HW5 Report

Ashish Kumar // ashishk2@illinois.edu (mailto:ashishk2@illinois.edu)

Problem 7.9

Let’s read the data in a dataframe. I ran the linear regression model on normal data and log scaled data. The R-Squared value of both models is reported below.

```
rm(list = ls())
library(stats)
library(ggplot2)
data = read.table("http://www.statsci.org/data/general/brunhild.txt", header = TRUE,
sep = '\t')

reg.lm = lm(data$Sulfate ~ data$Hours, data=data)
reg.resid = resid(reg.lm)
print(paste("R-Squared :", summary(reg.lm)$r.squared))
```

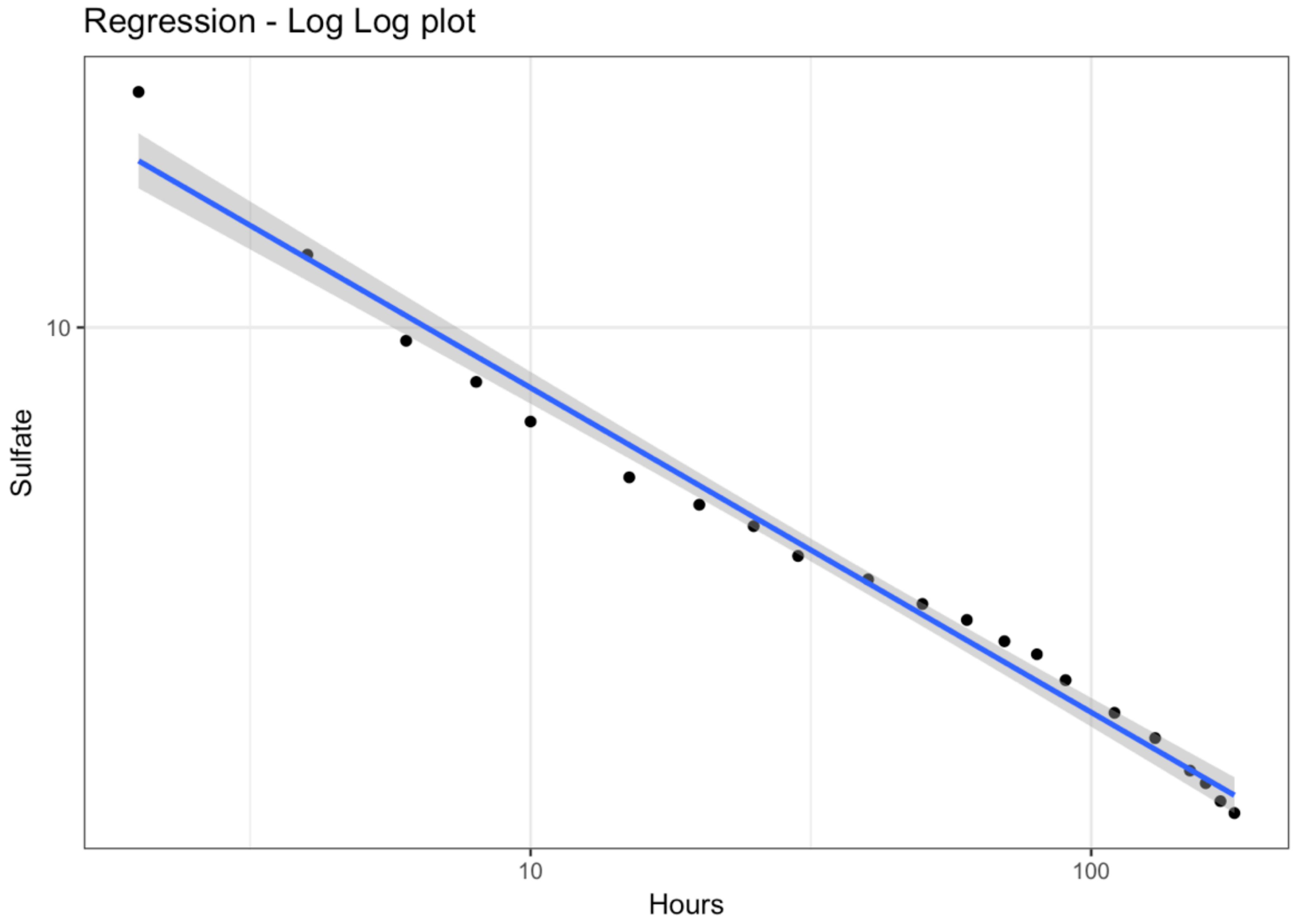
```
## [1] "R-Squared : 0.58656725751922"
```

```
data2 = as.data.frame(data)
data2$Hours = log(data$Hours)
data2$Sulfate = log(data$Sulfate)
reg2.lm = lm(data2$Sulfate ~ data2$Hours, data=data2)
reg2.resid = resid(reg2.lm)
print(paste("R-Squared (log) :", summary(reg2.lm)$r.squared))
```

```
## [1] "R-Squared (log) : 0.983925093100738"
```

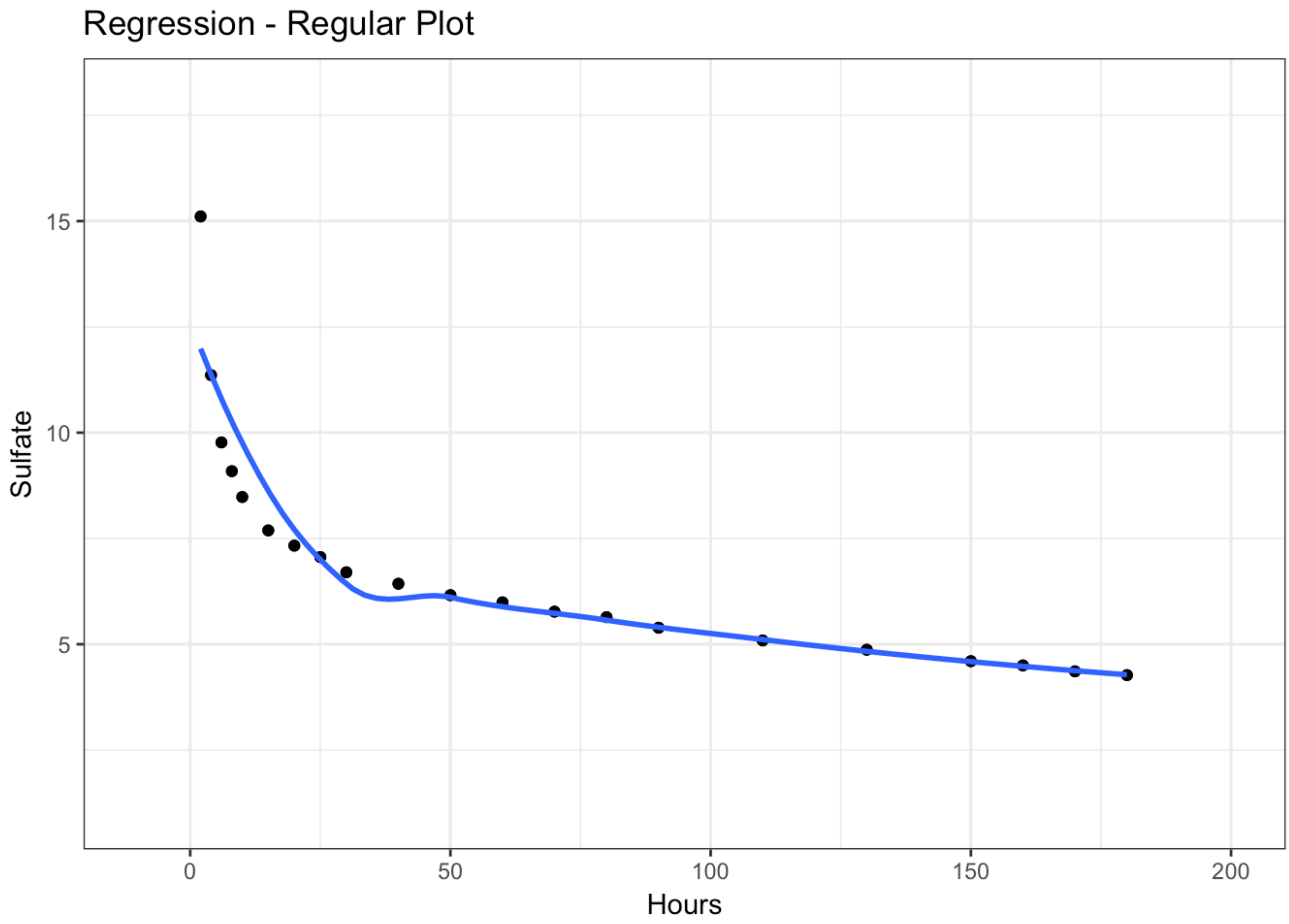
(a) Prepare a plot showing (a) the data points and (b) the regression line in log-log coordinates.

```
p1 = qplot(data$Hours, data$Sulfate,  xlab="Hours", ylab="Sulfate",
           main = "Regression - Log Log plot") + geom_smooth(method = lm)
p1 = p1 + scale_x_log10()
p1 = p1 + scale_y_log10()
p1 = p1 + theme_bw() + theme(panel.background = element_blank())
p1
```



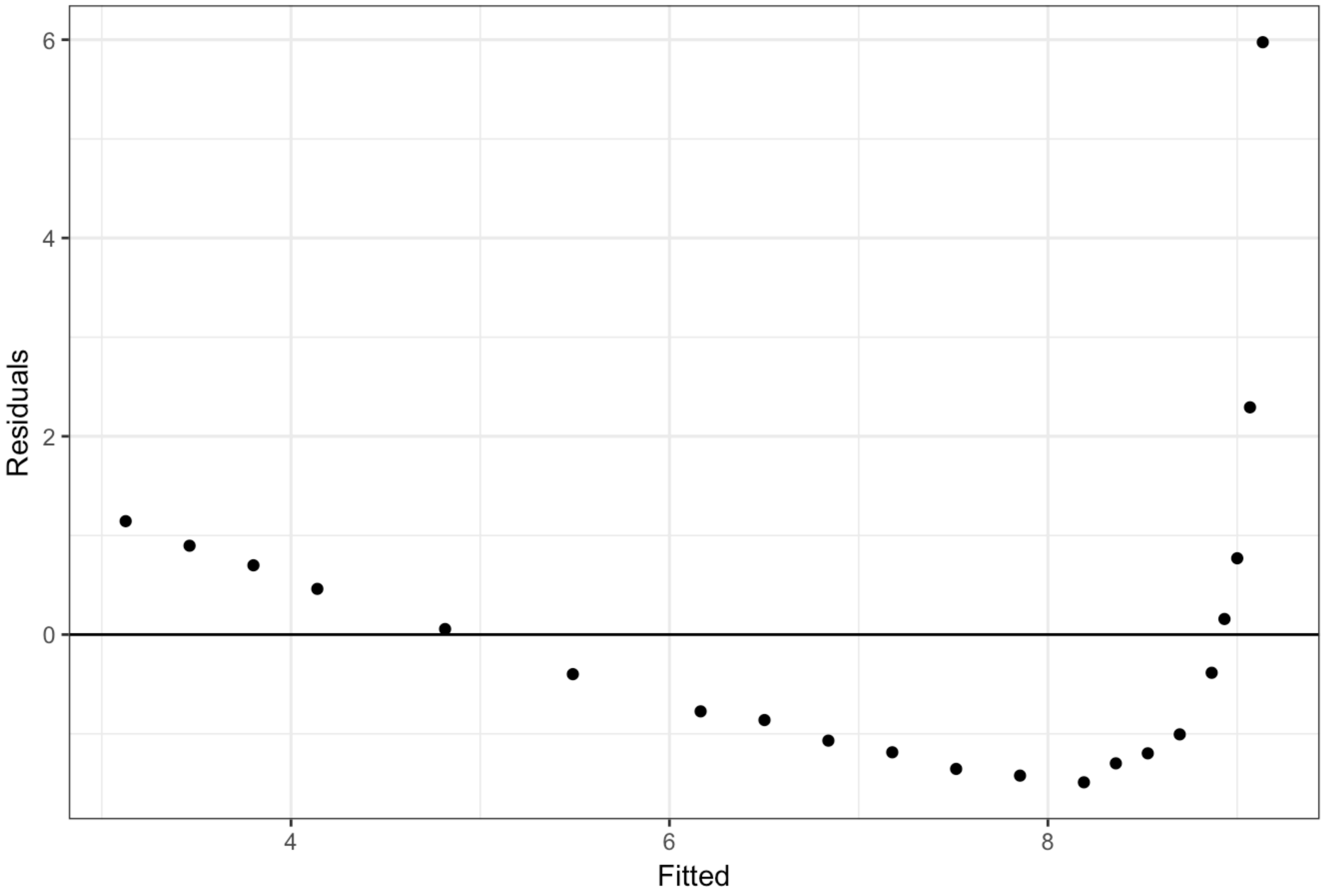
(b) Prepare a plot showing (a) the data points and (b) the regression curve in the original coordinates.

```
p2 = qplot(data$Hours, data$Sulfate, xlab="Hours", ylab="Sulfate", main="Regression -
Regular Plot",
      xlim = c(-10, 200), ylim = c(1, 18)) + geom_point() +
            geom_smooth(method = "loess",  se = FALSE)
p2 = p2 + theme_bw() + theme(panel.background = element_blank())
p2
```



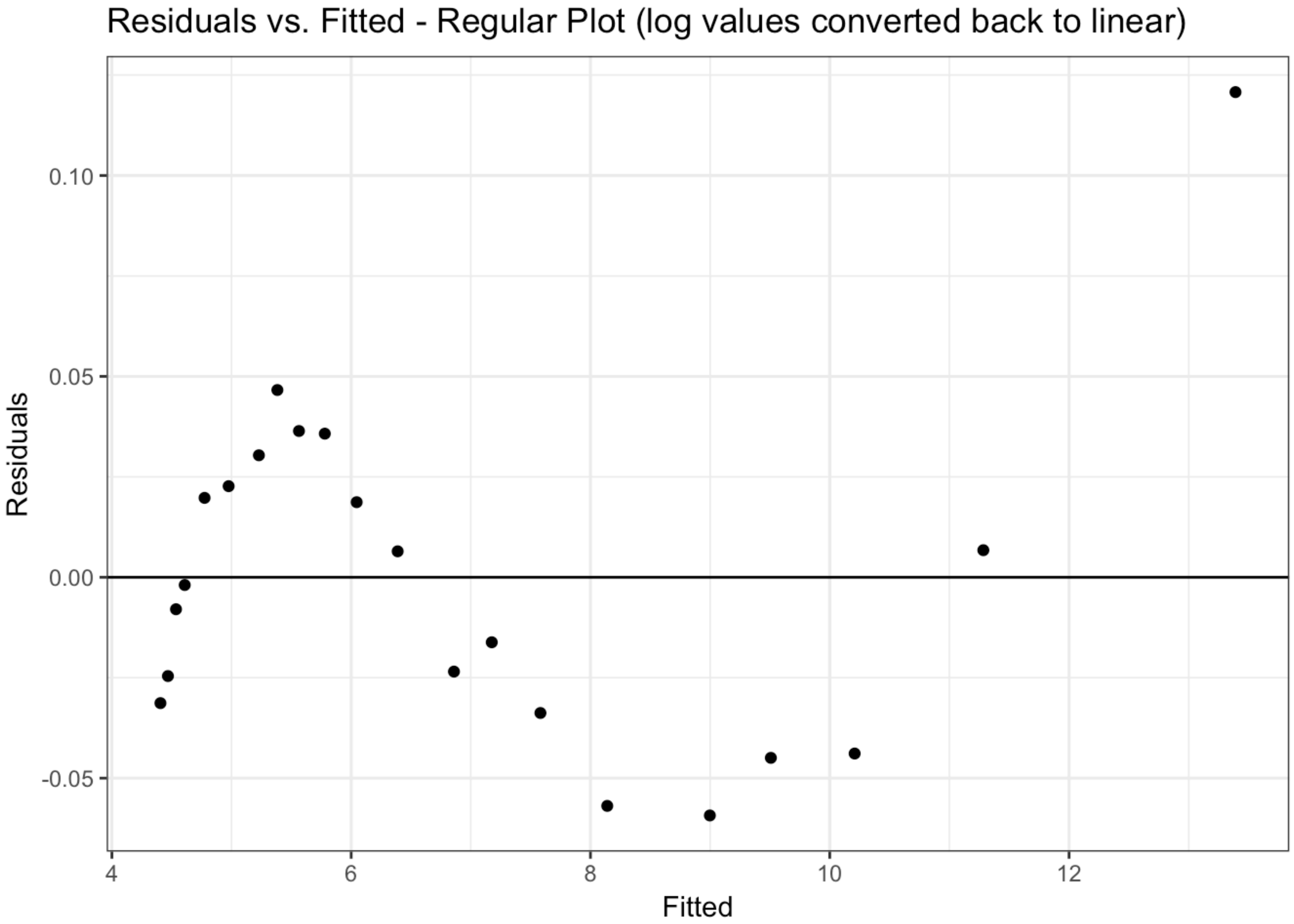
(c) Plot the residual against the fitted values in log-log and in original coordinates.

```
p3 = qplot(reg.lm$fitted.values, reg.resid,
     ylab="Residuals", xlab="Fitted",
     main="Residuals vs. Fitted - Regular Plot (linear)")
p3 = p3 + geom_abline(intercept = 0, slope = 0)
p3 = p3 + theme_bw() + theme(panel.background = element_blank())
p3
```

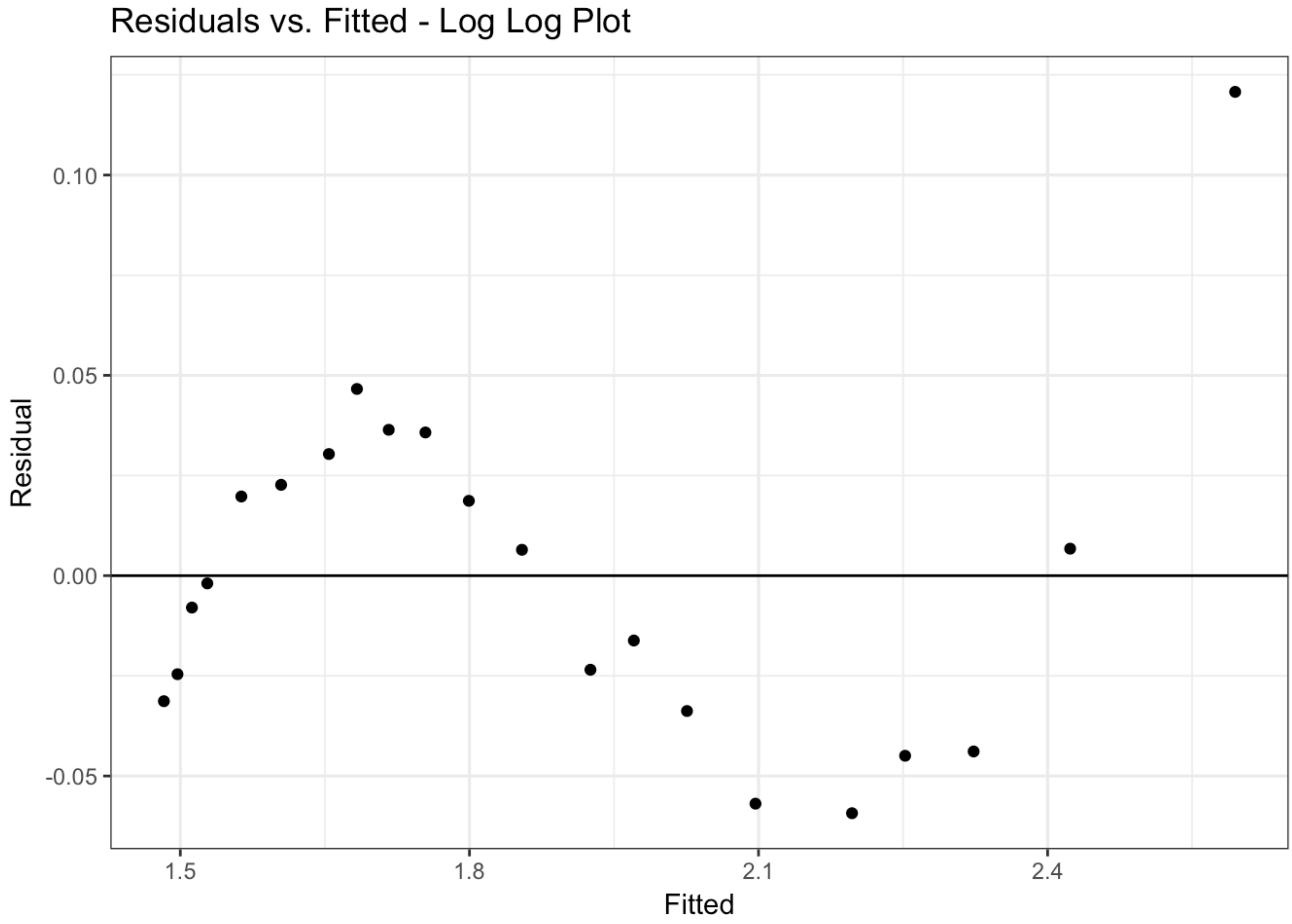


Residuals vs. Fitted - Regular Plot (linear)

```
p4 = qplot(exp(reg2.lm$fitted.values), reg2.resid,
     ylab="Residuals", xlab="Fitted",
     main="Residuals vs. Fitted - Regular Plot (log values converted back to linear)"
)
p4 = p4 + geom_abline(intercept = 0, slope = 0)
p4 = p4 + theme_bw() + theme(panel.background = element_blank())
p4
```



```
p5 = qplot(reg2.lm$fitted.values, reg2.resid,
     main="Residuals vs. Fitted - Log Log Plot", xlab="Fitted", ylab="Residual")
p5 = p5 + geom_abline(intercept = 0, slope = 0)
p5 = p5 + theme_bw() + theme(panel.background = element_blank())
p5
```



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

Good part The regression on log scale is better because regression line is a better fit as show in chart in part (a). It also has smaller residual values as evident in charts in part (c). It must also be noted that R-Square value on log-log scale is significantly better than R-Squared value on normal coordinates.

Bad Part The residuals have positive values for small fitted values and negative values for large fitted values. The points are not uniformly scattered across the zero residual line.

Problem 7.10

Build a linear regression of predicting the body mass from these diameters.

```
set.seed(2018)
data = read.table("http://www.statsci.org/data/oz/physical.txt", header = TRUE, sep =
'\t')

reg1.lm = lm(Mass ~ Fore + Bicep + Chest + Neck + Shoulder + Waist + Height + Calf +
Thigh + Head, data=data)
print(paste("1. R-Squared (Mass):", summary(reg1.lm)$r.squared))
```

```
## [1] "1. R-Squared (Mass): 0.977210661741324"
```

Now regress the cube root of mass against these diameters.

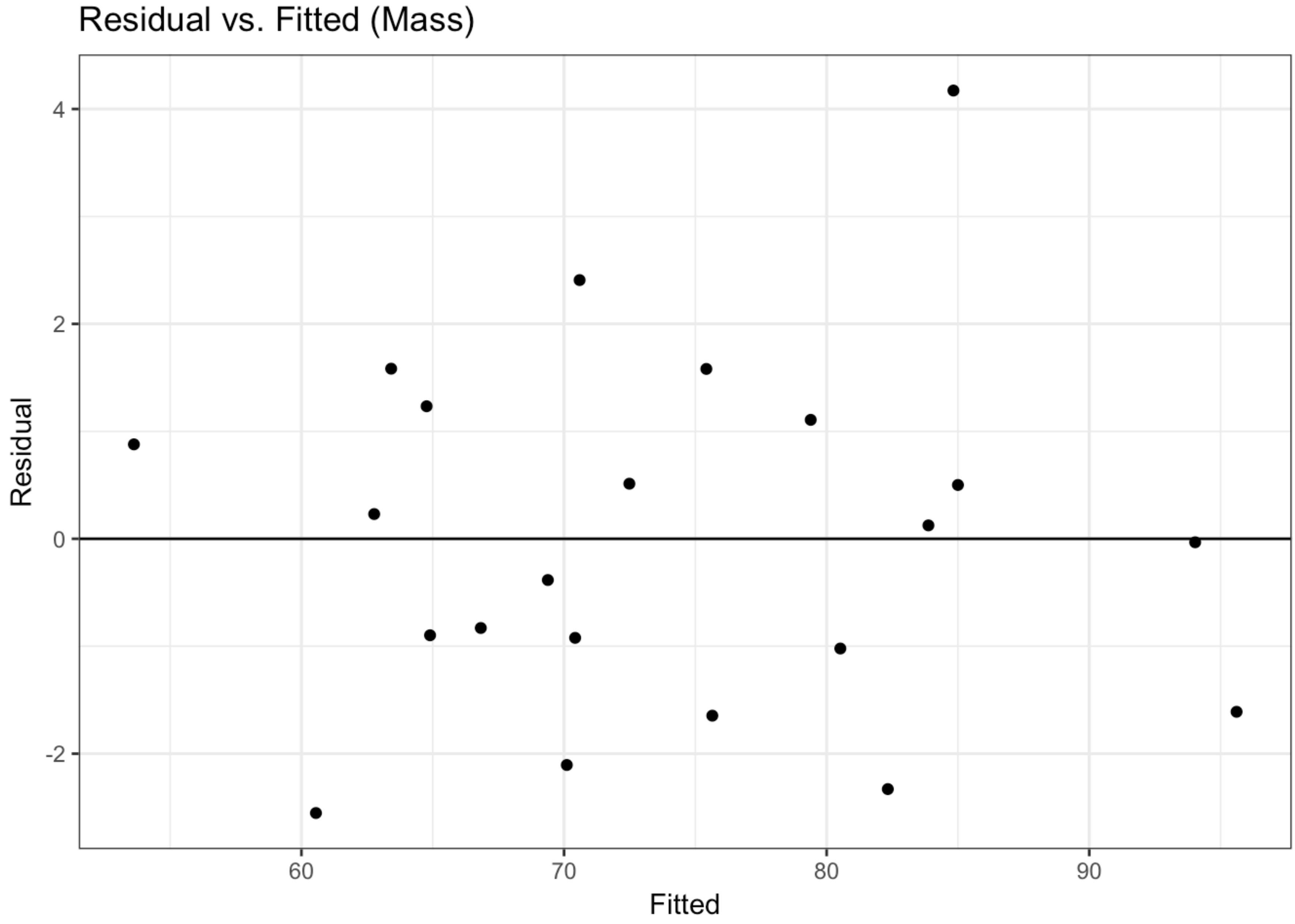
```
data$MassCubeRoot = data$Mass^(1/3)
reg2.lm = lm(MassCubeRoot ~ Fore + Bicep + Chest + Neck + Shoulder + Waist + Height +
Calf + Thigh + Head - Mass, data=data)

print(paste("2. R-Squared (Cube RootMass):", summary(reg2.lm)$r.squared))
```

```
## [1] "2. R-Squared (Cube RootMass): 0.975847620620573"
```

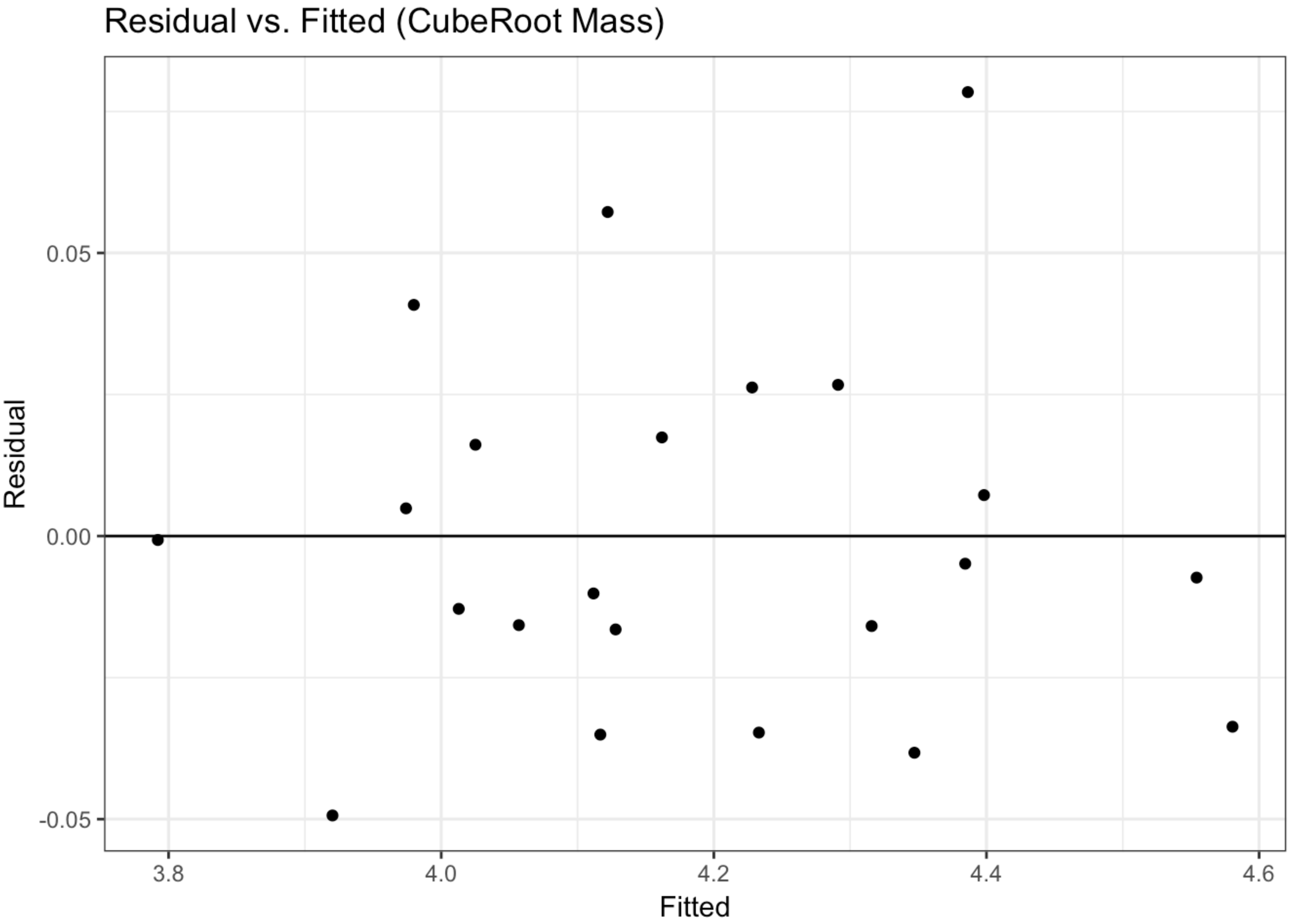
(a) Plot the residual against the fitted values for your regression.

```
p6 = qplot(reg1.lm$fitted.values, reg1.lm$residuals, xlab="Fitted",
      ylab="Residual",main="Residual vs. Fitted (Mass)") +
  geom_abline(intercept = 0, slope = 0)
p6 = p6 + theme_bw() + theme(panel.background = element_blank())
p6
```

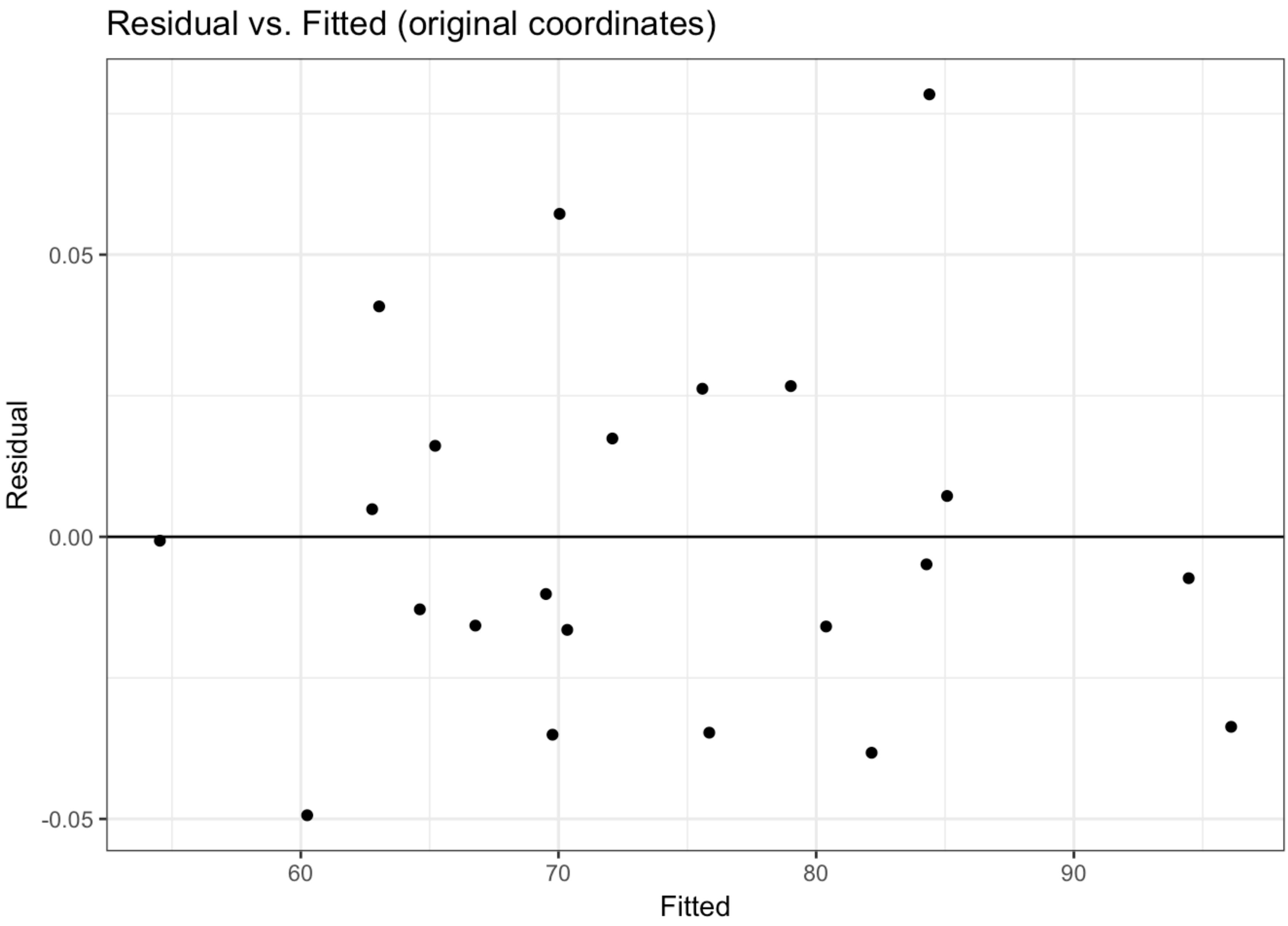


(b) Now regress the cube root of mass against these diameters. Plot the residual against the fitted values in both these cube root coordinates and in the original coordinates.

```
p7 = qplot(reg2.lm$fitted.values, reg2.lm$residuals, xlab="Fitted",
      ylab="Residual",main="Residual vs. Fitted (CubeRoot Mass)") +
  geom_abline(intercept = 0, slope = 0)
p7 = p7 + theme_bw() + theme(panel.background = element_blank())
p7
```



```
p8 = qplot(reg2.lm$fitted.values^3, reg2.lm$residuals, xlab="Fitted",
      ylab="Residual",main="Residual vs. Fitted (original coordinates)") +
  geom_abline(intercept = 0, slope = 0)
p8 = p8 + theme_bw() + theme(panel.background = element_blank())
p8
```



(c) Use plot to explain which regression is better?

> The plot in part (b) has smaller residual values even though R-squared values of two models are comparable. The regression using cube root of mass is better than regression using original mass values.

Problem 7.11

Read the data in the dataframe

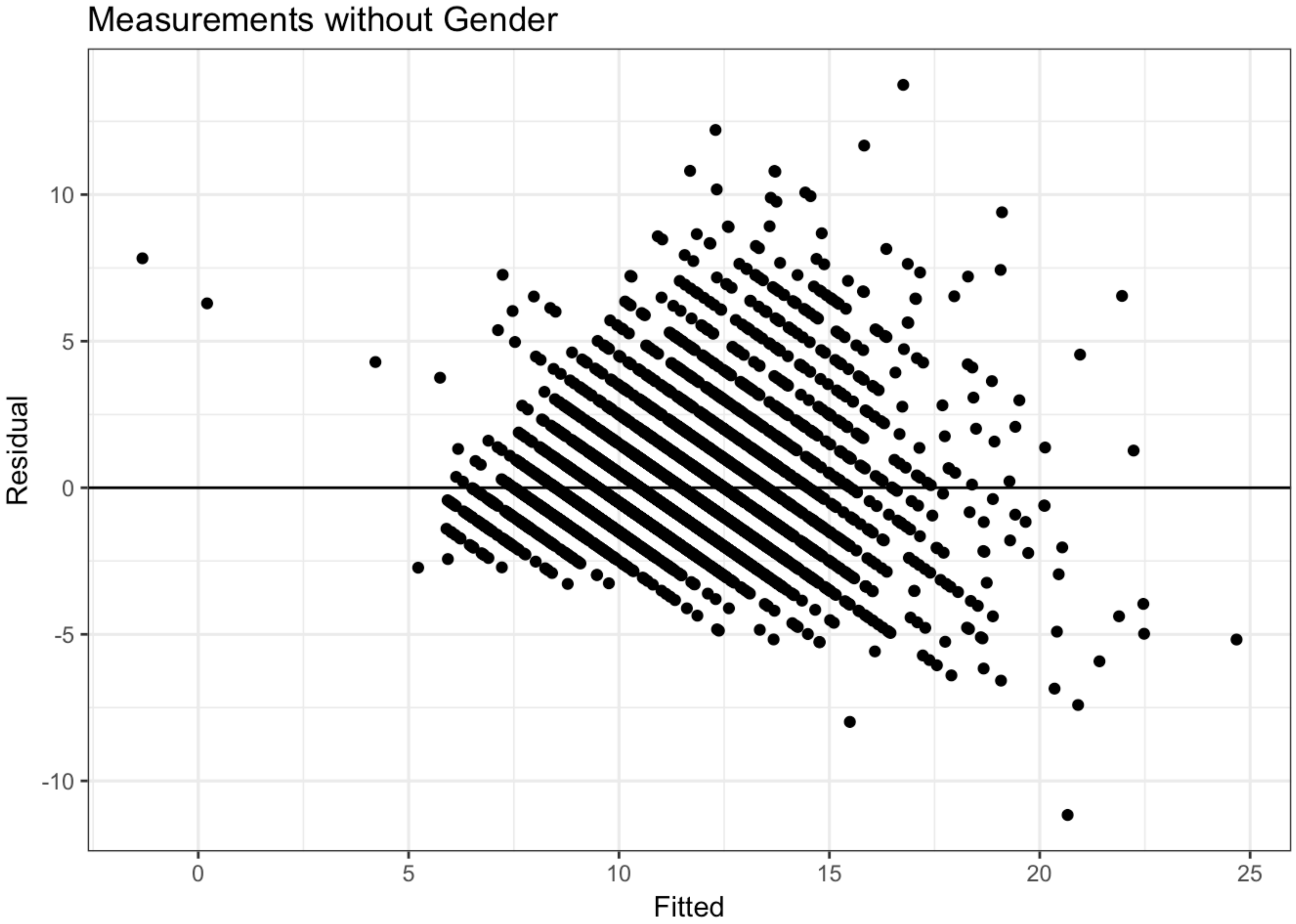
```
rm(list = ls())
library(stats)
library(ggplot2)
library(grid)
library(gridExtra)
library(glmnet)
library(caret)
library(ModelMetrics)
webLink = "https://archive.ics.uci.edu/ml/machine-learning-databases/abalone/abalone.data"
localLink = "~/cs498aml/sulphate/abalone.data"
data = read.table(webLink, header = FALSE, sep = ',')
```

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

```
data[,9] = data[,9] + 1.5 # age = rings + 1.5
reg1.lm = lm(V9 ~ V2 + V3 + V4 + V5 + V6 + V7 + V8, data=data)
print(paste("R-Squared (Measurements without Gender):", summary(reg1.lm)$r.squared))
```

```
## [1] "R-Squared (Measurements without Gender): 0.527629939991984"
```

```
p1 = qplot(reg1.lm$fitted.values, reg1.lm$residuals, xlab="Fitted",
      ylab="Residual",main="Measurements without Gender") +
  geom_abline(intercept = 0, slope = 0)
p1 = p1 + theme_bw() + theme(panel.background = element_blank())
p1
```

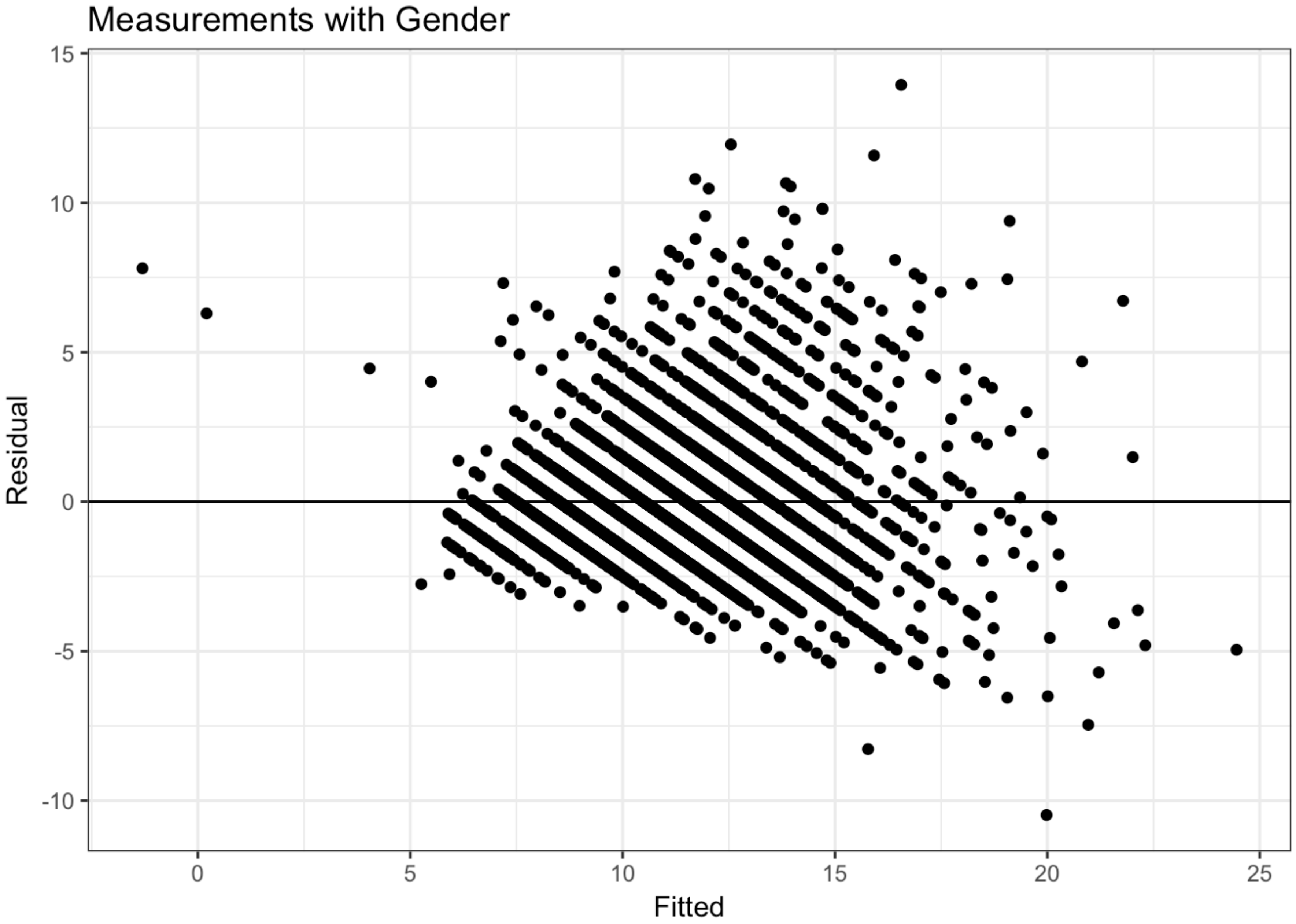


(b) Build a linear regression predicting the age from the measurements, including gender.

```
data$V10 = as.factor(data$V1)
reg2.lm = lm(V9 ~ V2 + V3 + V4 + V5 + V6 + V7 + V8 + V10, data=data)
print(paste("R-Squared (Measurements with Gender):", summary(reg2.lm)$r.squared))
```

```
## [1] "R-Squared (Measurements with Gender): 0.537884403021195"
```

```
p2 = qplot(reg2.lm$fitted.values, reg2.lm$residuals, xlab="Fitted",
      ylab="Residual",main="Measurements with Gender") +
  geom_abline(intercept = 0, slope = 0)
p2 = p2 + theme_bw() + theme(panel.background = element_blank())
p2
```

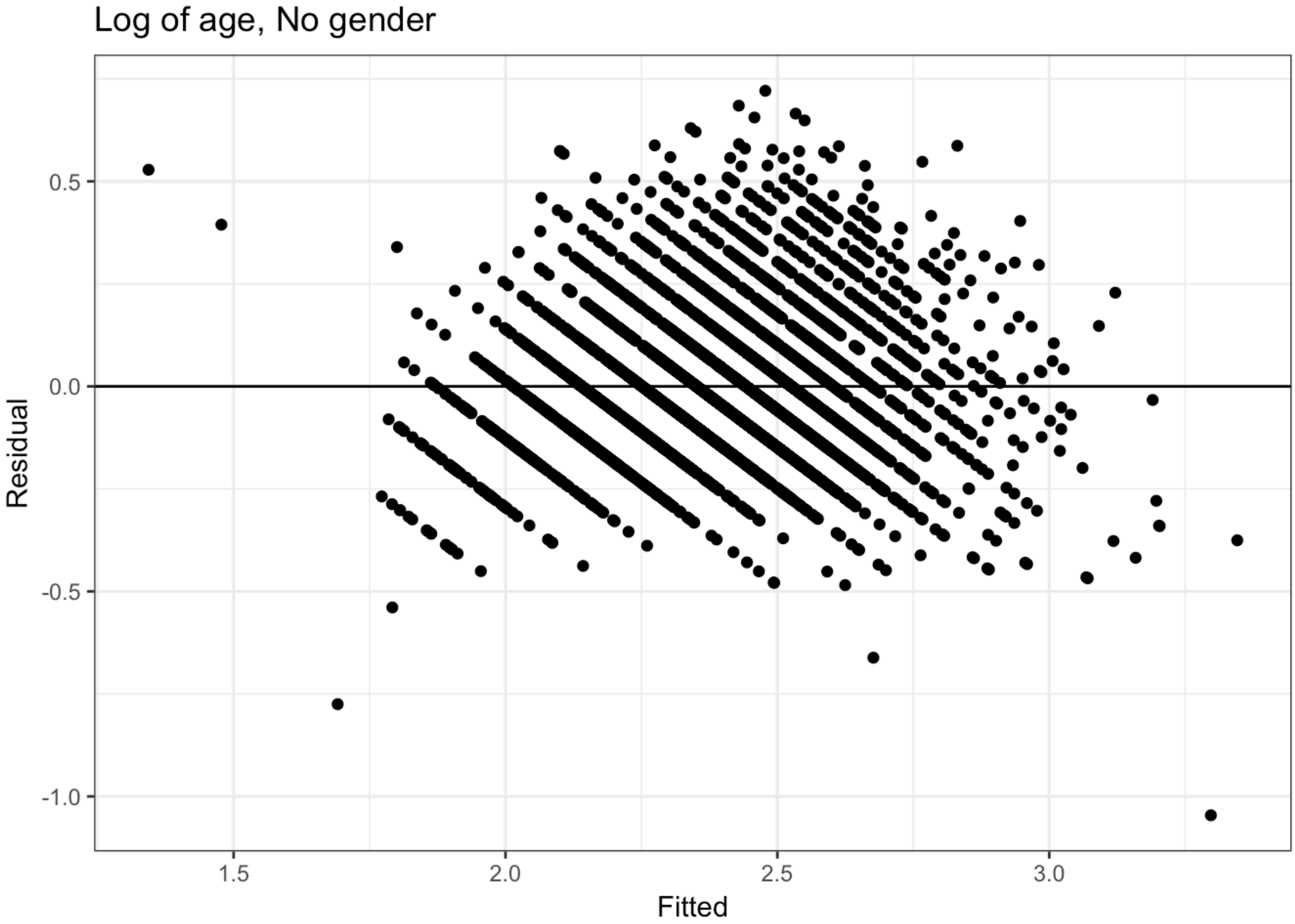


(c) Now build a linear regression predicting the log of age from the measurements, ignoring gender. Plot the residual against the fitted values.

```
data$V11 = log(data$V9)
reg3.lm = lm(V11 ~ V2 + V3 + V4 + V5 + V6 + V7 + V8, data=data)
print(paste("R-Squared (Log of age, No gender):", summary(reg3.lm)$r.squared))
```

```
## [1] "R-Squared (Log of age, No gender): 0.579693522453324"
```

```
p3 = qplot(reg3.lm$fitted.values, reg3.lm$residuals, xlab="Fitted",
      ylab="Residual",main="Log of age, No gender") +
  geom_abline(intercept = 0, slope = 0)
p3 = p3 + theme_bw() + theme(panel.background = element_blank())
p3
```

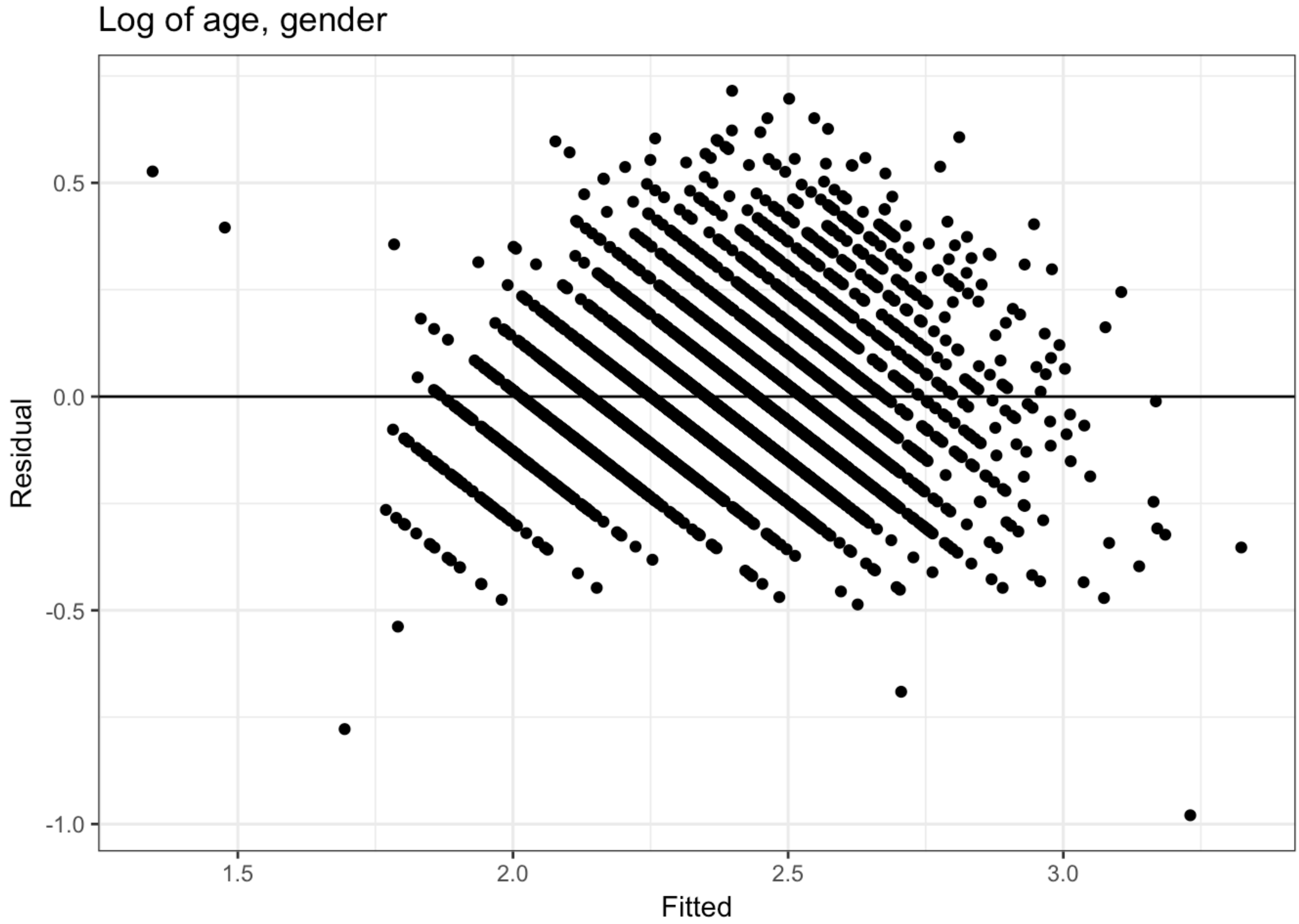


(d) Now build a linear regression predicting the log age from the measurements, including gender, represented as above. Plot the residual against the fitted values.

```
reg4.lm = lm(V11 ~ V2 + V3 + V4 + V5 + V6 + V7 + V8 + V10, data=data)
print(paste("R-Squared (Log of age, gender):", summary(reg4.lm)$r.squared))
```

```
## [1] "R-Squared (Log of age, gender): 0.593199170651908"
```

```
p4 = qplot(reg4.lm$fitted.values, reg4.lm$residuals, xlab="Fitted",
      ylab="Residual",main="Log of age, gender") +
  geom_abline(intercept = 0, slope = 0)
p4 = p4 + theme_bw() + theme(panel.background = element_blank())
p4
```



(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.

> The regression on log of age with gender included and gender excluded is very similar. I will use regression model outlined in part(c) because it uses fewer input variables and returns same level of results as regression with 8 variables in (d).

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

```r
library(stats)
library(ggplot2)
library(grid)
library(gridExtra)
library(glmnet)
library(caret)
library(ModelMetrics)
library(plotmo)
```

```r
set.seed(2018)
webLink = "https://archive.ics.uci.edu/ml/machine-learning-databases/abalone/abalone.data"
localLink = "~/cs498aml/sulphate/abalone.data"
data = read.table(webLink, header = FALSE, sep = ',')
data[,9] = data[,9] + 1.5 # age = rings + 1.5
data$V1 = as.factor(data$V1)
```

cross validated Glmnet : No gender, Age

```r
inp = as.data.frame.data.frame(data[,-c(1,9)])
y = data[,9]
reg1.glm = cv.glmnet(x = as.matrix(inp), y = y, type.measure = "mse", alpha = 0 ) #ridge
```

cross validated Glmnet : with gender, Age

```r
inp = data[,-9]
inp[,1] = as.numeric(inp[,1])
y = data[,9]
reg2.glm =  cv.glmnet(x = as.matrix(inp), y = y, type.measure = "mse",alpha = 0 ) #ridge
```

cross validated Glmnet : No gender, log(Age)

```r
inp = as.data.frame.data.frame(data[,-c(1,9)])
y = log(data[,9])
reg3.glm =  cv.glmnet(x = as.matrix(inp), y = y, type.measure = "mse",alpha = 0 ) #ridge
```

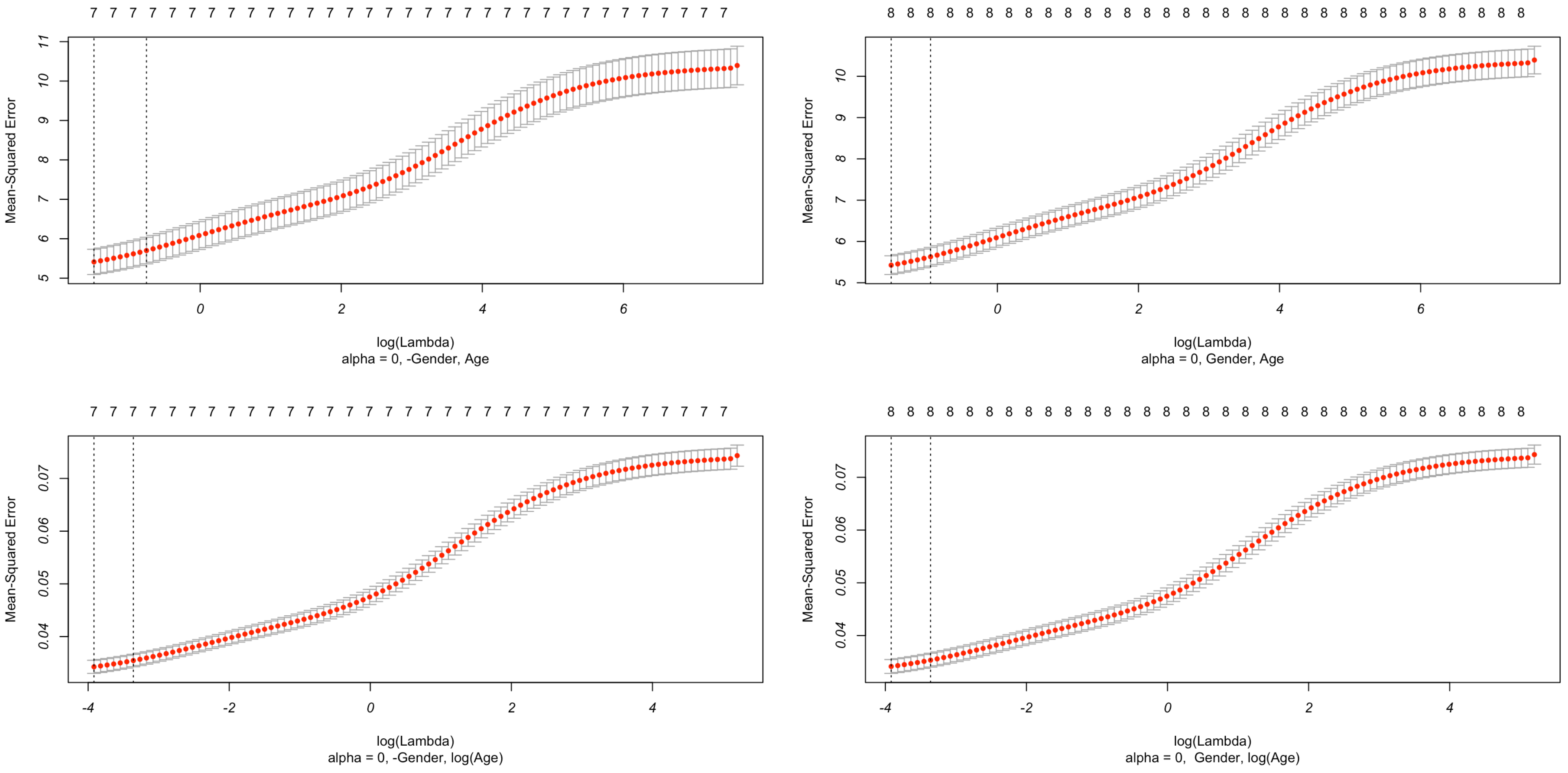
cross validated GLmnet : with gender, log(Age)

```
inp = data[,-9]
inp[,1] = as.numeric(inp[,1])
y = log(data[,9])
reg4.glm =  cv.glmnet(as.matrix(inp), y = y, type.measure = "mse", alpha=0)
```

```
cve = list()
cve[1] = min(reg1.glm$cvm)
cve[2] = min(reg2.glm$cvm)
cve[3] = exp(min(reg3.glm$cvm))
cve[4] = exp(min(reg4.glm$cvm))
```

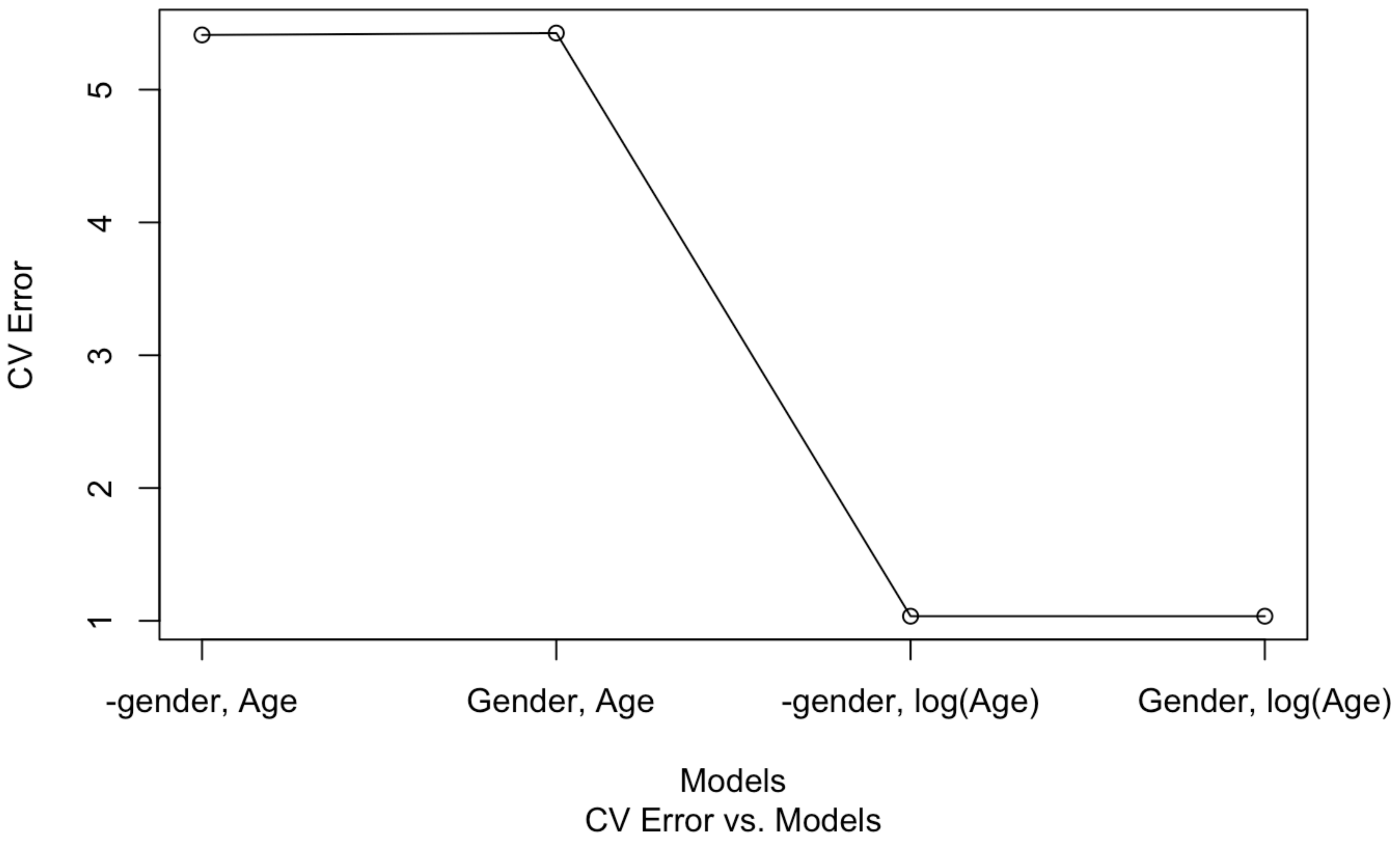
The lambda vs. MSE plot for all models shows that error increases as regularization value increases.

```
par(mfrow=c(2,2))
plot(reg1.glm, sub = paste("alpha = 0,", "-Gender, Age"), font = 3)
plot(reg2.glm, sub = paste("alpha = 0,", "Gender, Age"), font = 3)
plot(reg3.glm, sub = paste("alpha = 0,", "-Gender, log(Age)"), font = 3)
plot(reg4.glm, sub = paste("alpha = 0,", " Gender, log(Age)"), font = 3)
```



Cross validation error for regression on age on normal scale is higher when compared to error for regression on age on log scale even after we convert the values back from log scale ton normal scale.

```
plot(c(1:4), cve, sub="CV Error vs. Models", xlab="Models", ylab="CV Error", type="o"
, xaxt = "n")
axis(1, at=1:4, labels = list("-gender, Age"," Gender, Age",  "-gender, log(Age)","Ge
nder, log(Age)" ))
```



Glmnet Cross validation with regularization of zero

The purpose of this section is to show the CV error when glmnet is run without any regulgarization. I used lambda of 0 and 0.0001 (very small value) on all 4 models. The CV Errors are smaller compared to the case when regularization is used and we let cv.glmnet() selet the lambda values automatically.

```
set.seed(2018)
webLink = "https://archive.ics.uci.edu/ml/machine-learning-databases/abalone/abalone.
data"
localLink = "~/cs498aml/sulphate/abalone.data"
data = read.table(webLink, header = FALSE, sep = ',')
data[,9] = data[,9] + 1.5 # age = rings + 1.5
data$V1 = as.factor(data$V1)
inp = as.data.frame.data.frame(data[,-c(1,9)])
y = data[,9]
reg1.glm = cv.glmnet(x = as.matrix(inp), y = y, type.measure = "mse", alpha = 0, lamb
da = seq(0,0.0001,0.0001) ) #ridge

inp = data[,-9]
inp[,1] = as.numeric(inp[,1])
y = data[,9]
reg2.glm =  cv.glmnet(x = as.matrix(inp), y = y, type.measure = "mse",alpha = 0, lamb
da = seq(0,0.0001,0.0001) ) #ridge

inp = as.data.frame.data.frame(data[,-c(1,9)])
y = log(data[,9])
reg3.glm =  cv.glmnet(x = as.matrix(inp), y = y, type.measure = "mse",alpha = 0, lamb
da = seq(0,0.0001,0.0001)) #ridge

inp = data[,-9]
inp[,1] = as.numeric(inp[,1])
y = log(data[,9])
reg4.glm =  cv.glmnet(as.matrix(inp), y = y, type.measure = "mse", alpha=0, lambda =
seq(0,0.0001,0.0001))
```

```
cve2 = list()
cve2[1] = min(reg1.glm$cvm)
cve2[2] = min(reg2.glm$cvm)
cve2[3] = min(reg3.glm$cvm)
cve2[4] = min(reg4.glm$cvm)
print("Without Regularization")
```

```
## [1] "Without Regularization"
```

```
print(paste("CV Error :",cve2))
```

```
## [1] "CV Error : 5.02846070570803"   "CV Error : 5.03964493684931"
## [3] "CV Error : 0.0322139703177234" "CV Error : 0.0321727771542115"
```

```
print("With Regularization")
```

```
## [1] "With Regularization"
```

```
print(paste("CV Error :",cve))
```

```
## [1] "CV Error : 5.41193002013638" "CV Error : 5.42622533419345"
## [3] "CV Error : 1.03485206601606" "CV Error : 1.03473392592617"
```

Summary

> I used ridge model against 4 different data models: Gender -Age, No Gender - Age, Gender - log(Age), No Gender - Log(Age) and used cross validated glmnet against these models. The MSE increased with regularization values in every model. However the cross validation error is significantly smaller when age is tranformed into logarithmic scale. The regularization factor - lambda, in the ridge model did not help improve the regression model.

Citation

> Besides using google search and r language manuals, I consulted with Aruna Neervannan during this assignment but we did not work in a group.