# HW2_kexinx

Kexin Xie

Sep 10th 2021

## Problem 1

## Problem 2

### Part A

**1.** I want to learn how to use statistical software to facilitate my research, such as the management of data, files and code, etc.

**2.** I want to learn how to improve code efficiency.

**3.** I want to learn more advanced visualization software to make the article look more readable and attractive.

### Part B

#### Bernoulli Distribution

$$P(X = x|p) = p^x(1-p)^{1-x}; \quad x = 0, 1; \quad 0 \leq p \leq 1$$

#### Binomial Distribution

$$P(X = x|n, p) = \binom{n}{x} p^x(1-p)^{n-x}; \quad x = 0, 1, 2, ..., n; \quad 0 \leq p \leq 1$$

#### Chi squared Distribution

$$f(x|p) = \frac{1}{\Gamma(p/2)2^{p/2}} x^{p/2-1} e^{-x/2}; \quad 0 \leq x \leq \infty; \quad p = 1, 2, ...$$

## Problem 3

Reproducible research has become particularly important in modern research. Not only for the efficiency of future research, but also for the accurateness of current research. According to Sandve(2013), the steps to achieve repeatability research are as follows:

**1.** Data storage and organization is an essential step before data analysis. The data should be stored in useful, flexible, portable, nonproprietary formats or be loaded in an automated, effective way. This includes raw data, clean analysis-ready data, and steps in between.

**Comment:** I don't know how to name the data in a simple way, and which data format is the most space-saving and the easiest for subsequent operation.

**2.** Ensuring that the code is easy to read and can be reproduced for others. For instance, using the execution of programs for modifing data makes it easier to reperformed than manipulation steps and following a clean, consistent coding style make code more comprehensible.

**Comment:** I always write a lot of variable code that looks tedious, and I don't know how to make the code look concise and not verbose.

**3.** Making comments on every detail that may affect the execution of the step including the name and version of the program as well as exact parameters and inputs, in the form of simple shell scripts or makefiles at the command line, or in the form of stored workflows in a workflow management system.

**Comment:** I don't know how to use workflow management system.

**4.** Using a version control system to track evolution of code and storing all intermediate results in standardized formats. These allow us to uncover the bugs and check the environment without the need to have all executable operational.

**Comment:** I had a problem when creating the version control system that is git, but with the help of my classmates I solved the problem.

**5.** Creating a dynamic conclusions document and producing reproducible tables and figures directly from code instead of manual procedures.

**Comment:** When creating dynamic files, the format conversion often fails due to some symbol problems. I need to be extra careful every time I modify the code.

**6.** Providing enough details along with your textual interpretations so as to allow the exact underlying results, or at least some related results and some optionally suggestions, to be tracked down in the future.

**Comment:** Interpretation and description of the results should be reasonable and convincing, combined with statistical significance and actual conditions.

**7.** Providing public access to data, code, scripts, runs, and results.

**Comment:** Data and code shared on personal websites are only available as long as websites are maintained and can be difficult to transfer when researchers migrate to another domain or website provider.

## Problem 4

```
library(data.table)
covid_raw <- fread("https://opendata.ecdc.europa.eu/covid19/casedistribution/csv")
us <- covid_raw[covid_raw$countriesAndTerritories == 'United_States_of_America',]
us_filtered <- us[us$month %in% c(6:7),]
us_filtered$index <- rev(1:dim(us_filtered)[1])
fit<-lm(`Cumulative_number_for_14_days_of_COVID-19_cases_per_100000`~index, data=us_filtered)
```

### Part A

#### 1

```
library('knitr')
kable(summary(us_filtered),'simple')
```

| dateRep | day | month | year | cases | deaths | countriesAndTerr |
|---|---|---|---|---|---|---|
| Length:61 | Min. : 1.00 | Min. :6.000 | Min. :2020 | Min. :18665 | Min. : 242.0 | Length:61 |
| Class :character | 1st Qu.: 8.00 | 1st Qu.:6.000 | 1st Qu.:2020 | 1st Qu.:25540 | 1st Qu.: 500.0 | Class :character |
| Mode :character | Median :16.00 | Median :7.000 | Median :2020 | Median :45221 | Median : 767.0 | Mode :character |

| dateRep | day | month | year | cases | deaths | countriesAndTerr |
|---|---|---|---|---|---|---|
| NA | Mean :15.75 | Mean :6.508 | Mean :2020 | Mean :44666 | Mean : 791.6 | NA |
| NA | 3rd Qu.:23.00 | 3rd Qu.:7.000 | 3rd Qu.:2020 | 3rd Qu.:61796 | 3rd Qu.: 982.0 | NA |
| NA | Max. :31.00 | Max. :7.000 | Max. :2020 | Max. :78427 | Max. :2437.0 | NA |

We limit ourselves to 61 points, and there is no misssing datapoint.

**2**

```
library(stargazer)
```

```
##
## Please cite as:
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
summary(fit)
```

Call: lm(formula = `Cumulative_number_for_14_days_of_COVID-19_cases_per_100000` ~ index, data = us_filtered)

Residuals: Min 1Q Median 3Q Max -30.602 -16.555 0.738 15.196 45.251

Coefficients: Estimate Std. Error t value Pr(>|t|)
(Intercept) 42.8532 5.1649 8.297 1.72e-11 ***index 4.1065 0.1449 28.345 < 2e-16*** — Signif. codes: 0 '*** *0.001* ** *0.01* * 0.05 '.' 0.1 ' ' 1

Residual standard error: 19.92 on 59 degrees of freedom Multiple R-squared: 0.9316, Adjusted R-squared: 0.9304 F-statistic: 803.5 on 1 and 59 DF, p-value: < 2.2e-16

```
stargazer(fit,type='latex',single.raw=TRUE,title="Summary of Linear Model", align=TRUE)
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: , 9 13, 2021 - 21:43:31 % Requires LaTeX packages: dcolumn

Table 2: Summary of Linear Model

| | *Dependent variable:* |
|---|---|
| | 'Cumulative_number_for_14_days_of_COVID-19_cases_per_100000' |
| index | 4.107*** |
| | (0.145) |
| Constant | 42.853*** |
| | (5.165) |
| Observations | 61 |
| $R^2$ | 0.932 |
| Adjusted $R^2$ | 0.930 |
| Residual Std. Error | 19.922 (df = 59) |
| F Statistic | 803.464*** (df = 1; 59) |
| *Note:* | *p<0.1; **p<0.05; ***p<0.01 |

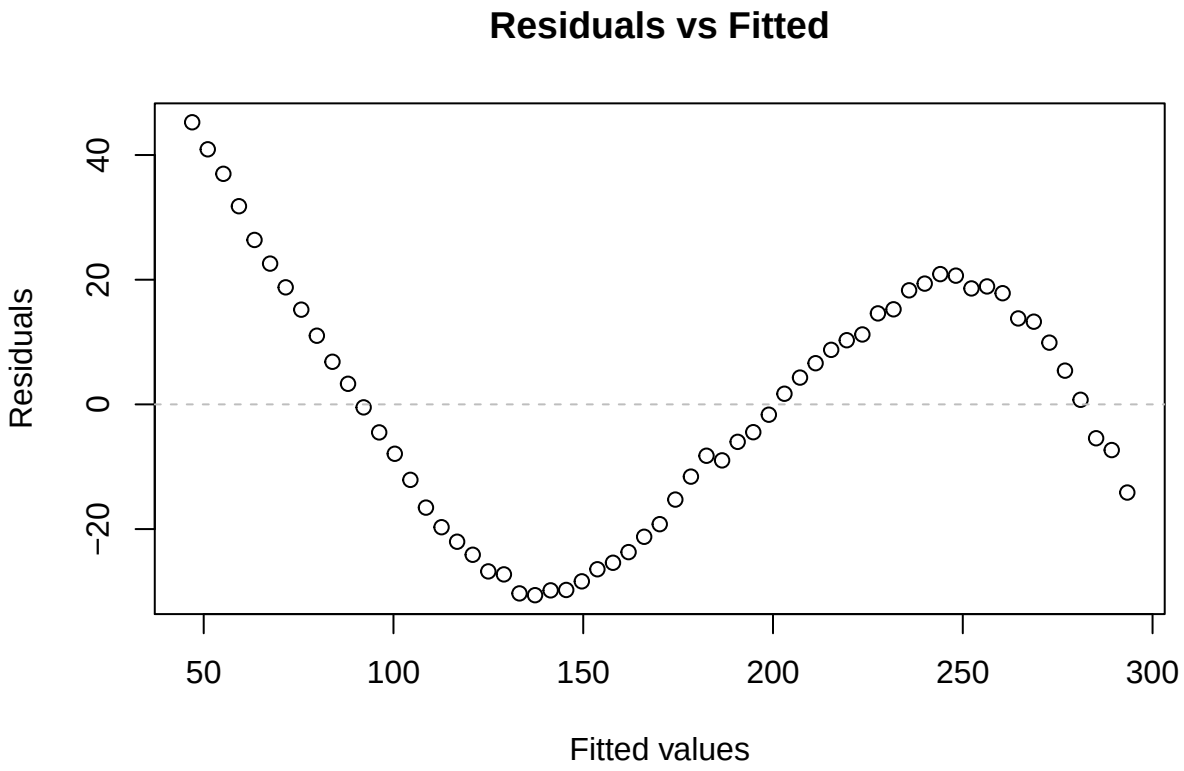
% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: , 9 13, 2021 - 21:43:31 % Requires LaTeX packages: dcolumn

Table 3: Summary of Linear Model

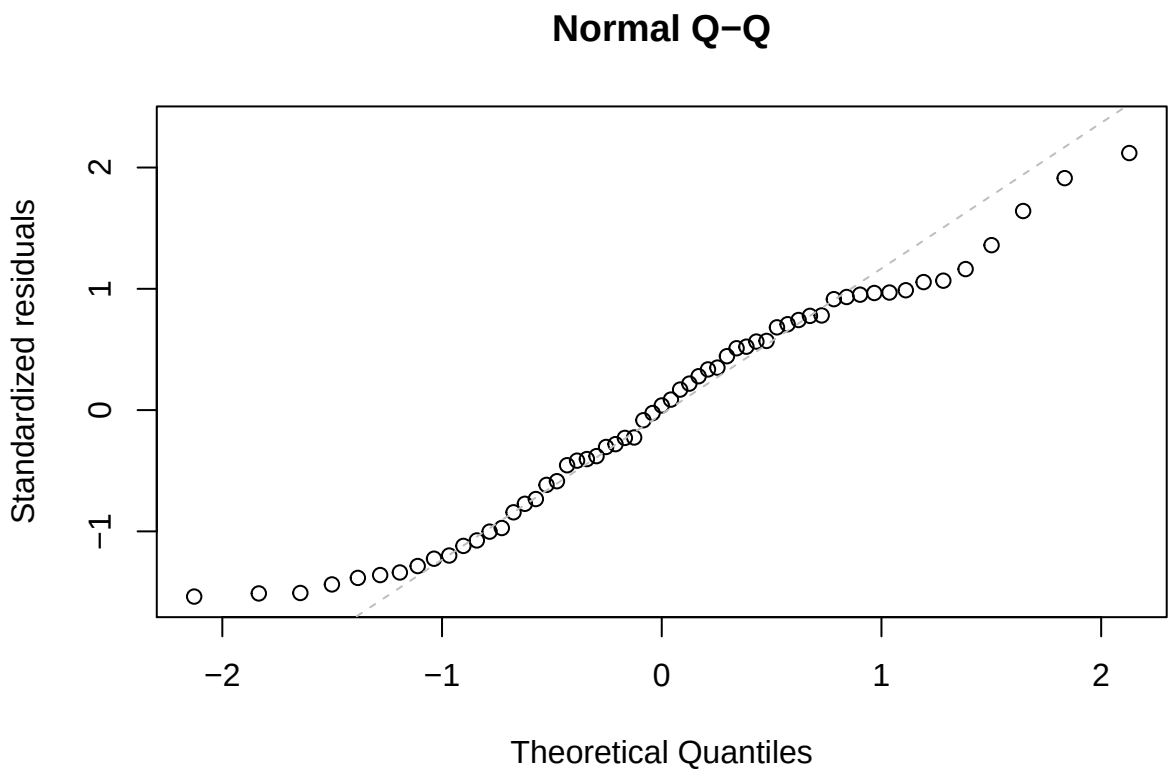
| TRUE |
|---|

**Part B**

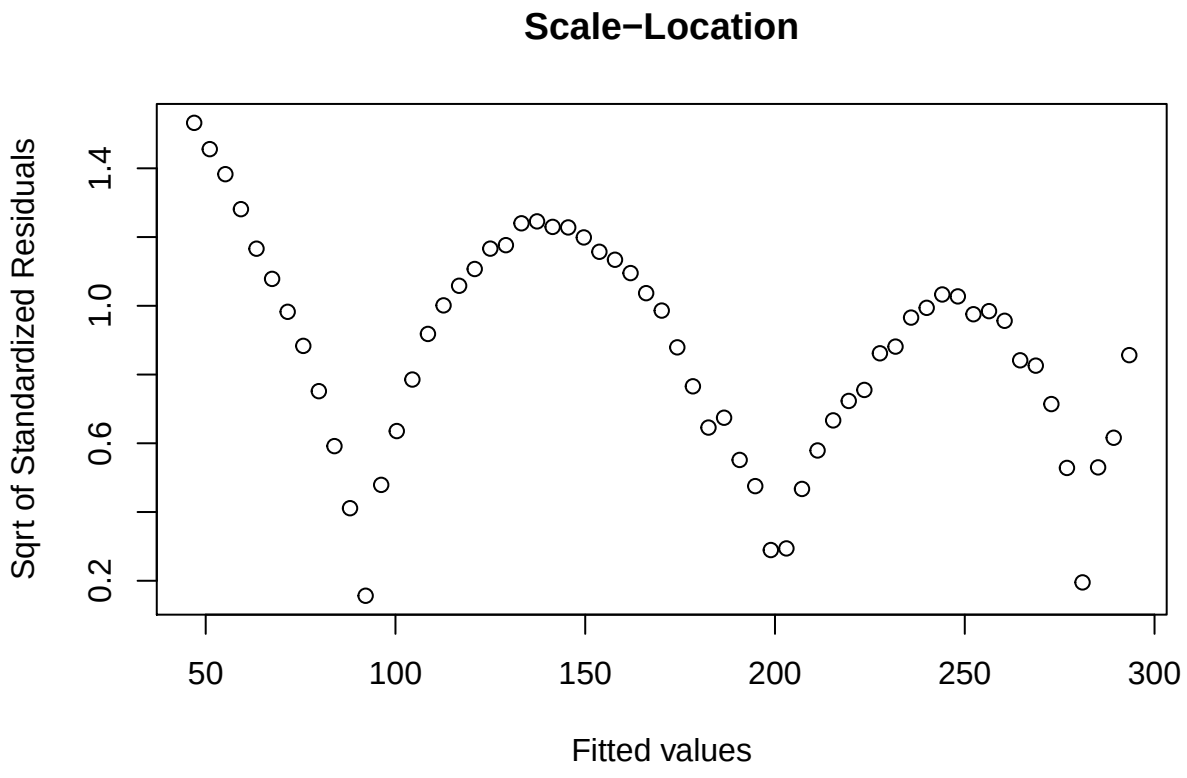
```
fit.diags <- broom::augment(fit)
# residuals vs fitted
plot(fit.diags$.fitted,fit.diags$.resid,xlab='Fitted values',
     ylab='Residuals',main='Residuals vs Fitted')
abline(0,0,col="gray",lty=2)
```



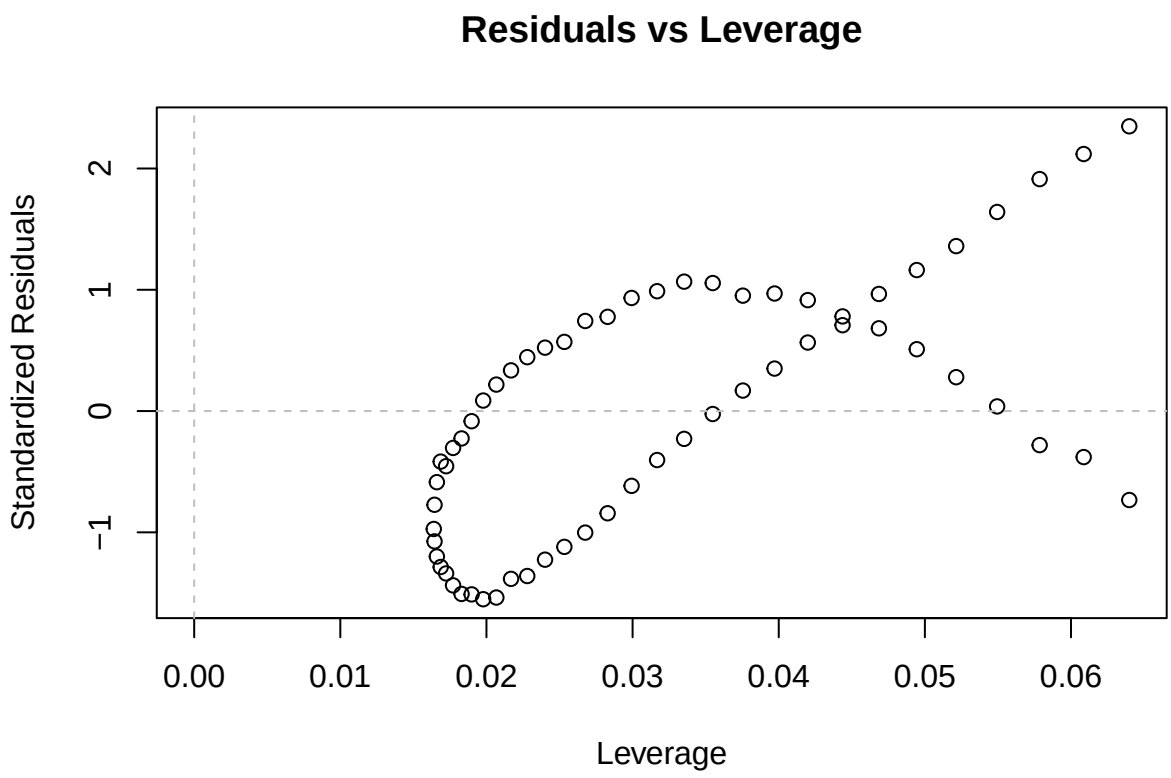

```
#plot(fit,1)

# Normal Q-Q
probs<-seq(0,1,length.out = length(fit.diags$.std.resid))
quantiles<-qnorm(probs,0,1)[order(order(fit.diags$.std.resid))]
     #order(order(x)) returns a rank order for the observations
plot(quantiles,fit.diags$.std.resid,xlab='Theoretical Quantiles',
     ylab='Standardized residuals',main='Normal Q-Q')
y<-quantile(fit.diags$.std.resid, c(0.25,0.75),names=FALSE, na.rm = TRUE)
x<-qnorm(c(0.25,0.75))
abline(y[1L] - (diff(y)/diff(x))*x[1L],diff(y)/diff(x),col="gray",lty=2)
```




```
#plot(fit,2)

# scale-location
plot(fit.diags$.fitted,sqrt(abs(fit.diags$.std.resid)),
     xlab='Fitted values',ylab='Sqrt of Standardized Residuals',main='Scale-Location')
```



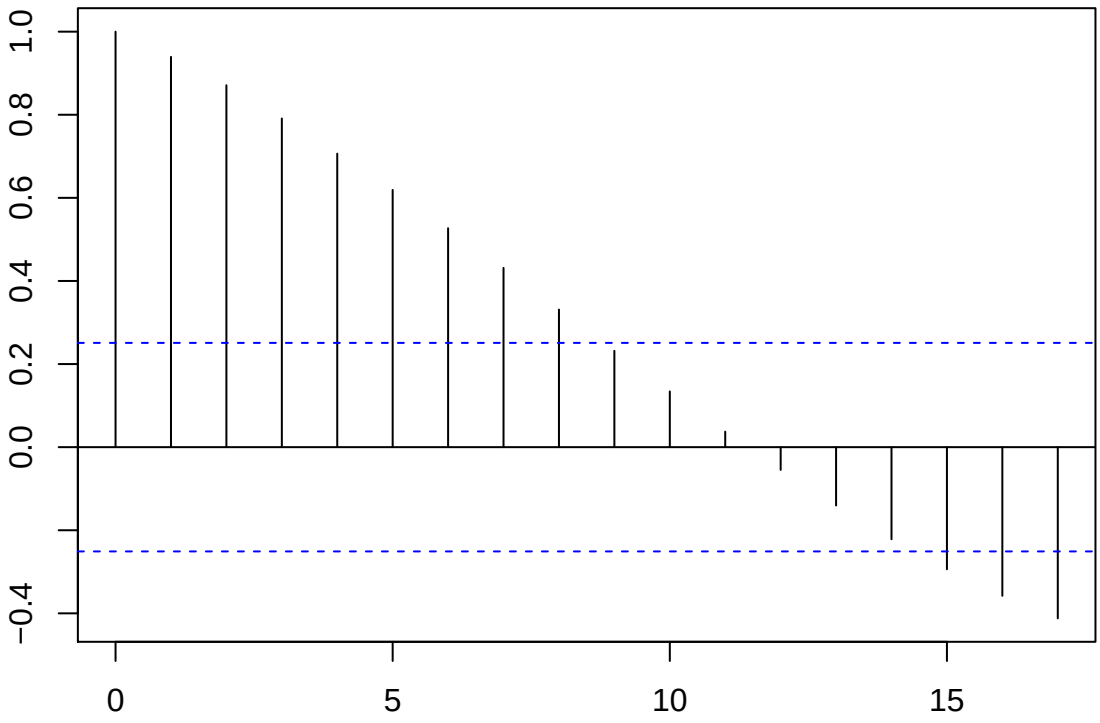
```
#plot(fit,3)

#residuals vs leverage
plot(fit.diags$.hat,fit.diags$.std.resid,xlim=c(0,max(fit.diags$.hat)),
     xlab='Leverage',ylab='Standardized Residuals',main='Residuals vs Leverage')
abline(h=0,col="gray",lty=2)
abline(v=0,col="gray",lty=2)
```




```
#plot(fit,5)
```

**Part C**

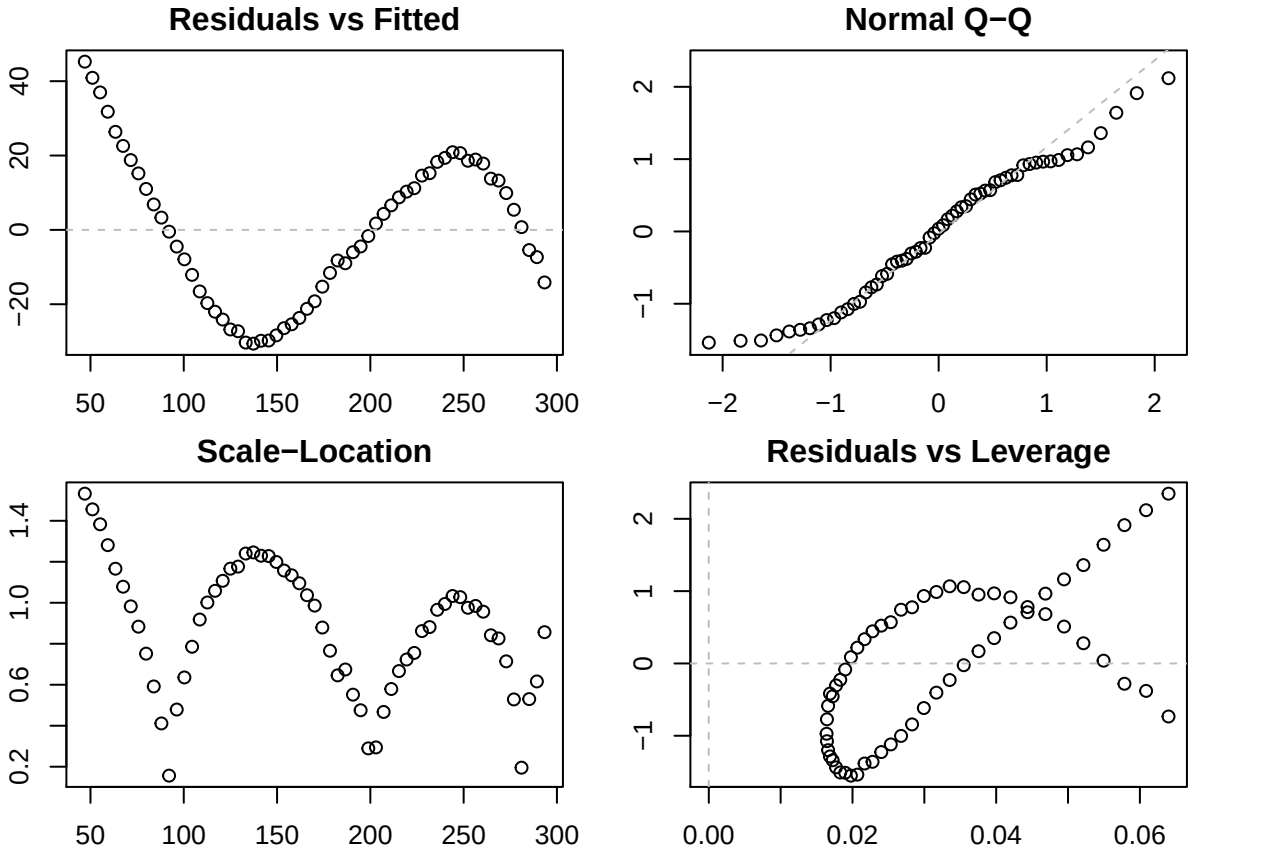
```
par(mfrow=c(1,1),mar=rep(3,4))
acf(fit.diags$.resid,type="correlation")
```




### Problem 5

```
par(mfrow=c(2,2),mar=rep(2,4))
# residuals vs fitted
plot(fit.diags$.fitted,fit.diags$.resid,xlab='Fitted values',
     ylab='Residuals',main='Residuals vs Fitted')
abline(0,0,col="gray",lty=2)
#plot(fit,1)

# Normal Q-Q
probs<-seq(0,1,length.out = length(fit.diags$.std.resid))
quantiles<-qnorm(probs,0,1)[order(order(fit.diags$.std.resid))]
#order(order(x)) returns a rank order for the observations
plot(quantiles,fit.diags$.std.resid,xlab='Theoretical Quantiles',
     ylab='Standardized residuals',main='Normal Q-Q')
y<-quantile(fit.diags$.std.resid, c(0.25,0.75),names=FALSE, na.rm = TRUE)
x<-qnorm(c(0.25,0.75))
abline(y[1L] - (diff(y)/diff(x))*x[1L],diff(y)/diff(x),col="gray",lty=2)
#plot(fit,2)

# scale-location
plot(fit.diags$.fitted,sqrt(abs(fit.diags$.std.resid)),
     xlab='Fitted values',ylab='Sqrt of Standardized Residuals',main='Scale-Location')
#plot(fit,3)

#residuals vs leverage
```

```
plot(fit.diags$.hat,fit.diags$.std.resid,xlim=c(0,max(fit.diags$.hat)),
     xlab='Leverage',ylab='Standardized Residuals',main='Residuals vs Leverage')
abline(h=0,col="gray",lty=2)
abline(v=0,col="gray",lty=2)
```




```
#plot(fit,5)
```