# Week 8 - Homework

*STAT 420, Summer 2019, Sushanta Panda*

- Exercise 1 (Writing Functions)
- Exercise 2 (Prostate Cancer Data)
- Exercise 3 (Why Bother?)
- Exercise 4 (Corrosion Data)
- Exercise 5 (Diamonds)

---

# Exercise 1 (Writing Functions)

**(a)** Write a function named `diagnostics` that takes as input the arguments:

- `model`, an object of class `lm()`, that is a model fit via `lm()`
- `pcol`, for controlling point colors in plots, with a default value of `grey`
- `lcol`, for controlling line colors in plots, with a default value of `dodgerblue`
- `alpha`, the significance level of any test that will be performed inside the function, with a default value of `0.05`
- `plotit`, a logical value for controlling display of plots with default value `TRUE`
- `testit`, a logical value for controlling outputting the results of tests with default value `TRUE`

The function should output:

- A list with two elements when `testit` is `TRUE`:
  - `p_val`, the p-value for the Shapiro-Wilk test for assessing normality
  - `decision`, the decision made when performing the Shapiro-Wilk test using the `alpha` value input to the function. "Reject" if the null hypothesis is rejected, otherwise "Fail to Reject."
- Two plots, side-by-side, when `plotit` is `TRUE`:
  - A fitted versus residuals plot that adds a horizontal line at $y = 0$, and labels the $x$-axis "Fitted" and the $y$-axis "Residuals." The points and line should be colored according to the input arguments. Give the plot a title.
  - A Normal Q-Q plot of the residuals that adds the appropriate line using `qqline()`. The points and line should be colored according to the input arguments. Be sure the plot has a title.

Consider using this function to help with the remainder of the assignment as well.

```
#Function created as part of the (a)
diagnostics = function(model = fit_1, pcol = "grey", lcol = "dodgerblue", alpha =
0.05, plotit = TRUE, testit = TRUE){

  p_val = shapiro.test(resid(model))$'p.value'
  decision = ifelse (p_val > alpha, "Fail to Reject","Reject")
  list_p_decision = list(p_val=p_val,decision=decision) #Create the List


  #if plotit = TRUE,
  if (plotit){
    par(mfrow=c(1,2))
    #Fitted Versus Residual Plot
    plot(fitted(model), resid(model), col = pcol, pch = 1, cex = 1, xlab = "Fitted
", ylab = "Residuals", main = "Fitted Versus Residual")
    abline (h = 0, col = lcol, lwd = 1, lty = 1)

    #Q-Q Plot
    qqnorm(resid(model), col = pcol, pch = 1, cex = 1, main = "Q-Q Plot of the Mod
el")
    qqline(resid(model), col = lcol, lwd = 1, lty = 1)
  }

  #if testit = TRUE
  if (testit){
    list_p_decision
  }
}
```

**(b)** Run the following code.

```
set.seed(420)

data_1 = data.frame(x = runif(n = 30, min = 0, max = 10),
                    y = rep(x = 0, times = 30))
data_1$y = with(data_1, 2 + 1 * x + rexp(n = 30))
fit_1 = lm(y ~ x, data = data_1)

data_2 = data.frame(x = runif(n = 20, min = 0, max = 10),
                    y = rep(x = 0, times = 20))
data_2$y = with(data_2, 5 + 2 * x + rnorm(n = 20))
fit_2 = lm(y ~ x, data = data_2)

data_3 = data.frame(x = runif(n = 40, min = 0, max = 10),
                    y = rep(x = 0, times = 40))
data_3$y = with(data_3, 2 + 1 * x + rnorm(n = 40, sd = x))
fit_3 = lm(y ~ x, data = data_3)
```

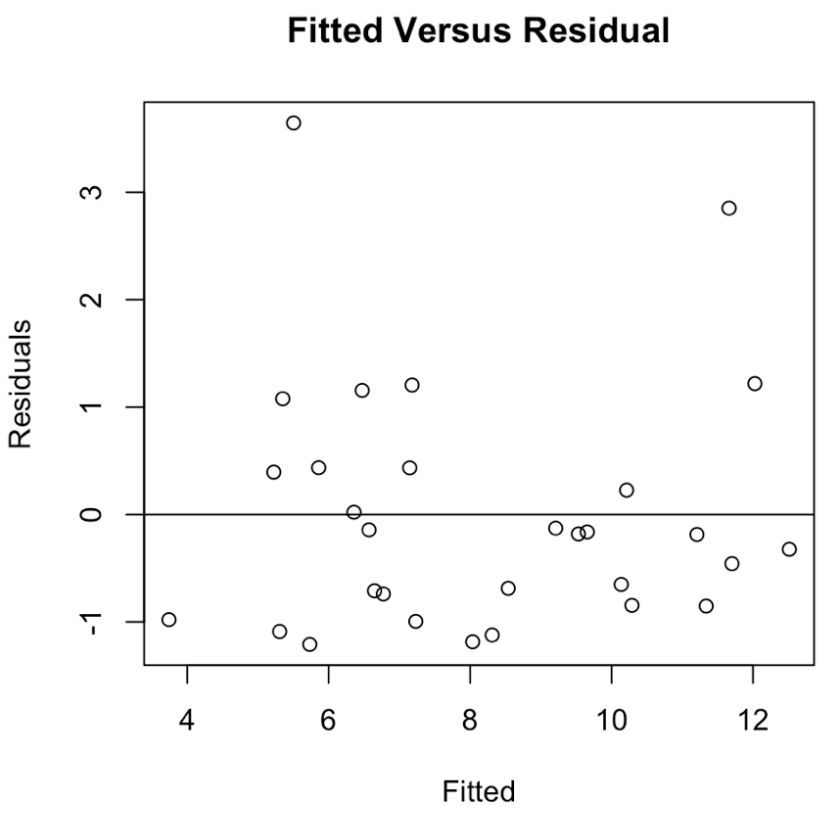
```
diagnostics(fit_1, plotit = FALSE)$p_val
```

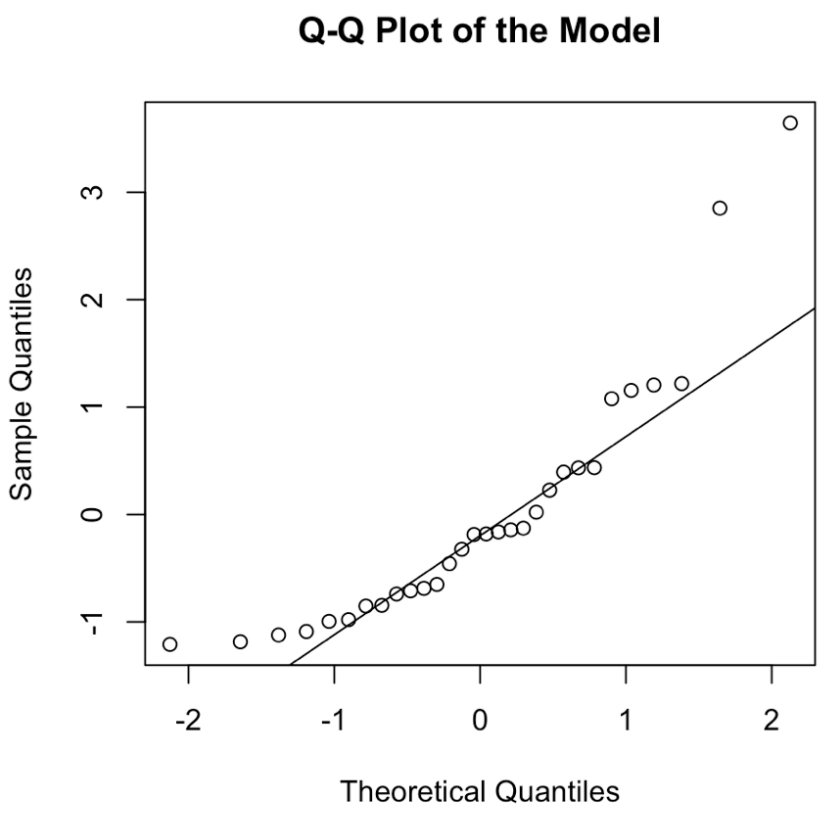
```
## [1] 0.0004299
```

```
diagnostics(fit_2, plotit = FALSE)$decision
```

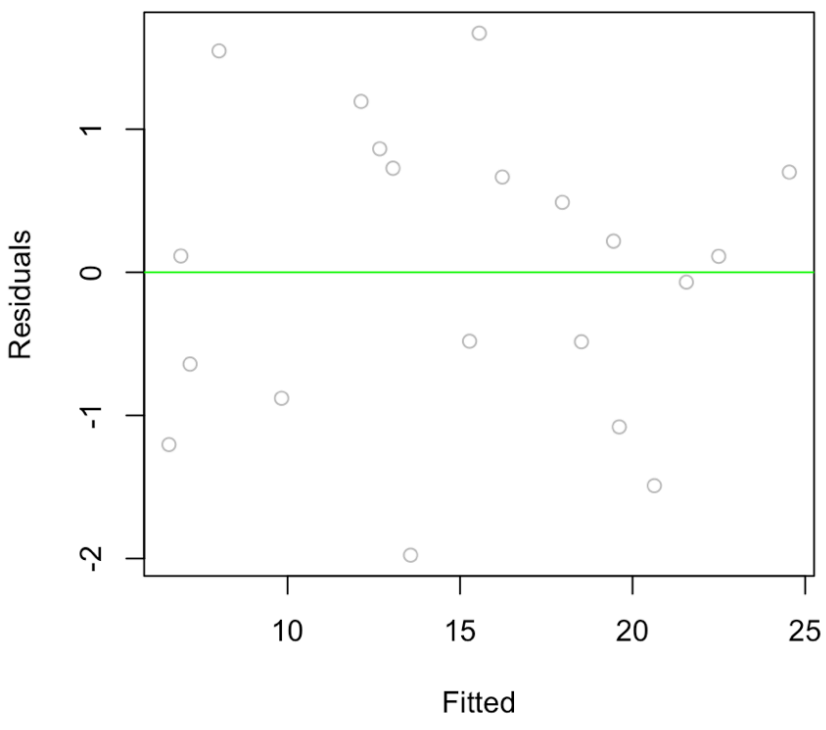
```
## [1] "Fail to Reject"
```

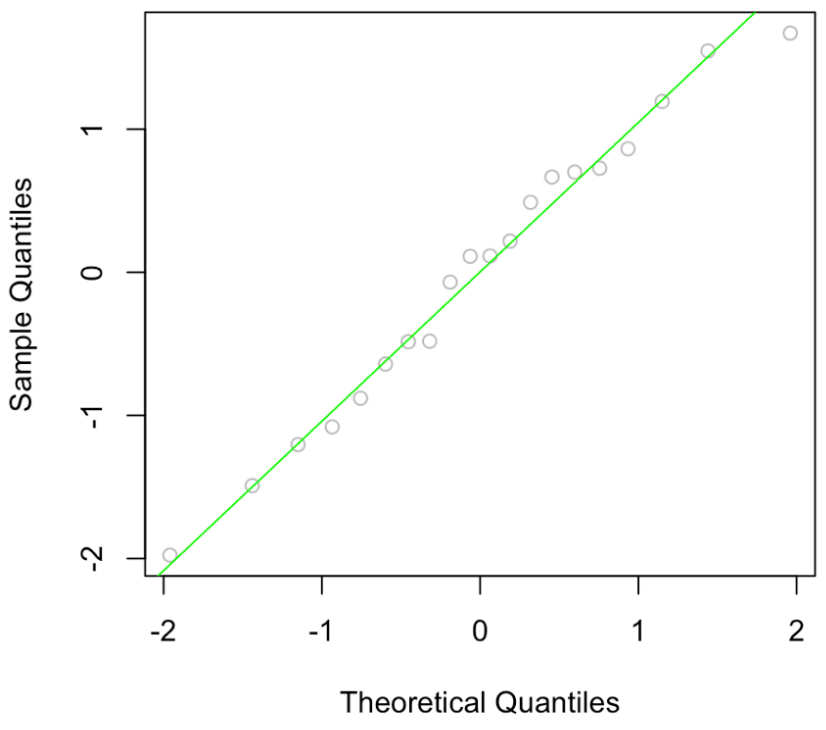
```
diagnostics(fit_1, testit = FALSE, pcol = "black", lcol = "black")
```



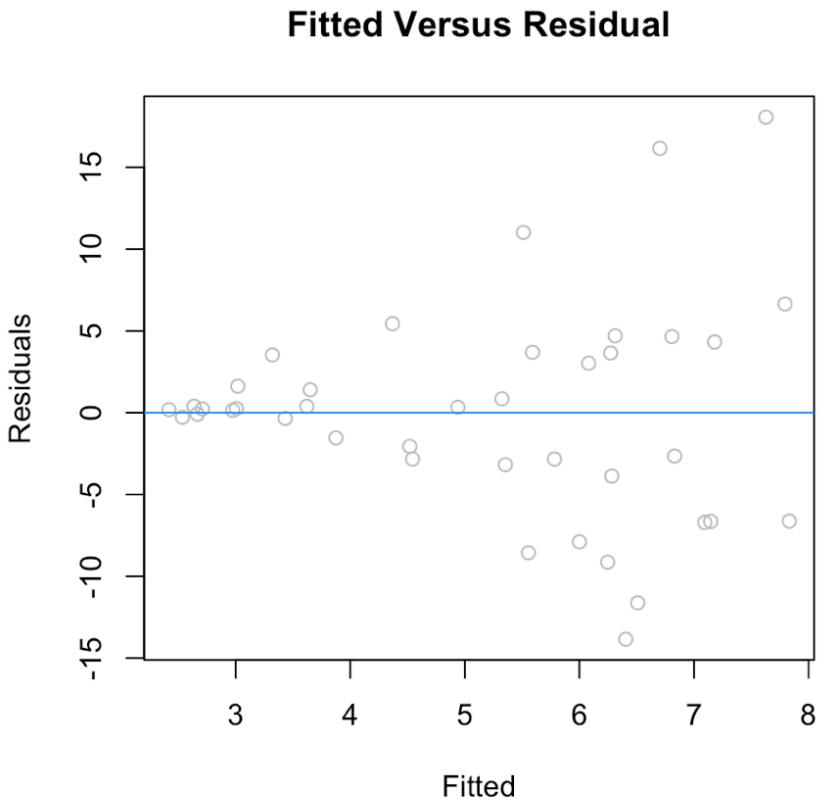



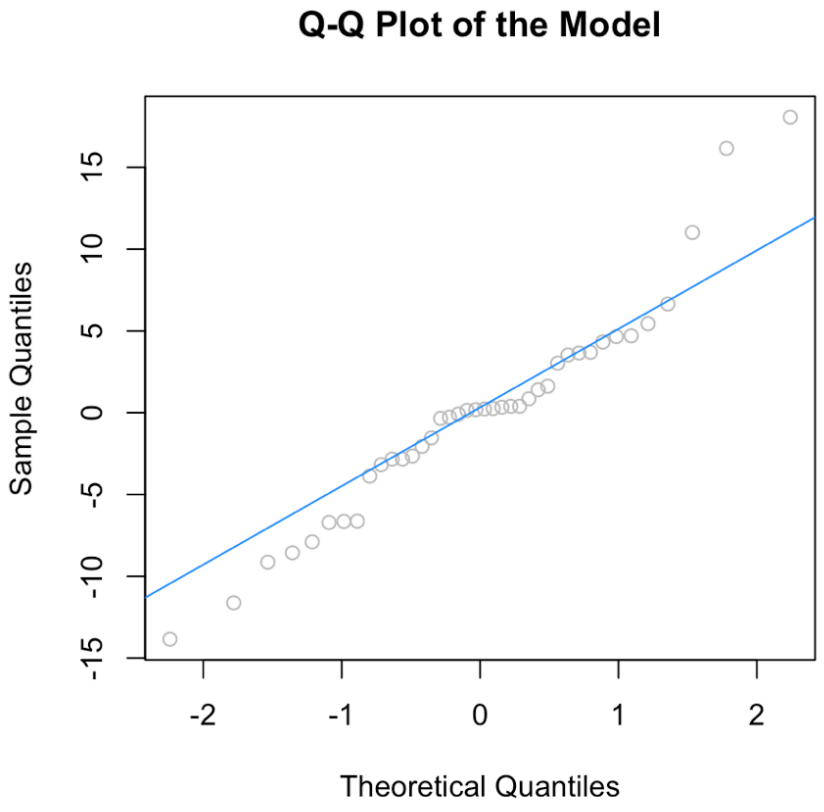

```
diagnostics(fit_2, testit = FALSE, pcol = "grey", lcol = "green")
```




```
diagnostics(fit_3)
```




```
## $p_val
## [1] 0.09105
##
## $decision
## [1] "Fail to Reject"
```

# Exercise 2 (Prostate Cancer Data)

For this exercise, we will use the `prostate` data, which can be found in the `faraway` package. After loading the `faraway` package, use `?prostate` to learn about this dataset.

```
library(faraway)
prostate_data = faraway::prostate
```

**(a)** Fit an additive multiple regression model with `lpsa` as the response and the remaining variables in the `prostate` dataset as predictors. Report the $R^2$ value for this model.

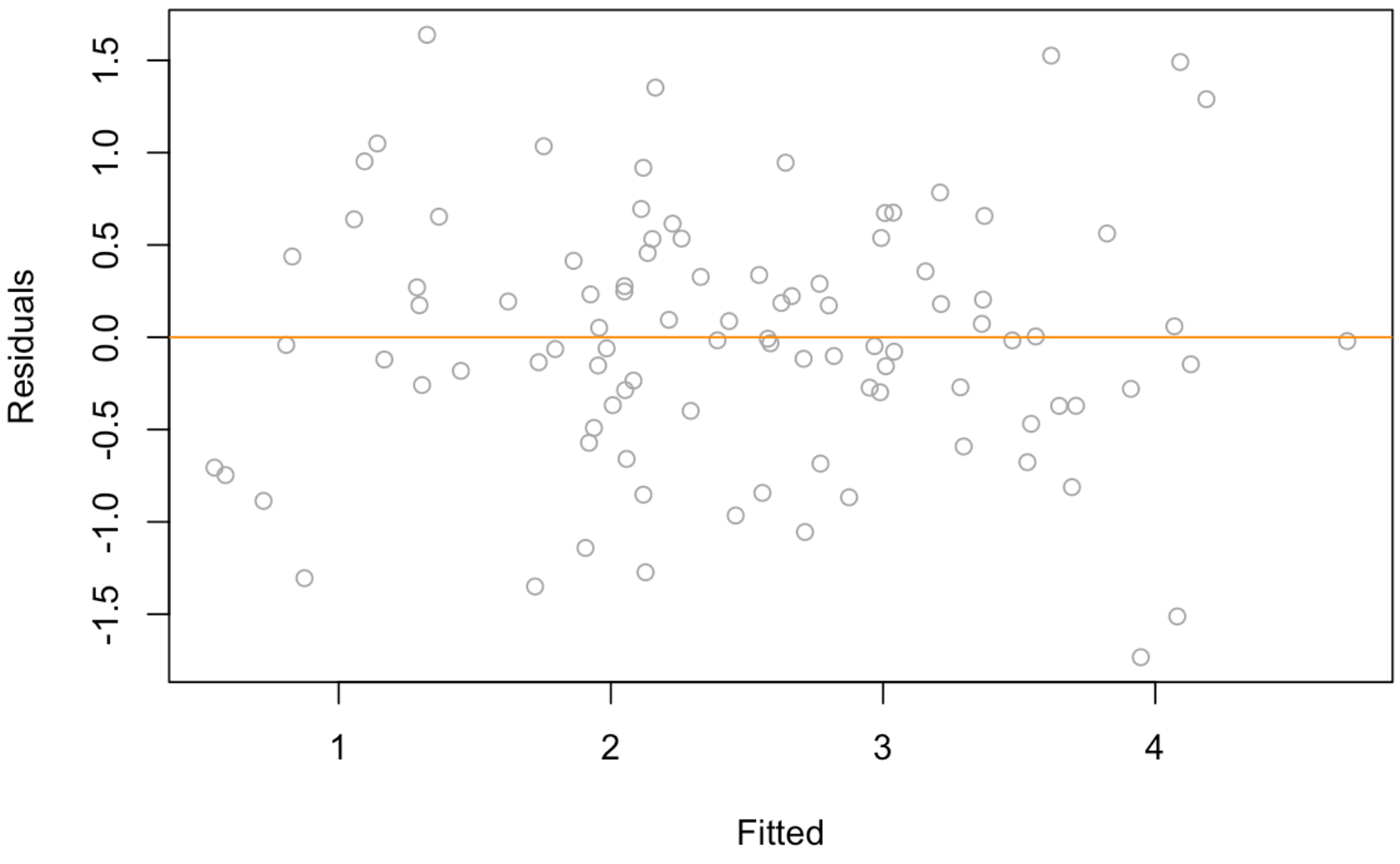
```
prostate_add_model = lm(lpsa~.,data = prostate_data)
summary(prostate_add_model)
```

```
##
## Call:
## lm(formula = lpsa ~ ., data = prostate_data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.733 -0.371 -0.017  0.414  1.638
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.66934    1.29639    0.52   0.6069
## lcavol       0.58702    0.08792    6.68  2.1e-09 ***
## lweight      0.45447    0.17001    2.67   0.0090 **
## age         -0.01964    0.01117   -1.76   0.0823 .
## lbph         0.10705    0.05845    1.83   0.0704 .
## svi          0.76616    0.24431    3.14   0.0023 **
## lcp         -0.10547    0.09101   -1.16   0.2496
## gleason      0.04514    0.15746    0.29   0.7750
## pgg45        0.00453    0.00442    1.02   0.3089
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.708 on 88 degrees of freedom
## Multiple R-squared:  0.655,  Adjusted R-squared:  0.623
## F-statistic: 20.9 on 8 and 88 DF,  p-value: <2e-16
```

$R^2$ value for this model is : **0.6548**

**(b)** Check the constant variance assumption for this model. Do you feel it has been violated? Justify your answer.

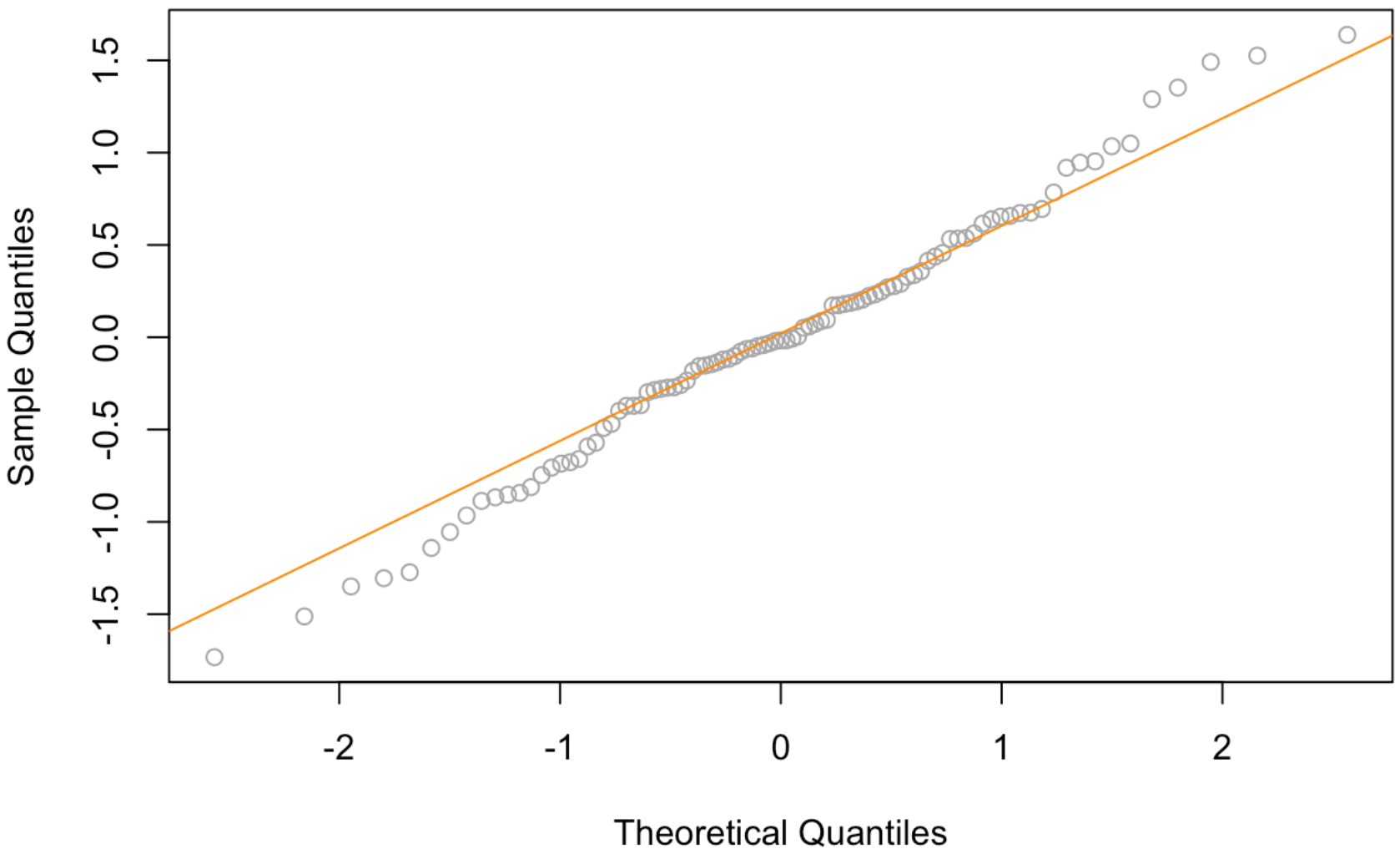
```
plot(fitted(prostate_add_model),resid(prostate_add_model),col = "darkgrey", pch =
1, cex = 1, xlab = "Fitted", ylab = "Residuals", main = "Fitted Vs Residual Plot f
or Additive Model")
abline(h = 0, col = "darkorange", lwd = 1, lty = 1)
```




From the above Fitted Versus Residual Plots of the Prostate Additive model, it seems it **assumption of constant varience is not suspect**, means the assumption of the constant varience **is not** violating by the model. As for the fitted values, the residuals uniformly distributed accross the orange line. Though there are few grey points which are bit higher / lower in certain areas against fitted valued, but rest all places it follows the uniformity

**(c)** Check the normality assumption for this model. Do you feel it has been violated? Justify your answer.

```
qqnorm(resid(prostate_add_model),col = "darkgrey", pch = 1, cex = 1, main = "Q-Q P
lot for the Additive Prostate Model")
qqline(resid(prostate_add_model),col = "darkorange", lwd = 1, lty = 1)
```




The Q-Q plots, show all the grey circle follows in the straight line, except for the upper and lower quantiles, which seems a slightly fat tails, however most of the other follows the straight line. Since it follows the straight line, the **model doesn't violates the normal distribution**

**(d)** Check for any high leverage observations. Report any observations you determine to have high leverage.

Below are the leverage which are high

```
hatvalues(prostate_add_model)[hatvalues(prostate_add_model) > 2 * mean(hatvalues(p
rostate_add_model))]
```

```
##     32     37     41     74     92
## 0.3305 0.2184 0.2410 0.1912 0.2092
```

Below are the observations which have high leverage

```
prostate_data[hatvalues(prostate_add_model) > 2 * mean(hatvalues(prostate_add_mode
l)),]
```

```
##    lcavol lweight age    lbph svi    lcp gleason pgg45  lpsa
## 32 0.1823   6.108  65  1.7047   0 -1.386       6     0 2.008
## 37 1.4231   3.657  73 -0.5798   0  1.658       8    15 2.158
## 41 0.6206   3.142  60 -1.3863   0 -1.386       9    80 2.298
## 74 1.8390   3.237  60  0.4383   1  1.179       9    90 3.075
## 92 2.5329   3.678  61  1.3481   1 -1.386       7    15 4.130
```

**(e)** Check for any influential observations. Report any observations you determine to be influential.

Below are the influential observations

```
high_cd = cooks.distance(prostate_add_model) > (4/length(cooks.distance(prostate_a
dd_model)))
prostate_data[high_cd,]
```

```
##     lcavol lweight age    lbph svi    lcp gleason pgg45  lpsa
## 32  0.1823   6.108  65  1.7047   0 -1.386       6     0 2.008
## 39  2.6610   4.085  68  1.3737   1  1.833       7    35 2.214
## 47  2.7279   3.995  79  1.8795   1  2.657       9   100 2.569
## 69 -0.4463   4.409  69 -1.3863   0 -1.386       6     0 2.963
## 95  2.9074   3.396  52 -1.3863   1  2.464       7    10 5.143
## 96  2.8826   3.774  68  1.5581   1  1.558       7    80 5.478
## 97  3.4720   3.975  68  0.4383   1  2.904       7    20 5.583
```

**(f)** Refit the additive multiple regression model without any points you identified as influential. Compare the coefficients of this fitted model to the previously fitted model.

```
library(knitr)
prostate_add_model_fix = lm(lpsa~.,data = prostate_data[!high_cd,])
prostate_add_model_fix
```

```
##
## Call:
## lm(formula = lpsa ~ ., data = prostate_data[!high_cd, ])
##
## Coefficients:
## (Intercept)       lcavol      lweight          age          lbph          svi
##    -0.24608      0.56499      0.54443     -0.01856      0.13328      0.74475
##         lcp      gleason        pgg45
##    -0.15542      0.11472      0.00665
```

```
compare_coefficient = data.frame(
  Coefficient = c("Intercept","lcavol","lweight","age","lbph","svi","lcp","gleason","pgg45"),
  prostate_add_model = c(summary(prostate_add_model)$coefficient[1],summary(prostate_add_model)$coefficient[2],summary(prostate_add_model)$coefficient[3],summary(prostate_add_model)$coefficient[4],summary(prostate_add_model)$coefficient[5],summary(prostate_add_model)$coefficient[6],summary(prostate_add_model)$coefficient[7],summary(prostate_add_model)$coefficient[8],summary(prostate_add_model)$coefficient[9]),
  prostate_add_model_fix = c(summary(prostate_add_model_fix)$coefficient[1],summary(prostate_add_model_fix)$coefficient[2],summary(prostate_add_model_fix)$coefficient[3],summary(prostate_add_model_fix)$coefficient[4],summary(prostate_add_model_fix)$coefficient[5],summary(prostate_add_model_fix)$coefficient[6],summary(prostate_add_model_fix)$coefficient[7],summary(prostate_add_model_fix)$coefficient[8],summary(prostate_add_model_fix)$coefficient[9])
)
kable(compare_coefficient, format = "pandoc",padding = 2,caption = "Compare Coefficient before and After Removal of Influencers")
```

Compare Coefficient before and After Removal of Influencers

| Coefficient | prostate_add_model | prostate_add_model_fix |
|---|---|---|
| Intercept | 0.6693 | -0.2461 |
| lcavol | 0.5870 | 0.5650 |
| lweight | 0.4545 | 0.5444 |
| age | -0.0196 | -0.0186 |
| lbph | 0.1071 | 0.1333 |
| svi | 0.7662 | 0.7447 |
| lcp | -0.1055 | -0.1554 |
| gleason | 0.0451 | 0.1147 |
| pgg45 | 0.0045 | 0.0066 |

From the above table, it seems that other than the `intercept` , rest other estimated parameters do have the same sign and not a very big difference. However the value of the `intercept` is significantly different in both the model

**(g)** Create a data frame that stores the observations that were “removed” because they were influential. Use the two models you have fit to make predictions with these observations. Comment on the difference between these two sets of predictions.

```
influential_data =  prostate_data[high_cd,]
influential_data
```

```
##     lcavol lweight age    lbph svi    lcp gleason pgg45  lpsa
## 32  0.1823   6.108  65  1.7047   0 -1.386       6     0 2.008
## 39  2.6610   4.085  68  1.3737   1  1.833       7    35 2.214
## 47  2.7279   3.995  79  1.8795   1  2.657       9   100 2.569
## 69 -0.4463   4.409  69 -1.3863   0 -1.386       6     0 2.963
## 95  2.9074   3.396  52 -1.3863   1  2.464       7    10 5.143
## 96  2.8826   3.774  68  1.5581   1  1.558       7    80 5.478
## 97  3.4720   3.975  68  0.4383   1  2.904       7    20 5.583
```

```
predict(prostate_add_model,newdata = influential_data)
```

```
##    32    39    47    69    95    96    97
## 2.875 3.947 4.081 1.325 3.618 4.188 4.092
```

```
predict(prostate_add_model_fix,newdata = influential_data)
```

```
##    32    39    47    69    95    96    97
## 3.106 3.898 4.284 1.340 3.327 4.220 3.905
```

```
compare_prediction = data.frame(
  Observation = c("1","2","3","4","5","6","7"),
  Prediction_Add = c(predict(prostate_add_model,newdata = influential_data)[1],pre
dict(prostate_add_model,newdata = influential_data)[2],predict(prostate_add_model,
newdata = influential_data)[3],predict(prostate_add_model,newdata = influential_da
ta)[4],predict(prostate_add_model,newdata = influential_data)[5],predict(prostate_
add_model,newdata = influential_data)[6],predict(prostate_add_model,newdata = infl
uential_data)[7]),
  Prediction_Add_Fix = c(predict(prostate_add_model_fix,newdata = influential_data
)[1],predict(prostate_add_model_fix,newdata = influential_data)[2],predict(prostat
e_add_model_fix,newdata = influential_data)[3],predict(prostate_add_model_fix,newd
ata = influential_data)[4],predict(prostate_add_model_fix,newdata = influential_da
ta)[5],predict(prostate_add_model_fix,newdata = influential_data)[6],predict(prost
ate_add_model_fix,newdata = influential_data)[7])
)
kable(compare_prediction, format = "pandoc",padding = 2,caption = "Compare Predict
ion before and After Removal of Influencers")
```

Compare Prediction before and After Removal of Influencers

| | Observation | Prediction_Add | Prediction_Add_Fix |
|---|---|---|---|
| 32 | 1 | 2.875 | 3.107 |
| 39 | 2 | 3.947 | 3.898 |
| 47 | 3 | 4.081 | 4.284 |
| 69 | 4 | 1.325 | 1.340 |
| 95 | 5 | 3.618 | 3.327 |

| | | | |
|---|---|---|---|
| 96 | 6 | 4.188 | 4.220 |
| 97 | 7 | 4.092 | 3.905 |

It seems that, prediction seems equal with not very large difference from both the model, also it seems the prediction has no impact on output if we don't keep these observation in the dataset.

---

# Exercise 3 (Why Bother?)

**Why** do we care about violations of assumptions? One key reason is that the distributions of the parameter esimators that we have used are all reliant on these assumptions. When the assumptions are violated, the distributional results are not correct, so our tests are garbage. **Garbage In, Garbage Out!**

Consider the following setup that we will use for the remainder of the exercise. We choose a sample size of 50.

```
n = 50
set.seed(420)
x_1 = runif(n, 0, 5)
x_2 = runif(n, -2, 2)
```

Consider the model,

$$Y = 4 + 1x_1 + 0x_2 + \epsilon.$$

That is,

- $\beta_0 = 4$
- $\beta_1 = 1$
- $\beta_2 = 0$

We now simulate `y_1` in a manner that does **not** violate any assumptions, which we will verify. In this case $\epsilon \sim N(0, 1)$.

```
library(lmtest)
```

```
## Warning: package 'lmtest' was built under R version 3.5.2
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 3.5.2
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
set.seed(1)
y_1 = 4 + 1 * x_1 + 0 * x_2 + rnorm(n = n, mean = 0, sd = 1)
fit_1 = lm(y_1 ~ x_1 + x_2)
bptest(fit_1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  fit_1
## BP = 4.4, df = 2, p-value = 0.1
```

Then, we simulate `y_2` in a manner that **does** violate assumptions, which we again verify. In this case $\epsilon \sim N(0, \sigma = |x_2|)$.

```
set.seed(1)
y_2 = 4 + 1 * x_1 + 0 * x_2 + rnorm(n = n, mean = 0, sd = abs(x_2))
fit_2 = lm(y_2 ~ x_1 + x_2)
bptest(fit_2)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  fit_2
## BP = 8.4, df = 2, p-value = 0.01
```

**(a)** Use the following code after changing `birthday` to your birthday.

```
num_sims = 2500
p_val_1 = rep(0, num_sims)
p_val_2 = rep(0, num_sims)
birthday = 19770411
set.seed(birthday)

for(i in 1:2500){
  y_1 = 4 + 1 * x_1 + 0 * x_2 + rnorm(n = n, mean = 0, sd = 1)
  fit_1 = lm(y_1 ~ x_1 + x_2)
  p_val_1[i] = summary(fit_1)$coefficient[3,4]


  y_2 = 4 + 1 * x_1 + 0 * x_2 + rnorm(n = n, mean = 0, sd = abs(x_2))
  fit_2 = lm(y_2 ~ x_1 + x_2)
  p_val_2[i] = summary(fit_2)$coefficient[3,4]
}
```

Repeat the above process of generating `y_1` and `y_2` as defined above, and fit models with each as the response `2500` times. Each time, store the p-value for testing,

$$\beta_2 = 0,$$

using both models, in the appropriate variables defined above. (You do not need to use a data frame as we have in the past. Although, feel free to modify the code to instead use a data frame.)

**(b)** What proportion of the `p_val_1` values is less than 0.01? Less than 0.05? Less than 0.10? What proportion of the `p_val_2` values is less than 0.01? Less than 0.05? Less than 0.10? Arrange your results in a table. Briefly explain these results.

```
library(knitr)
Proportion_report = data.frame(
  p_value = c("p_value_1","p_value_1","p_value_1","p_value_2","p_value_2","p_value
_2"),
  alpha_value = c("< 0.01","< 0.05","< 0.1","< 0.01","< 0.05","< 0.1"),
  Proportion = c(mean(p_val_1<0.01),mean(p_val_1<0.05),mean(p_val_1<0.1),mean(p_va
l_2<0.01),mean(p_val_2<0.05),mean(p_val_2<0.1))
)

kable(Proportion_report, format = "pandoc",padding = 2,caption = "Summary of Propo
rtion Results")
```

Summary of Proportion Results

| p_value | alpha_value | Proportion |
|---|---|---|
| p_value_1 | < 0.01 | 0.0096 |
| p_value_1 | < 0.05 | 0.0492 |
| p_value_1 | < 0.1 | 0.1040 |
| p_value_2 | < 0.01 | 0.0292 |
| p_value_2 | < 0.05 | 0.0980 |
| p_value_2 | < 0.1 | 0.1672 |

From the above table, it seems the proportion of the P_value_1 for any given alpha value is less than as compared to the p_value_2. Since the assumption of constant varience is violated for the simulation 2, the p value for the simulation of the data 2 seems not equal distribution and parameters estimation shouldn't be correct.

---

# Exercise 4 (Corrosion Data)

For this exercise, we will use the `corrosion` data, which can be found in the `faraway` package. After loading the `faraway` package, use `?corrosion` to learn about this dataset.

```
library(faraway)
```

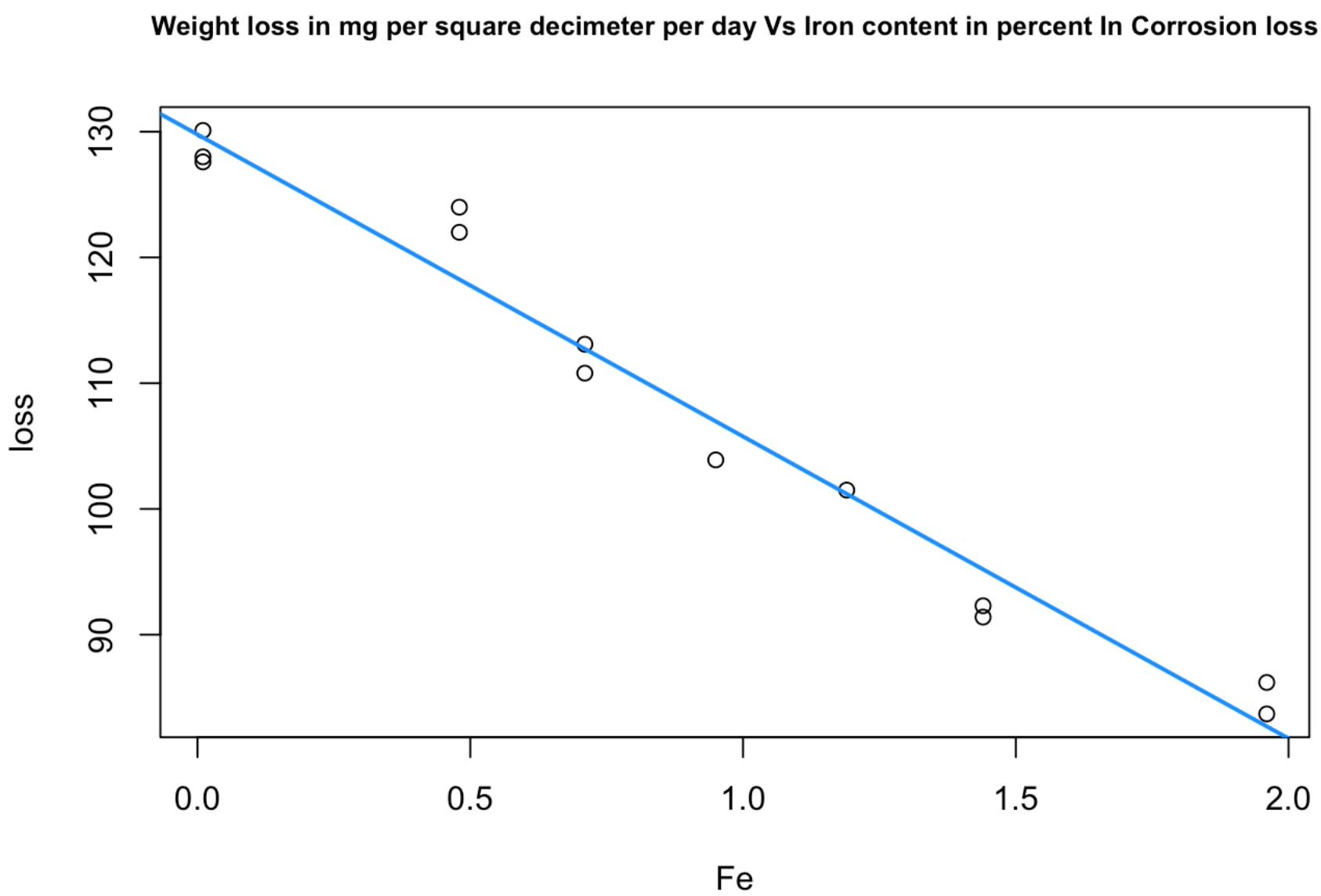
**(a)** Fit a simple linear regression with `loss` as the response and `Fe` as the predictor. Plot a scatterplot and add the fitted line. Check the assumptions of this model.

```
corrosion_data = faraway::corrosion
corrosion_slr_fit = lm(loss ~ Fe, data = corrosion_data)
corrosion_slr_fit
```

```
##
## Call:
## lm(formula = loss ~ Fe, data = corrosion_data)
##
## Coefficients:
## (Intercept)           Fe
##         130          -24
```

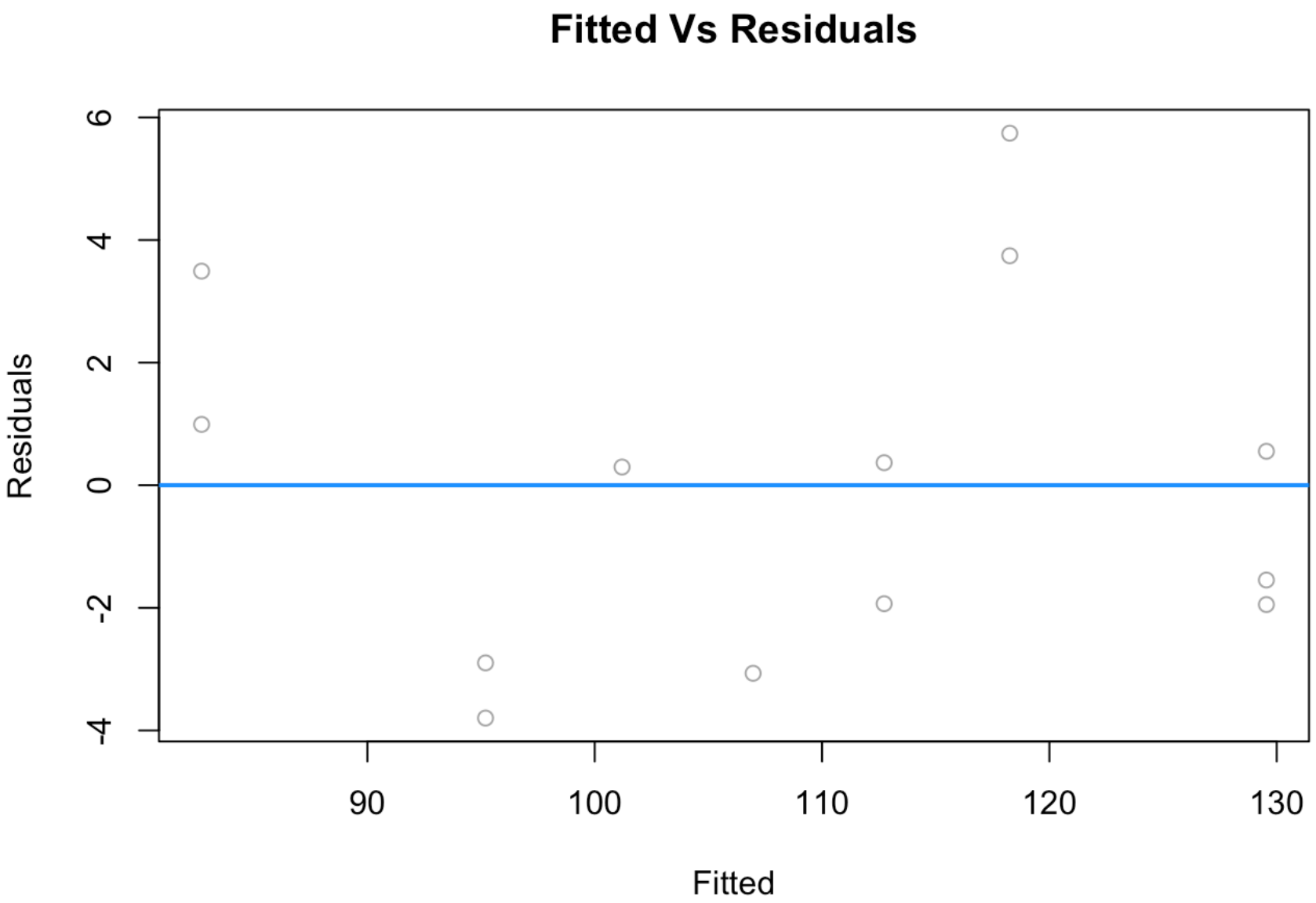
Below is the scatter plot between Weight loss in mg per square decimeter per day Vs Iron content in percent In Corrosion loss and the fitted regression line

```
plot(loss~Fe, data=corrosion_data,pch = 1, cex = 1, main = "Weight loss in mg per
square decimeter per day Vs Iron content in percent In Corrosion loss",cex.main =
0.8)
abline(corrosion_slr_fit, col = "dodgerblue", lwd = 2, lty = 1)
```



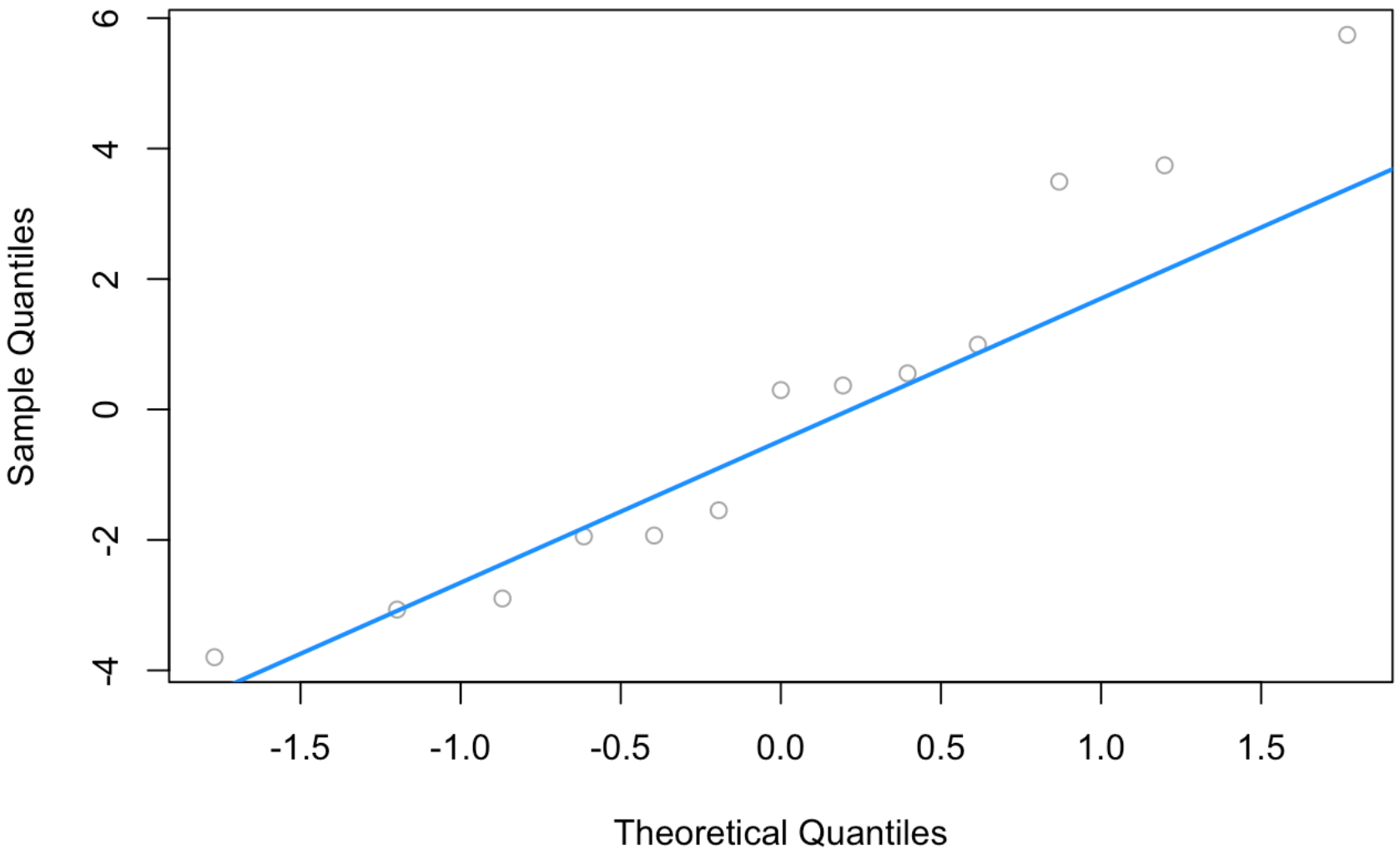

**Assumption of constant varience**

```
#Plot the Fitted Vs Residuals of Corrosion Data
plot(fitted(corrosion_slr_fit),resid(corrosion_slr_fit),col = "darkgrey", pch = 1,
cex = 1, xlab = "Fitted", ylab = "Residuals", main = "Fitted Vs Residuals")
abline(h = 0, col = "dodgerblue", lwd = 2, lty = 1)
```




**Comments:** The diagnostic plots between Fitted Versus Residuals for the SLR of the corrosion dataset seems that, the **constant variance assumption is suspect**, as the grey circles are not uniformly distributed accross the blue line, for any fitted values

```
#Plot the Q-Q Plot
qqnorm(resid(corrosion_slr_fit),col = "darkgrey", pch = 1, cex = 1, main = "Q-Q Pl
ot")
qqline(resid(corrosion_slr_fit), col = "dodgerblue", lwd = 2, lty = 1)
```




**Comments:** The diagnostic plots Q-Q plot of the SLR of the corrosion dataset seems that, the **normality assumption is suspect**, though the grey circles have follow the blue line, seems have fat tail for the higher quantiles.

**(b)** Fit higher order polynomial models of degree 2, 3, and 4. For each, plot a fitted versus residuals plot and comment on the constant variance assumption. Based on those plots, which of these three models do you think are acceptable? Use a statistical test(s) to compare the models you just chose. Based on the test, which is preferred? Check the normality assumption of this model. Identify any influential observations of this model.

```
corrosion_model_1 = lm(loss ~ Fe+I(Fe^2), data = corrosion_data)
corrosion_model_2 = lm(loss ~ Fe+I(Fe^2)+I(Fe^3), data = corrosion_data)
corrosion_model_3 = lm(loss ~ Fe+I(Fe^2)+I(Fe^3)+I(Fe^4), data = corrosion_data)

corrosion_model_1
```

```
##
## Call:
## lm(formula = loss ~ Fe + I(Fe^2), data = corrosion_data)
##
## Coefficients:
## (Intercept)           Fe      I(Fe^2)
##      130.32       -26.22          1.16
```

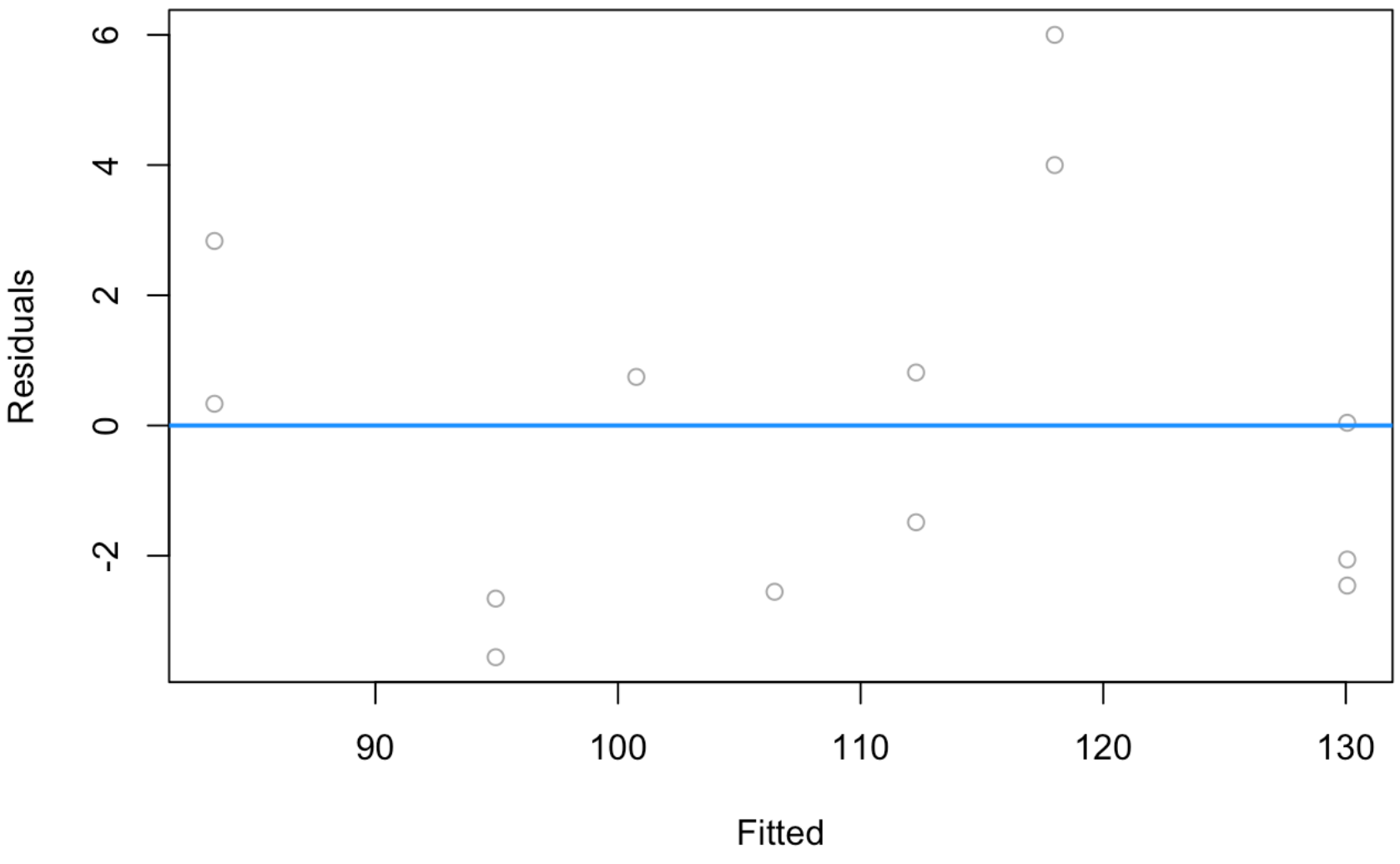
```
corrosion_model_2
```

```
##
## Call:
## lm(formula = loss ~ Fe + I(Fe^2) + I(Fe^3), data = corrosion_data)
##
## Coefficients:
## (Intercept)           Fe      I(Fe^2)      I(Fe^3)
##      128.89        -5.52       -29.21        10.52
```

```
corrosion_model_3
```

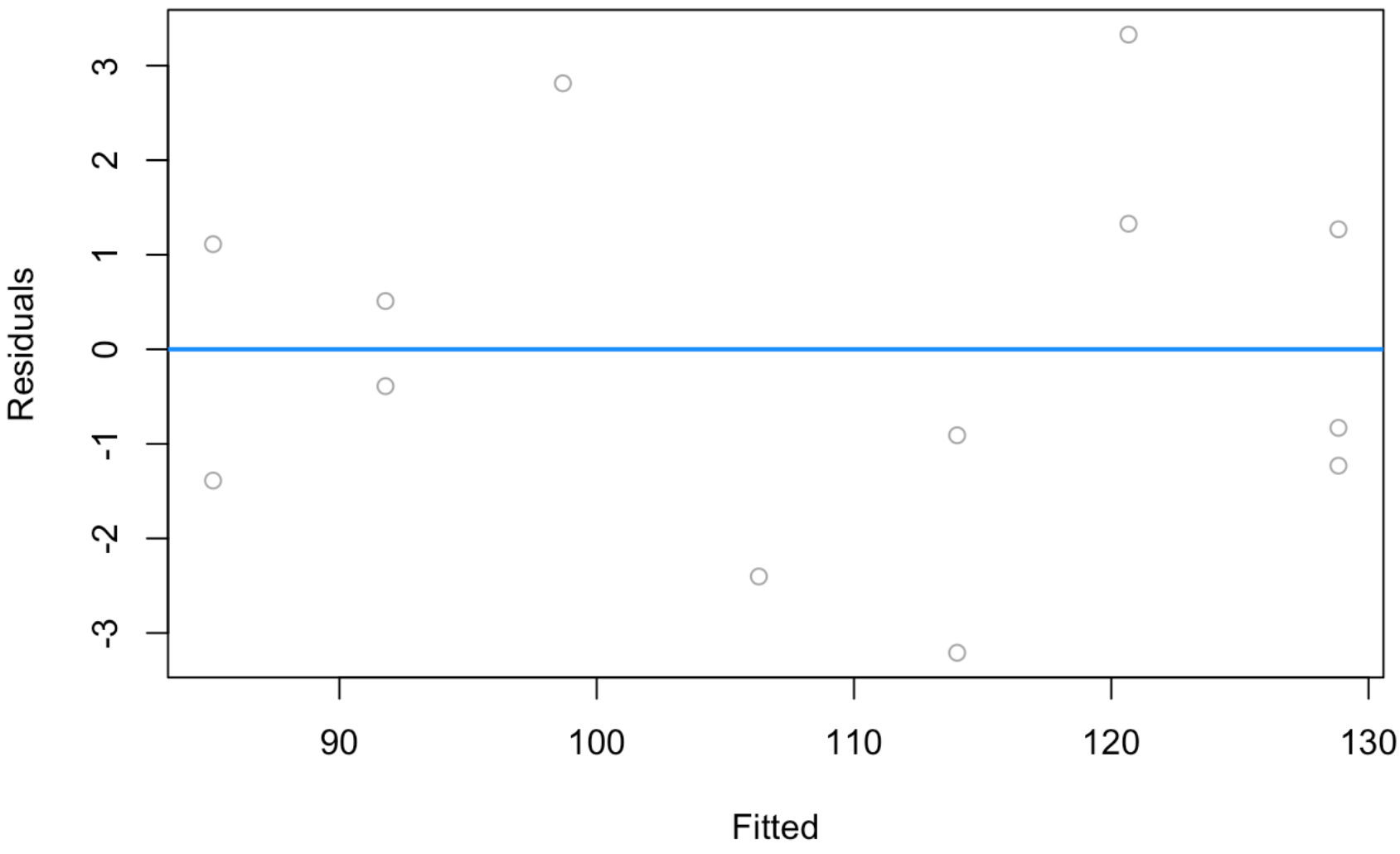
```
##
## Call:
## lm(formula = loss ~ Fe + I(Fe^2) + I(Fe^3) + I(Fe^4), data = corrosion_data)
##
## Coefficients:
## (Intercept)           Fe      I(Fe^2)      I(Fe^3)      I(Fe^4)
##       128.5         19.9        -99.2         68.8        -14.9
```

```
plot(fitted(corrosion_model_1),resid(corrosion_model_1),col = "darkgrey", pch = 1,
cex = 1, xlab = "Fitted", ylab = "Residuals", main = "Fitted Vs Residuals - polyno
mial models of degree 2")
abline(h = 0, col = "dodgerblue", lwd = 2, lty = 1)
```



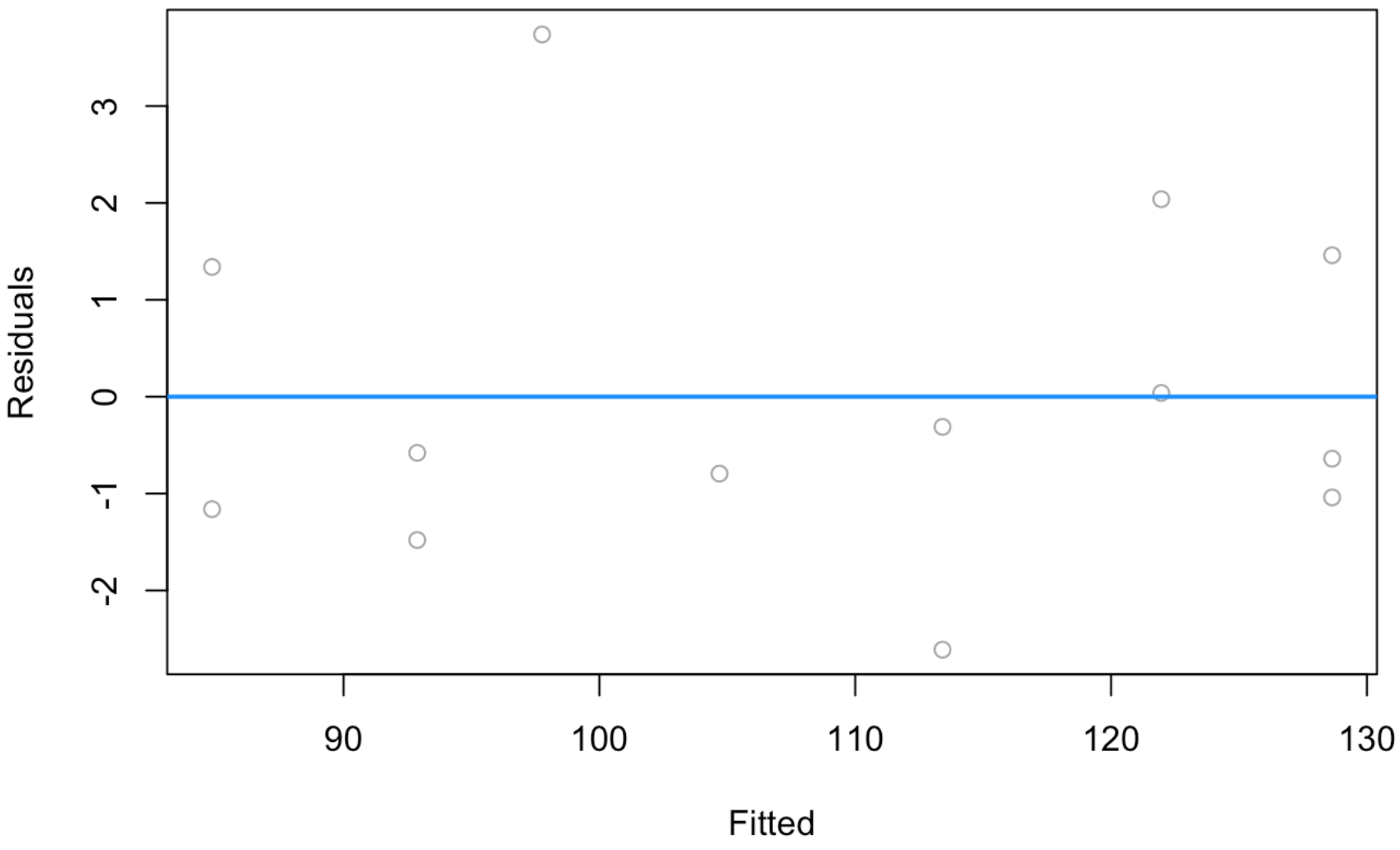

**Comments:** The diagnostic plots between Fitted Versus Residuals for the corrosion dataset for polynomial degree 2 seems that, the **constant variance assumption is suspect**, as the grey circles are not uniformly distributed accross the blue line, for any fitted values

```
plot(fitted(corrosion_model_2),resid(corrosion_model_2),col = "darkgrey", pch = 1,
cex = 1, xlab = "Fitted", ylab = "Residuals", main = "Fitted Vs Residuals - polyno
mial models of degree 3")
abline(h = 0, col = "dodgerblue", lwd = 2, lty = 1)
```




**Comments:** The diagnostic plots between Fitted Versus Residuals for the corrosion dataset for polynomial degree 3 seems that, the **constant variance assumption is suspect**, as the grey circles are not uniformly distributed accross the blue line, for any fitted values

```
plot(fitted(corrosion_model_3),resid(corrosion_model_3),col = "darkgrey", pch = 1,
cex = 1, xlab = "Fitted", ylab = "Residuals", main = "Fitted Vs Residuals - polyno
mial models of degree 4")
abline(h = 0, col = "dodgerblue", lwd = 2, lty = 1)
```




**Comments:** The diagnostic plots between Fitted Versus Residuals for the corrosion dataset for polynomial degree 4 seems that, the **constant variance assumption is suspect**, as the grey circles are not uniformly distributed accross the blue line, for any fitted values

**(b.1) Which of these three models do you think are acceptable** From the above three plots, it seems **Model 2 (models have fitted of Polynomial with degree 3) is accepatble**, as though the residuals are not uniform accross the blue line, however is better as compared to other two model (Polynomial of degree 2 and 4)

**(b.2) Statistical test(s) to compare the models you just chose**

Let do the F Test between the Model 1 (Polynomial of degree 2) and Model 2 (Polynomial of degree 3)

```
anova(corrosion_model_1,corrosion_model_2)
```

```
## Analysis of Variance Table
##
## Model 1: loss ~ Fe + I(Fe^2)
## Model 2: loss ~ Fe + I(Fe^2) + I(Fe^3)
##   Res.Df   RSS Df Sum of Sq  F Pr(>F)
## 1     10 100.1
## 2      9  45.1  1        55 11  0.009 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It seems the P value is : **0.009** which is small if we take value of $\alpha$ (0.05), hence we **reject the Model 1 (Polynomial of degree 2)**

Since we reject the Model 1 (Polynomial of degree 2), let's compare the Model 2 (Polynomial of degree 3) and Model 3 (Polynomial of degree 4) with F Test and verify the p value for $\alpha$ (0.05)

```
anova(corrosion_model_2,corrosion_model_3)
```

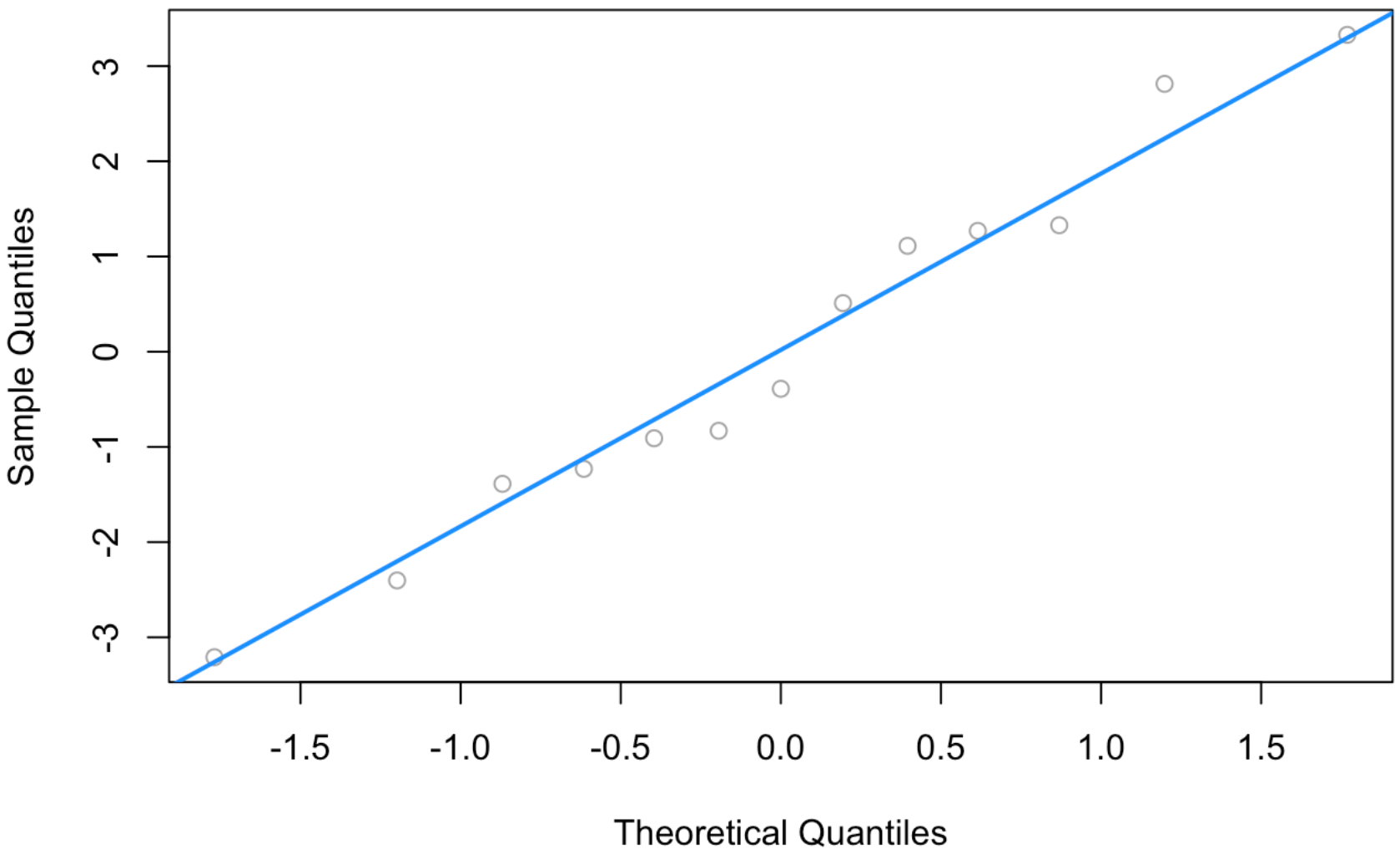
```
## Analysis of Variance Table
##
## Model 1: loss ~ Fe + I(Fe^2) + I(Fe^3)
## Model 2: loss ~ Fe + I(Fe^2) + I(Fe^3) + I(Fe^4)
##   Res.Df  RSS Df Sum of Sq   F Pr(>F)
## 1      9 45.1
## 2      8 35.0  1      10.1 2.3   0.17
```

It seems the P value is : **0.1675** which is higher than that of $\alpha$ (0.05), hence we **failed to reject the Model 2 (Polynomial of degree 3)**

Based on the test, **Model 2 (Polynomial of degree 3) is preferred**

**(b.3) normality assumption of this model**

```
qqnorm(resid(corrosion_model_2),col = "darkgrey", pch = 1, cex = 1, main = "Q-Q pl
ot corrosion where the predictor is fitted with Polynomial of Degree 3",cex.main =
0.8)
qqline(resid(corrosion_model_2),col = "dodgerblue", lwd = 2, lty = 1)
```




**Comments:** The diagnostic plots Q-Q plot of the corrosion dataset of Polynomial of degree 3, the **normality assumption is not suspect**, as the grey circles followed the blue line

**Identify any influential observations of this model**

```
high_cd = cooks.distance(corrosion_model_2) > (4/length(cooks.distance(corrosion_m
odel_2)))
corrosion_data[high_cd,]
```

```
## [1] Fe   loss
## <0 rows> (or 0-length row.names)
```

It seems, there is **no influential observation for the Model 2 (Polynomial of degree 3)**

---

# Exercise 5 (Diamonds)

The data set `diamonds` from the `ggplot2` package contains prices and characteristics of 54,000 diamonds. For this exercise, use `price` as the response variable $y$, and `carat` as the predictor $x$. Use `?diamonds` to learn more.

```
library(ggplot2)
```

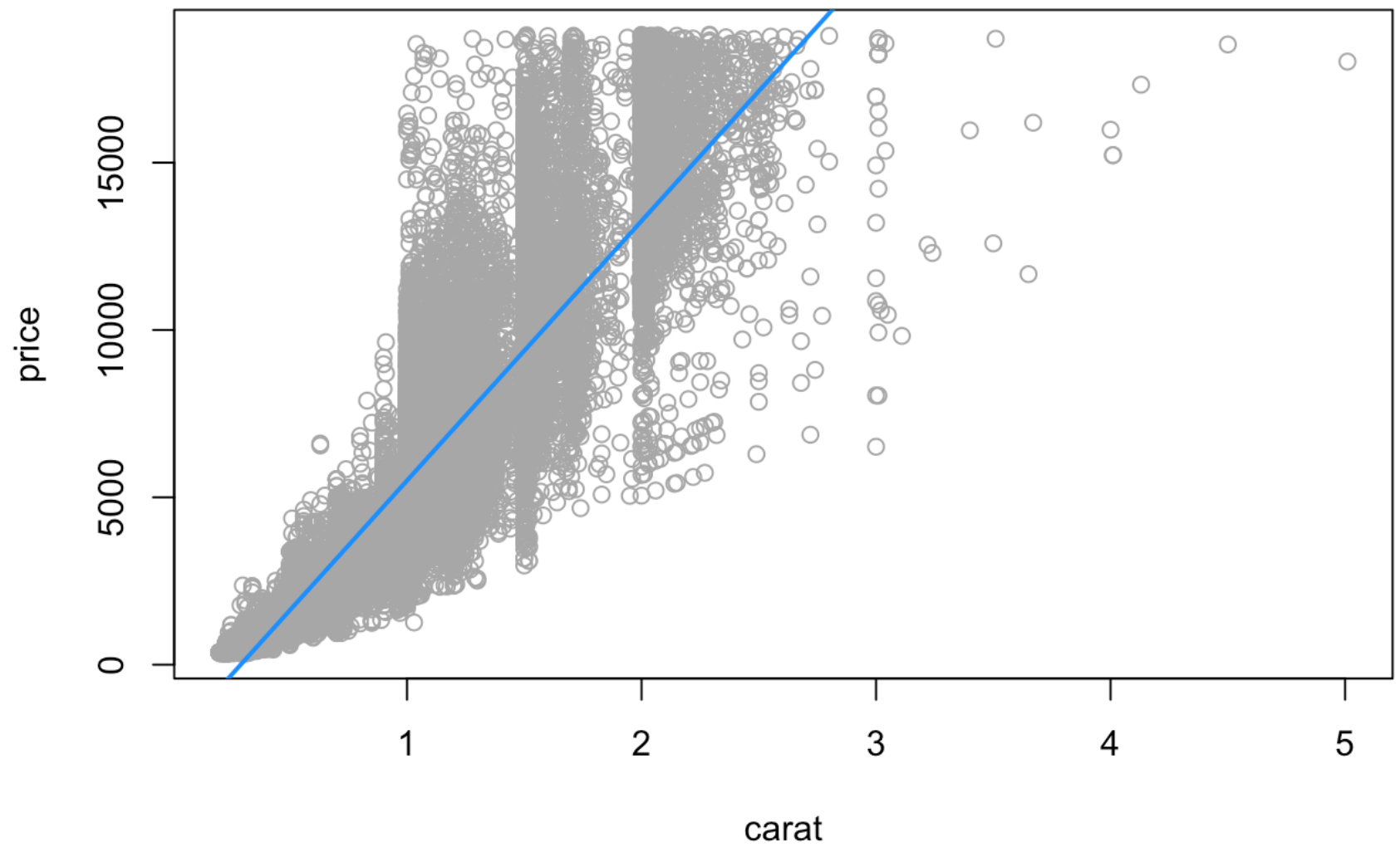
**(a)** Fit a linear model with `price` as the response variable $y$, and `carat` as the predictor $x$. Return the summary information of this model.

```
diamonds_data = ggplot2::diamonds
diamonds_slr = lm(price~carat,data = diamonds_data)
summary(diamonds_slr)
```

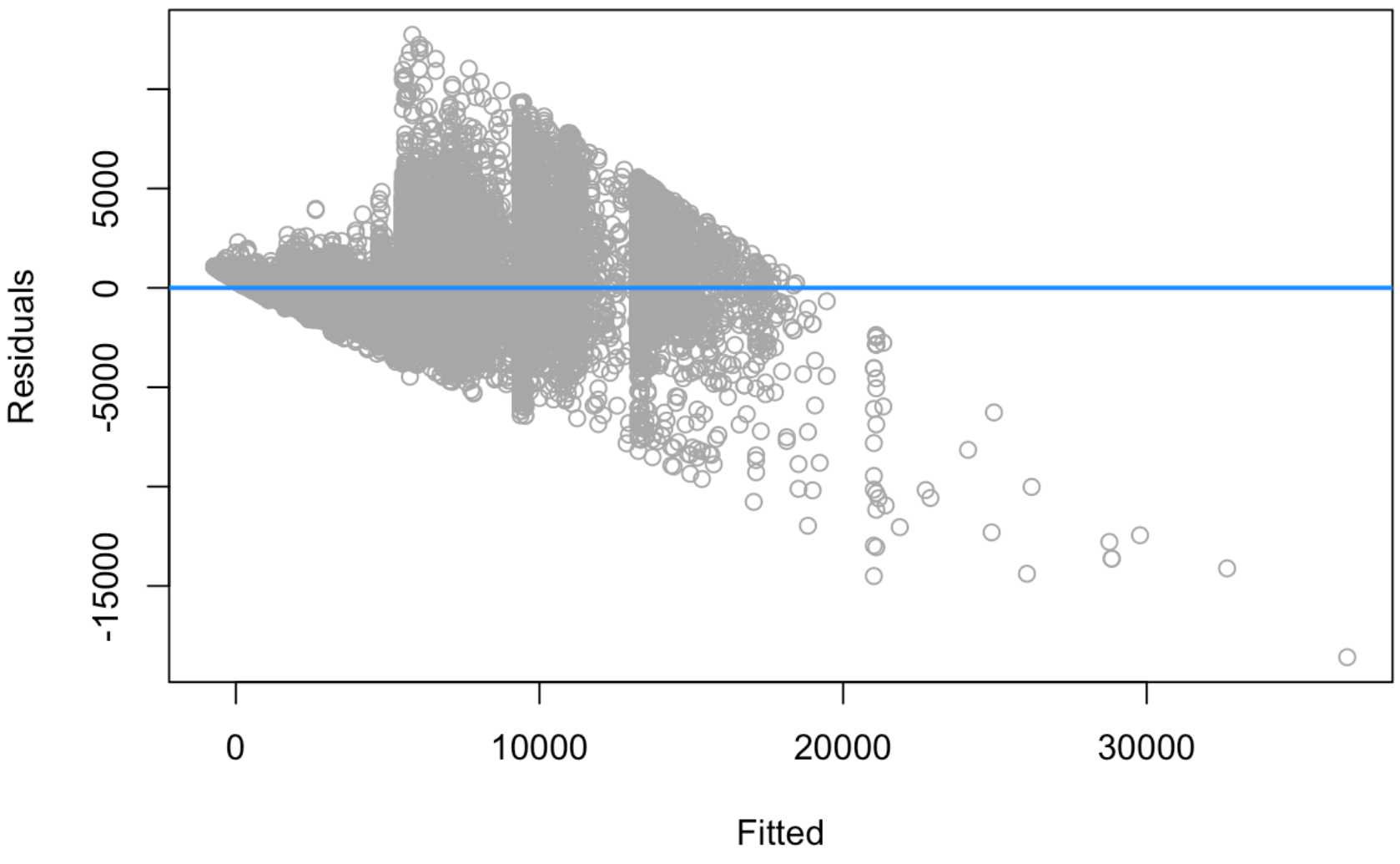
```
##
## Call:
## lm(formula = price ~ carat, data = diamonds_data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -18585   -805    -19    537  12732
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2256.4       13.1    -173   <2e-16 ***
## carat         7756.4       14.1     551   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1550 on 53938 degrees of freedom
## Multiple R-squared:  0.849,  Adjusted R-squared:  0.849
## F-statistic: 3.04e+05 on 1 and 53938 DF,  p-value: <2e-16
```

**(b)** Plot a scatterplot of price versus carat and add the line for the fitted model in part **(a)**. Using a fitted versus residuals plot and/or a Q-Q plot, comment on the diagnostics.

```
plot(price~carat,data = diamonds_data,col = "darkgrey", pch = 1, cex = 1,main = "C
arat Vs Price of the Diamonds")
abline(diamonds_slr,col = "dodgerblue", lwd = 2, lty = 1)
```

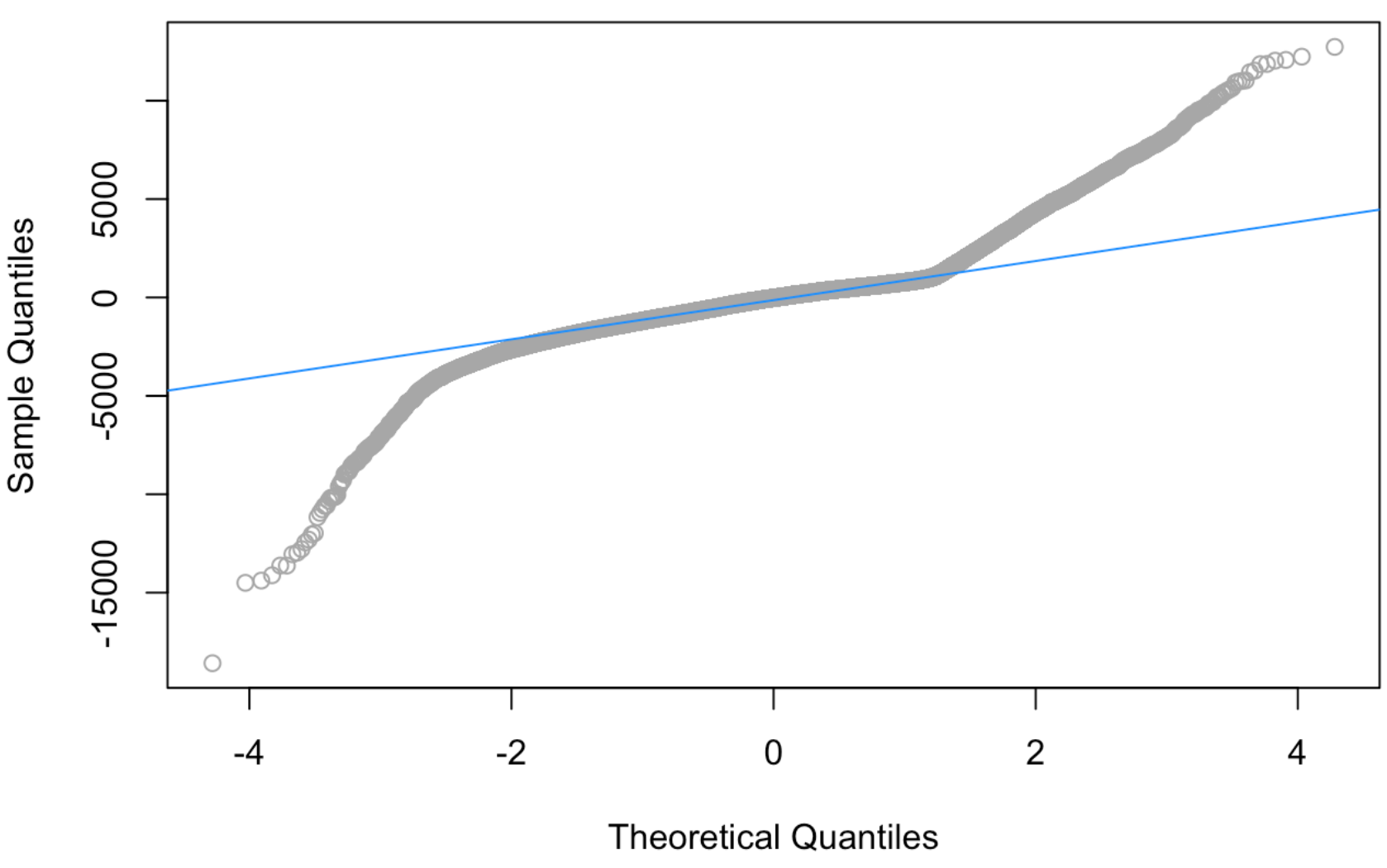



```
plot(fitted(diamonds_slr),resid(diamonds_slr),col = "darkgrey", pch = 1, cex = 1,
xlab = "Fitted", ylab = "Residuals", main = "Fitted Versus Residuals for the SLR f
or Diamond dataset")
abline ( h = 0, col = "dodgerblue", lwd = 2, lty = 1)
```




**Comments:** The diagnostic plots between Fitted Versus Residuals for the SLR of the diamond dataset seems that, the **constant variance assumption is suspect**, as the grey circles are not uniformly distributed accross the blue line. For the lower fitted value it's on the upper side and grow uniformly accross the blue line, where as for the higher fitted values, it below the blue line.

```
qqnorm(resid(diamonds_slr), col = "darkgrey", pch = 1, cex = 1, main = "Q-Q plot o
f the SLR for the Diamond dataset")
qqline(resid(diamonds_slr), col = "dodgerblue", lwd = 1, lty = 1)
```



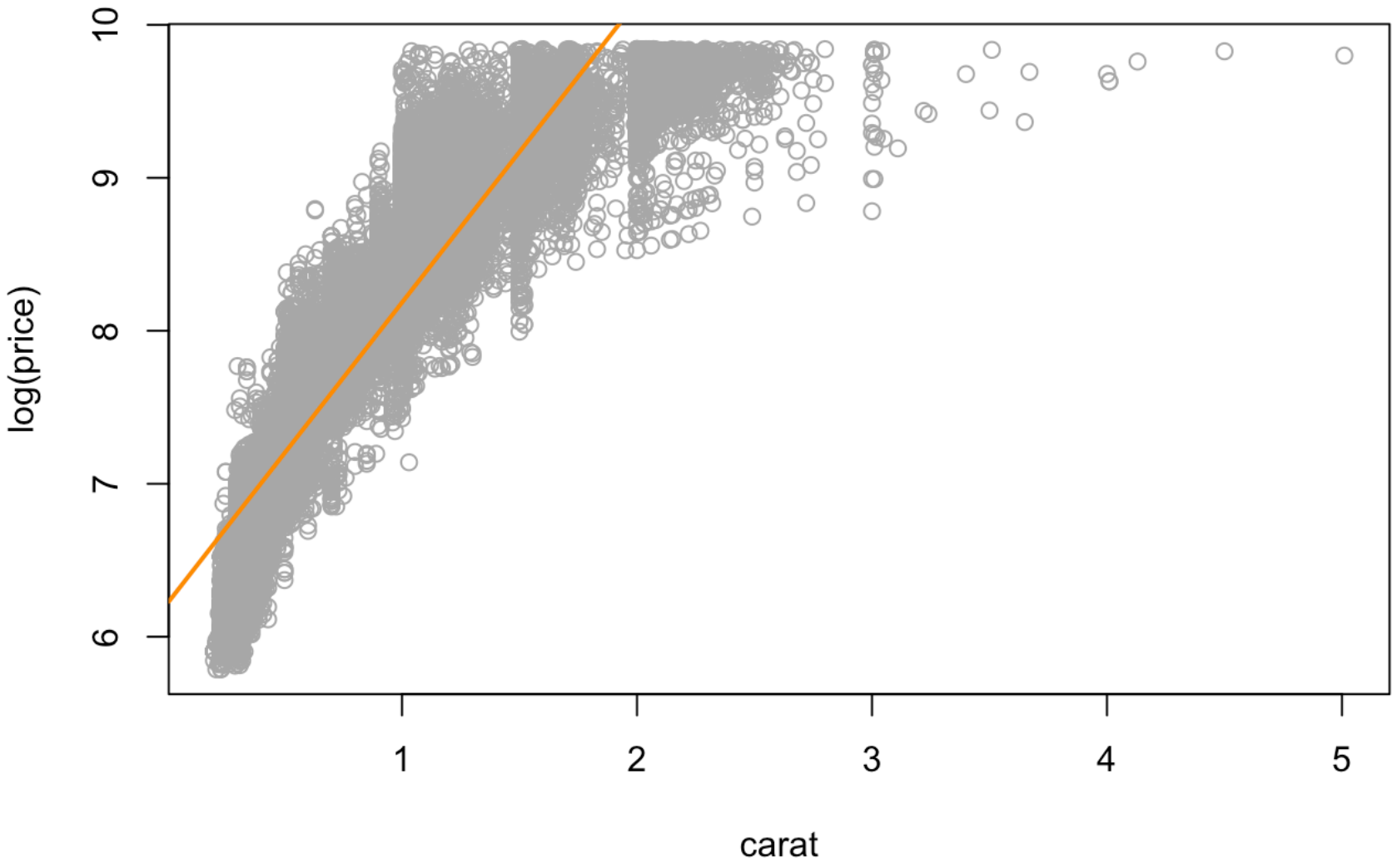

**Comments:** The diagnostic plots Q-Q plot of the SLR of the diamond dataset seems that, the **normality assumption is suspect**, as the grey circles are not follow the blue line, seems higher above the blue line for higher value of Therotical quantiles and below the blue line for lower therotical quantiles.

**(c)** Seeing as the price stretches over several orders of magnitude, it seems reasonable to try a log transformation of the response. Fit a model with a logged response, plot a scatterplot of log-price versus carat and add the line for the fitted model, then use a fitted versus residuals plot and/or a Q-Q plot to comment on the diagnostics of the model.

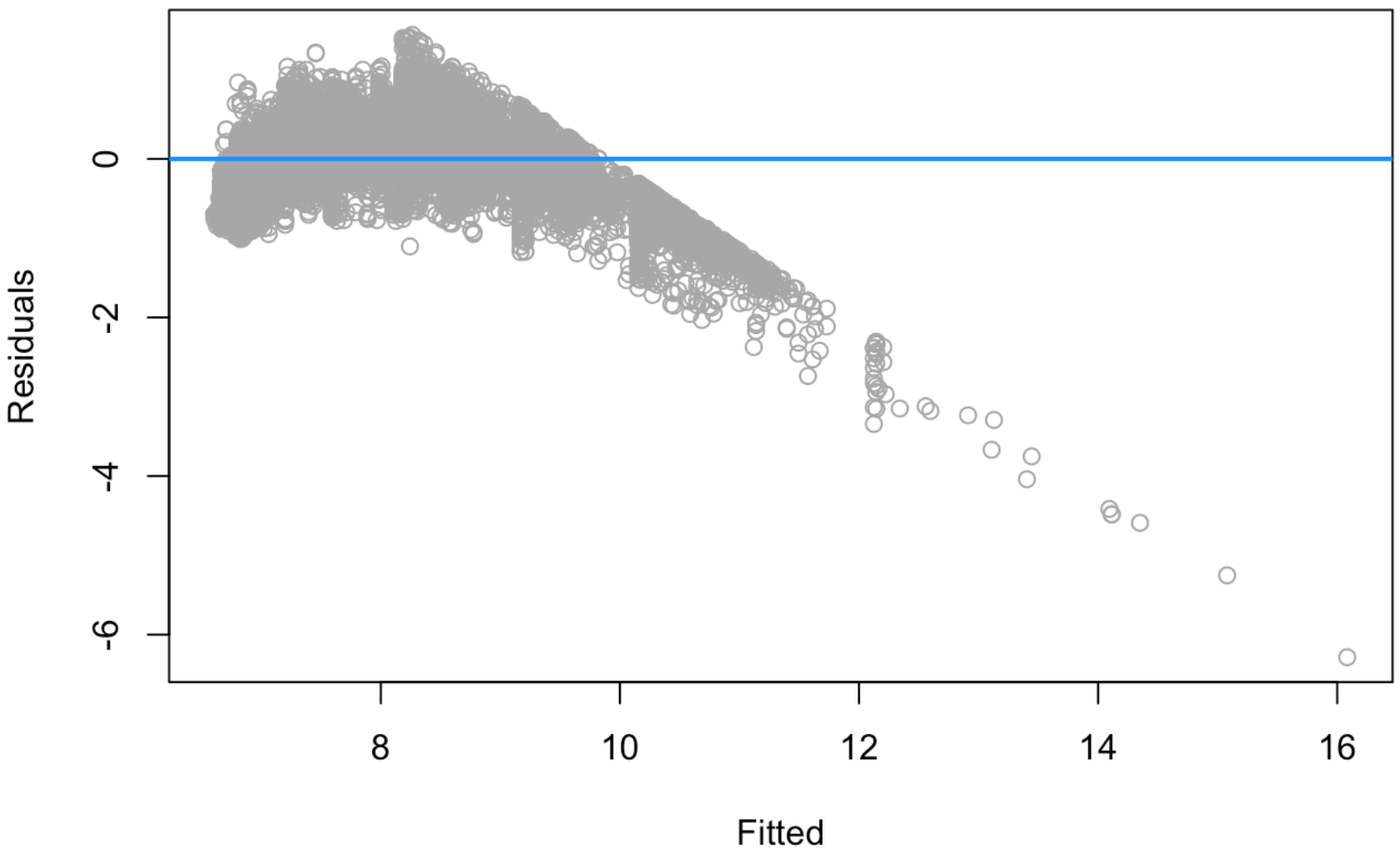
```
diamonds_slr_log = lm(log(price)~carat, data = diamonds_data)
diamonds_slr_log
```

```
##
## Call:
## lm(formula = log(price) ~ carat, data = diamonds_data)
##
## Coefficients:
## (Intercept)        carat
##        6.22         1.97
```

```
plot(log(price)~carat, data = diamonds_data, col = "darkgrey", pch = 1, cex = 1)
abline(diamonds_slr_log,lwd = 2, col = "darkorange", lty = 1)
```

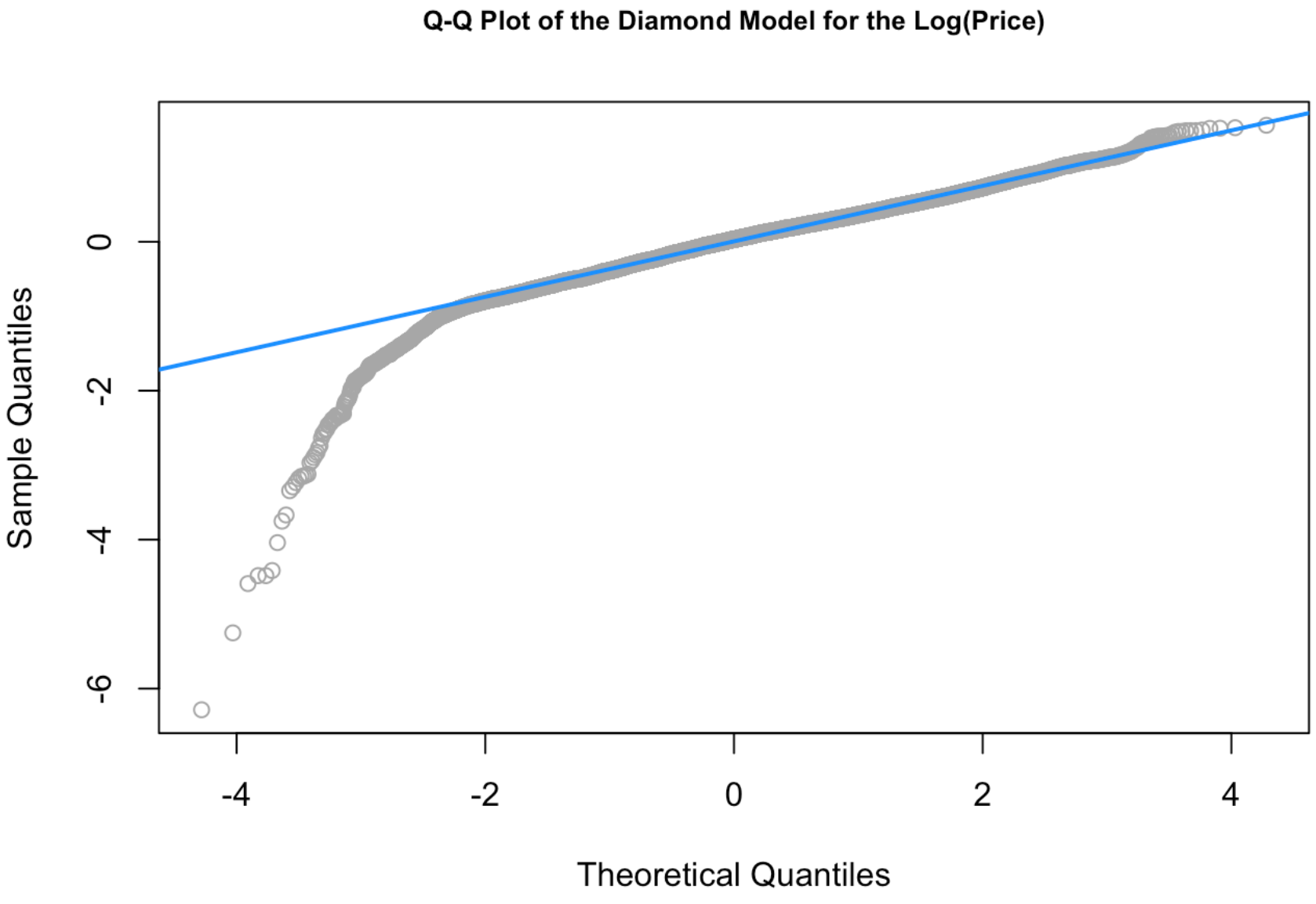



```
#Plot the Fitted Versus Residual Plot
plot(fitted(diamonds_slr_log),resid(diamonds_slr_log),col = "darkgrey", pch = 1, c
ex = 1, xlab = "Fitted", ylab = "Residuals", main = "Fitted Versus Residual Plot o
f Diamond model for Log(Price)",cex.main = 0.8)
abline(h = 0, col = "dodgerblue", lwd = 2, lty = 1)
```



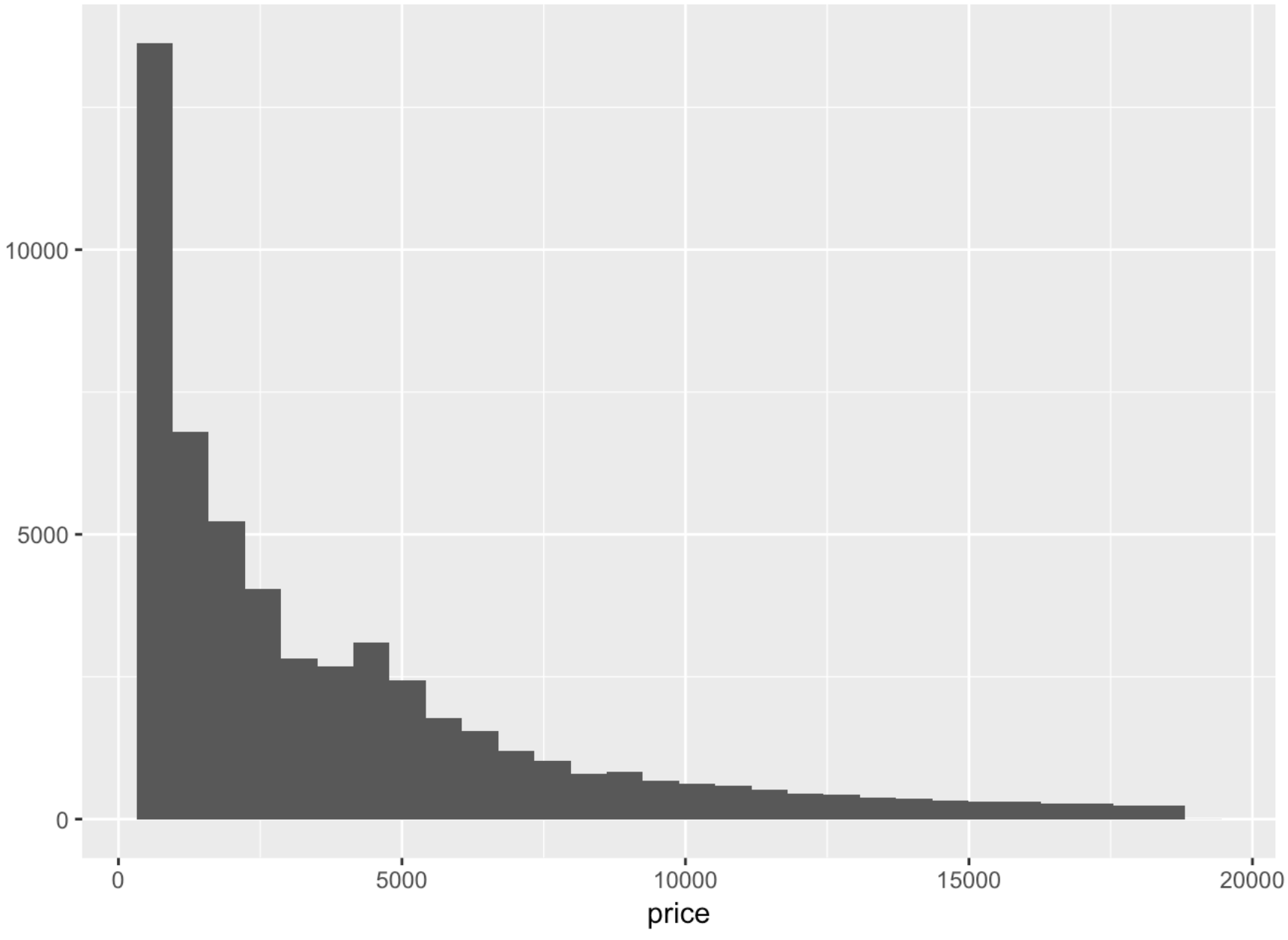

**Comments:** The diagnostic plots between Fitted Versus Residuals for the log response of the SLR of the diamond dataset seems that, the **constant variance ssumption is suspect**, as the grey circles are not uniformly distributed accross the blue line. For the lower fitted value it's on the both the side side accross the blue line, however as the fitted values increses, it only present on the lower part of the blue line.

```
#Plot the Q-Q Plot
qqnorm(resid(diamonds_slr_log),col = "darkgrey", pch = 1, cex = 1, main = "Q-Q Plo
t of the Diamond Model for the Log(Price)",cex.main = 0.8)
qqline(resid(diamonds_slr_log),col = "dodgerblue", lwd = 2, lty = 1)
```




**Comments:** The diagnostic plots Q-Q plot of the log response of the SLR of the diamond dataset seems that, the **normality assumption is suspect**, though the grey circles have followed the blue line for the higher quantiles, however not dowing well for the lower quantiles.

```
qplot(price, data = diamonds, bins = 30)
```

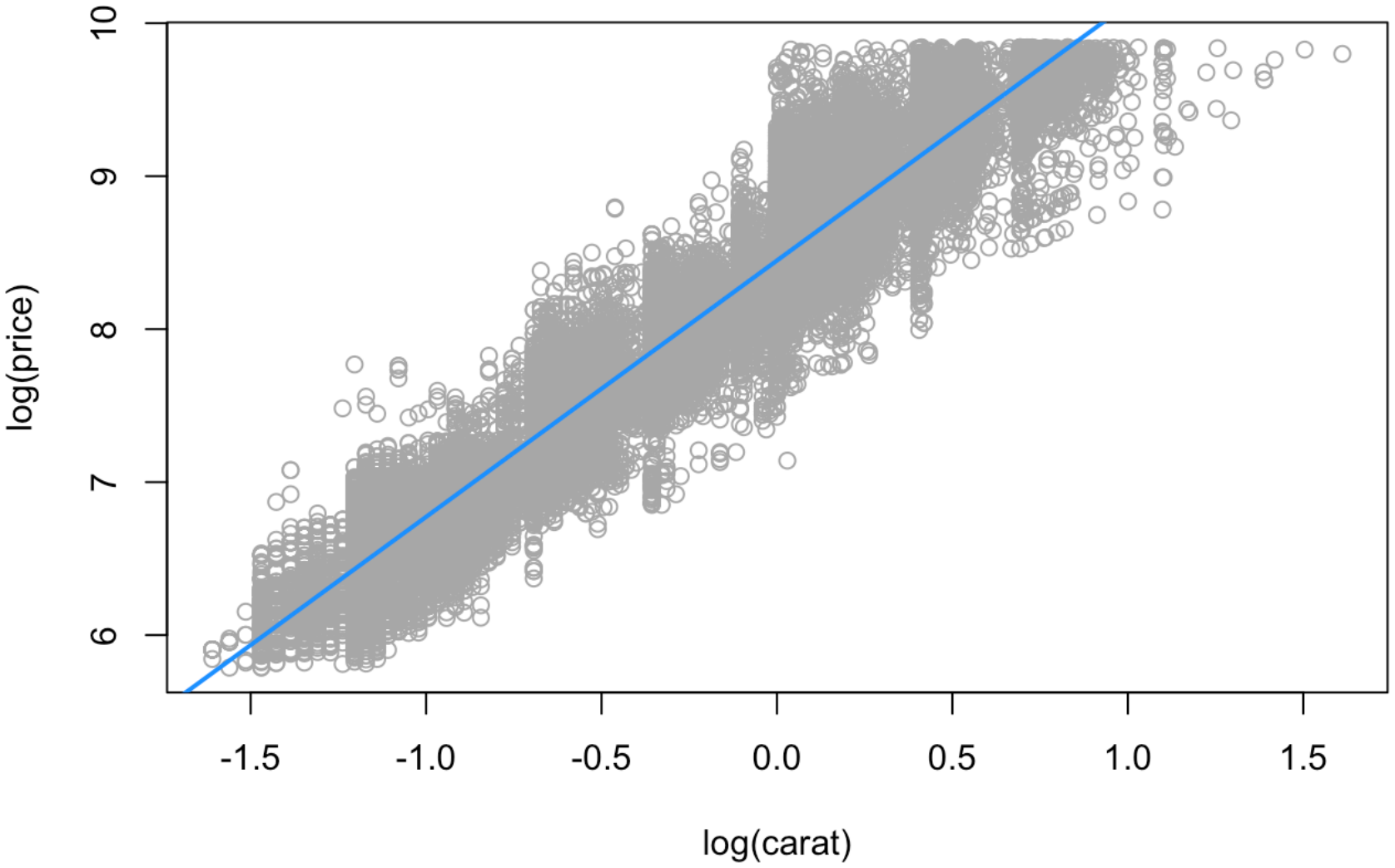



**(d)** Try adding log transformation of the predictor. Fit a model with a logged response and logged predictor, plot a scatterplot of log-price versus log-carat and add the line for the fitted model, then use a fitted versus residuals plot and/or a Q-Q plot to comment on the diagnostics of the model.

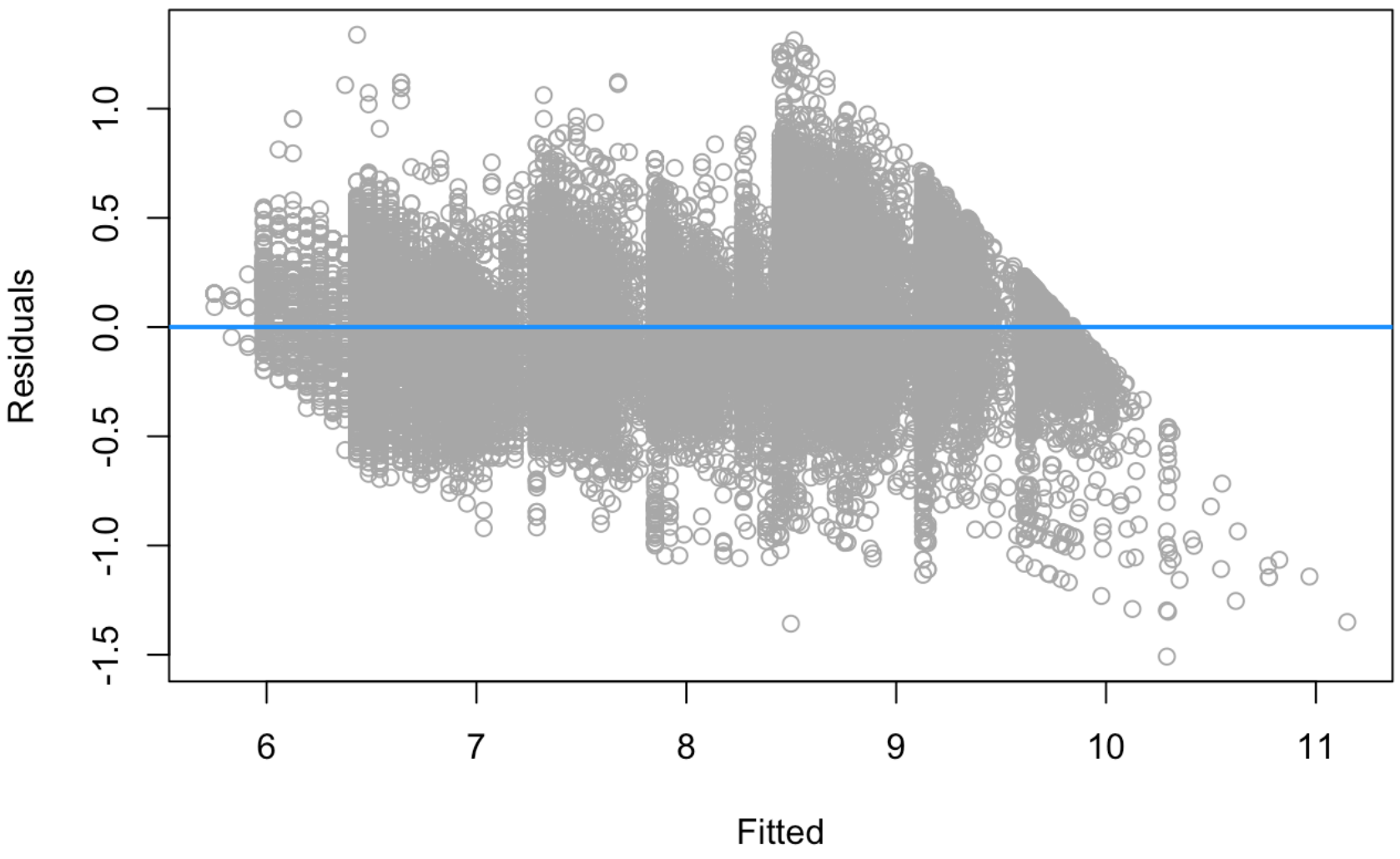
```
diamonds_slr_predict_log = lm(log(price)~log(carat), data = diamonds_data)
diamonds_slr_predict_log
```

```
##
## Call:
## lm(formula = log(price) ~ log(carat), data = diamonds_data)
##
## Coefficients:
## (Intercept)   log(carat)
##        8.45         1.68
```

```
plot(log(price)~log(carat), data = diamonds_data, col = "darkgrey", pch = 1, cex =
1,cex.main = 0.8)
abline(diamonds_slr_predict_log,lwd = 2, col = "dodgerblue", lty = 1)
```

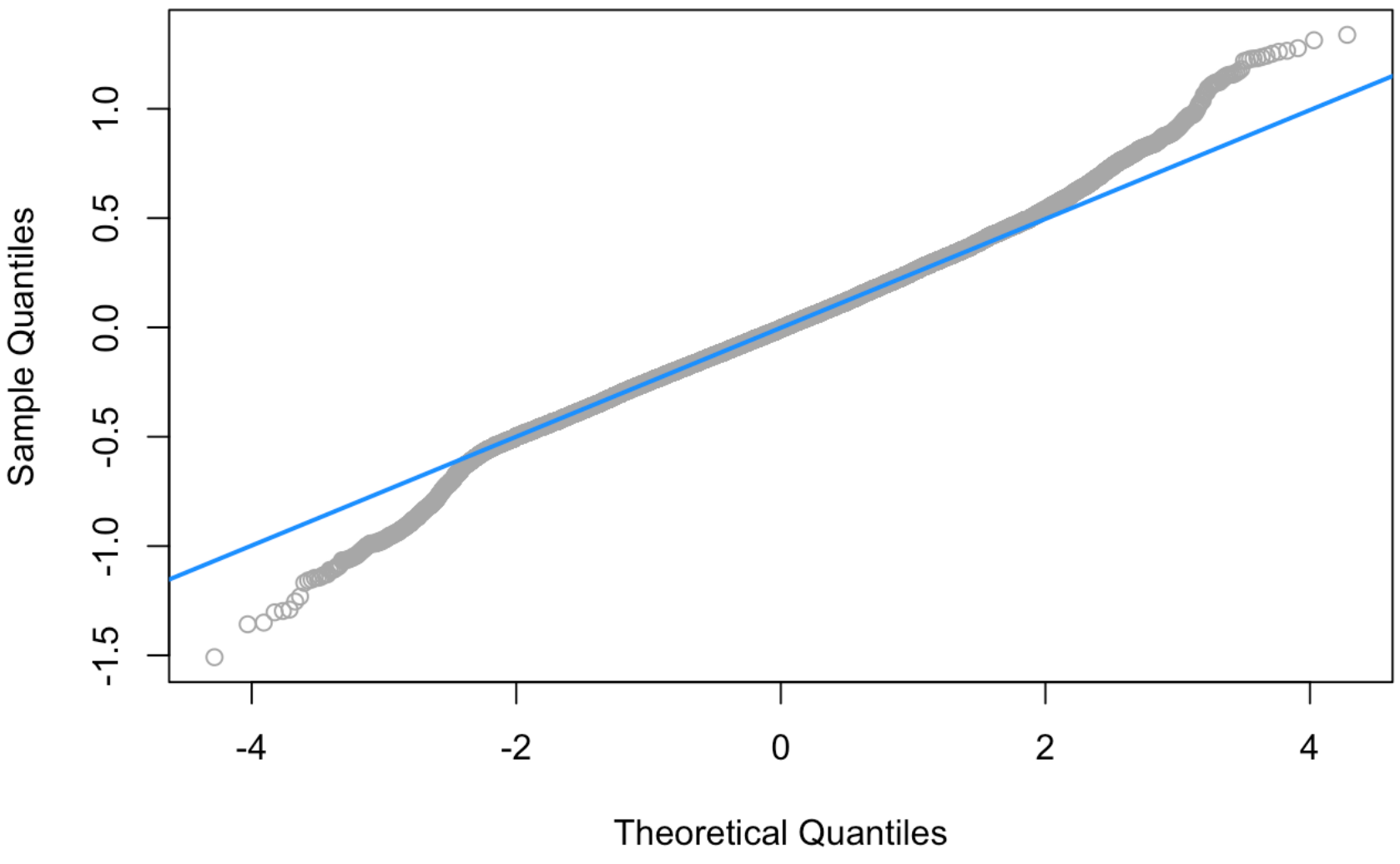



```
#Plot the Fitted Versus Residual Plot
plot(fitted(diamonds_slr_predict_log),resid(diamonds_slr_predict_log),col = "darkg
rey", pch = 1, cex = 1, xlab = "Fitted", ylab = "Residuals", main = "Fitted Versus
Residual Plot of Diamond model for Log(Price) and Log(carat)",cex.main = 0.8)
abline(h = 0, col = "dodgerblue", lwd = 2, lty = 1)
```




**Comments:** The diagnostic plots between Fitted Versus Residuals for the log response and log predictors of the SLR of the diamond dataset seems that, the **constant variance ssumption is suspect**, as the grey circles are uniformly distributed accross the blue line till the fitted value of 9.5. However after the fitted value of 9.5, the circles are appearing below the blue line, which breaks the uniformity.

```
#Plot the Q-Q Plot
qqnorm(resid(diamonds_slr_predict_log),col = "darkgrey", pch = 1, cex = 1, main =
"Q-Q Plot of the Diamond Model for the Log(Price) and Log(carat)",cex.main = 0.8)
qqline(resid(diamonds_slr_predict_log),col = "dodgerblue", lwd = 2, lty = 1)
```




**Comments:** The diagnostic plots Q-Q plot of the log response and lot predictors of the SLR of the diamond dataset seems that, the **normality assumption is suspect**, though the grey circles have followed the blue line for the for all the middle level quantiles, however not doing well for the lower and higher quantiles and have a fat tail.

**(e)** Use the model from part **(d)** to predict the price (in dollars) of a 3-carat diamond. Construct a 99% prediction interval for the price (in dollars).

```
exp(predict(diamonds_slr_predict_log,newdata = data.frame(carat = log(3)) , interv
al = "prediction", level = 0.99))
```

```
##    fit  lwr   upr
## 1 5466 2779 10752
```