Stat Econ 2 First

Victor Z. Nygaard

Last compiled on 01. oktober, 2021

Stat Econ 2 Assignment 1

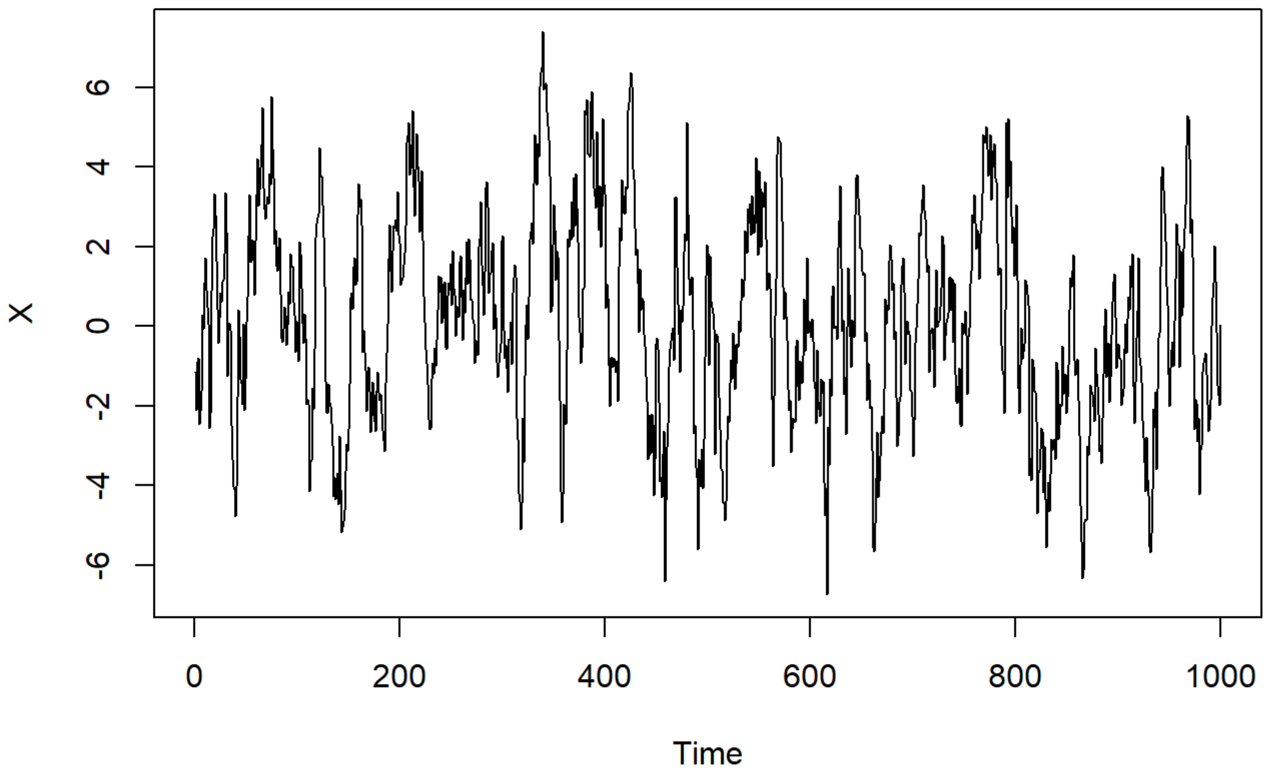
Exercise 1

We might note that we are dealing with an AR(1) process with slope 0.9 and, assuming a central requirement, central t - distributed noise with ten degrees of freedom. We simulate noise terms Z_t for $t = 1, \ldots, 1000$

```
set.seed(314)
desiredlag <- 20
phi <- 0.9
n <- 10^3
Z <- rt(df = 10, n=n)
```

As such we may simulate the AR(1) process using the update scheme defining the 'stopped' AR(1) process

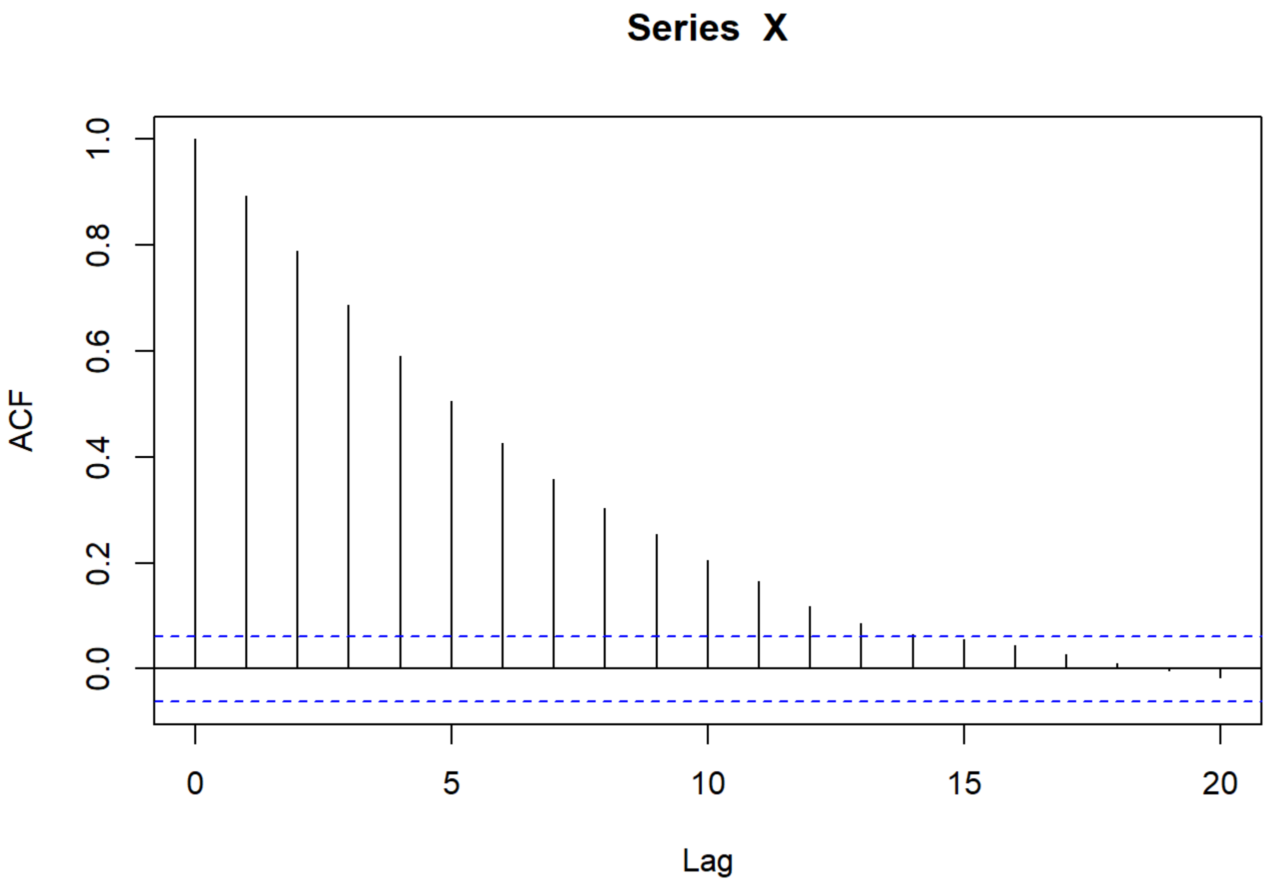
```
X <- rep(NA,n)
X[1] <- Z[1]
for (j in 2:n) {
   X[j] <- phi*X[j-1]+Z[j]
}
ts.plot(X)
```



a)

We plot the `acf` confidence bands:

```
acf(X, lag.max = desiredlag, plot = T)
```



We would also have done the permutations using a homemade function that would repeatedly use the sample function - I did not get this to work.

b)

We note that our AR(1) model is causal as $|\phi| < 1$. Following example 4.25, we will need to calculate the sample autocovariance function:

$$\gamma_{n,X}(h) := \frac{1}{n} \sum_{t=1}^{n-h} \left(X_t - \overline{X}_n \right) \left(X_{t+h} - \overline{X}_n \right)$$

and autocorrelation function:

$$\rho_{n,X}(h) := \frac{\gamma_{n,X}(h)}{\gamma_{n,X}(0)}$$

Note thus in particular that we may calculate $\gamma_{n,X}(1)$, $\gamma_{n,X}(0)$ in R with the following homemade function

```
gamma <- function(X,h) {
   n <- length(X)
   gamt <- 0
   for (t in 1:(n-h)) {
      tempt <- (X[t]-mean(X))*(X[t+h]-mean(X))
      gamt <- gamt + tempt
   }
   1/n*gamt
}
```

Yielding

```
gamma(X,1)
```

```
## [1] 5.640264
```

```
gamma(X,0)
```

```
## [1] 6.317588
```

Such that for

$$\hat{\phi}_n = \frac{\gamma_{n,X}(1)}{\gamma_{n,X}(0)} \equiv \rho_{n,X}(1)$$

$$\hat{\sigma}_n^2 = \gamma_{n,X}(0)\left(1 - \rho_{n,X}^2(1)\right)$$

```
(rho1 <- gamma(X,1)/gamma(X,0))
```

```
## [1] 0.8927876
```

```
(phih <- rho1)
```

```
## [1] 0.8927876
```

```
(sigmah2 <- gamma(X,0)*(1-rho1^2))
```

```
## [1] 1.28203
```

Let $\nu = 10$ be the degrees of freedom of our student-t distributed random noise Z_t. As is surmised on page 47-48 in the lecture notes, in dealing with a causal AR(1) process driven by iid noise Z_t with variance $\sigma^2 = \frac{\nu}{\nu-2} = \frac{10}{8} = \frac{5}{4}$, we have asymptotic normality of $\hat{\phi}$ with corresponding asymptotic mean ϕ and asymptotic variance $\frac{\sigma^2\Gamma_p^{-1}}{n}$ i.e.

$$\hat{\phi}_n \overset{as}{\sim} \mathcal{N}\left(\phi, \frac{\sigma^2\Gamma_{p=1}^{-1}}{n}\right)$$

or equivalently

$$\sqrt{n}\left(\hat{\phi} - \phi\right) \xrightarrow{d} \mathcal{N}\left(0, \sigma^2\Gamma_1^{-1}\right).$$

With this we may for our fixed $n = 1000$ determine that $\left(\phi - \frac{1.96}{\sqrt{n}}\sqrt{\sigma^2\Gamma_1^{-1}},\ \phi + \frac{1.96}{\sqrt{n}}\sqrt{\sigma^2\Gamma_1^{-1}}\right)$ will be an asymptotic 95% confidence interval for ϕ. Estimating Γ_1 via the sample autocovariance function, we find:

$$\tilde{\Gamma}_1 := \gamma_{n,X}(1-1) = \gamma_{n,X}(0) = 6.3175881$$

such that

$$\tilde{\Gamma}_1^{-1} = \frac{1}{\tilde{\Gamma}_1} = 0.1582883 \neq 0$$

such that we may rewrite the confidence bands as

$$\left(\phi - \frac{1.96}{\sqrt{n}}\sqrt{\sigma^2 0.1582883},\ \phi + \frac{1.96}{\sqrt{n}}\sqrt{\sigma^2 0.1582883}\right)$$

Inserting the other estimates and $n = 1000$ we get

$$
\left(\hat{\phi}-\frac{1.96}{\sqrt{10^3}}\sqrt{\hat{\sigma}^2 0.1582883},\ \hat{\phi}+\frac{1.96}{\sqrt{10^3}}\sqrt{\hat{\sigma}^2 0.1582883}\right)
$$

$$
=\left(0.8927876-\frac{1.96}{\sqrt{10^3}}\sqrt{0.2029302},\ 0.8927876+\frac{1.96}{\sqrt{10^3}}\sqrt{0.2029302}\right)
$$

$$
=(0.8648667, 0.9207085)
$$

c)

i+ii)

We calculate the residuals:

```
Zh <- rep(NA,n)
Zh[1] <- X[1]
for (j in 2:n) {
   Zh[j] <- X[j]+phih*X[j-1]
}
```

We do a reordering of these in the requested fashion using `sample`:

```
Zhs<-sample(Zh)
```

We define a new sample:

```
Xhs <- rep(NA,n)
Xhs[1] <- Zhs[1]
for (j in 2:n) {
   Xhs[j] <- phih*Xhs[j-1]+Zhs[j]
}
```

iii)

part1)

As in b)

```
(hsrho1 <- gamma(Xhs,1)/gamma(Xhs,0))
```

```
## [1] 0.9068954
```

```
(hsphih <- hsrho1)
```

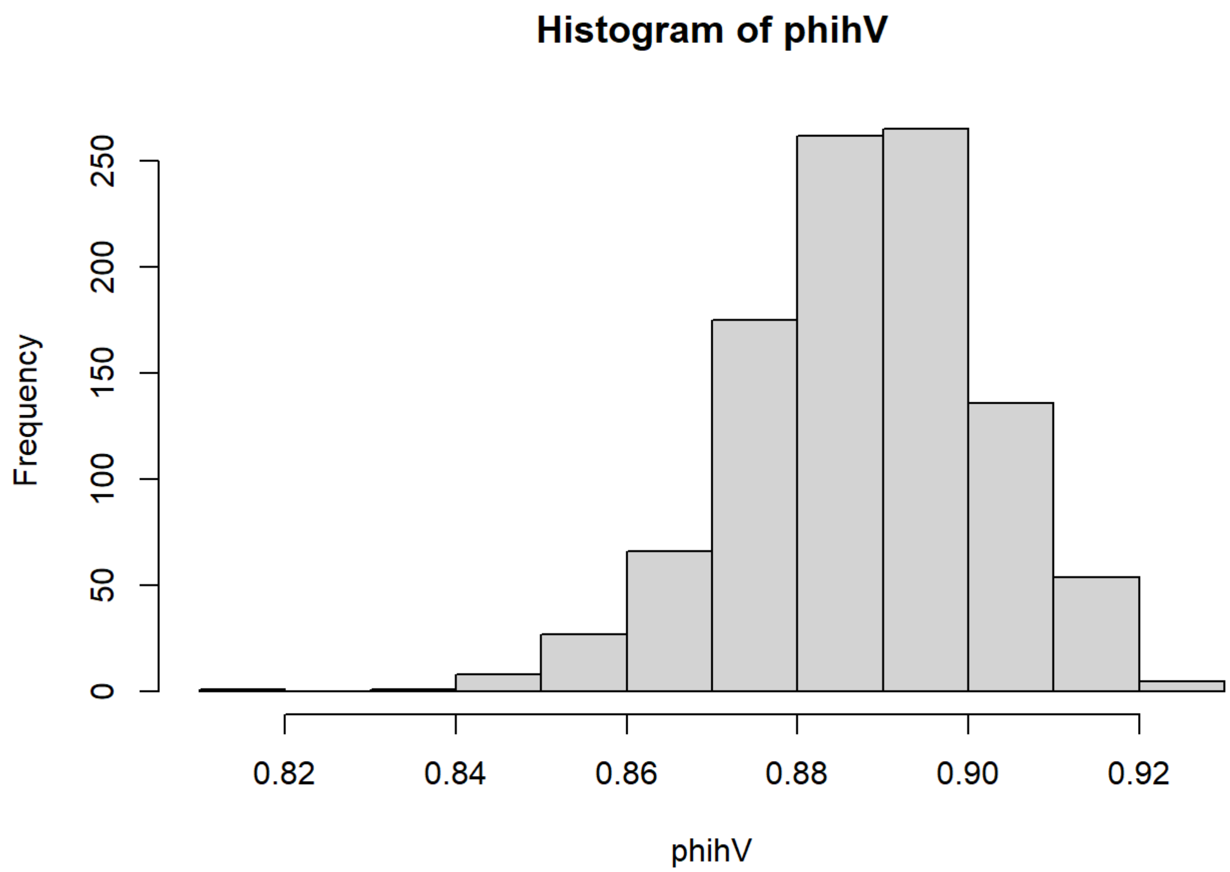
```
## [1] 0.9068954
```

part2)

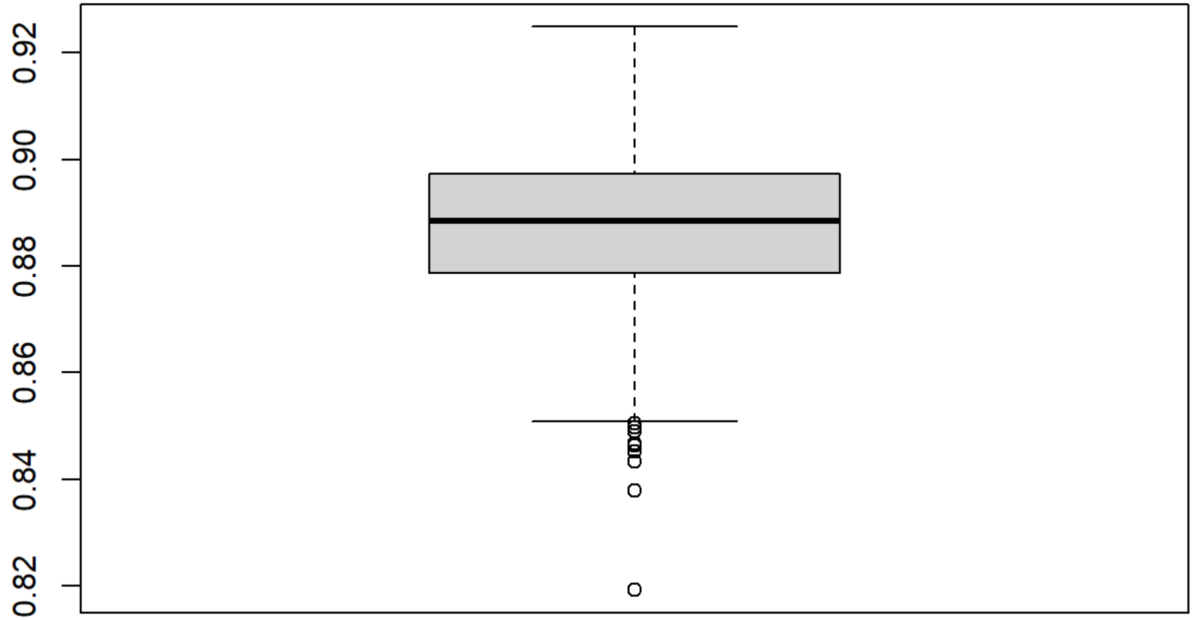
We repeat the tasks done in the previous exercises by writing a function to do so

```
boots <- function(Zh,n) { #Simulating the Bootstrap data
   q <- length(Zh)
   XhsM <- matrix(NA,nrow=n,ncol=q)
   for (i in 1:n) {
      Zhsi <- sample(Zh)
      XhsM[i,1] <- Zhsi[1]
      for (j in 2:q) {
        XhsM[i,j] <- phih*XhsM[i,j-1]+Zhsi[j]
      }
   }
   XhsM
}

dat <- boots(Zh,1000)
YW <- function(Zh,n) { #Calculating the YW's
   q <- length(Zh)
   dat <- boots(Zh,n)
   phihV <- rep(NA,n)
   for (i in 1:n) {
      phihV[i] <- gamma(dat[i,],1)/gamma(dat[i,],0)
   }
   phihV
}
phihV <- YW(Zh,n)
hist(phihV)
```




```
boxplot(phihV)
```




```
quantile(phihV, c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.8570629 0.9138498
```

Comparing this confidence interval to the asymptotic one achieved in b):

```
rbind(quantile(phihV, c(0.025, 0.975)), c(phih - 1.96/(sqrt(n))*sqrt(sigmah2*1/gamma(X,0)),phih + 1.96/(sqrt(n))*sqrt(sigmah
2*1/gamma(X,0))))
```

```
##           2.5%     97.5%
## [1,] 0.8570629 0.9138498
## [2,] 0.8648667 0.9207085
```

we notice a great similarity, though the asymptotic confidence interval seems shifted approximately $\cong 0.007$ in comparison to the bootstrap interval.

Exercise 2

We may import the data

```
Data <- read_csv("Data.csv",col_types =
                       cols(col_date(),
                            col_double(),
                            col_double(),
                            col_double(),
                            col_double(),
                            col_double(),
                            col_integer()))
```

```
## Warning: 876 parsing failures.
## row       col expected actual       file
##  32 Open      a double   null 'Data.csv'
##  32 High      a double   null 'Data.csv'
##  32 Low       a double   null 'Data.csv'
##  32 Close     a double   null 'Data.csv'
##  32 Adj Close a double   null 'Data.csv'
## ... ......... ........ ...... ..........
## See problems(...) for more details.
```

```
head(Data, 10)
```

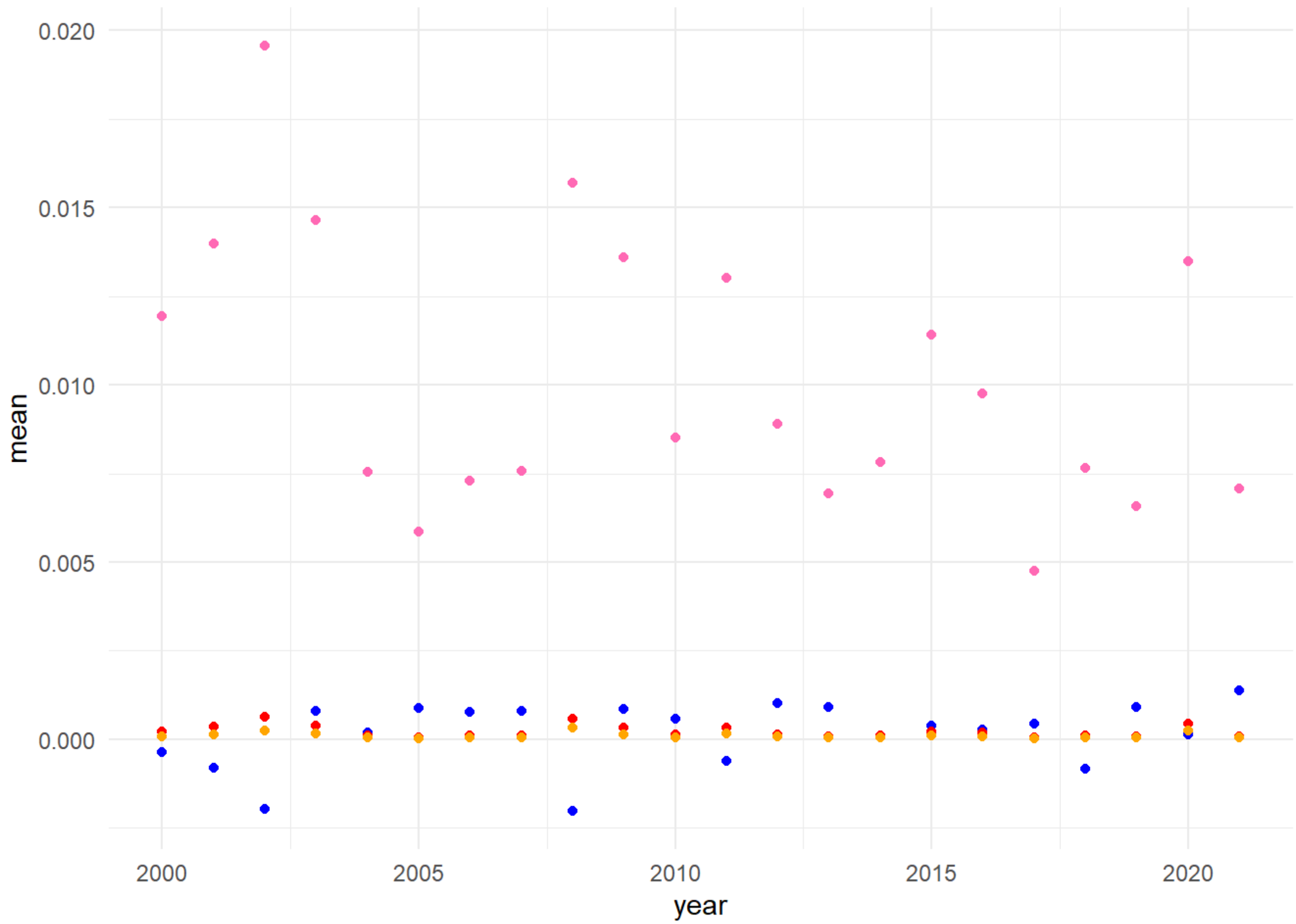
```
## # A tibble: 10 x 7
##    Date        Open  High   Low Close `Adj Close` Volume
##    <date>     <dbl> <dbl> <dbl> <dbl>       <dbl>  <int>
##  1 1990-03-01 1796. 1796. 1796. 1796.       1796.      0
##  2 1990-03-02 1805. 1805. 1805. 1805.       1805.      0
##  3 1990-03-05 1838. 1838. 1838. 1838.       1838.      0
##  4 1990-03-06 1820. 1820. 1820. 1820.       1820.      0
##  5 1990-03-07 1842. 1842. 1842. 1842.       1842.      0
##  6 1990-03-08 1862. 1862. 1862. 1862.       1862.      0
##  7 1990-03-09 1859. 1859. 1859. 1859.       1859.      0
##  8 1990-03-12 1844. 1844. 1844. 1844.       1844.      0
##  9 1990-03-13 1867. 1867. 1867. 1867.       1867.      0
## 10 1990-03-14 1877. 1877. 1877. 1877.       1877.      0
```

We may create a yearly data set, and filter for data after 2000 and remove `NA`'s

```
Data_new <- Data %>% filter(year(Date)>=2000) %>%  mutate(returns = (Close - lag(Close))/lag(Close),
                                                          logreturns = log(1+returns),
                                                          abslogreturn = abs(logreturns),
                                                          absdiff = abs(returns - logreturns))
Data_new_clean <- Data_new %>% na.omit()
Data_new_new <- Data_new %>% group_by(year(Date)) %>% rename('year' = 'year(Date)') %>% summarise(mean = mean(logreturns, n
a.rm = T), var = var(logreturns, na.rm = T), absmean = mean(abs(logreturns), na.rm = T), absvar = var(abs(logreturns), na.rm
= T))
```

We plot these:

```
ggplot(Data_new_new) + geom_point(aes(x=year, y=mean), colour = 'blue') + geom_point(aes(x=year, y=var), colour = 'red') + g
eom_point(aes(x=year, y=absmean), colour = 'hotpink') + geom_point(aes(x=year, y=absvar), colour = 'orange')
```



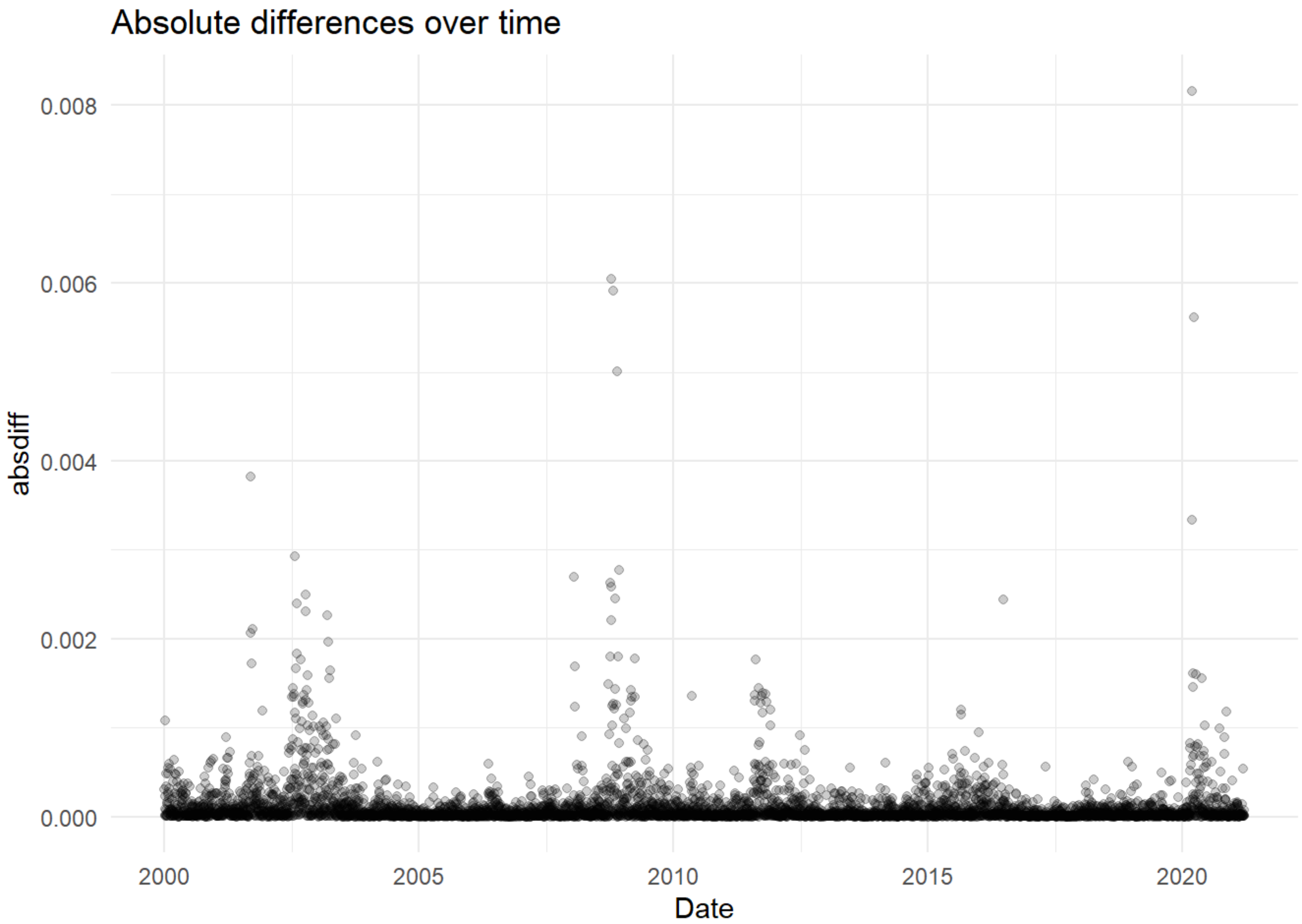
None of the requested quantiles vary a lot, so we cannot reject the possibility of underlying ergodicity.

Exercise 3

a)

We might plot the absolute differences:

```
ggplot(Data_new_clean, aes(x=Date, y=absdiff)) + geom_point(alpha = 0.2) + ggtitle("Absolute differences over time")
```



and calculate the maximum of the absolute difference between returns and logreturns:

```
max(Data_new_clean$absdiff)
```

```
## [1] 0.008162438
```

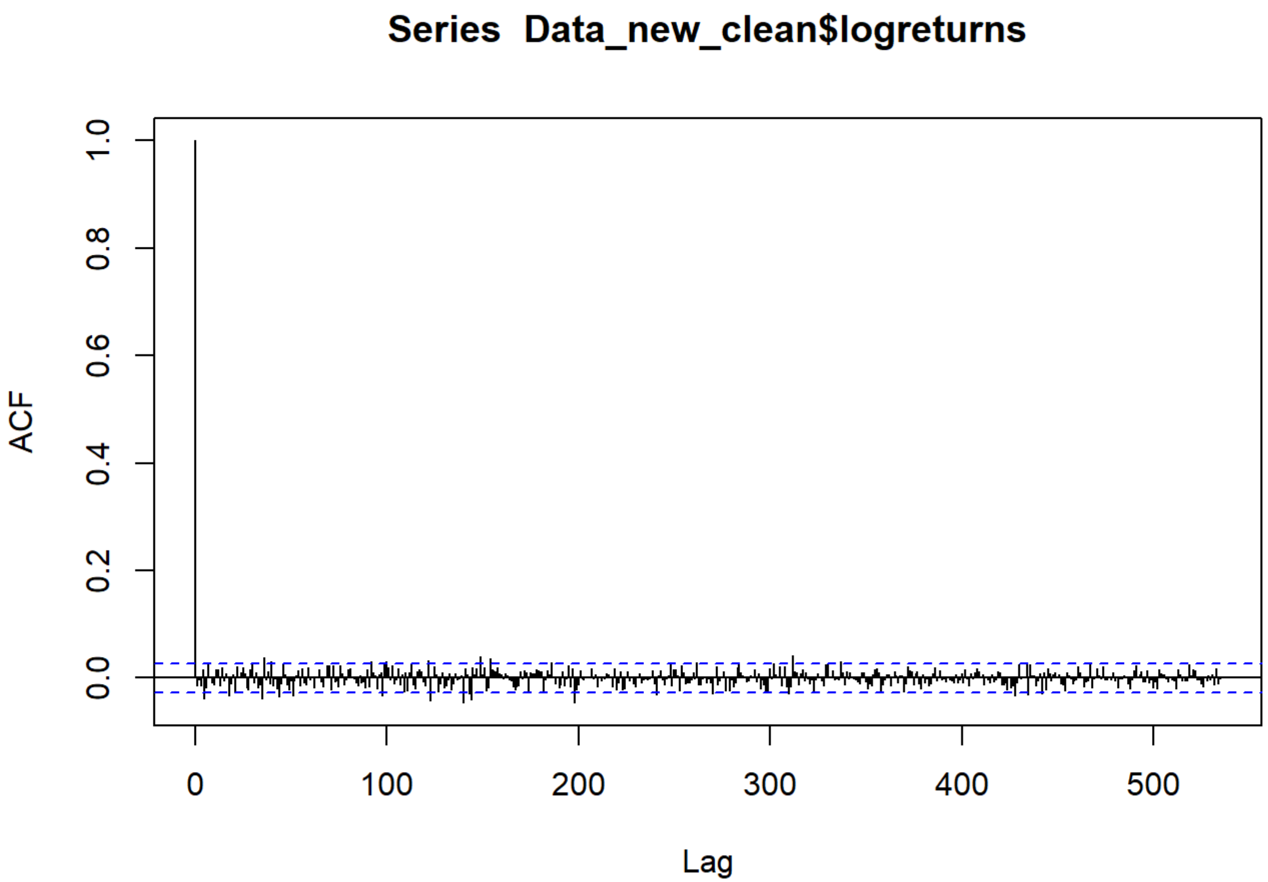
b)

We note that the dataset `Data_new_clean` has `nrow(Data_new_clean) =5353` data points, such that 10% of the sample size of the log return time series will be of the size `floor(nrow(Data_new_clean)/10) =535` . We may use `acf` to calculate this many lags for the sample autocorrelation function for the log-return time series, its absolute value, and its square:

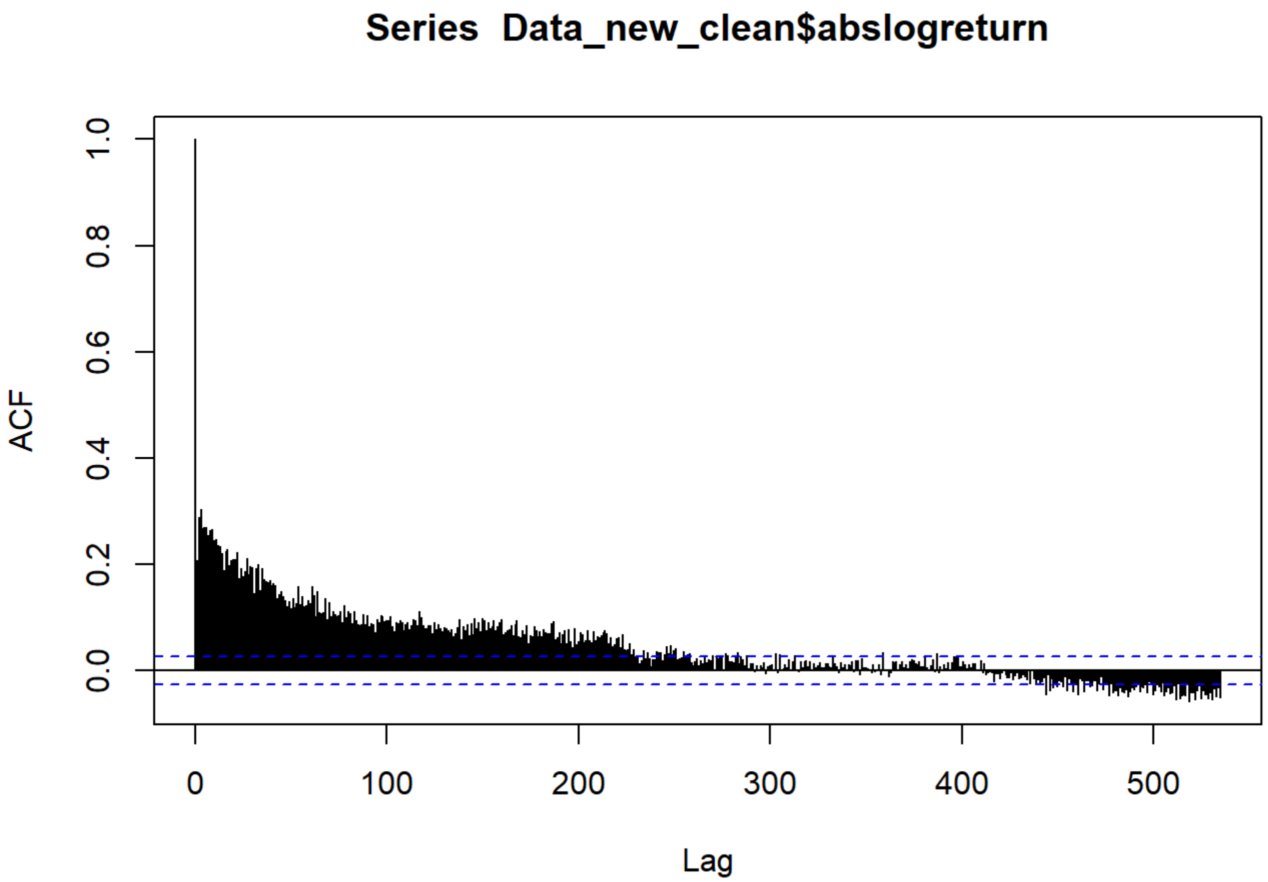
```
autocorLogRet <- acf(Data_new_clean$logreturns, lag.max = floor(nrow(Data_new_clean)/10), plot=F)
autocorLogRetAbs <- acf(Data_new_clean$abslogreturn, lag.max = floor(nrow(Data_new_clean)/10), plot=F)
autocorLogRetSquared <- acf((Data_new_clean$logreturns)^2, lag.max = floor(nrow(Data_new_clean)/10), plot=F)
```

We may also choose to plot each of these:

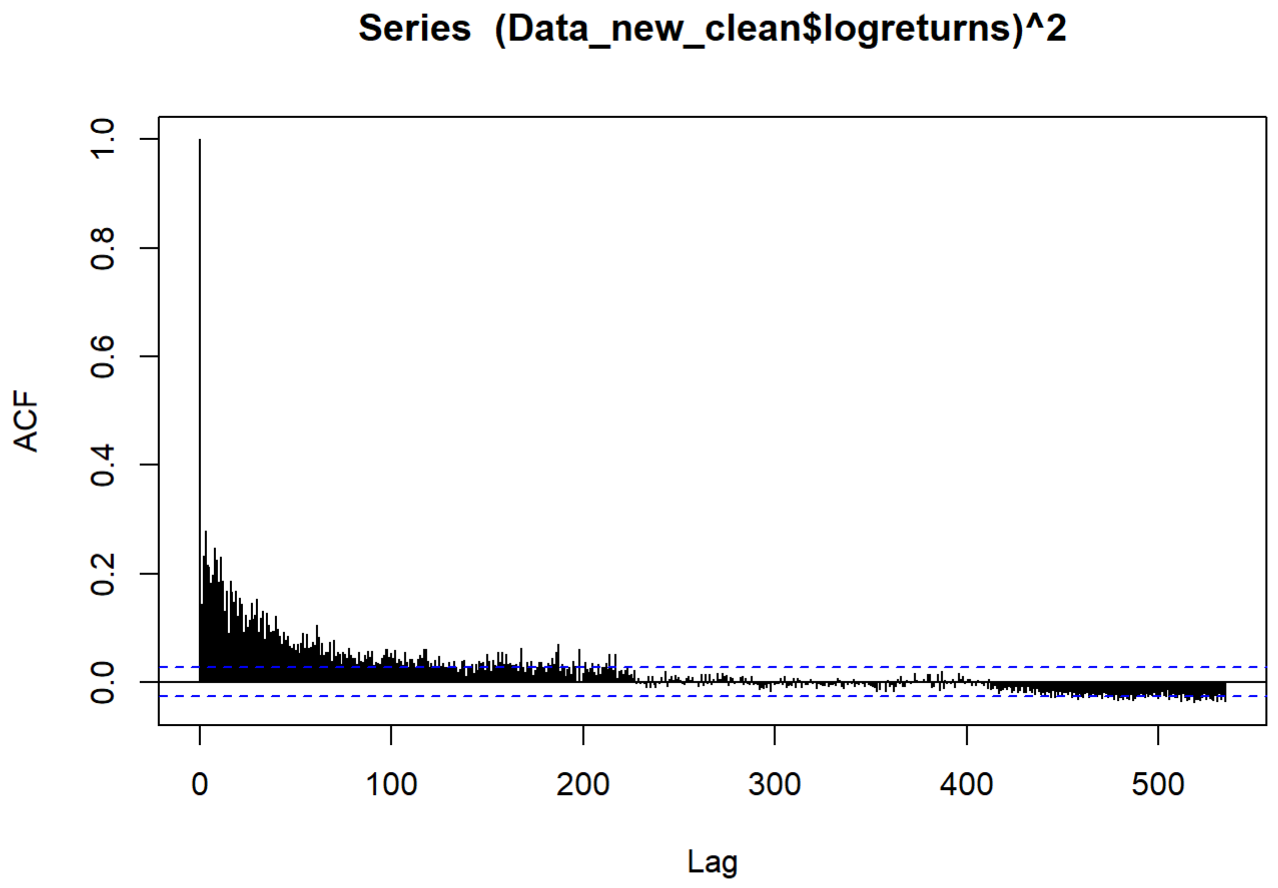
```
acf(Data_new_clean$logreturns, lag.max = floor(nrow(Data_new_clean)/10), plot=T)
```




```
acf(Data_new_clean$abslogreturn, lag.max = floor(nrow(Data_new_clean)/10), plot=T)
```




```
acf((Data_new_clean$logreturns)^2, lag.max = floor(nrow(Data_new_clean)/10), plot=T)
```



c)

We will fit the AR model using `ar.yw`

```
Data_logreturn <- Data_new_clean[,c(1,9)]
modellr <- ar.yw(Data_logreturn, aic = T)
```

We may see a summary of the model:

```
summary(modellr)
```

```
##              Length Class  Mode
## order            1   -none- numeric
## ar               4   -none- numeric
## var.pred         4   -none- numeric
## x.mean           2   -none- numeric
## aic             38   -none- numeric
## n.used           1   -none- numeric
## n.obs            1   -none- numeric
## order.max        1   -none- numeric
## partialacf     148   -none- numeric
## resid        10706   -none- numeric
## method           1   -none- character
## series           1   -none- character
## frequency        1   -none- numeric
## call             3   -none- call
```

d)

We would simulate from the model from c) with the `arima.sim` function, though I've have been unable to do so, as it throws an error, that I've not solved:

```
arima.sim(model = list(ar = modellr), n = nrow(Data_logreturn), rand.gen = rt(n = 1, df = 4))
```

The plotting of the different sample autocorrelations would then have been accomplished based on the simulated data analagously to how it was done in b).

Exercise 4

Not solved

Exercise 5

We may import the dataset:

```
ss_yearly <- sunspot.year
```

a)

Not solved ### b) **Not solved**

c)

Not solved