Time Series - Computer lab A

Group 6

7-9-2019

Contents


```
library(latex2exp)
library(tseries)
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

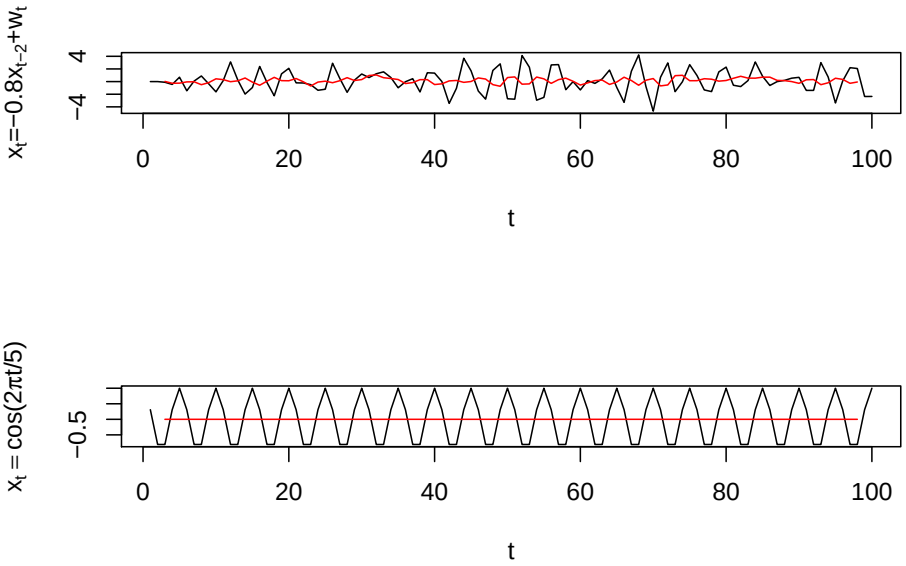
```
library(MASS)
library(astsa)
library(ggplot2)
library(birk)

rm(list=ls())
set.seed(12345)
```

Assignment 1

a)

Generate two time series $x_t = -0.8x_{t-2} + w_t$, where $x_1 = x_0 = 0$ and $x_t = \cos(2\pi t/5)$ with 100 observations each. Apply a smoothing filter $v_t = 0.2(x_t + x_{t-1} + x_{t-2} + x_{t-3} + x_{t-4})$ to these two series and compare how the filter has affected them.

```
# filter0 <- function(xs){
#   vs <- c()
#   for (i in 5:length(xs)) {
#     vs <- c(vs, 0.2*(xs[i]+xs[i-1]+xs[i-2]+xs[i-3]+xs[i-4]))
#   }
#   vs
# }

xs1 <- c(0, 0)
n   <- 100
ws  <- rnorm(n)
for (i in 3:n) {
 xs1 <- c(xs1, -0.8*xs1[i-2] + ws[i])
}
vs1 <-stats::filter(xs1, filter = rep(0.2, 5))

xs2 <- c()
for (i in 1:n) {
  xs2 <- c(xs2, cos((2*pi*i)/5))
}
vs2 <- stats::filter(xs2, filter = rep(0.2, 5))
```



The black lines in both plots are observations of time series and the red lines are filtered observations. Filter can remove the uncorrelated noise from time series. We can observe that the red lines become smoother after filtering. Especially for $x_t = \cos(2\pi t/5)$ since there is no relation among observations for such series, the line becomes a straight line after filtering.

b)

Consider time series $x_t - 4x_{t-1} + 2x_{t-2} + x_{t-5} = w_t + 3w_{t-2} + w_{t-4} - 4w_{t-6}$. Write an appropriate R code to investigate whether this time series is casual and invertible.

ARMA(p,q) model can be written as

$$x_t = c + w_t + \sum_i \phi(B)x_{t-i} + \sum_i \theta(B)w_{t-i} \quad ,$$

so our time series can also be re-written as

$$x_t = w_t + 4x_{t-1} - 2x_{t-2} - x_{t-5} + 3x_{t-2} + w_{t-4} - 4w_{t-6}.$$

```
# Causality
z <- c(1,4,-2,0,0,-1)
root <- polyroot(z)
abs(root)>1
```

```
## [1] FALSE  TRUE  TRUE  TRUE  TRUE
```

```
# Invertibility
z <- c(1,0,3,0,1,0,-4)
root <- polyroot(z)
abs(root)>1
```

```
## [1] FALSE FALSE FALSE FALSE  TRUE  TRUE
```

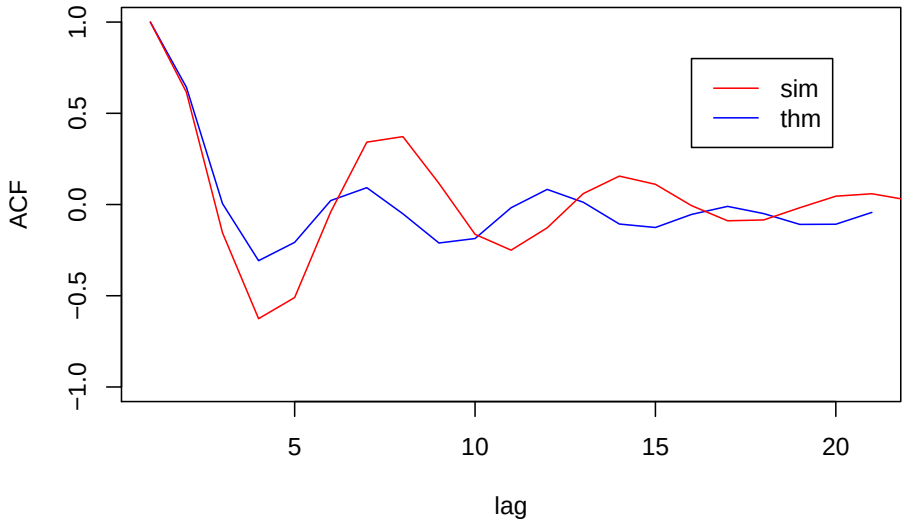
Since not all the modules of roots of $\phi(z') = 0$ and $\theta(z') = 0$ are larger than 1, we can conclude that this series is not casual or invertible.

c)

Use built-in R functions to simulate 100 observations from the process $x_t + 3/4x_{t-1} = w_t - 1/9w_{t-2}$, compute sample ACF and theoretical ACF, use seed 54321. Compare the ACF plots.

```
set.seed(54321)
# sample ACF
sim <- arima.sim(n=100, model=list(ar=c(1, -3/4), ma=c(1,0,-1/9)))
acf.sim <- acf(sim, plot = F)

# theoretical ACF
acf.thm <- ARMAacf(ar=c(1, -3/4), ma=c(1,0,-1/9), lag.max = 21)
```

```
plot(acf.sim$acf, type="l", ylim=c(-1,1), col="blue", xlab="lag", ylab="ACF")
lines(acf.thm, col="red")
legend(16,0.8,legend=c("sim","thm"), col=c("red","blue"),lty=1)
```



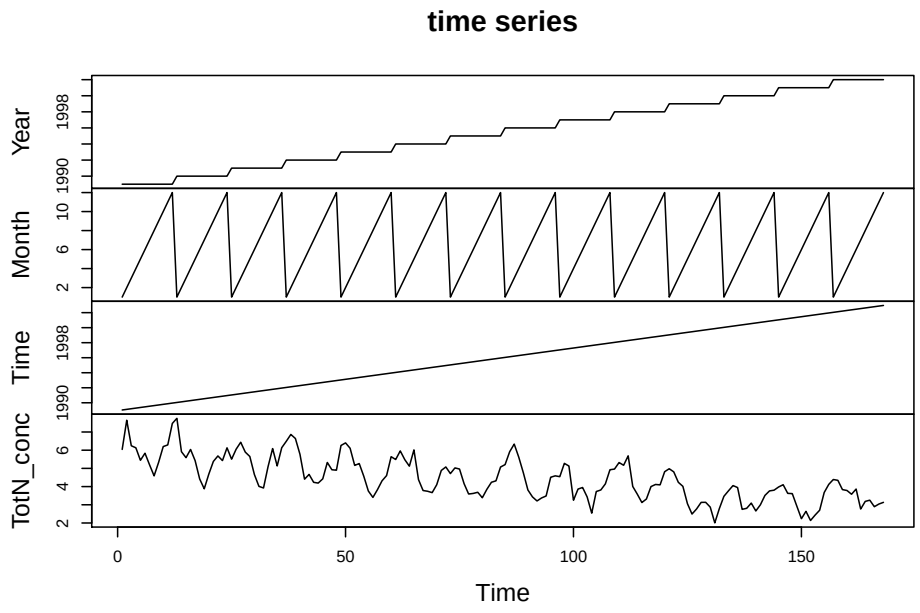
Although the ACF from simulated series are much rougher compared with the theoretical ACF, their trends are quite similar and converge to zero respectively.

Assignment 2

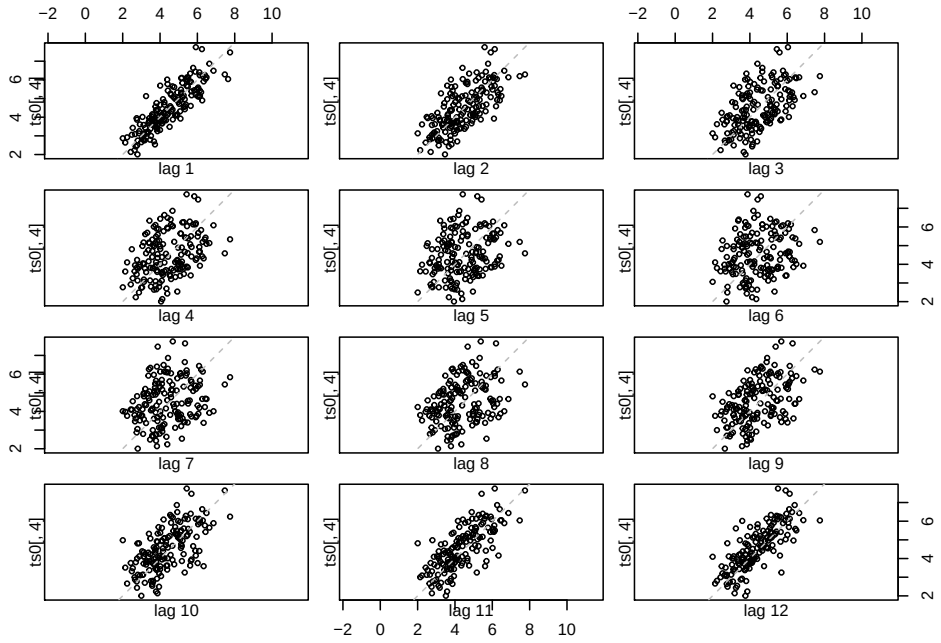
a)

Import the data to R, convert it appropriately to ts object (use function ts()) and explore it by plotting the time series, creating scatter plots of x_t against $x_{t-1}, ..., x_{t-12}$. Analyze the time series plot and the scatter plots? Are there any trends, linear or seasonal, in the time series? When during the year is the concentration highest? Are there any special patterns in the data or scatter plots? Does the variance seem to change over time? Which variables in the scatter plots seem to have a significant relation to each other?

```
rm(list=ls())
rhine <- read.csv2("Rhine.csv")
ts0 <- ts(rhine)

plot(ts0, main="time series")
```




```
lag.plot(ts0[,4], lag=12)
```



There is an obvious seasonal trend in the whole series based on `month`.

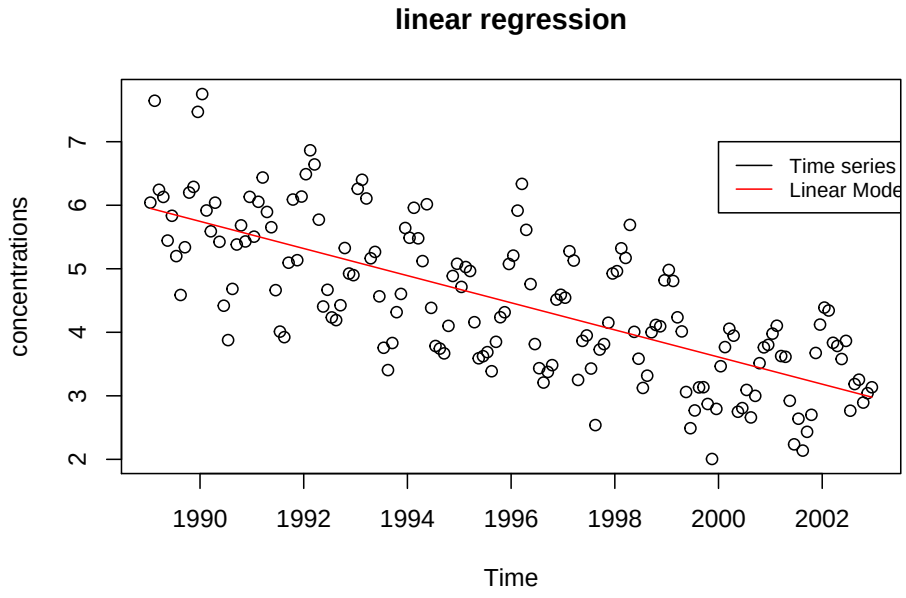
The lag scatter plots show that the variance is change seasonally as well. The variance of differences would increase and be largest when lag is around 6, and decrease until lag is 12. Thus, variable `month` might be the most significiant relationship for all observations.

b)

Eliminate the trend by fitting a linear model with respect to t to the time series. Is there a significant time trend? Look at the residual pattern and the sample ACF of the residuals and comment how this pattern might be related to seasonality of the series.

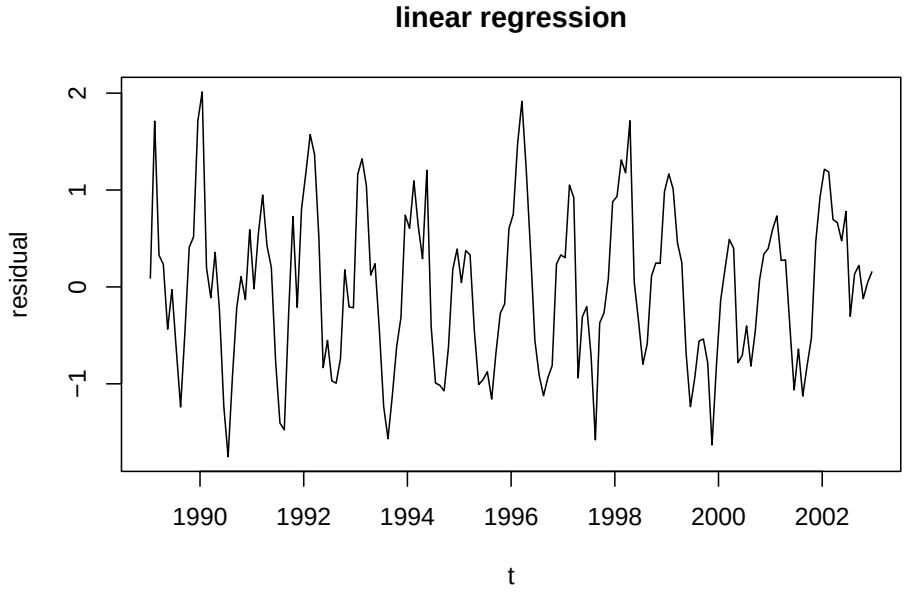
```
mdl <- lm(TotN_conc~Time, rhine)
```

```
plot(rhine$Time, rhine$TotN_conc, xlab="Time", ylab="concentrations", main = "linear regression")
lines(rhine$Time,mdl$fitted.values, col="red")
legend(2000,7, legend = c("Time series" , "Linear Model"),
       col = c("black","red"),lty = 1, cex=0.8)
```

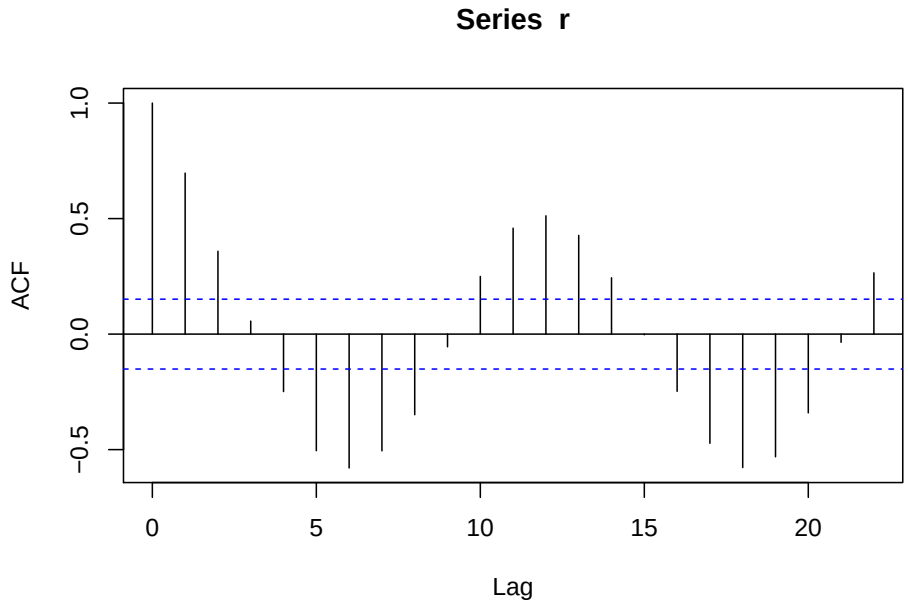


The plot shows an evident time trend against the concentrations of total nitrogen in the Rhine River. We use residuals to detrend.

```
r <- mdl$residuals
plot(y=r,x=rhine$Time, type="l", xlab="t", ylab = "residual", main = "linear regression")
```




```
acf(r)
```



The residual pattern shows that the residual changes seasonally and be lowest in Summer and highest in Winter. It seems that there is a negative correlation among the residuals during Summer, since the residuals vary slightly and the residuals could not increase or decrease fast. On the contrast, there might be a positive correlation among residuals during Winter, because the residuals change widely.

The ACF of residuals shows that the concentrations of total nitrogen have a positive correlation with the closest 3 months, and a negative correlation with the other month, which confirms our previous inference. Since there might be a periodic trend in the residual pattern, it is preferable to assume that such residual series is not stationary.

c)

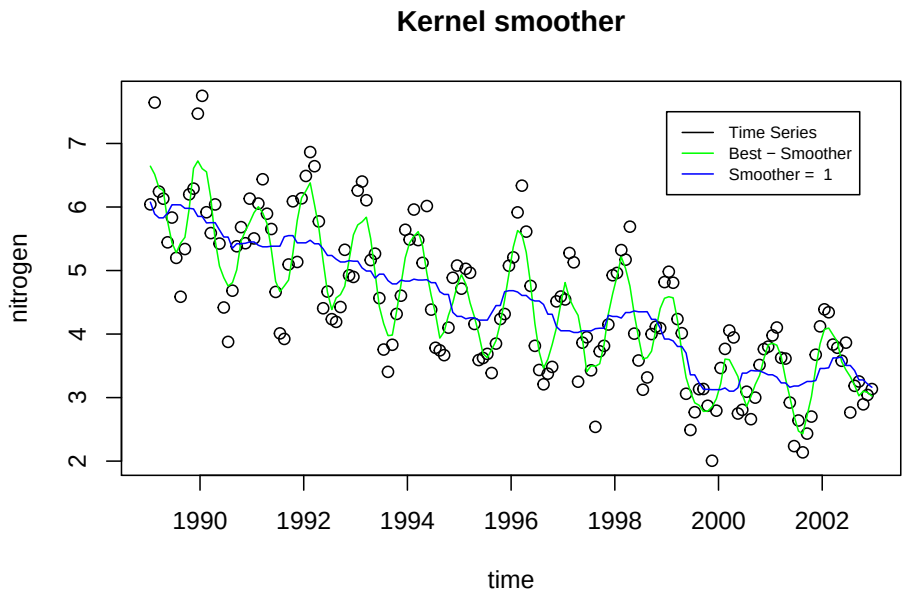
Eliminate the trend by fitting a kernel smoother with respect to t to the time series (choose a reasonable bandwidth yourself so the fit looks reasonable). Analyze the residual pattern and the sample ACF of the residuals and compare it to the ACF from step b). Conclusions? Do residuals seem to represent a stationary series?

```
# fit kernel smoother

# choose a reasonable bandwidth yourself
bandwidth_values = seq(from = 0.2, to = 1, by = 0.001)

# fit kernel smoother with different bandwidth_values

# save the values in matrix with a row for residuals and columns for bandwidth values
residuals = vector("numeric", length = length(bandwidth_values))
counter = 0
for (i in bandwidth_values) {

  counter  = counter +1

  # fit the smoother
  kernel_model = ksmooth(x = rhine[,3], y = rhine[,4], # x = time & y = nitrogen
```

```
                        bandwidth = i)

  # compute the residuals
  residuals[counter] = sum(rhine[,4] - kernel_model$y)
}

# find best residuals
best_bandwidth_index = which.closest(residuals, 0)
best_bandwidth_value = bandwidth_values[best_bandwidth_index]

# fit the best kernel_model
kernel_model_best = ksmooth(x = rhine[,3], y = rhine[,4], # x = time & y = nitrogen
                        bandwidth = best_bandwidth_value)

# fit a bad bandwidth value
kernel_model_worst = ksmooth(x = rhine[,3], y = rhine[,4], # x = time & y = nitrogen
                        bandwidth = 1)
```

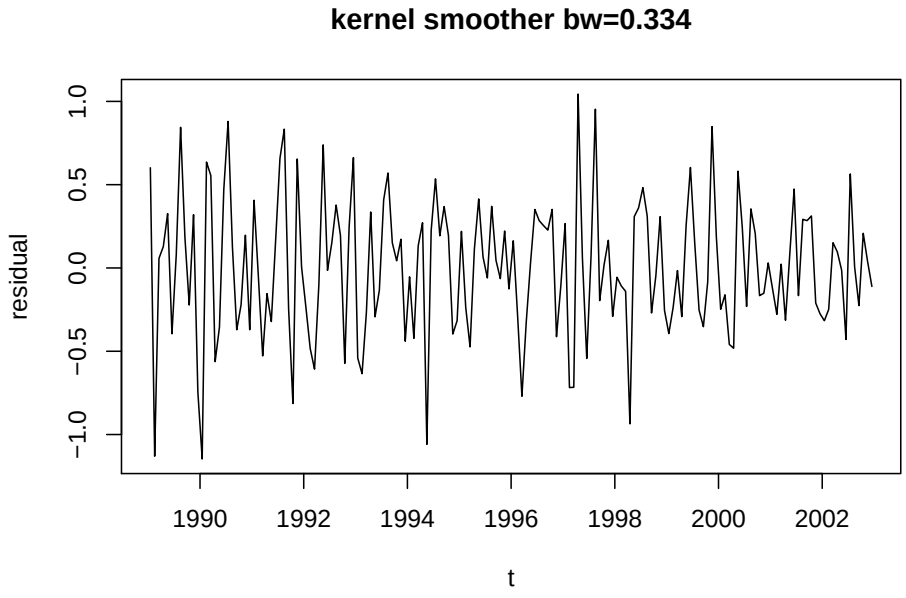
The best smoothing parameter is:

```
## [1] 0.334
```

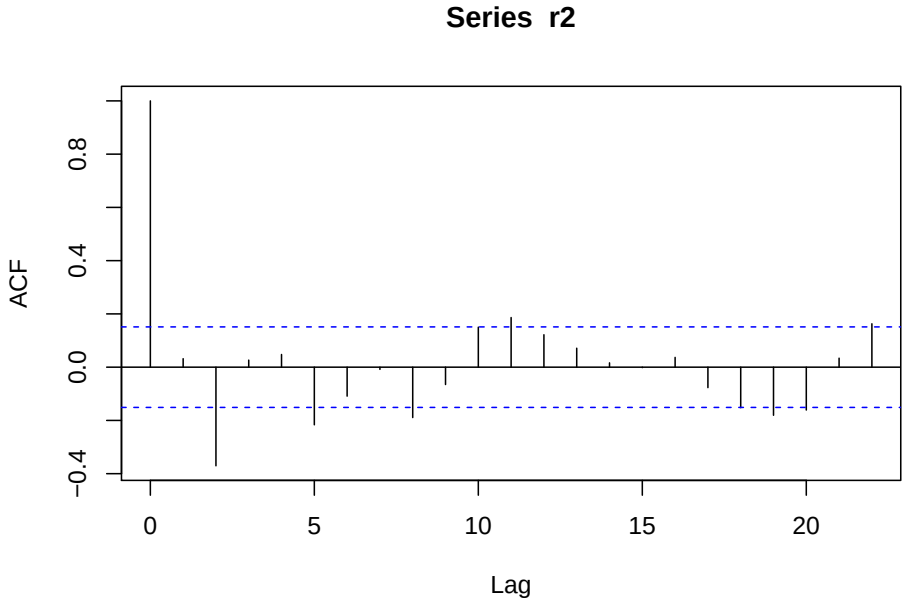


For this I searched for the best smoother between the values of 0.2 and 1 with step 0.001. If we set the smoother to a value between 0 and 0.1, the result would be a smoother that perfectly fits the course of nitrogen. Furthermore, I chose an extreme value of 1 to compare these smoother with each other.

```
r2 <- kernel_model_best$y-rhine$TotN_conc
plot(y=r2, x=rhine$Time, type="l", xlab="t", ylab="residual", main = "kernel smoother bw=0.334")
```




```
acf(r2)
```



The residual pattern of kernel smoother is much rougher than the one with linear regression. It seems that there is no correlation among the residuals.

According to ACF plot, the ACF decays when $lag \neq 0$. There is no obvious correlation among residuals, which confirms our results. It is possible that such residuals is stationary.

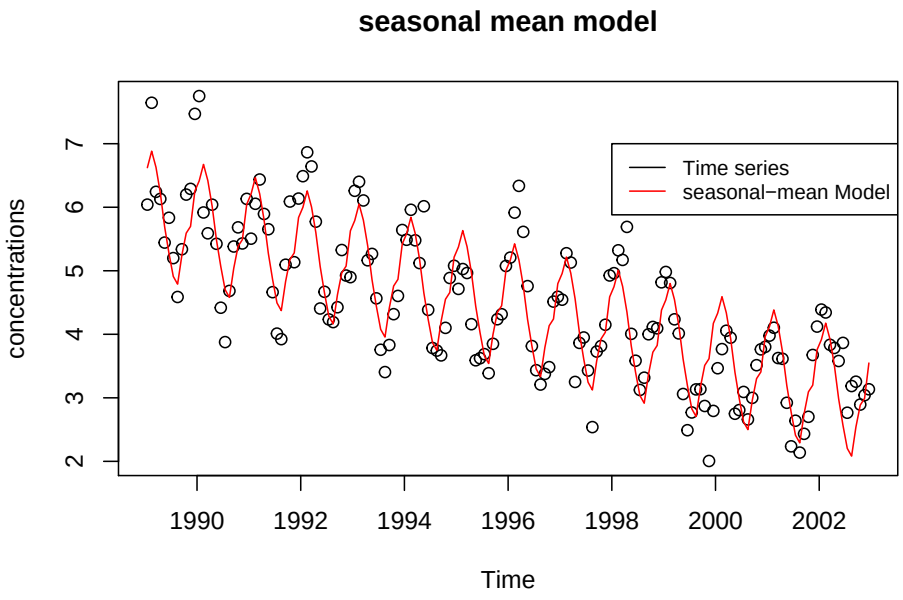
d)

Eliminate the trend by fitting the following so-called seasonal means model:

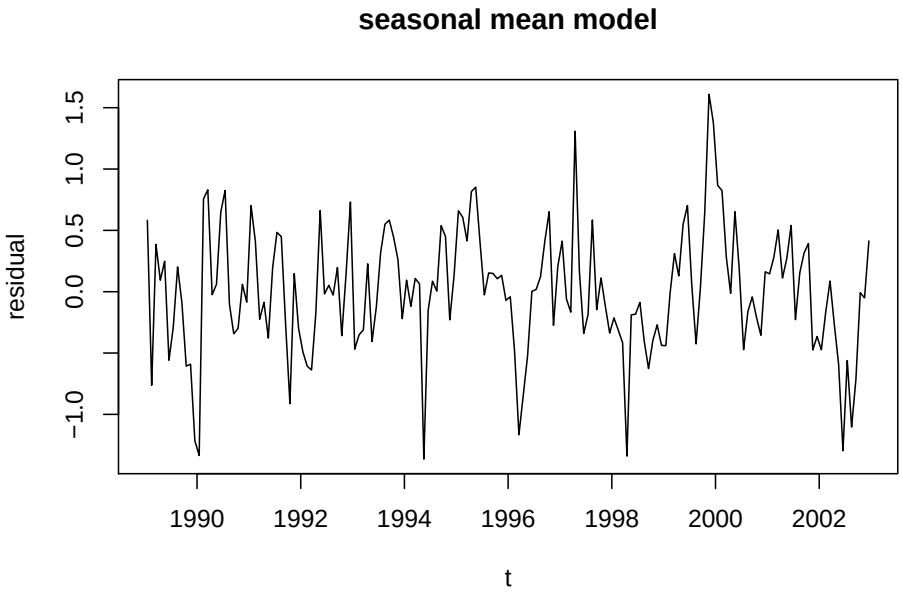
$$x_t = \alpha_0 + \alpha_1 t + \beta_2 I(month = 2) + ... + \beta_{12} I(month = 12) + w_t$$

where $I(x) = 1$ if x is true and 0 otherwise. Fitting of this model will require you to augment data with a categorical variable showing the current month, and then fitting a usual linear regression. Analyze the residual pattern and the ACF of residuals.

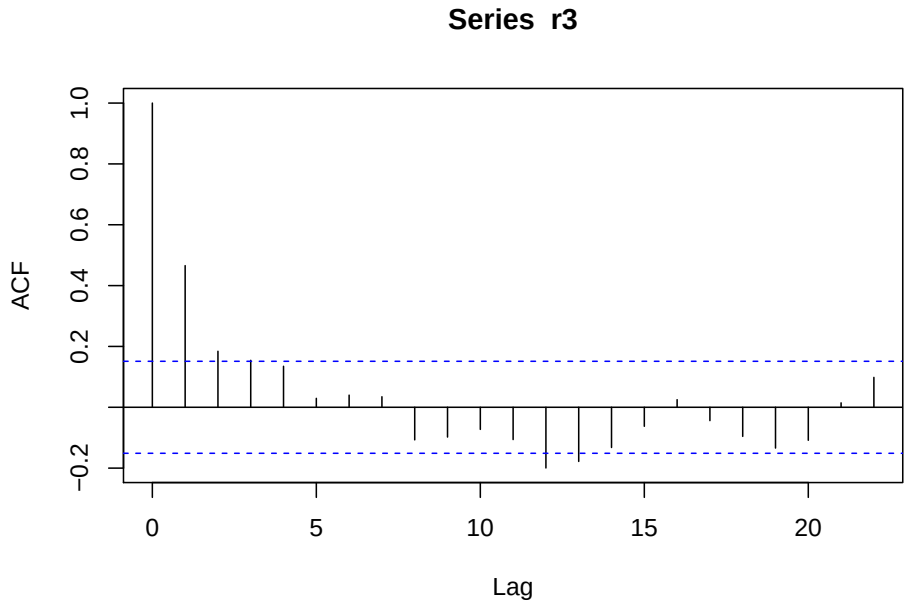
```
# data <- cbind(rhine[,3], matrix(0, nrow = nrow(rhine), ncol = 11))
# for (i in 1:nrow(rhine)) {
#   j <- rhine[i,2] # month
#   if(j>1){
#     data[i, j] <- 1
#   }
# }
# data <- cbind(1,data)
#
# p<- solve((t(data)%*%data),tol=4e-25)%*%t(data)%*%rhine$TotN_conc
# y.pred <- data%*%p

data <- cbind(rhine[,3], matrix(0, nrow = nrow(rhine), ncol = 11))
for (i in 1:nrow(rhine)) {
  j <- rhine[i,2] # month
  if(j>1){
    data[i, j] <- 1
  }
}
data <- as.data.frame(cbind(data, rhine$TotN_conc))
colnames(data) <- c("a1","b2","b3","b4","b5",
                    "b6","b7","b8","b9","b10","b11","b12","TotN_conc")
mdl2 <- lm(TotN_conc~.,data)

plot(rhine$Time, rhine$TotN_conc, xlab="Time", ylab="concentrations", main="seasonal mean model")
lines(rhine$Time,mdl2$fitted.values, col="red")
legend(1998,7, legend = c("Time series" , "seasonal-mean Model"),
       col = c("black","red"),lty = 1, cex=0.8)
```




```
r3 <- mdl2$fitted.values-rhine$TotN_conc
plot(y=r3, x=rhine$Time, type="l", xlab="t", ylab="residual", main = "seasonal mean model")
```




```
acf(r3)
```



The residuals from seasonal-mean model seems as the combination of results between linear regression model and kernel smoother model. It is smoother than kernel smoother, but rougher and more random than linear model.

Considering the ACF plot, it converges to 0 very quickly without seasonally, it seems to that our seasonal-mean model detrends our time series well and the output might be stationary.

e)

Perform stepwise variable selection in model from step d). Which model gives you the lowest AIC value? Which variables are left in the model?

```
set.seed(54321)
step.model <- MASS::stepAIC(mdl2, trace = 1)
```

```
## Start:  AIC=-202.02
## TotN_conc ~ a1 + b2 + b3 + b4 + b5 + b6 + b7 + b8 + b9 + b10 +
##     b11 + b12
##
##        Df Sum of Sq     RSS      AIC
## - b3    1     0.011  43.248 -203.979
## - b12   1     0.220  43.456 -203.170
## <none>               43.237 -202.023
## - b2    1     0.535  43.772 -201.955
## - b4    1     0.840  44.076 -200.790
## - b11   1     3.944  47.180 -189.358
## - b5    1     5.196  48.432 -184.958
## - b10   1     5.345  48.582 -184.441
## - b9    1    10.694  53.930 -166.894
## - b6    1    11.128  54.365 -165.545
## - b7    1    18.090  61.326 -145.303
## - b8    1    20.509  63.745 -138.804
## - a1    1   118.387 161.624   17.499
##
```

```
## Step:  AIC=-203.98
## TotN_conc ~ a1 + b2 + b4 + b5 + b6 + b7 + b8 + b9 + b10 + b11 +
##     b12
##
##        Df Sum of Sq     RSS     AIC
## - b12   1     0.363  43.611 -204.57
## <none>               43.248 -203.98
## - b2    1     0.614  43.862 -203.61
## - b4    1     1.253  44.501 -201.18
## - b11   1     5.542  48.790 -185.72
## - b5    1     7.254  50.502 -179.93
## - b10   1     7.457  50.704 -179.26
## - b9    1    14.724  57.971 -156.75
## - b6    1    15.314  58.562 -155.05
## - b7    1    24.726  67.973 -130.01
## - b8    1    27.989  71.237 -122.14
## - a1    1   118.376 161.624   15.50
##
## Step:  AIC=-204.57
## TotN_conc ~ a1 + b2 + b4 + b5 + b6 + b7 + b8 + b9 + b10 + b11
##
##        Df Sum of Sq     RSS     AIC
## <none>               43.611 -204.57
## - b4    1     0.949  44.560 -202.96
## - b2    1     1.090  44.701 -202.43
## - b11   1     5.218  48.829 -187.59
## - b5    1     6.989  50.600 -181.60
## - b10   1     7.202  50.813 -180.90
## - b9    1    14.882  58.493 -157.25
## - b6    1    15.508  59.119 -155.46
## - b7    1    25.623  69.234 -128.93
## - b8    1    29.155  72.766 -120.57
## - a1    1   119.298 162.908   14.83
```

```
step.model
```

```
##
## Call:
## lm(formula = TotN_conc ~ a1 + b2 + b4 + b5 + b6 + b7 + b8 + b9 +
##     b10 + b11, data = data)
##
## Coefficients:
## (Intercept)           a1           b2           b4           b5
##    421.8678      -0.2088       0.3222      -0.3007      -0.8159
##          b6           b7           b8           b9          b10
##     -1.2153      -1.5622      -1.6665      -1.1907      -0.8285
##         b11
##     -0.7052
```

After stepwise variable selection by `stepAIC()`, the model

$$TotN\_conc \sim a_0 + a_1 t + b_2 + b_4 + b_5 + b_6 + b_7 + b_8 + b_9 + b_{10} + b_{11}$$

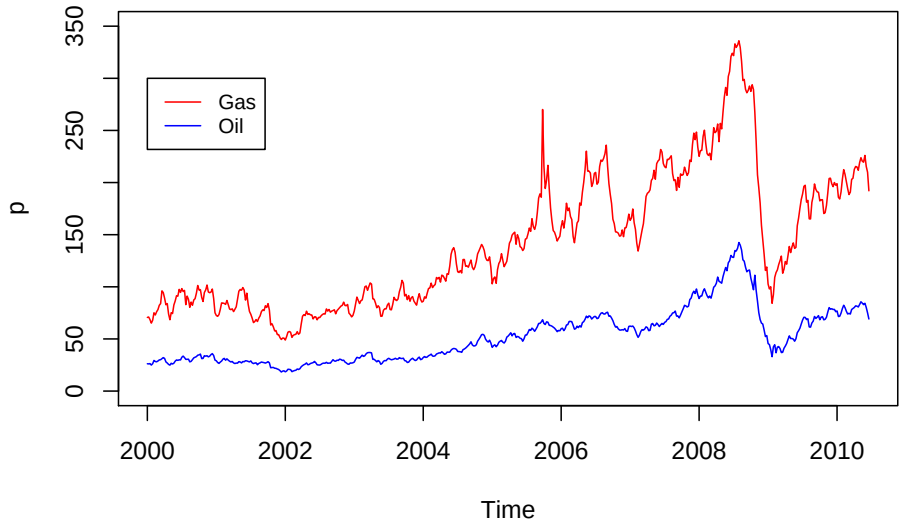
gives us the lowest AIC value. The remaining variables are shown in the notation of model.

Assignment 3

a)

Plot the given time series in the same graph. Do they look like stationary series? Do the processes seem to be related to each other? Motivate your answer.

```
data(oil)
data(gas)

graphics::plot(oil, col="blue",type="l",ylab="p", ylim=c(0,350))
graphics::lines(gas, col="red")
graphics::legend(2000,300, legend = c("Gas" , "Oil"),
       col = c("red","blue"),lty = 1, cex=0.8)
```

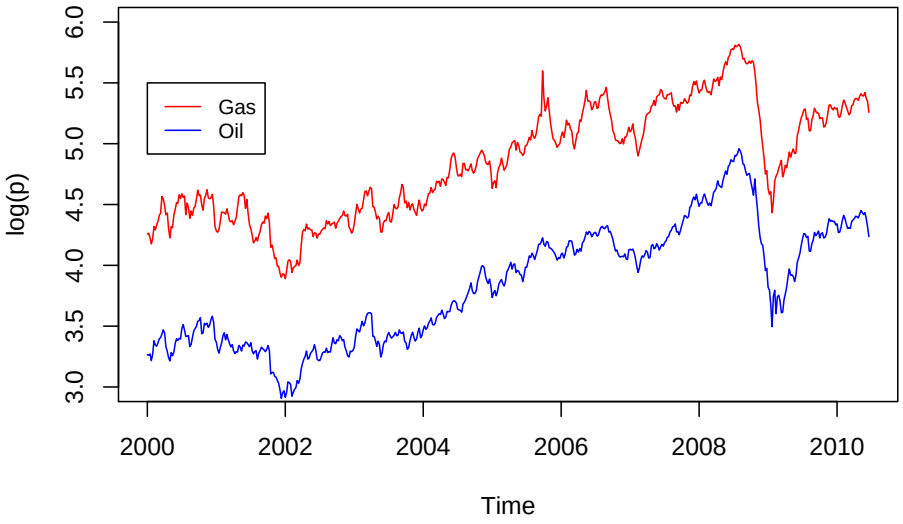


The red line represents the price of gas and the blue one represents the price of oil. They are not stationary, since both of them have an obvious trend from 2000 to 2009. Additionally, such two lines seem to be correlated mutually.

b)

Apply log-transform to the time series and plot the transformed data. In what respect did this transformation made the data easier for the analysis?

```
d1 <- log(oil)
d2 <- log(gas)
graphics::plot(d1, col="blue",type="l",ylab="log(p)", ylim=c(3,6))
graphics::lines(d2, col="red")
graphics::legend(2000,5.5, legend = c("Gas" , "Oil"),
       col = c("red","blue"),lty = 1, cex=0.8)
```

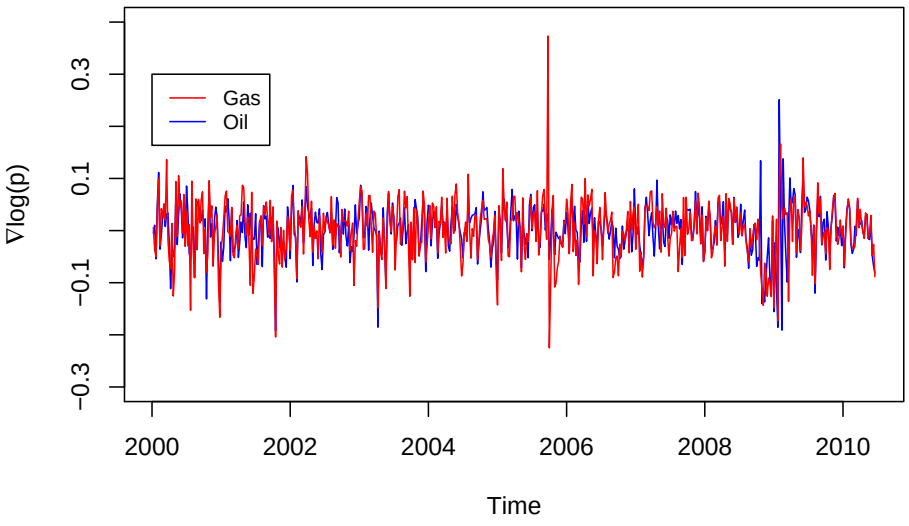


The similarity between gas and oil lines increases in accordance with logarithmization, but their trends are still there, because logarithmization can make the variances of both series more stable and closer to each other.

c)

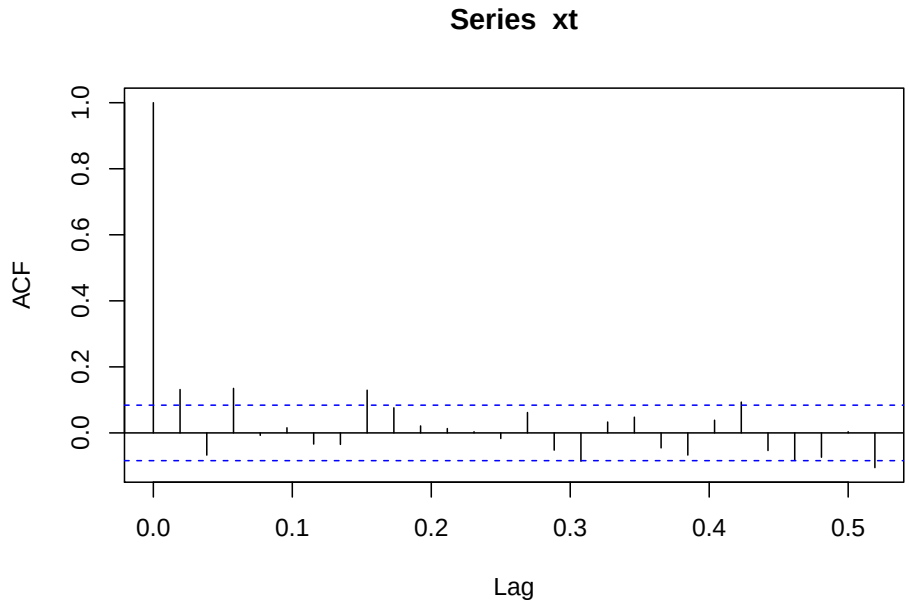
To eliminate trend, compute the first difference of the transformed data, plot the detrended series, check their ACFs and analyze the obtained plots. Denote the data obtained here as x_t (oil) and y_t (gas).

```
xt <- diff(d1, lag=1)
yt <- diff(d2, lag=1)
```

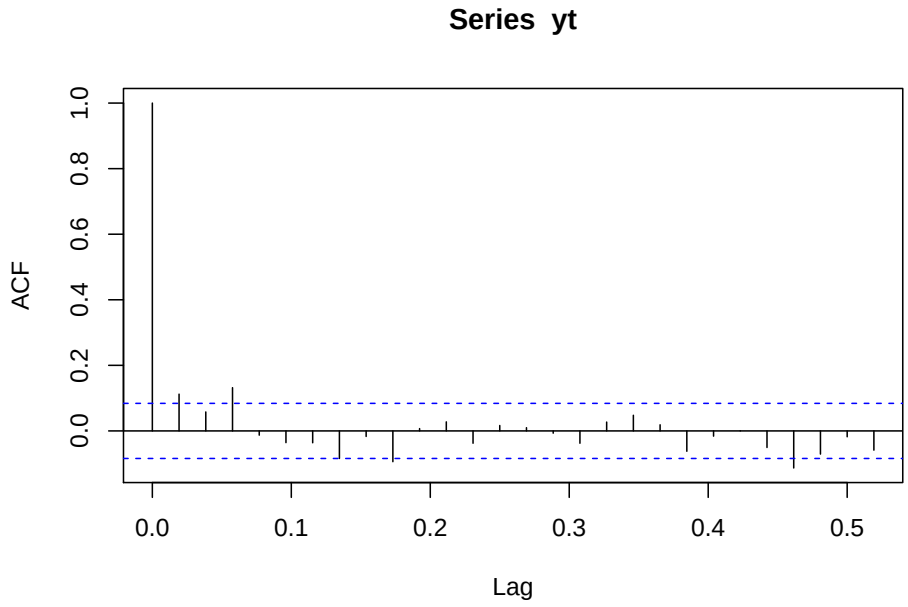


Using 1-order difference, we get two stationary-like series.

```
acf(xt)
```




```
acf(yt)
```

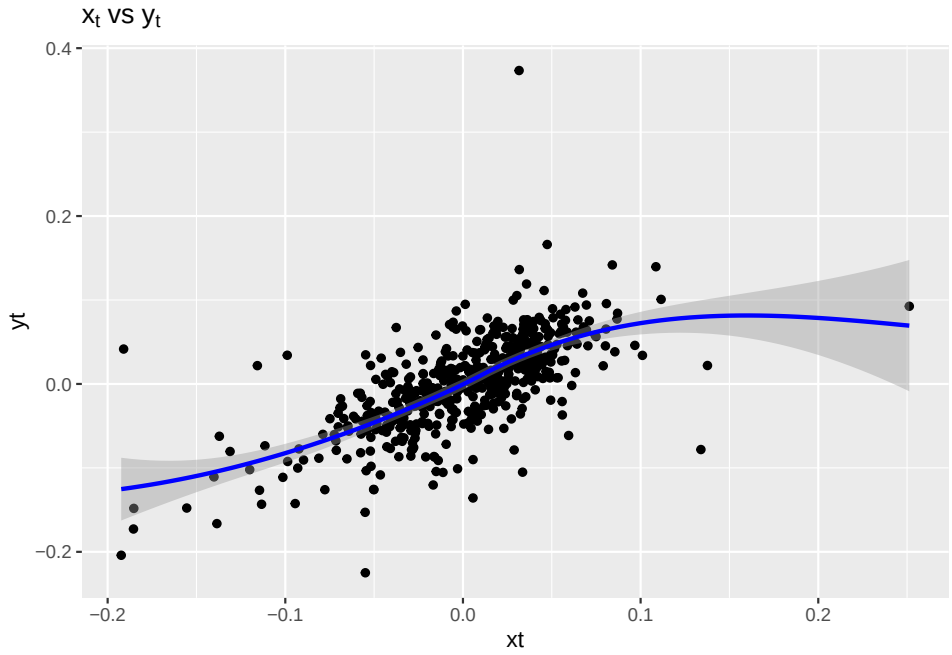


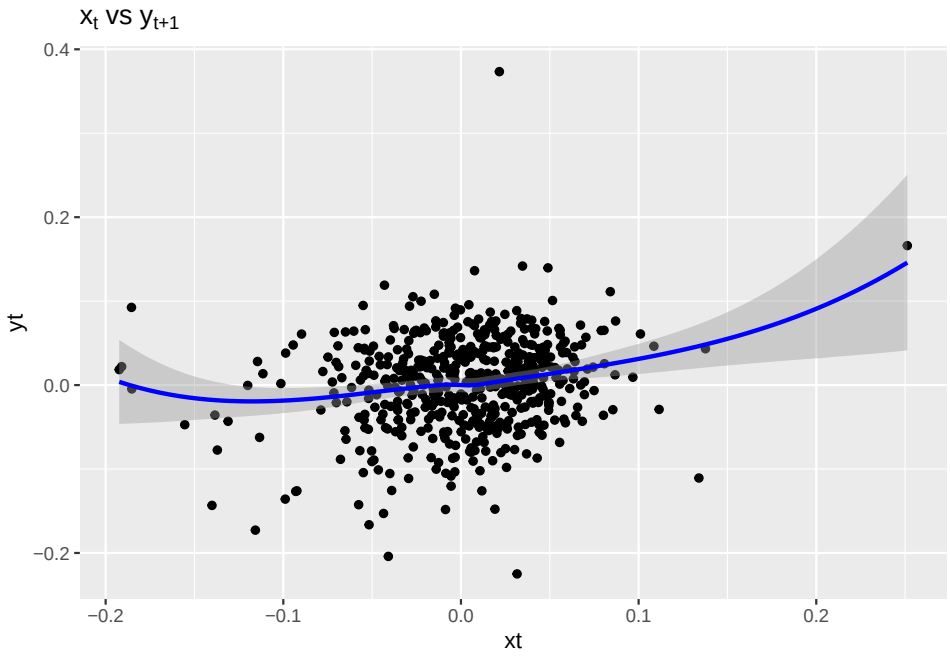
The ACFs of both x_t and y_t show that both time series are not autocorrelated to themselves.

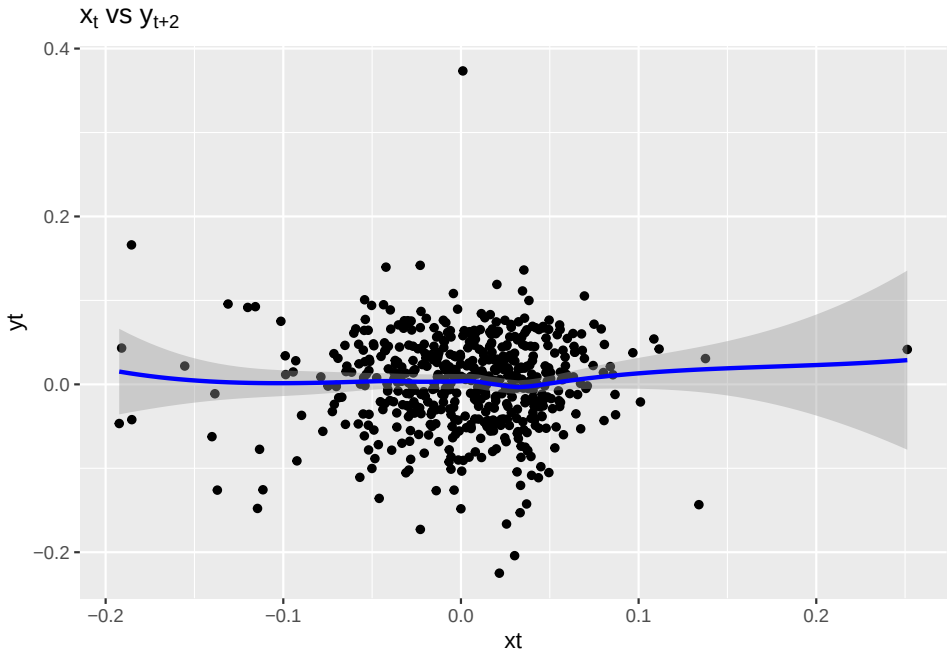
d)

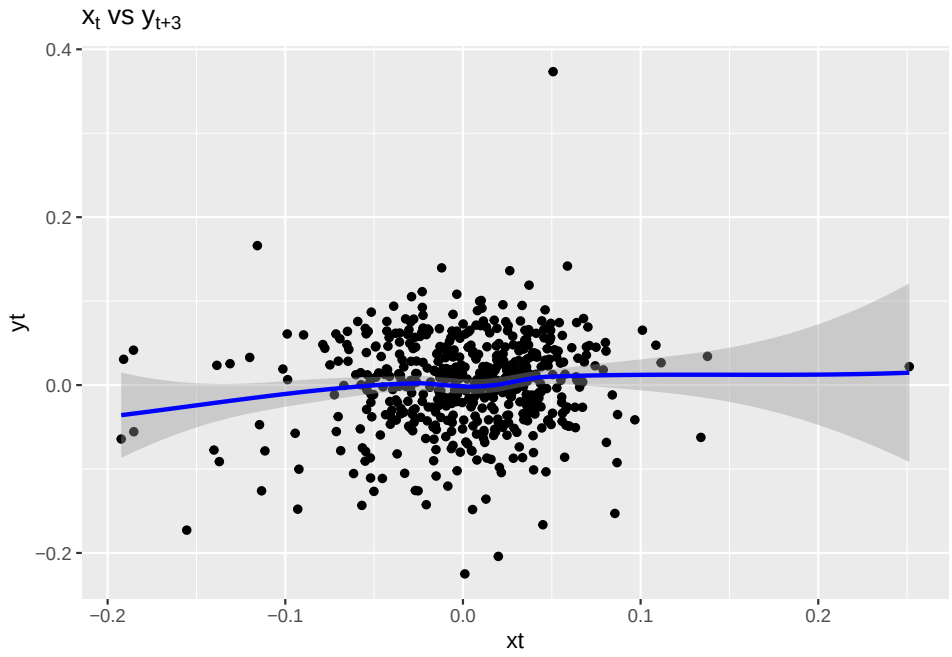
Exhibit scatterplots of x_t and y_t for up to three weeks of lead time of x_t; include a nonparametric smoother in each plot and comment the results: are there outliers? Are the relationships linear? Are there changes in the trend?

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```




x_t vs y_{t+1}
0.4
0.2
0.0
−0.2
yt
−0.2
−0.1
0.0
0.1
0.2
xt


x_t vs y_{t+2}
0.4
0.2
0.0
−0.2
yt
−0.2
−0.1
0.0
0.1
0.2
xt



We use nonparameteric smoother `loess` given by `ggplot::stat_smooth` to fit such four scatter plots. In all plots, there exists several outliers and they change slight with different lag, but such outliers may not affect our fitting dramatically. To sum up, there exits a linear relation between x_t and y_t, but such rerlationship will disappear (fitting values tend to be horizontal line) when the lag increaes.

e)

Fit the following model:

$$y_t = a_0 + a_1 I(x_t > 0) + \beta_1 x_t + \beta_2 x_{t-1} + w_t$$

and check which coefficients seem to be significant. How can this be interpreted? Analyze the residual pattern and the ACF of the residuals.

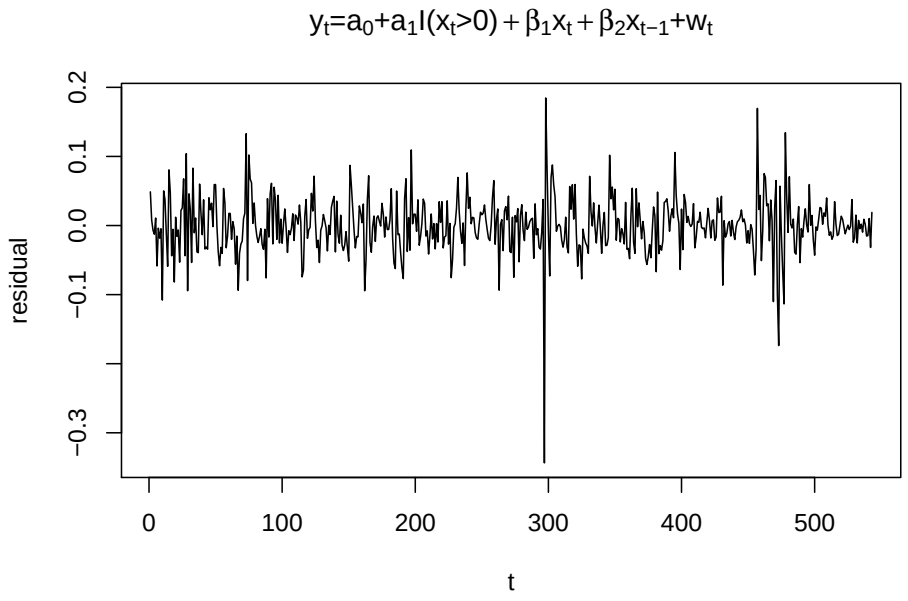
```
df <- as.data.frame(matrix(0, ncol = 4, nrow=543))
df[,1] <- ifelse(xt[-1]>0,1,0)
df[,2] <- xt[-1]
df[,3] <- xt[-544]
df[,4] <- yt[-1]
colnames(df) <- c("a1","b1","b2","y")
mdl3 <- lm(y~., df)
summary(mdl3)
```

```
##
## Call:
## lm(formula = y ~ ., data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.18460 -0.02167 -0.00030  0.02176  0.34352
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.006110   0.003455  -1.768  0.07759 .
```

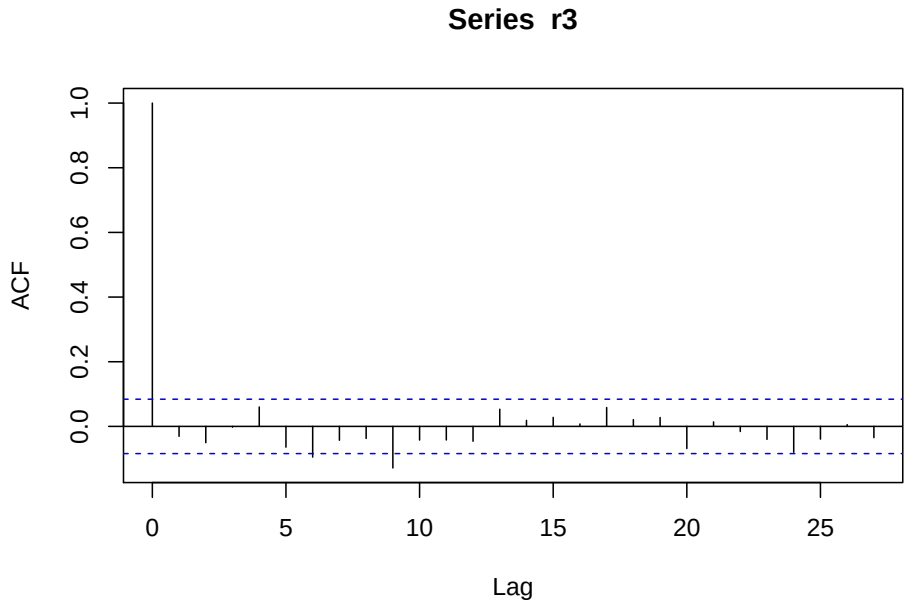
```
## a1           0.011785   0.005514   2.137  0.03303 *
## b1           0.687749   0.058380  11.781  < 2e-16 ***
## b2           0.112152   0.038570   2.908  0.00379 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04171 on 539 degrees of freedom
## Multiple R-squared:  0.4558, Adjusted R-squared:  0.4528
## F-statistic: 150.5 on 3 and 539 DF,  p-value: < 2.2e-16
```

The linear model shows that β_1 has the largest significance compared with other coefficients. This means there is a strong connection between y and x at time t.

```
r3 <- mdl3$fitted.values-yt[-1]
plot(y=r3, x=1:543, type="l", xlab="t", ylab="residual",
     main = TeX("$y_t=a_0+a_1 I(x_t>0)+\\beta_1x_t+\\beta_2x_{t-1}+w_t$"))
```




```
acf(r3)
```



It seems that the residuals have no trend, and the ACF shows there is no autocorrelation among such time series. Thus, this residuals seem to be stationary.

Appendix

```
knitr::opts_chunk$set(echo = TRUE, out.height = "250px")
library(latex2exp)
library(tseries)
library(MASS)
library(astsa)
library(ggplot2)
library(birk)

rm(list=ls())
set.seed(12345)
# filter0 <- function(xs){
#   vs <- c()
#   for (i in 5:length(xs)) {
#     vs <- c(vs, 0.2*(xs[i]+xs[i-1]+xs[i-2]+xs[i-3]+xs[i-4]))
#   }
#   vs
# }

xs1 <- c(0, 0)
n   <- 100
ws  <- rnorm(n)
for (i in 3:n) {
 xs1 <- c(xs1, -0.8*xs1[i-2] + ws[i])
}
vs1 <-stats::filter(xs1, filter = rep(0.2, 5))

xs2 <- c()
for (i in 1:n) {
  xs2 <- c(xs2, cos((2*pi*i)/5))
}
vs2 <- stats::filter(xs2, filter = rep(0.2, 5))

par(mfrow=c(2,1))
plot(xs1,type="l", ylab = TeX("$x_t=-0.8x_{t-2}+w_t$"), xlab="t")
lines(vs1, col="red")
plot(xs2,type="l", ylab = TeX("$x_t=\\cos(2\\pi t/5)$"), xlab="t")
lines(vs2, col="red")
# Causality
z <- c(1,4,-2,0,0,-1)
root <- polyroot(z)
abs(root)>1

# Invertibility
z <- c(1,0,3,0,1,0,-4)
root <- polyroot(z)
abs(root)>1
set.seed(54321)
# sample ACF
sim <- arima.sim(n=100, model=list(ar=c(1, -3/4), ma=c(1,0,-1/9)))
acf.sim <- acf(sim, plot = F)
```

```r
# theoretical ACF
acf.thm <- ARMAacf(ar=c(1, -3/4), ma=c(1,0,-1/9), lag.max = 21)
plot(acf.sim$acf, type="l", ylim=c(-1,1), col="blue", xlab="lag", ylab="ACF")
lines(acf.thm, col="red")
legend(16,0.8,legend=c("sim","thm"), col=c("red","blue"),lty=1)
rm(list=ls())
rhine <- read.csv2("Rhine.csv")
ts0 <- ts(rhine)

plot(ts0, main="time series")
lag.plot(ts0[,4], lag=12)
mdl <- lm(TotN_conc~Time, rhine)
plot(rhine$Time, rhine$TotN_conc, xlab="Time", ylab="concentrations", main = "linear regression")
lines(rhine$Time,mdl$fitted.values, col="red")
legend(2000,7, legend = c("Time series" , "Linear Model"),
       col = c("black","red"),lty = 1, cex=0.8)
r <- mdl$residuals
plot(y=r,x=rhine$Time, type="l", xlab="t", ylab = "residual", main = "linear regression")
acf(r)
# fit kernel smoother

# choose a reasonable bandwidth yourself
bandwidth_values = seq(from = 0.2, to = 1, by = 0.001)

# fit kernel smoother with different bandwidth_values

# save the values in matrix with a row for residuals and columns for bandwidth values
residuals = vector("numeric", length = length(bandwidth_values))
counter = 0
for (i in bandwidth_values) {

  counter  = counter +1

  # fit the smoother
  kernel_model = ksmooth(x = rhine[,3], y = rhine[,4], # x = time & y = nitrogen
                         bandwidth = i)

  # compute the residuals
  residuals[counter] = sum(rhine[,4] - kernel_model$y)
}

# find best residuals
best_bandwidth_index = which.closest(residuals, 0)
best_bandwidth_value = bandwidth_values[best_bandwidth_index]

# fit the best kernel_model
kernel_model_best = ksmooth(x = rhine[,3], y = rhine[,4], # x = time & y = nitrogen
                         bandwidth = best_bandwidth_value)

# fit a bad bandwidth value
kernel_model_worst = ksmooth(x = rhine[,3], y = rhine[,4], # x = time & y = nitrogen
                         bandwidth = 1)
```

```r
best_bandwidth_value
# plot of smoother

plot(rhine[,3], rhine[,4],
     main = "Kernel smoother",
     xlab = "time", ylab = "nitrogen")
lines(x = kernel_model_best$x, y = kernel_model_best$y,
      col = "green")
lines(x = kernel_model_worst$x, y = kernel_model_worst$y,
      col = "blue")
legend(1999, 7.5, legend = c("Time Series", "Best - Smoother", "Smoother =  1"),
       col = c("black", "green", "blue"),
       lty = 1, cex = 0.7)
r2 <- kernel_model_best$y-rhine$TotN_conc
plot(y=r2, x=rhine$Time, type="l", xlab="t", ylab="residual", main = "kernel smoother bw=0.334")
acf(r2)
# data <- cbind(rhine[,3], matrix(0, nrow = nrow(rhine), ncol = 11))
# for (i in 1:nrow(rhine)) {
#   j <- rhine[i,2] # month
#   if(j>1){
#     data[i, j] <- 1
#   }
# }
# data <- cbind(1,data)
#
# p<- solve((t(data)%*%data),tol=4e-25)%*%t(data)%*%rhine$TotN_conc
# y.pred <- data%*%p

data <- cbind(rhine[,3], matrix(0, nrow = nrow(rhine), ncol = 11))
for (i in 1:nrow(rhine)) {
  j <- rhine[i,2] # month
  if(j>1){
    data[i, j] <- 1
  }
}
data <- as.data.frame(cbind(data, rhine$TotN_conc))
colnames(data) <- c("a1","b2","b3","b4","b5",
                    "b6","b7","b8","b9","b10","b11","b12","TotN_conc")
mdl2 <- lm(TotN_conc~.,data)

plot(rhine$Time, rhine$TotN_conc, xlab="Time", ylab="concentrations", main="seasonal mean model")
lines(rhine$Time,mdl2$fitted.values, col="red")
legend(1998,7, legend = c("Time series" , "seasonal-mean Model"),
       col = c("black","red"),lty = 1, cex=0.8)
r3 <- mdl2$fitted.values-rhine$TotN_conc
plot(y=r3, x=rhine$Time, type="l", xlab="t", ylab="residual", main = "seasonal mean model")
acf(r3)
set.seed(54321)
step.model <- MASS::stepAIC(mdl2, trace = 1)
step.model
data(oil)
data(gas)
```

```
graphics::plot(oil, col="blue",type="l",ylab="p", ylim=c(0,350))
graphics::lines(gas, col="red")
graphics::legend(2000,300, legend = c("Gas" , "Oil"),
       col = c("red","blue"),lty = 1, cex=0.8)
d1 <- log(oil)
d2 <- log(gas)
graphics::plot(d1, col="blue",type="l",ylab="log(p)", ylim=c(3,6))
graphics::lines(d2, col="red")
graphics::legend(2000,5.5, legend = c("Gas" , "Oil"),
       col = c("red","blue"),lty = 1, cex=0.8)
xt <- diff(d1, lag=1)
yt <- diff(d2, lag=1)
plot(xt, col="blue",type="l",ylab=TeX("$\\nabla\\log(p)$"),ylim=c(-0.3,0.4))
lines(yt, col="red")
graphics::legend(2000,0.3, legend = c("Gas" , "Oil"),
       col = c("red","blue"),lty = 1, cex=0.8)
acf(xt)
acf(yt)
ggplot(data.frame(xt=xt, yt=yt), aes(x=xt, y=yt)) +
  geom_point() +
  stat_smooth(method="loess", color="Blue") +
  ggtitle(TeX("x_t vs y_{t}"))
ggplot(data.frame(xt=xt[1:543], yt=yt[2:544]), aes(x=xt, y=yt)) +
  geom_point() +
  stat_smooth(method="loess", color="Blue") +
  ggtitle(TeX("x_t vs y_{t+1}"))
ggplot(data.frame(xt=xt[1:542], yt=yt[3:544]), aes(x=xt, y=yt)) +
  geom_point() +
  stat_smooth(method="loess", color="Blue") +
  ggtitle(TeX("x_t vs y_{t+2}"))
ggplot(data.frame(xt=xt[1:541], yt=yt[4:544]), aes(x=xt, y=yt)) +
  geom_point() +
  stat_smooth(method="loess", color="Blue") +
  ggtitle(TeX("x_t vs y_{t+3}"))
df <- as.data.frame(matrix(0, ncol = 4, nrow=543))
df[,1] <- ifelse(xt[-1]>0,1,0)
df[,2] <- xt[-1]
df[,3] <- xt[-544]
df[,4] <- yt[-1]
colnames(df) <- c("a1","b1","b2","y")
mdl3 <- lm(y~., df)
summary(mdl3)
r3 <- mdl3$fitted.values-yt[-1]
plot(y=r3, x=1:543, type="l", xlab="t", ylab="residual",
     main = TeX("$y_t=a_0+a_1 I(x_t>0)+\\beta_1x_t+\\beta_2x_{t-1}+w_t$"))
acf(r3)
```