Lab 3

Junjie Zhang 517021911128

Task 1 (8 points)

(a) (1 point)

Succesfully render this file.

(b) (1 point)

```
chem_pro.df = read.table(file = chem_pro.csv, sep = ",", header = TRUE)
```

```
str(chem_pro.df)
```

```
## 'data.frame':    44 obs. of  4 variables:
##  $ yield     : num  55.5 54.8 52.2 50.4 49.3 ...
##  $ conversion: num  11.8 11.9 12.1 12.1 12 ...
##  $ flow      : num  119 105 97 101 44 ...
##  $ ratio     : Factor w/ 40 levels "0.036","0.089",..: 16 2 3 6 5 1 7 9 11 24 ...
```

It can be seen that there are three regressors, which are `yield`, `conversion`, `flow` and `ratio`, and one response, which is `yield`. And since the ratio is factor, there may be some issues.

Clean `ratio`

```
levels(chem_pro.df$ratio)
```

```
##  [1] "0.036" "0.089" "0.094" "0.097" "0.1"   "0.108" "0.113" "0.116"
##  [9] "0.123" "0.126" "0.135" "0.136" "0.143" "0.152" "0.153" "0.155"
## [17] "0.16"  "0.161" "0.164" "0.166" "0.169" "0.17"  "0.18"  "0.183"
## [25] "0.184" "0.188" "0.192" "0.194" "0.195" "0.197" "0.201" "0.211"
## [33] "0.215" "0.221" "0.222" "0.223" "0.225" "0.229" "0.233" "0>163"
```

It can be seen that the last element in ratio seems to be a typo.

```
ratio_typo = which (chem_pro.df$ratio=="0>163")
chem_pro.df$ratio = as.character(chem_pro.df$ratio)
chem_pro.df$ratio[ratio_typo] = "0.163"
chem_pro.df$ratio = as.double(chem_pro.df$ratio)
```

We then keep a record on the typo and then correct it. Then we take a close look at the data again.

```
kable(summary(chem_pro.df))
```

yield	conversion	flow	ratio
Min. :40.05	Min. :-11.12	Min. : 30.0	Min. :0.0360
1st Qu.:48.05	1st Qu.: 11.19	1st Qu.:221.5	1st Qu.:0.1358
Median :51.28	Median : 11.89	Median :325.0	Median :0.1675
Mean :50.68	Mean : 11.11	Mean :291.9	Mean :0.1658
3rd Qu.:53.91	3rd Qu.: 12.04	3rd Qu.:380.0	3rd Qu.:0.1980

yield	conversion	flow	ratio
Max. :59.34	Max. : 12.36	Max. :523.0	Max. :0.2330

It seems like that the *min* of `conversion` is far from it's 1*st Qu*, there may be some issues. And the flow seems to be ok.

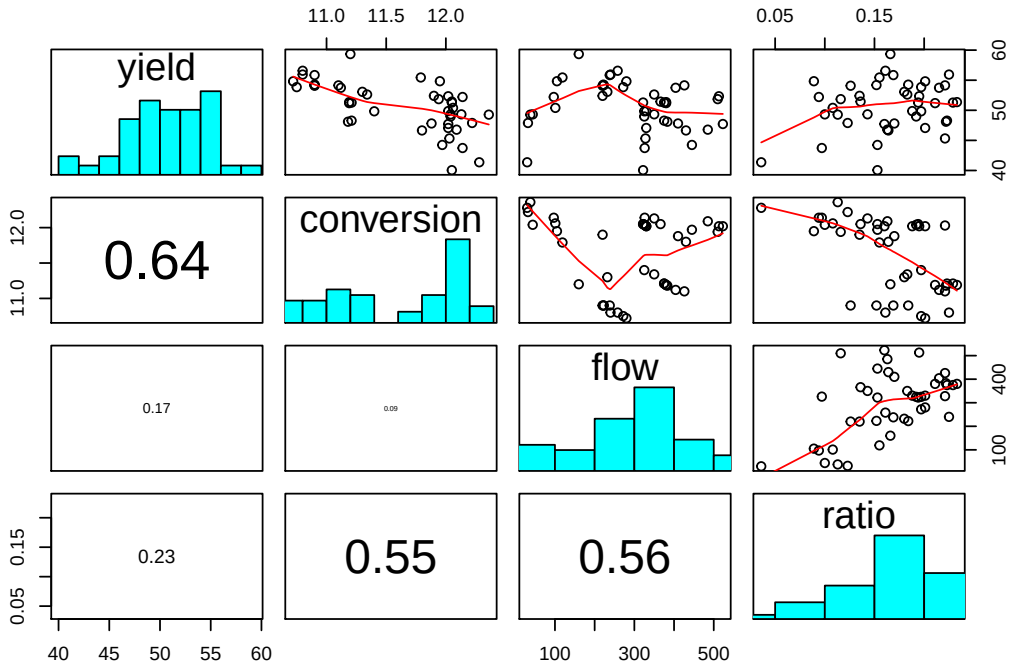
Clean `conversion`

```
conversion_typo = which(chem_pro.df$conversion<=-10)
chem_pro.df$conversion[conversion_typo] = -chem_pro.df$conversion[conversion_typo]
```

We keep record of the possible typo in `conversion` and correct it.

(c) (1 point)

```
panel.hist <- function(x, ...)
{
    usr <- par("usr"); on.exit(par(usr))
    par(usr = c(usr[1:2], 0, 1.5) )
    h <- hist(x, plot = FALSE)
    breaks <- h$breaks; nB <- length(breaks)
    y <- h$counts; y <- y/max(y)
    rect(breaks[-nB], 0, breaks[-1], y, col = "cyan", ...)
}
panel.cor <- function(x, y, digits = 2, prefix = "", cex.cor, ...)
{
    usr <- par("usr"); on.exit(par(usr))
    par(usr = c(0, 1, 0, 1))
    r <- abs(cor(x, y))
    txt <- format(c(r, 0.123456789), digits = digits)[1]
    txt <- paste0(prefix, txt)
    if(missing(cex.cor)) cex.cor <- 0.8/strwidth(txt)
    text(0.5, 0.5, txt, cex = cex.cor * r)
}
pairs(chem_pro.df[1:4],upper.panel = panel.smooth,lower.panel = panel.cor,
      diag.panel = panel.hist)
```

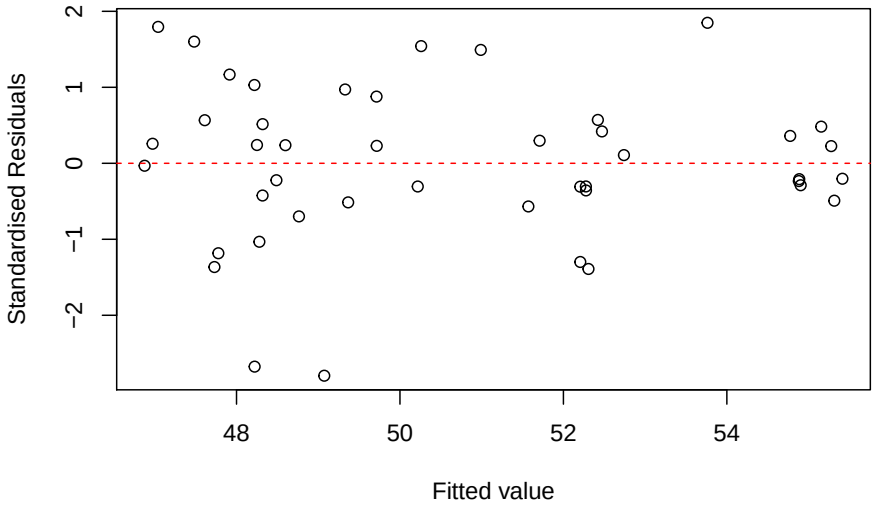


(d) (1 point)

```
chem_pro.LM = lm(yield~conversion+flow+ratio, data = chem_pro.df)
```

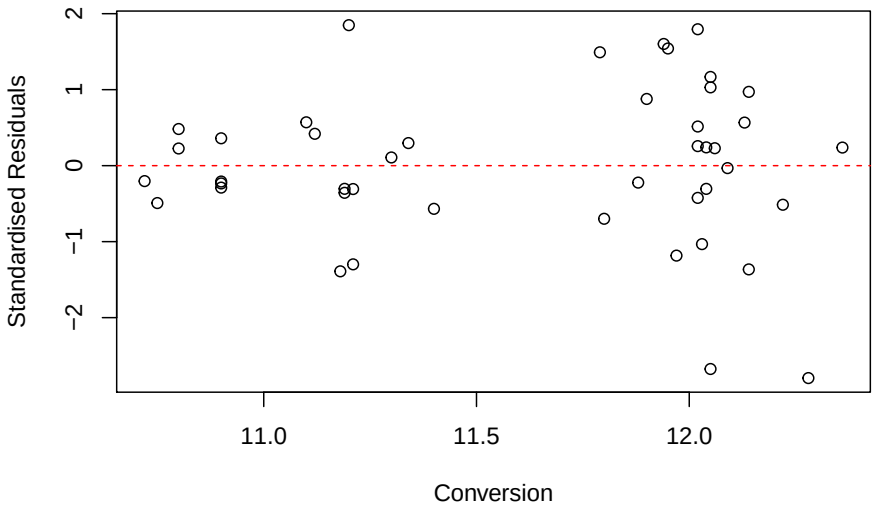
- Standardised residual Vs fitted value

```
plot(fitted.values(chem_pro.LM),rstandard(chem_pro.LM),
     xlab = "Fitted value",ylab = "Standardised Residuals")
abline(a = 0, b = 0, lty = 2, col = "red")
```



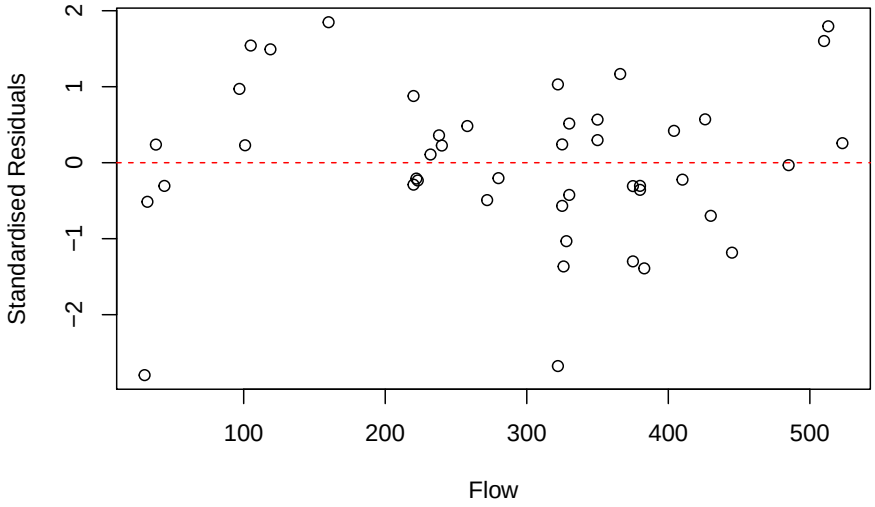
- Standardised residual Vs conversion

```
plot(chem_pro.df$conversion,rstandard(chem_pro.LM),
     xlab = "Conversion",ylab = "Standardised Residuals")
abline(a = 0, b = 0, lty = 2, col = "red")
```



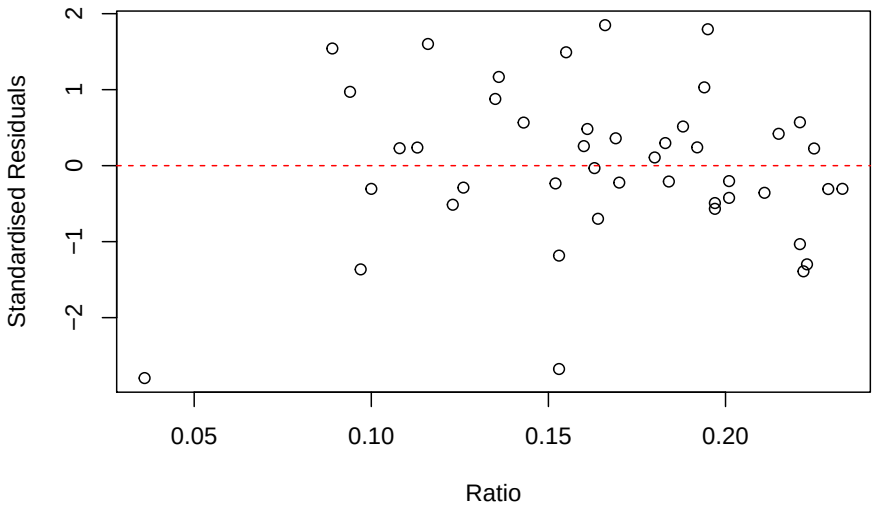
- Standardised residual Vs flow

```
plot(chem_pro.df$flow,rstandard(chem_pro.LM), xlab = "Flow",ylab = "Standardised Residuals")
abline(a = 0, b = 0, lty = 2, col = "red")
```



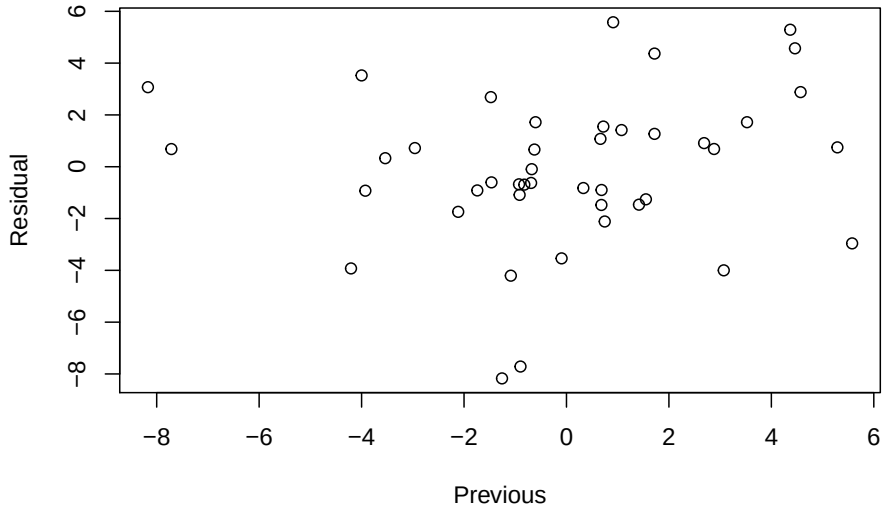
- Standardised residual Vs ratio

```
plot(chem_pro.df$ratio,rstandard(chem_pro.LM),
     xlab = "Ratio",ylab = "Standardised Residuals")
abline(a = 0, b = 0, lty = 2, col = "red")
```



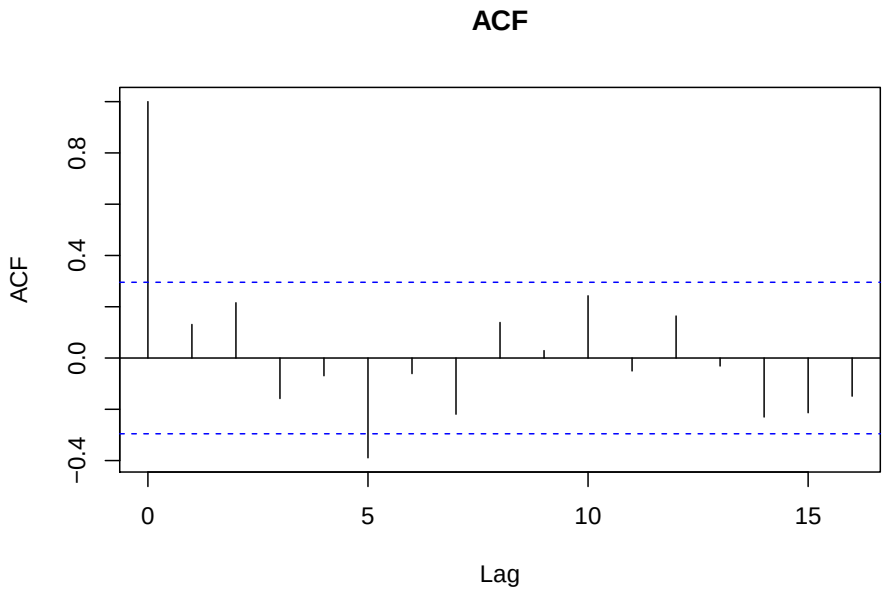
- Residual Vs Previous Residual

```
plot(chem_pro.LM$residuals[-length(chem_pro.LM$residuals)],
     chem_pro.LM$residuals[-1],
     xlab = "Previous", ylab = "Residual")
```



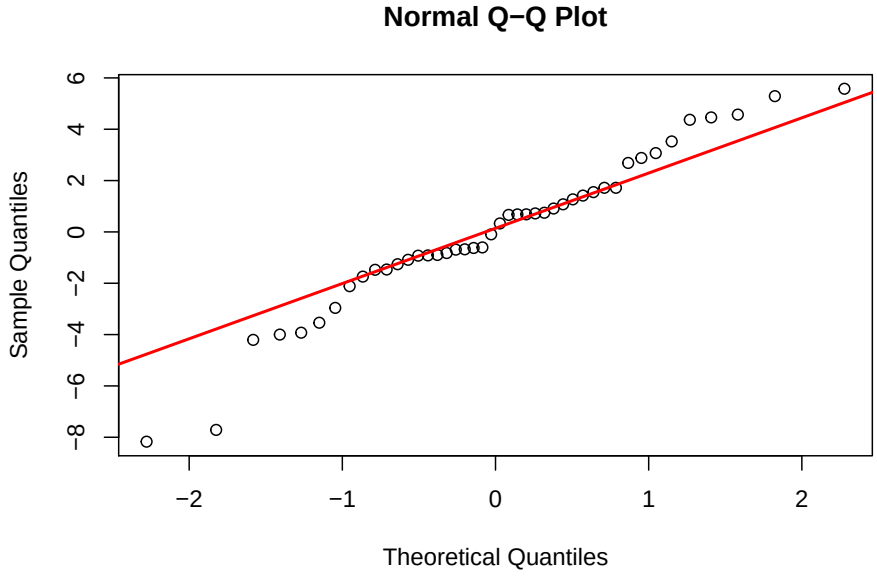
- Residual Autcorrelation (ACF)

```
acf(chem_pro.LM$residuals, main = "ACF")
```



- Q-Q Normal

```
qqnorm(chem_pro.LM$residuals)
qqline(chem_pro.LM$residuals,lwd=2,col=2)
```



(e) (1 point)

```
correlation.matrix = cor(chem_pro.df[-1])
temp = solve(correlation.matrix)
VIF.vec = c(temp[1,1],temp[2,2],temp[3,3])
names(VIF.vec) = c("conversion", "flow", "ratio")
kable(VIF.vec,col.names = "VIF")
```

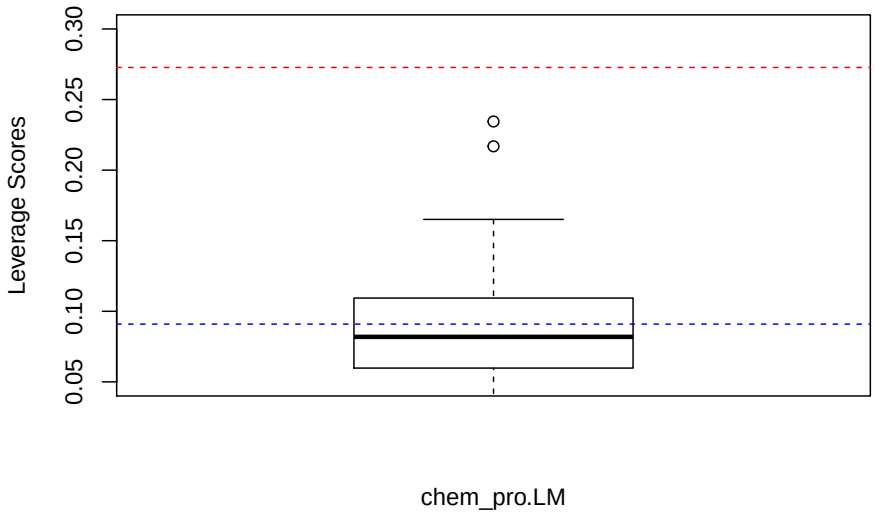
	VIF
conversion	1.580323
flow	1.606860

	VIF
ratio	2.276409

It is the same as what we found in the class.

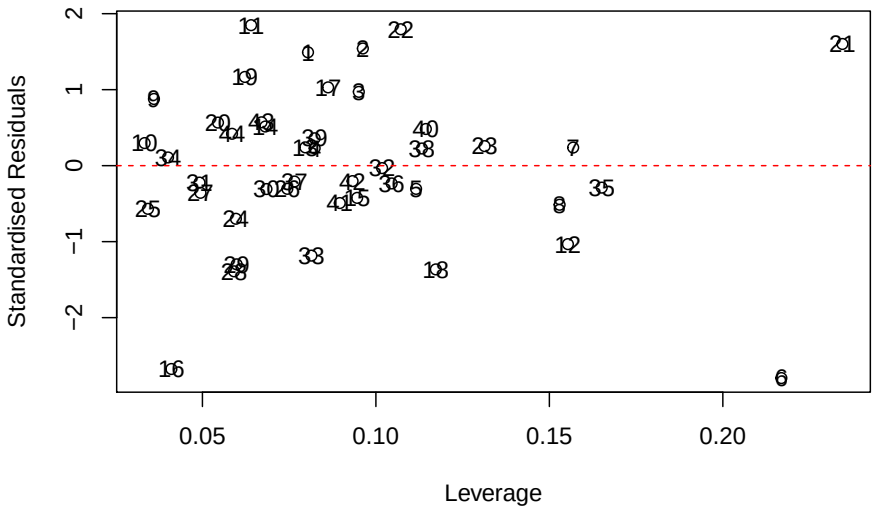
(f) (1 point)

```
pii.vec = hatvalues(chem_pro.LM)
boxplot(pii.vec, xlab = "chem_pro.LM", ylab = "Leverage Scores",
        ylim = c(0.05, 0.3))
excessive <- 3 * (3 + 1) / 44
abline(a = mean(pii.vec), b = 0, lty = 2, col = "blue")
abline(a = excessive, b = 0, lty = 2, col = "red")
```



(g) (1 point)

```
plot(hatvalues(chem_pro.LM),rstandard(chem_pro.LM), xlab = "Leverage",
     ylab = "Standardised Residuals")
text(x=hatvalues(chem_pro.LM),y=rstandard(chem_pro.LM),labels = 1:44,font=1)
abline(a=0,b=0,lty=2,col="red")
```



(h) (1 point)

```
im = influence.measures(chem_pro.LM)
im
```

```
## Influence measures of
##   lm(formula = yield ~ conversion + flow + ratio, data = chem_pro.df) :
##
##      dfb.1_  dfb.cnvr dfb.flow dfb.rati   dffit cov.r   cook.d    hat inf
## 1  -0.15379  0.171205 -0.37075  0.21414  0.4483 0.957 4.87e-02 0.0804
## 2   0.10348 -0.044898 -0.11423 -0.24833  0.5123 0.958 6.33e-02 0.0962
## 3  -0.02789  0.062869 -0.12017 -0.07752  0.3146 1.111 2.48e-02 0.0951
## 4  -0.01039  0.017236 -0.03453 -0.00626  0.0678 1.200 1.18e-03 0.0823
## 5   0.01094 -0.021314  0.06432  0.00689 -0.1070 1.234 2.93e-03 0.1116
## 6  -0.31180  0.115894  0.17695  0.97151 -1.6193 0.592 5.41e-01 0.2169   *
## 7  -0.05076  0.058095 -0.07330  0.03086  0.1015 1.305 2.64e-03 0.1569   *
## 8   0.10158 -0.115249  0.17135 -0.08036 -0.2165 1.272 1.19e-02 0.1530
## 9  -0.02284  0.038443 -0.03617 -0.02638  0.1692 1.062 7.20e-03 0.0360
## 10  0.02164 -0.022460  0.01842 -0.00746  0.0546 1.135 7.61e-04 0.0333
## 11  0.19537 -0.185560 -0.28038  0.04779  0.4993 0.827 5.85e-02 0.0640
## 12  0.35865 -0.342510  0.18667 -0.38167 -0.4433 1.176 4.90e-02 0.1553
## 13 -0.05581  0.054864 -0.02031  0.04898  0.0699 1.195 1.25e-03 0.0797
## 14 -0.10685  0.105545 -0.03182  0.08916  0.1380 1.156 4.85e-03 0.0681
## 15  0.10868 -0.105522  0.04425 -0.10339 -0.1357 1.200 4.70e-03 0.0946
## 16  0.29033 -0.321183 -0.09788 -0.01622 -0.6034 0.524 7.66e-02 0.0411   *
## 17 -0.25558  0.250764 -0.10094  0.22932  0.3167 1.087 2.50e-02 0.0862
## 18 -0.09085  0.045557 -0.29539  0.38361 -0.5032 1.036 6.19e-02 0.1173
## 19 -0.02490  0.044340  0.17963 -0.14403  0.3021 1.027 2.26e-02 0.0622
## 20 -0.04657  0.054612  0.05471 -0.02995  0.1349 1.133 4.63e-03 0.0545
## 21  0.28057 -0.240441  0.80109 -0.69167  0.9045 1.109 1.96e-01 0.2345
## 22 -0.28662  0.273128  0.35287  0.08608  0.6404 0.886 9.67e-02 0.1072
## 23 -0.00630  0.007837  0.08246 -0.03858  0.0987 1.265 2.50e-03 0.1313
## 24 -0.00246 -0.000810 -0.13055  0.06366 -0.1750 1.120 7.75e-03 0.0597
## 25  0.00460 -0.000362  0.01388 -0.04641 -0.1062 1.109 2.87e-03 0.0343
## 26  0.00934 -0.002575  0.01017 -0.05271 -0.0855 1.185 1.87e-03 0.0745
## 27 -0.01633  0.021263 -0.01283 -0.02067 -0.0806 1.149 1.66e-03 0.0494
## 28 -0.01828  0.043894 -0.01005 -0.15639 -0.3528 0.965 3.04e-02 0.0591
```

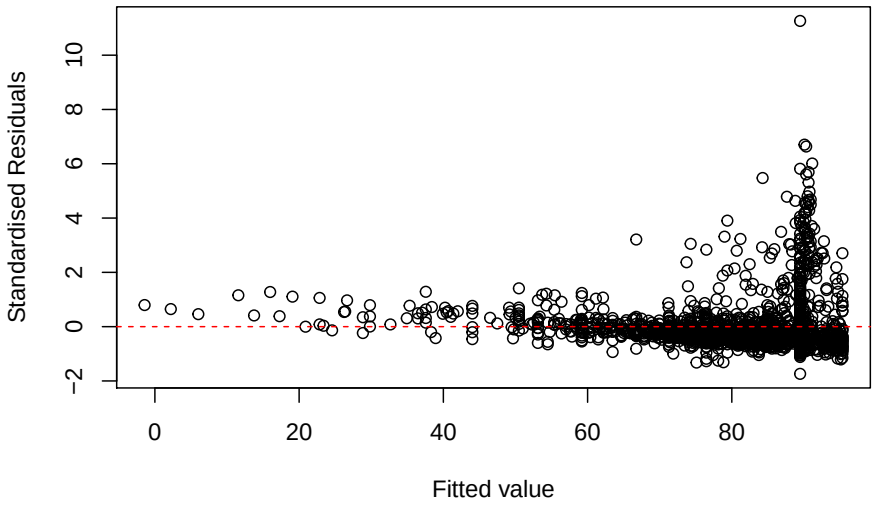
```
## 29  0.00710  0.017014  0.01493 -0.17090 -0.3307 0.991 2.69e-02 0.0599
## 30  0.00809 -0.001782  0.00932 -0.04943 -0.0825 1.177 1.74e-03 0.0685
## 31  0.01242 -0.013054 -0.02847  0.00527 -0.0501 1.158 6.43e-04 0.0492
## 32  0.00253 -0.002703 -0.00775  0.00261 -0.0107 1.232 2.94e-05 0.1018
## 33  0.01843 -0.029428 -0.27092  0.14880 -0.3542 1.045 3.11e-02 0.0814
## 34  0.00500 -0.005066 -0.01017  0.00589  0.0218 1.151 1.21e-04 0.0401
## 35 -0.11592  0.110300 -0.03173  0.09279 -0.1270 1.314 4.13e-03 0.1651   *
## 36 -0.06936  0.067273 -0.00626  0.04110 -0.0790 1.229 1.60e-03 0.1045
## 37 -0.03747  0.038115  0.01600  0.00120 -0.0594 1.193 9.04e-04 0.0765
## 38  0.01881 -0.023149 -0.03880  0.03878  0.0797 1.242 1.63e-03 0.1133
## 39  0.08636 -0.085527 -0.00198 -0.03544  0.1066 1.190 2.91e-03 0.0823
## 40  0.15475 -0.152393  0.03017 -0.08905  0.1717 1.221 7.51e-03 0.1144
## 41 -0.10567  0.110095  0.01319  0.00863 -0.1529 1.186 5.96e-03 0.0896
## 42 -0.04383  0.046000  0.00491  0.00218 -0.0645 1.215 1.06e-03 0.0933
## 43  0.03744 -0.047960  0.04730  0.02774  0.1516 1.148 5.85e-03 0.0670
## 44  0.02885 -0.035479  0.02799  0.01741  0.1034 1.155 2.73e-03 0.0585
```

Task 2 (6 points)

(a) (1 point)

```
usare.df = read.table(file = USA_real_estate.txt, sep = "", header = TRUE)
usare.LM = lm(mppsf~pnh+pms, data = usare.df)
```

```
usare.df = read.table(file = USA_real_estate.txt, sep = "", header = TRUE)
usare.LM = lm(mppsf~pnh+pms, data = usare.df)
plot(fitted.values(usare.LM),rstandard(usare.LM), xlab = "Fitted value",
     ylab = "Standardised Residuals")
abline(a = 0, b = 0, lty = 2, col = "red")
```



It can be seen that as the `fitted.value` increases, the standardised residual varies in a larger range. Thus, there is a heteroskedasticity problem.

(b) (1 point)

```
z = 2 * log(abs(usare.LM$residuals))
auxiliary.LM = lm(z ~ pnh + pms, data = usare.df)
```

```
w.vec = 1 / exp(auxiliary.LM$fitted.values)
```

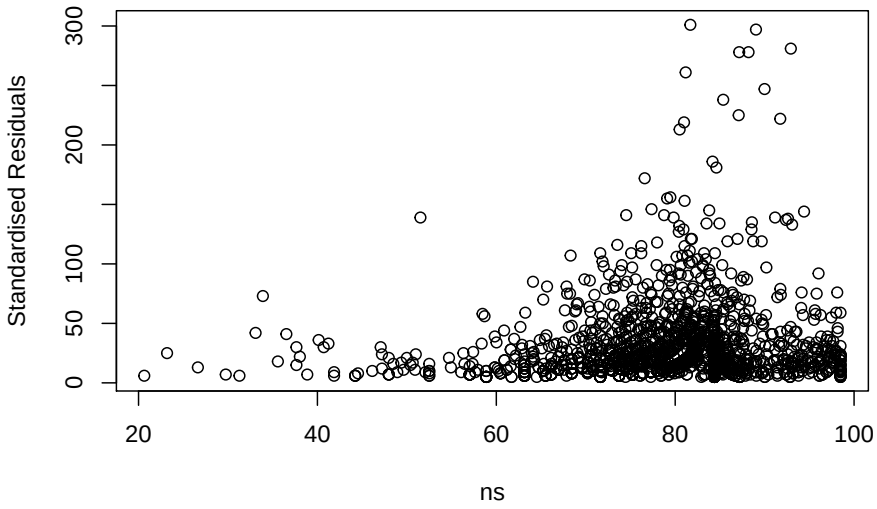
(c) (1 point)

```
usare.WLS = lm(mppsf~pnh+pms, weights = w.vec, data = usare.df)
```

(d) (1 point)

Since `ns` is one of the original data in the calculation of 'mppsf', it may affect the orignal ϵ. Then, we may choose it as a weight.

```
plot(fitted.values(usare.WLS),usare.df$ns, xlab = "ns",
     ylab = "Standardised Residuals")
```



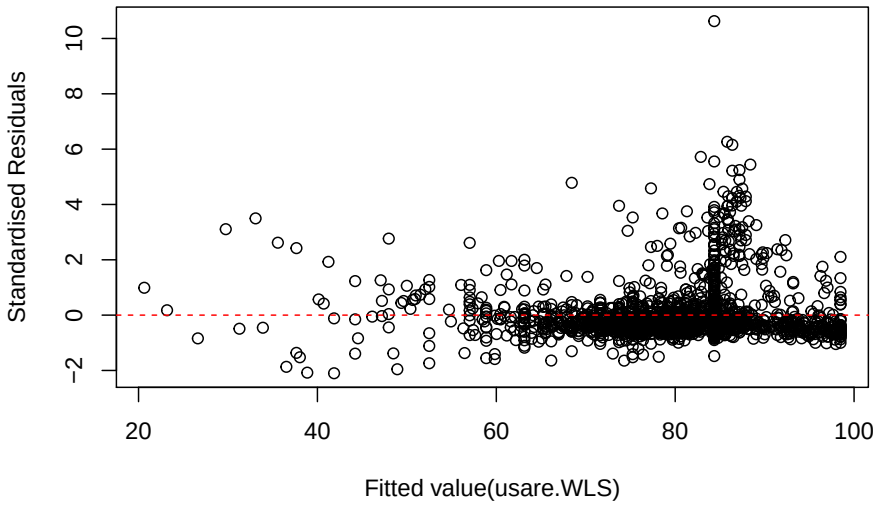
Moreover, it can be seen that the residual increase as `ns` increase, which indicates the heteroskedasticity may be solved by choosing `ns` as a weight.

(e) (1 point)

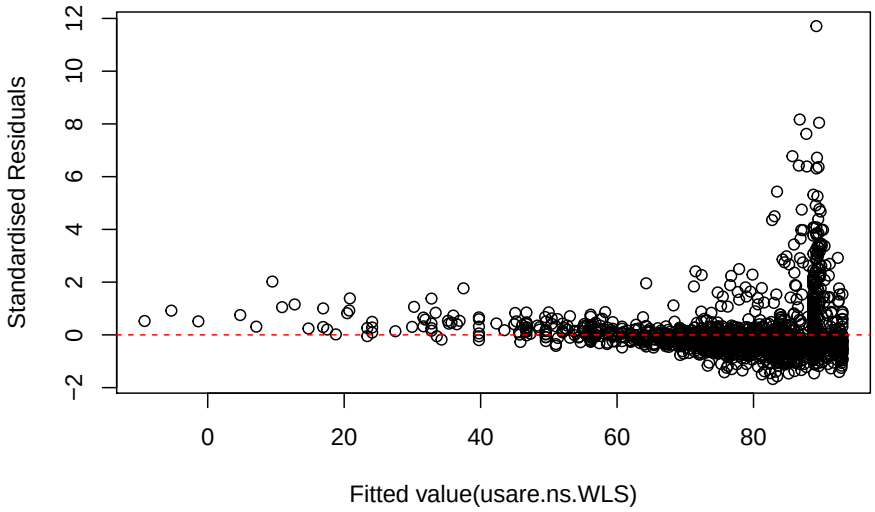
```
usare.ns.WLS = lm(mppsf~pnh+pms, weights = ns, data = usare.df)
```

(f) (1 point)

```
plot(fitted.values(usare.WLS),rstandard(usare.WLS), xlab = "Fitted value(usare.WLS)",
     ylab = "Standardised Residuals")
abline(a = 0, b = 0, lty = 2, col = "red")
```




```
plot(fitted.values(usare.ns.WLS),rstandard(usare.ns.WLS), xlab = "Fitted value(usare.ns.WLS)",
     ylab = "Standardised Residuals")
abline(a = 0, b = 0, lty = 2, col = "red")
```



Based on the plots, we find the one of `usare.WLS` model has a better pattern. Thus, the `usare.WLS` model is better.

Task 3 (5 points)

The data `grossboxoffice` is about yearly gross box office receipts from moives screened in Australia.

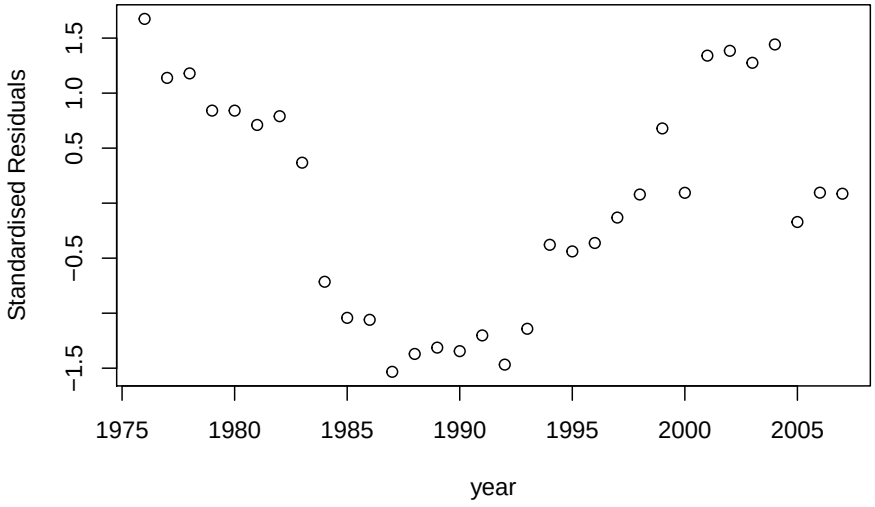
(a) (1 point)

```
gbo.df = read.table("grossboxoffice.txt", header = T)
gbo.LM = lm(GrossBoxOffice ~ year, data = gbo.df)
summary(gbo.LM)
```

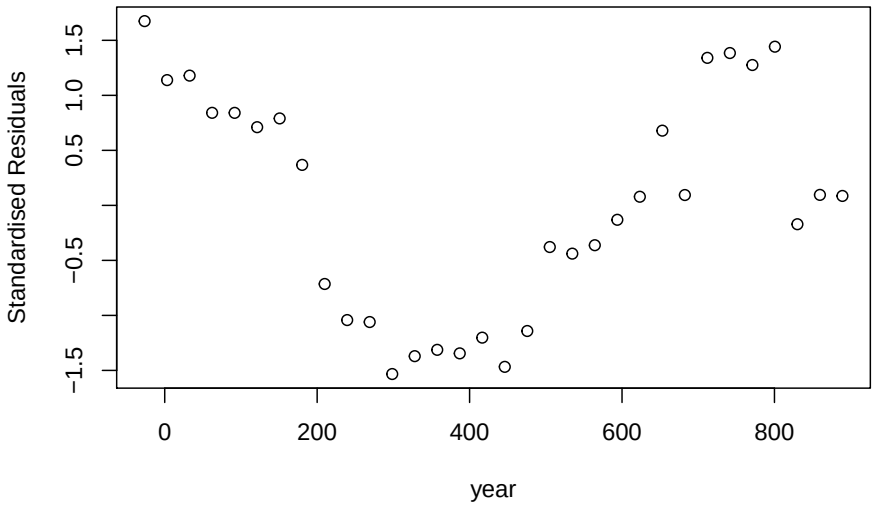
```
##
## Call:
## lm(formula = GrossBoxOffice ~ year, data = gbo.df)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -116.382  -79.197    6.083   62.260  121.697
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -58386.485   2952.825  -19.77   <2e-16 ***
## year            29.534      1.483   19.92   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 77.44 on 30 degrees of freedom
## Multiple R-squared:  0.9297, Adjusted R-squared:  0.9274
## F-statistic: 396.8 on 1 and 30 DF,  p-value: < 2.2e-16
```

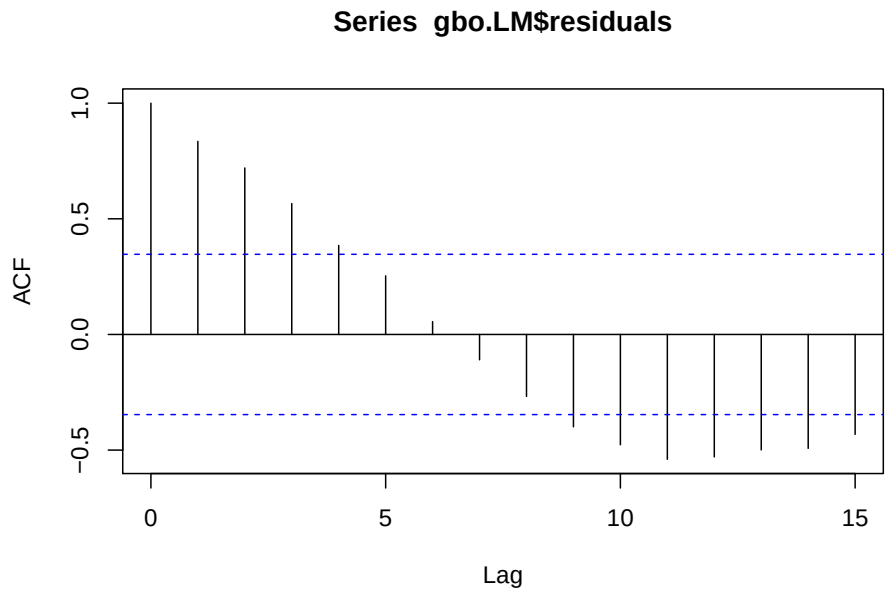
```
plot(gbo.df$year,rstandard(gbo.LM),xlab = "year",
     ylab = "Standardised Residuals")
```



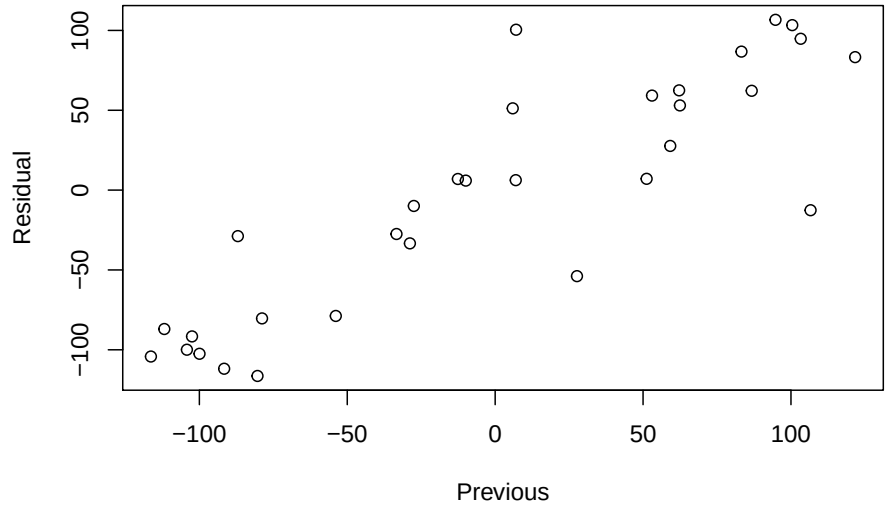

```
plot(fitted.values(gbo.LM),rstandard(gbo.LM),xlab = "year",
     ylab = "Standardised Residuals")
```



```r
acf(gbo.LM$residuals)
```




```r
plot(gbo.LM$residuals[-length(gbo.LM$residuals)],
     gbo.LM$residuals[-1],
     xlab = "Previous", ylab = "Residual")
```



Based on the acf plot and the plot of previous residual vs residual, the correlation between the residuals can be detected. Moreover, the pattern of residuals is unusual. Thus, `gbo.LM` is invalid.

(b) (1 point)

AR(1)

```r
yt = gbo.df$GrossBoxOffice
lag.df = data.frame(x1 = yt[-c(1,32,31)], x2 = yt[-c(1,2,32)], x3 = yt[-c(1,2,3)],
                    y = yt[-(32:30)])
cor(lag.df$x1,lag.df$y)
```

```
## [1] 0.9892474
```

Since the correlation is high, the AR(1) model is possible.

AR(2)

```
cor(lag.df$x2,lag.df$y)
```

```
## [1] 0.9817914
```

Since the correlation is high, the AR(2) model is possible.

AR(3)

```
cor(lag.df$x3,lag.df$y)
```

```
## [1] 0.9718559
```

Since the correlation is not that high, AR(3) model may be possible.

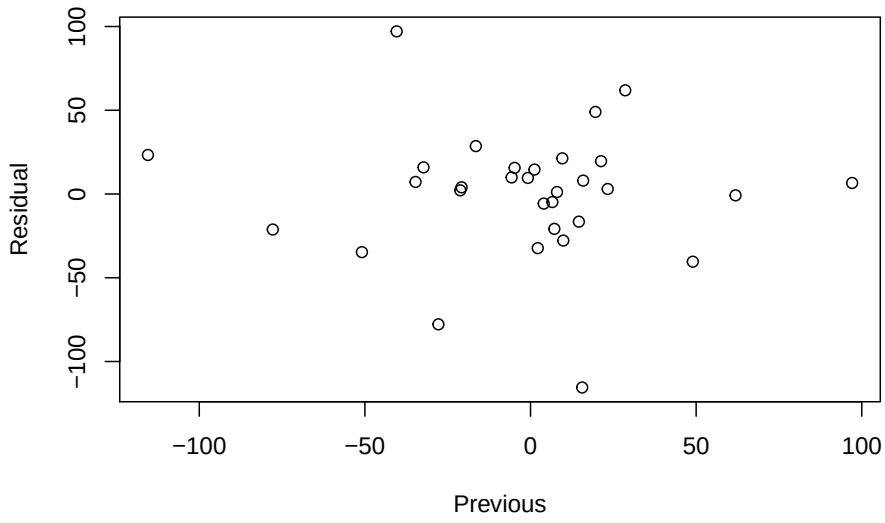
(c) (1 point)

```
gbo.final.M = arima(gbo.df$GrossBoxOffice, order = c(0,1,0),
xreg =  gbo.df$year)
pre_1975 = predict(gbo.final.M,newxreg=1975)
pre_1975
```

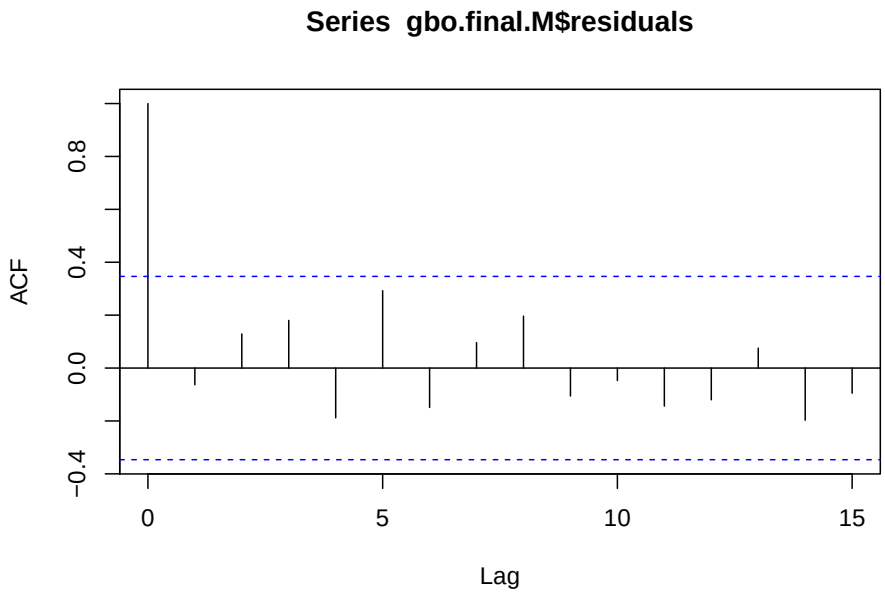
```
## $pred
## Time Series:
## Start = 33
## End = 33
## Frequency = 1
## [1] 69.49032
##
## $se
## Time Series:
## Start = 33
## End = 33
## Frequency = 1
## [1] 37.76236
```

(d) (1 point)

```
plot(gbo.final.M$residuals[-length(gbo.final.M$residuals)],
    gbo.final.M$residuals[-1],
     xlab = "Previous", ylab = "Residual")
```




```
acf(gbo.final.M$residuals)
```



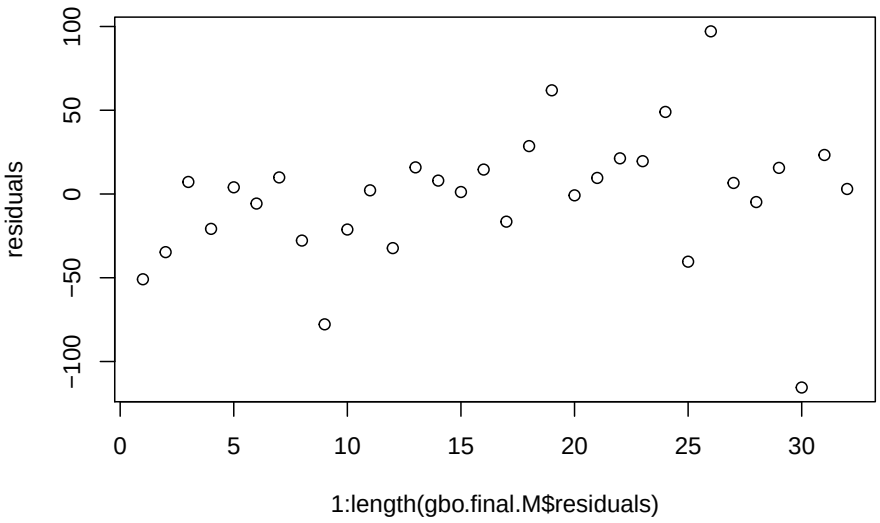
Based on the plot, it can be seen the residuals are independent with each other, which means the model is valid.

(e) (1 point)

Based on the residual plot in the linear model part, it seems that the model should include the high order term, but in the AR model we do not include. Although the residual looks ok, it may be some problems.

(f) (1 point)

```
plot(1:length(gbo.final.M$residuals),gbo.final.M$residuals,ylab = "residuals")
```



Outliers are points with index 26, 30.