

Spatio-Temporal Data Analysis Project

2020-04-19



1 Patterns in foreign sims connected to OpenWiFi-Milan

Author: Bernardi Riccardo - 864018

Contents

1	Patterns in foreign sims connected to OpenWiFi-Milan	1
2	Loading the Data	5
3	Exploration of the Data	5
4	Trend recognition	7
4.1	Detrending using LM	8
5	Removing seasonality	9
6	Check Residuals	15
7	Arima	17
8	Auto Arima	18
9	GARCH Model	20
10	Searching for multi seasonalities	23
11	Transforming into msts	26
12	Conclusions	28
13	TODO	28

List of Figures

List of Tables

2 Loading the Data

```
setwd("~/Documents/GitHub/STDA-project-proposal")
set.seed(25061997)

require(zoo)

## Loading required package: zoo
##
## Attaching package: 'zoo'
## The following objects are masked from 'package:base':
##
##      as.Date, as.Date.numeric

require(xts)

## Loading required package: xts
data <- read.csv("sims2018milan.csv", sep = ";")
data <- rbind(data, read.csv("foreignsim_2019-11-07.csv", sep = ";"))

data$prefix <- NULL
data$country <- NULL
data$num <- NULL
data$total.ita.sim <- NULL

ll <- aggregate.data.frame(data$total.foreign.sim, by=list(data$date), FUN=mean)
names(ll)[2] <- "total.foreign.sim"
names(ll)[1] <- "Date"
data <- ll

data <- data[-c(656,657,658), ]
```

3 Exploration of the Data

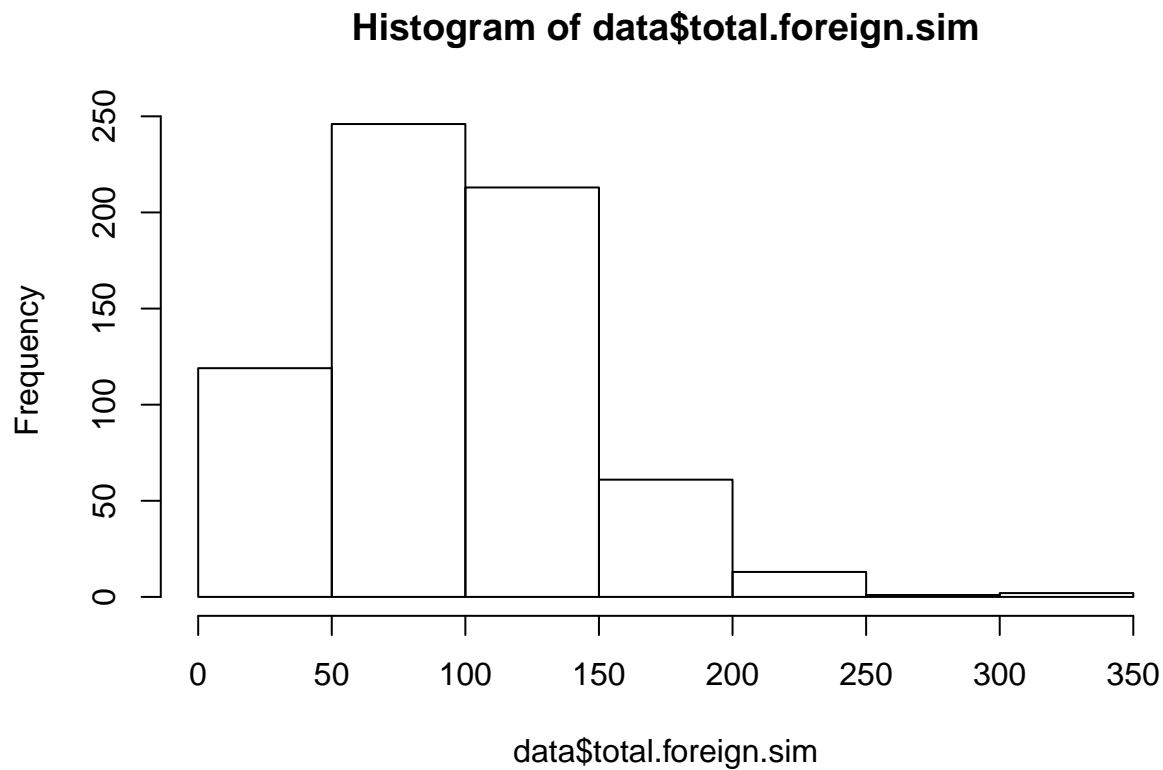
```
print("minimum, lower-hinge, median, upper-hinge, maximum")

## [1] "minimum, lower-hinge, median, upper-hinge, maximum"

fivenum(data$total.foreign.sim)

## [1] 13 59 95 124 344

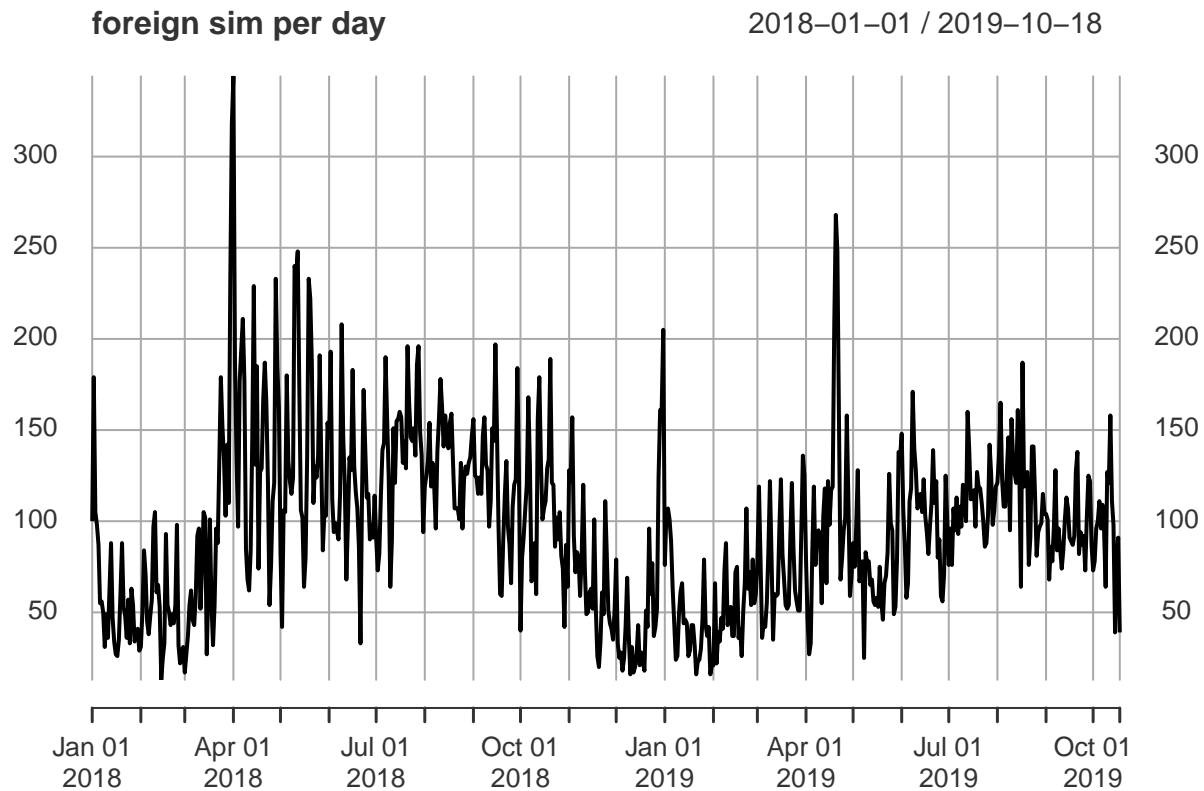
hist(data$total.foreign.sim)
```



We loaded the dataset from the various datasets aggregating into only one dataset with 655 rows representing 2 years of data gathered. Starting from 01.01.2018 to 30.10.2019. Data is here sette

```
data$Date <- as.Date(data$Date, format = "%Y-%m-%d")
#typeof(data$date[1])
data.xts <- xts(data$total.foreign.sim, order.by=data$Date, frequency = 7)
data.ts <- ts(data$total.foreign.sim, frequency = 7)

main <- "foreign sim per day"
ylab<-"Tot of sim in that day"
plot(data.xts,ylab=ylab,main=main)
```

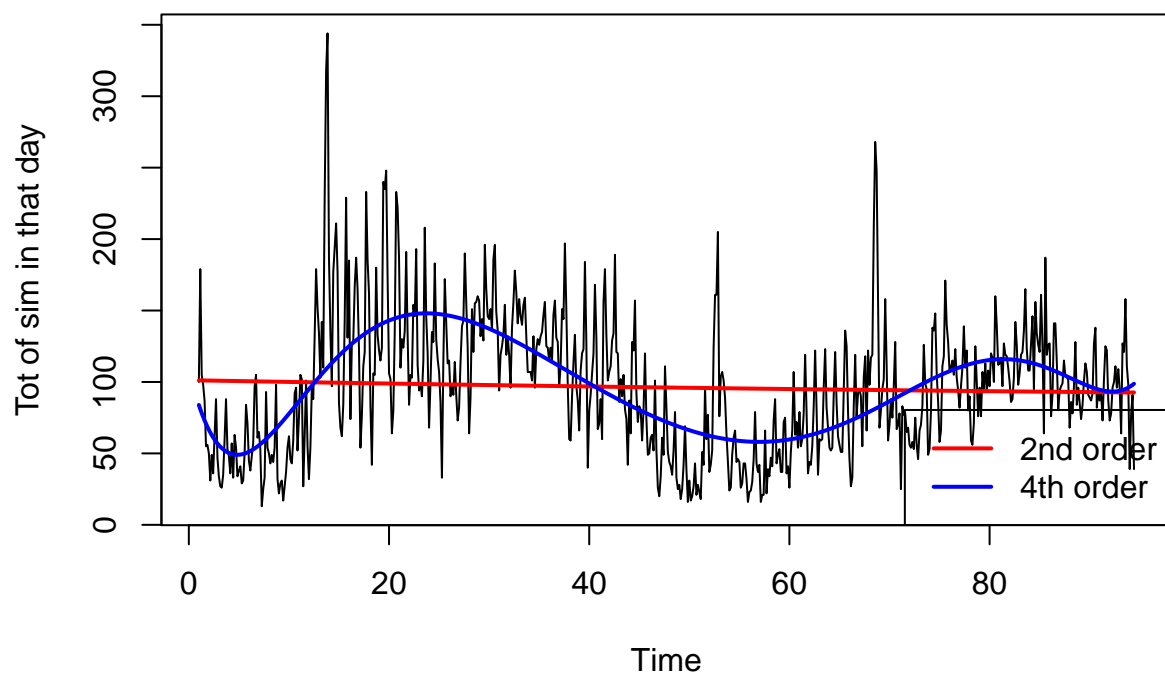


4 Trend recognition

```
tt<-as.numeric(time(data.ts))
fit2<-lm(data.ts~poly(tt,degree=2,raw=TRUE))
fit4<-lm(data.ts~poly(tt,degree=8,raw=TRUE))

main <- "foreign sim per day"
plot(data.ts,ylab=ylab,main=main)
lines(tt,predict(fit2),col='red',lwd=2)
lines(tt,predict(fit4),col='blue',lwd=2)
legend("bottomright",legend = c("2nd order","4th order"),lwd=2,lty=1,col=c("red","blue"))
```

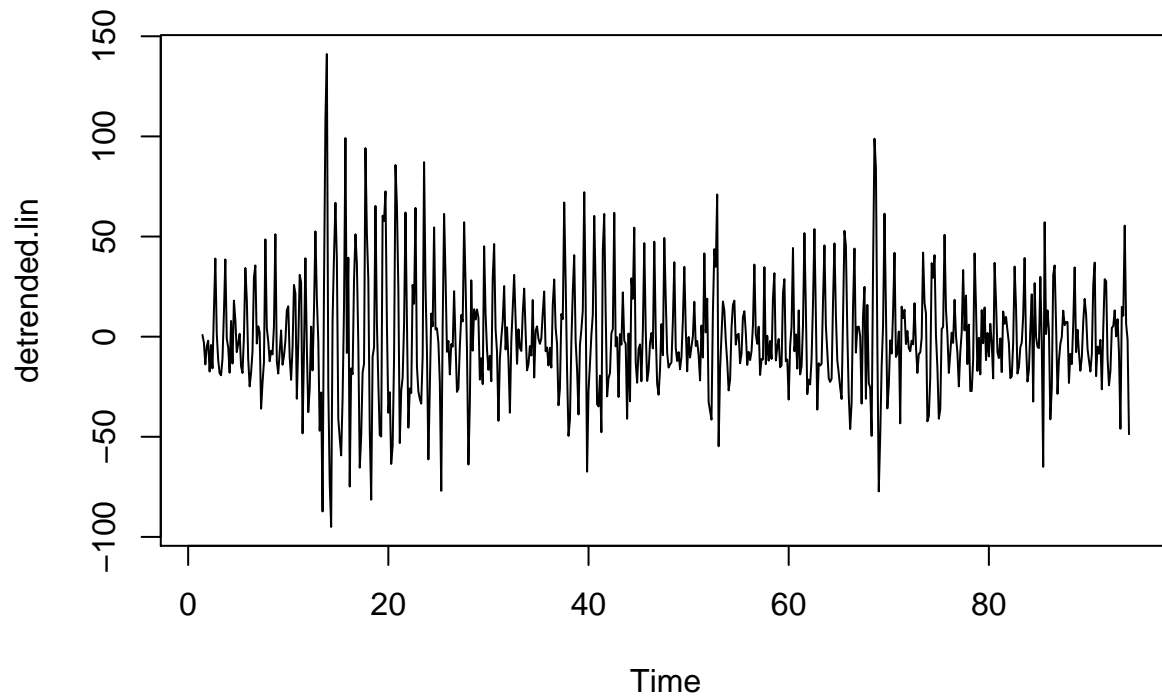
foreign sim per day



4.1 Detrending using LM

```
require(fpp)
```

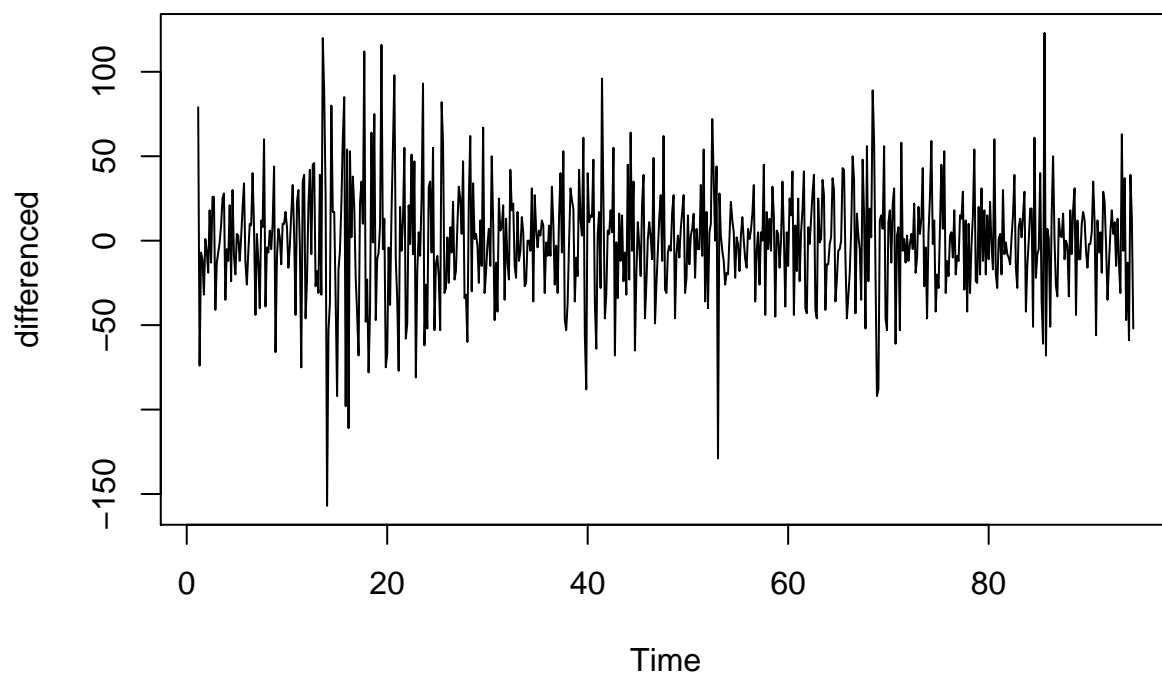
```
## Loading required package: fpp
## Loading required package: forecast
## Warning: package 'forecast' was built under R version 3.6.2
## Registered S3 method overwritten by 'quantmod':
##   method      from
##   as.zoo.data.frame zoo
## Loading required package: fma
## Loading required package: expsmooth
## Loading required package: lmtest
## Loading required package: tseries
detrended.lin <- data.ts - ma(data.ts, order = 7, centre = T)
#detrended.lin <- data.ts - predict(fit4)
plot(detrended.lin)
```

5 Removing seasonality

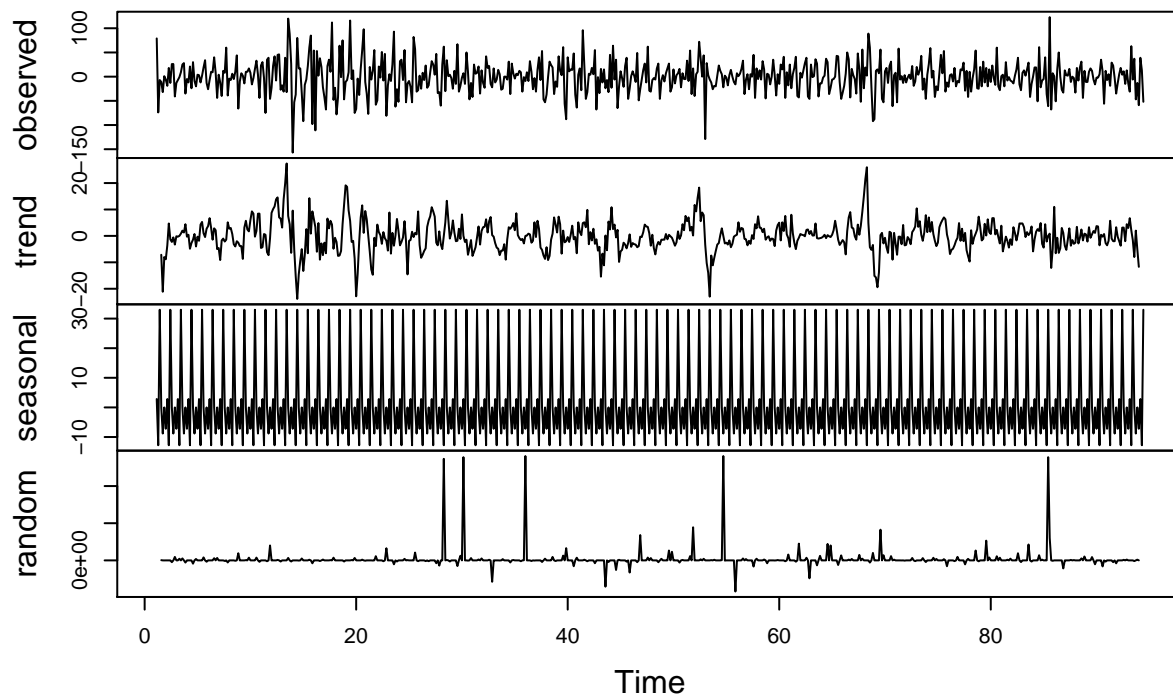
A good idea is to differentiate before decomposing. With the multiplicative model

```
differenced <- diff(data.ts)
plot(differenced)
```



```
decomposed <- decompose(differenced, type = "multiplicative")
plot(decomposed)
```

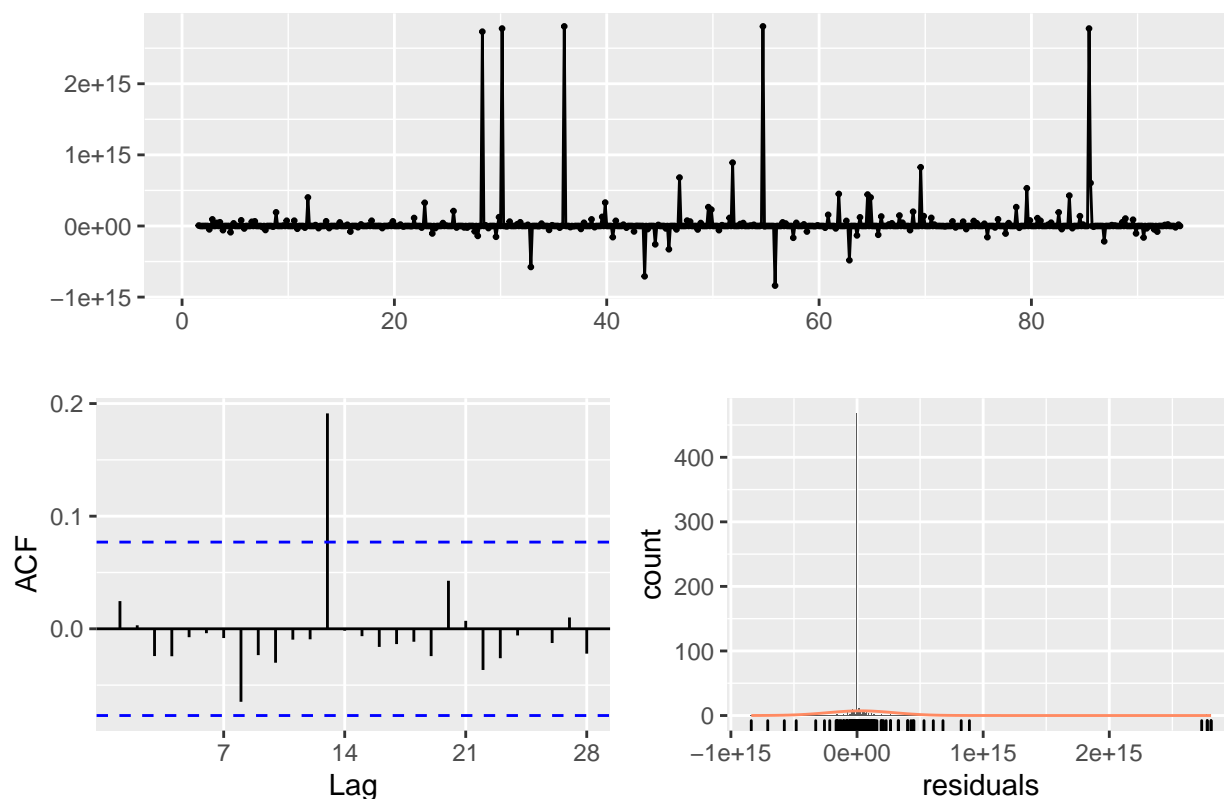
Decomposition of multiplicative time series



```
checkresiduals(decomposed$random)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of  
## freedom for this model.
```

Residuals



```
Box.test(decomposed$random, lag=5, fitdf=0)
```

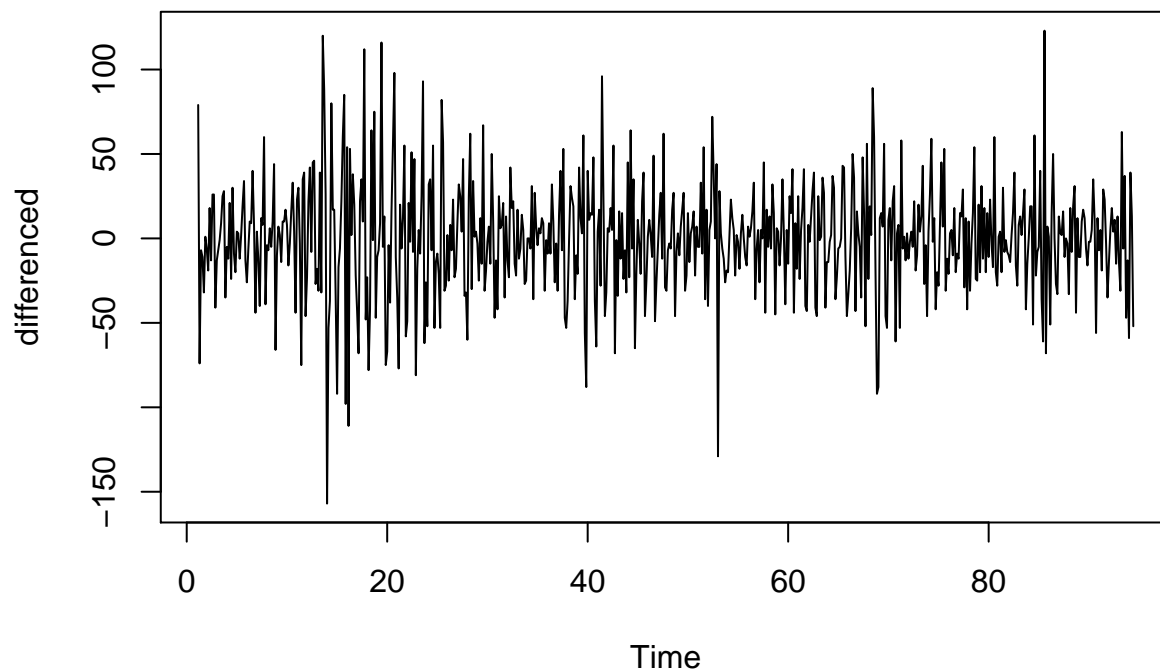
```
##
## Box-Pierce test
##
## data: decomposed$random
## X-squared = 1.1981, df = 5, p-value = 0.9451
```

```
Box.test(decomposed$random, lag=5, fitdf=0, type="Lj")
```

```
##
## Box-Ljung test
##
## data: decomposed$random
## X-squared = 1.2068, df = 5, p-value = 0.9442
```

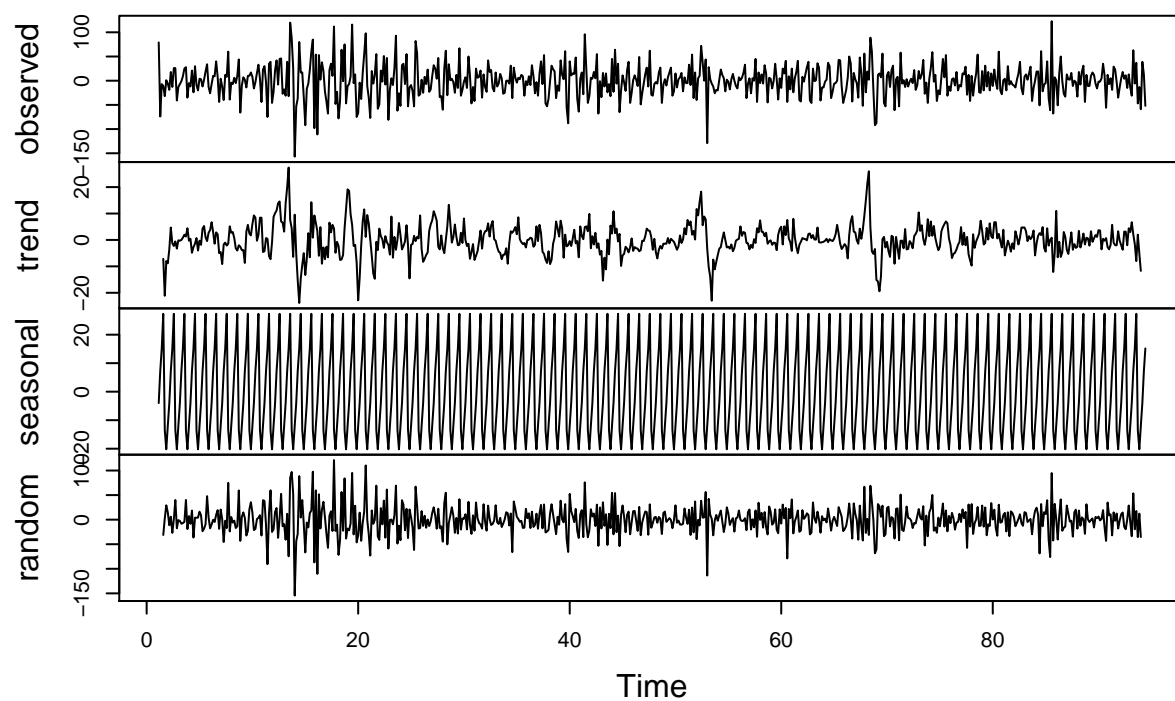
With the additive model This model doesn't work at all

```
differenced <- diff(data.ts)
plot(differenced)
```



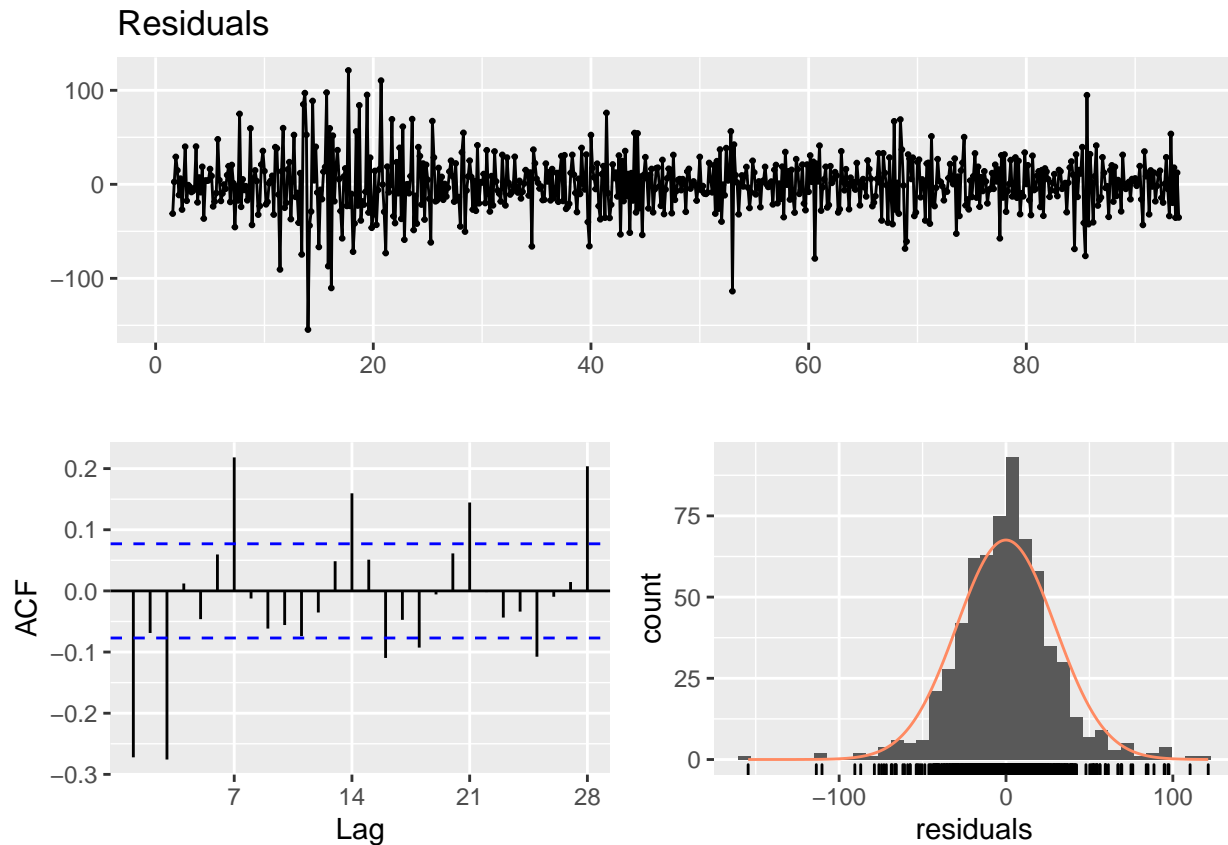
```
decomposed <- decompose(differenced,type = "additive")
plot(decomposed)
```

Decomposition of additive time series



```
checkresiduals(decomposed$random)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```



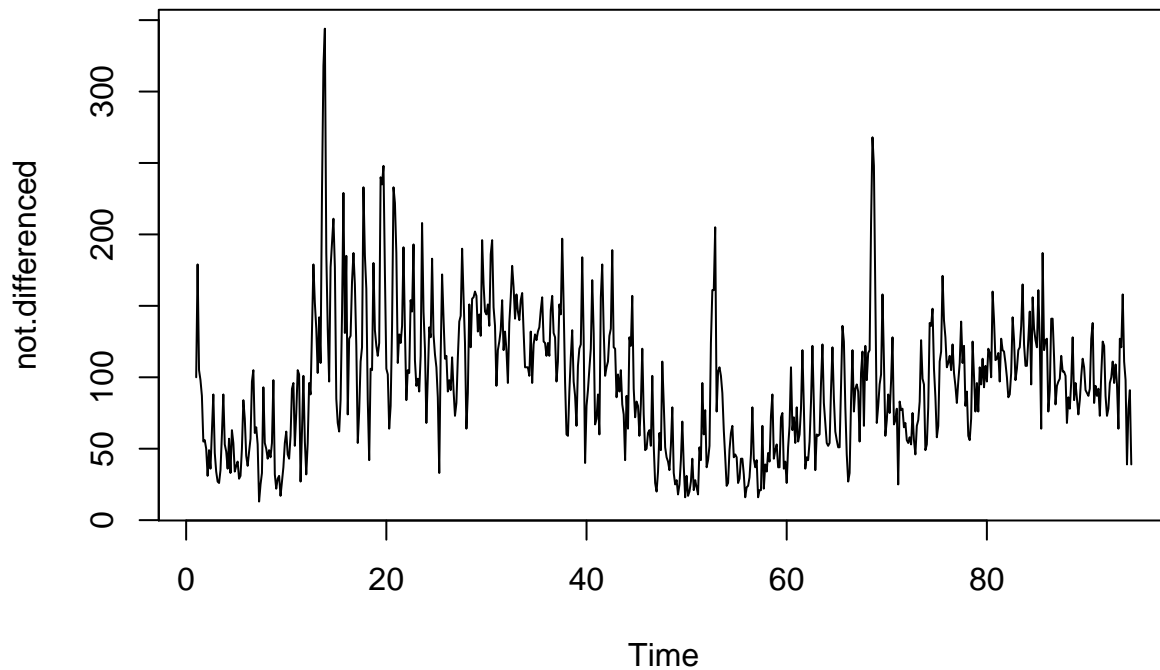
```
Box.test(decomposed$random, lag=5, fitdf=0)

##
## Box-Pierce test
##
## data: decomposed$random
## X-squared = 101.8, df = 5, p-value < 2.2e-16
Box.test(decomposed$random, lag=5, fitdf=0, type="Lj")
```

```
##
## Box-Ljung test
##
## data: decomposed$random
## X-squared = 102.44, df = 5, p-value < 2.2e-16
```

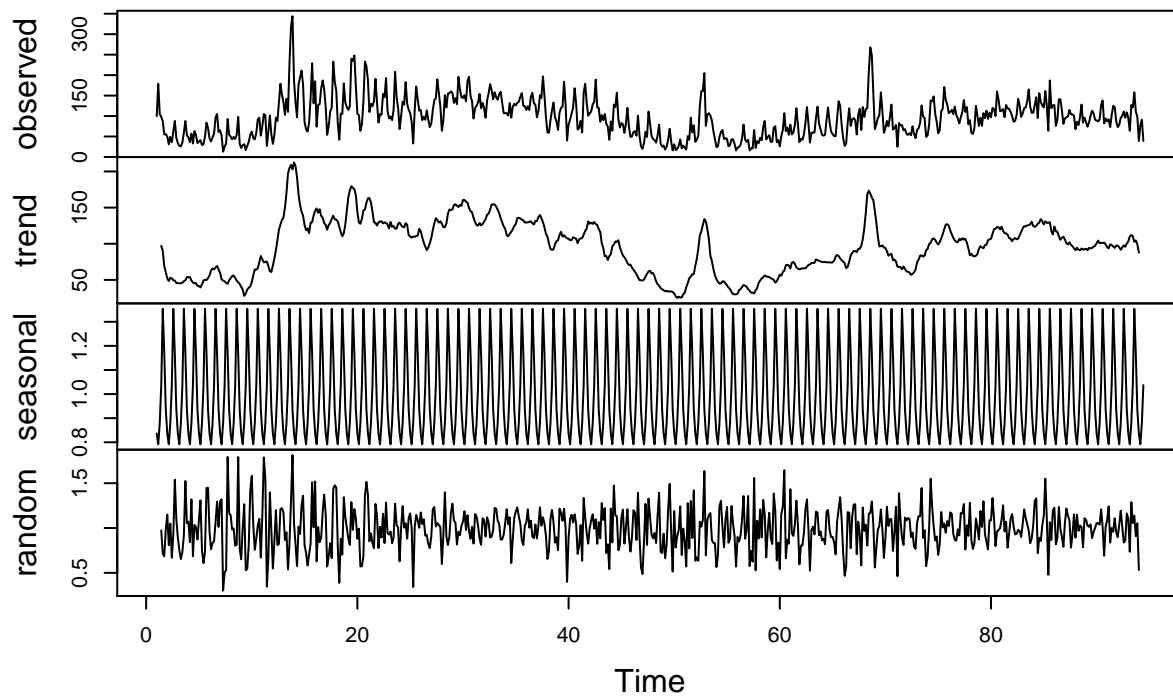
Without the first differentiation the result will have been much worse:

```
not.differenced <- data.ts
plot(not.differenced)
```



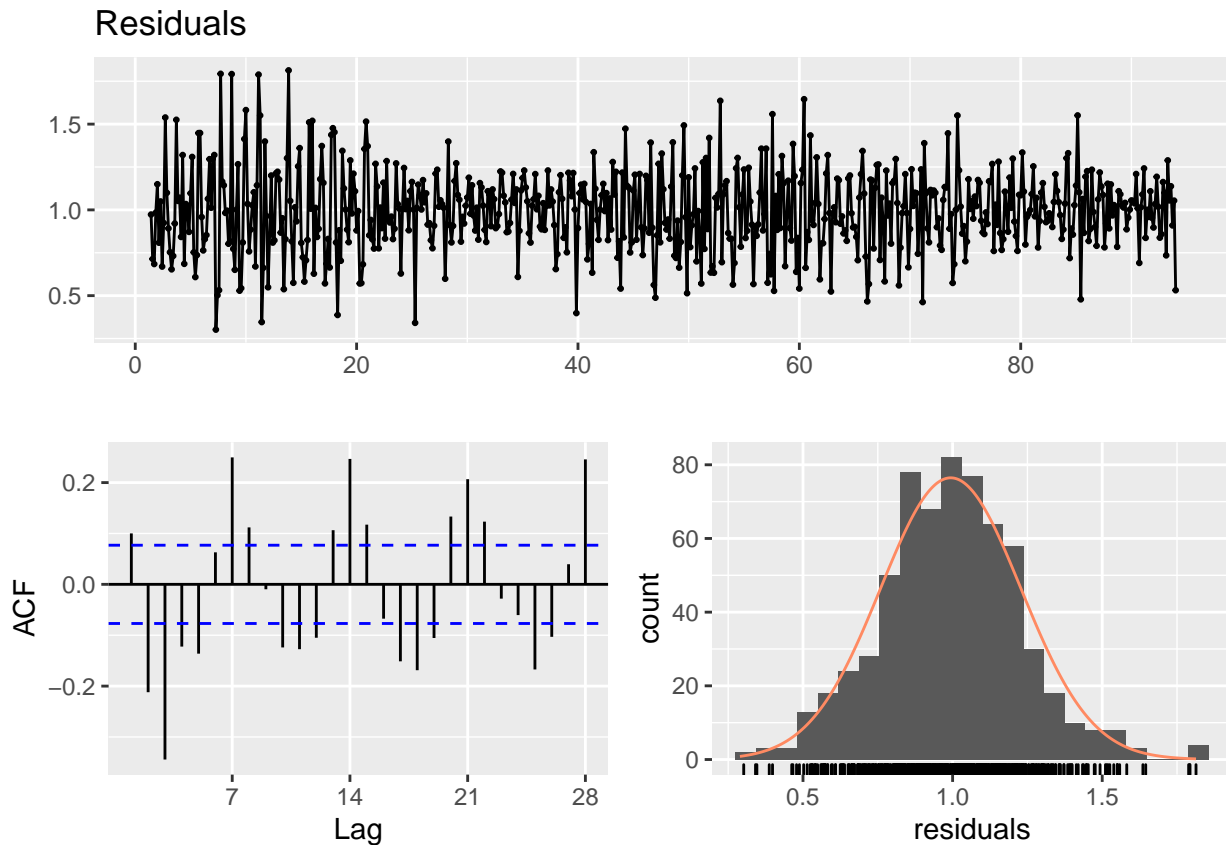
```
decomposed <- decompose(not.differenced,type = "multiplicative")
plot(decomposed)
```

Decomposition of multiplicative time series



```
checkresiduals(decomposed$random)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```



```
Box.test(decomposed$random, lag=5, fitdf=0)
```

```
##
## Box-Pierce test
##
## data: decomposed$random
## X-squared = 134.37, df = 5, p-value < 2.2e-16
```

```
Box.test(decomposed$random, lag=5, fitdf=0, type="Lj")
```

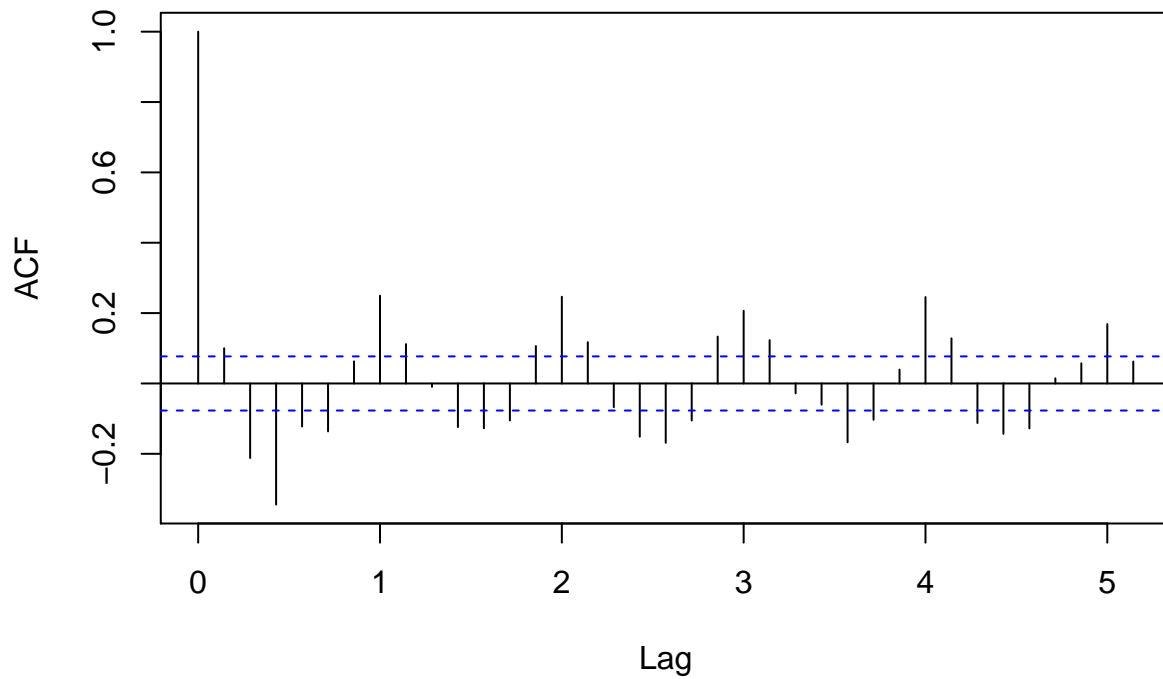
```
##
## Box-Ljung test
##
## data: decomposed$random
## X-squared = 135.4, df = 5, p-value < 2.2e-16
```

Every 7 lags the peak recurs

6 Check Residuals

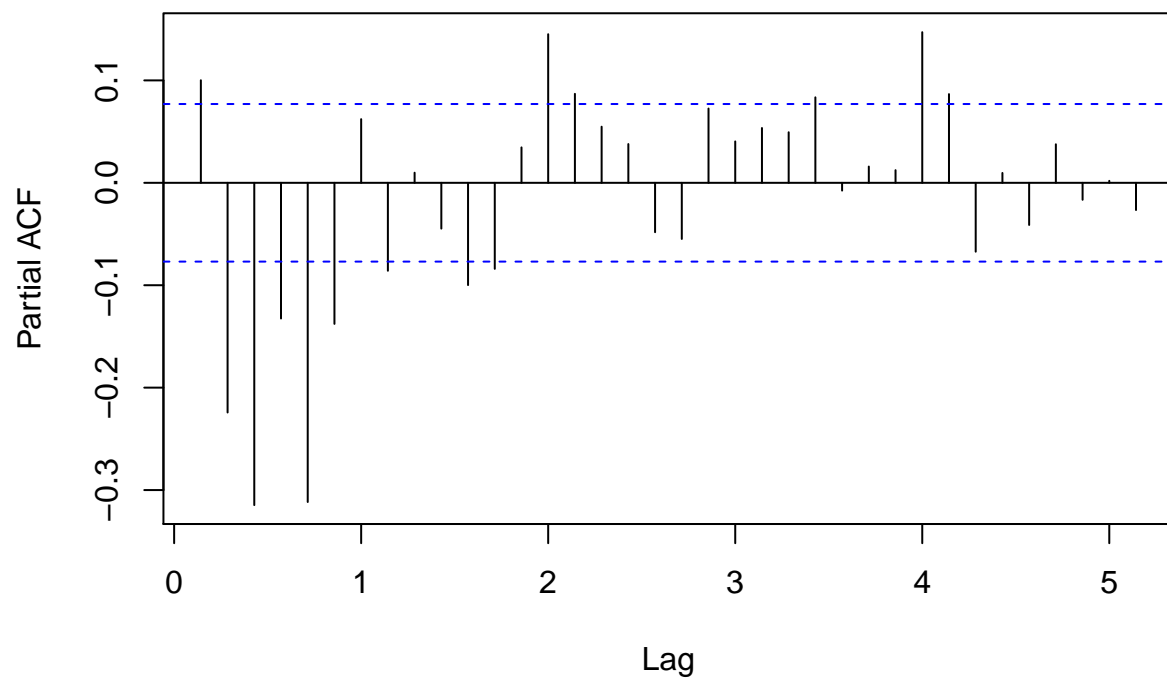
```
acf(na.omit(decomposed$random), main = "Standardized Residuals", 36)
```

Standardized Residuals



```
pacf(na.omit(decomposed$random), main = "Standardized Residuals", 36)
```

Standardized Residuals



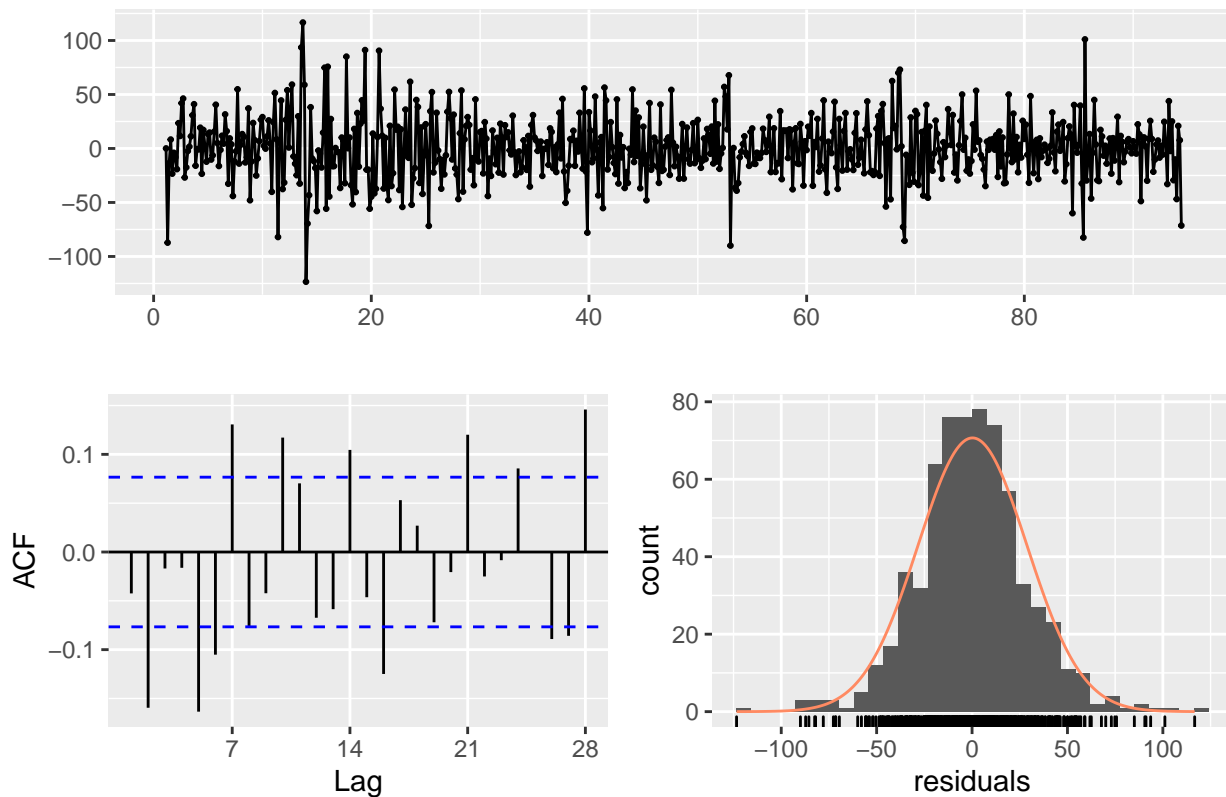
7 Arima

```
require(fpp)
fit <- Arima(differenced, order=c(3,1,3))
summary(fit)
```

```
## Series: differenced
## ARIMA(3,1,3)
##
## Coefficients:
##          ar1      ar2      ar3      ma1      ma2      ma3
##      0.8858 -0.5556 -0.3444 -2.1937  2.0793 -0.8856
## s.e.  0.0407  0.0494  0.0389  0.0211  0.0424  0.0267
##
## sigma^2 estimated as 825:  log likelihood=-3120.88
## AIC=6255.75  AICc=6255.93  BIC=6287.12
##
## Training set error measures:
##              ME      RMSE      MAE  MPE  MAPE      MASE      ACF1
## Training set 0.2368323 28.56859 21.52922 NaN  Inf  0.7603388 -0.04237868
```

```
checkresiduals(fit)
```

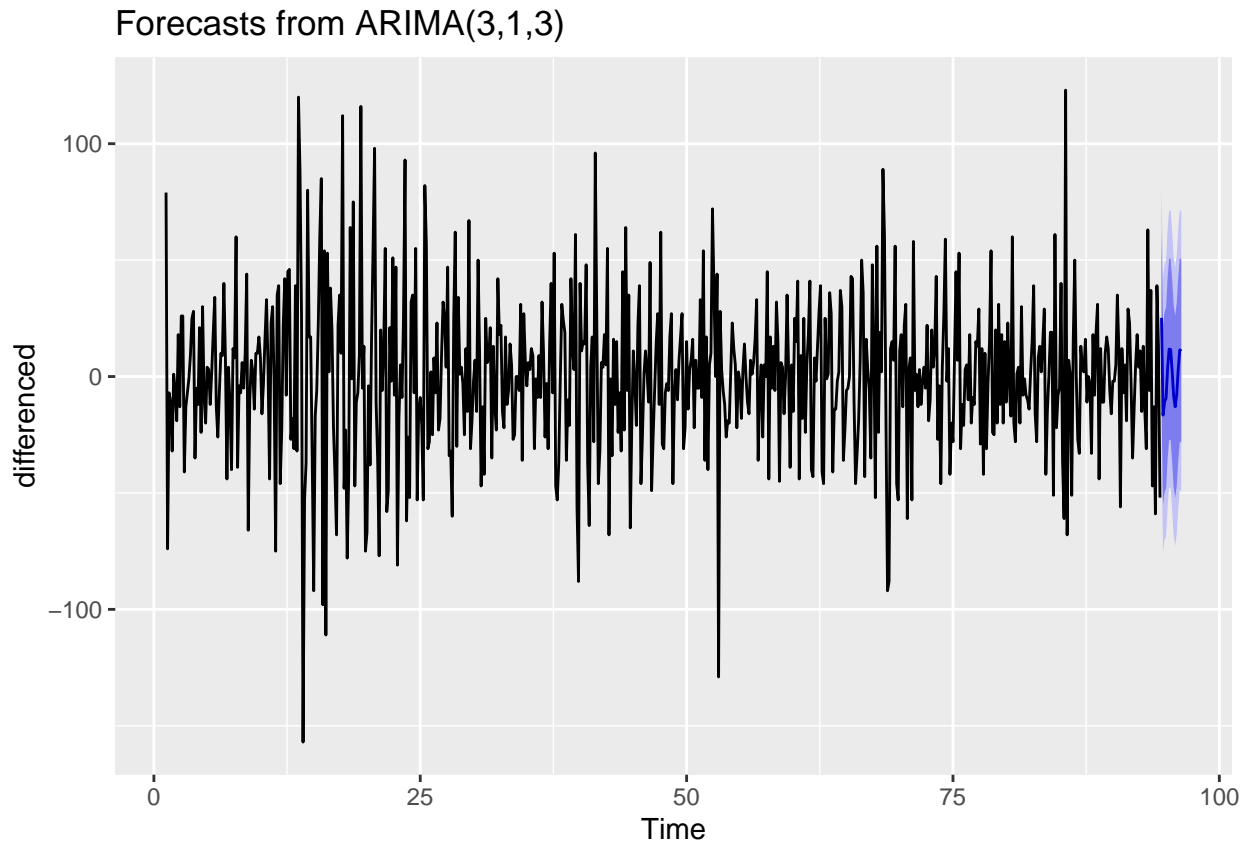
Residuals from ARIMA(3,1,3)



```
##
## Ljung-Box test
##
## data: Residuals from ARIMA(3,1,3)
## Q* = 84.817, df = 8, p-value = 5.218e-15
```

```
##
## Model df: 6.    Total lags used: 14
```

```
autoplot(forecast(fit))
```



8 Auto Arima

```
require(fpp)
auto.arima(data.ts,stepwise=TRUE,trace=TRUE,ic = "bic")
```

```
##
## Fitting models using approximations to speed things up...
##
## ARIMA(2,1,2)(1,0,1)[7] with drift          : Inf
## ARIMA(0,1,0) with drift                    : 6473.005
## ARIMA(1,1,0)(1,0,0)[7] with drift          : 6346.759
## ARIMA(0,1,1)(0,0,1)[7] with drift          : 6395.033
## ARIMA(0,1,0)                               : 6466.527
## ARIMA(1,1,0) with drift                    : 6469
## ARIMA(1,1,0)(2,0,0)[7] with drift          : 6306.944
## ARIMA(1,1,0)(2,0,1)[7] with drift          : Inf
## ARIMA(1,1,0)(1,0,1)[7] with drift          : Inf
## ARIMA(0,1,0)(2,0,0)[7] with drift          : 6343.017
## ARIMA(2,1,0)(2,0,0)[7] with drift          : 6312.557
## ARIMA(1,1,1)(2,0,0)[7] with drift          : 6244.129
## ARIMA(1,1,1)(1,0,0)[7] with drift          : 6286.522
## ARIMA(1,1,1)(2,0,1)[7] with drift          : Inf
```

```

## ARIMA(1,1,1)(1,0,1)[7] with drift : Inf
## ARIMA(0,1,1)(2,0,0)[7] with drift : 6298.145
## ARIMA(2,1,1)(2,0,0)[7] with drift : 6238.386
## ARIMA(2,1,1)(1,0,0)[7] with drift : 6264.384
## ARIMA(2,1,1)(2,0,1)[7] with drift : Inf
## ARIMA(2,1,1)(1,0,1)[7] with drift : Inf
## ARIMA(3,1,1)(2,0,0)[7] with drift : 6234.382
## ARIMA(3,1,1)(1,0,0)[7] with drift : 6258.62
## ARIMA(3,1,1)(2,0,1)[7] with drift : Inf
## ARIMA(3,1,1)(1,0,1)[7] with drift : Inf
## ARIMA(3,1,0)(2,0,0)[7] with drift : 6300.76
## ARIMA(4,1,1)(2,0,0)[7] with drift : Inf
## ARIMA(3,1,2)(2,0,0)[7] with drift : 6240.819
## ARIMA(2,1,2)(2,0,0)[7] with drift : 6243.815
## ARIMA(4,1,0)(2,0,0)[7] with drift : 6291.779
## ARIMA(4,1,2)(2,0,0)[7] with drift : Inf
## ARIMA(3,1,1)(2,0,0)[7] : 6227.929
## ARIMA(3,1,1)(1,0,0)[7] : 6252.359
## ARIMA(3,1,1)(2,0,1)[7] : Inf
## ARIMA(3,1,1)(1,0,1)[7] : Inf
## ARIMA(2,1,1)(2,0,0)[7] : 6232.056
## ARIMA(3,1,0)(2,0,0)[7] : 6294.277
## ARIMA(4,1,1)(2,0,0)[7] : 6234.823
## ARIMA(3,1,2)(2,0,0)[7] : 6234.373
## ARIMA(2,1,0)(2,0,0)[7] : 6306.074
## ARIMA(2,1,2)(2,0,0)[7] : 6237.558
## ARIMA(4,1,0)(2,0,0)[7] : 6285.297
## ARIMA(4,1,2)(2,0,0)[7] : Inf
##
## Now re-fitting the best model(s) without approximations...
##
## ARIMA(3,1,1)(2,0,0)[7] : 6244.074
##
## Best model: ARIMA(3,1,1)(2,0,0)[7]

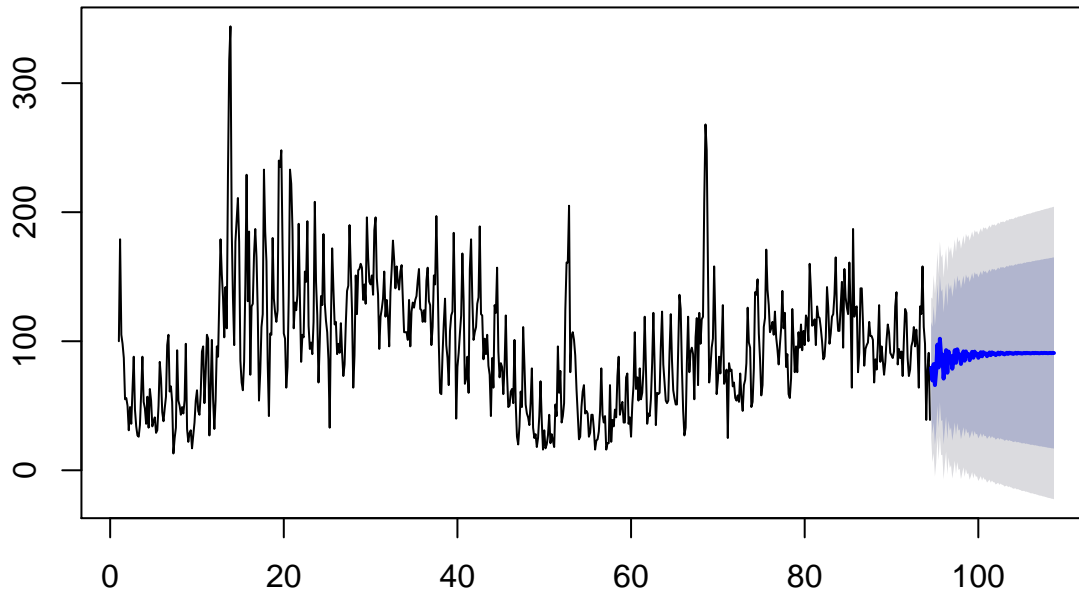
## Series: data.ts
## ARIMA(3,1,1)(2,0,0)[7]
##
## Coefficients:
##          ar1      ar2      ar3      ma1      sar1      sar2
##          0.5742  0.1332 -0.1068 -0.9754  0.3347  0.2233
## s.e.  0.0404  0.0473   0.0411   0.0128  0.0402  0.0416
##
## sigma^2 estimated as 769.1:  log likelihood=-3099.35
## AIC=6212.69   AICc=6212.87   BIC=6244.07

aa <- Arima(order = c(3,1,1), seasonal = c(2,0,0), y = data.ts)

plot(forecast(aa,data.ts))

```

Forecasts from ARIMA(3,1,1)(2,0,0)[7]



The plot is not good but AIC and BIC are very high, we should try with a multi seasonal decomposition

```
frequency(data.ts)
```

```
## [1] 7
```

9 GARCH Model

```
library(rugarch)
```

```
## Loading required package: parallel
```

```
##
```

```
## Attaching package: 'rugarch'
```

```
## The following object is masked from 'package:stats':
```

```
##
```

```
##      sigma
```

```
library(xts)
```

```
fit = ugarchfit(ugarchspec(), data.ts, out.sample = 10)
```

```
c(is(sigma((fit))), is(fitted(fit)), is(residuals(fit)))
```

```
## [1] "xts"      "oldClass" "xtsORzoo" "xts"      "oldClass" "xtsORzoo" "xts"
```

```
## [8] "oldClass" "xtsORzoo"
```

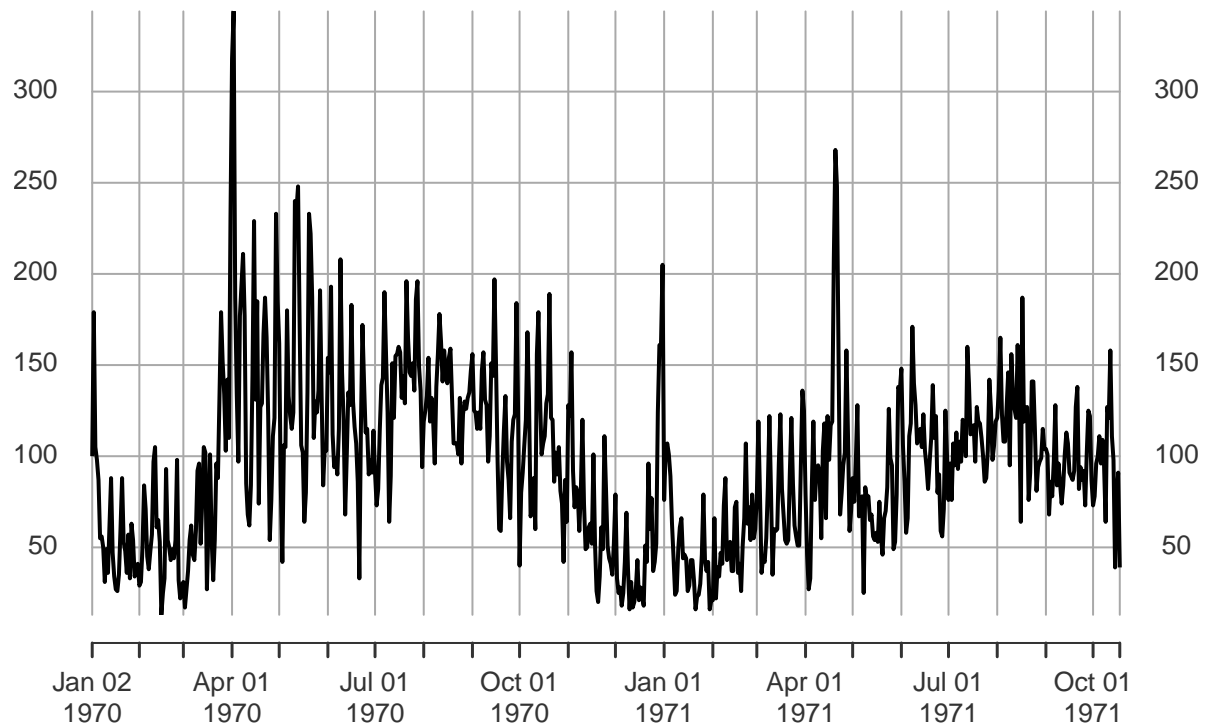
```
## [1] 'xts' 'xts' 'xts'
```

```
##
```

```
plot(xts(fit@model$modeldata$data, fit@model$modeldata$index), auto.grid = FALSE,  
     minor.ticks = FALSE, main = 'Conditional Mean')
```

Conditional Mean

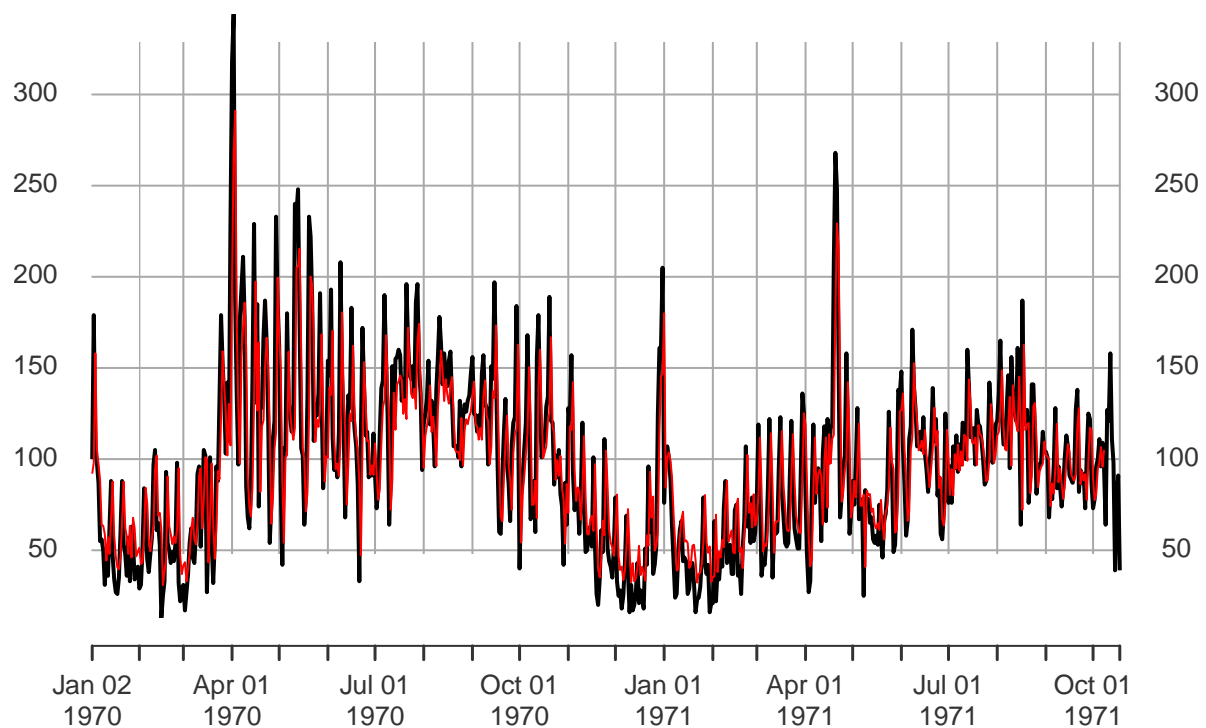
1970-01-02 01:00:00 / 1971-10-18 01:00:00



```
lines(fitted(fit), col = 2)  
grid()
```

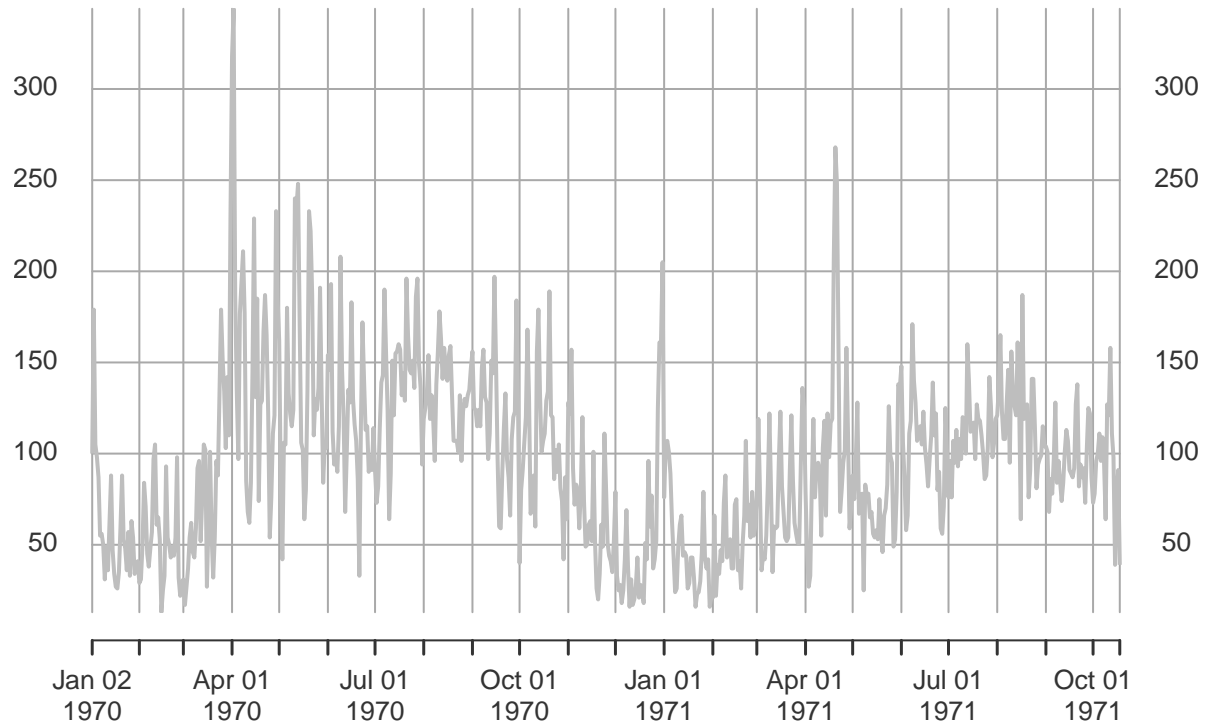
Conditional Mean

1970-01-02 01:00:00 / 1971-10-18 01:00:00



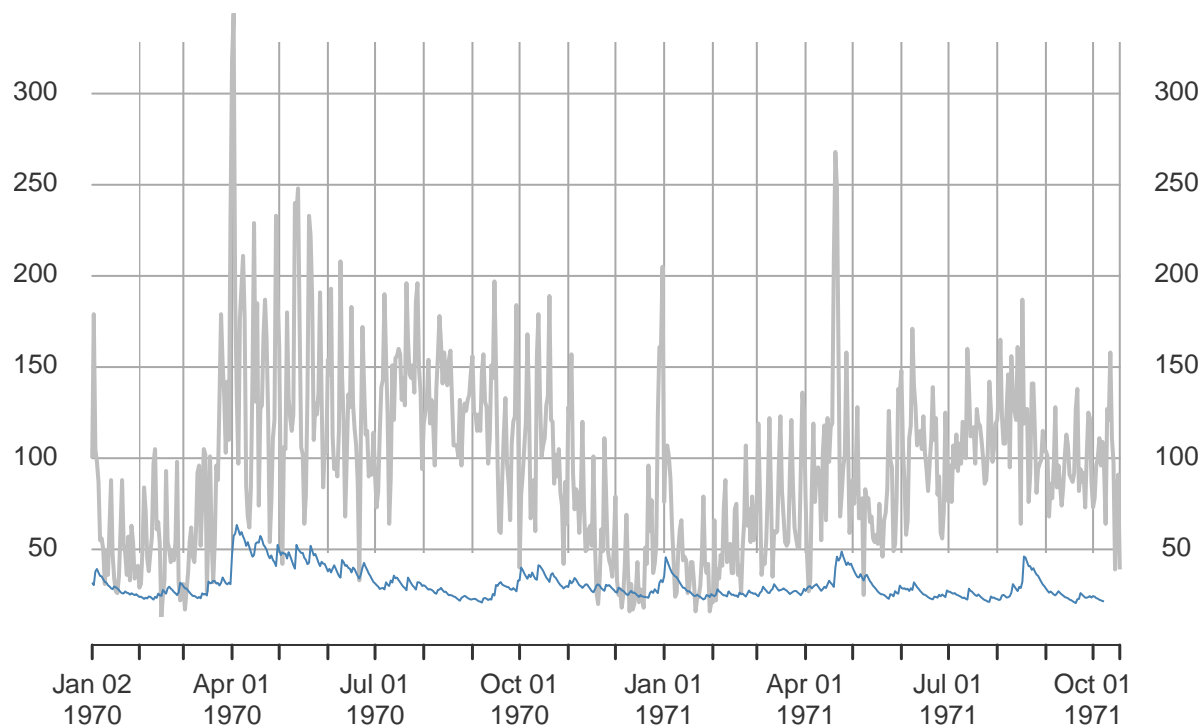
```
plot(xts(abs(fit@model$modeldata$data), fit@model$modeldata$index), auto.grid = FALSE,
     minor.ticks = FALSE, main = 'S&P500 Conditional Sigma', col = 'grey')
```

S&P500 Conditional Sigma 1970-01-02 01:00:00 / 1971-10-18 01:00:00



```
lines(sigma(fit), col = 'steelblue')
grid()
```

S&P500 Conditional Sigma 1970-01-02 01:00:00 / 1971-10-18 01:00:00



10 Searching for multi seasonalities

without differentiation residuals looks pretty bad

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 3.6.2
```

```
##
```

```
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
##      date, intersect, setdiff, union
```

```
library(dplyr)
```

```
##
```

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

```
## The following objects are masked from 'package:lubridate':
```

```
##
```

```
##      intersect, setdiff, union
```

```
## The following objects are masked from 'package:xts':
```

```
##
```

```
##      first, last
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

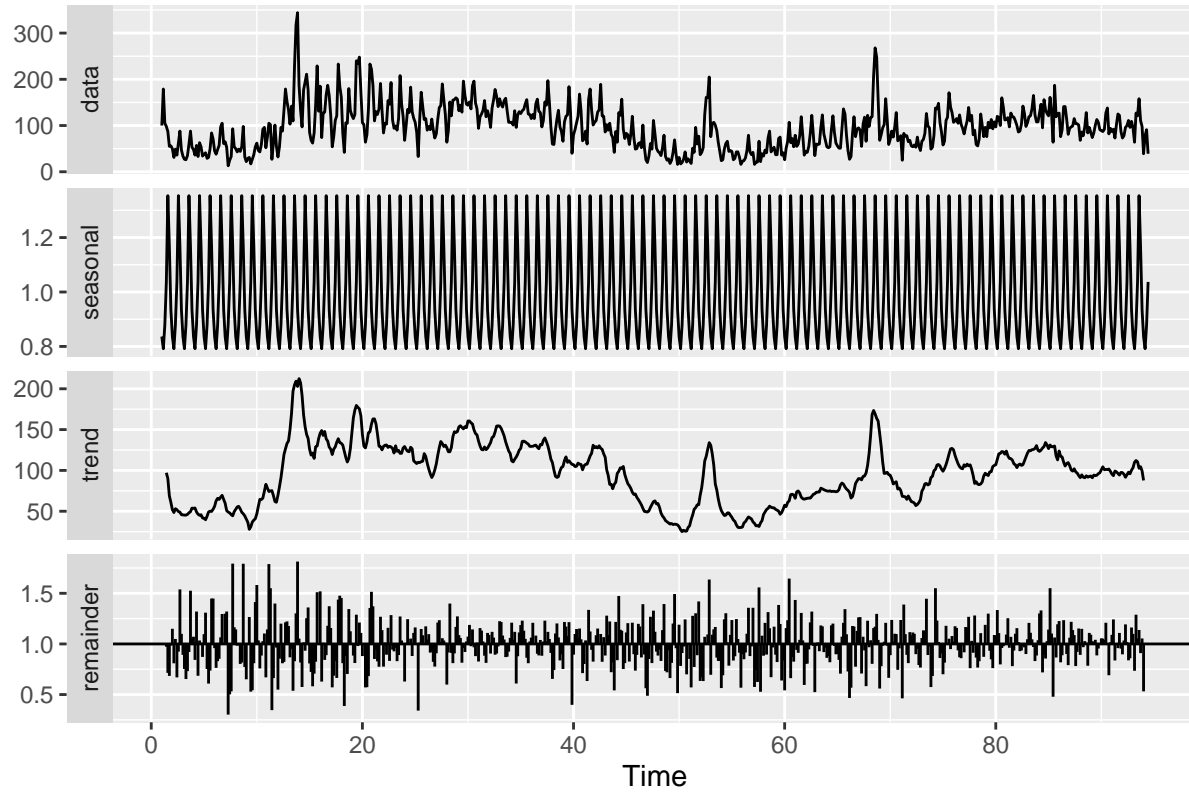
```
##      filter, lag
```

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

library(forecast)
library(ggplot2)
library(scales)

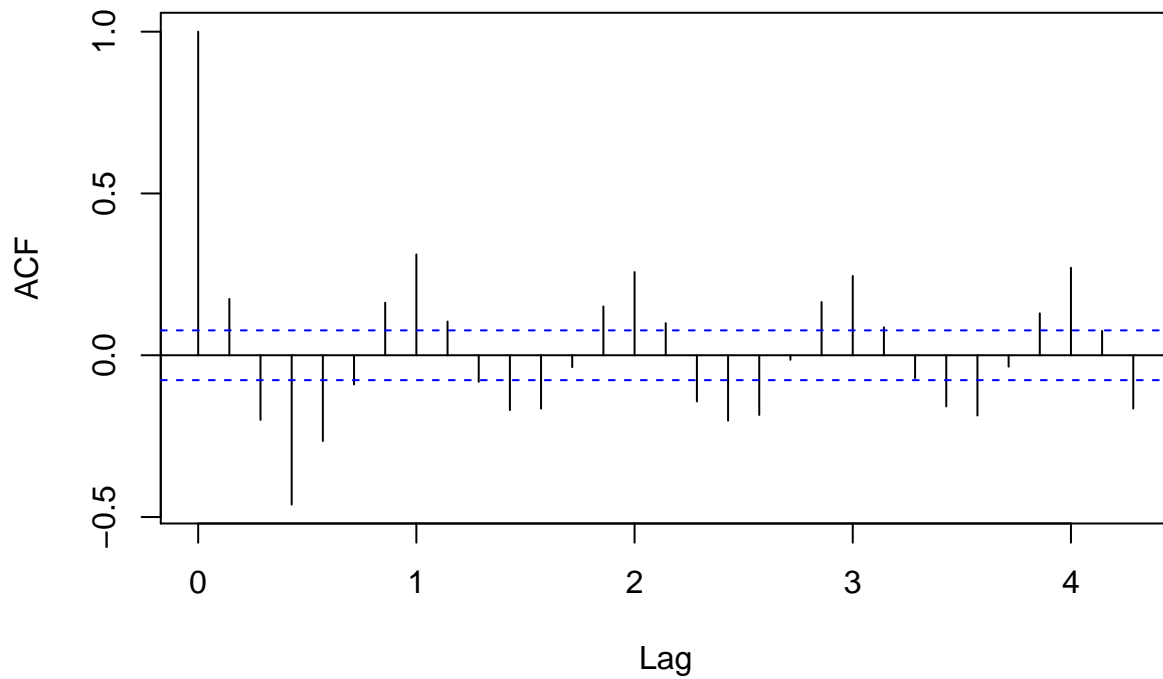
data.ts %>% decompose(type="multiplicative") %>% autoplot()
```

Decomposition of multiplicative time series



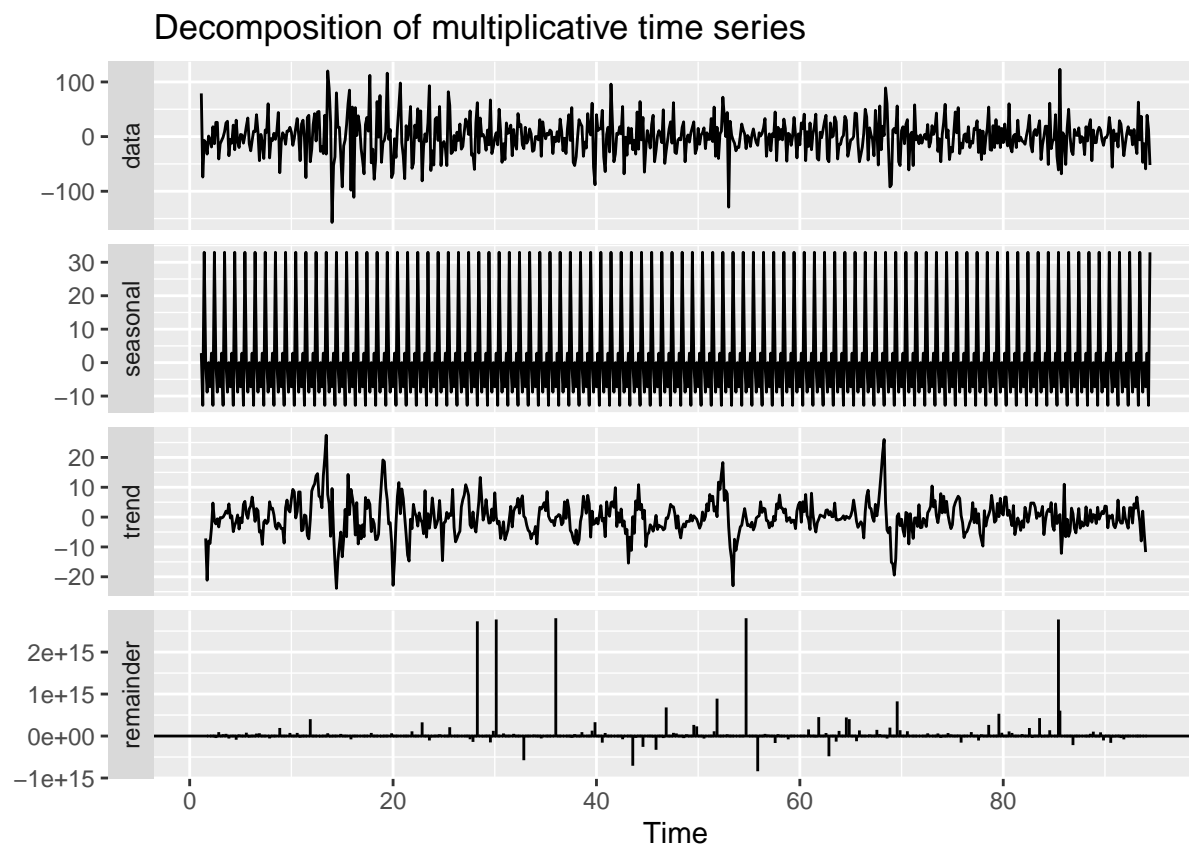
```
dec <- decompose(data.ts)
acf(na.omit(dec$random), lag.max = 30)
```


Series na.omit(dec\$random)



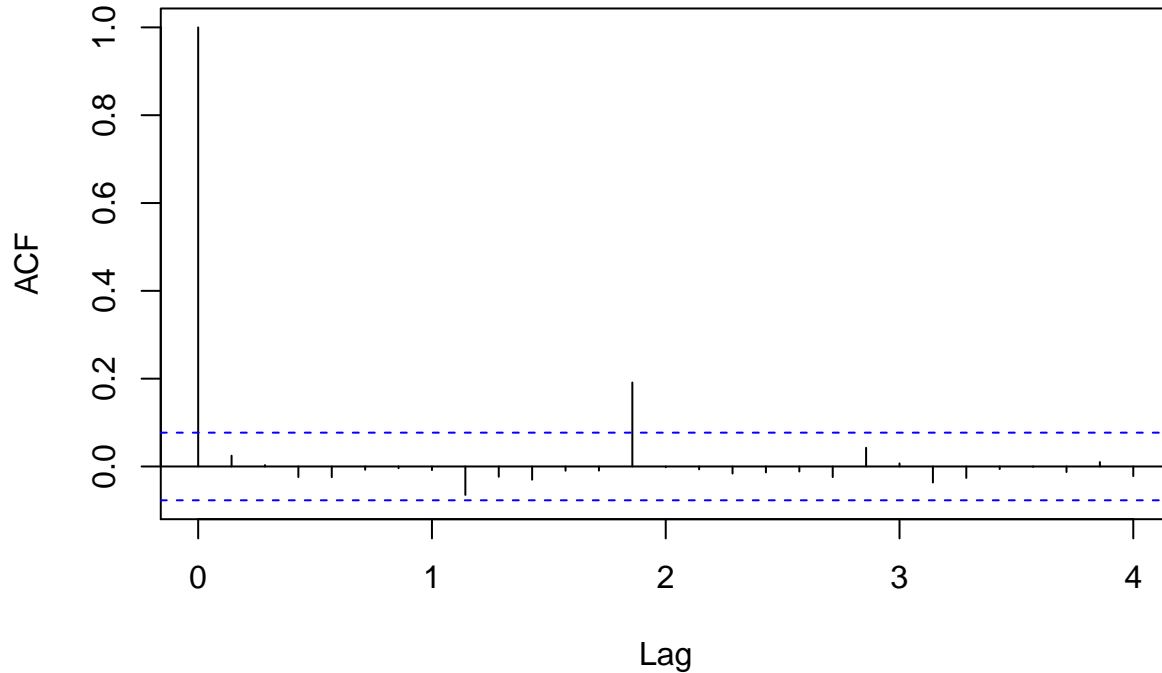
trying with differentiation and a multiplicative model:

```
differenced %>% decompose(type="multiplicative") %>% autoplot()
```



```
dec <- decompose(differenced,type="multiplicative")
acf(na.omit(dec$random))
```

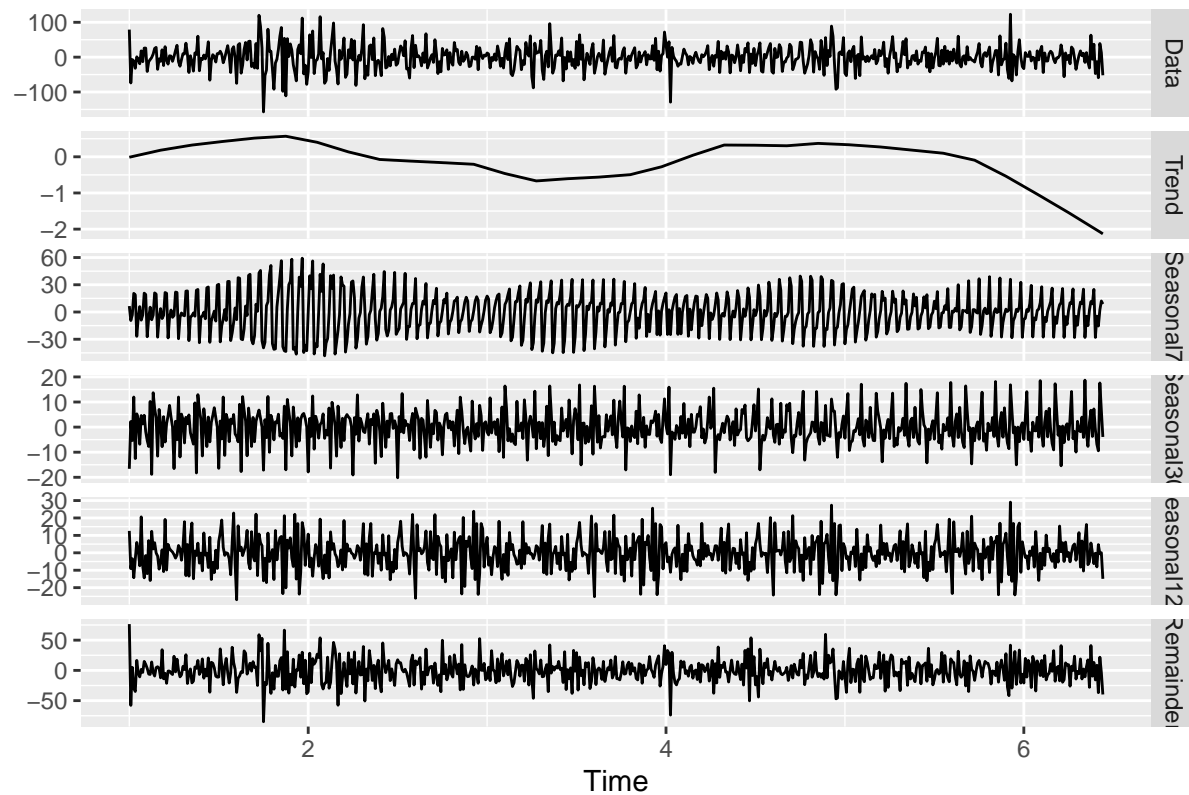
Series na.omit(dec\$random)



Looks better than before but we can still see every 5(*7) a seasonality/trend left. 5*7 is about a month, probably there is a monthly seasonality

11 Transforming into msts

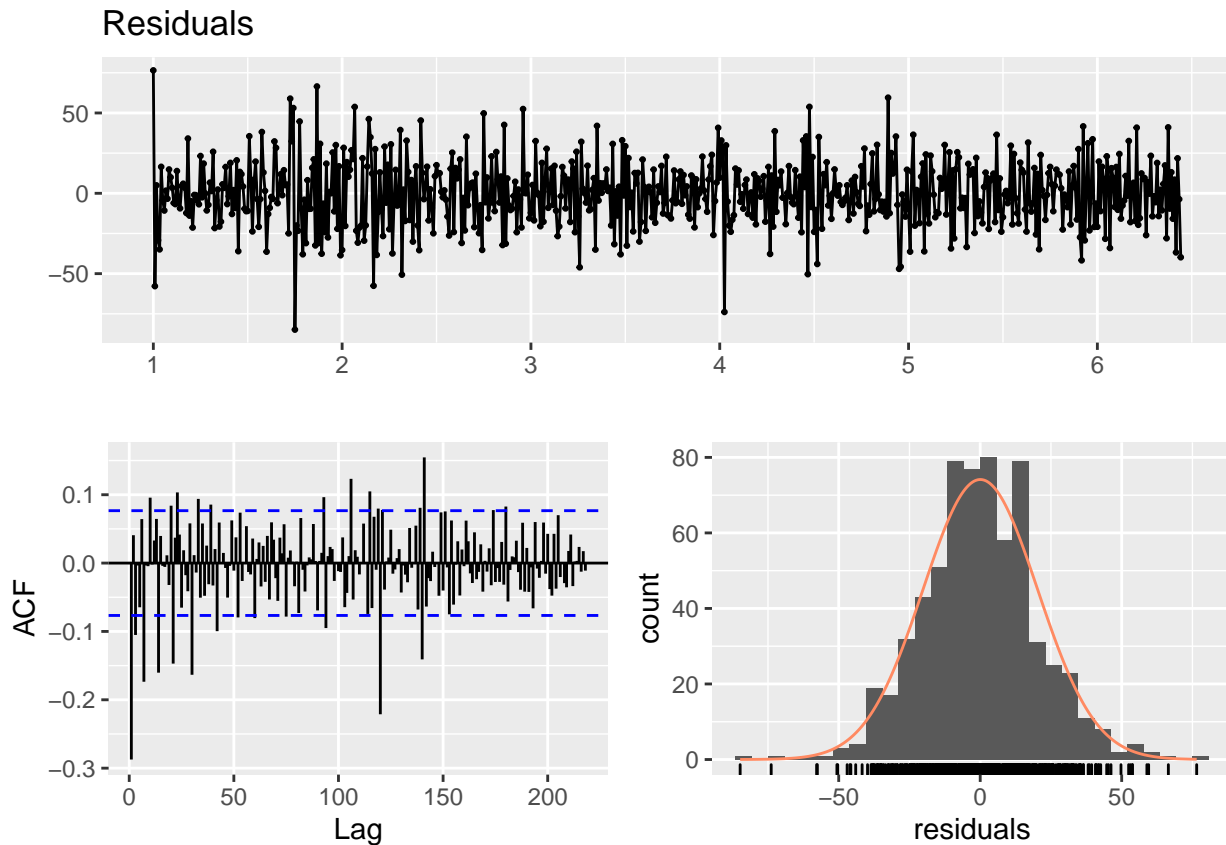
```
require(forecast)
msts_cons <- msts(as.data.frame(diff(data.ts)), seasonal.periods = c(7, 30, 4*30))
msts_cons %>% mstl() %>% autoplot()
```



```
decomposed <- mstl(msts_cons)
```

```
checkresiduals(remainder(decomposed))
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```



```
Box.test(remainder(decomposed), lag=5, fitdf=0)
```

```
##
## Box-Pierce test
##
## data: remainder(decomposed)
## X-squared = 65.062, df = 5, p-value = 1.088e-12
```

```
Box.test(remainder(decomposed), lag=5, fitdf=0, type="Lj")
```

```
##
## Box-Ljung test
##
## data: remainder(decomposed)
## X-squared = 65.402, df = 5, p-value = 9.248e-13
```

12 Conclusions

It was really interesting!

13 TODO

prima diff, poi prima diff seasonal, check acf pacf, check no trend(trend se con decadono a 0 velocemente)
 identificare i picchi identificare l'estate doppia seasonality una settimanale e una annuale ARCH GARCH
 VAR<— stabilizzare con trasformazioni