Tasks 1 and 2

2024-02-26

Setup

import libraries:

```
library(fpp2)
```

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

```
## -- Attaching packages --------------------------------------------- fpp2 2.5 --
```

```
## v ggplot2   3.4.4      v fma       2.5
## v forecast  8.21.1     v expsmooth 2.3
```

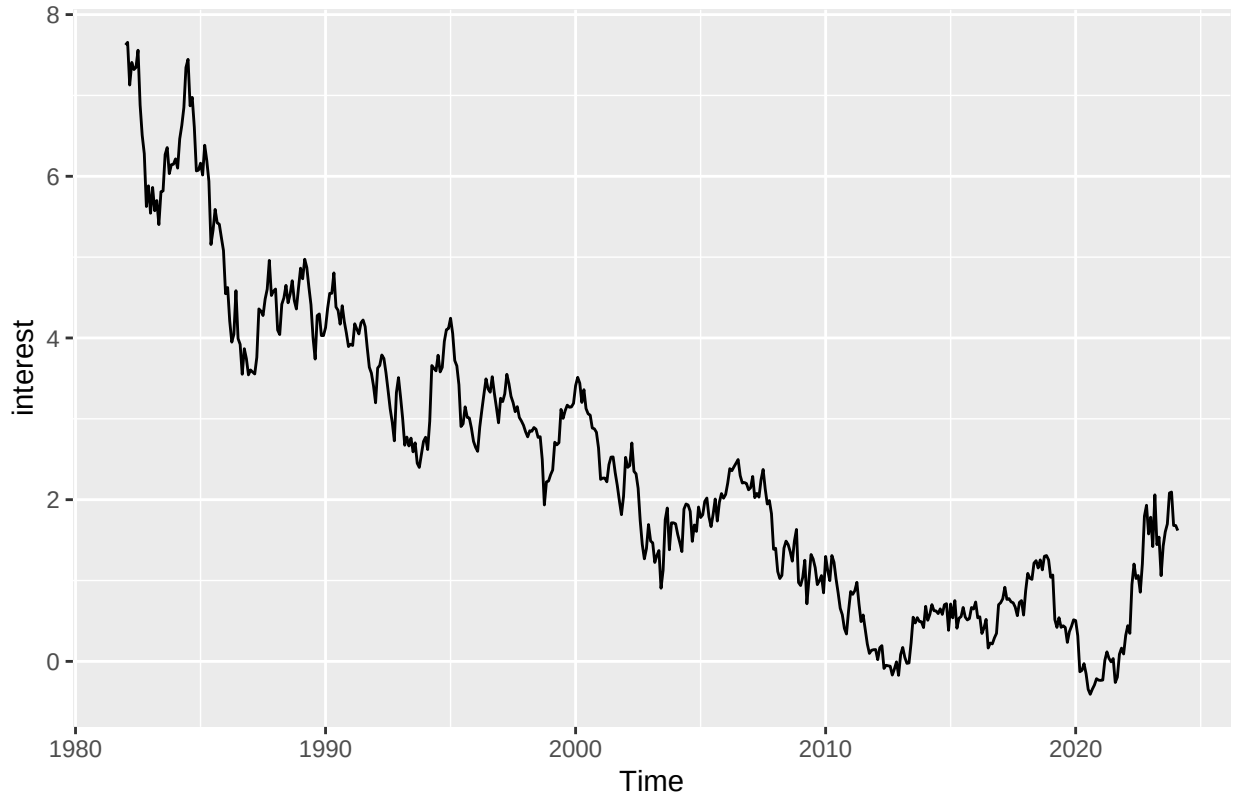
```
##
```

```
library(tseries)
library(urca)
```

Import the data:

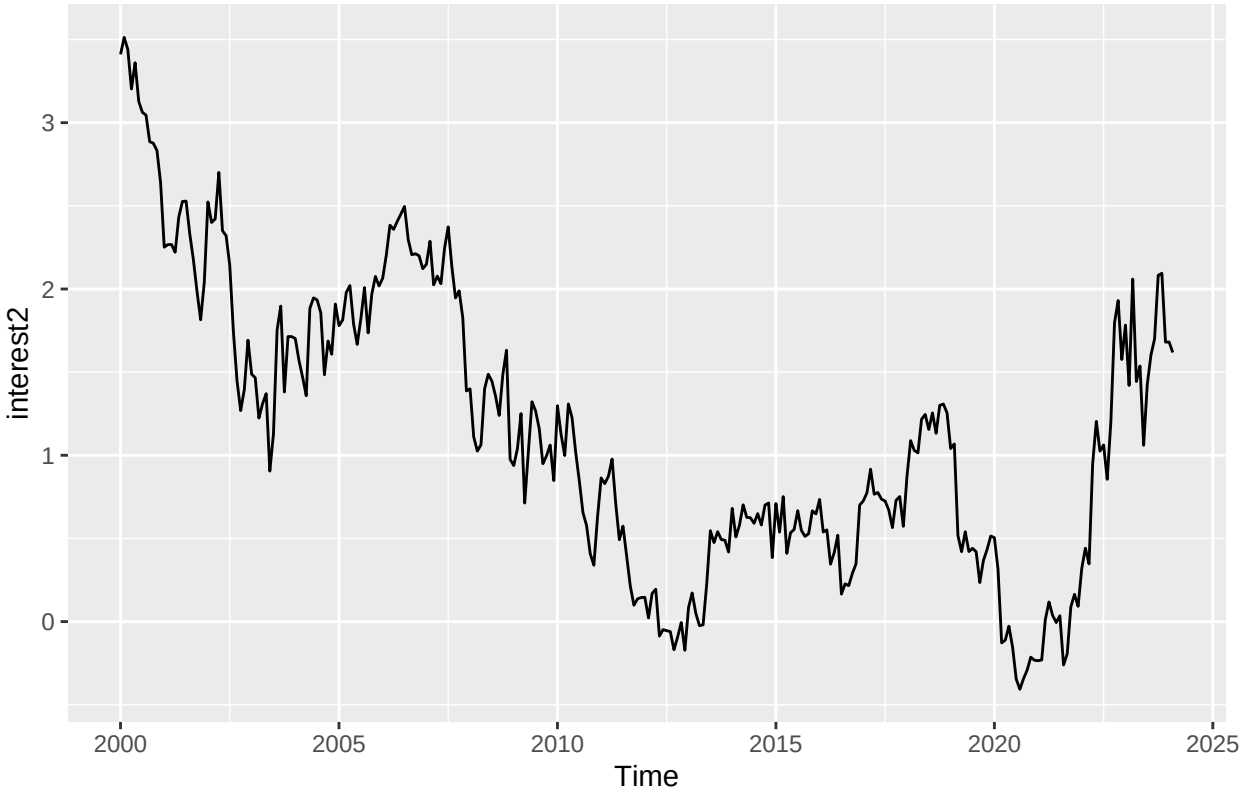
import 10 years real interest rate time series from csv (source:https://fred.stlouisfed.org/graph/?g=1hoLl):

```
REAINTRATREARAT10Y <- read.csv("C:\\Users\\ss\\Downloads\\REAINTRATREARAT10Y.csv")
interest <- ts(REAINTRATREARAT10Y[, "REAINTRATREARAT10Y"], frequency = 12, start = c(1982, 1))
autoplot(interest)
```



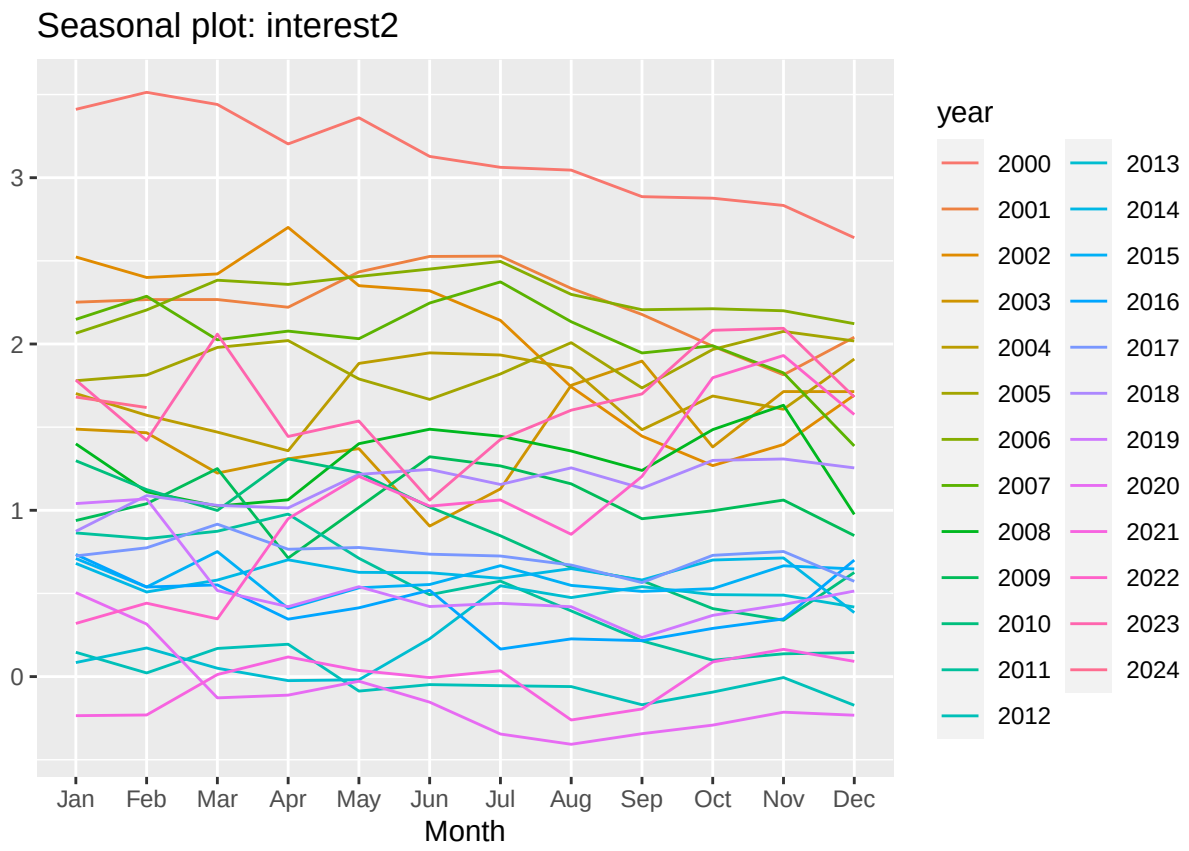
cut a window starting from year 2000

```
interest2 <- window(interest, frequency = 12, start = c(2000, 1))
autoplot(interest2)
```



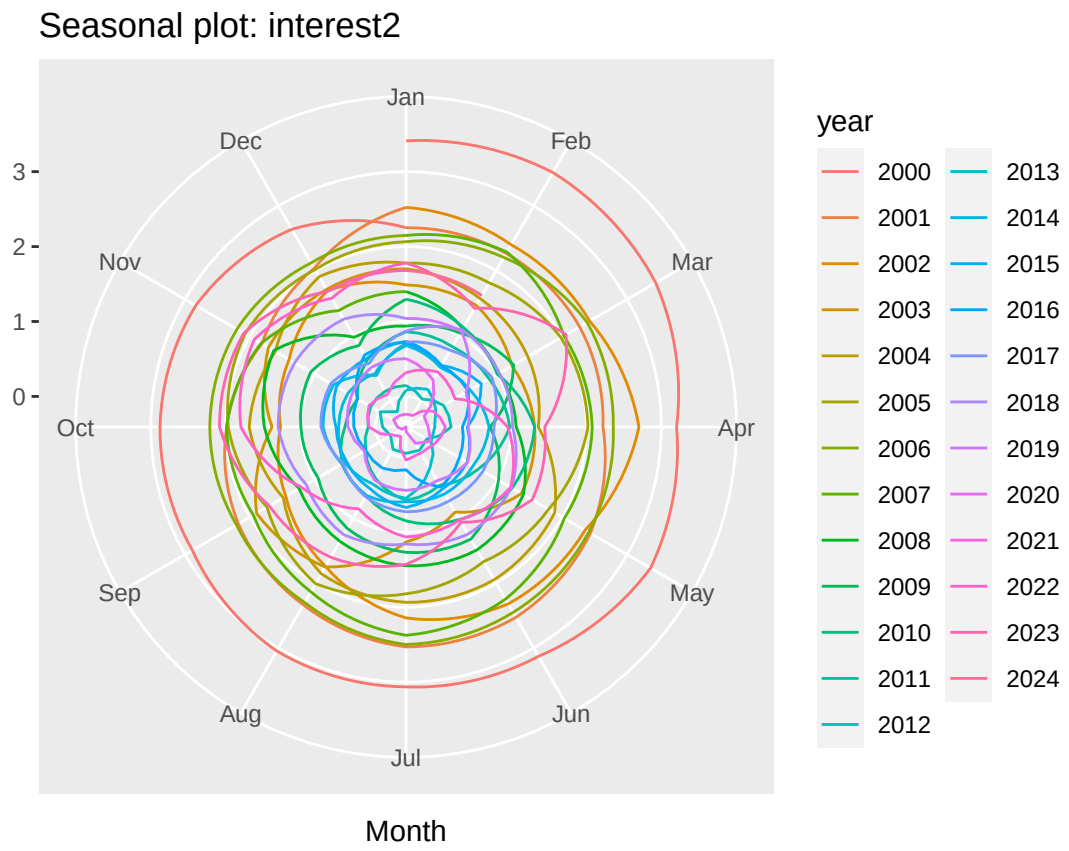
Seasonality:

```
ggseasonplot(interest2)
```

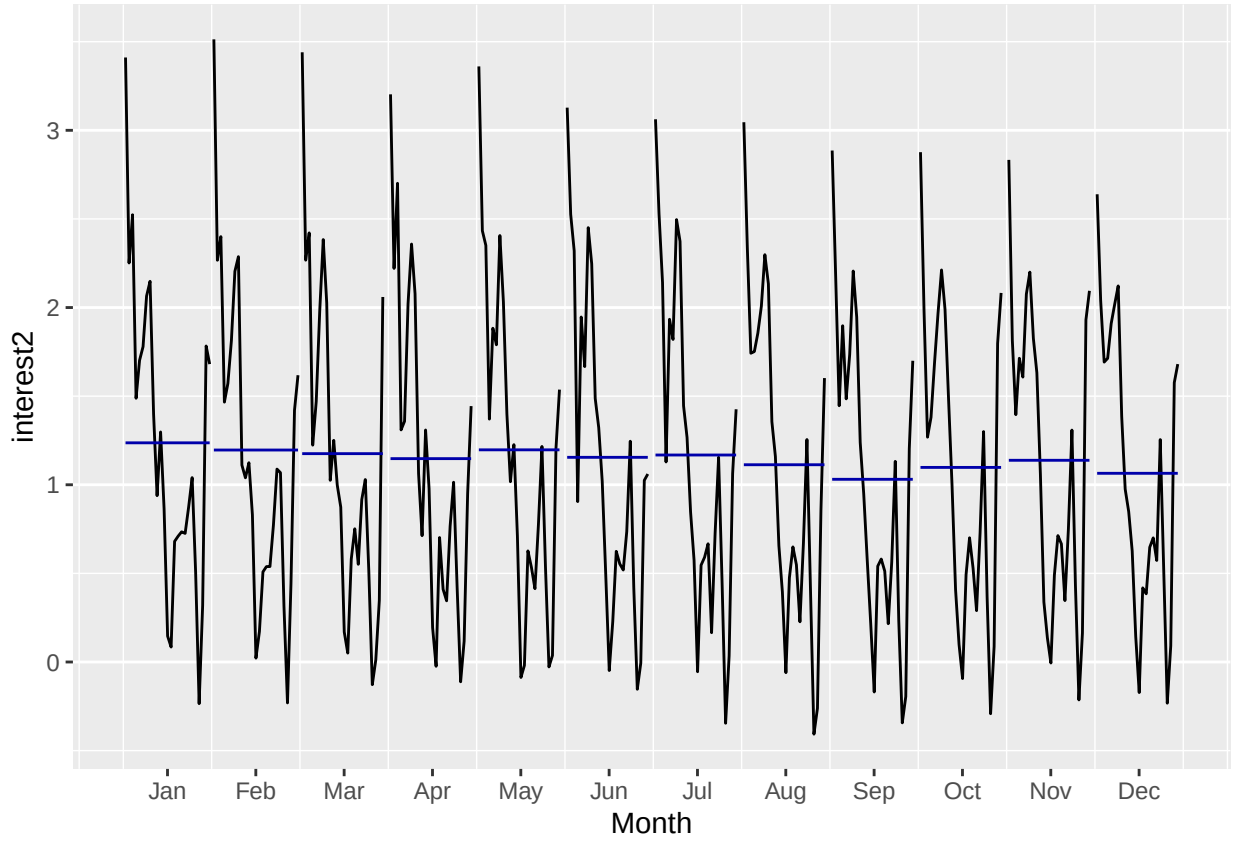


Checking polar seasonal plot

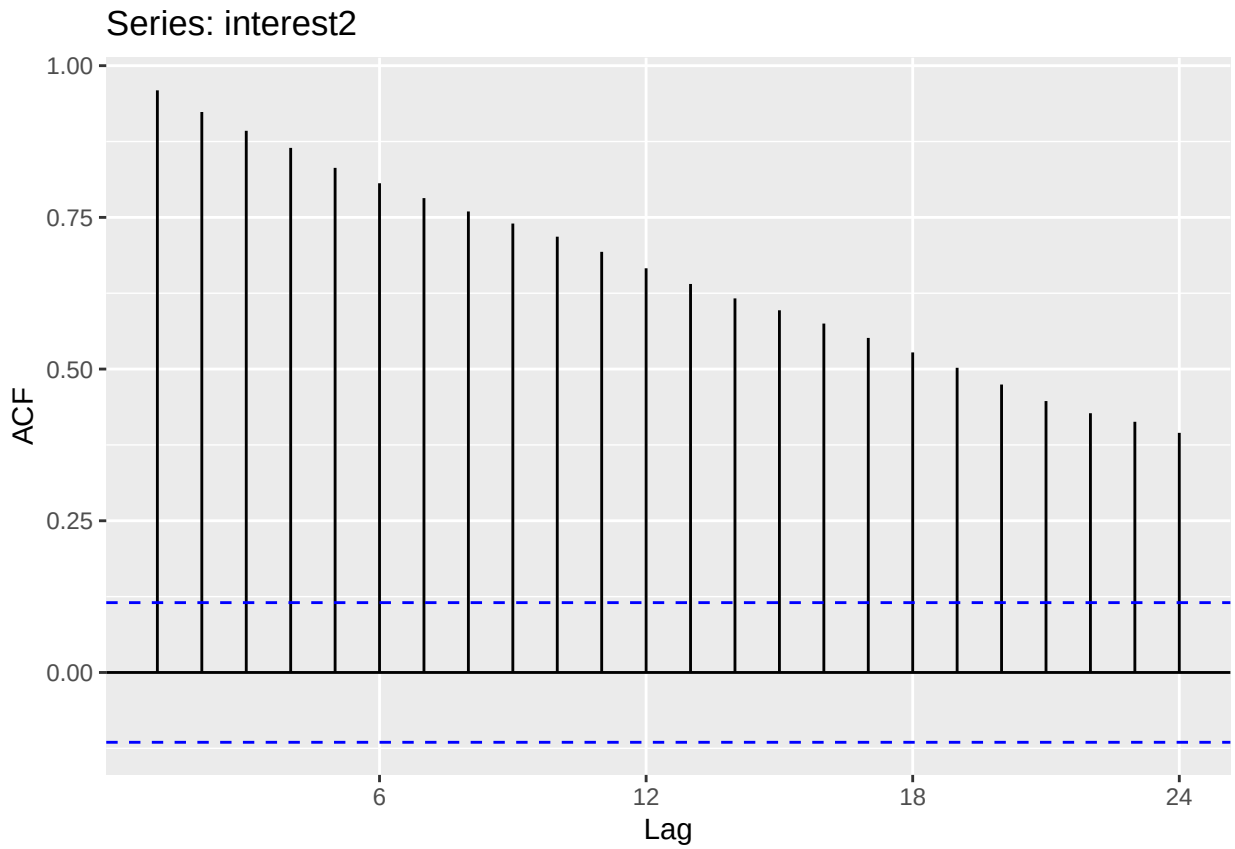
```
ggseasonplot(interest2, polar = TRUE)
```




```
ggsubseriesplot(interest2)
```




```
ggAcf(interest2)
```



Seasonal plots have no clear pattern, and nothing could be deduced out of them. The data has a roughly decreasing trend, apart from the region after the year 2020, which is probably due to the pandamic, when the investments stopped and people tended to save money in banks (in this case, banks will have high supply), and therefore the interest had a sudden decrease, after which the interest rate started to increase. In general, there are many jumps and spikes, the "ups" and "downs" in the time series are not seasonal, and their distances are not uniform, therefore they resemble a cyclic data rather than a seasonal one.

Stationarity:

augmented Dicky-Fuller test (adf) for stationarity, the assumed null hypothesis H_0 is that the data is NOT stationary

```
adf.test(interest2)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  interest2
## Dickey-Fuller = -1.8251, Lag order = 6, p-value = 0.6499
## alternative hypothesis: stationary
```

p-value is large, H_0 is not rejected (Not stationary)

```
adf.test(diff(interest2))
```

```
## Warning in adf.test(diff(interest2)): p-value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  diff(interest2)
## Dickey-Fuller = -7.4093, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

p-value is small, H_0 is rejected, the timeseries is stationary.

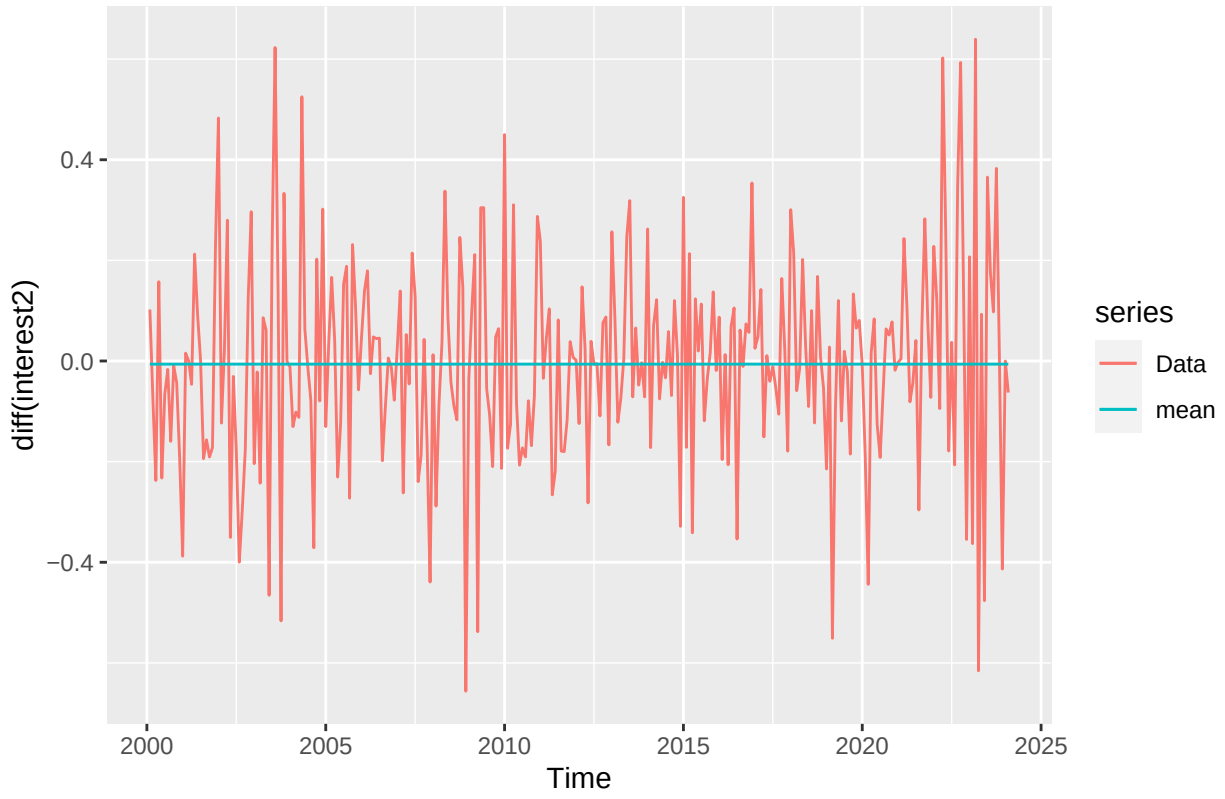
an alternitave way to find the necessary differentiation order to produce stationarity is using (ndiffs) function

```
ndiffs(interest2)
```

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

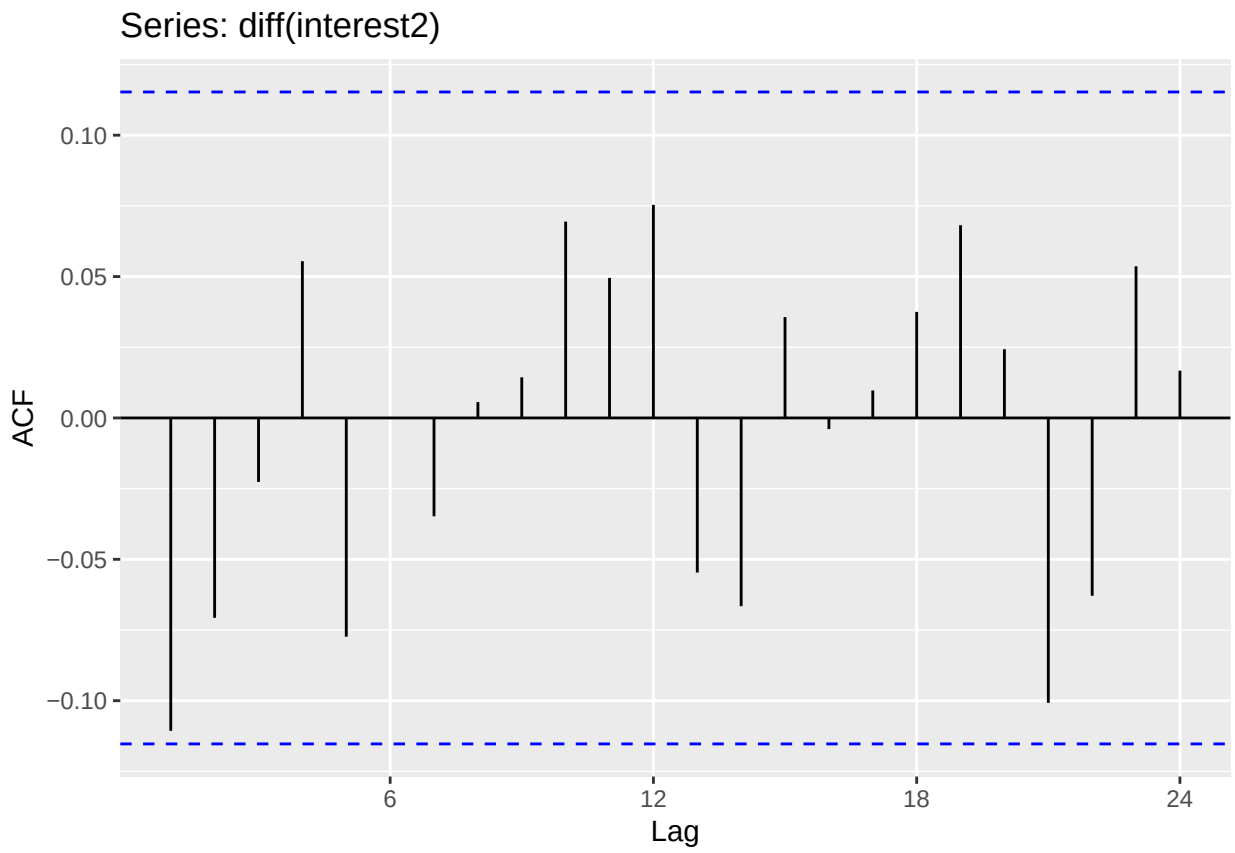
Analyzing the resulting differentiated time-series:

```
autoplot(diff(interest2), series = "Data")+
  autolayer(fitted(meanf(diff(interest2))), series = "mean")
```



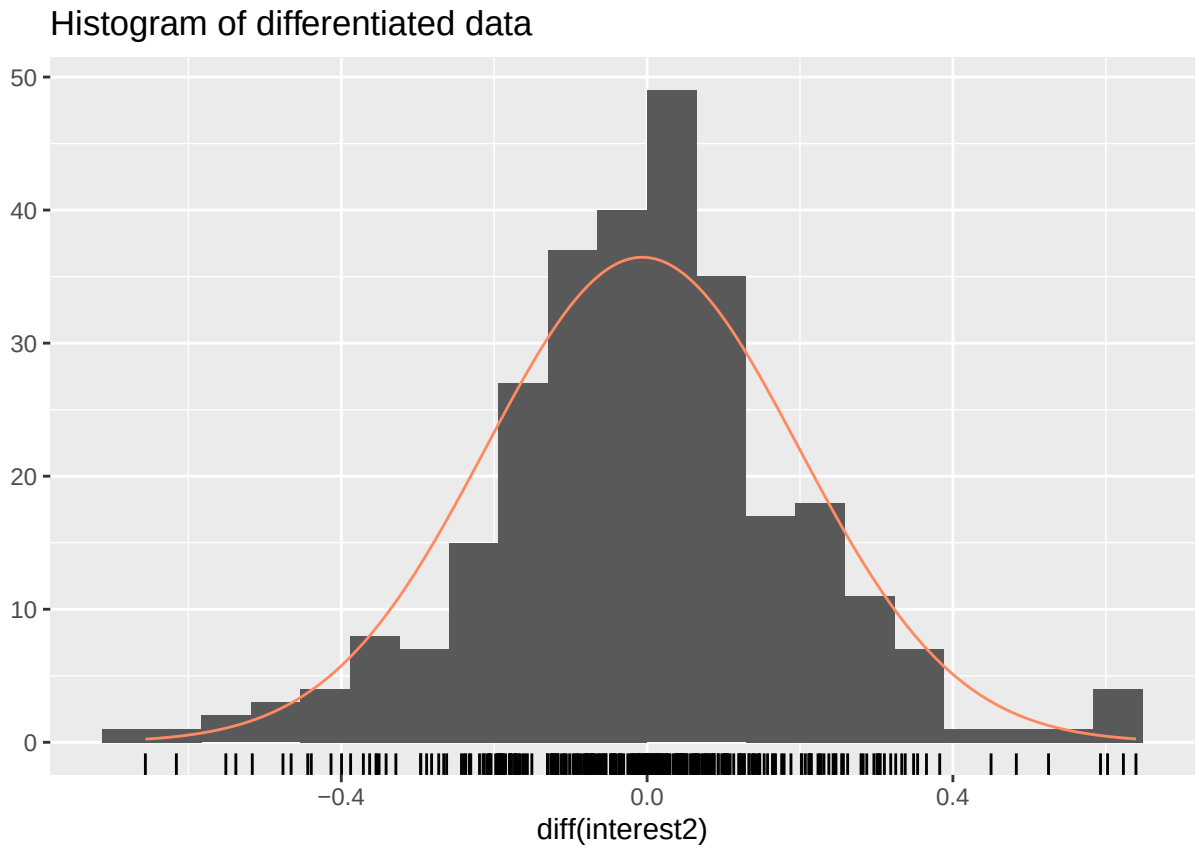
The mean is roughly constant, and close to 0.

```
ggAcf(diff(interest2))
```



The differentiated time-series has insignificant autocorrelation.

```
gghistogram(diff(interest2), add.normal=TRUE) +
ggtitle("Histogram of differentiated data") + ylab("")
```



the differentiated data has a normal distribution, which suggests that it is a white noise. To test this claim, we can use Ljung-Box test (H_0 assumes that the data is a white noise)

```
Box.test(diff(interest2), lag = 10, type = c("Box-Pierce", "Ljung-Box"))
```

```
##
##  Box-Pierce test
##
## data:  diff(interest2)
## X-squared = 9.5604, df = 10, p-value = 0.4799
```

p is relatively large, thus H_0 is not rejected, we can assume that the differentiated data is not correlated.

Arima forecast

I'll first use auto.arima which will automatically determine the order of the ARIMA forecast:

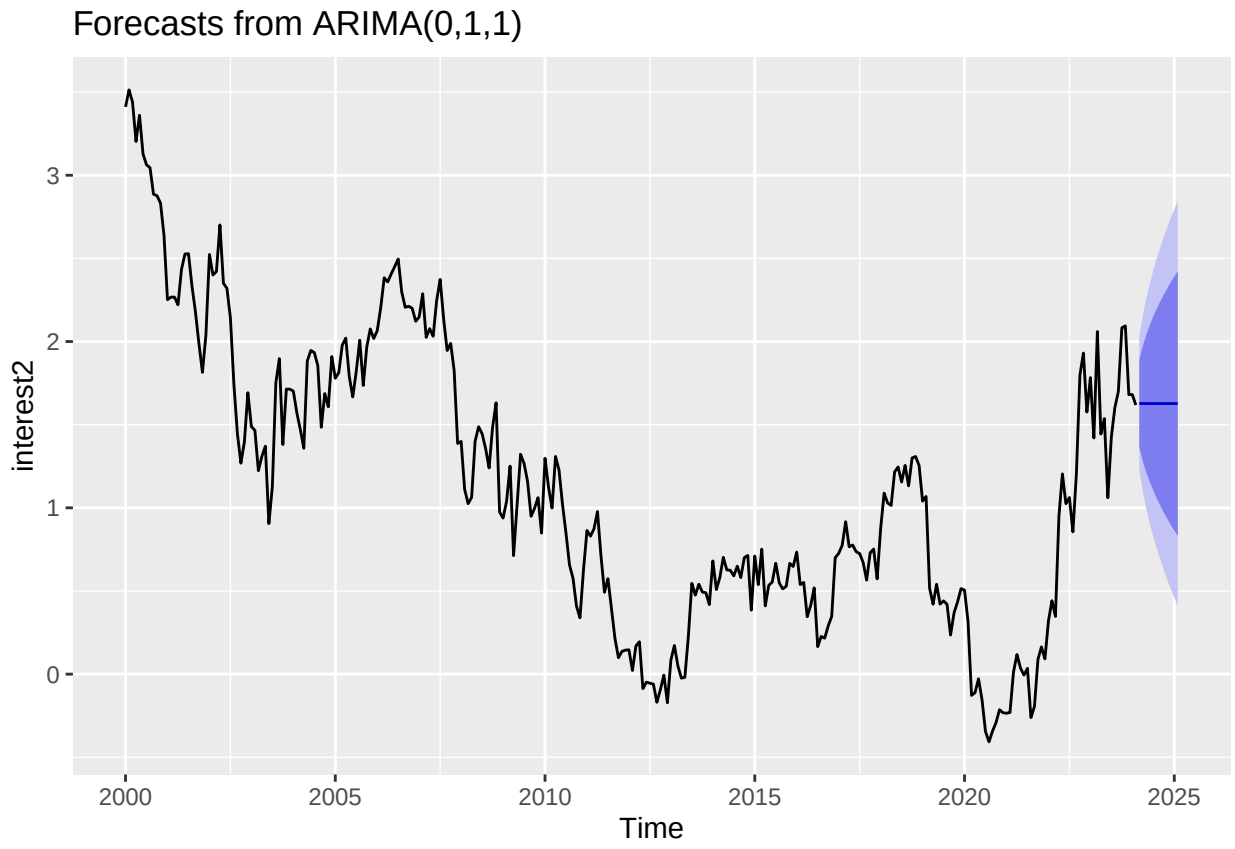
```
auto.arima(interest2, seasonal = FALSE)
```

```
## Series: interest2
## ARIMA(0,1,1)
##
## Coefficients:
```

```
##           ma1
##       -0.1300
## s.e.   0.0632
##
## sigma^2 = 0.04142:  log likelihood = 50.49
## AIC=-96.99   AICc=-96.94   BIC=-89.65
```

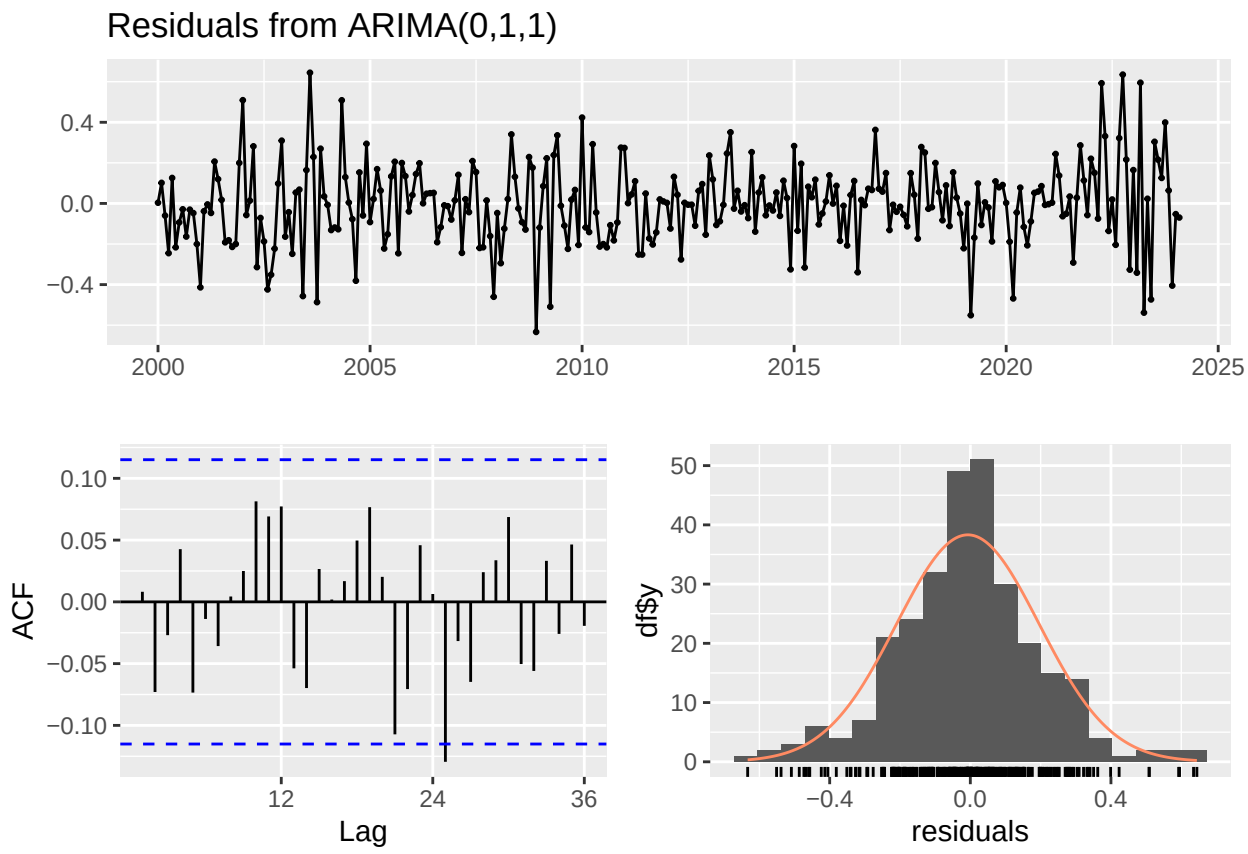
Plotting the forecast:

```
auto.arima(interest2, seasonal = FALSE) %>% forecast::forecast(h=12) %>% autoplot()
```



Check residuals

```
checkresiduals(auto.arima(interest2, seasonal = FALSE) %>% forecast::forecast(h=12))
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)
## Q* = 21.145, df = 23, p-value = 0.5722
##
## Model df: 1.   Total lags used: 24
```

the sum of the residuals squared:

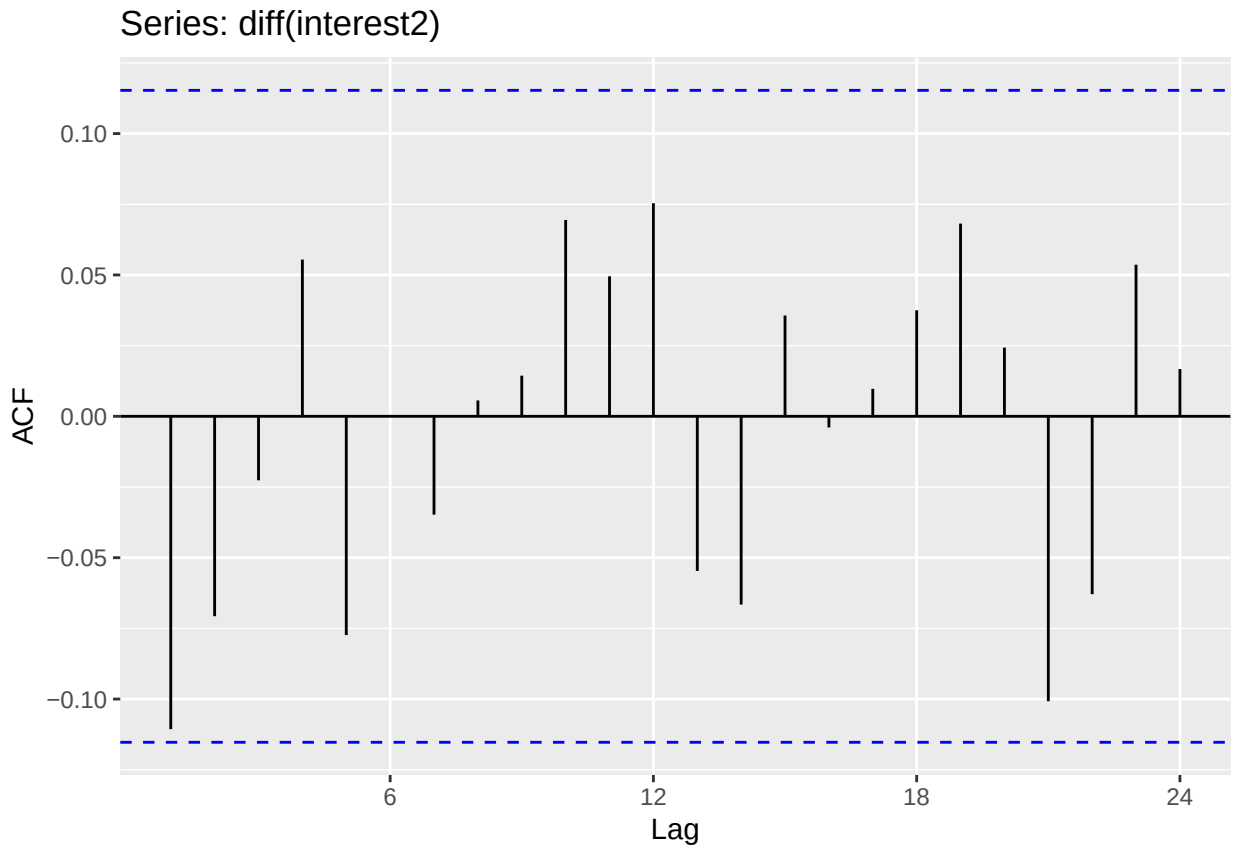
```
print(sum(residuals(auto.arima(interest2, seasonal = FALSE))^2))
```

```
## [1] 11.93
```

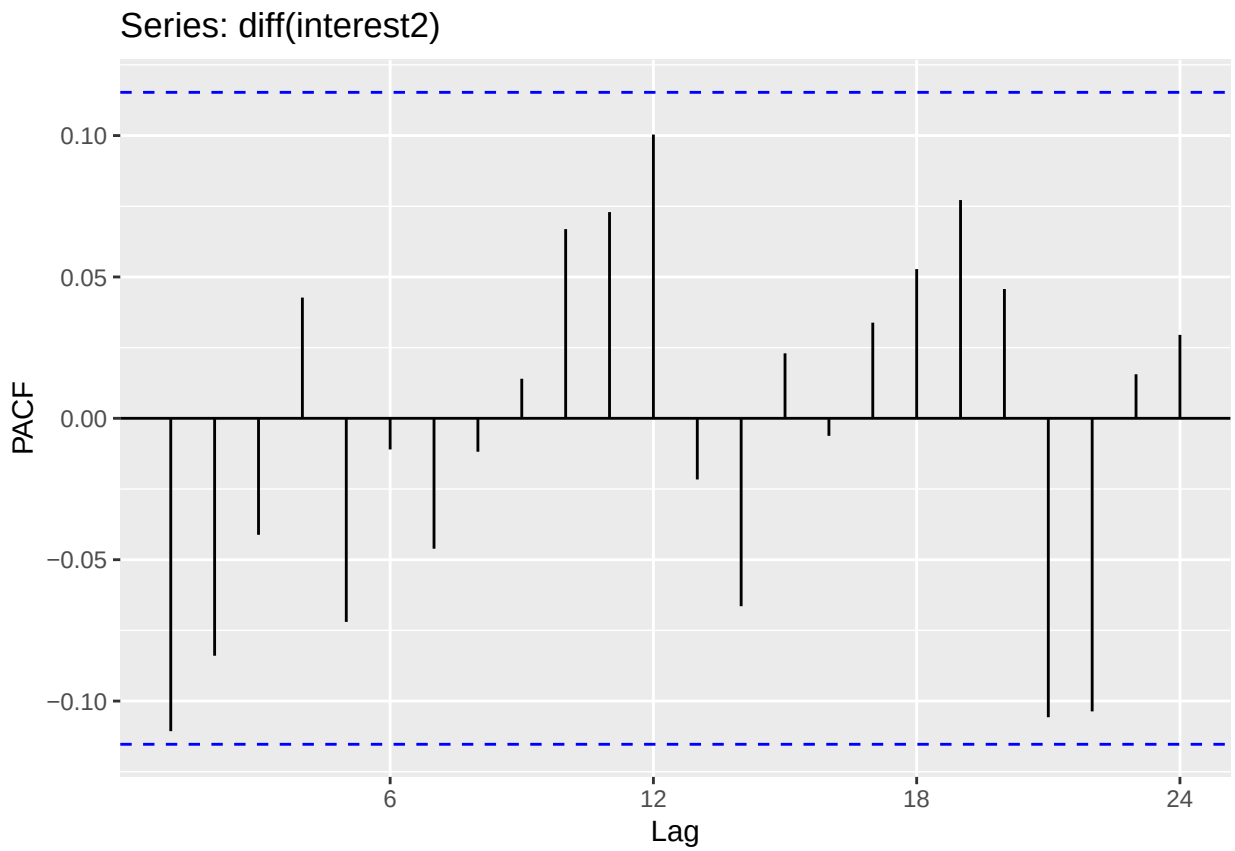
the p value in the Ljung-Box test is relatively big, therefore the ARIMA$(0,0,1)$ residuals do correspond to white noise.

suggested method from fpp2 section 8.5

```
ggAcf(diff(interest2))
```




```
ggPacf(diff(interest2))
```



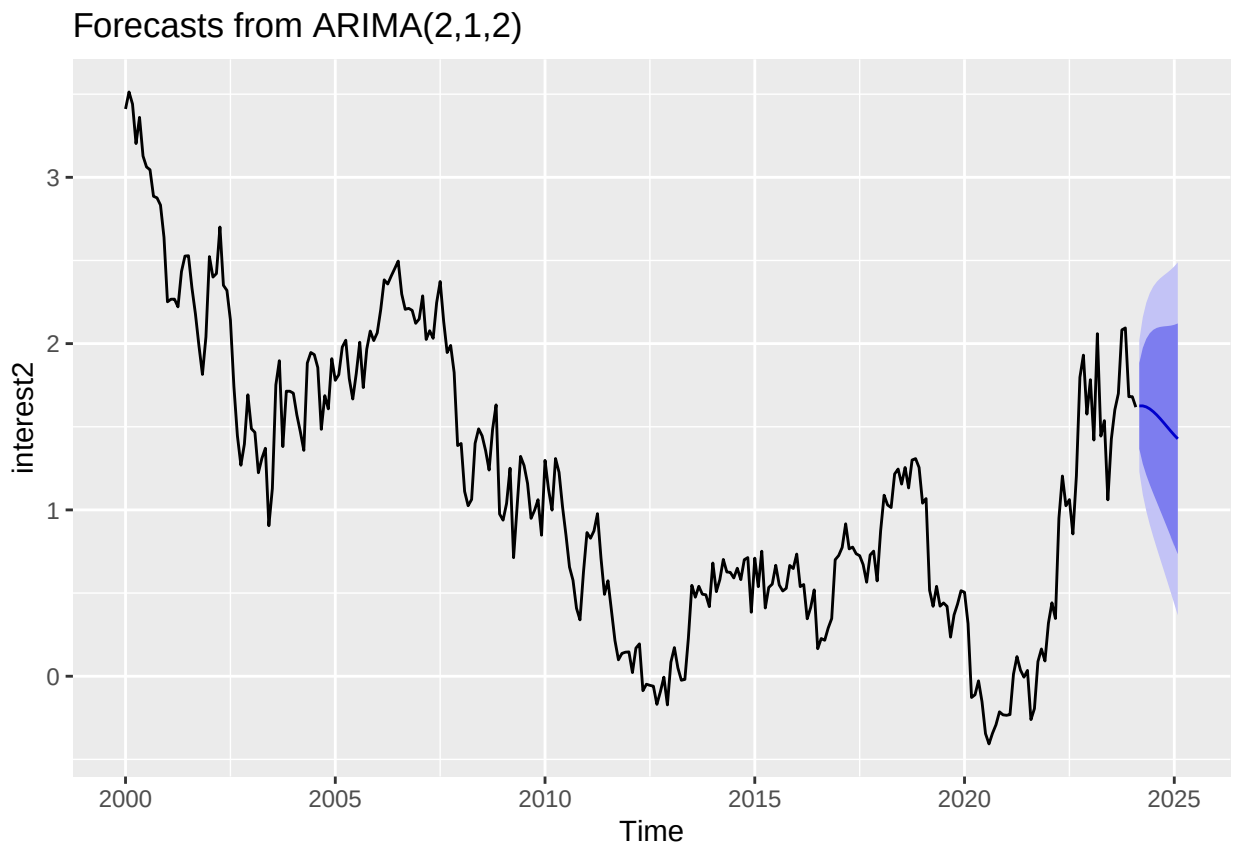
Neither of the two figures above is informative, None of them has a clear sinusoidal or decaying pattern, and there are no significant spikes.

Minimize AIC instead of AICc

```
minimum = -96.99
for(i in 0:6){for(j in 0:6){if(AIC(Arima(interest2, order = c(j, 1, i))) < minimum)
  {
  minimum = AIC(Arima(interest2, order = c(j, 1, i)))
  m = c(j, i)
  }}}
print(m)
```

```
## [1] 2 2
```

```
Arima(interest2, c(2, ndiffs(interest2), 2)) %>% forecast::forecast(h=12) %>% autoplot()
```

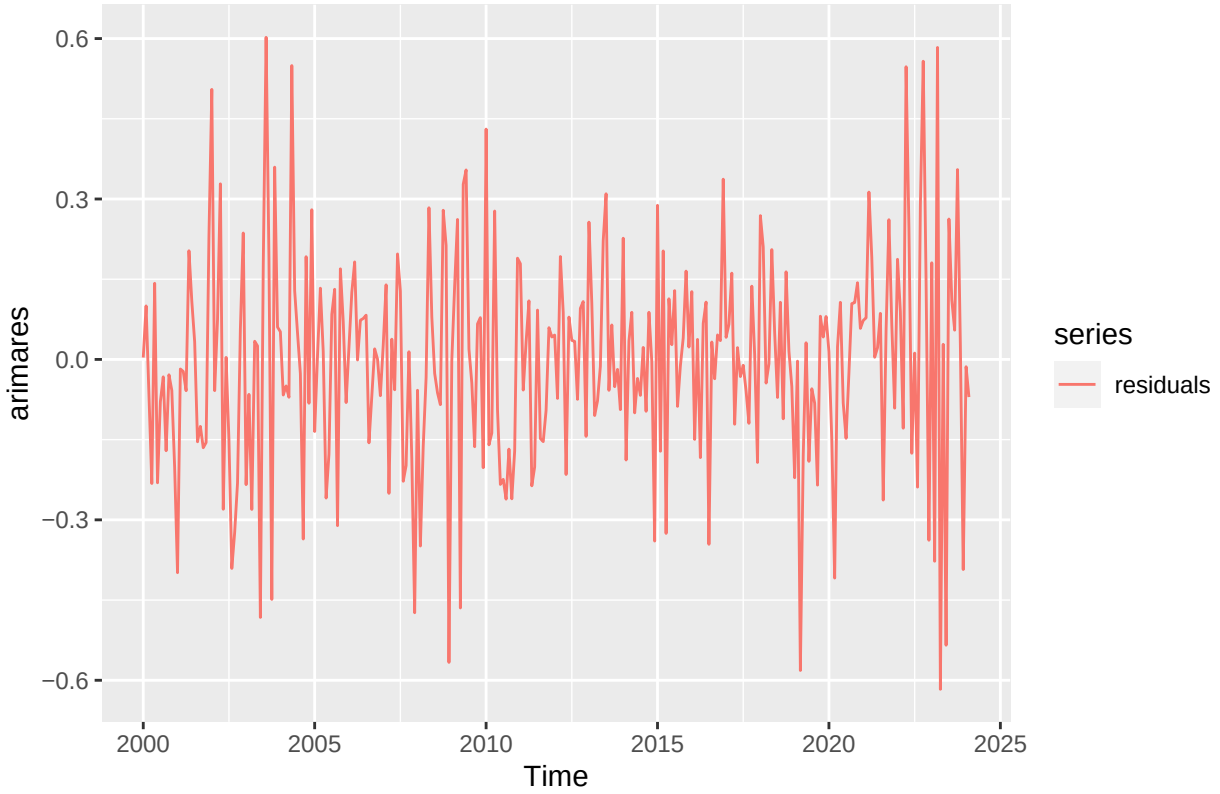


checking the fitting of this arima (ARIMA(2, 1, 2)) forecast

```
arimafit <- fitted(Arima(interest2, c(2, 1, 2)))
autoplot(arimafit, series = "Fitted") + autolayer(interest2, series = "Data")
```

Now checking the residuals:

```
arimares <- residuals(Arima(interest2, c(2, 1, 2)))
autoplot(arimares, series = "residuals")
```



Check autocorrelation:

which implies that the residuals of the arima fitting does correspond to white noise

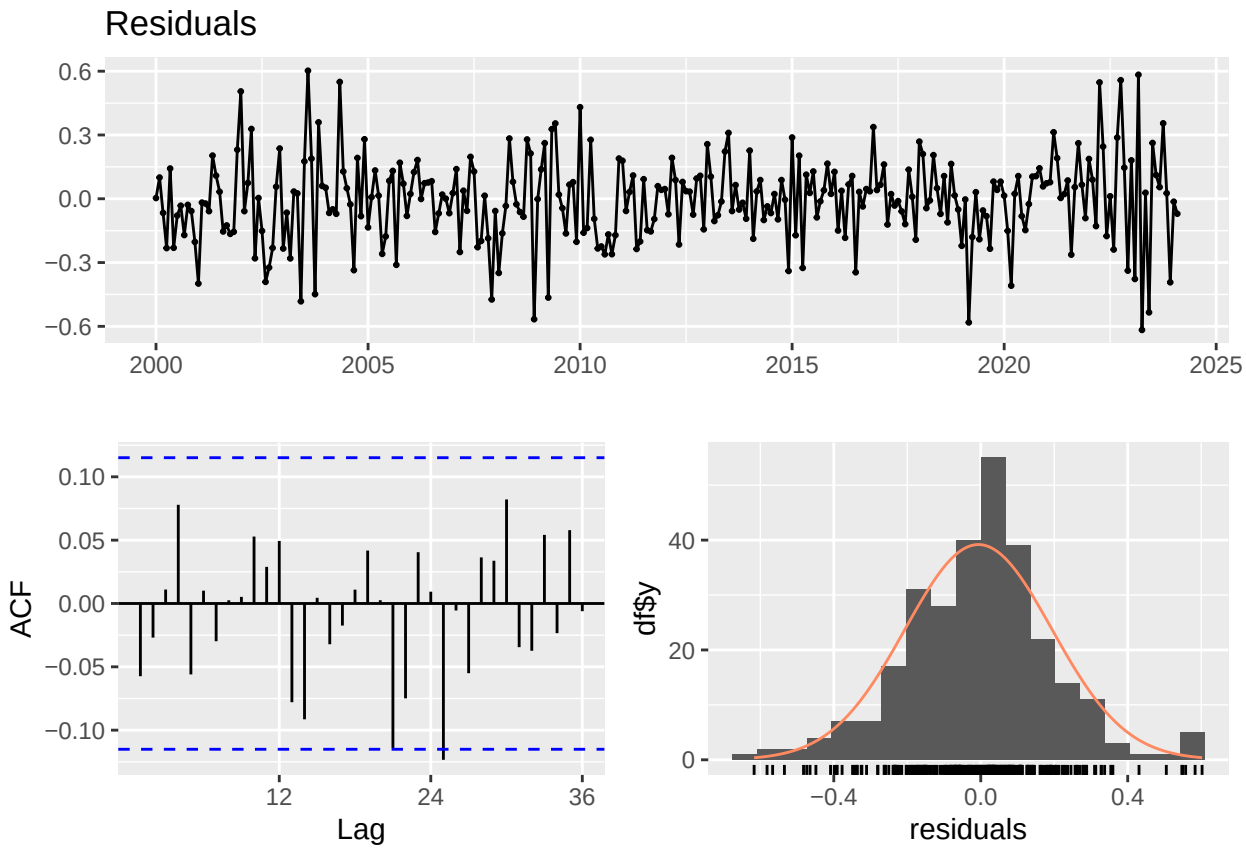
Testing residuals by Ljung-box test: H_0: the residuals are not distinguishable from white noise.

```
Box.test(arimares, lag=10, type = c("Box-Pierce", "Ljung-Box"))
```

```
##
##  Box-Pierce test
##
## data:  arimares
## X-squared = 4.9775, df = 10, p-value = 0.8927
```

p is large, H_0 is not rejected and the residuals are not correlated

```
checkresiduals(arimares, lag=10, test="LB")
```




```
##
##  Ljung-Box test
##
## data:  Residuals
## Q* = 5.0944, df = 10, p-value = 0.8848
##
## Model df: 0.   Total lags used: 10
```

```
sum(arimares^2)
```

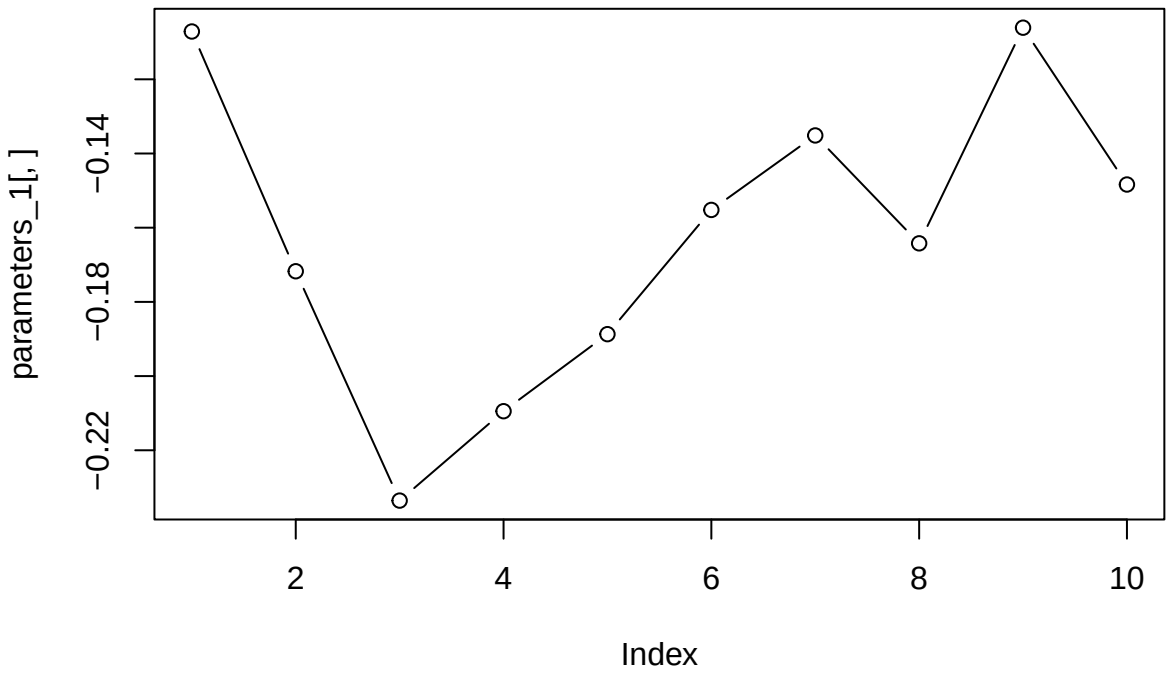
```
## [1] 11.56855
```

Test for model's parameter's stability:

I will use Arima(0, 1, 1) model to solve for different time windows, and see what coefficients we get with each time

```
parameters_1 =c()

s = 2000
if(s == 1982){
  for (i in 1:30){
    parameters_1 <- cbind(parameters_1, coefficients(Arima(window(interest, frequency = 12, start=c(198
```

```
}else{
  for (i in 1:10){
    parameters_1 <- cbind(parameters_1, coefficients(Arima(window(interest, frequency = 12, start=c(200
}

plot(parameters_1[, ], type='b')
```




```
ndiffs(parameters_1)
```

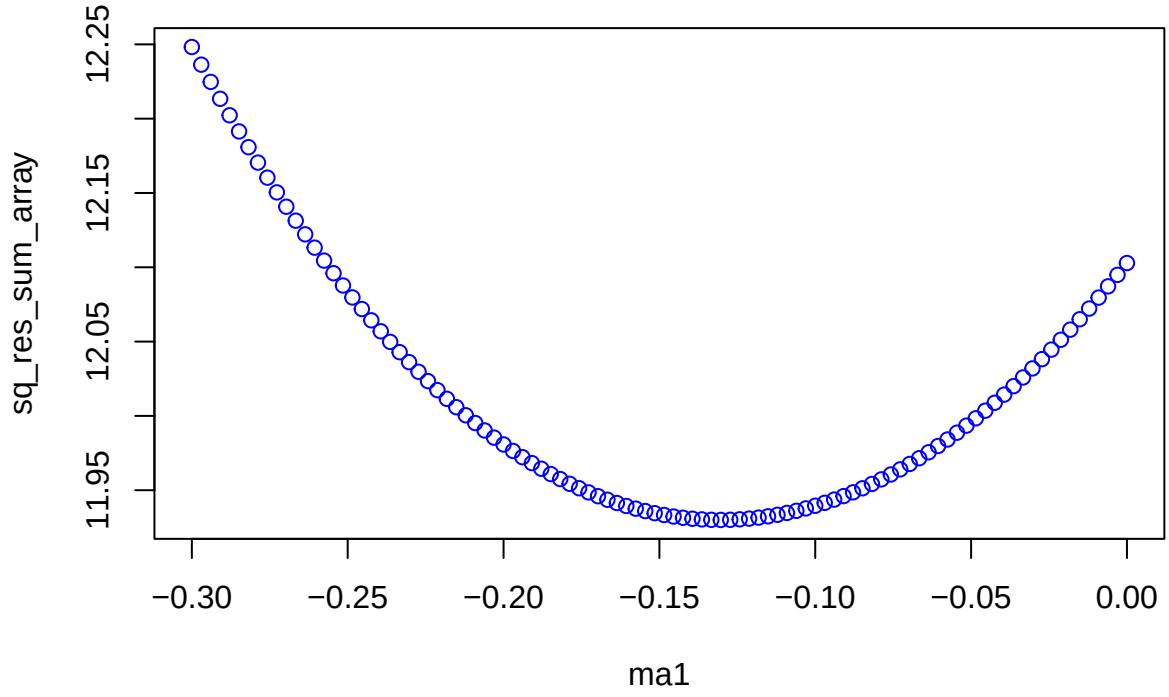
```
## [1] 0
```

which implies that the list parameters_1 is stationary

Since the model ARIMA(0, 1, 1) specified by auto.arima has only 1 parameter (ma1), I will plot the p

```
# define linspace equivalent function:
linspace <- function(start, end, n) {
  return(seq(from = start, to = end, length.out = n))
}

ma1 = linspace(-0.3, 0, 100)

# define a function that sums the square of the residuals of the fixed Arima fit:
sumressq <- function(timeseries, MA1){
  return(sum(residuals(arima(timeseries, c(0, 1, 1), fixed = c(MA1)))^2))
```

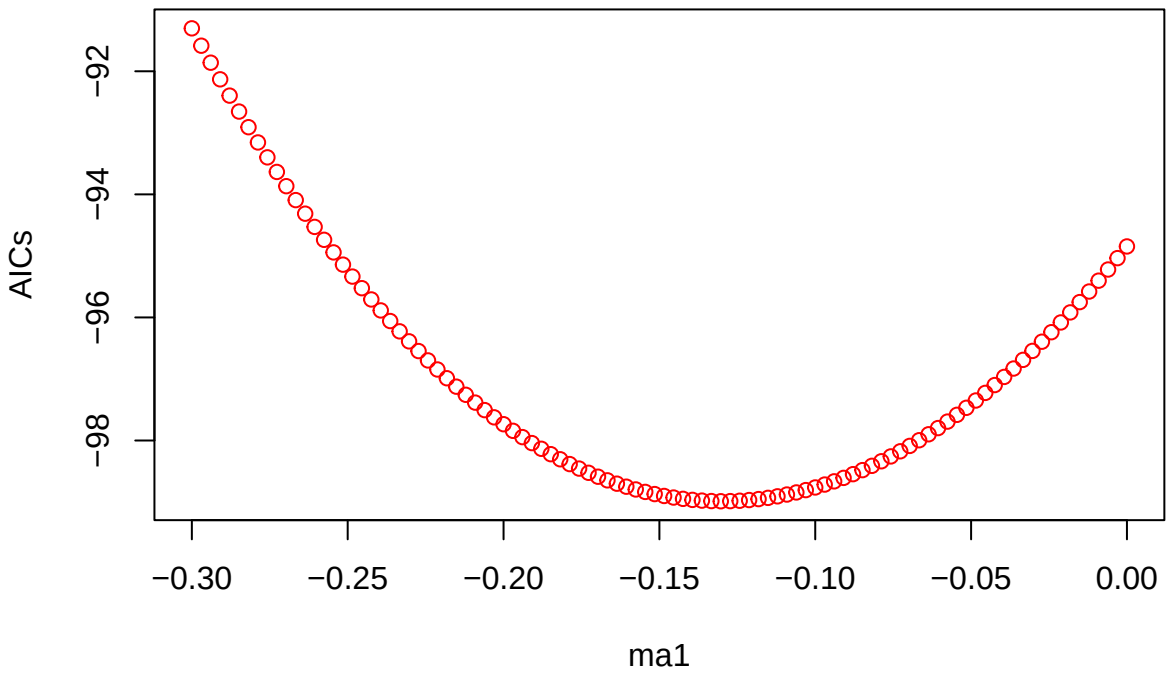
```
}

sq_res_sum_array = numeric(100)

# find the error in each point
for(i in 1:100){
  sq_res_sum_array[i] = sumressq(interest2, ma1[i])
}

plot(ma1, sq_res_sum_array, col="blue")
```



Judging by the shape of the figure, it suggests that the model parameters are stable.

using AIC instead of residuals square sum:

```
AICs = numeric(100)

# find the error in each point
for(i in 1:100){
  AICs[i] = AIC(arima(interest2, c(0, 1, 1), fixed = c(ma1[i])))
}

plot(ma1, AICs, col="red")
```


AICs
−92
−94
−96
−98
−0.30
−0.25
−0.20
−0.15
−0.10
−0.05
0.00
ma1