Homework for week2

徐润奇 222000944

2.2.5 Autocorrelation

```
wave.dat <- "D:\\教学资料\\研究生\\时间序列\\data\\chapter2 wave.dat.txt"
wave <- read.table(wave.dat, header = TRUE)
attach(wave)
ts.plot(waveht)
plot(ts(waveht[1:60]))
#the lag 1 autocorrelation for waveht
acf_1 <- acf(waveht)$acf
acf_2 <- acf(waveht)$acf[2]
plot(waveht[1:396],waveht[2:397])
```

Function attach()

Sometimes we may use **attach()** function to search the data from a data-frame if we use it frequently.

Let's take an example:

```
# Clear the working environment
rm(list=ls(all=TRUE))
# Define a dataframe
(Data <- data.frame(a = 1:5, b = 6:10))
  a  b
1 1  6
2 2  7
3 3  8
4 4  9
5 5 10
# Define variables out of dataframe
> (a <- 11:15)
[1] 11 12 13 14 15
> (c <- 16:20)
[1] 16 17 18 19 20
```

Here we created a special situation: Inside matrix Data we have a list a , but we also define a variable a outside in the global environment. So what should we do to search these 2 'a' if we want to use them?

First we want to search b:

```
> b
Error: Cannot find the object 'b'
```

We can see we cannot find object b directly if it's inside a data-frame.

Then we applying $attach(data)$:

```
> attach(Data)
The following object is masked _by_ .GlobalEnv:
    a
> a
[1] 11 12 13 14 15
> b
[1]  6  7  8  9 10
```

Here we can see: after $attach()$, we can find variable b directly.

But for our 'special' a, it has 2 'status'. We can see the **priority** of recall a is global environment. If we want to use a inside the data-frame., we shall use $data(a)$:

```
# Comparison
> a
[1] 11 12 13 14 15
> Data$a
[1] 1 2 3 4 5
```

And we can use $detach()$ to turn back original situation.

Warning ! ! ! :

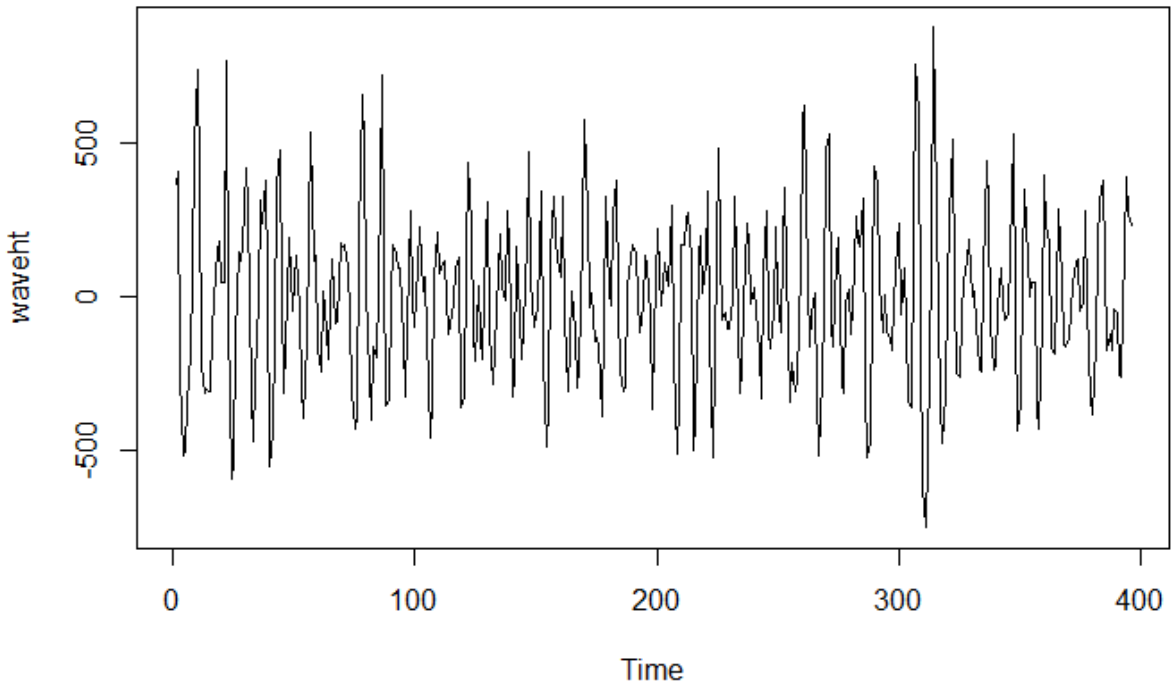
We may meet some troubles here:

```
The following object is masked from wave (pos = 3):

    waveht

The following object is masked from wave (pos = 4):

    waveht
```

Even if we could run the code and give out the result, the default won't disappear.

The help page for `attach()` notes that *attach can lead to confusion*. The Google R Style Manual provides clear advice on this point, providing the following advice about `attach()`: *The possibilities for creating errors when using attach are numerous. Avoid it*

By applying $attach()$, then we plot the variable $waveht$:



Function acf()

Function $acf()$ is used to find the autocorrrelation for our data.

$acf()$ is defined as :

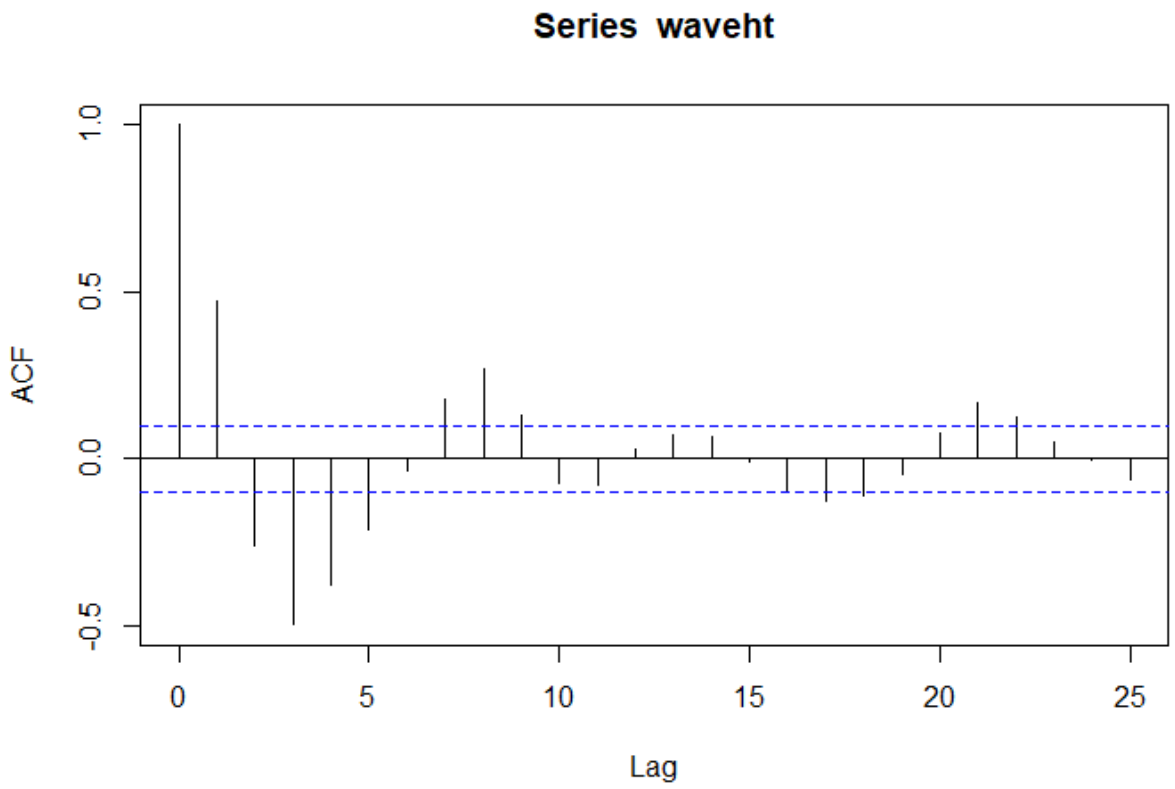
```
acf(x,type=c("correlation","covariance","partial"),lag.max=n)
```

x is the input data, $type$ is the type of autocorrelations. $lag.x$

Define a variable acf_1, such that: The maximum lag order, n determines the length of abscissa in acf graph.

```
acf_1 <- acf(waveht)$acf
```

```
> acf_1
, , 1

              [,1]
 [1,]  1.000000000
 [2,]  0.470256396
 [3,] -0.262911528
 [4,] -0.498917020
 [5,] -0.378706643
 [6,] -0.214992933
```



Since nearly excpect around origin, nearly all the bars lies inside the blue line, it means no obvious correlation for this group of data.

By the way, we can use the command:

```
acf_2 <- acf(waveht)$acf[2]
# output will be 0.47
```

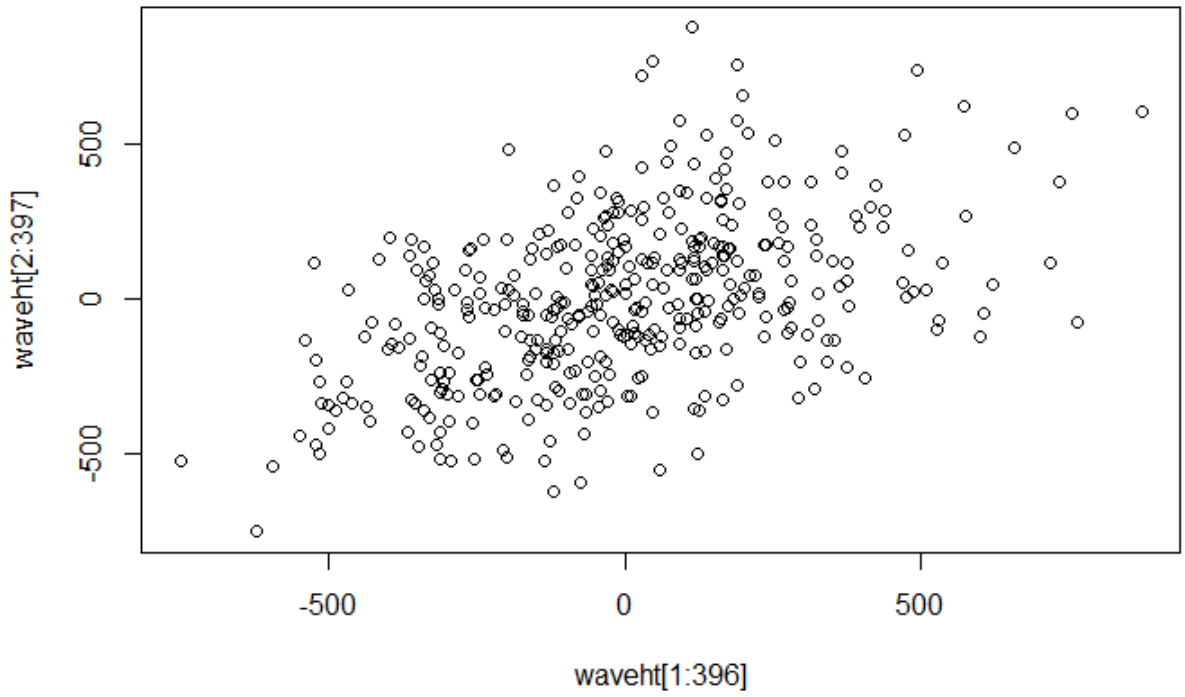
To check each term in the time series.

lagn scatterplot

If we want to find the autocorrelation directly by figure, we can

```
plot(waveht[1:396],waveht[2:397])
```

The figure shows the raltion between a term with next term.

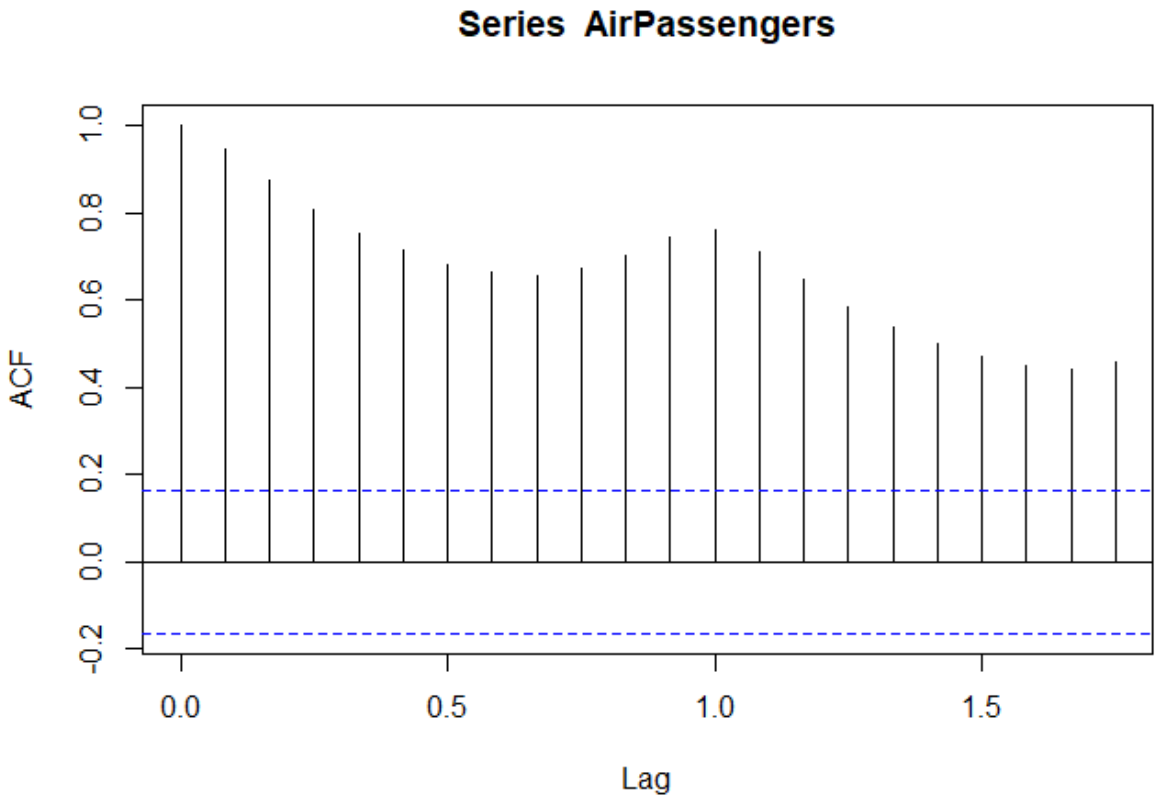


We can see roughly it has no obvious autocorrelation.

2.3.2 examples based on air passenger series

```
acf(AirPassengers)
data(AirPassengers)
AP <- AirPassengers
AP.decom <- decompose(AP, "multiplicative")
class(AP.decom$random[7:138])
length(AP)
plot(ts(c))
acf(AP.decom$random[7:138])
sd(AP[7:138])
sd(AP[7:138] - AP.decom$trend[7:138])
sd(AP.decom$random[7:138])
```

First of all we draw a acf figure of data:



We can see here is a strong effect of autocorrelations.

Compare data AP wtih AP eliminated trend:

```
> sd(AP[7:138])
[1] 109.4187
> sd(AP[7:138] - AP.decom$trend[7:138])
[1] 41.11491
```

We can see after substract the trend, standard diviation cut down obviouly.

Page 42.2 Vineyard

2. The following data are the volumes, relative to nominal contents of 750 ml, of 16 bottles taken consecutively from the filling machine at the Serendipity Shiraz vineyard:

$$39, 35, 16, 18, 7, 22, 13, 18, 20, 9, -12, -11, -19, -9, -2, 16.$$

The following are the volumes, relative to nominal contents of 750 ml, of consecutive bottles taken from the filling machine at the Cagey Chardonnay vineyard:

$$47, -26, 42, -10, 27, -8, 16, 6, -1, 25, 11, 1, 25, 7, -5, 3$$

The data are also available from the website in the file `ch2ex2.dat`.

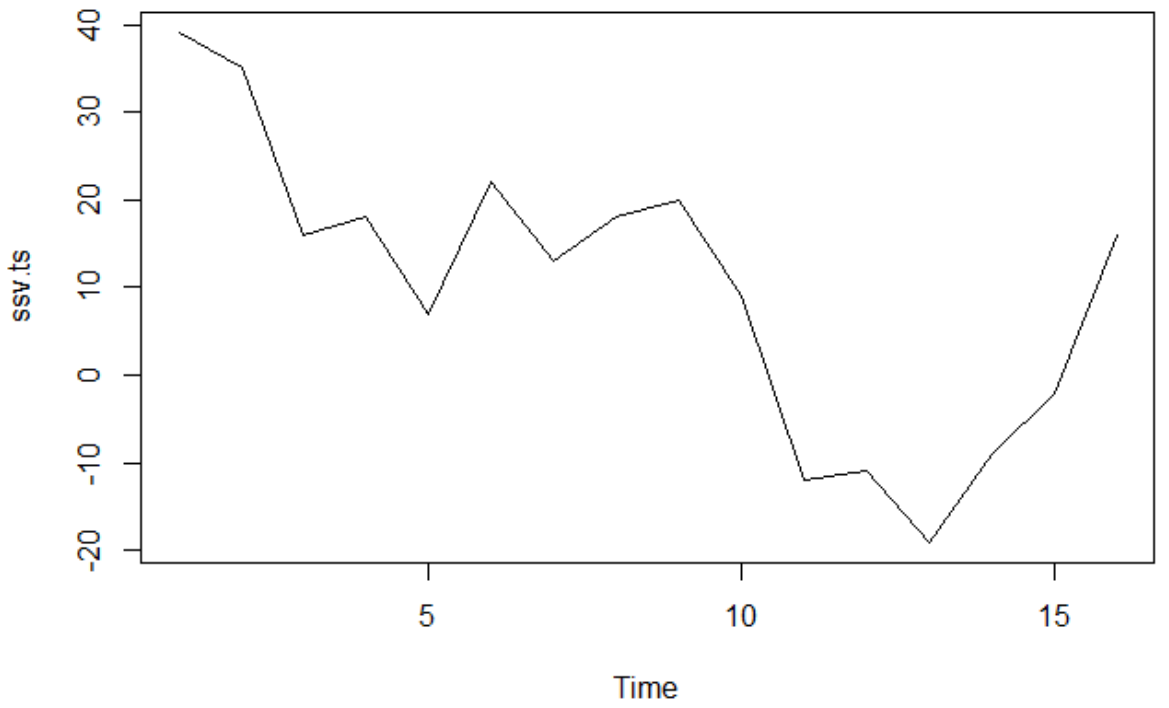
a) Produce time plots of the two time series.
b) For each time series, draw a lag 1 scatter plot.
c) Produce the acf for both time series and comment.

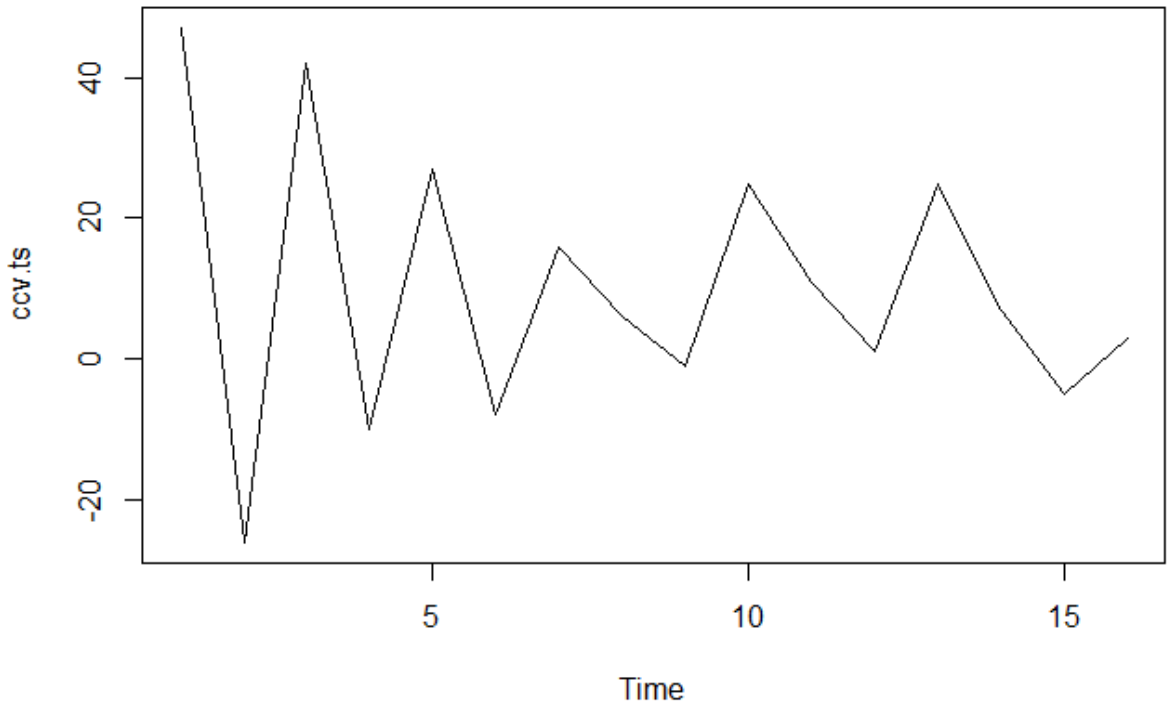
```
ssv <- matrix(c(39, 35, 16, 18, 7, 22, 13, 18, 20, 9, -12, -11, -19, -9, -2,
16))
colnames(ssv) <- "ssv"
ccv <- matrix(c(47, -26, 42, -10, 27, -8, 16, 6, -1, 25, 11, 1, 25, 7, -5, 3))
```

```
colnames(ccv) <- "ccv"
# Produce time plots of the two time series
ssv.ts <- ts(ssv)
ccv.ts <- ts(ccv)
ts.plot(ssv.ts)
ts.plot(ccv.ts)
# For each time series, draw a lag 1 scatter plot.
plot(ssv.ts[1:15],ssv.ts[2:16])
plot(ccv.ts[1:15],ccv.ts[2:16])
# Produce the acf for both time series and comment
acf_ssv <- acf(ssv.ts)$acf
acf_ccv <- acf(ccv.ts)$acf
```

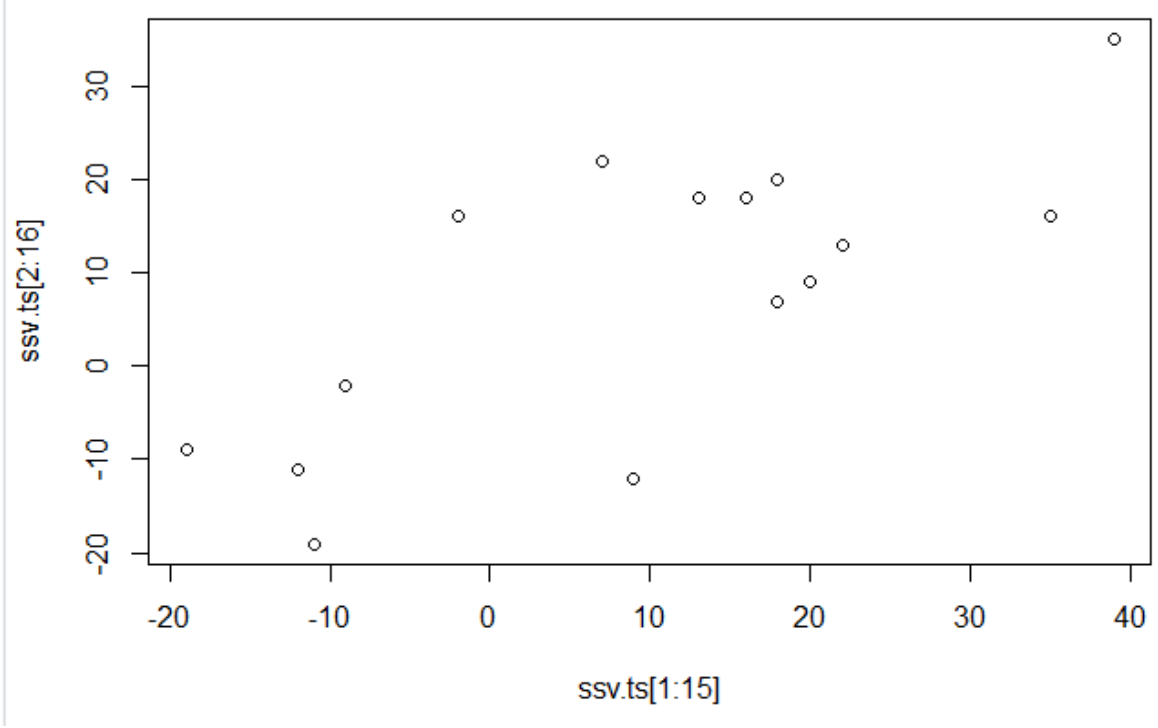
First we input the data, save the data from 2 different vinyards as ssv and ccv.

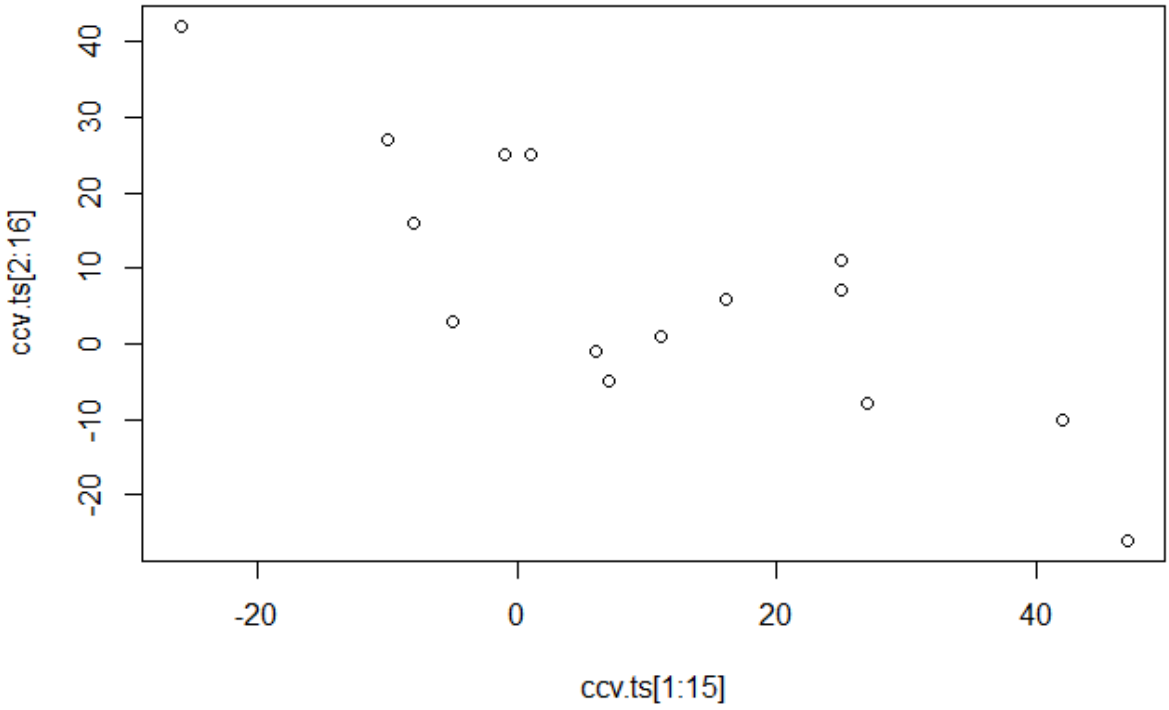
Plot the figure by using command ts.plot, we get:



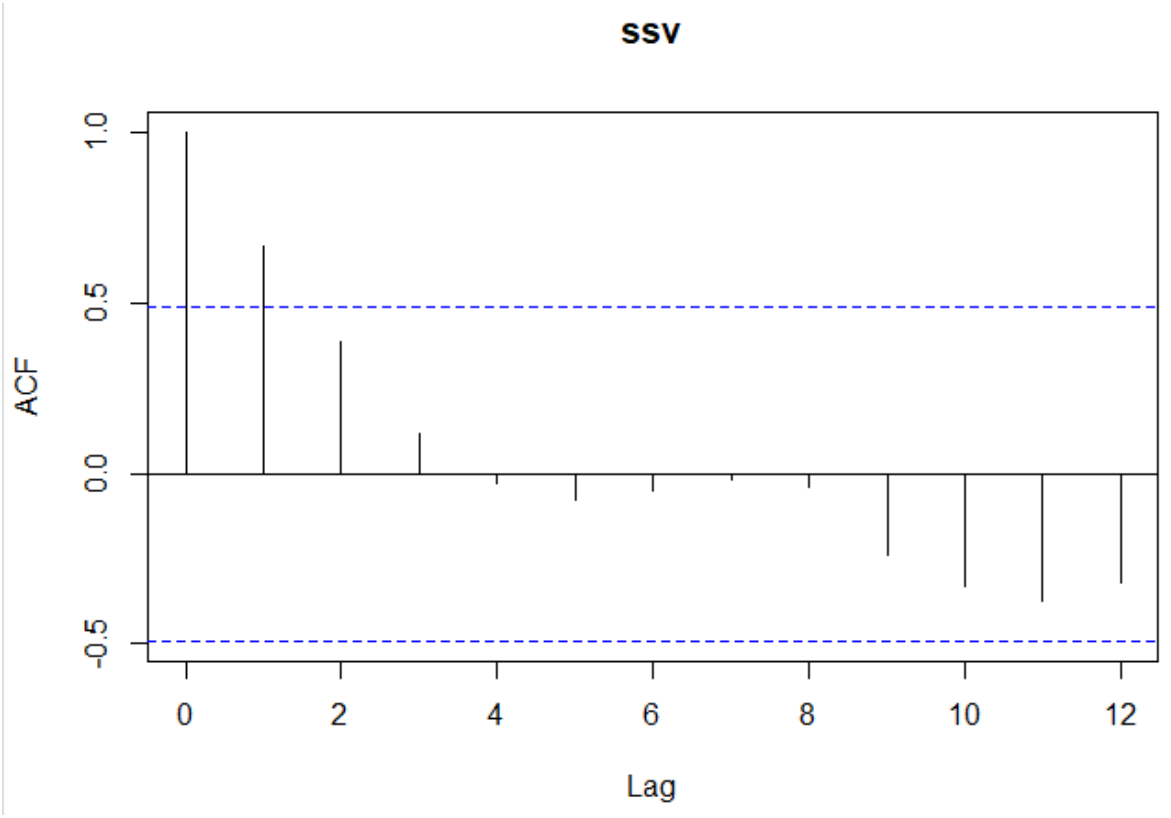


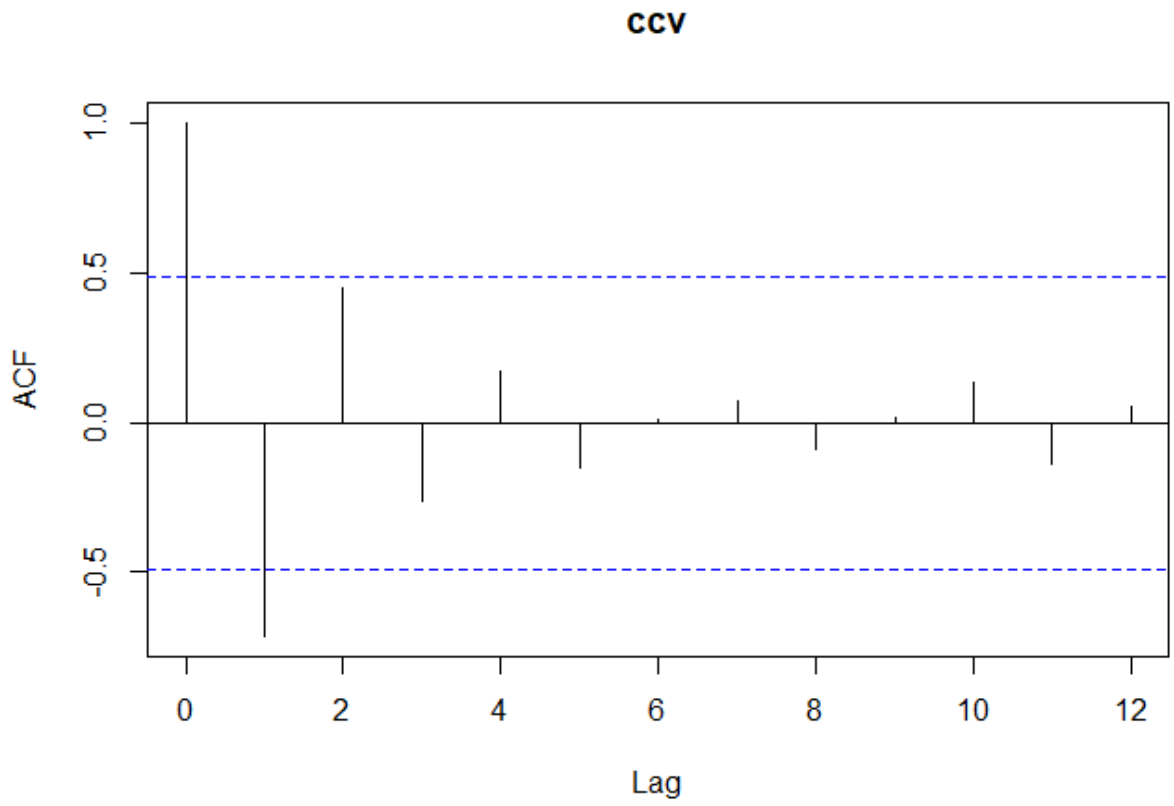
For each time series, we draw a scatter plot as below:





Produce the acf for both time series we get :





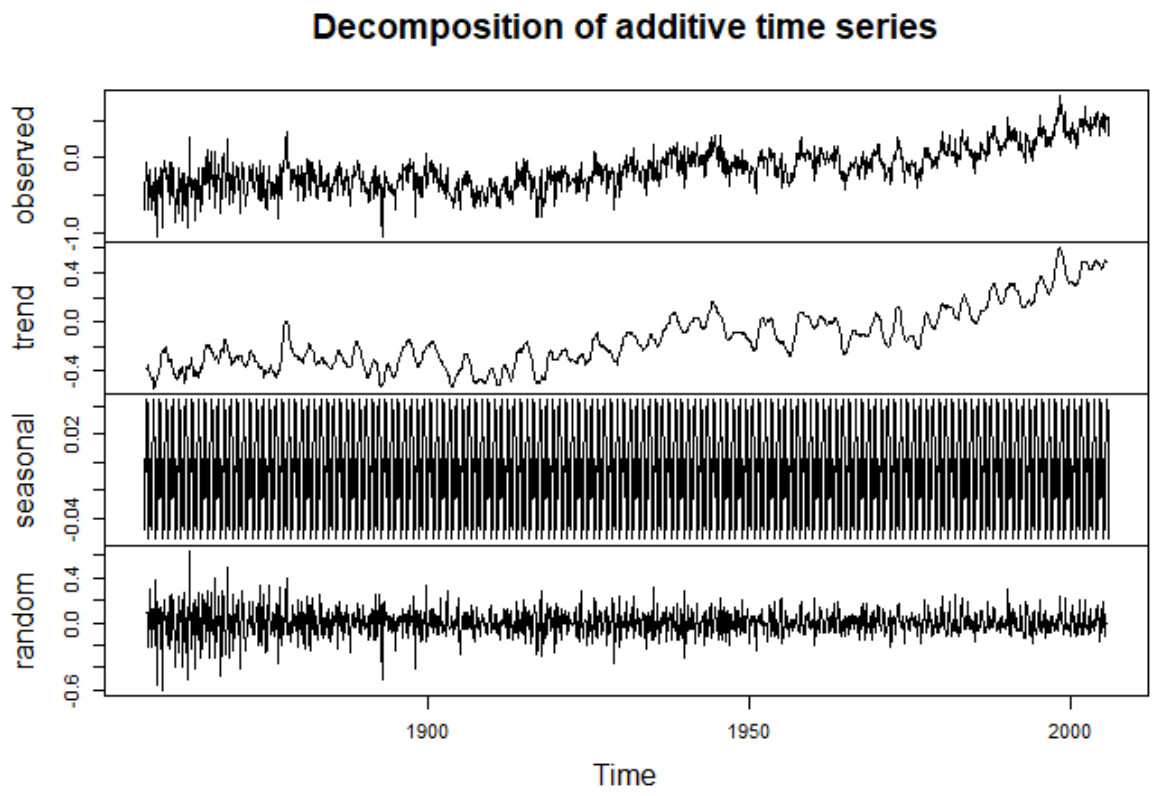
We can see both for SSV and CCV, we have no obvious autocorrelation for them.

Page 42.3 global temperature

3. Carry out the following exploratory time series analysis using the global temperature series from §1.4.5.
   a) Decompose the series into the components trend, seasonal effect, and residuals. Plot these components. Would you expect these data to have a substantial seasonal component? Compare the standard deviation of the original series with the deseasonalised series. Produce a plot of the trend with a superimposed seasonal effect.
   b) Plot the correlogram of the residuals (random component) from part (a). Comment on the plot, with particular reference to any statistically significant correlations.

```
www <- "D://教学资料//研究生//时间序列//data//chapter1 global.txt"
global <- scan(www)
global.ts <- ts(global, st = c(1856, 1), end = c(2005, 12),fr = 12)
# Decompose the series
global.decom= decompose(global.ts)
plot(global.decom)

#Produce a plot of the trend with a superimposed seasonal effect.
global.deseasonal <- global.ts - global.decom$seasonal
ts.plot(cbind(global.deseasonal,global.deseasonal+global.ts),lty =1:2)

#Compare the standard deviation of the original series with the deseasonalised
series
sd(global.ts)
sd(global.deseasonal)


#Plot the correlogram of the residuals (random component)
global.random <- global.decom$random
acf_random <- acf(global.random,na.action = na.pass)$acf
```

First of all for this question we decompose the data, it give us the figure:



We can see after eliminating the random, seasonal, trend effect, the trend of temperature is rising in a steady pace.

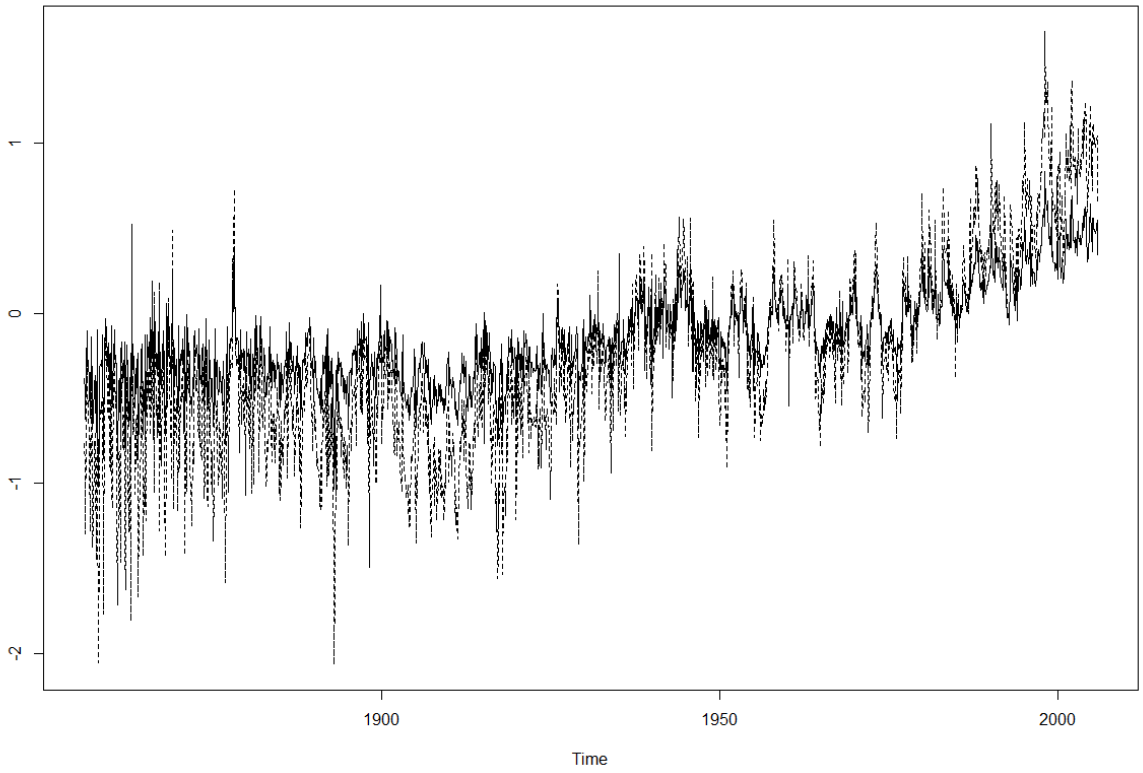
As we expect, this series of data will be effectted by the seasonal effects.

Compare the standard deviation of the original series with the deseasonalised series, we get:

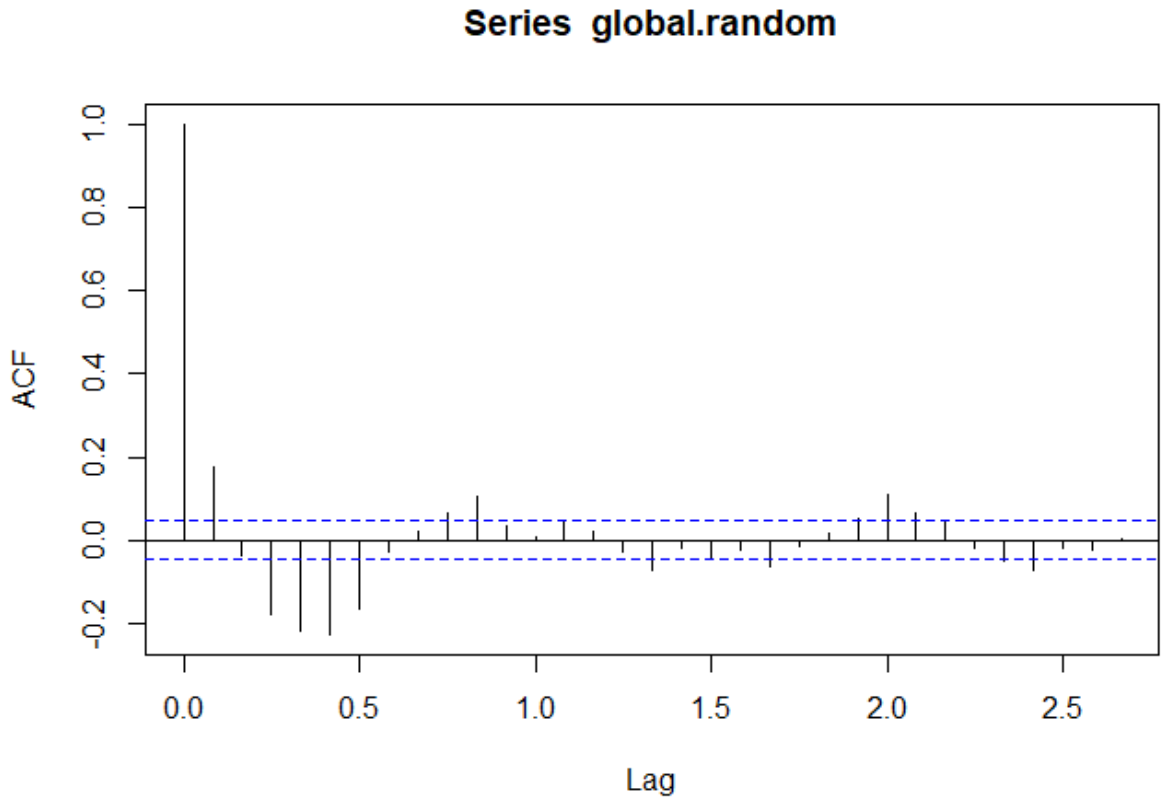
```
> sd(global.ts)
[1] 0.273536
> sd(global.deseasonal)
[1] 0.2715033
```

It obviously has difference for deviation.

Then we produce a plot of the trend with a superimposed seasonal effect:



We use acf function to plot the correlogram of the residuals (random component) :



We can see no such obvious autocorrelations for the random term in this case.

42.5

5. a) Prove Equation (2.15), using the following properties of summation, expectation, and covariance:

$$\sum_{i=1}^{n} x_i \sum_{j=1}^{m} y_j = \sum_{i=1}^{n} \sum_{j=1}^{m} x_i y_j$$
$$E\left[\sum_{i=1}^{n} x_i\right] = \sum_{i=1}^{n} E(x_i)$$
$$\mathrm{Cov}(x, y) = E(xy) - E(x)\,E(y)$$

$$\mathrm{Cov}\left(\sum_{i=1}^{n} x_i, \sum_{j=1}^{m} y_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{m} \mathrm{Cov}(x_i, y_j) \qquad (2.15)$$

proof

Proof :

$$\mathrm{cov}\left(\sum x_i, \sum y_i\right)$$

$$= E\left(\left[\sum x_i - E\left(\sum x_i\right)\right]\left[\sum y_j - E\left(\sum y_j\right)\right]\right)$$

$$= E\left(\sum x_i \sum y_j\right) - E\left(\sum x_i\right) E\left(\sum y_j\right)$$

$$= E\left(\sum x_i y_j\right) - E\left(\sum x_i\right) E\left(\sum y_j\right)$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{m} E(x_i y_j) - \sum_{i=1}^{n}\sum_{j=1}^{m} E(x_i)\,E(y_j)$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{m}\left[E(x_i y_j) - E(x_i)\,E(y_i)\right]$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{m} \mathrm{cov}(x_i, y_j) \quad \square$$