Time Series Models
Econometrics
Sai Sandeep Chandaluri

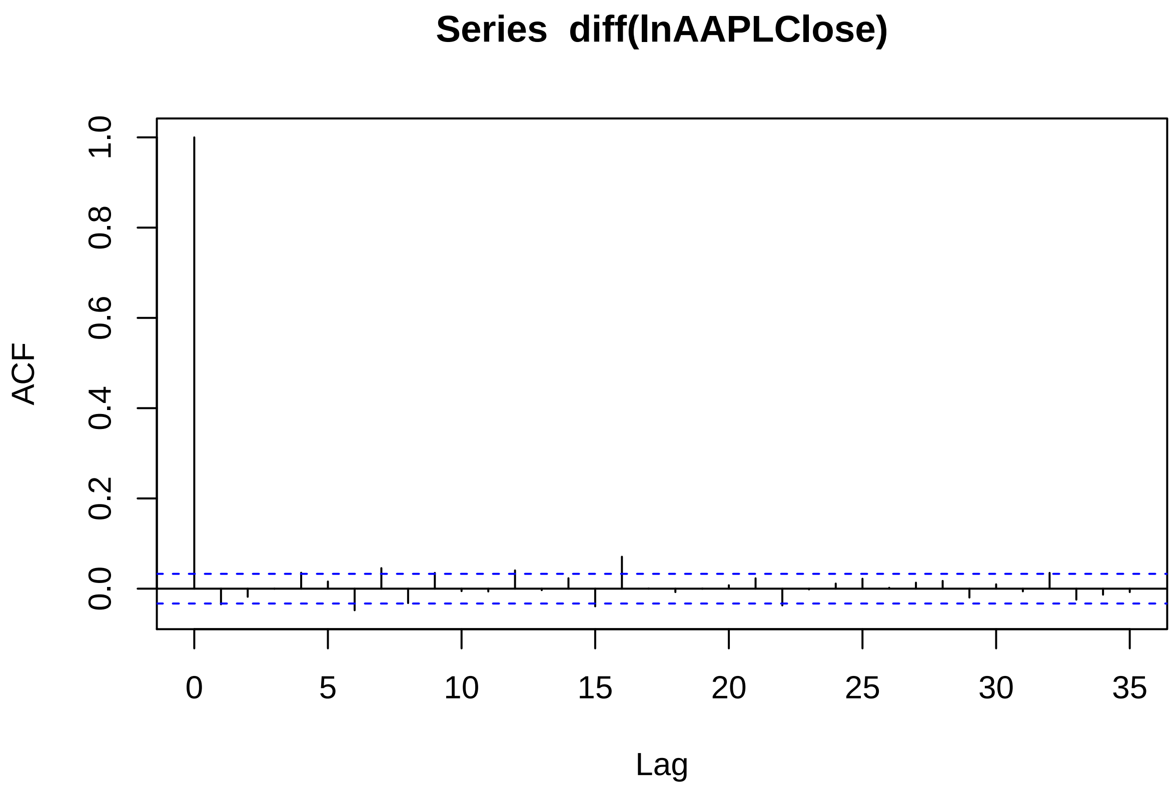
Question 1

Apple stock prices are retrieved and autocorrelations of difference log prices are plotted.

```
library(quantmod)
getSymbols("AAPL")
```

```
## [1] "AAPL"
```

```
lnAAPLClose = log(AAPL[,4])

acf(diff(lnAAPLClose), na.action = na.pass)
```



Question 2

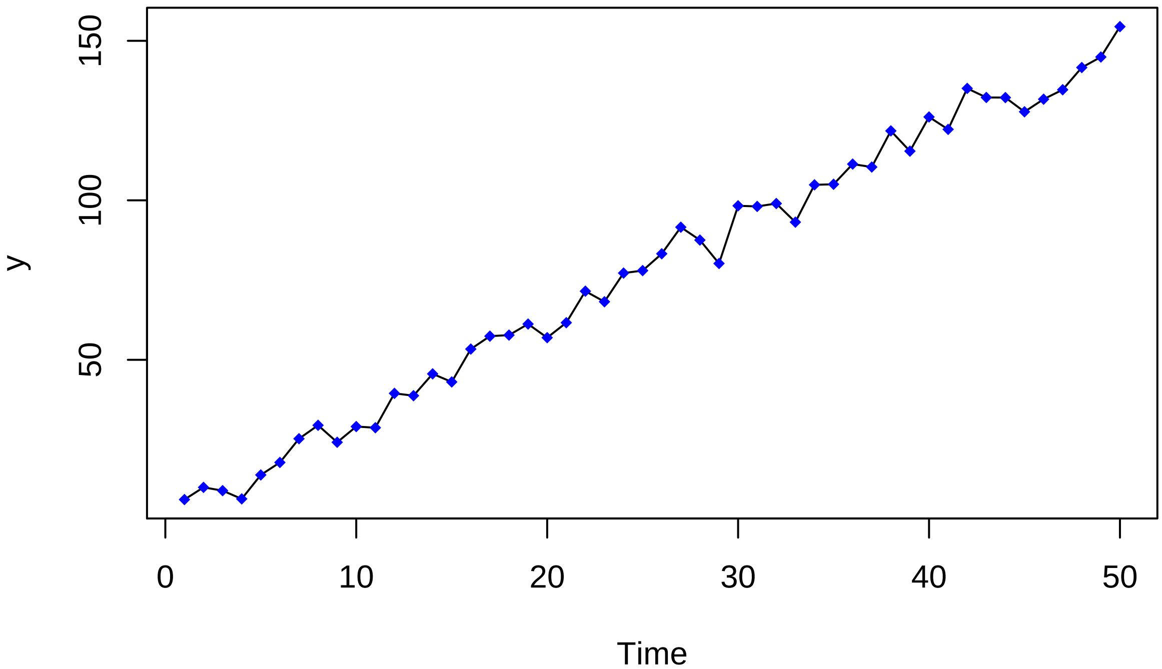
a. A linear time trend: $y_t = \alpha + \beta t + \varepsilon_t$

```
t=1:50
y = 2 + 3*t + rnorm(50, sd = 4)

summary(lm(y~t))
```

```
##
## Call:
## lm(formula = y ~ t)
##
## Residuals:
```

```
##     Min      1Q Median     3Q    Max
## -8.146 -2.870  1.426  3.421  9.243
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.96326    1.28157   1.532    0.132
## t            2.97649    0.04374  68.051   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.463 on 48 degrees of freedom
## Multiple R-squared:  0.9897, Adjusted R-squared:  0.9895
## F-statistic:  4631 on 1 and 48 DF,  p-value: < 2.2e-16
```

```r
plot(y, type="l", xlab="Time"); points(y, pch=18, cex = 0.8, col = "blue")
```



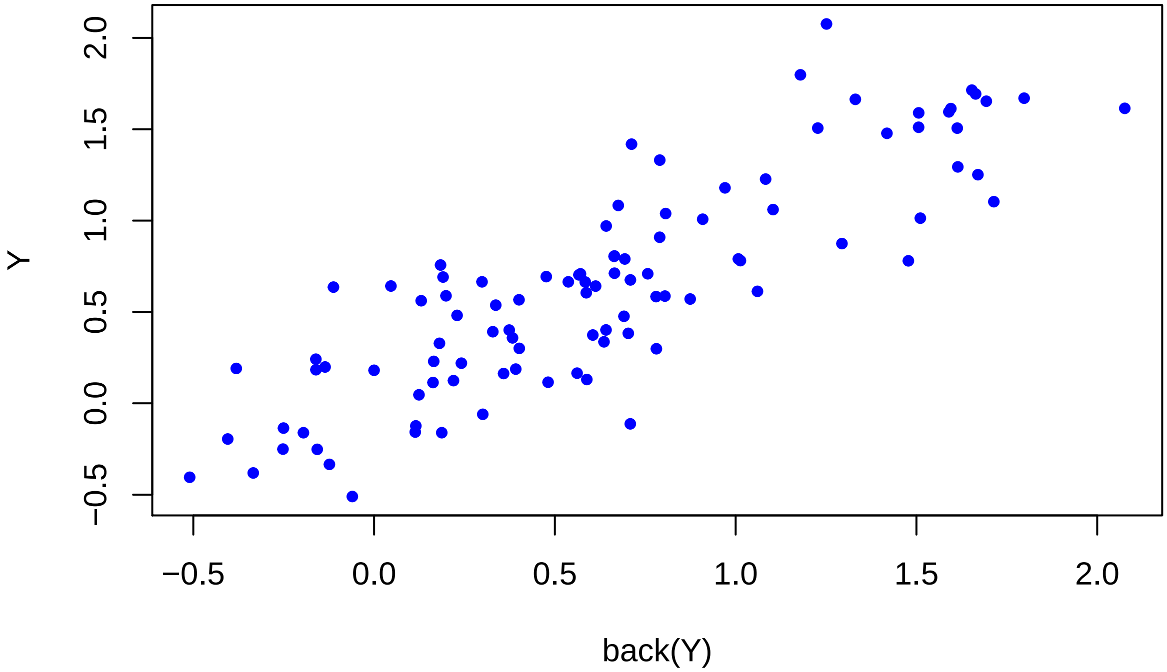
b. An AR(1): $y_t = \alpha + \beta y_{t-1} + \varepsilon_t$

```r
library(DataAnalytics)
Y = double(100)
Y[1] = 0
for (i in seq(1,100,1)){
Y[i+1] = 0.1 + 0.8*Y[i] + rnorm(1, sd = 0.3)
}
lmSumm(lm(Y~back(Y)))
```

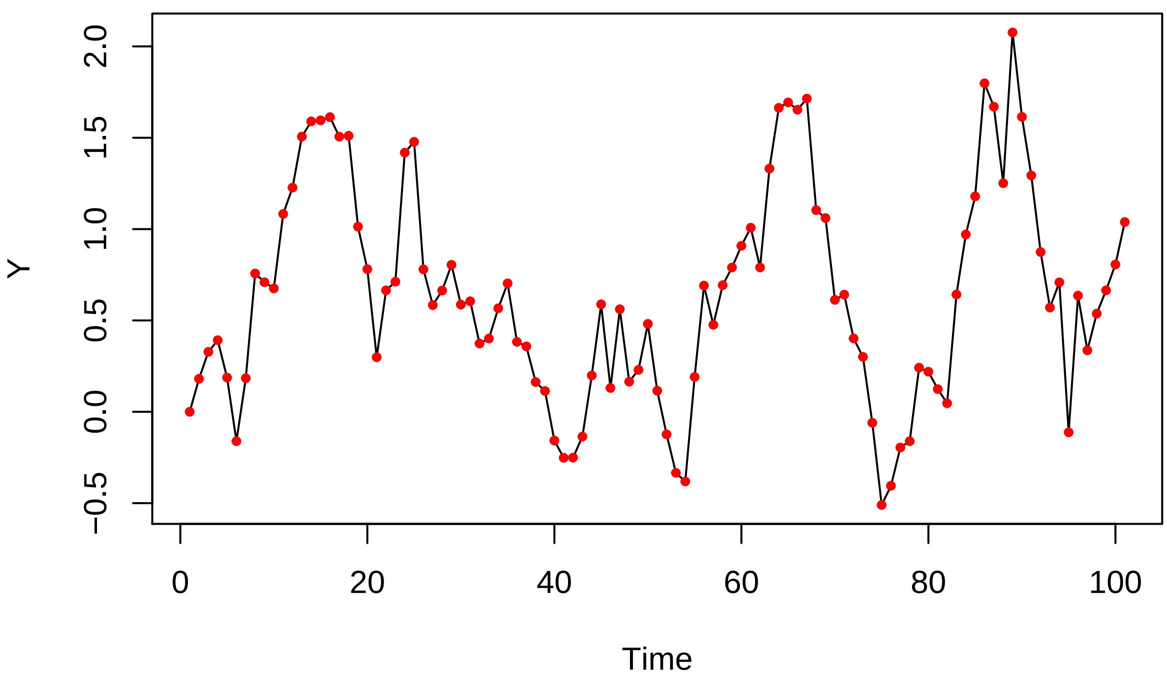
```
## Multiple Regression Analysis:
##     2 regressors(including intercept) and 100 observations
##
## lm(formula = Y ~ back(Y))
##
## Coefficients:
##             Estimate Std Error t value p value
```

```
## (Intercept)   0.1075   0.04595    2.34   0.021
## back(Y)       0.8476   0.05290   16.02   0.000
## ---
## Standard Error of the Regression:  0.3123
## Multiple R-squared:  0.724  Adjusted R-squared:  0.721
## Overall F stat: 256.67 on 1 and 98 DF, pvalue= 0
```

```r
plot(Y ~ back(Y),pch=20, col="blue")
```




```r
plot(Y, type="l", xlab="Time", ylab="Y"); points(Y, pch=20, cex = 0.8, col = "red")
```

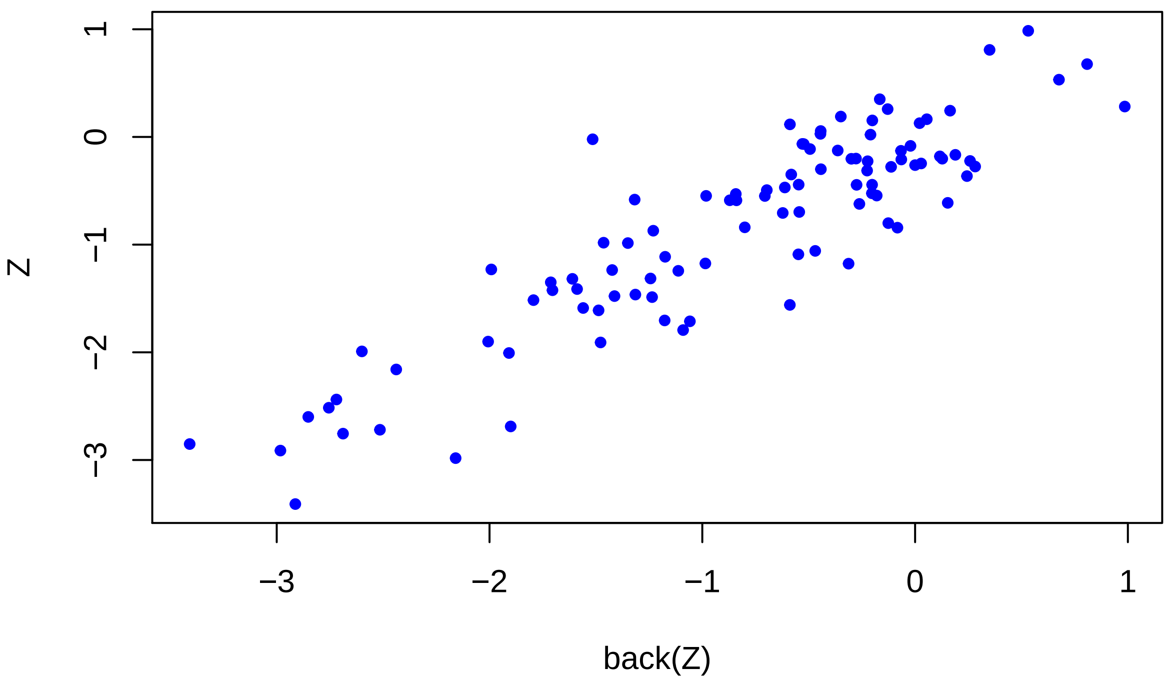


c. A random walk: $y_t = y_{t-1} + \varepsilon_t$

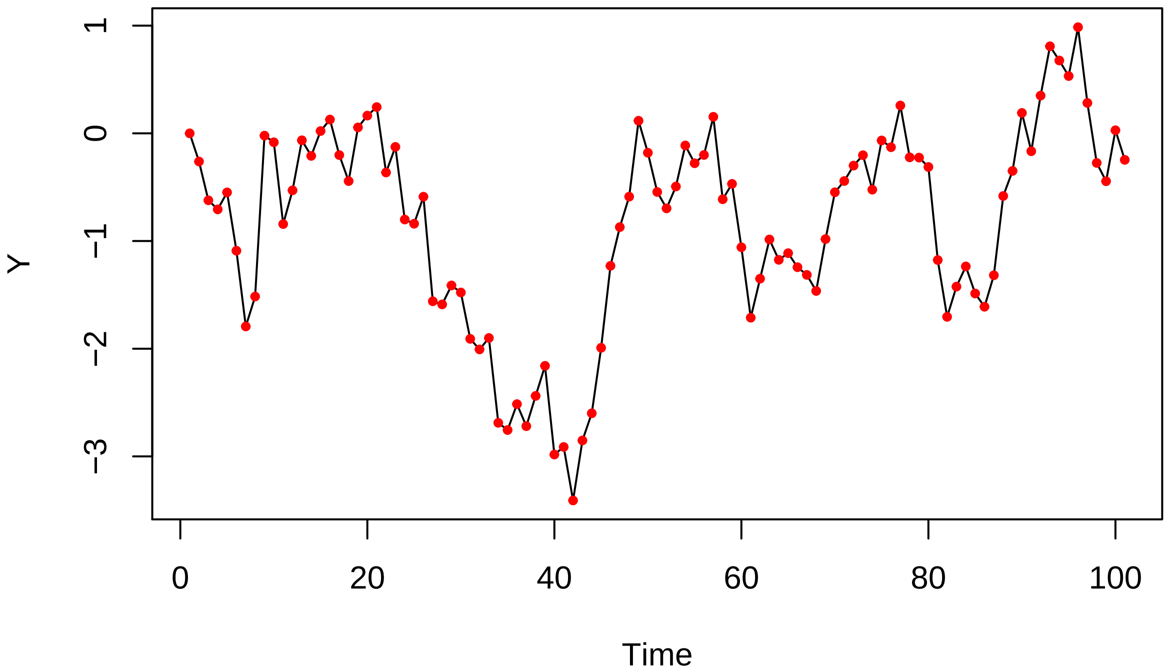
```r
library(DataAnalytics)
Z = double(100)
Z[1] <- 0
for (i in seq(1,100,1)){
Z[i+1] = Z[i] + rnorm(1, sd = 0.4)
}
lmSumm(lm(Z~back(Z)))
```

```
## Multiple Regression Analysis:
##     2 regressors(including intercept) and 100 observations
##
## lm(formula = Z ~ back(Z))
##
## Coefficients:
##             Estimate Std Error t value p value
## (Intercept) -0.09243   0.05658   -1.63   0.106
## back(Z)      0.89280   0.04504   19.82   0.000
## ---
## Standard Error of the Regression:  0.4209
## Multiple R-squared:  0.8  Adjusted R-squared:  0.798
## Overall F stat: 393 on 1 and 98 DF, pvalue= 0
```

```r
plot(Z ~ back(Z),pch=20, col="blue")
```




```r
plot(Z, type="l", xlab="Time", ylab="Y"); points(Z, pch=20, cex = 0.8, col = "red")
```



Question 3

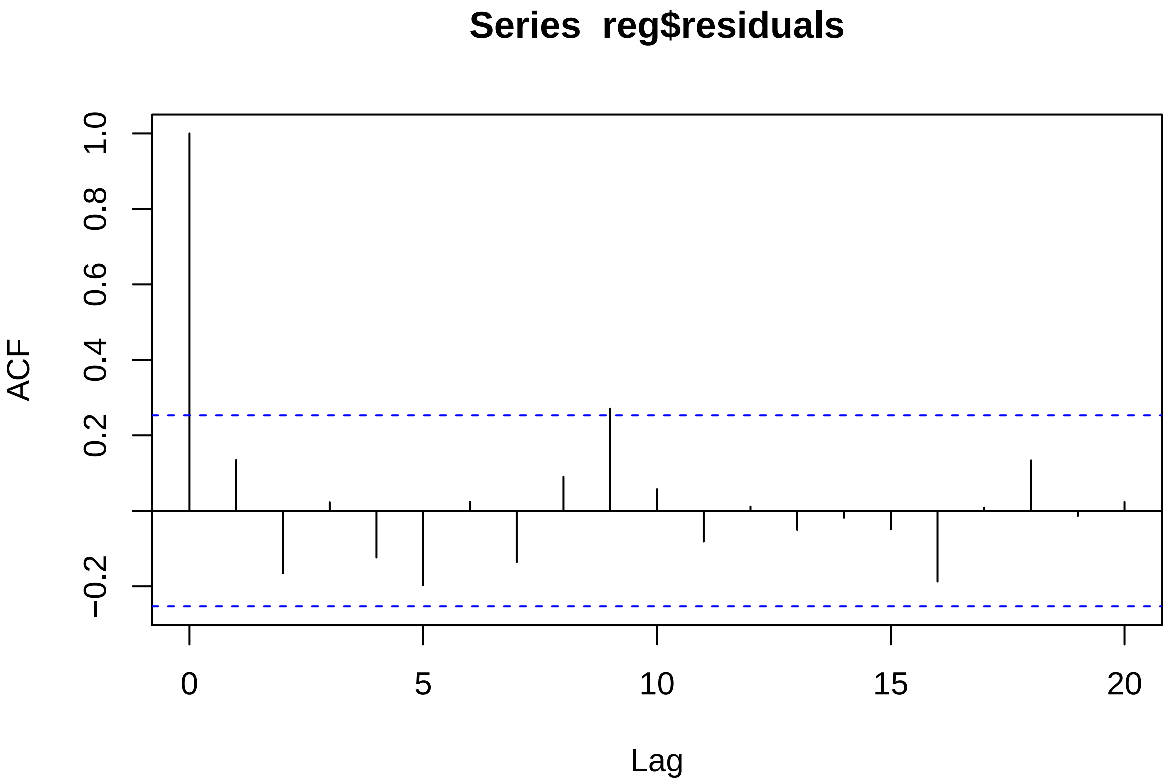
a. Regression using three lags:

```
data("beerprod")
reg = lm(beerprod$b_prod~back(beerprod$b_prod)+back(beerprod$b_prod,6)+back(beerprod$b_prod,12),data=bee
lmSumm(reg)
```

```
## Multiple Regression Analysis:
##     4 regressors(including intercept) and 60 observations
##
## lm(formula = beerprod$b_prod ~ back(beerprod$b_prod) + back(beerprod$b_prod,
##     6) + back(beerprod$b_prod, 12), data = beerprod)
##
## Coefficients:
##                            Estimate Std Error t value p value
## (Intercept)                 7.83100   3.08000    2.54   0.014
## back(beerprod$b_prod)       0.04601   0.06570    0.70   0.487
## back(beerprod$b_prod, 6)   -0.21900   0.09554   -2.29   0.026
## back(beerprod$b_prod, 12)   0.68820   0.09445    7.29   0.000
## ---
## Standard Error of the Regression:  0.6491
## Multiple R-squared:  0.891  Adjusted R-squared:  0.885
## Overall F stat: 152.16 on 3 and 56 DF, pvalue= 0
```

b. From the ACF plot for residuals, we can see that there is no autocorrelation left. This is also evident from Box-Ljung test, with a p-value of 0.4799, we fail to reject the null hypothesis (Autocorrelation is zero between beer production and residuals for lag of 20 periods together). Hence, almost all the predictability in lag periods is absorbed into regression.

```r
acf(reg$residuals, lag=20)
```

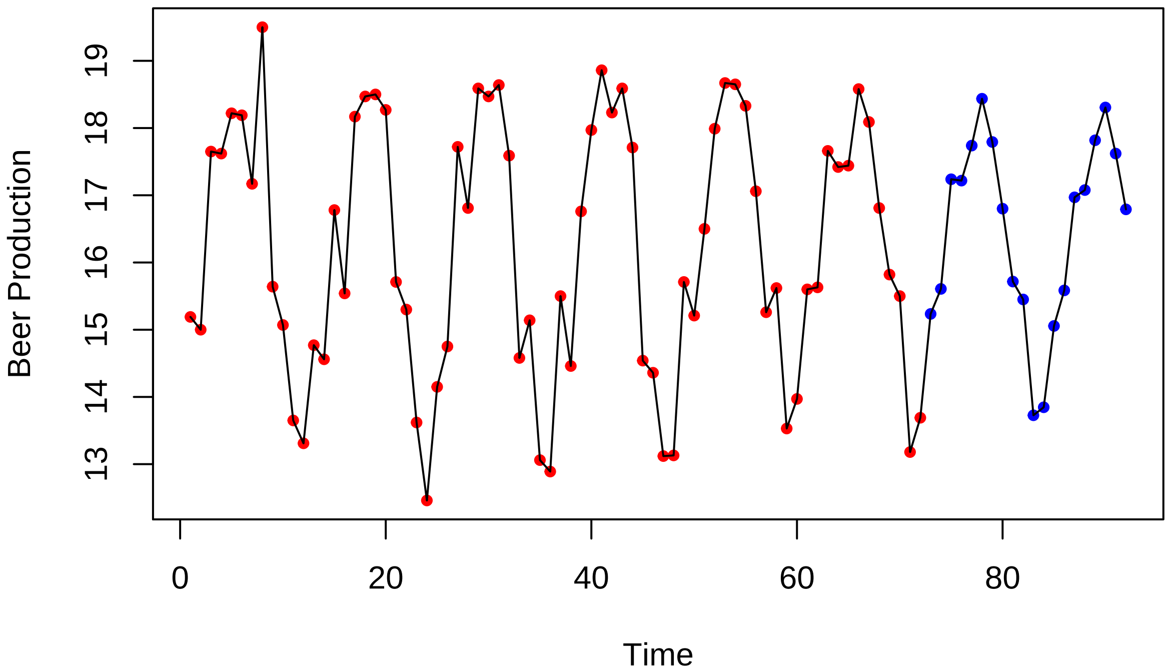



```r
Box.test(reg$res, type = "Ljung", lag = 20)
```

```
##
##  Box-Ljung test
##
## data:  reg$res
## X-squared = 19.652, df = 20, p-value = 0.4799
```

c.

```r
data("beerprod")
n = length(beerprod$b_prod)

b0 <- reg$coefficients[1]
b1 <- reg$coefficients[2]
b2 <- reg$coefficients[3]
b3 <- reg$coefficients[4]

beer_pred = beerprod$b_prod

for (i in 1:20){

  beer_pred[72+i] = b0 + b1*beer_pred[72-1+i] + b2*beer_pred[72-6+i] + b3*beer_pred[72-12+i]

}
t=seq(length(beer_pred))
plot(t, beer_pred, pch=20, col=ifelse(t>72, "blue", "red"), xlab="Time", ylab="Beer Production")
lines(beer_pred)
```



Question 4

a. We know that:

$$\beta = \rho * \frac{\sigma_y}{\sigma_x}$$

Assuming that AR(1) model is stationary, we can say the time series has a constant variance. Hence, standard deviation of dependent variable σ_y is equal to the standard deviation of its lag σ_x.

Therefore, we can prove that $\beta = \rho$, for a stationary AR(1) model.

b. In the lecture slides for Chapter 4, slide 19 states, "if all the true autocorrelations are 0, then the standard deviation of the sample autocorrelations is about $1/\sqrt{T}$". Prove this for an AR(1) model. (Hint: recall the formula for s_{b_1} from the Chapter 1 slides.)

We know that:

$$s_{b_1} = \sqrt{\frac{s^2}{(T-1)s_x^2}}$$

From (a), we can see that $\beta = \rho$. Hence, the standard deviation of sample autocorrelations is same as the expected standard deviation (i.e, standard error) of coefficient on the lagged dependent variable.

Also, since all the true autocorrelations are 0, all the variablity in the dependent variable is only due to the error term. Mathematically, for an AR(1) model,

$$y_t = \beta_0 + \beta_1 y_{t-1} + \varepsilon$$

Taking variance to the above equation, we get

$$\sigma_y^2 = \frac{\sigma^2}{1 - \beta_1^2}$$

Applying the fact that $\sigma_y = \sigma_x$ above for AR(1) model and taking for sample, we get

$$s_x^2 = \frac{s^2}{1 - b_1^2}$$

We know, $b = \rho = 0$ here. Hence, $s_x = s$. Using this result for s_{b_1} above, we get

$$s_{b_1} = \sqrt{\frac{1}{(T-1)}}$$

So for large sample sizes, we can prove that standard deviation of the sample autocorrelations is about $1/\sqrt{T}$

Question 5

Let's explore the log transformation to address nonlinearity and heterogeneity using the `diamonds` dataset in the `ggplot2` package. Because this is a large dataset, we will focus only on the subset of the data where the cut is "ideal" and the color is "D". Thus, for this question, you should be working with 2,834 data points.

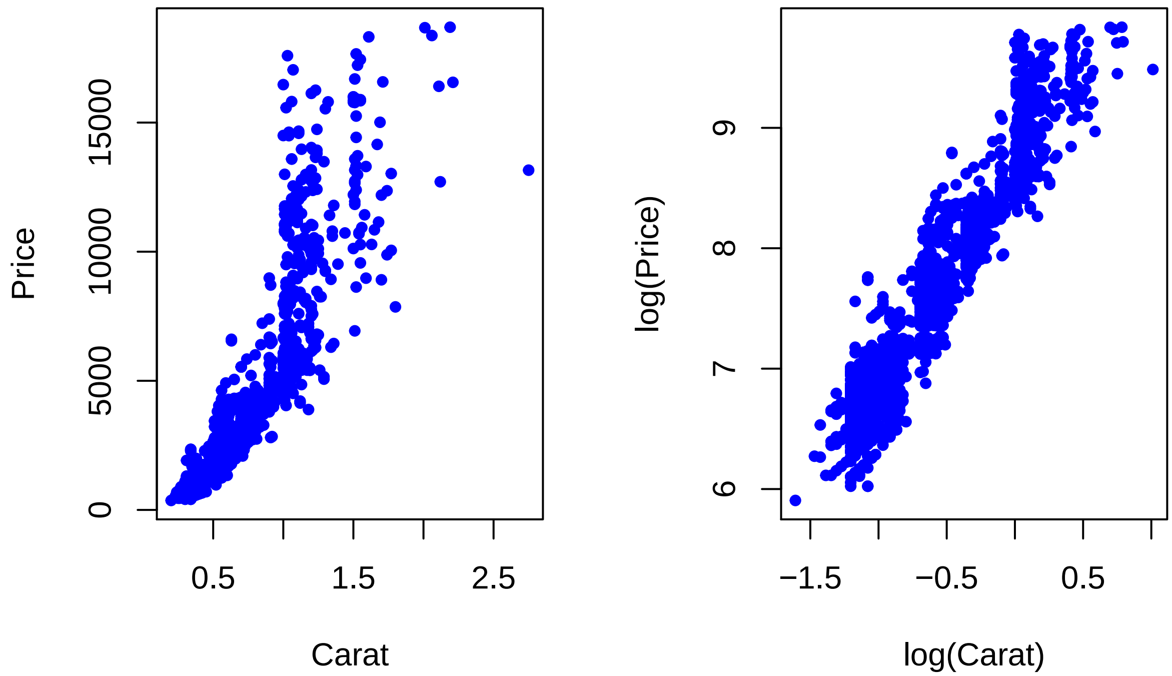
```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:xts':
##
##     first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
data("diamonds")

diamonds <- filter(diamonds, cut == "Ideal", color == 'D')
```

a) Plot of (1) carat vs price, and (2) log(carat) vs log(price)

```
par(mfrow=c(1,2))
plot(diamonds$price ~ diamonds$carat ,pch=20, col="blue", xlab = "Carat", ylab = "Price")
plot(log(diamonds$price) ~ log(diamonds$carat) ,pch=20, col="blue", xlab = "log(Carat)", ylab = "log(Pr
```



b)

```
lmSumm(lm(log(price) ~ log(carat) + clarity, data = diamonds))
```

```
## Multiple Regression Analysis:
##     9 regressors(including intercept) and 2834 observations
##
## lm(formula = log(price) ~ log(carat) + clarity, data = diamonds)
##
## Coefficients:
##              Estimate Std Error t value p value
## (Intercept)  8.805000  0.007370 1194.74   0.000
## log(carat)   1.892000  0.006054  312.52   0.000
## clarity.L    1.163000  0.026140   44.49   0.000
## clarity.Q   -0.003312  0.025820   -0.13   0.898
## clarity.C    0.205900  0.021190    9.72   0.000
## clarity^4    0.030980  0.015490    2.00   0.046
## clarity^5    0.014080  0.010840    1.30   0.194
## clarity^6    0.010610  0.007953    1.33   0.182
## clarity^7    0.064090  0.006578    9.74   0.000
## ---
## Standard Error of the Regression:  0.1401
## Multiple R-squared:  0.973  Adjusted R-squared:  0.973
## Overall F stat: 12650.95 on 8 and 2825 DF, pvalue= 0
```

We can see that the Levels: I1 $<$ SI2 $<$ SI1 $<$ VS2 $<$ VS1 $<$ VVS2 $<$ VVS1 $<$ IF

For unit change in 'IF' clarity, log(Price) changes by 0.064 For unit change in 'SI2' clarity, log(Price) changes by 1.163

Price premium of 'IF' over 'SI2': $e^{1.099} = 3$ times.

c)

```
diamonds_others <- filter(diamonds, clarity  %in% c("I1","SI2","VS2","VS1","VVS2","VVS1"))
diamonds_IF <- filter(diamonds, clarity == "IF")
diamonds_SI1 <- filter(diamonds, clarity == "SI1")
 df <- data.frame(
        y = log(diamonds_others$price),
        x = log(diamonds_others$carat),
        clarity = diamonds_others$clarity
    )
 df1 <- data.frame(
       y1 = log(diamonds_IF$price),
       x1 = log(diamonds_IF$carat),
       clarity = diamonds_IF$clarity
    )

 df2 <- data.frame(
       y1 = log(diamonds_SI1$price),
       x1 = log(diamonds_SI1$carat),
       clarity = diamonds_SI1$clarity
    )


p <- ggplot()+geom_point(data = df, aes(x = x,y = y, colour=clarity))+ scale_color_manual(values = c("I1
p
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```