CS498AML Applied Machine Learning Homework 4

Huamin Zhang, Rongzi Wang

Mar 4, 2017

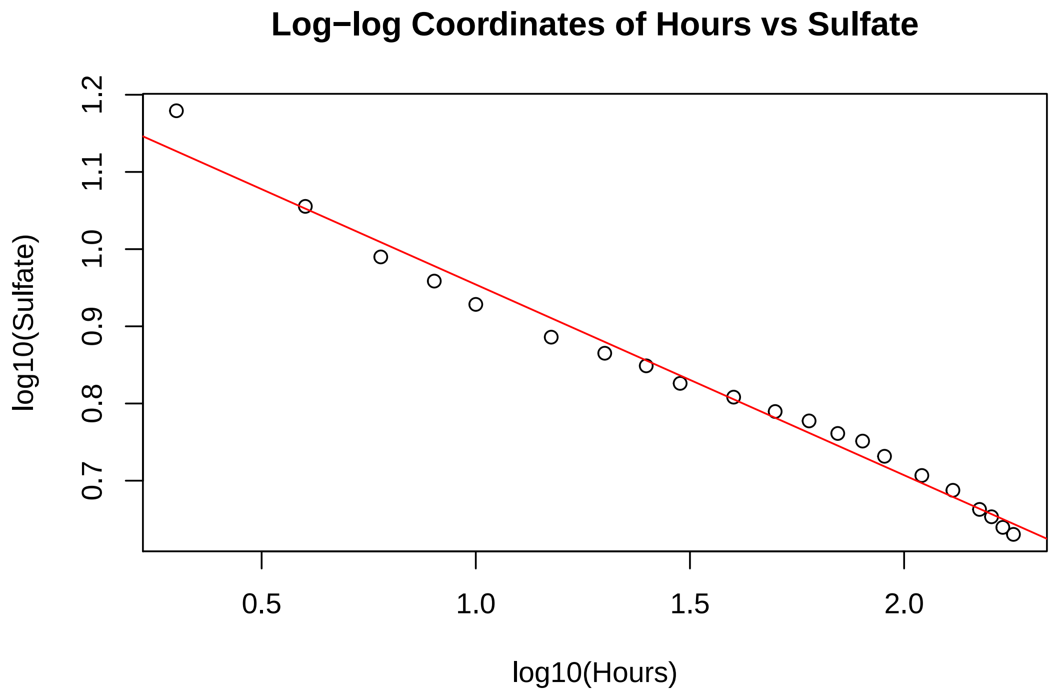
7.9.

At http://www.statsci.org/data/general/brunhild.html, you will find a dataset that measures the concentration of a sulfate in the blood of a baboon named Brunhilda as a function of time. Build a linear regression of the log of the concentration against the log of time.

(a) Prepare a plot showing (a) the data points and (b) the regression line in log-log coordinates.

Answer:

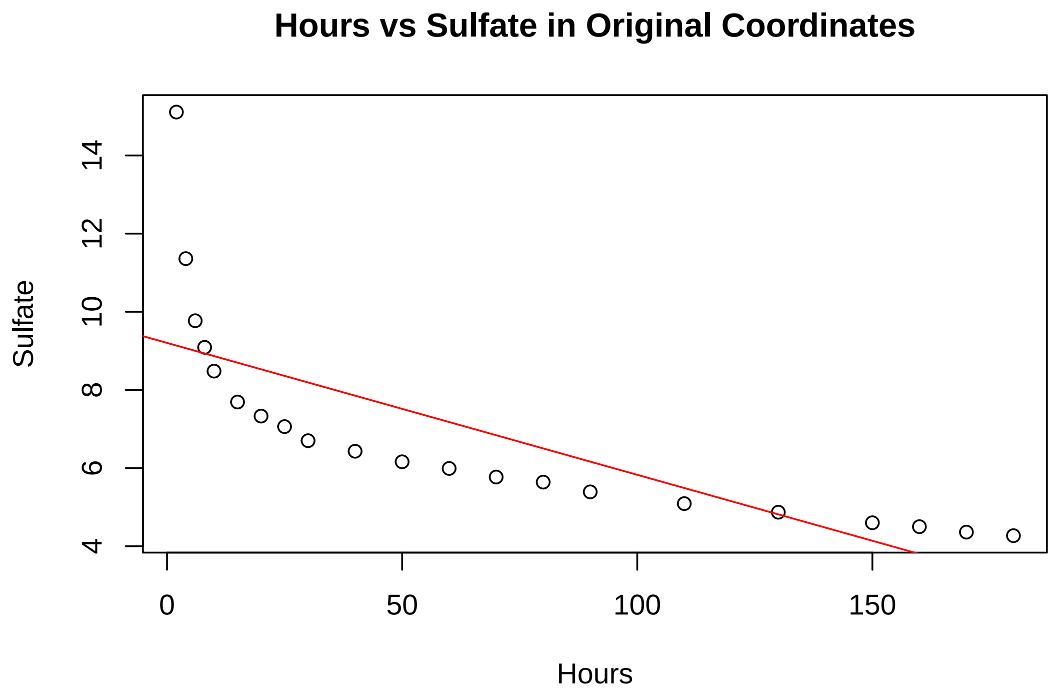
```
rm(list = ls())
setwd("C:/Users/98302/Desktop/hw4")
df <- read.table("brunhild.txt", header= TRUE)
plot(log10(df$Hours), log10(df$Sulfate), main = "Log-log Coordinates of Hours vs Sulfate",
     xlab = "log10(Hours)", ylab = "log10(Sulfate)")
loglog <- lm(log10(df$Sulfate)~log10(df$Hours))
abline(loglog, col="red")
```



(b) Prepare a plot showing (a) the data points and (b) the regression curve in the original coordinates.

Answer:

```r
plot(df$Hours, df$Sulfate, main = "Hours vs Sulfate in Original Coordinates",
     xlab = "Hours", ylab = "Sulfate")
original <- lm(df$Sulfate~df$Hours)
abline(original, col="red")
```

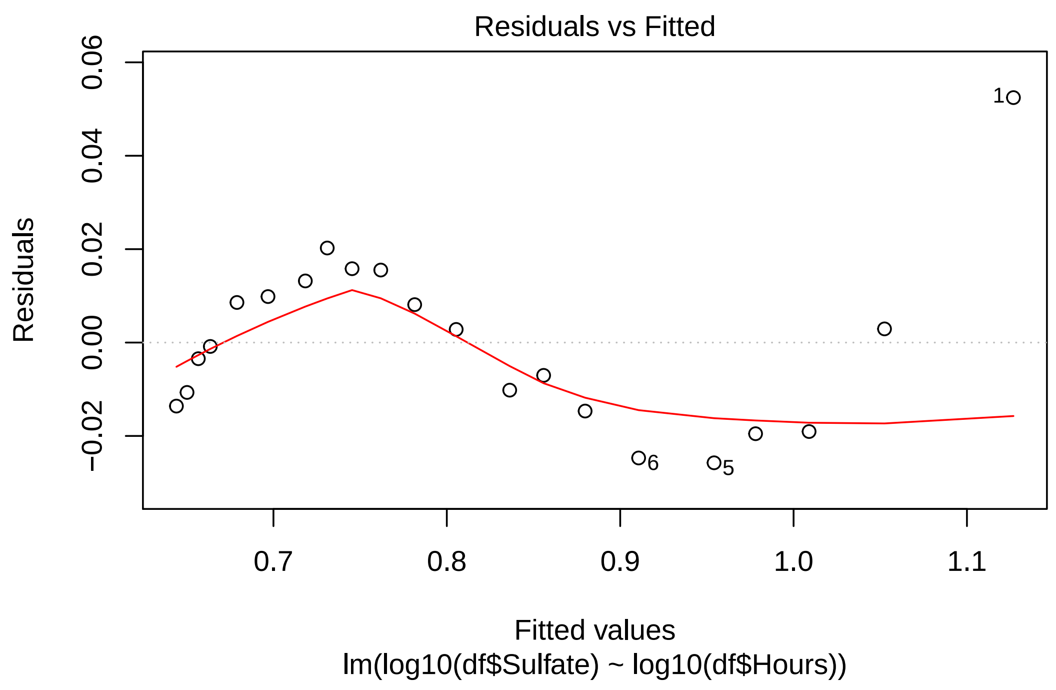


(c) Plot the residual against the fitted values in log-log and in original coordinates.

Answer:

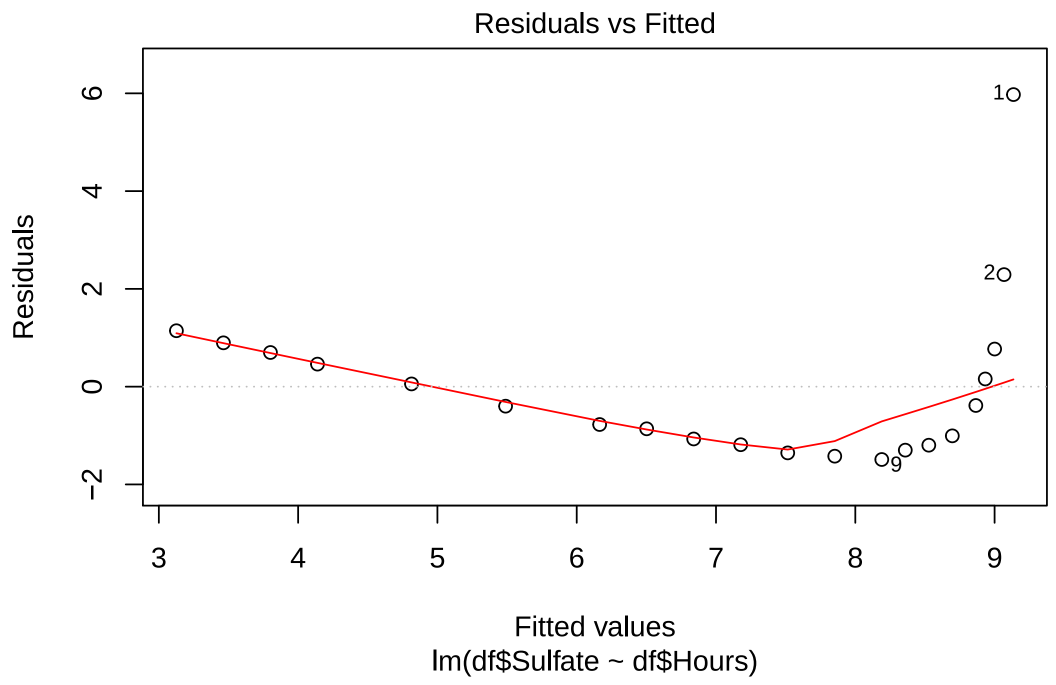
The residual against the fitted values in log-log coordinates.

```r
plot(loglog,1)
```



The residual against the fitted values in original coordinates.

```r
plot(original,1)
```

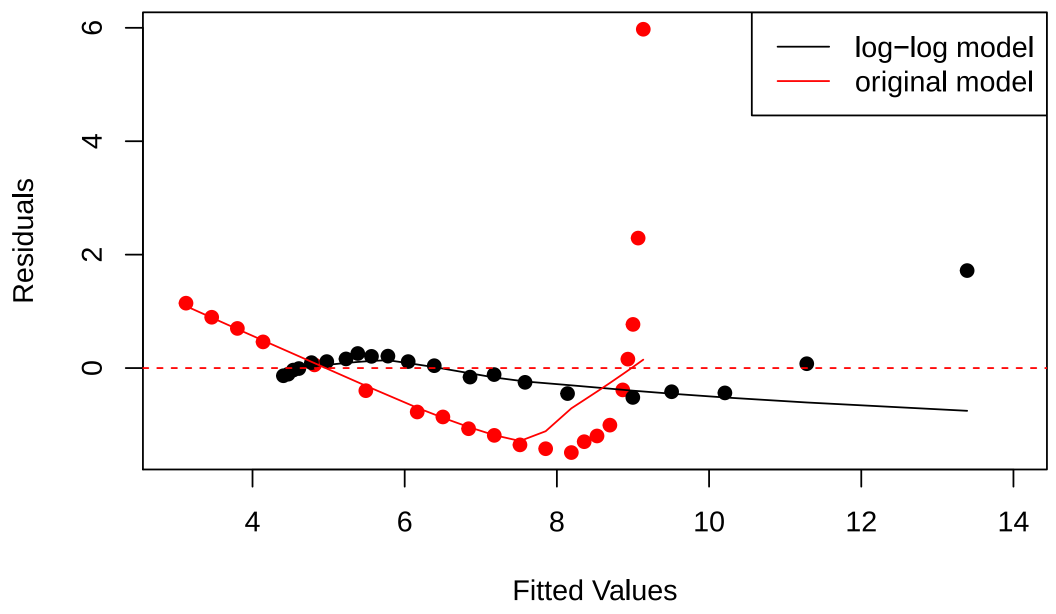


(d) Use your plots to explain whether your regression is good or bad and why.

Answer:

To compare these two model, we plot the residual against the fitted values in the same original coordinates.

```
residual_log = df$Sulfate - 10^loglog$fitted.values
plot(original$fitted.values,original$residuals,col="red",pch=19,xlim = c(3,14),xlab = "Fitted Values", 
points(10^loglog$fitted.values,residual_log,pch=19)
lines(lowess(10^loglog$fitted.values,residual_log))
lines(lowess(original$fitted.values,original$residuals),col="red")
legend("topright",c("log-log model","original model"),col=c("black","red"),lty  = 1)
abline(h = 0, col="red", lty = 2)
```



So we think the line of log-log model is closer to the line `residual = 0`. Also, we calculate the RSE(Mean Squared Error) in the same same original coordinates and R-squared.(Since the value of R^2 is not affected by the units of y, we just give the R^2 of these two model in different units)

```
RSE_loglog = sum(residual_log^2) / dim(df)[1]
RSE_original = sum(original$residuals^2) / dim(df)[1]
R2_loglog = summary(loglog)$r.squared
R2_original = summary(original)$r.squared
RSE_loglog
```

```
## [1] 0.1974717
```

```
RSE_original
```

```
## [1] 2.800099
```

```
R2_loglog
```

```
## [1] 0.9839251
```

```
R2_original
```

```
## [1] 0.5865673
```

We found the RSE of log-log model(=0.1974717) is smaller than the original model(=2.8000992), and also the R^2 of log-log model(=0.9839251) is larger than the original model(=0.5865673 < 0.7). Conbined with the result of residual vs fitted value plot, we think the log-log regression model is better, the original model isn't a good model.

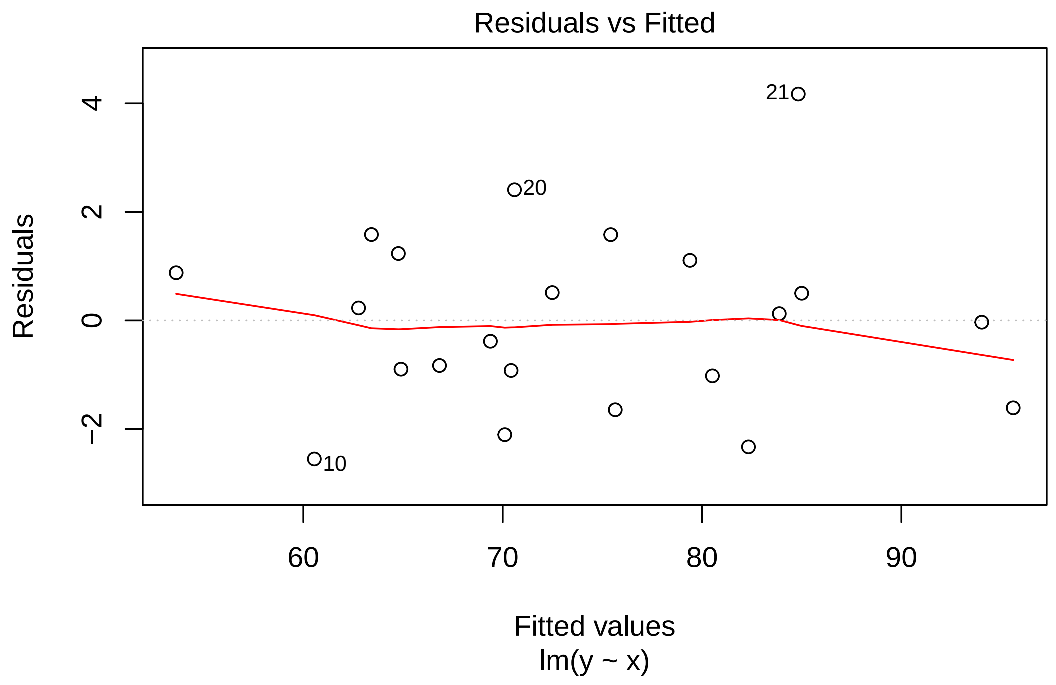
7.10.

At http://www.statsci.org/data/oz/physical.html, you will find a dataset of measurements by M. Larner, made in 1996. These measurements include body mass, and various diameters. Build a linear regression of predicting the body mass from these diameters.

(a) Plot the residual against the fitted values for your regression.

Answer:

```
rm(list = ls())
setwd("C:/Users/98302/Desktop/hw4")
data = read.table("physical.txt",header = T)
x = as.matrix(data[,-1])
y = data[,1]
result_original <- lm(y~x)
plot(result_original,1)
```



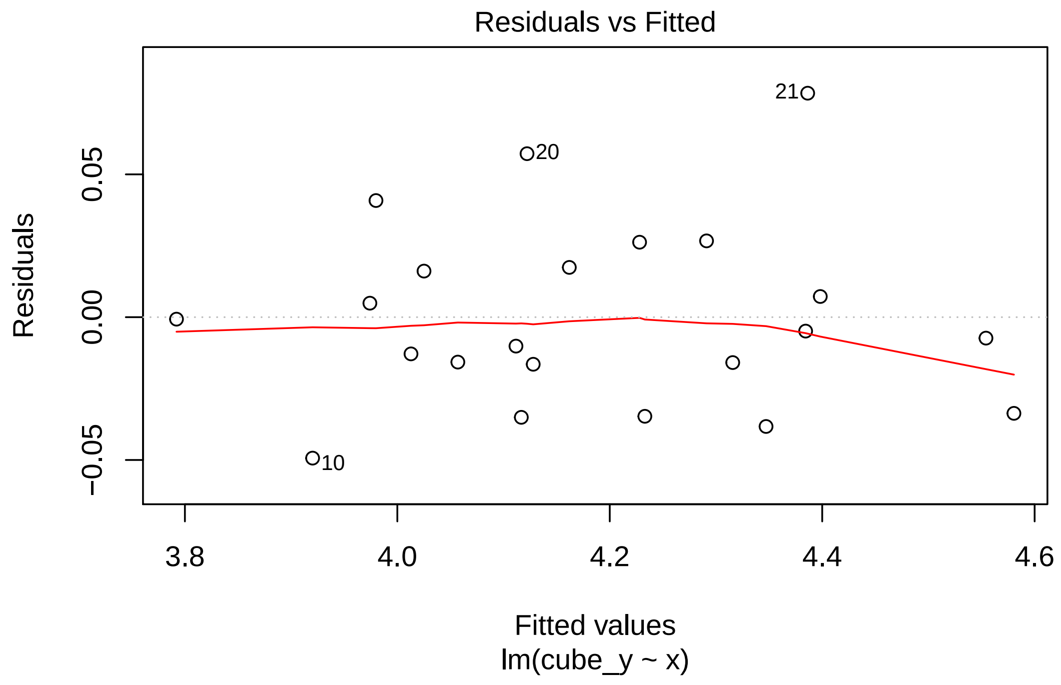
(b) Now regress the cube root of mass against these diameters. Plot the residual against the fitted values in both these cube root coordinates and in the original coordinates.

Answer:

```
cube_y = y^(1/3)
result_cube <- lm(cube_y~x)
```

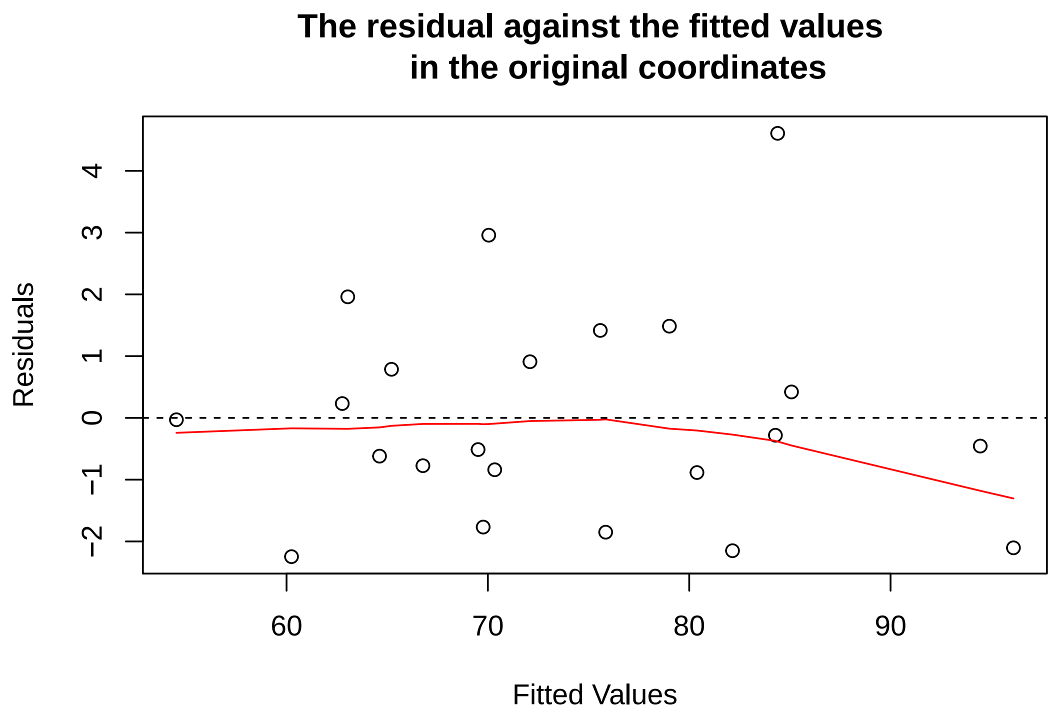
The residual against the fitted values in the cube root coordinates

```
plot(result_cube,1)
```



The residual against the fitted values in the original coordinates.

```
predict2_original = result_cube$f^3
residual2_original <- y - predict2_original
plot(predict2_original,residual2_original,xlab = "Fitted Values",
     ylab = "Residuals",main = "The residual against the fitted values
     in the original coordinates")
lines(lowess(predict2_original,residual2_original),col="red")
abline(h = 0, col="black", lty = 2)
```

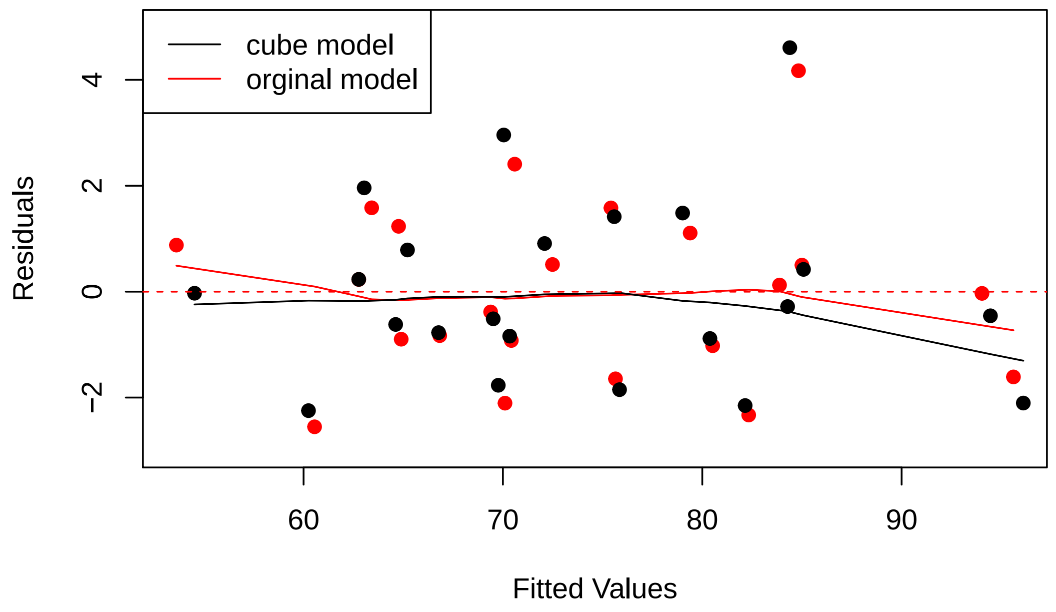


(c) Use your plots to explain which regression is better.

Answer:

To compare these two model, we plot the residual against the fitted values in the same original coordinates.

```
plot(result_original$fitted.values,result_original$residuals,col="red",pch=19,
     xlab = "Fitted Values", ylab = "Residuals",ylim=c(-3,5))
points(predict2_original,residual2_original,pch=19)
lines(lowess(result_original$fitted.values,result_original$residuals),col="red")
lines(lowess(predict2_original,residual2_original))
legend("topleft",c("cube model","orginal model"),col=c("black","red"),lty  = 1)
abline(h = 0, col="red", lty = 2)
```



So we think the plots of these two regression are almost the same. The cube model performs better when fitted value is small but performs worse when fitted value is large. Also, we calculate the RSE(Mean Squared Error) in the same same original coordinates and R-squared.(Since the value of R^2 is not affected by the units of y, we just give the R^2 of these two model in different units)

```
RSE_cube = sum(residual2_original^2) / dim(data)[1]
RSE_original = sum(result_original$residuals^2) / dim(data)[1]
R2_cube = summary(result_cube)$r.squared
R2_original = summary(result_original)$r.squared
RSE_cube
```

```
## [1] 2.880683
```

```
RSE_original
```

```
## [1] 2.614712
```

```
R2_cube
```

```
## [1] 0.9758476
```

```
R2_original
```

```
## [1] 0.9772107
```

We found the RSE of original model(=2.6147117) is smaller than the cube model(=2.8806831), and also the R^2 of original model(=0.9772107) is larger than the cube model(=0.9758476). Conbined with the result of residual vs fitted value plot, we think both regression curves fit most of the data points, so these two models are both good regression. And their performance(R^2,RSM,residual) are almost the same, maybe the original model is a litte better.(I think the result may due to that there are 10 features but only 22 examples, we need more examples to compare these two regression.)

7.11.

At https://archive.ics.uci.edu/ml/datasets/Abalone, you will find a dataset of measurements by W. J. Nash, T. L. Sellers, S. R. Talbot, A. J. Cawthorn and W. B. Ford, made in 1992. These are a variety of measurements of blacklip abalone (Haliotis rubra; delicious by repute) of various ages and genders.

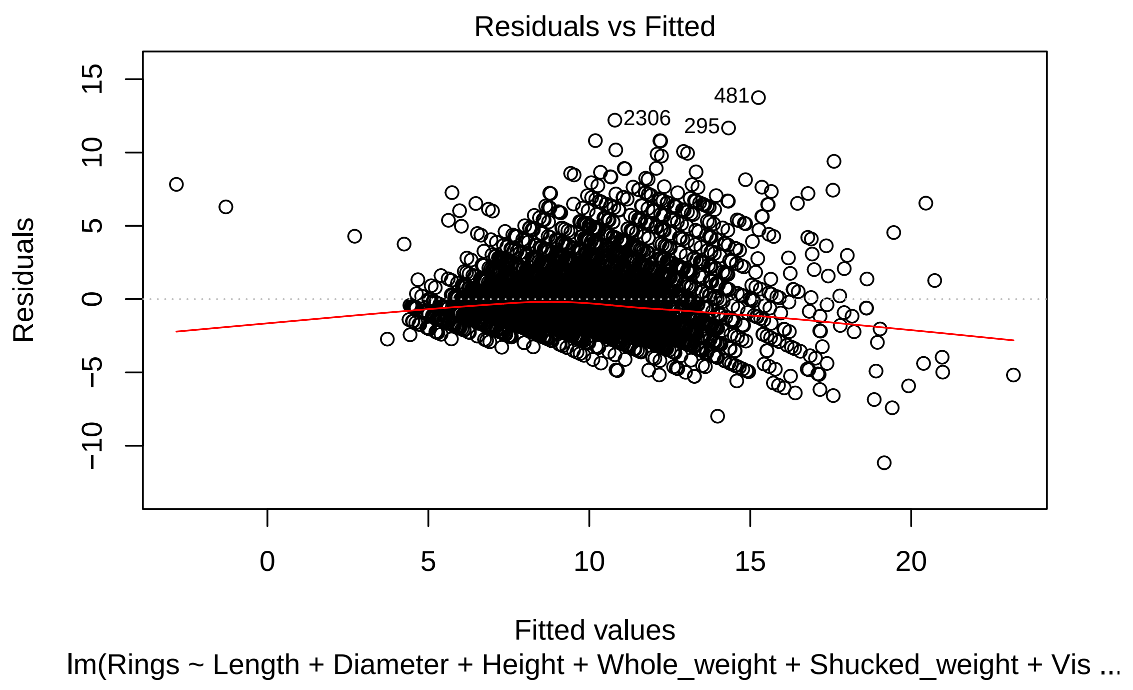
(a) Build a linear regression predicting the age from the measurements, ignoring gender. Plot the residual against the fitted values.

Answer:

```r
rm(list = ls())
setwd("C:/Users/98302/Desktop/hw4")
df <- read.csv("abalone.data", header=F)
colnames(df) = c("Sex", "Length","Diameter","Height","Whole_weight",
                 "Shucked_weight","Viscera_weight","Shell_weight","Rings")
fit.lm <- lm(formula = Rings ~ Length+Diameter+Height+Whole_weight+
                Shucked_weight+Viscera_weight+Shell_weight, data=df)
summary(fit.lm)
```

```
##
## Call:
## lm(formula = Rings ~ Length + Diameter + Height + Whole_weight +
##     Shucked_weight + Viscera_weight + Shell_weight, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.1632  -1.3613  -0.3885   0.9054  13.7440
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2.9852     0.2691  11.092  < 2e-16 ***
## Length          -1.5719     1.8248  -0.861    0.389
## Diameter        13.3609     2.2371   5.972 2.53e-09 ***
## Height          11.8261     1.5481   7.639 2.70e-14 ***
## Whole_weight     9.2474     0.7326  12.622  < 2e-16 ***
## Shucked_weight -20.2139     0.8233 -24.552  < 2e-16 ***
## Viscera_weight  -9.8297     1.3040  -7.538 5.82e-14 ***
## Shell_weight     8.5762     1.1367   7.545 5.54e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.218 on 4169 degrees of freedom
## Multiple R-squared:  0.5276, Adjusted R-squared:  0.5268
## F-statistic: 665.2 on 7 and 4169 DF,  p-value: < 2.2e-16
```

```r
plot(fit.lm,1)
```



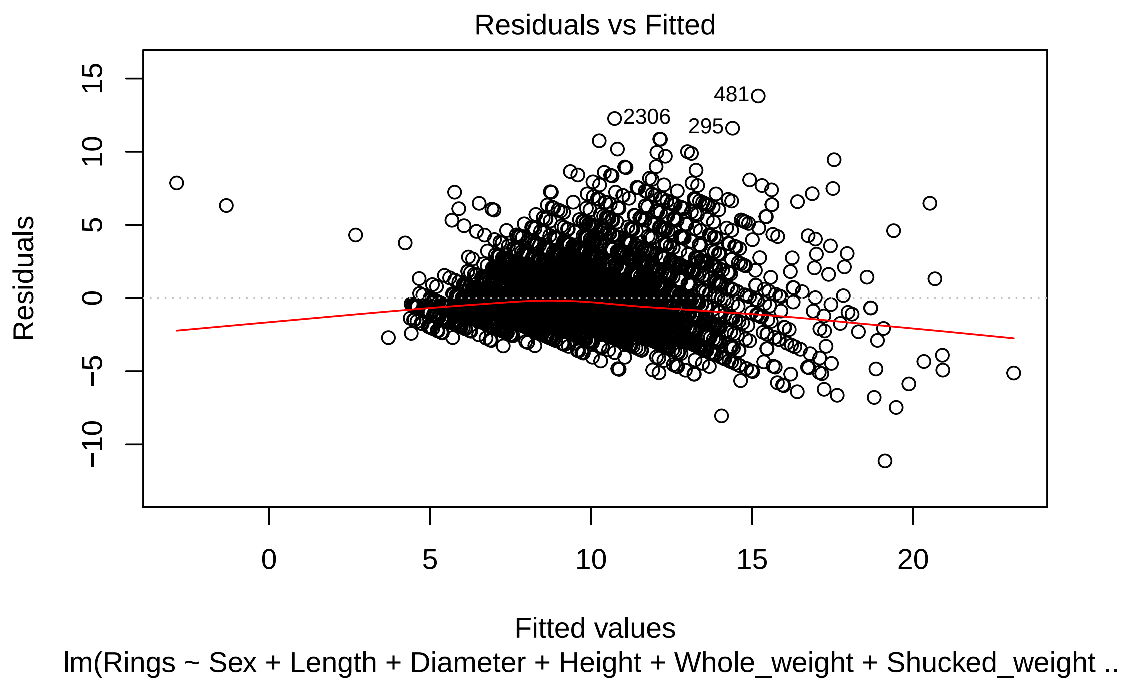
(b) Build a linear regression predicting the age from the measurements, including gender. There are three levels for gender; I'm not sure whether this has to do with abalone biology or difficulty in determining gender. You can represent gender numerically by choosing 1 for one level, 0 for another, and -1 for the third. Plot the residual against the fitted values.

Answer:

```r
df$Sex = as.character(df$Sex)
df$Sex[df$Sex %in% "M"] <- 1
df$Sex[df$Sex %in% "F"] <- -1
df$Sex[df$Sex %in% "I"] <- 0
df$Sex = as.numeric(df$Sex)
fit_sex.lm <- lm(formula = Rings ~ Sex+Length+Diameter+Height+
                    Whole_weight+Shucked_weight+Viscera_weight+Shell_weight, data=df)
summary(fit_sex.lm)
```

```
##
## Call:
## lm(formula = Rings ~ Sex + Length + Diameter + Height + Whole_weight +
##     Shucked_weight + Viscera_weight + Shell_weight, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.1275  -1.3636  -0.3918   0.9034  13.8092
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2.96304    0.26948  10.995  < 2e-16 ***
## Sex              0.06368    0.04195   1.518    0.129
## Length          -1.57721    1.82448  -0.864    0.387
## Diameter        13.42050    2.23707   5.999 2.15e-09 ***
## Height          11.86439    1.54809   7.664 2.23e-14 ***
## Whole_weight     9.25049    0.73253  12.628  < 2e-16 ***
## Shucked_weight -20.28094    0.82436 -24.602  < 2e-16 ***
## Viscera_weight  -9.76110    1.30458  -7.482 8.87e-14 ***
## Shell_weight     8.58057    1.13656   7.550 5.33e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.217 on 4168 degrees of freedom
## Multiple R-squared:  0.5279, Adjusted R-squared:  0.527
## F-statistic: 582.6 on 8 and 4168 DF,  p-value: < 2.2e-16
```

```
plot(fit_sex.lm,1)
```



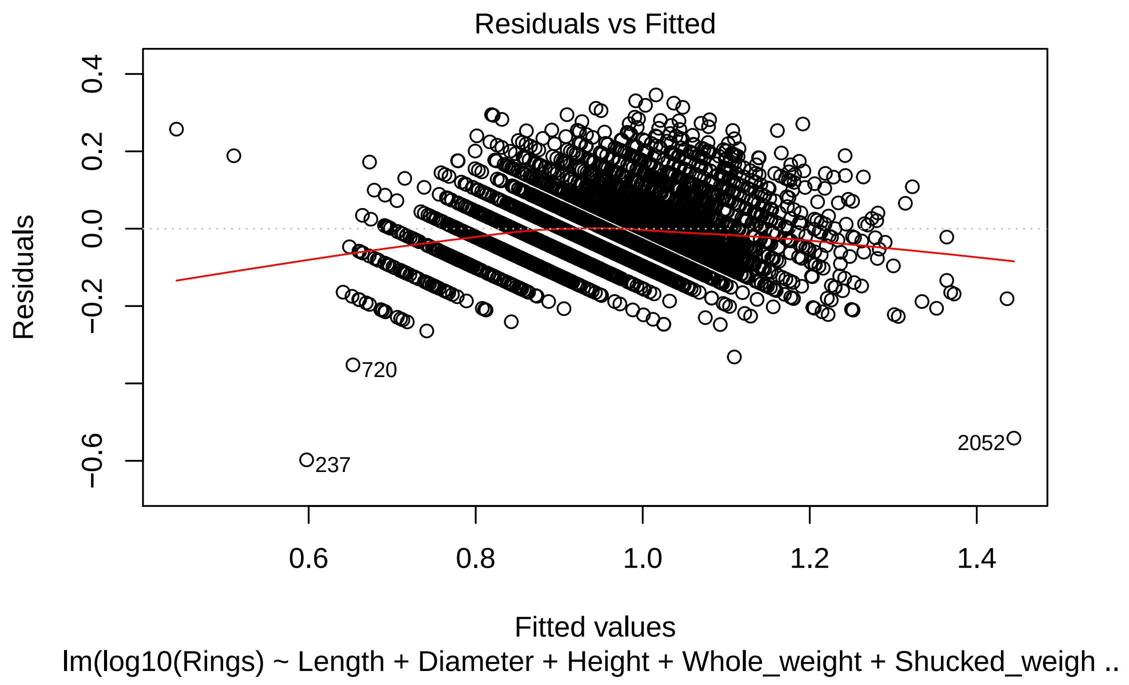
(c) Now build a linear regression predicting the log of age from the measurements, ignoring gender. Plot the residual against the fitted values.

Answer:

```
log_fit.lm <- lm(formula = log10(Rings) ~ Length+Diameter+Height+Whole_weight+
                    Shucked_weight+Viscera_weight+Shell_weight, data=df)
summary(log_fit.lm)
```

```
##
## Call:
## lm(formula = log10(Rings) ~ Length + Diameter + Height + Whole_weight +
##     Shucked_weight + Viscera_weight + Shell_weight, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59753 -0.05962 -0.00969  0.04869  0.34578
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.53831    0.01085  49.611  < 2e-16 ***
## Length          0.17663    0.07357   2.401   0.0164 *
## Diameter        0.73050    0.09019   8.099 7.20e-16 ***
## Height          0.57622    0.06242   9.232  < 2e-16 ***
## Whole_weight    0.27755    0.02954   9.396  < 2e-16 ***
## Shucked_weight -0.74016    0.03319 -22.298  < 2e-16 ***
## Viscera_weight -0.32631    0.05257  -6.207 5.94e-10 ***
## Shell_weight    0.25534    0.04583   5.571 2.69e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08942 on 4169 degrees of freedom
## Multiple R-squared:  0.5855, Adjusted R-squared:  0.5848
## F-statistic: 841.2 on 7 and 4169 DF,  p-value: < 2.2e-16
```

```
plot(log_fit.lm,1)
```



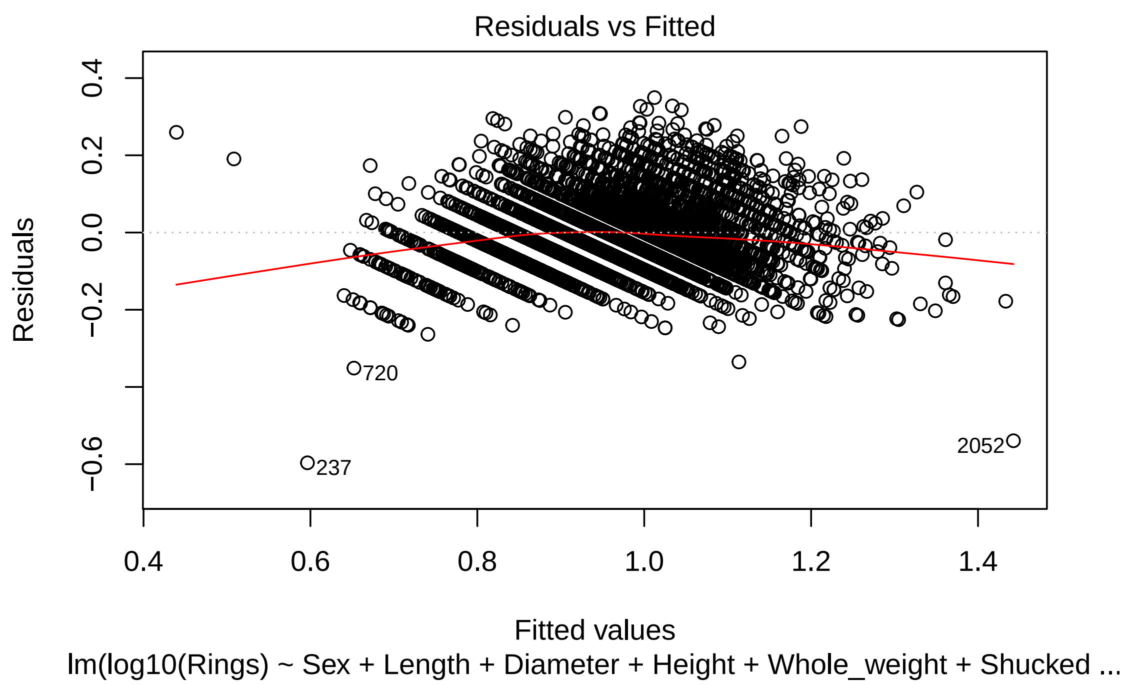
(d) Now build a linear regression predicting the log age from the measurements, including gender, represented as above. Plot the residual against the fitted values.

Answer:

```
log_fit_sex.lm <- lm(formula = log10(Rings) ~ Sex+Length+Diameter+Height+Whole_weight+
                         Shucked_weight+Viscera_weight+Shell_weight, data=df)
summary(log_fit_sex.lm)
```

```
##
## Call:
## lm(formula = log10(Rings) ~ Sex + Length + Diameter + Height +
##     Whole_weight + Shucked_weight + Viscera_weight + Shell_weight,
##     data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.59645 -0.05956 -0.01006  0.04861  0.34941
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.537040   0.010862  49.443  < 2e-16 ***
## Sex             0.003656   0.001691   2.162   0.0307 *
## Length          0.176320   0.073538   2.398   0.0165 *
## Diameter        0.733920   0.090169   8.139 5.19e-16 ***
## Height          0.578421   0.062398   9.270  < 2e-16 ***
## Whole_weight    0.277725   0.029526   9.406  < 2e-16 ***
## Shucked_weight -0.744013   0.033227 -22.392  < 2e-16 ***
## Viscera_weight -0.322376   0.052583  -6.131 9.56e-10 ***
## Shell_weight    0.255584   0.045811   5.579 2.57e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08938 on 4168 degrees of freedom
## Multiple R-squared:  0.5859, Adjusted R-squared:  0.5851
## F-statistic: 737.3 on 8 and 4168 DF,  p-value: < 2.2e-16
```

```
plot(log_fit_sex.lm,1)
```

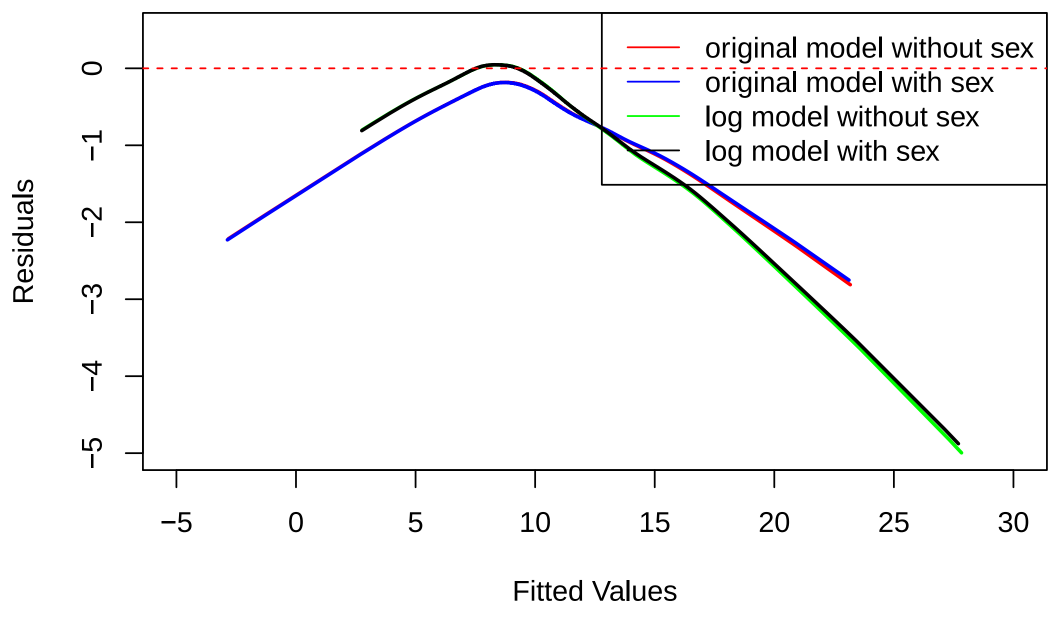


(e) It turns out that determining the age of an abalone is possible, but difficult (you section the shell, and count rings). Use your plots to explain which regression you would use to replace this procedure, and why.

Answer:

To compare these four model, we plot the lowess curves of residual against the fitted values in the same original coordinates.

```
predict_log = 10^log_fit.lm$fitted.values
residual_log = df$Rings - predict_log
predict_log_sex = 10^log_fit_sex.lm$fitted.values
residual_log_sex = df$Rings - predict_log_sex
plot(lowess(fit.lm$fitted.values,fit.lm$residuals),col="red",lty = 1,
     lwd = 2,type = "l",xlim=c(-5,30),ylim=c(-5,0.5), xlab = "Fitted Values",
     ylab = "Residuals")
lines(lowess(fit_sex.lm$fitted.values,fit_sex.lm$residuals),col="blue",
      lty = 1,  lwd = 2)
lines(lowess(predict_log ,residual_log),col="green",lty = 1, lwd = 2)
lines(lowess(predict_log_sex,residual_log_sex),col="black",lty = 1,  lwd = 2)
legend("topright",c("original model without sex","original model with sex",
      "log model without sex","log model with sex"), col=c("red","blue",
      "green","black"),lty  = 1)
abline(h = 0, col="red", lty = 2)
```



So we think the plots of these two regression are almost the same. Also, we calculate the RSE(Mean Squared Error) in the same same original coordinates and R-squared.(Since the value of R^2 is not affected by the units of y, we just give the R^2 of these two model in different units)

```
RSE_original = sum(fit.lm$residuals^2) / dim(df)[1]
RSE_original_sex = sum(fit_sex.lm$residuals^2) / dim(df)[1]
RSE_log = sum(residual_log^2) / dim(df)[1]
RSE_log_sex = sum(residual_log_sex^2) / dim(df)[1]

R2_original = summary(fit.lm)$r.squared
R2_original_sex = summary(fit_sex.lm)$r.squared
R2_log = summary(log_fit.lm)$r.squared
R2_log_sex = summary(log_fit_sex.lm)$r.squared

RSE_original
```

```
## [1] 4.909237
```

```
RSE_original_sex
```

```
## [1] 4.906524
```

```
RSE_log
```

```
## [1] 5.31064
```

```
RSE_log_sex
```

```
## [1] 5.302952
```

```
R2_original
```

```
## [1] 0.5276299
```

```r
R2_original_sex
```

```
## [1] 0.5278909
```

```r
R2_log
```

```
## [1] 0.5854699
```

```r
R2_log_sex
```

```
## [1] 0.5859342
```

Based on the 4 residual vs fitted plots above, we observe that the 4 fitted trends are relatively close to each other. From the 4 plots, we see that the log models are closer to 0 (less residuals) from 0 to 15 and slightly higher residuals as fitted values get bigger. It is difficult to spot the systematic error purely base on the plots.

By comparing the numerical data, we observe that the log models give higher R squared than the original model (more accurate prediction), while the original models show less residual standard error than the log model. The log model with sex appears to be slightly better than the other three, even though the difference is insignificant.

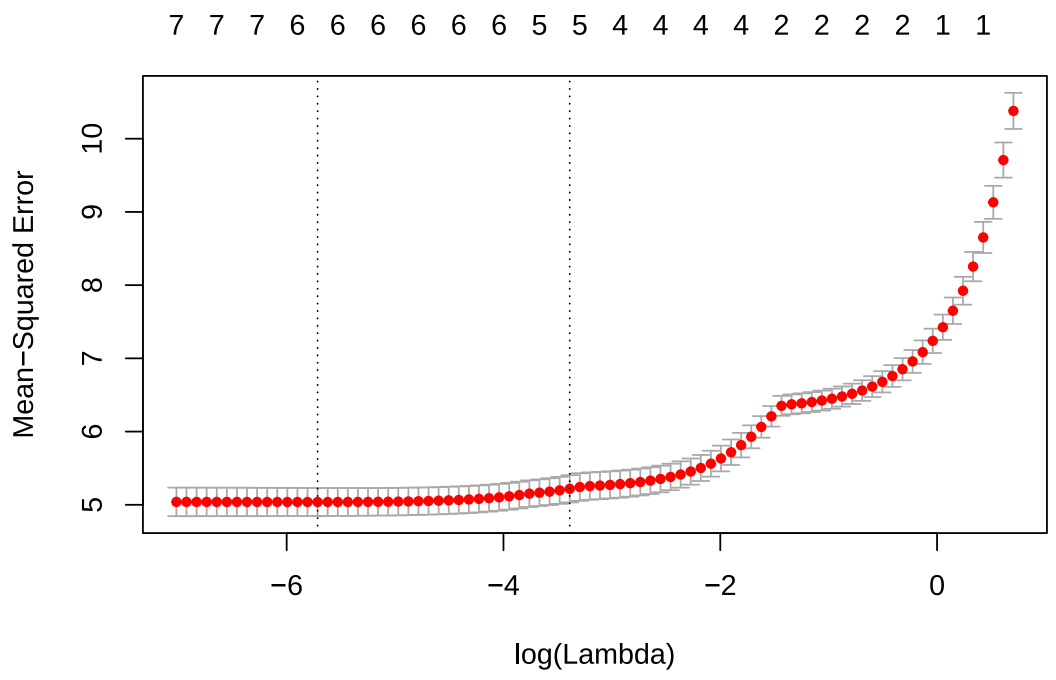
(f) Can you improve these regressions by using a regularizer? Use glmnet to obtain plots of the cross-validated prediction error.

Answer:

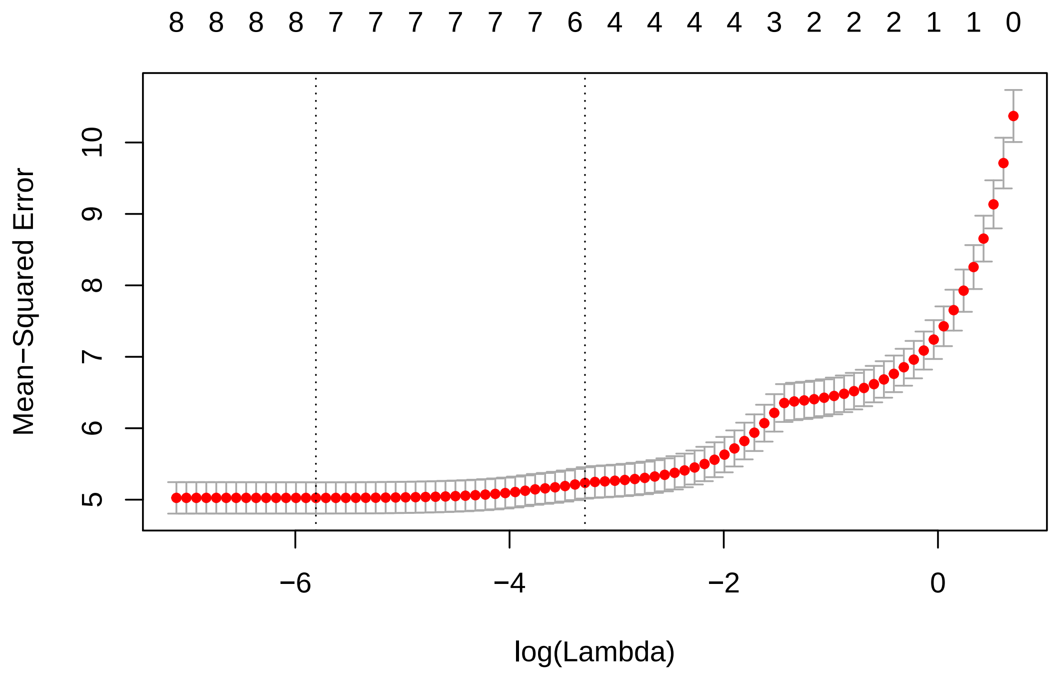
```r
library(glmnet)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:base':
##
##     crossprod, tcrossprod

## Loading required package: foreach

## Loaded glmnet 2.0-5
```

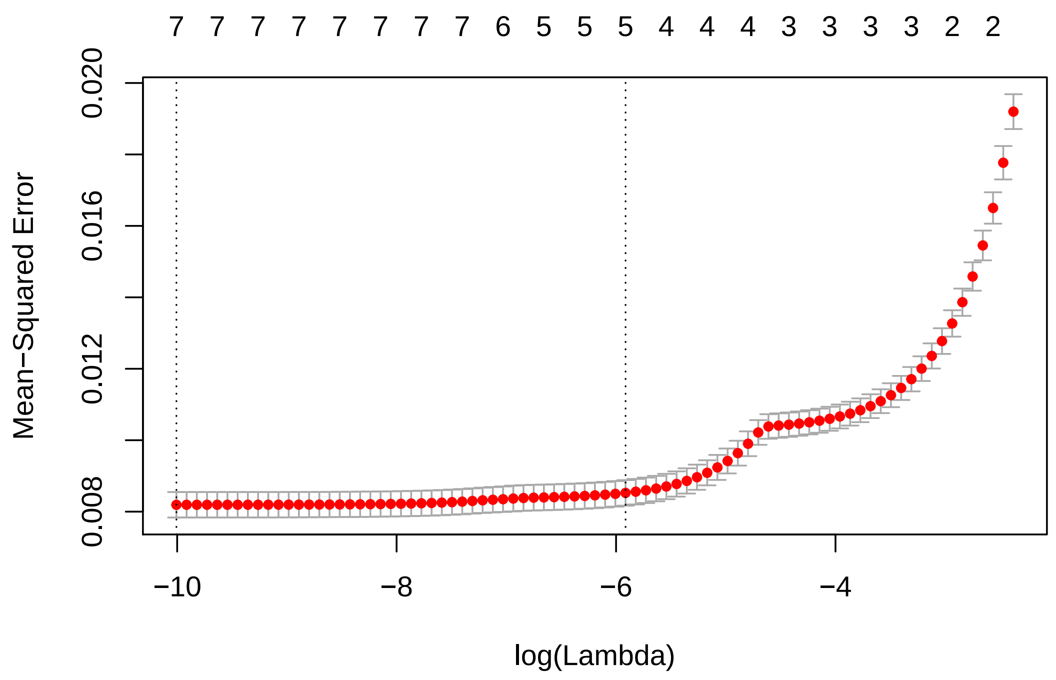
```r
x1 <- as.matrix(data.frame(df[2:8]))
x2 <- as.matrix(data.frame(df[1:8]))
y <- as.matrix(data.frame(df$Rings))

result1.cv <- cv.glmnet(x = x1, y = y, family = "gaussian", alpha = 1, nfold = 10)
result2.cv <- cv.glmnet(x = x2, y = y, family = "gaussian", alpha = 1, nfold = 10)
result3.cv <- cv.glmnet(x = x1, y = log10(y), family = "gaussian", alpha = 1, nfold = 10)
result4.cv <- cv.glmnet(x = x2, y = log10(y), family = "gaussian", alpha = 1, nfold = 10)

plot(result1.cv)
```



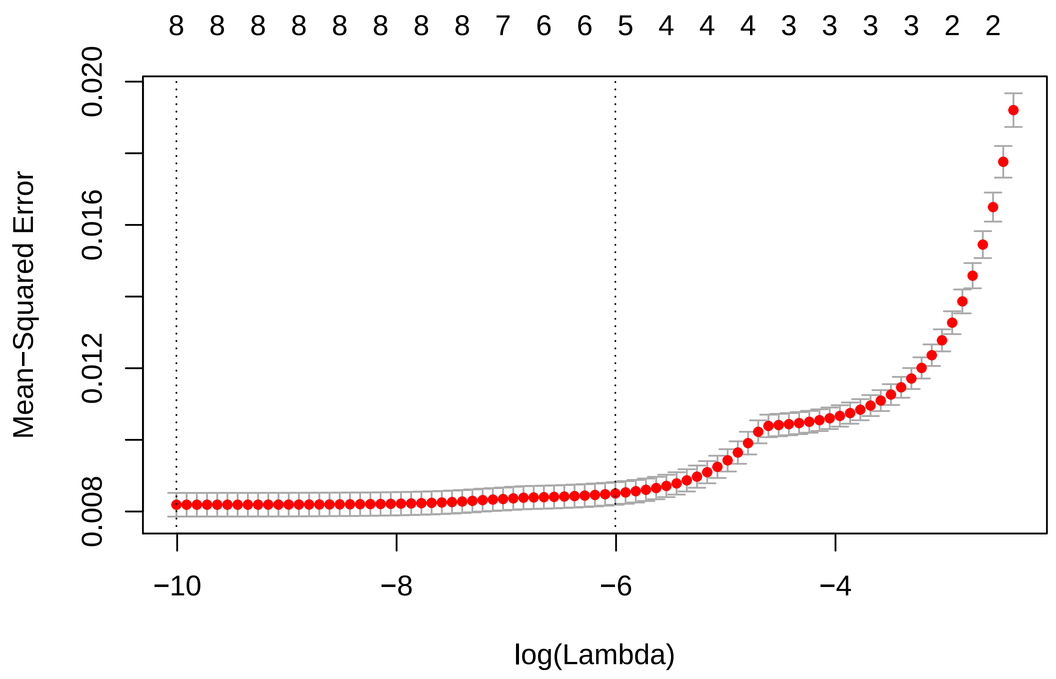
```
plot(result2.cv)
```



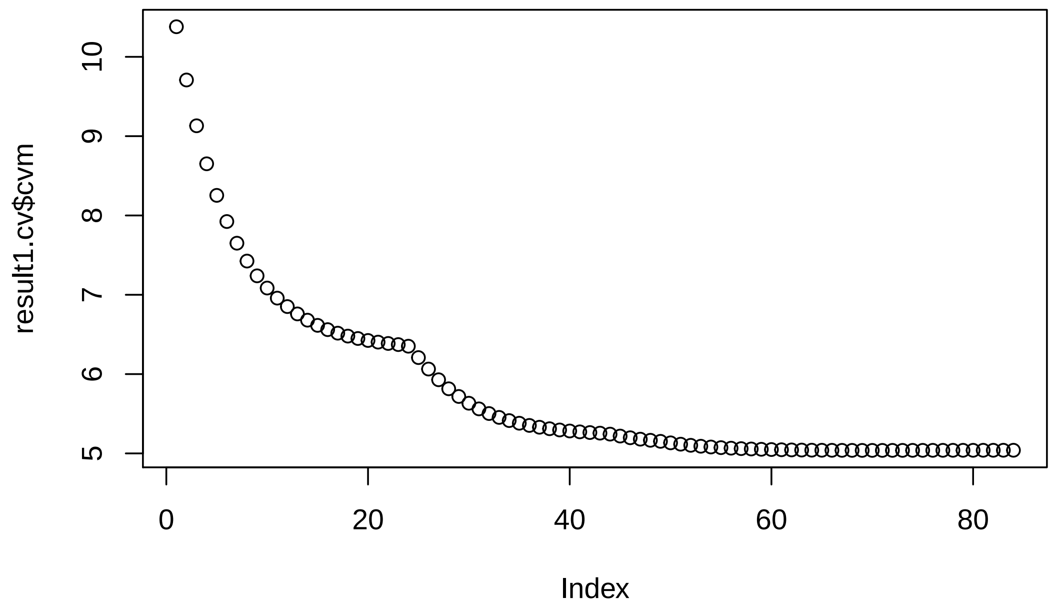
```
plot(result3.cv)
```



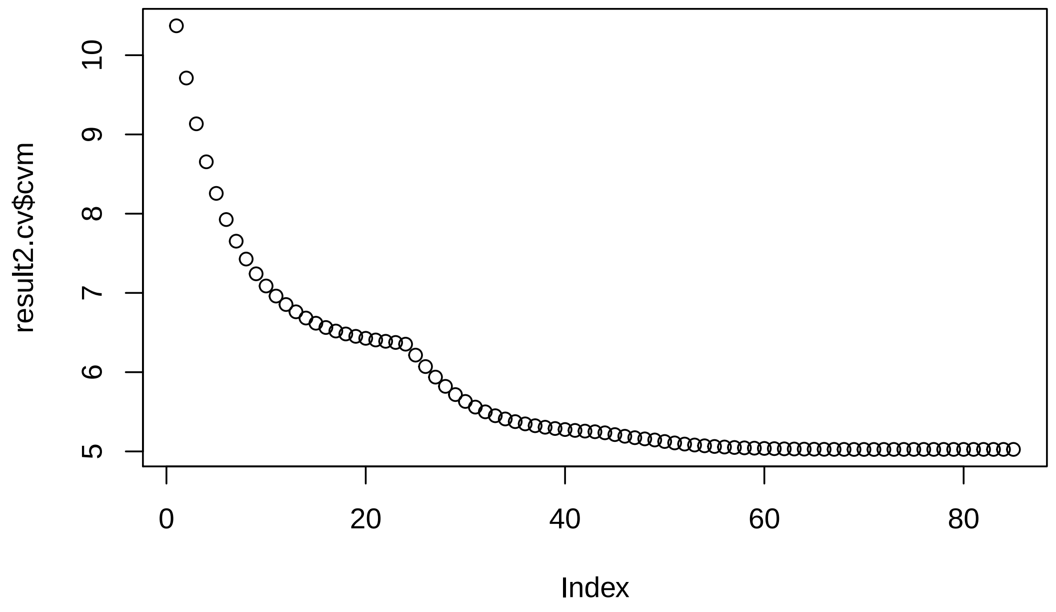
```
plot(result4.cv)
```



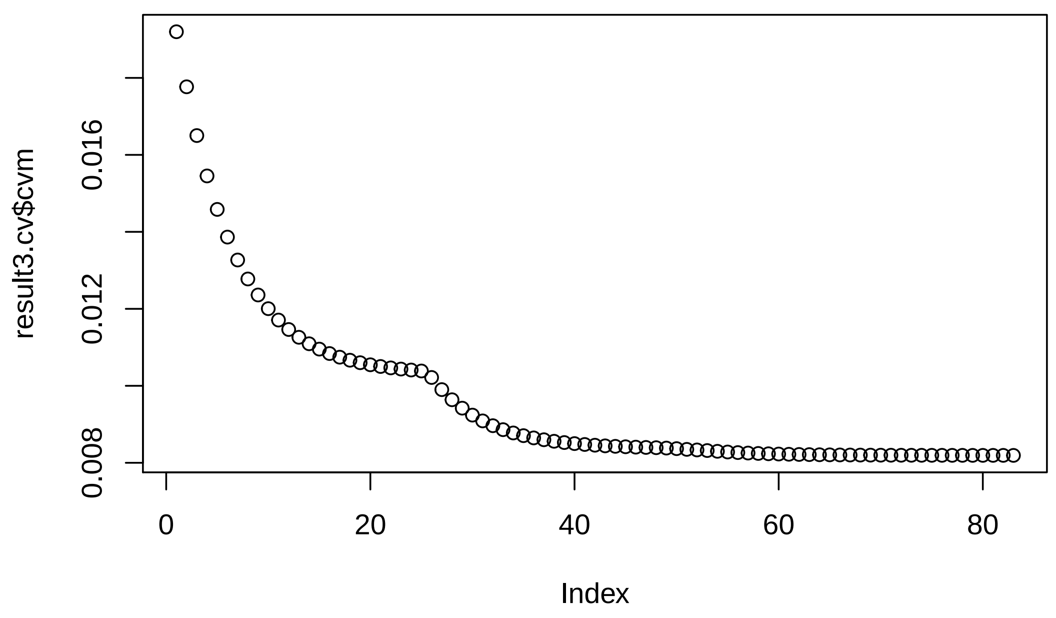
```r
plot(result1.cv$cvm)
```



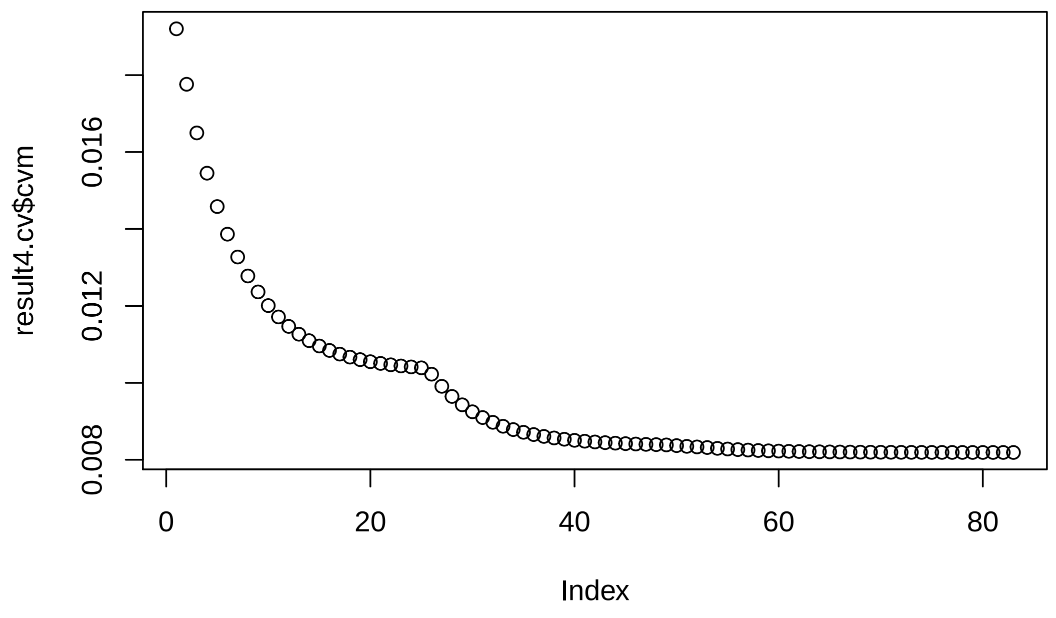
```r
plot(result2.cv$cvm)
```



```r
plot(result3.cv$cvm)
```



```r
plot(result4.cv$cvm)
```



```r
result1.pred <- predict(result1.cv, s = result1.cv$lambda.1se, newx = x1)
result2.pred <- predict(result2.cv, s = result2.cv$lambda.1se, newx = x2)
result3.pred <- predict(result3.cv, s = result3.cv$lambda.1se, newx = x1)
result4.pred <- predict(result4.cv, s = result4.cv$lambda.1se, newx = x2)

RSE1 = mean((result1.pred - y)^2)
RSE2 = mean((result2.pred - y)^2)
RSE3 = mean((10^result3.pred - y)^2)
RSE4 = mean((10^result4.pred - y)^2)

RSE1
```

```
## [1] 5.097416
```

```r
RSE2
```

```
## [1] 5.121213
```

```r
RSE3
```

```
## [1] 5.579606
```

```r
RSE4
```

```
## [1] 5.563596
```

```r
R2_1 = var(result1.pred) / var(y)
R2_2 = var(result2.pred) / var(y)
R2_3 = var(result3.pred) / var(log10(y))
R2_4 = var(result4.pred) / var(log10(y))
as.numeric(R2_1)
```

```
## [1] 0.4658981
```

```r
as.numeric(R2_2)
```

```
## [1] 0.4617833
```

```r
as.numeric(R2_3)
```

```
## [1] 0.4991395
```

```r
as.numeric(R2_4)
```

```
## [1] 0.5053233
```

According to the plots, error increases with lambda, so when lambda = 0, there is a smallest MSE which means without a regularizer. Also, we can find that when model is with a regularizer, its RSE is larger and its R^2 is smaller. So it did not improve these regressions by using a regularizer. But, with a regularizer, may it will perform better with unseen data.