autoregression analysis

Luca Comba

3/22/2021

Read dataset

```
fargo <- read.csv("C:/Users/Luca/Desktop/SPRING2021/Statistical Research/Correct Dataset/fargo.c
sv", encoding="UTF-8")
grand <- read.csv("C:/Users/Luca/Desktop/SPRING2021/Statistical Research/Correct Dataset/grand.c
sv", encoding="UTF-8")
```

Creating time series

Now for each of the variable we are creating a time series.

For the discharge.

```
# Discharge
grand_discharge_ts <- ts(grand$discharge,
                         start = 2014,
                         end = 2020,
                         frequency = 365)

fargo_discharge_ts <- ts(fargo$discharge,
                         start = 2014,
                         end = 2020,
                         frequency = 365)
```

For the average temperature.

```
# Average Temperature
grand_temp_ts <- ts(grand$avg.temp,
                         start = 2014,
                         end = 2020,
                         frequency = 365)

fargo_temp_ts <- ts(fargo$avg.temp,
                         start = 2014,
                         end = 2020,
                         frequency = 365)
```

For the Evapotranspiration.

```
# Evapotranspiration
grand_pen_ts <- ts(grand$penman.pet,
                         start = 2014,
                         end = 2020,
                         frequency = 365)

fargo_pen_ts <- ts(fargo$penman.pet,
                         start = 2014,
                         end = 2020,
                         frequency = 365)
```

For the rainfall.

```
# Rainfall
grand_rainfall_ts <- ts(grand$rainfall,
                         start = 2014,
                         end = 2020,
                         frequency = 365)

fargo_rainfall_ts <- ts(fargo$rainfall,
                         start = 2014,
                         end = 2020,
                         frequency = 365)
```

For the snow depth

```
# Snow Depth
grand_snow_ts <- ts(grand$snow.depth,
                         start = 2014,
                         end = 2020,
                         frequency = 365)

fargo_snow_ts <- ts(fargo$snow.depth,
                         start = 2014,
                         end = 2020,
                         frequency = 365)
```

For the top soil temperature.

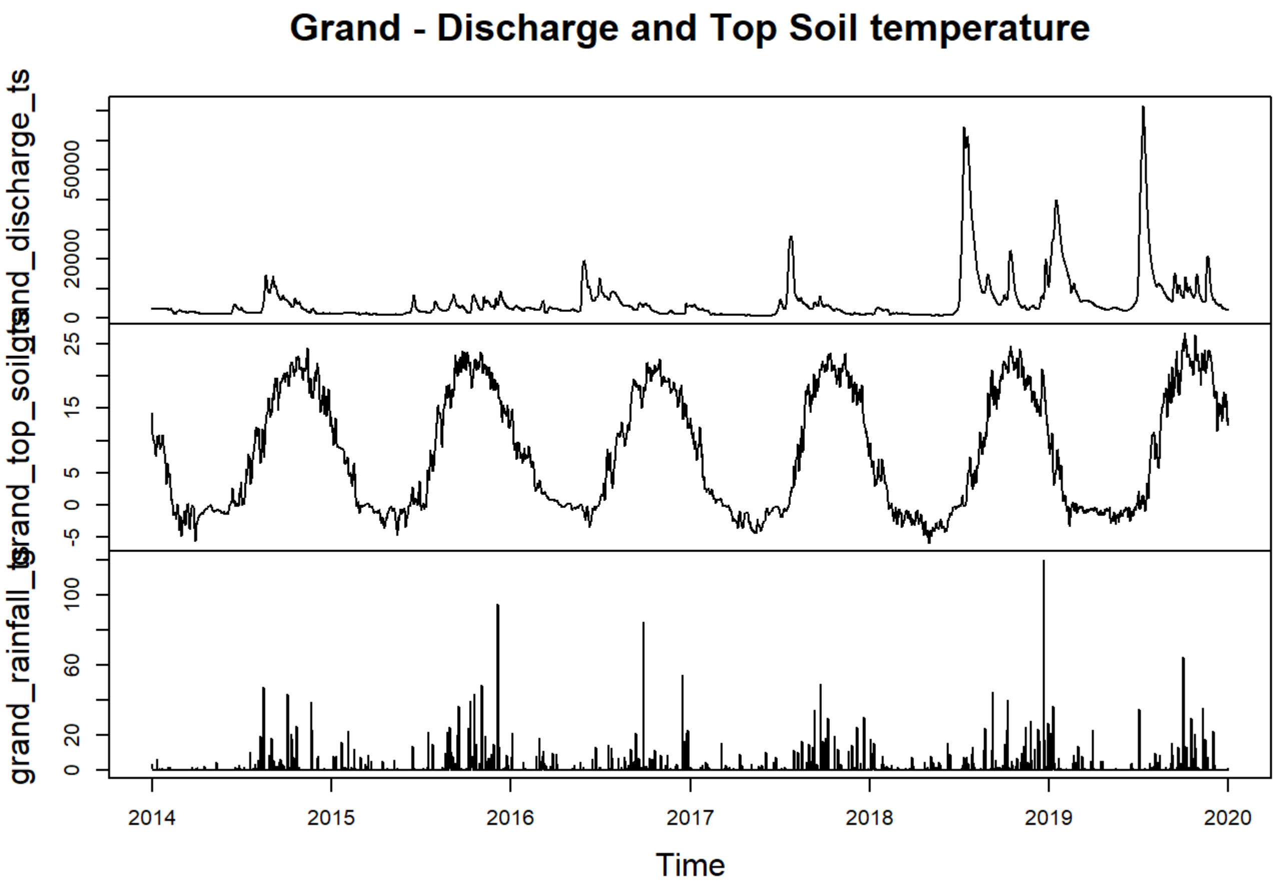
```
# Top Soil temperature
grand_top_soil_ts <- ts(grand$top.soil.temp,
                         start = 2014,
                         end = 2020,
                         frequency = 365)

fargo_top_soil_ts <- ts(fargo$top.soil.temp,
                         start = 2014,
                         end = 2020,
                         frequency = 365)
```

Plotting some time-series

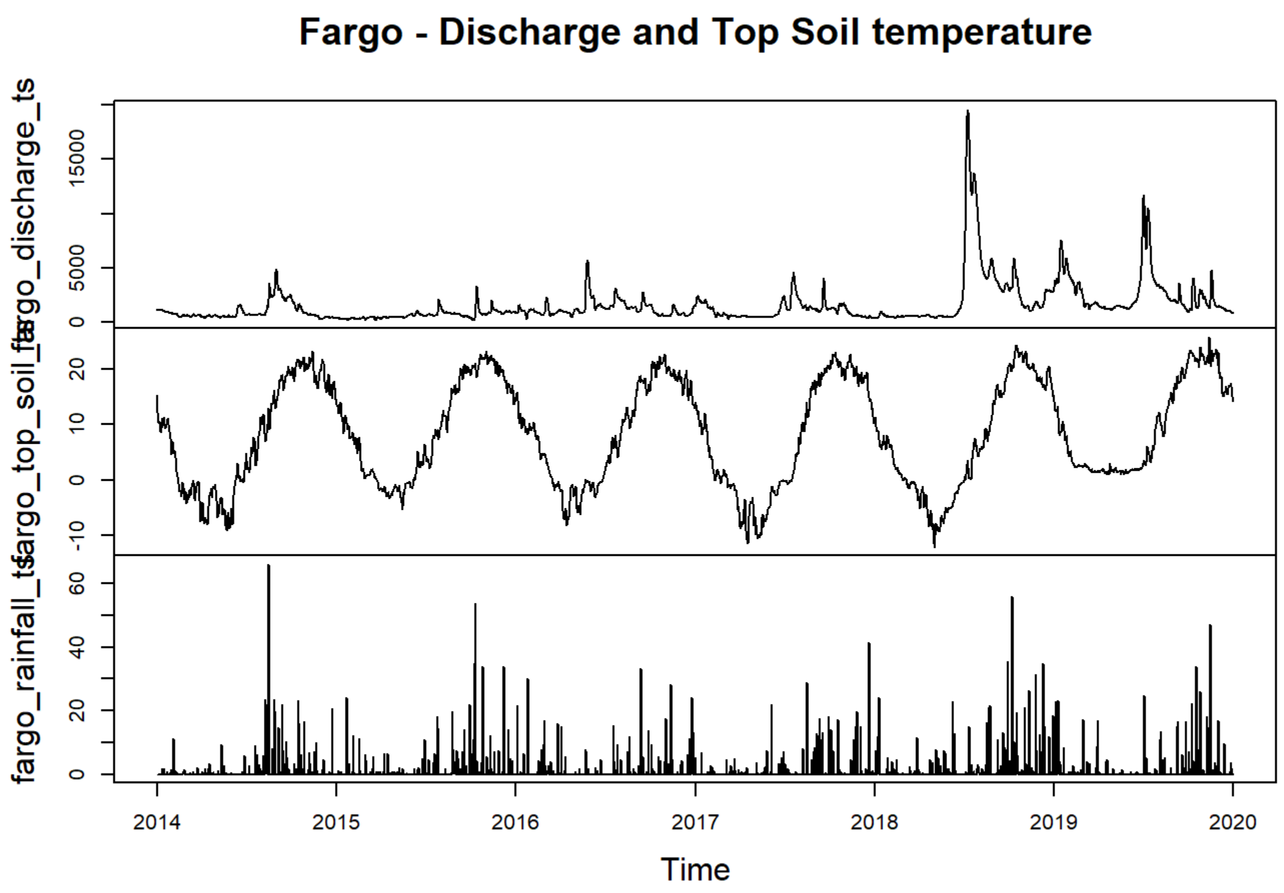
This is for Grand Forks

```
plot(cbind(grand_discharge_ts,grand_top_soil_ts,grand_rainfall_ts), main = "Grand - Discharge an
d Top Soil temperature")
```



This is for Fargo

```
plot(cbind(fargo_discharge_ts,fargo_top_soil_ts,fargo_rainfall_ts), main = "Fargo - Discharge an
d Top Soil temperature")
```

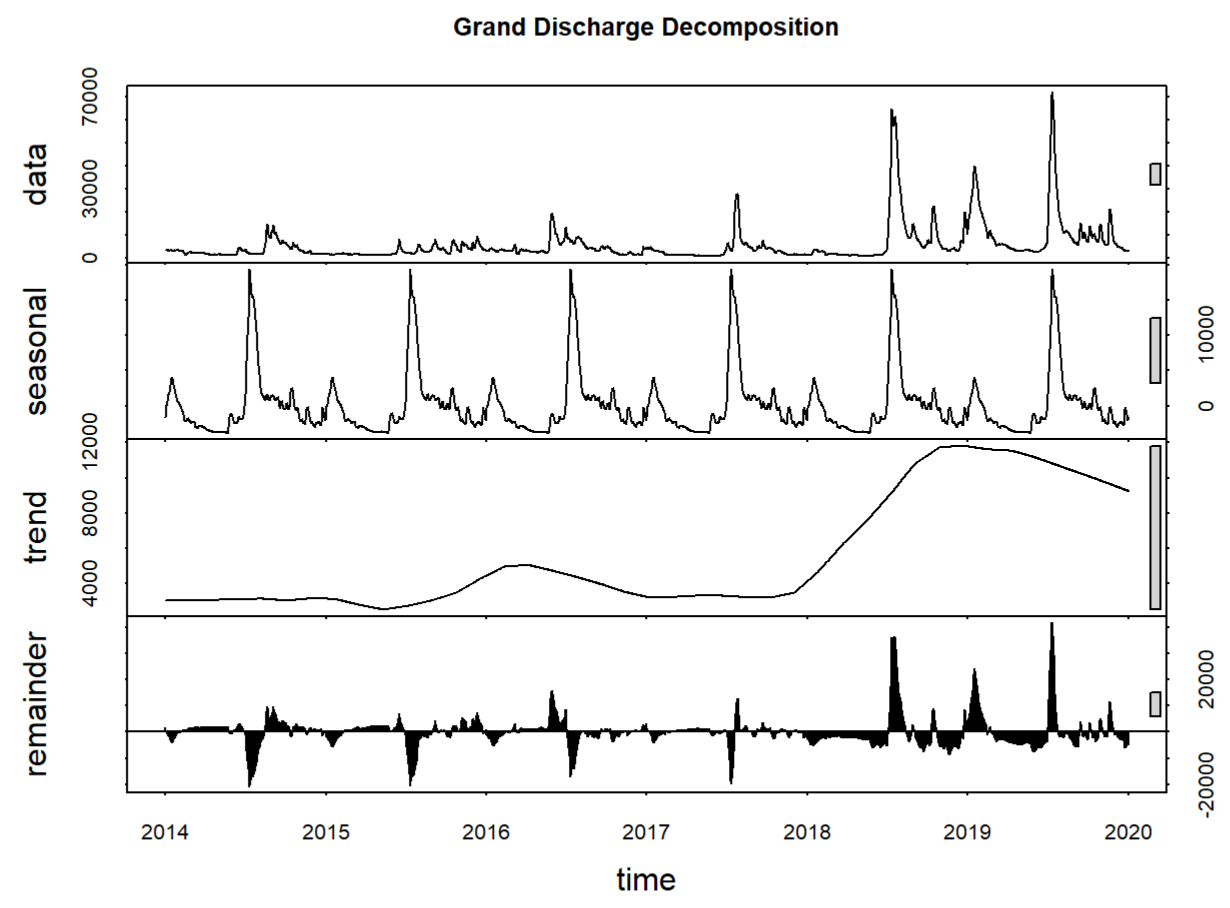


Seasonal Decomposition

Because our data shows that the time seires have a seasonal trend, then components of the time seires can be decomposed. In R we can do it with the `stl()` method.

Here is an exmple using Grand’s discharge time-series.

```
grand_discharge_fit <- stl(grand_discharge_ts, s.window="period")
plot(grand_discharge_fit, main = "Grand Discharge Decomposition")
```



Here is the decomposition for the rest of the data.

```
# Grand
grand_pen_fit <- stl(grand_pen_ts, s.window="period")
grand_rainfall_fit <- stl(grand_rainfall_ts, s.window="period")
grand_snow_fit <- stl(grand_snow_ts, s.window="period")
grand_temp_fit <- stl(grand_temp_ts, s.window="period")
grand_top_soil_fit <- stl(grand_top_soil_ts, s.window="period")

# Fargo
fargo_discharge_fit <- stl(fargo_discharge_ts, s.window="period")
fargo_pen_fit <- stl(fargo_pen_ts, s.window="period")
fargo_rainfall_fit <- stl(fargo_rainfall_ts, s.window="period")
fargo_snow_fit <- stl(fargo_snow_ts, s.window="period")
fargo_temp_fit <- stl(fargo_temp_ts, s.window="period")
fargo_top_soil_fit <- stl(fargo_top_soil_ts, s.window="period")
```

Another way to decompose the Time-series is to use `decompose`

```
# Grand
grand_discharge_fit <- decompose(grand_discharge_ts)
grand_pen_fit <- decompose(grand_pen_ts)
grand_rainfall_fit <- decompose(grand_rainfall_ts)
grand_snow_fit <- decompose(grand_snow_ts)
grand_temp_fit <- decompose(grand_temp_ts)
grand_top_soil_fit <- decompose(grand_top_soil_ts)

# Fargo
fargo_discharge_fit <- decompose(fargo_discharge_ts)
fargo_pen_fit <- decompose(fargo_pen_ts)
fargo_rainfall_fit <- decompose(fargo_rainfall_ts)
fargo_snow_fit <- decompose(fargo_snow_ts)
fargo_temp_fit <- decompose(fargo_temp_ts)
fargo_top_soil_fit <- decompose(fargo_top_soil_ts)
```

Vector Autoregression Analysis

After having uploaded the dataset and the correct libraries is time for creating time series objects.

We would need to check that has been completed correctly by plotting the data.

Autocorrelation

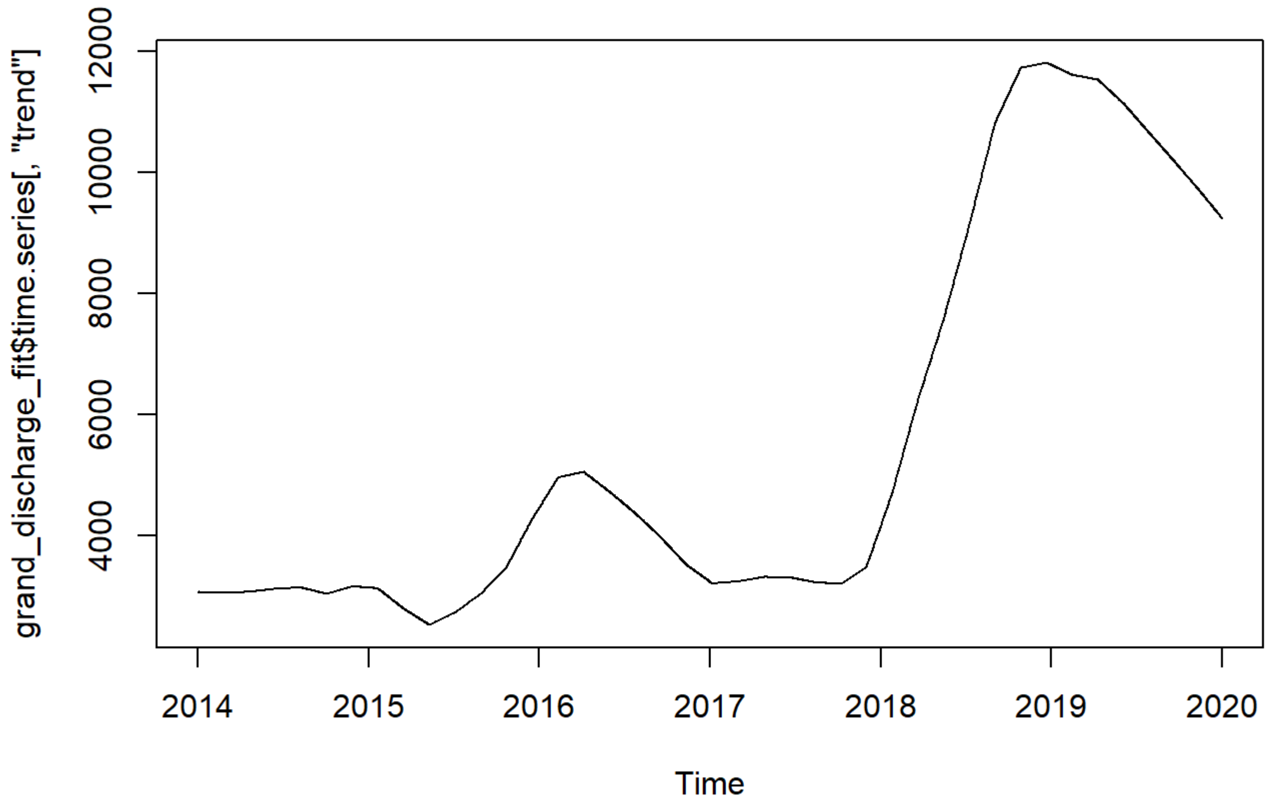
we would now need to check for autocorrelation.

```
head(grand_discharge_fit$time.series)
```

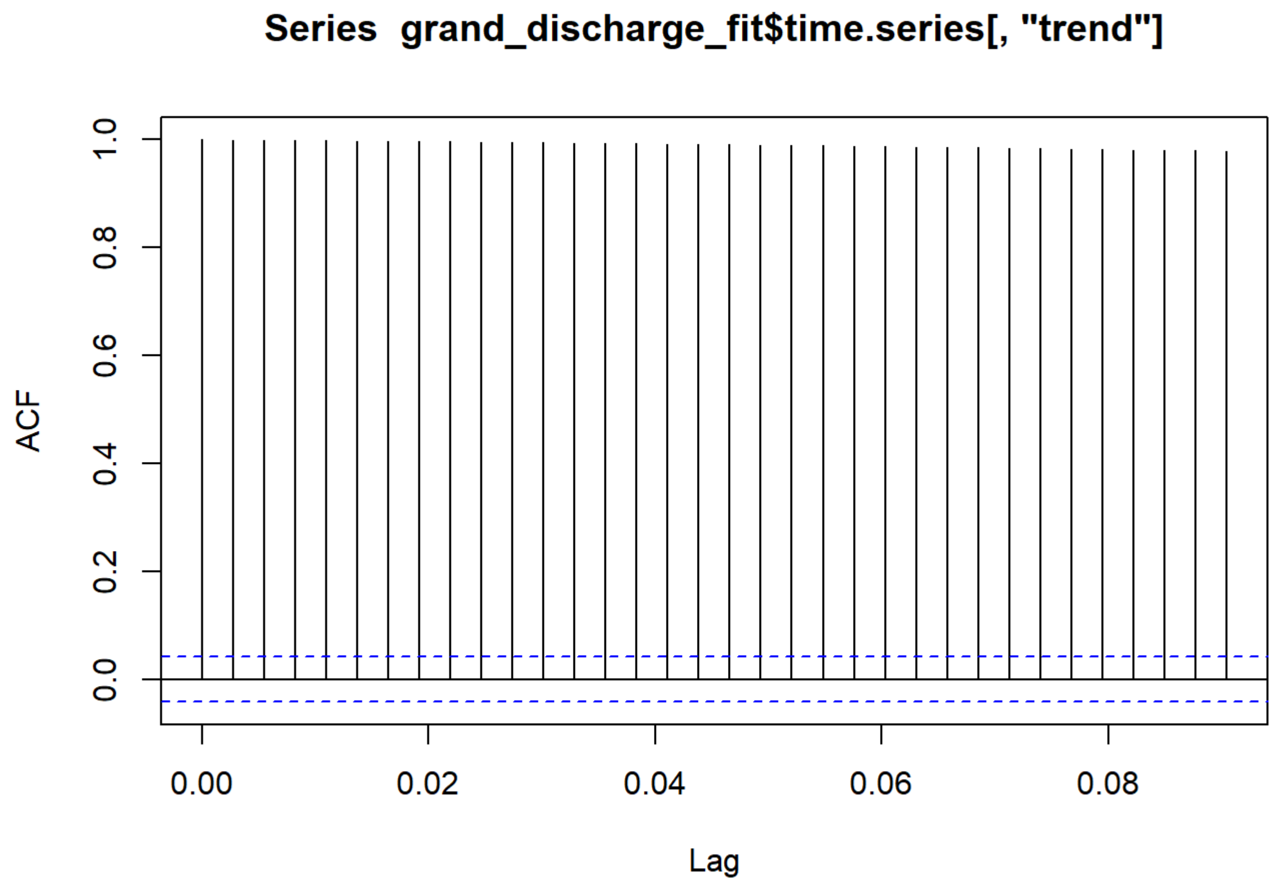
```
##         seasonal    trend  remainder
## [1,] -1587.69165 3066.106 1831.58573
## [2,]  -395.95636 3065.850  570.10663
## [3,]   -90.06571 3065.594  334.47217
## [4,]    27.49160 3065.337  187.17105
## [5,]   250.04891 3065.081  -75.13007
## [6,]   615.93956 3064.825 -440.76453
```

Trends in the data for the Grand’s discharge:

```
plot(grand_discharge_fit$time.series[,"trend"])
```



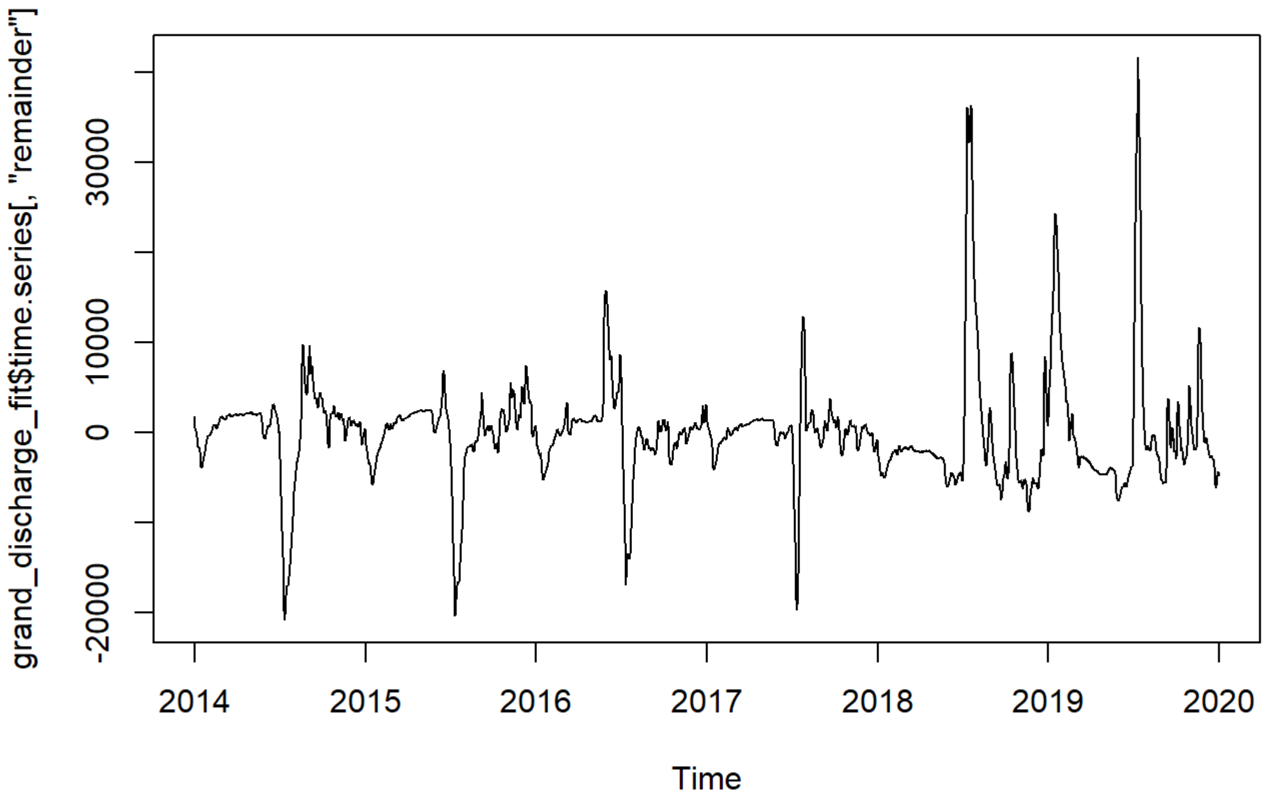
```
acf(grand_discharge_fit$time.series[,"trend"])
```



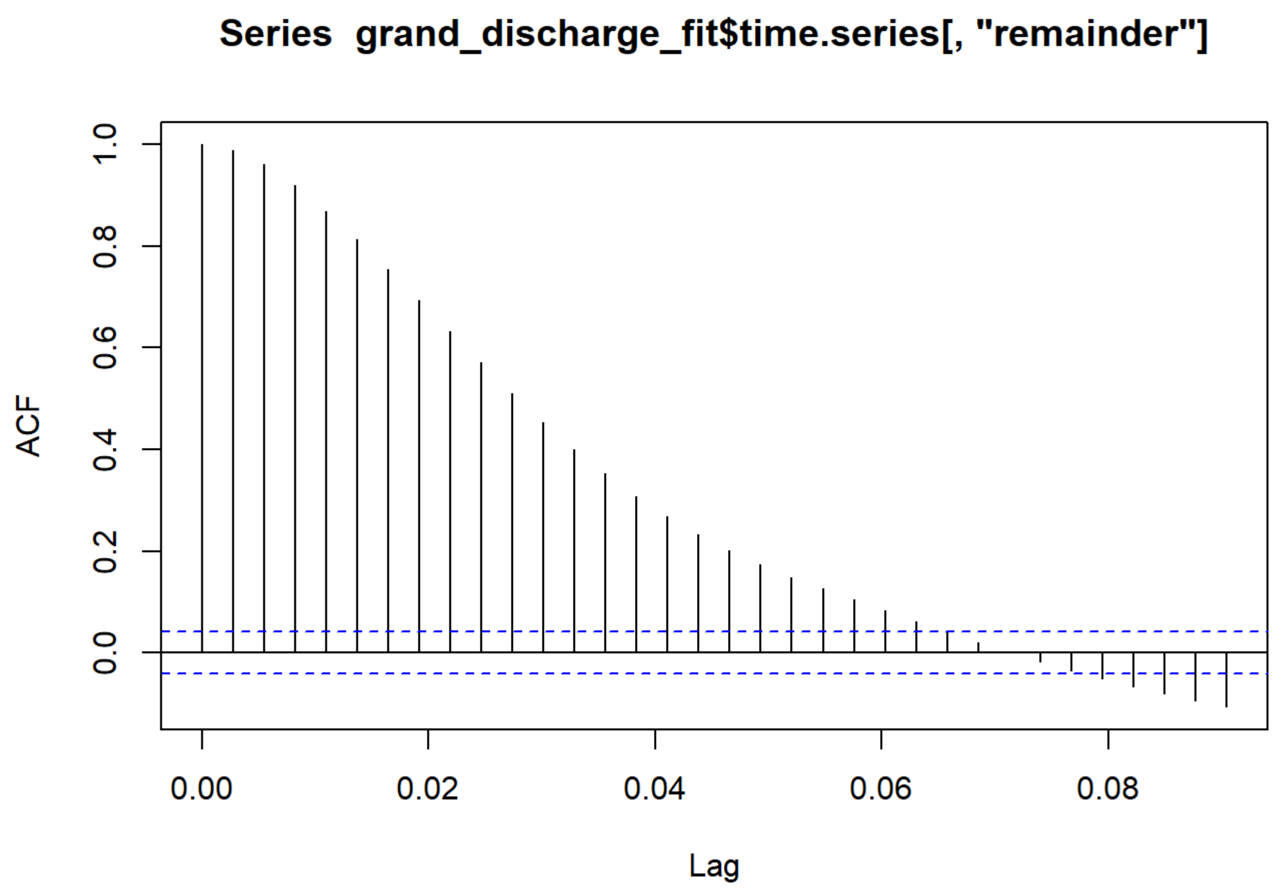
For proceding to take out the seasonality, which means that after using decomposition into three components: trend, seasonal, and irregular, and then *apply VAR to the residuals (irregular components of the time series).*

The *Remainder* column in the Seasonal Decomposition of Time Series by Loess can be plotted and use it for the Vector Autoregression analysis.

```
plot(grand_discharge_fit$time.series[,"remainder"])
```



```
acf(grand_discharge_fit$time.series[,"remainder"])
```



Then the residual, or called remainder, can be save into the fitted variables and used for the analysis.

```
# Grand
grand_discharge_fit = grand_discharge_fit$time.series[,"remainder"]
grand_pen_fit = grand_pen_fit$time.series[,"remainder"]
grand_rainfall_fit = grand_rainfall_fit$time.series[,"remainder"]
grand_snow_fit = grand_snow_fit$time.series[,"remainder"]
grand_temp_fit = grand_temp_fit$time.series[,"remainder"]
grand_top_soil_fit = grand_top_soil_fit$time.series[,"remainder"]

# Fargo
fargo_discharge_fit = fargo_discharge_fit$time.series[,"remainder"]
fargo_pen_fit = fargo_pen_fit$time.series[,"remainder"]
fargo_rainfall_fit = fargo_rainfall_fit$time.series[,"remainder"]
fargo_snow_fit = fargo_snow_fit$time.series[,"remainder"]
fargo_temp_fit = fargo_temp_fit$time.series[,"remainder"]
fargo_top_soil_fit = fargo_top_soil_fit$time.series[,"remainder"]
```

Analysis

The analysis will start with the Grand Fork data. I have followed closely the tutorial on kevinkotze.github.io (https://kevinkotze.github.io/ts-7-tut/)

Analysis for Grand Forks

First we can try the set of variables Discharge and Top Soil.

```
# Selection of the variables
grand_bind_dis_top <- cbind(grand_discharge_fit, grand_top_soil_fit)
colnames(grand_bind_dis_top) <- c("discharge", "soil")
```

For the model selection and estimation we can use the VARselect method.

```
# Set up the analysis
grand_model <- VARselect(grand_bind_dis_top, lag.max = 12, type = "const")
grand_model$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      6      6      5      6
```

Because it seems that the best lag is 6 that would mean that the expected best p is 6.

```
grand_est <- VAR(grand_bind_dis_top, p = 6, type = "const", season = NULL,
    exog = NULL)
summary(grand_est)
```

```
## 
## VAR Estimation Results:
## ========================= 
## Endogenous variables: discharge, soil 
## Deterministic variables: const 
## Sample size: 2185 
## Log Likelihood: -18569.701 
## Roots of the characteristic polynomial:
## 0.9037 0.9037 0.8436 0.6197 0.6197 0.5761 0.5761 0.4926 0.4926 0.4802 0.4802 0.4194
## Call:
## VAR(y = grand_bind_dis_top, p = 6, type = "const", exogen = NULL)
## 
## 
## Estimation results for equation discharge: 
## ========================================== 
## discharge = discharge.l1 + soil.l1 + discharge.l2 + soil.l2 + discharge.l3 + soil.l3 + discha
rge.l4 + soil.l4 + discharge.l5 + soil.l5 + discharge.l6 + soil.l6 + const 
## 
##               Estimate Std. Error t value Pr(>|t|)    
## discharge.l1   2.38359    0.02136 111.565  < 2e-16 ***
## soil.l1      -10.99502   10.52749  -1.044   0.2964    
## discharge.l2  -2.31004    0.05491 -42.073  < 2e-16 ***
## soil.l2        0.71405   16.07047   0.044   0.9646    
## discharge.l3   1.44571    0.07198  20.085  < 2e-16 ***
## soil.l3        8.02230   16.71181   0.480   0.6312    
## discharge.l4  -0.78785    0.07195 -10.949  < 2e-16 ***
## soil.l4       11.23022   16.71221   0.672   0.5017    
## discharge.l5   0.35039    0.05486   6.388 2.06e-10 ***
## soil.l5      -27.29582   16.05964  -1.700   0.0893 .  
## discharge.l6  -0.09569    0.02136  -4.480 7.86e-06 ***
## soil.l6       14.75591   10.48958   1.407   0.1597    
## const          0.02526    8.05600   0.003   0.9975    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## 
## Residual standard error: 376.5 on 2172 degrees of freedom
## Multiple R-Squared: 0.9961,  Adjusted R-squared: 0.9961 
## F-statistic: 4.649e+04 on 12 and 2172 DF,  p-value: < 2.2e-16 
## 
## 
## Estimation results for equation soil: 
## ===================================== 
## soil = discharge.l1 + soil.l1 + discharge.l2 + soil.l2 + discharge.l3 + soil.l3 + discharge.l
4 + soil.l4 + discharge.l5 + soil.l5 + discharge.l6 + soil.l6 + const 
## 
##                Estimate Std. Error t value Pr(>|t|)    
## discharge.l1 -1.792e-05  4.358e-05  -0.411 0.680984    
## soil.l1       1.153e+00  2.147e-02  53.709  < 2e-16 ***
## discharge.l2  5.728e-05  1.120e-04   0.512 0.609033    
## soil.l2      -4.344e-01  3.278e-02 -13.253  < 2e-16 ***
## discharge.l3 -6.483e-05  1.468e-04  -0.442 0.658833    
## soil.l3       1.210e-01  3.409e-02   3.550 0.000393 ***
## discharge.l4  2.749e-05  1.468e-04   0.187 0.851408
## soil.l4       2.750e-02  3.409e-02   0.807 0.419851
## discharge.l5  7.300e-06  1.119e-04   0.065 0.947984
## soil.l5      -3.364e-02  3.276e-02  -1.027 0.304477
## discharge.l6 -1.772e-05  4.357e-05  -0.407 0.684191
## soil.l6       2.427e-02  2.139e-02   1.134 0.256720
## const        -1.574e-03  1.643e-02  -0.096 0.923700
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.768 on 2172 degrees of freedom
## Multiple R-Squared: 0.7863,  Adjusted R-squared: 0.7851
## F-statistic: 666.1 on 12 and 2172 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##            discharge    soil
## discharge 141788.449 -7.0460
## soil          -7.046  0.5899
##
## Correlation matrix of residuals:
##           discharge     soil
## discharge   1.00000 -0.02436
## soil       -0.02436  1.00000
```

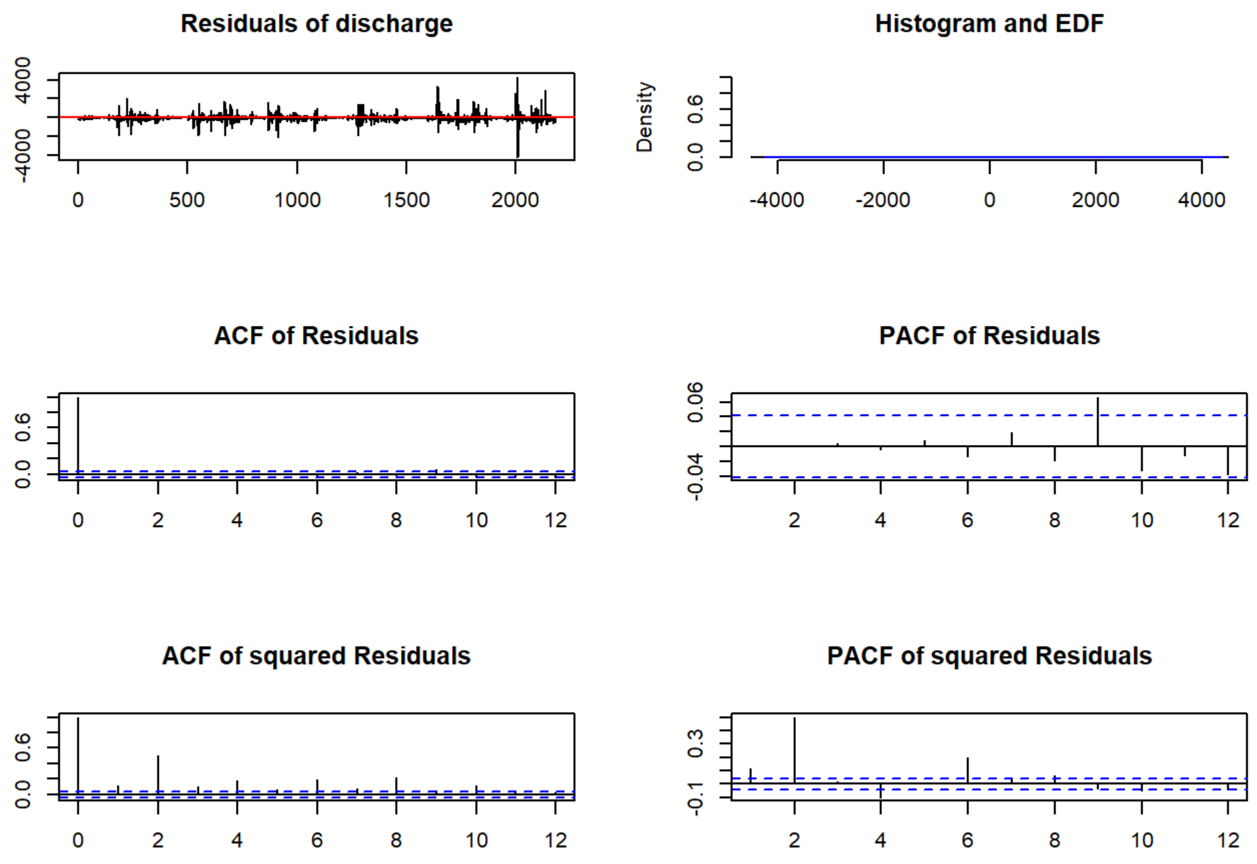
We can test the model fit with many tests for example Portmanteau-test.

One question that we might have is what would would be the best value for the `lags.pt` ?

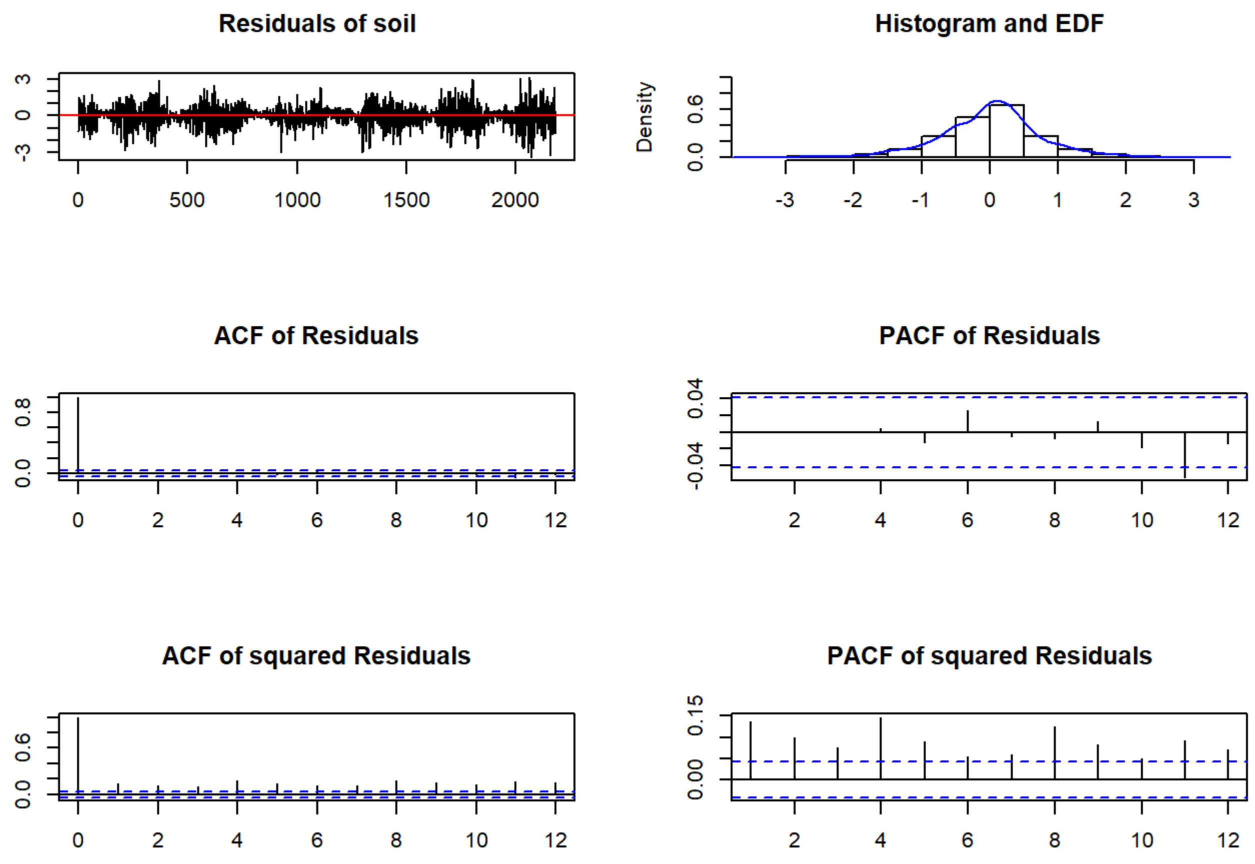
```
grand_serial <- serial.test(grand_est, lags.pt = 12, type = "PT.asymptotic")
grand_serial
```

```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object grand_est
## Chi-squared = 49.789, df = 24, p-value = 0.001506
```

```
plot(grand_serial, names = "discharge")
```



```
plot(grand_serial, names = "soil")
```



“To interpret these statistics note that if a p-value is greater than 5% would generally indicate that there is an absence of serial correlation. To test for heteroscedasticity in the residuals we can perform a multivariate ARCH Lagrange-Multiplier test.”

```
grand_arch <- arch.test(grand_est, lags.multi = 12, multivariate.only = TRUE)
grand_arch
```

```
##
##  ARCH (multivariate)
##
## data:  Residuals of VAR object grand_est
## Chi-squared = 1184.5, df = 108, p-value < 2.2e-16
```

Now we have the problem of heteroscedasticity. This means that our data has heteroscedasticity since the p value is less than 0.05.

We would also need to check for normality

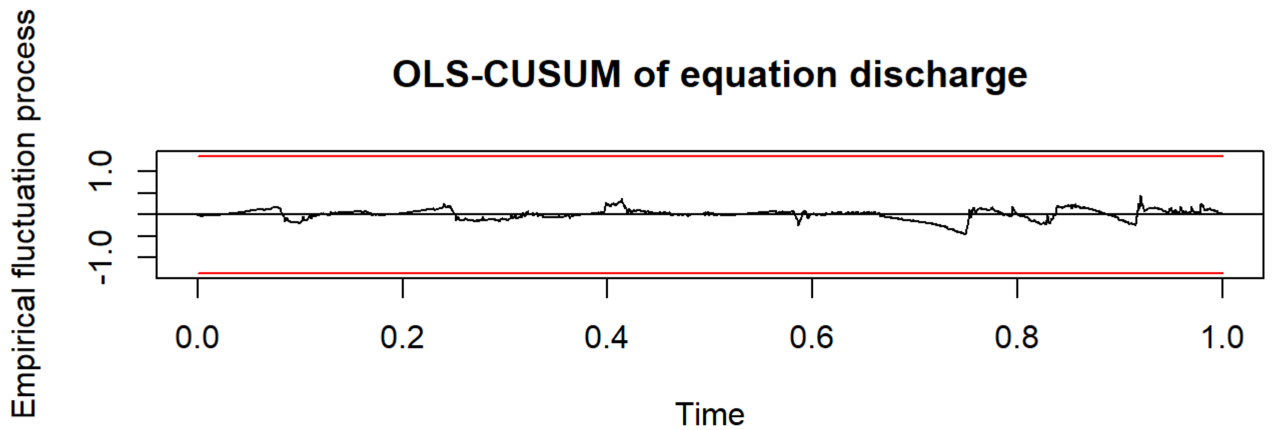
```
grand_norm <- normality.test(grand_est, multivariate.only = TRUE)
grand_norm
```

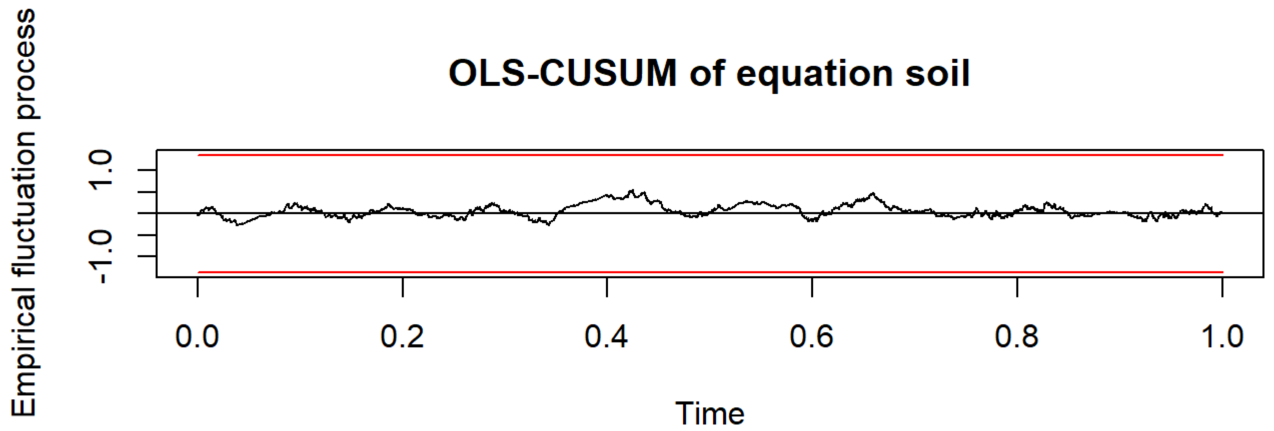
```
## $JB
##
##  JB-Test (multivariate)
##
## data:  Residuals of VAR object grand_est
## Chi-squared = 64364, df = 4, p-value < 2.2e-16
##
##
## $Skewness
##
##  Skewness only (multivariate)
##
## data:  Residuals of VAR object grand_est
## Chi-squared = 400.67, df = 2, p-value < 2.2e-16
##
##
## $Kurtosis
##
##  Kurtosis only (multivariate)
##
## data:  Residuals of VAR object grand_est
## Chi-squared = 63963, df = 2, p-value < 2.2e-16
```

In conclusion the data indicates that the residual are not normally distributed.

Then lastly to test for the structural break in the residuals we can apply a CUSUM test.

```
grand_cusum <- stability(grand_est, type = "OLS-CUSUM")
plot(grand_cusum)
```





Other methods to use are: Granger causality, IRFs and variance decompositions.

Resources used

Here is a list of some website used

- Understanding Decomposition (https://anomaly.io/seasonal-trend-decomposition-in-r/index.html)
- Documentation of stl() (https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/stl)
- Vector Autoregression Tutorial 1 (https://kevinkotze.github.io/ts-7-tut/)
- General R time series (https://www.statmethods.net/advstats/timeseries.html)

Also another question was asked on StackOverflow (https://stackoverflow.com/questions/67066604/multiple-time-series-in-r-using-ts?noredirect=1#comment118547063_67066604)