HW 4
MFE 402: Econometrics
Professor Rossi
Student: Xiahao Wang

This problem set is designed to review material on time series and advanced regression topics. Include both your R code and output in your answers.

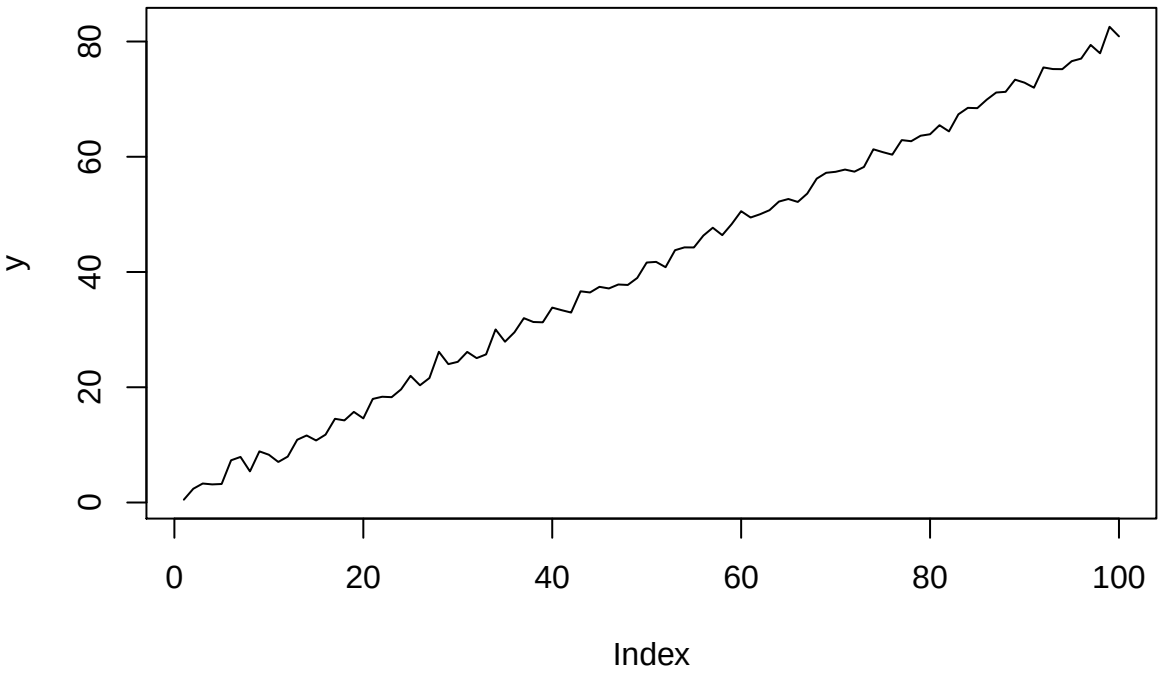
Question 1

Simulate data for the following models and provide a plot of each:

a. A linear time trend: $y_t = \alpha + \beta t + \varepsilon_t$
b. An AR(1): $y_t = \alpha + \beta y_{t-1} + \varepsilon_t$
c. A random walk: $y_t = y_{t-1} + \varepsilon_t$

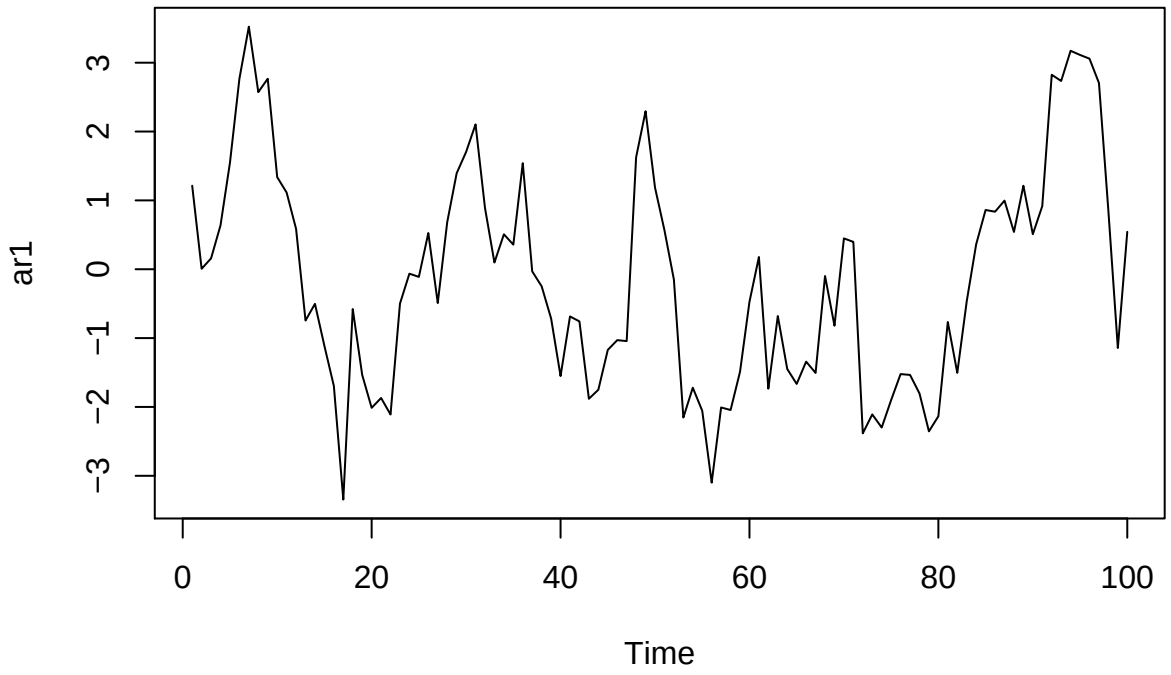
(a) linear trend model

```
num <- 100
alpha <- 0.5
beta <- 0.8
sigma <- 1
mu <- 0
t <- 1:num
y <- double(num)
for(i in 1:num){
  y[i] <- alpha + beta * t[i] + rnorm(1, mean = mu, sd=sigma )
}
plot(y, type = "l" )
```



(b) AR(1) Model simulate using arima.sim function

```
ar1 <- arima.sim(model = list(order = c(1, 0, 0), ar = .9), n = 100)
plot(ar1)
```



or simulate using FRED data

```
library(DataAnalytics)
library(quantmod)
```

```
## Loading required package: xts
```

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

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

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

```
## Loading required package: TTR
```

```
## Version 0.4-0 included new data defaults. See ?getSymbols.
```

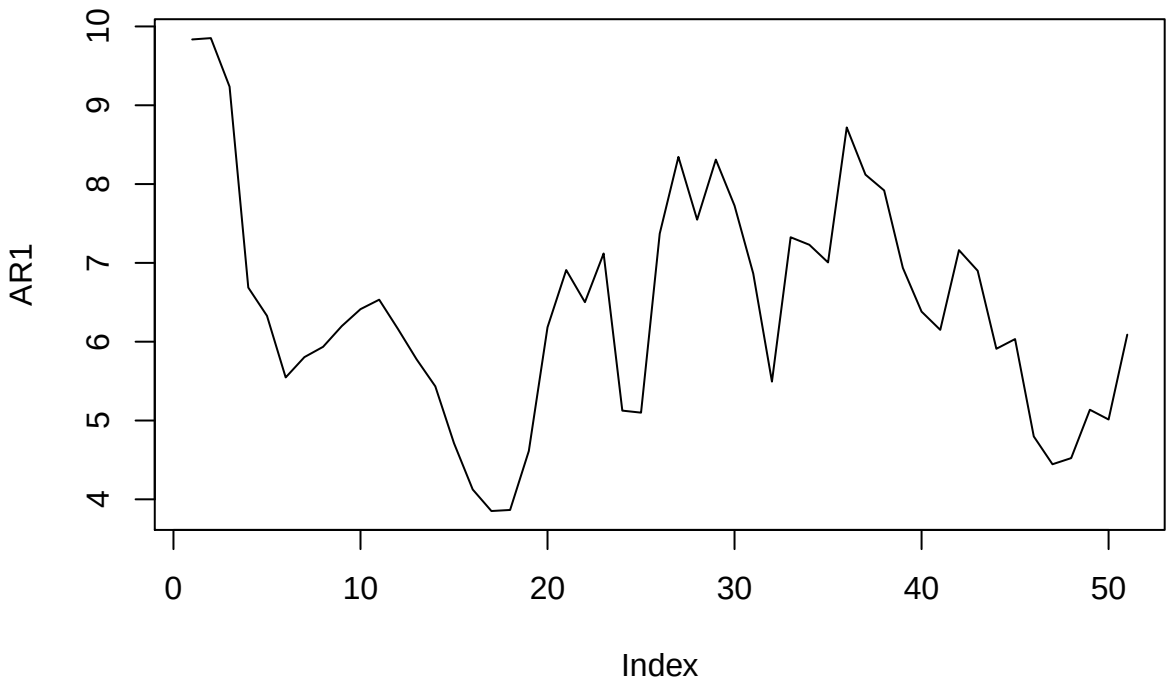
```
getSymbols("GDPC1",src = "FRED")
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

```
## [1] "GDPC1"
```

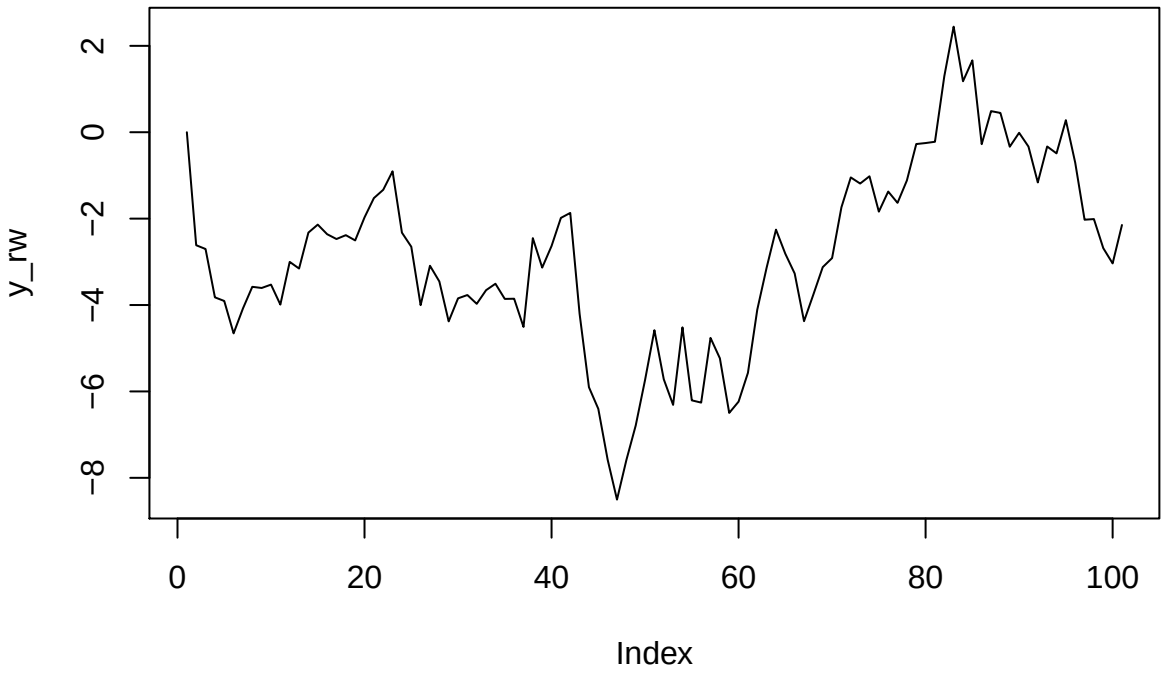
```
lnGDP <- log(GDPC1)
trend <-1:length(lnGDP)
out <- lm(lnGDP~trend)
```

```
lnGDP <- as.vector(lnGDP)
nstep <- 50
out.ar <- lm(lnGDP~back(lnGDP))
pred.ar <- double(nstep+1)
pred.ar[1] <- lnGDP[length(lnGDP)]
for(i in 1:nstep){
  pred.ar[i+1] <- out.ar$coef[1] + out.ar$coef[2] * pred.ar[i] + rnorm(1, mean = mu, sd=sigma )
}

plot(pred.ar, type="l", ylab="AR1")
```



(c) random walk

```
y_rw <- double(num)

for(i in 1:num){
  y_rw[i+1] <- y_rw[i] + rnorm(1, mean = mu, sd=sigma )
}
plot(y_rw, type="l")
```



Question 2

a. Using the `beerprod` data from the `DataAnalytics` package, regress beer production on its 1-period, 6-period, and 12-period lags. This should be one regression, not three separate regressions.
b. Test to see if there is any autocorrelation left in the residuals. Comment on what you find.
c. Predict beer production for the next 20 months. Plot your prediction.

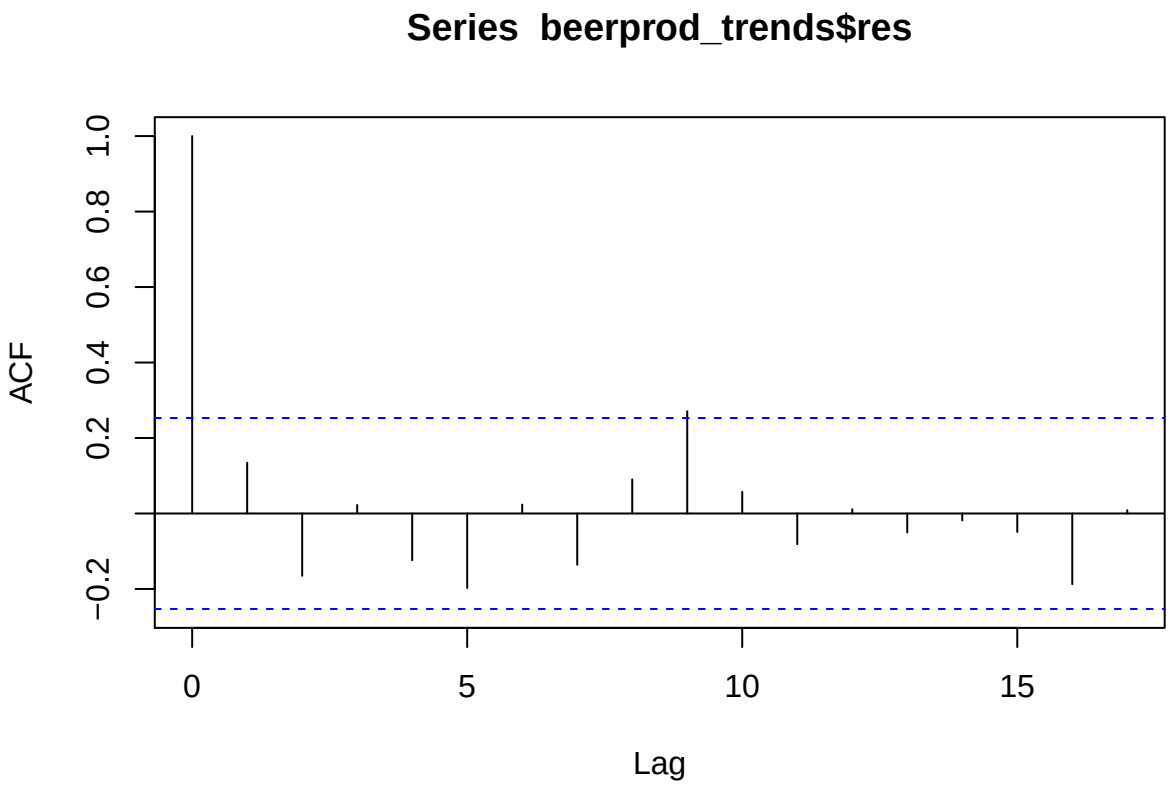
(a)

```
data("beerprod")
beerprod$b_prod.lag1 = back(beerprod$b_prod)
beerprod$b_prod.lag6 = back(beerprod$b_prod,6)
beerprod$b_prod.lag12 = back(beerprod$b_prod,12)
beerprod_trends <- lm(b_prod ~ b_prod.lag1 + b_prod.lag6 + b_prod.lag12, data = beerprod)
lmSumm(beerprod_trends)
```

```
## Multiple Regression Analysis:
##     4 regressors(including intercept) and 60 observations
##
## lm(formula = b_prod ~ b_prod.lag1 + b_prod.lag6 + b_prod.lag12,
##     data = beerprod)
##
## Coefficients:
##              Estimate Std Error t value p value
## (Intercept)   7.83100   3.08000    2.54   0.014
## b_prod.lag1   0.04601   0.06570    0.70   0.487
## b_prod.lag6  -0.21900   0.09554   -2.29   0.026
## b_prod.lag12  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)

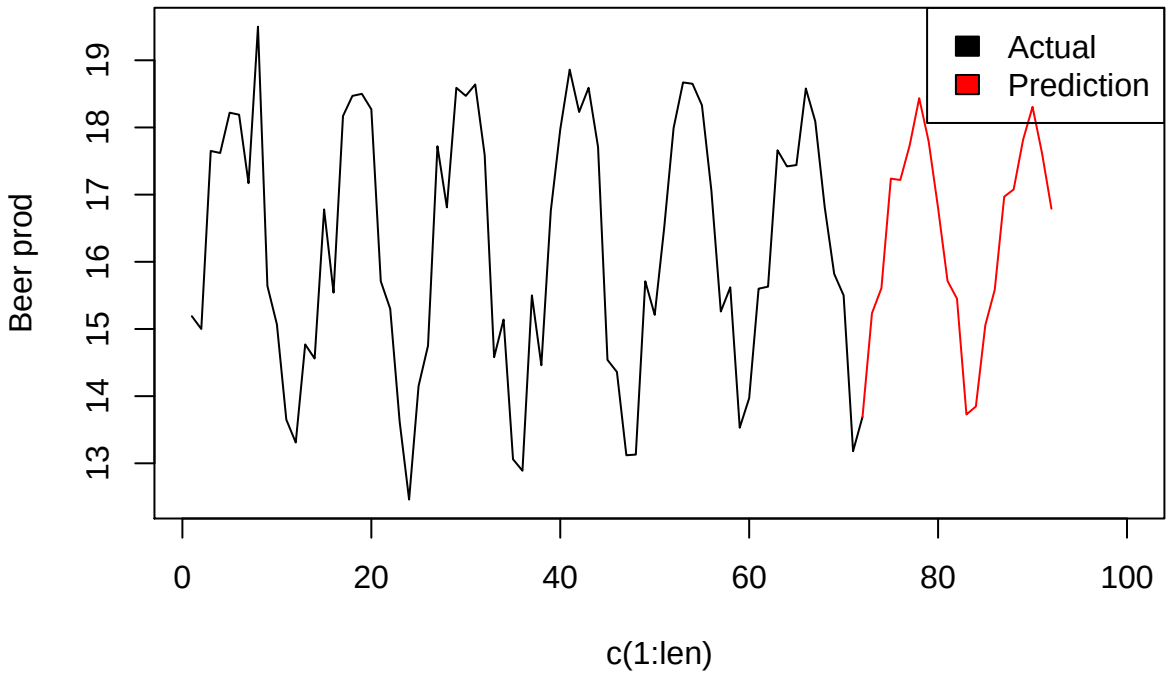
```
acf(beerprod_trends$res,na.action=na.omit)
```



According to the graph, most of the residuals are below the level, which indicates a zero autocorrelation

(c)

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

data <- beerprod$b_prod
len <- length(data)

for(i in 1:20){
  data[len+i] <- b0 + b1*data[len+i-1] + b2*data[len+i-6] + b3*data[len+i-12]
}
plot(x = c(1:len),y = data[1:len],type = 'l', xlim = c(1,100), ylab= "Beer prod")
lines(x = c(len:length(data)), y = data[len:length(data)], type = 'l', col = 'red')
legend("topright",legend=c("Actual","Prediction"),fill=c("black","red"))
```



Question 3

a. Assuming the AR(1) model is stationary, prove that the coefficient on the lagged dependent variable (β) is equal to the correlation between the dependent variable and its lag (ρ).

Stationarity usually means covariance stationarity, where the expected value, variance, and autocovariances of y_t do not depend on t, that is, they do not change over time

correlation is:

$$\rho_1 = \frac{cov(y_t, y_{t-1})}{var(y_t)} = \frac{cov(y_t, y_{t+1})}{var(y_t)}$$

to simplify the prove, assume mean = 0, then α is 0 and $y_t = \beta y_{t-1} + \varepsilon_t$

$$cov(y_t, y_{t-1}) = E[y_t \cdot y_{t-1}] = E[y_t \cdot y_{t+1}]$$
$$= E[y_t \cdot (\beta y_t + \varepsilon_{t+1})] = E[\beta \cdot y_t^2 + y_t \cdot \varepsilon_{t+1}] = \beta var(y_t)$$
$$\rho_1 = \frac{cov(y_t, y_{t-1})}{var(y_t)} = \frac{\beta var(y_t)}{var(y_t)} = \beta$$

b. In the lecture slides for Chapter 4, slide 15 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.)

Same as part a, assume $\beta_0 = 0$ for simple illustration

forumal for standard error: $s_{b_1} = \sqrt{\frac{s^2}{(N-1)S_x^2}}$

Given autocorrelation is zero, $cov(y_t, y_{t-1}) = 0$, hence $\beta = 0$ hence $y_t \sim$ N(0, σ_ε^2) and $y_{t-1} \sim$ N(0, σ_ε^2)

hence $s_{b_1} = \sqrt{\frac{\sigma_\varepsilon^2}{(N-1)\sigma_\varepsilon^2}} = \frac{1}{\sqrt{N-1}}$

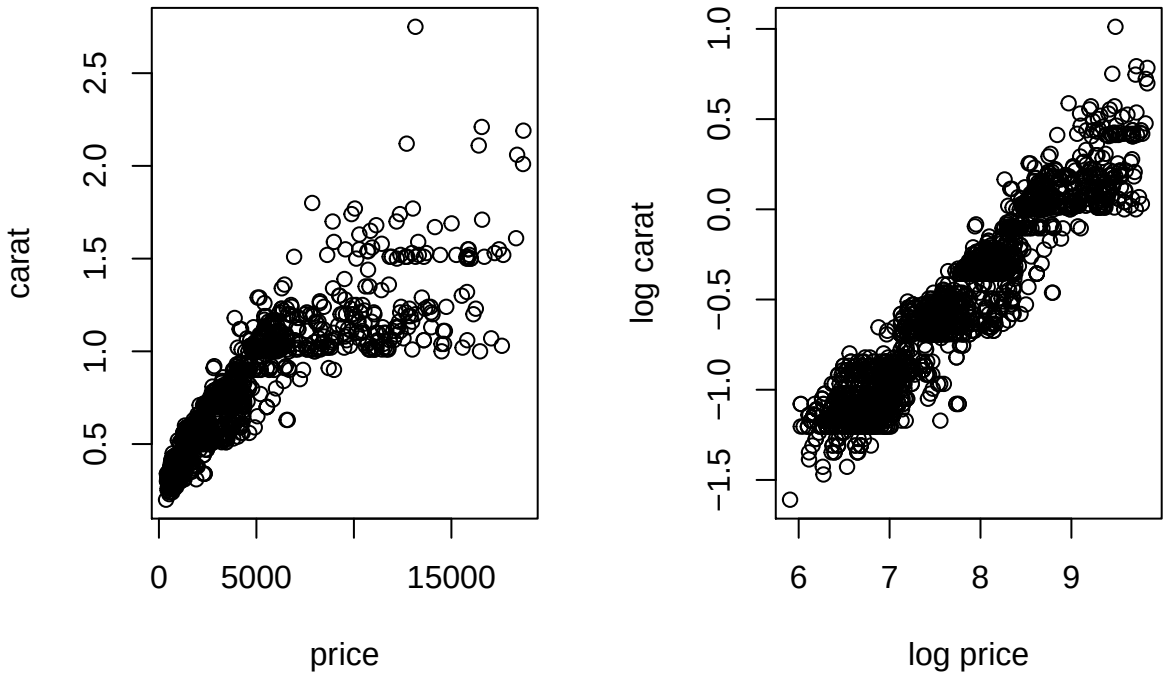
Lag 1, only N-1 time period in the regresion. N-1 = T, Hence the standard deviation of the sample autocorrelation is $1/\sqrt{T}$

Question 4

Let's explore the log transformation to address nonlinearity and heterogeneity. To do so, we will use 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.

a) Plot (1) carat vs price, and (2) log(carat) vs log(price). Use `par(mfrow=c(1,2))` to put two plots side by side.

```
library(ggplot2)
data("diamonds")
ideal_diamonds <- diamonds[diamonds$cut=="Ideal" & diamonds$color=="D",]
par(mfrow=c(1,2))
plot(y = ideal_diamonds$carat, x = ideal_diamonds$price, ylab= "carat", xlab="price")
plot(y = log(ideal_diamonds$carat), x = log(ideal_diamonds$price), ylab = "log carat", xlab="log price")
```



b) Regress log(price) on log(carat) and dummy variables for the levels of clarity. What price premium does a diamond with clarity "IF" command relative to a diamond with clarity "SI2"?

```
diamonds_clarity <- ideal_diamonds[,c(1,4,7)]
diamonds_clarity$logcarat <- log(diamonds_clarity$carat)
diamonds_clarity$logprice <- log(diamonds_clarity$price)
diamonds_clarity <- na.omit(diamonds_clarity)

library(fastDummies)
diamonds_dummy <- dummy_cols(diamonds_clarity, select_columns = "clarity")
lm_diamond <- diamonds_dummy[,-c(1:3,6)]
#taking SI1 as base
out <- lm(formula = logprice ~ ., data = lm_diamond)
price_premium <-out$coefficients[8] - out$coefficients[3]
lmSumm(out)
```

```
## Multiple Regression Analysis:
```

```
##     9 regressors(including intercept) and 2834 observations
##
## lm(formula = logprice ~ ., data = lm_diamond)
##
## Coefficients:
##              Estimate Std Error t value p value
## (Intercept)    8.5990  0.006424 1338.59       0
## logcarat       1.8920  0.006054  312.52       0
## clarity_SI2   -0.1848  0.009140  -20.22       0
## clarity_VS1    0.2329  0.009097   25.60       0
## clarity_VVS1   0.5567  0.012830   43.39       0
## clarity_VS2    0.1970  0.006981   28.22       0
## clarity_VVS2   0.4172  0.009800   42.57       0
## clarity_IF     0.9333  0.026970   34.60       0
## clarity_I1    -0.5067  0.039260  -12.91       0
## ---
## 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
```

price premium is:

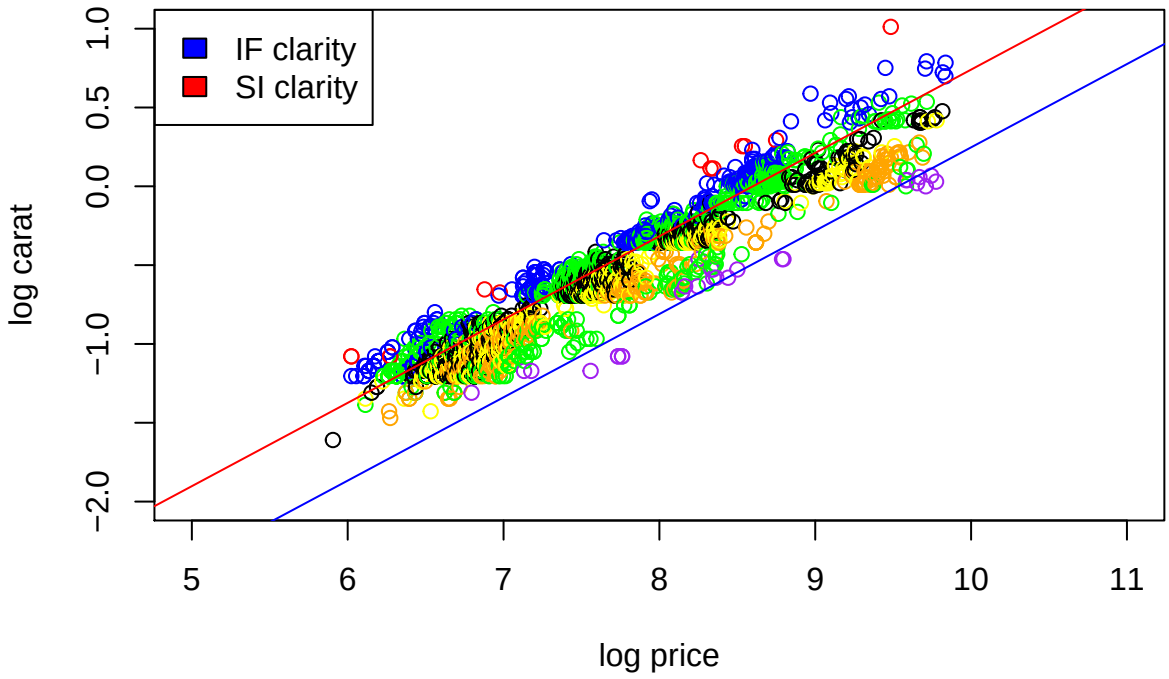
```
price_premium
```

```
## clarity_IF
##   1.118106
```

```
# or use the following code below lm recognizes factor
# out2 <- lm(log(price) ~ log(carat) + factor(clarity, order = F), data=diamonds_clarity)
```

c) Repeat the second plot in part (a) above (i.e., log(carat) vs log(price)) but make 2 additions. First, color each point by its level of clarity. Second, add the fitted regression line for the following two levels clarity: "IF" and "SI1". Be sure to match the color of each line to the color of the corresponding points.

```
plot(xlab="log price", ylab = "log carat", y = diamonds_clarity$logcarat, x = diamonds_clarity$logprice
#IF clarity
abline(a=-(out$coefficients[1] + out$coefficients[8])/out$coefficients[2], b=1/out$coefficients[2], col=

#SI clarity
abline(a=-out$coefficients[1]/out$coefficients[2], b=1/out$coefficients[2], col="red")
legend("topleft",legend=c("IF clarity","SI clarity"), fill=c("blue","red"))
```



Question 5

a. Using the R dataset mtcars, calculate the correlation between vehicle fuel efficiency (as measured by mpg) and engine displacement (disp). Then construct a bootstrapped 95% confidence interval for the correlation.

b. Plot the distribution of your bootstrapped correlations and label the sample correlation.

(a) correlation between mpg and disp

```
data(mtcars)
cor_mpg_disp <- cor(mtcars$mpg, mtcars$disp)
cor_mpg_disp
```

```
## [1] -0.8475514
```

```
reg_data <- data.frame(mtcars$mpg, mtcars$disp)
reg_data <- na.omit(reg_data)
colnames(reg_data) <- c("MPG", "DISP")
N <- nrow(reg_data)

B <- 10000
bs_coef <- matrix(0, nrow=B, ncol = 1)
for(b in 1:B){
  bs_sample <- reg_data[sample(1:N,size=N,replace = TRUE),]
  cor_dist <- cor(bs_sample$MPG, bs_sample$DISP)
  bs_coef[b,] <-cor_dist
}

int <- quantile(bs_coef[,1], probs=c(0.975, 0.025))
CI.bootstrap <- c(2*cor_mpg_disp-int[1], 2*cor_mpg_disp-int[2])
```

confidence interval:

```
CI.bootstrap
```

```
##      97.5%       2.5%
## -0.9358143 -0.7829262
```

(b)

```
hist(bs_coef[,1],breaks=40,col="magenta",xlab="",ylab="",
main="Bootstrap Dist of Correlation between MPG and DISP",
sub=paste("Sample Correlation = ",round(cor_mpg_disp,digits=3)))
abline(v=cor_mpg_disp,lwd=3,col="yellow")
legend("topright",legend=c("Sample Correlation"), fill=c("yellow"))
```