
  Chisq<-SS_2/2
  if (is.na(df)) {df <- coefficients(model_U)}
  1-pchisq(Chisq, df)
}

#Using the Fitted Values
score_test_fitted_values(model)
```

```
## [1] 0.02841596
```

We can regenerate all the results in the table by changing the model for U. We can see they match the book after accounting for rounding.

```
score_test <- function(model,formula){
  model_summary <-summary(model)
  rss <- residuals(model_summary, type="pearson")
  S.sq <-df.residual(model)*(model_summary$sigma)^2/sum(!is.na(rss))
  U <-(rss^2)/S.sq
  model_U <- lm(as.formula(formula),data=model$model)
  df <- sum(!is.na(coefficients(model_U))) - 1
  SS<-anova(model_U)$"Sum Sq"
  SS_2<-sum(SS)-SS[length(SS)]
  Chisq<-SS_2/2
  if (is.na(df)) {df <- coefficients(model_U)}
  1-pchisq(Chisq, df)
}


#GasPres
score_test(model,formula="U ~ GasPres")
```

```
## [1] 0.0190138
```

```
#TankTemp
score_test(model,formula="U ~ TankTemp")
```

```
## [1] 0.001837131
```

```
#TankTemp,GasPres
score_test(model,formula="U ~ TankTemp + GasPres")
```

```
## [1] 0.002769486
```

```
#TankTemp,GasTemp,TankPres,GasPres
score_test(model,formula="U ~ TankTemp + GasTemp + TankPres + GasPres")
```

```
## [1] 0.008102013
```