Report 5

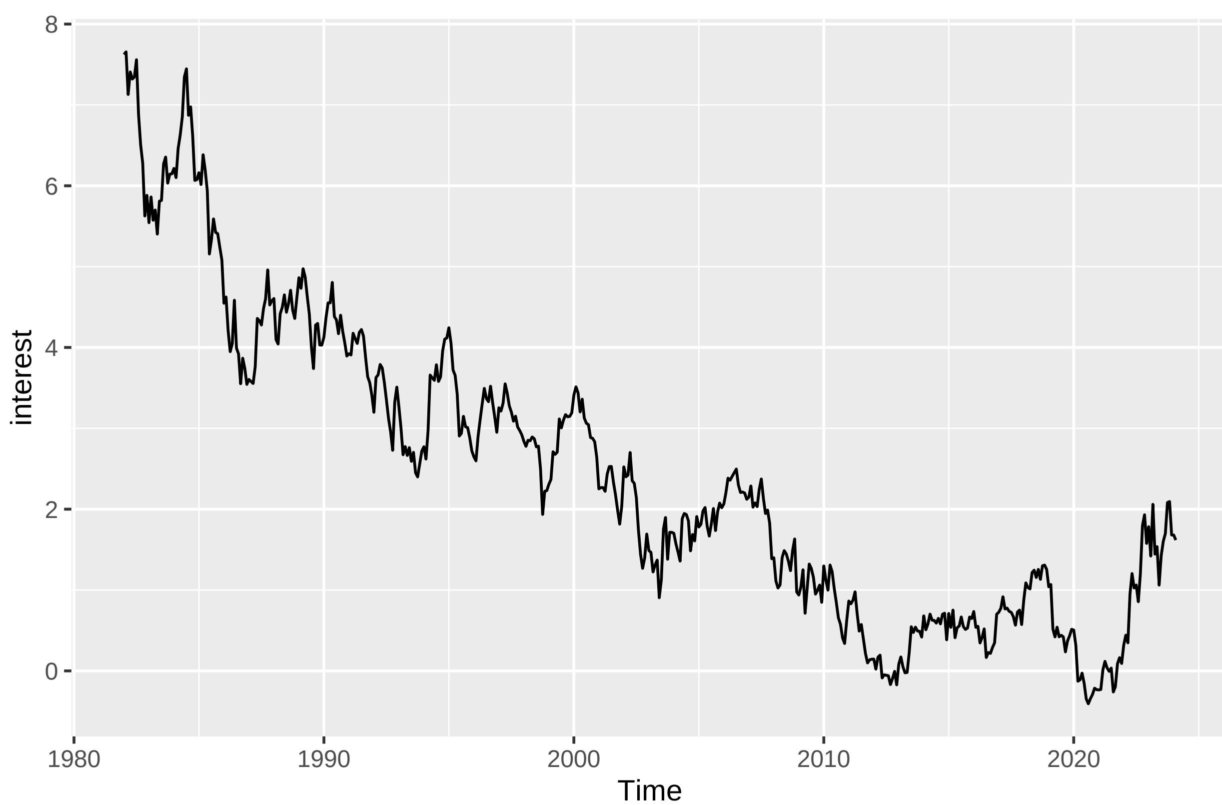
Khaled Hasan

2024-03-28

Import the data:

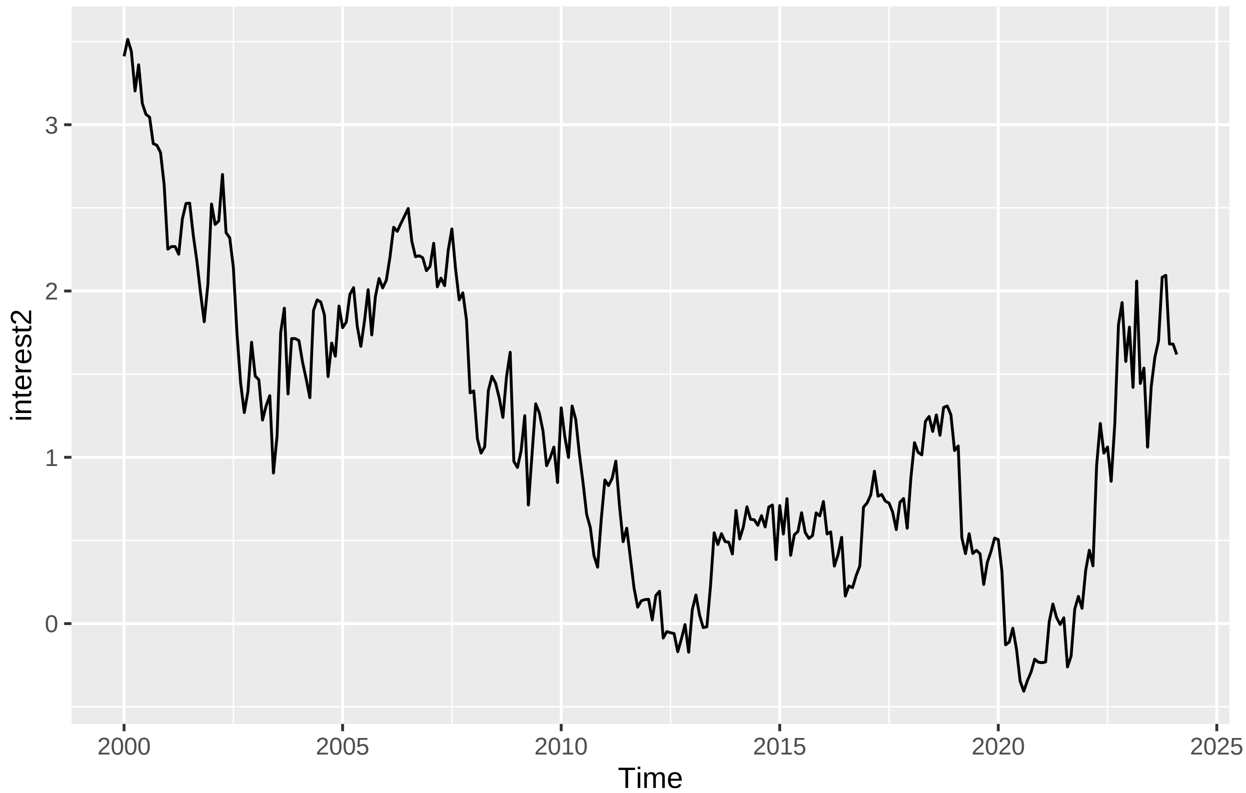
import 10 years real interest rate time series from csv (source:https://fred.stlouisfed.org/graph/?g=1hoLl):

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



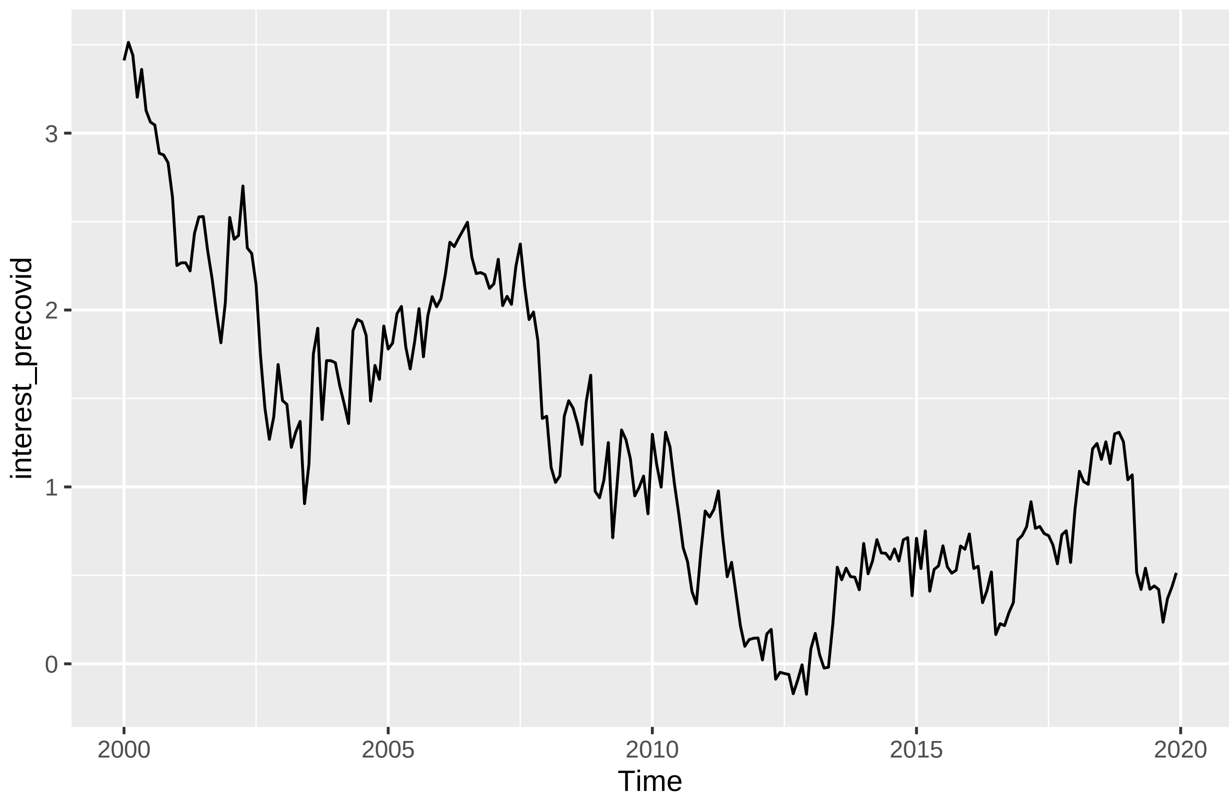
cut a window starting from year 2000

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



Excluding the post pandamic era (2020-):

```
interest_precovid = window(interest2, frequency = 12, end=c(2019, 12))

autoplot(interest_precovid)
```



GARCH

Arch test:

First I will use Arch LM-test to determine whether Arch effects are present

```
ArchTest(interest2)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  interest2
## Chi-squared = 252.34, df = 12, p-value < 2.2e-16
```

Since $p \ll 0.01$ H_0 is rejected, and we take H_a instead, which is that there are arcg effects and therefore using garch model is reasonable.

GARCH order

```
garch(interest2, control = garch.control(grad="numeric", trace = FALSE))
```

```
##
## Call:
```

```
## garch(x = interest2, control = garch.control(grad = "numeric",     trace = FALSE))
##
## Coefficient(s):
##        a0         a1         b1
## 6.745e-01  9.576e-01  1.332e-12
```

I will therefore assume the model to be a garch(1, 1) (the default)

GARCH fit

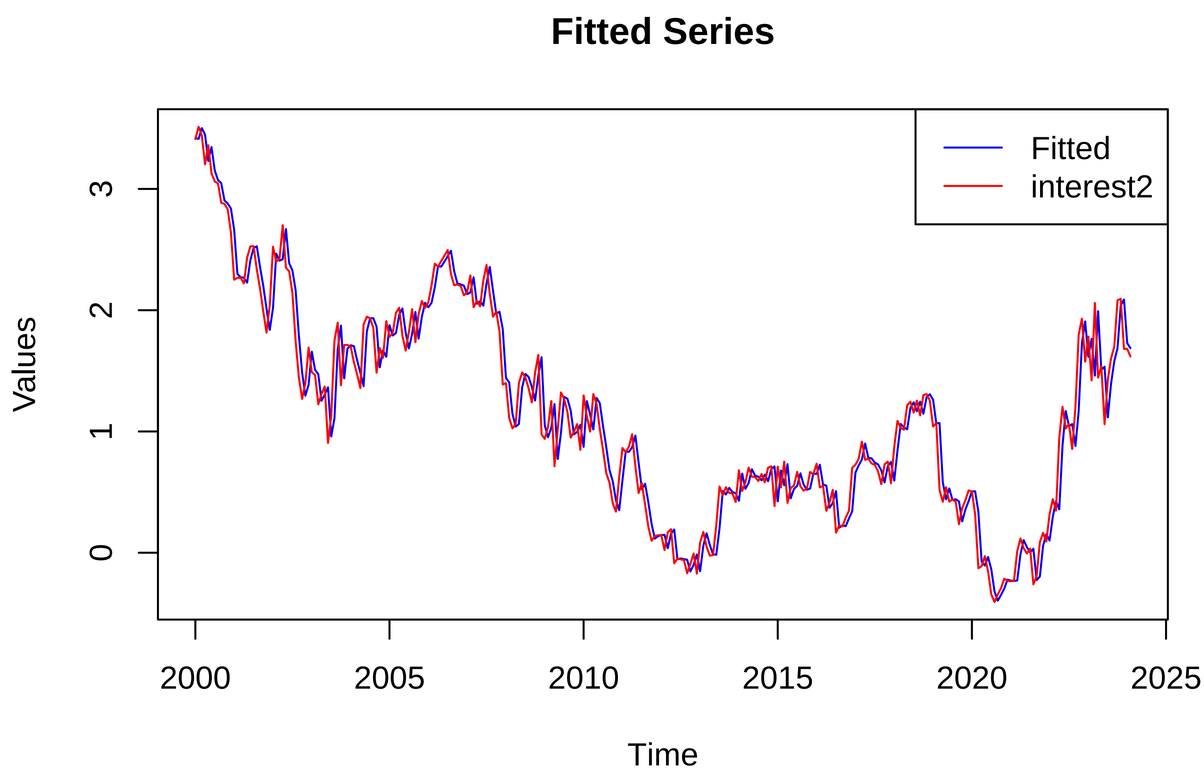
```r
garch_fit = rugarch::ugarchfit(rugarch::ugarchspec(), interest2)
garch_fit
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      3.420184    0.187901  18.2021 0.000000
## ar1     0.999523    0.003922 254.8694 0.000000
## ma1    -0.114114    0.068327  -1.6701 0.094895
## omega   0.002715    0.001572   1.7276 0.084067
## alpha1  0.078512    0.032561   2.4112 0.015899
## beta1   0.858108    0.056513  15.1843 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      3.420184    0.014075 243.0028 0.000000
## ar1     0.999523    0.003691 270.7913 0.000000
## ma1    -0.114114    0.079177  -1.4413 0.149511
## omega   0.002715    0.000974   2.7889 0.005288
## alpha1  0.078512    0.027645   2.8400 0.004512
## beta1   0.858108    0.028623  29.9795 0.000000
##
## LogLikelihood : 59.86668
##
## Information Criteria
## ------------------------------------
##
## Akaike       -0.37149
## Bayes        -0.29557
## Shibata      -0.37233
```

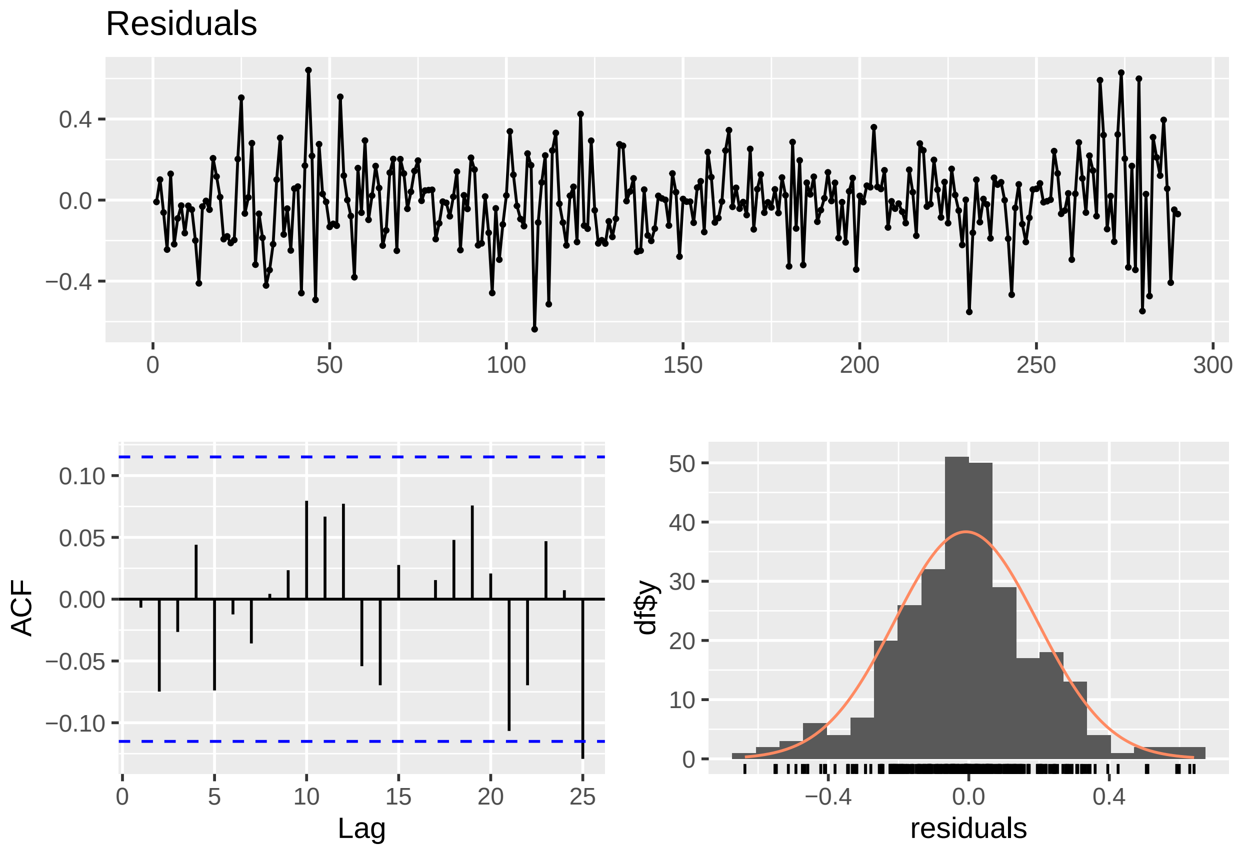
```
## Hannan-Quinn -0.34107
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.1214  0.7276
## Lag[2*(p+q)+(p+q)-1][5]    2.3742  0.8405
## Lag[4*(p+q)+(p+q)-1][9]    3.4199  0.8173
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.2593  0.6106
## Lag[2*(p+q)+(p+q)-1][5]    0.6883  0.9251
## Lag[4*(p+q)+(p+q)-1][9]    1.9014  0.9164
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.4429 0.500 2.000  0.5057
## ARCH Lag[5]    0.5821 1.440 1.667  0.8591
## ARCH Lag[7]    1.9056 2.315 1.543  0.7373
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  0.4508
## Individual Statistics:
## mu     0.003226
## ar1    0.068274
## ma1    0.137340
## omega  0.106164
## alpha1 0.137645
## beta1  0.119309
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.49 1.68 2.12
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias          1.14651 0.2525
## Negative Sign Bias 0.24784 0.8044
## Positive Sign Bias 0.01063 0.9915
## Joint Effect       2.37151 0.4990
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     19.38       0.4328
```

```
## 2    30      34.83       0.2103
## 3    40      44.07       0.2658
## 4    50      48.28       0.5024
##
##
## Elapsed time : 0.1963761
```

```
{
garch_fit_results = ts(fitted(garch_fit@fit), start = c(2000, 1), frequency = 12)
plot(garch_fit_results, type = "l", col = "blue", xlab = "Time", ylab = "Values", main = "Fitted Series
lines(interest2, col = "red")
legend("topright", legend = c("Fitted", "interest2"), col = c("blue", "red"), lty = 1)
}
```




```
checkresiduals(garch_fit@fit)
```




```
##
##  Ljung-Box test
##
## data:  Residuals
## Q* = 6.5776, df = 10, p-value = 0.7646
##
## Model df: 0.   Total lags used: 10
```

arima fit for comparison sake:

```
arimafit = auto.arima(interest2, stepwise = FALSE, approximation = FALSE)
residuals_of_arima_fit = residuals(arimafit)
```

Now I will compare the aic of the 2 fits:

```
{
print(AIC(arimafit))
print(-2*garch_fit@fit$LLH + 2*length(garch_fit@fit$coef))
}
```

```
## [1] -96.98683
## [1] -107.7334
```

The AIC values suggests that the garch model is performing better than the Arima fit.

Forcasting

I used "ugarchforecast" function, unfortunately this was not compatible with the forecast function from forecast library. Therefore I had to do the plotting manually

```
garchForecast = rugarch::ugarchforecast(garch_fit, n.ahead = 12)

garchForecastVals = ts(garchForecast@forecast$seriesFor, start = c(2024, 2), frequency = 12)

Low1 = ts(-garchForecast@forecast$sigmaFor + garchForecast@forecast$seriesFor, start = c(2024, 2), frequ

High1 = ts(garchForecast@forecast$sigmaFor + garchForecast@forecast$seriesFor, start = c(2024, 2), frequ

autoplot(interest2)+autolayer(garchForecastVals, series = "garch forecast")+autolayer(Low1, series = "-
```

Testing other models

Below is an AIC table for different orders of garch models, the table suggests that the best model for the data at hand is the garch (1, 1) model.

```
AICgarch <- function(ord){
    garFit = rugarch::ugarchfit(rugarch::ugarchspec(variance.model = list(garchOrder=ord)), interest2)
    return(-2*garFit@fit$LLH + 2*length(garFit@fit$coef))
}
```

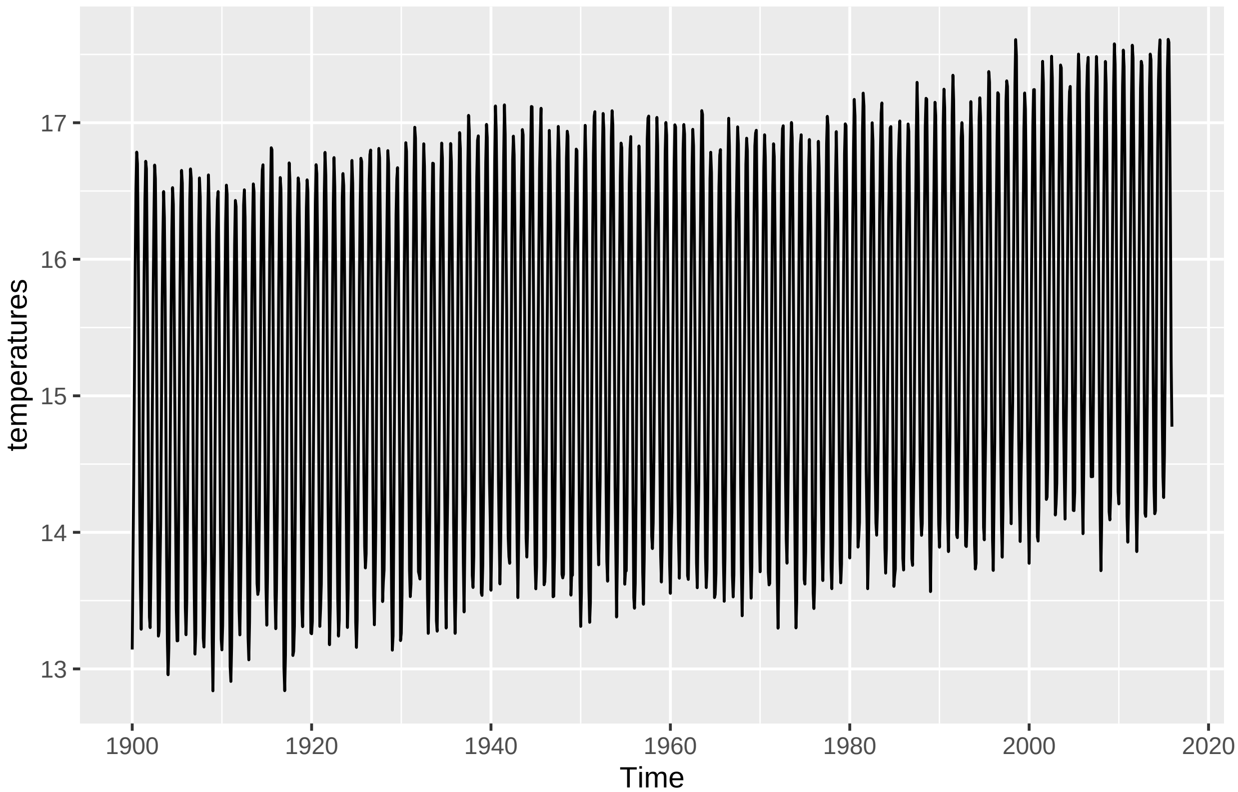
```r
results <- matrix(NA, nrow = 10, ncol = 10)

for (i in 0:9) {
    for (j in 0:9) {
      if(j == 0 && i == 0){
        results[1, 1] <- -1
      }else {
        results[i+1, j+1] <- AICgarch(c(i, j))
      }
    }
}

print(results)
```

```
##              [,1]       [,2]       [,3]       [,4]       [,5]      [,6]
##  [1,]   -1.00000  -92.53576  -90.53667  -88.53660  -86.53933 -84.54034
##  [2,]  -96.06844 -107.73336 -105.58795 -103.50489 -101.44704 -99.50784
##  [3,]  -97.58454 -106.01779 -104.01779 -101.91409 -100.10362 -98.08588
##  [4,]  -96.30808 -104.34407 -102.34407 -100.34407  -98.28506 -96.22159
##  [5,]  -97.38466 -103.74066 -101.74066  -99.74066  -97.74066 -95.59500
##  [6,]  -96.76451 -102.11581 -100.11581  -98.12482  -96.17075 -94.39831
##  [7,] -100.28167 -100.07000  -99.65629  -97.65629  -95.71964 -93.94961
##  [8,]  -99.43815  -97.89106  -97.64979  -96.55035  -93.69409 -91.92266
##  [9,]  -97.20944  -95.85377  -95.63837  -94.35247  -93.07913 -91.07913
## [10,]  -95.20570  -93.85659  -93.74180  -92.60987  -91.57204 -89.57204
##             [,7]      [,8]      [,9]     [,10]
##  [1,] -82.54097 -80.54299 -78.54647 -76.55330
##  [2,] -97.45555 -95.47939 -93.41980 -91.40524
##  [3,] -96.11324 -94.13133 -92.09420 -90.26184
##  [4,] -95.11095 -93.06262 -90.88415 -88.84008
##  [5,] -93.77584 -91.77815 -89.85817 -87.93984
##  [6,] -93.37047 -93.50156 -91.26212 -89.30322
##  [7,] -96.07656 -94.65627 -92.36764 -90.21716
##  [8,] -94.52002 -92.87224 -90.53084 -88.38692
##  [9,] -92.20445 -90.53084 -88.53084 -86.38692
## [10,] -90.03033 -88.38693 -86.38692 -84.38692
```

Temperature Data

```r
GLOBALTEMPERATURE = read.csv(file = "C:\\Users/ss/Desktop/Time_series_Analysis/GlobalTemperatures_1900.
temperatures <- ts(GLOBALTEMPERATURE[8], frequency = 12, start = c(1900, 1))
uncertainties <- ts(GLOBALTEMPERATURE[9], frequency = 12, start = c(1900, 1))
```

```r
autoplot(temperatures)
```

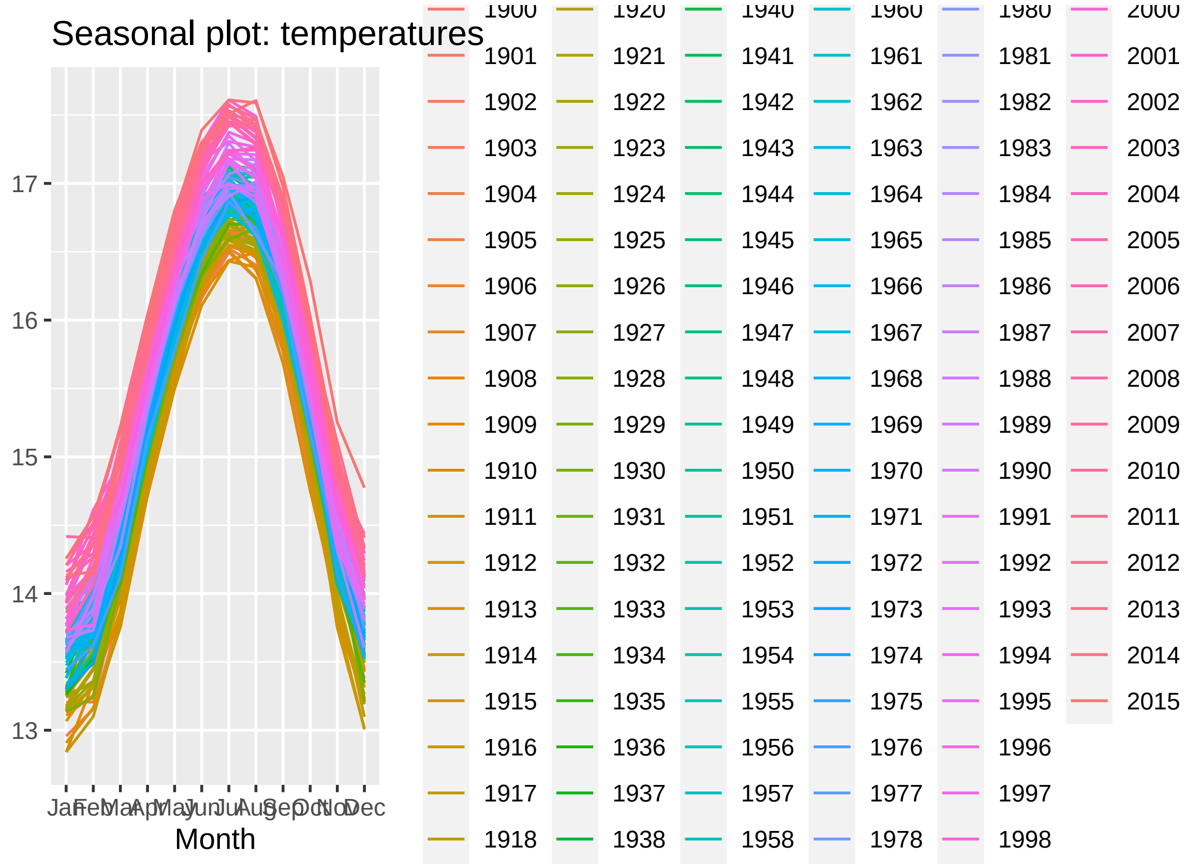


The data looks too noisy, and it seems impossible to dray any useful conclusions from it

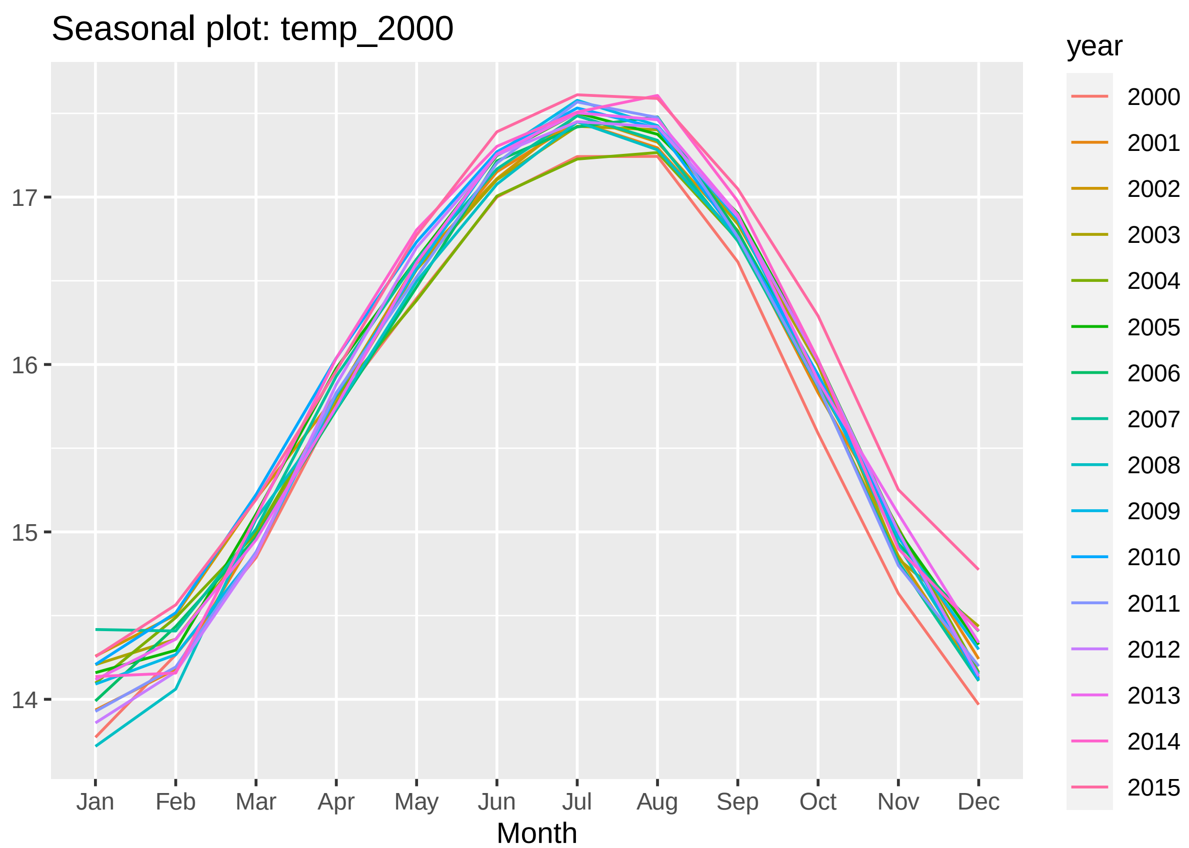
I will first cut the data from year 2000 onwards

```
temp_2000 <- window(temperatures, start = c(2000, 1))
```

```
ggseasonplot(temperatures)
```



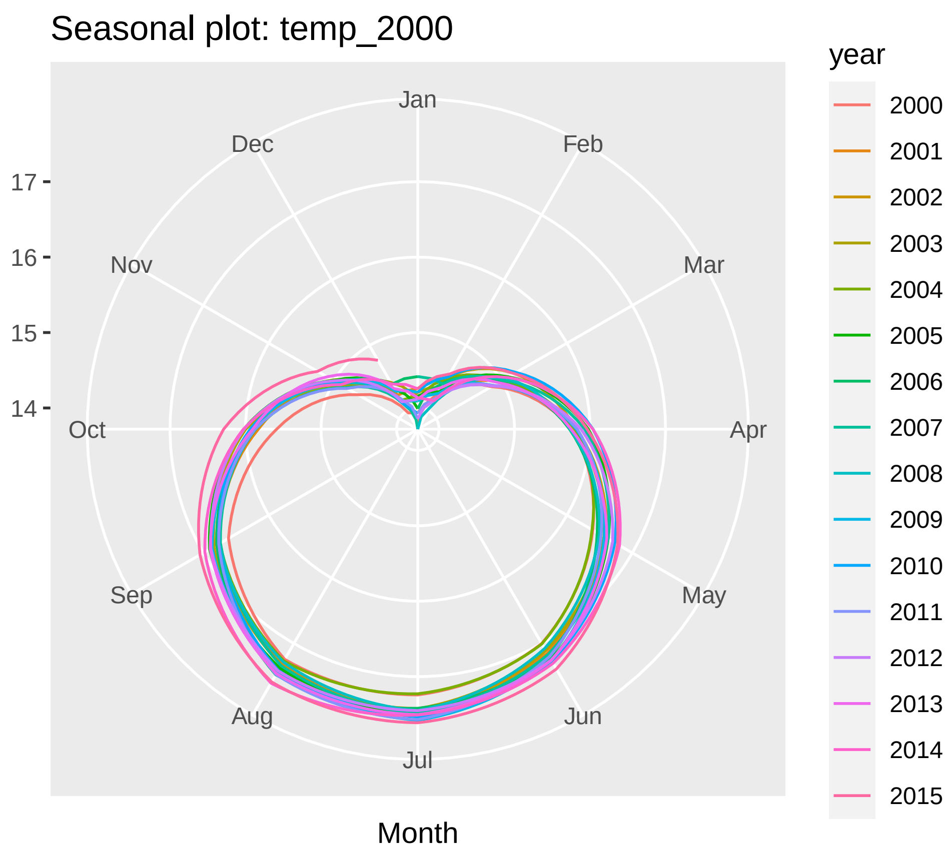

```
ggseasonplot(temp_2000)
```



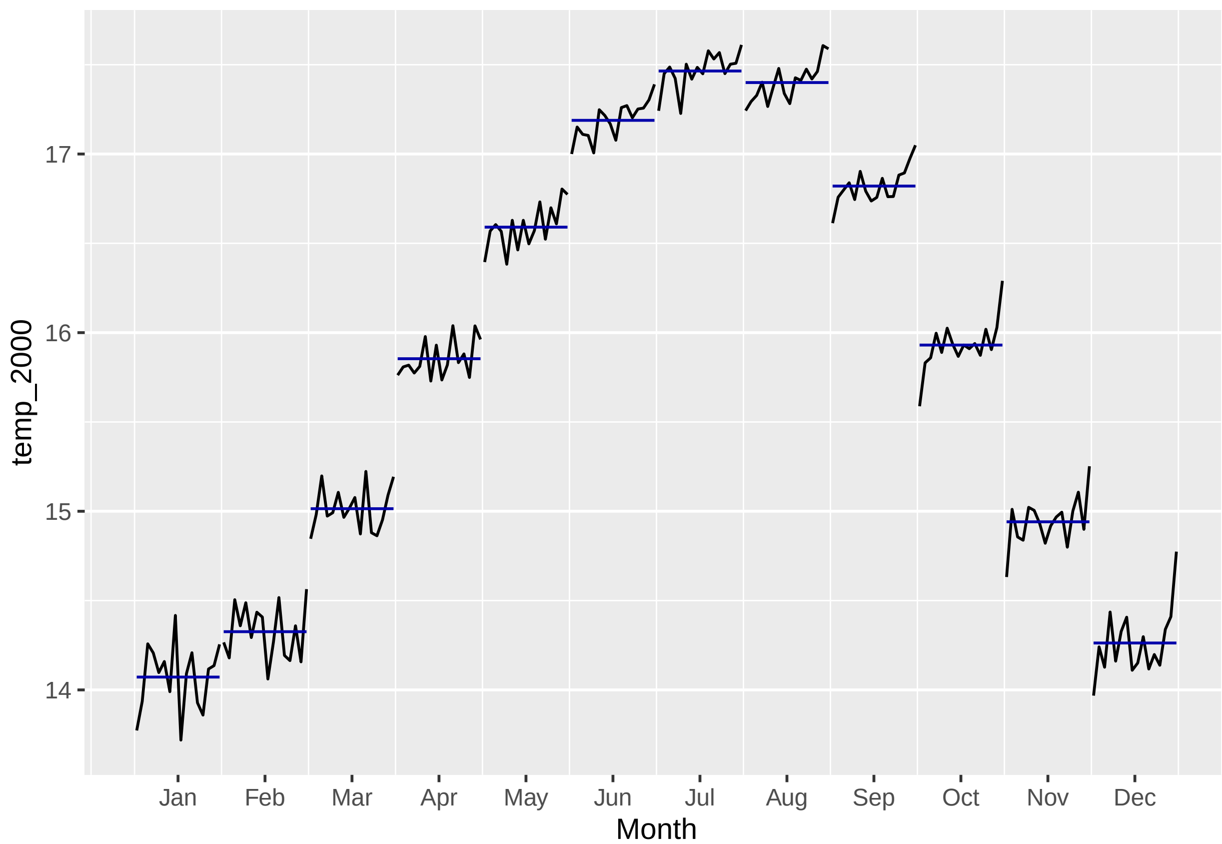
a clear seasonality is shown, where the temperature tend to be at a maximum between june and August (summer), there is also a general upward trend in the data.

if we looked at a polar version of the data

```
ggseasonplot(temp_2000, polar = TRUE)
```




```
ggsubseriesplot(temp_2000)
```



Stationarity

Dicky-Fuller test (H_0: data is not stationary (H_0: unit root exists):

```r
adf.test(temp_2000, k = 1)
```

```
## Warning in adf.test(temp_2000, k = 1): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  temp_2000
## Dickey-Fuller = -25.266, Lag order = 1, p-value = 0.01
## alternative hypothesis: stationary
```

The dicky Fuller test suggested that the data is stationary (H_a is accepted) which is weird considering the general upward trend one can notice in the data. Trying with KPASS test (H_0 : data is stationary)

```r
kpss.test(temp_2000)
```

```
## Warning in kpss.test(temp_2000): p-value greater than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  temp_2000
## KPSS Level = 0.030139, Truncation lag parameter = 4, p-value = 0.1
```

again, the hypothesis that the data is stationary was not rejected, which means that the data might be stationary, despite the general trend in the data.

by applying both tests on the original dataset:

```
{print(adf.test(temperatures))
print(kpss.test(temperatures))}
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  temperatures
## Dickey-Fuller = -3.6658, Lag order = 11, p-value = 0.02635
## alternative hypothesis: stationary

## Warning in kpss.test(temperatures): p-value smaller than printed p-value

##
##  KPSS Test for Level Stationarity
##
## data:  temperatures
## KPSS Level = 3.4221, Truncation lag parameter = 7, p-value = 0.01
```

since $p \ll 0.01$ in KPSS test, H_0 is rejected which implies that the data is not stationary. the p value for the Dicky-Fuller test is also small.

check the differentiated data:

```
{print(adf.test(diff(temperatures)))
print(kpss.test(diff(temperatures)))}
```

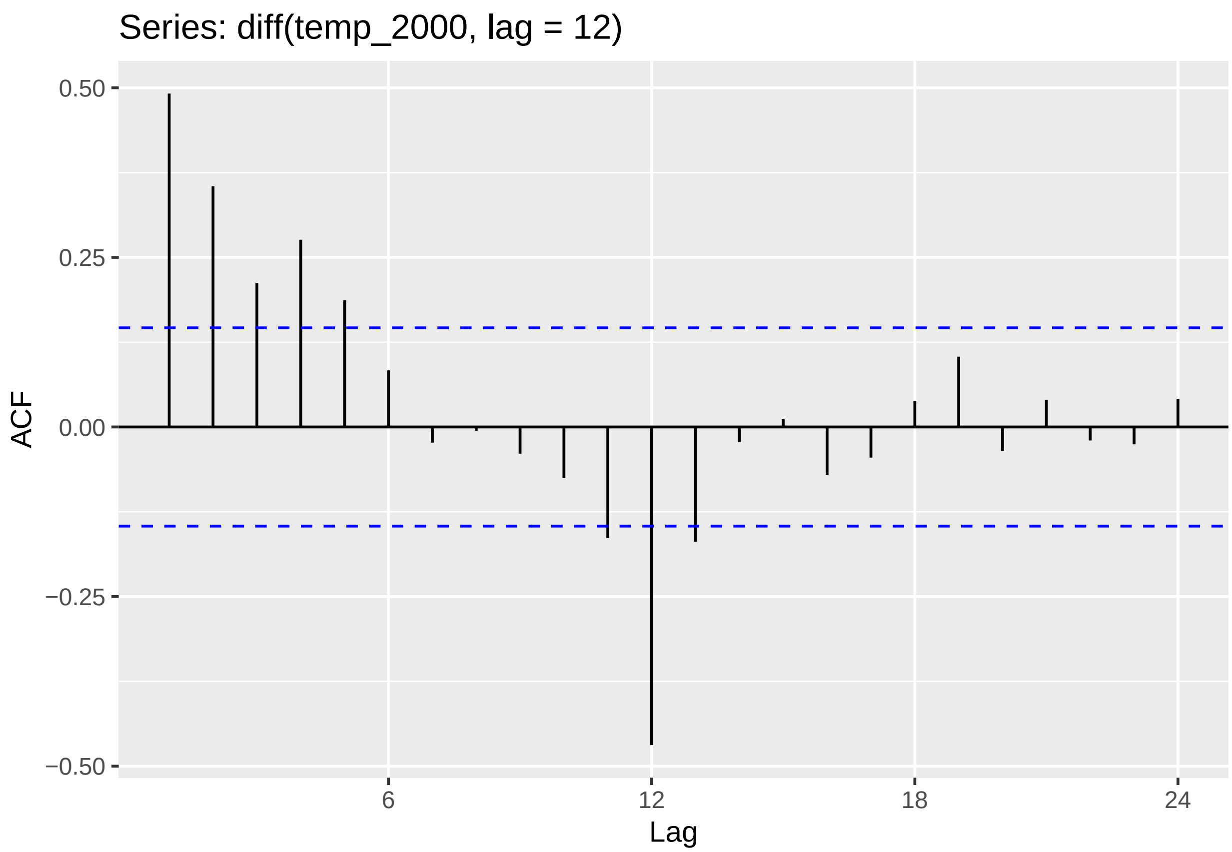
```
## Warning in adf.test(diff(temperatures)): p-value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  diff(temperatures)
## Dickey-Fuller = -17.916, Lag order = 11, p-value = 0.01
## alternative hypothesis: stationary

## Warning in kpss.test(diff(temperatures)): p-value greater than printed p-value

##
##  KPSS Test for Level Stationarity
##
## data:  diff(temperatures)
## KPSS Level = 0.0043223, Truncation lag parameter = 7, p-value = 0.1
```

the differentiated data is stationary.

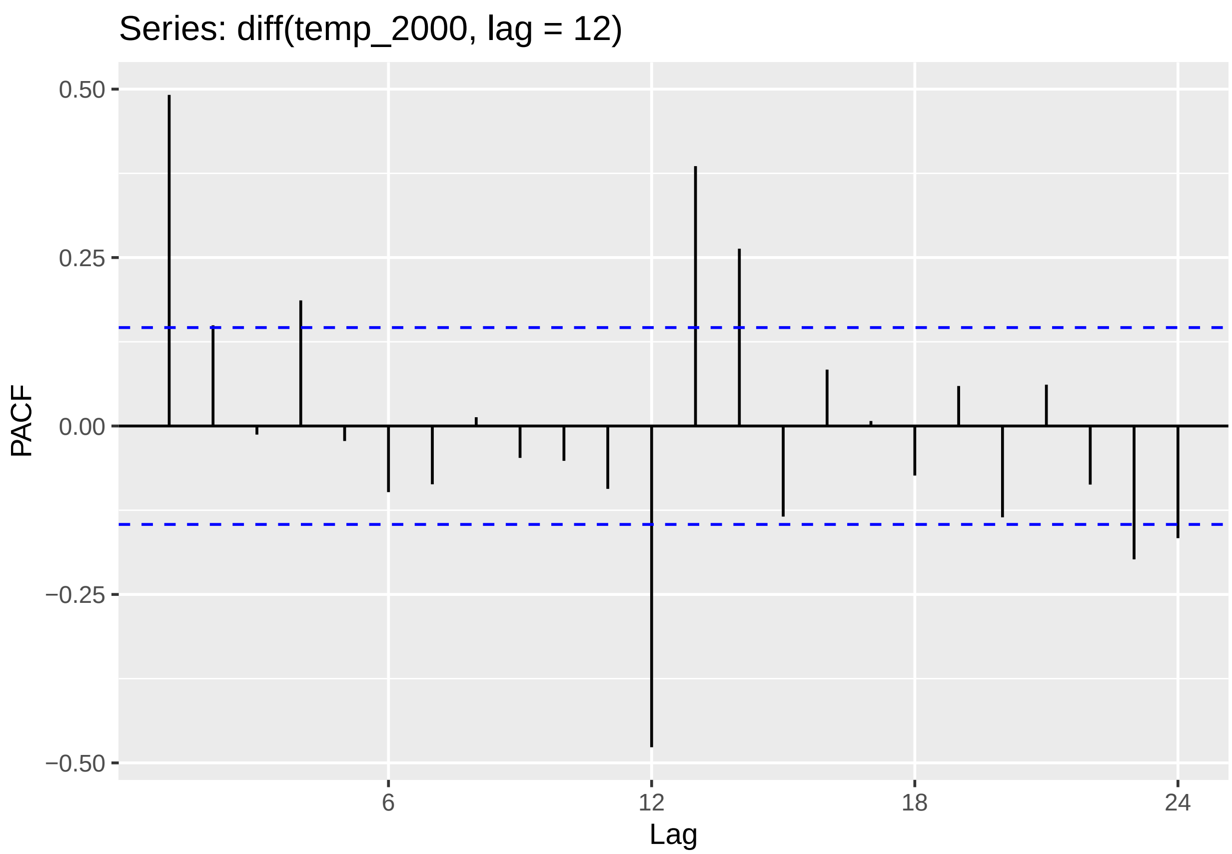
Despite no indication that the data has changed its trend, using a subset of the data seems to have introduced a bias in which the subset seemed stationary while it is not. maybe in this specific case, the predicted overall slope was considered to be too small to be significant. which made it the unstability unpredictable by the KPSS and ADF tests.

Fitting arima model

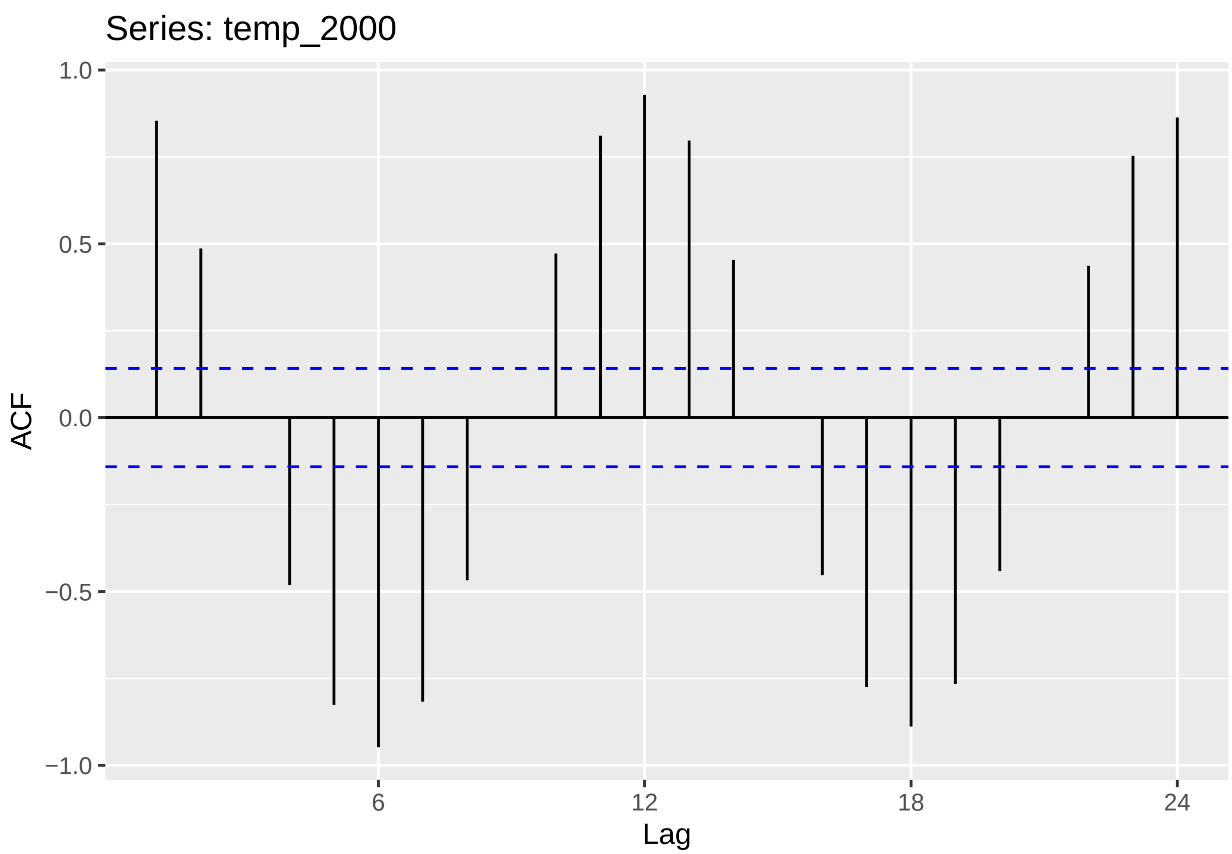
```r
ggAcf(diff(temp_2000, lag = 12))
```



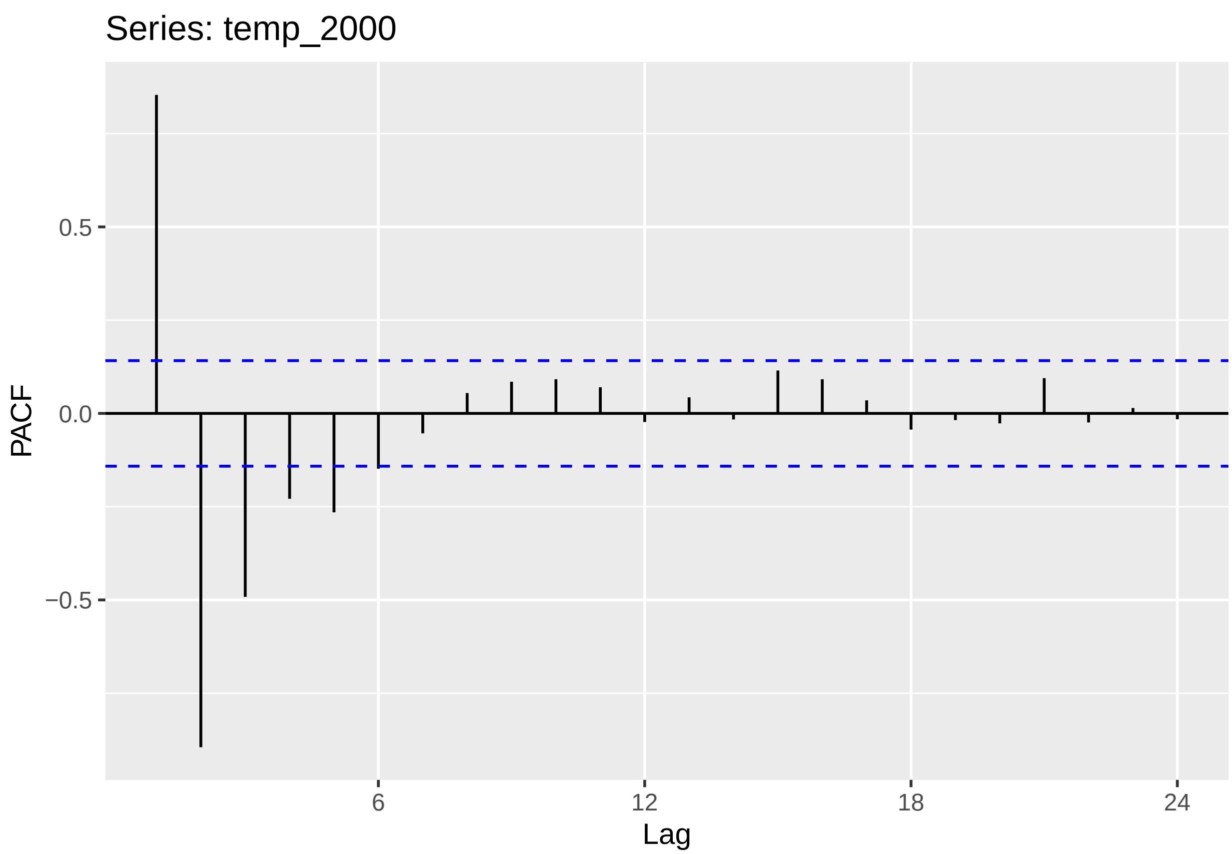

```r
ggPacf(diff(temp_2000, lag = 12))
```




```r
ggAcf(temp_2000)
```




```r
ggPacf(temp_2000)
```



in terms of the seasonal parts, there are strong spikes in the ACF plot at the seasonal lag= 12 similar argument with the PACF plot. Whereas in the non-seasonal parts, we have spikes at $p = 1, 2$ and the ACF is sinusoidal ($q = 0$)

which suggests a sARIMA model of order (0, 1, 1)[12], (1, 1, 0)[12] or (1,1 , 1)[12] and the non-seasonal part is (1, 0, 0), (2, 0, 0)

Using auto.arima:

```r
temp_arima_fit <- auto.arima(temp_2000, approximation = FALSE, stepwise = FALSE)
summary(temp_arima_fit)
```

```
## Series: temp_2000
## ARIMA(2,0,1)(1,1,1)[12] with drift
##
## Coefficients:
##           ar1     ar2     ma1     sar1     sma1   drift
##       -0.0794  0.6451  0.6358  -0.2897  -0.8181  0.0013
## s.e.   0.1159  0.0810  0.1255   0.0963   0.1067  0.0005
##
## sigma^2 = 0.009379:  log likelihood = 158.08
## AIC=-302.15   AICc=-301.5   BIC=-279.8
##
## Training set error measures:
##                         ME       RMSE        MAE        MPE      MAPE      MASE
## Training set 0.004146291 0.09219274 0.07130547 0.01833381 0.4616249 0.5353934
##                      ACF1
```

```
## Training set 0.01137257
```

trying other models

```r
print(c(
  AIC(Arima(temp_2000, c(2, 0, 0), c(1, 1, 1), include.drift = TRUE))<AIC(temp_arima_fit),
  AIC(Arima(temp_2000, c(1, 0, 0), c(1, 1, 1), include.drift = TRUE))<AIC(temp_arima_fit),
  AIC(Arima(temp_2000, c(2, 0, 0), c(0, 1, 1), include.drift = TRUE))<AIC(temp_arima_fit),
  AIC(Arima(temp_2000, c(1, 0, 0), c(0, 1, 1), include.drift = TRUE))<AIC(temp_arima_fit),
  AIC(Arima(temp_2000, c(2, 0, 0), c(1, 1, 0), include.drift = TRUE))<AIC(temp_arima_fit),
  AIC(Arima(temp_2000, c(1, 0, 0), c(1, 1, 0), include.drift = TRUE))<AIC(temp_arima_fit)
))
```

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

Forcasting

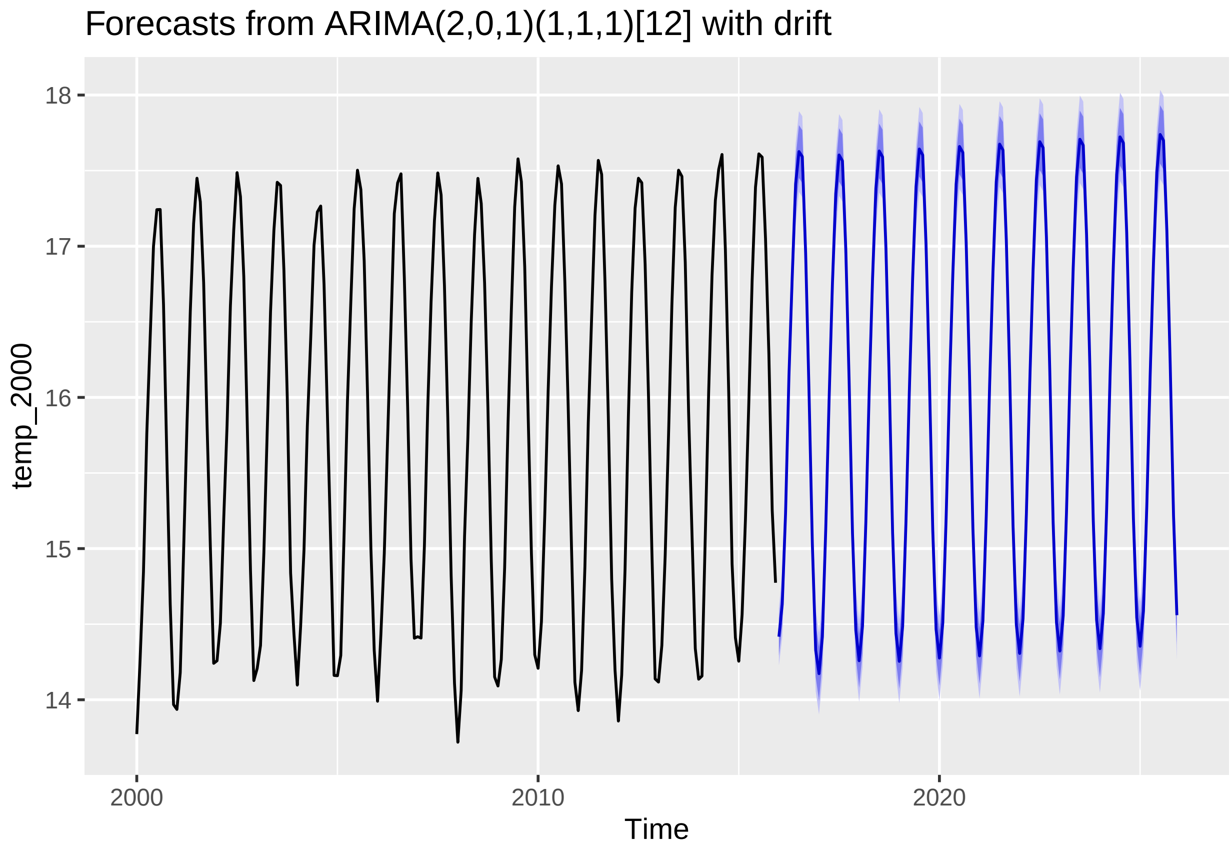
```r
arima_Forecast_temp = forecast(temp_arima_fit, h = 120)
arima_Forecast_temp
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Jan 2016       14.41731 14.29315 14.54147 14.22743 14.60720
## Feb 2016       14.63149 14.48940 14.77357 14.41419 14.84878
## Mar 2016       15.23645 15.07598 15.39693 14.99103 15.48188
## Apr 2016       16.13987 15.97481 16.30493 15.88744 16.39231
## May 2016       16.80719 16.63609 16.97829 16.54552 17.06886
## Jun 2016       17.41202 17.23960 17.58445 17.14832 17.67573
## Jul 2016       17.62634 17.45176 17.80092 17.35934 17.89334
## Aug 2016       17.59268 17.41772 17.76765 17.32509 17.86028
## Sep 2016       16.95340 16.77763 17.12916 16.68459 17.22221
## Oct 2016       16.04440 15.86853 16.22027 15.77543 16.31337
## Nov 2016       15.02914 14.85297 15.20531 14.75971 15.29857
## Dec 2016       14.32722 14.15102 14.50341 14.05775 14.59668
## Jan 2017       14.17251 13.99619 14.34883 13.90285 14.44217
## Feb 2017       14.41636 14.23994 14.59278 14.14655 14.68617
## Mar 2017       15.12071 14.94425 15.29718 14.85083 15.39059
## Apr 2017       15.97927 15.80277 16.15577 15.70934 16.24921
## May 2017       16.74860 16.57208 16.92511 16.47864 17.01855
## Jun 2017       17.34808 17.17156 17.52461 17.07811 17.61806
## Jul 2017       17.60378 17.42725 17.78031 17.33380 17.87376
## Aug 2017       17.56483 17.38830 17.74137 17.29484 17.83482
## Sep 2017       16.98044 16.80391 17.15698 16.71045 17.25043
## Oct 2017       16.10724 15.93070 16.28378 15.83725 16.37723
## Nov 2017       15.10285 14.92632 15.27939 14.83286 15.37284
## Dec 2017       14.45948 14.28294 14.63601 14.18949 14.72947
## Jan 2018       14.25802 14.07955 14.43650 13.98507 14.53097
## Feb 2018       14.48825 14.30919 14.66732 14.21441 14.76210
## Mar 2018       15.17180 14.99205 15.35156 14.89689 15.44672
## Apr 2018       16.03949 15.85955 16.21943 15.76430 16.31468
## May 2018       16.78472 16.60453 16.96491 16.50914 17.06030
## Jun 2018       17.38282 17.20258 17.56307 17.10716 17.65849
```

```
## Jul 2018       17.63029 17.44995 17.81063 17.35448 17.90609
## Aug 2018       17.59069 17.41033 17.77104 17.31486 17.86651
## Sep 2018       16.99299 16.81260 17.17338 16.71711 17.26887
## Oct 2018       16.10779 15.92740 16.28819 15.83191 16.38368
## Nov 2018       15.10206 14.92166 15.28246 14.82616 15.37796
## Dec 2018       14.44054 14.26013 14.62094 14.16463 14.71644
## Jan 2019       14.25388 14.07275 14.43502 13.97686 14.53090
## Feb 2019       14.48720 14.30586 14.66854 14.20986 14.76453
## Mar 2019       15.17765 14.99604 15.35926 14.89991 15.45540
## Apr 2019       16.04207 15.86040 16.22374 15.76422 16.31991
## May 2019       16.79489 16.61313 16.97666 16.51690 17.07289
## Jun 2019       17.39294 17.21116 17.57473 17.11492 17.67096
## Jul 2019       17.64323 17.46141 17.82505 17.36515 17.92130
## Aug 2019       17.60349 17.42166 17.78532 17.32540 17.88157
## Sep 2019       17.00995 16.82811 17.19179 16.73185 17.28806
## Oct 2019       16.12800 15.94615 16.30984 15.84989 16.40610
## Nov 2019       15.12287 14.94102 15.30472 14.84476 15.40098
## Dec 2019       14.46643 14.28459 14.64828 14.18833 14.74454
## Jan 2020       14.27565 14.09280 14.45849 13.99601 14.55529
## Feb 2020       14.50794 14.32480 14.69108 14.22786 14.78803
## Mar 2020       15.19651 15.01300 15.38001 14.91586 15.47715
## Apr 2020       16.06178 15.87819 16.24538 15.78100 16.34257
## May 2020       16.81249 16.62876 16.99621 16.53150 17.09347
## Jun 2020       17.41049 17.22674 17.59424 17.12946 17.69152
## Jul 2020       17.66001 17.47621 17.84382 17.37891 17.94111
## Aug 2020       17.62027 17.43646 17.80408 17.33915 17.90138
## Sep 2020       17.02556 16.84174 17.20939 16.74442 17.30671
## Oct 2020       16.14264 15.95881 16.32647 15.86149 16.42378
## Nov 2020       15.13736 14.95353 15.32120 14.85621 15.41852
## Dec 2020       14.47943 14.29560 14.66327 14.19828 14.76058
## Jan 2021       14.28986 14.10512 14.47460 14.00732 14.57240
## Feb 2021       14.52244 14.33743 14.70744 14.23950 14.80538
## Mar 2021       15.21156 15.02623 15.39690 14.92812 15.49501
## Apr 2021       16.07658 15.89116 16.26199 15.79301 16.36014
## May 2021       16.82791 16.64237 17.01344 16.54415 17.11166
## Jun 2021       17.42591 17.24035 17.61147 17.14212 17.70970
## Jul 2021       17.67566 17.49006 17.86127 17.39181 17.95952
## Aug 2021       17.63591 17.45030 17.82153 17.35205 17.91978
## Sep 2021       17.04155 16.85593 17.22718 16.75766 17.32545
## Oct 2021       16.15891 15.97328 16.34453 15.87501 16.44280
## Nov 2021       15.15368 14.96804 15.33931 14.86978 15.43758
## Dec 2021       14.49618 14.31054 14.68181 14.21228 14.78008
## Jan 2022       14.30625 14.11970 14.49281 14.02094 14.59157
## Feb 2022       14.53875 14.35192 14.72557 14.25303 14.82447
## Mar 2022       15.22771 15.04055 15.41487 14.94148 15.51394
## Apr 2022       16.09280 15.90556 16.28004 15.80644 16.37916
## May 2022       16.84395 16.65659 17.03131 16.55740 17.13050
## Jun 2022       17.44196 17.25457 17.62934 17.15537 17.72854
## Jul 2022       17.69164 17.50421 17.87907 17.40499 17.97830
## Aug 2022       17.65189 17.46445 17.83933 17.36523 17.93856
## Sep 2022       17.05743 16.86998 17.24489 16.77074 17.34412
## Oct 2022       16.17470 15.98725 16.36216 15.88801 16.46140
## Nov 2022       15.16946 14.98200 15.35692 14.88276 15.45616
## Dec 2022       14.51184 14.32437 14.69930 14.22514 14.79853
```

```
## Jan 2023       14.32202 14.13365 14.51038 14.03393 14.61010
## Feb 2023       14.55453 14.36590 14.74316 14.26605 14.84302
## Mar 2023       15.24354 15.05458 15.43250 14.95455 15.53253
## Apr 2023       16.10861 15.91957 16.29765 15.81950 16.39773
## May 2023       16.85981 16.67065 17.04898 16.57051 17.14911
## Jun 2023       17.45782 17.26863 17.64701 17.16848 17.74716
## Jul 2023       17.70752 17.51829 17.89675 17.41812 17.99693
## Aug 2023       17.66777 17.47854 17.85701 17.37836 17.95719
## Sep 2023       17.07334 16.88409 17.26260 16.78390 17.36278
## Oct 2023       16.19064 16.00138 16.37989 15.90120 16.48008
## Nov 2023       15.18540 14.99614 15.37466 14.89595 15.47485
## Dec 2023       14.52781 14.33855 14.71707 14.23836 14.81726
## Jan 2024       14.33796 14.14780 14.52812 14.04714 14.62878
## Feb 2024       14.57047 14.38005 14.76089 14.27925 14.86169
## Mar 2024       15.25947 15.06872 15.45022 14.96774 15.55119
## Apr 2024       16.12454 15.93371 16.31537 15.83270 16.41639
## May 2024       16.87573 16.68478 17.06668 16.58370 17.16776
## Jun 2024       17.47374 17.28276 17.66471 17.18167 17.76580
## Jul 2024       17.72343 17.53242 17.91445 17.43130 18.01557
## Aug 2024       17.68368 17.49266 17.87471 17.39154 17.97583
## Sep 2024       17.08925 16.89821 17.28028 16.79708 17.38141
## Oct 2024       16.20653 16.01549 16.39757 15.91436 16.49870
## Nov 2024       15.20129 15.01025 15.39234 14.90912 15.49347
## Dec 2024       14.54369 14.35265 14.73474 14.25152 14.83587
## Jan 2025       14.35385 14.16192 14.54579 14.06031 14.64739
## Feb 2025       14.58636 14.39417 14.77856 14.29243 14.88030
## Mar 2025       15.27537 15.08285 15.46788 14.98094 15.56979
## Apr 2025       16.14044 15.94784 16.33303 15.84589 16.43499
## May 2025       16.89163 16.69891 17.08434 16.59690 17.18636
## Jun 2025       17.48964 17.29690 17.68237 17.19487 17.78440
## Jul 2025       17.73933 17.54655 17.93212 17.44450 18.03417
## Aug 2025       17.69958 17.50679 17.89237 17.40474 17.99443
## Sep 2025       17.10515 16.91234 17.29796 16.81028 17.40002
## Oct 2025       16.22244 16.02963 16.41525 15.92757 16.51731
## Nov 2025       15.21720 15.02439 15.41001 14.92232 15.51208
## Dec 2025       14.55960 14.36679 14.75241 14.26472 14.85448
```

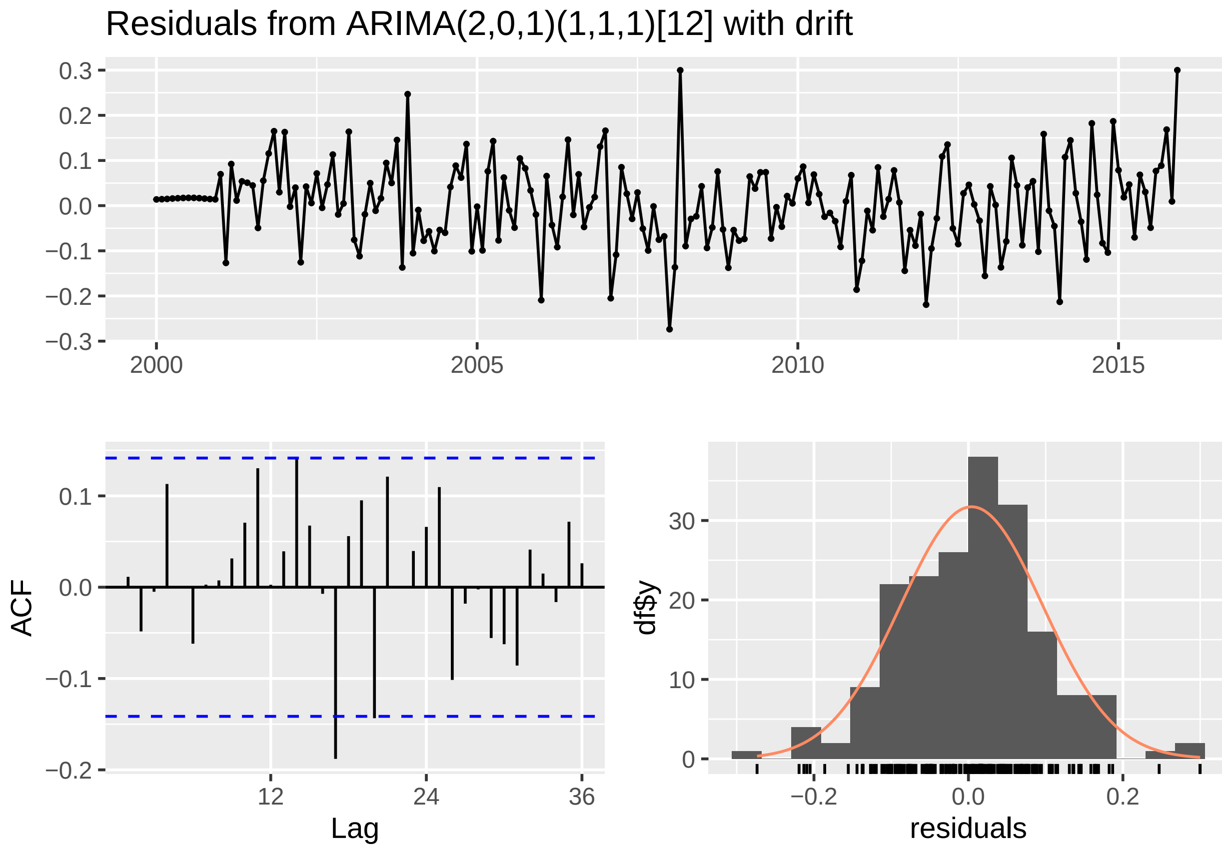
```r
autoplot(arima_Forecast_temp)
```



the forecasted value on jan 2024 was 14.33796 ^oC which is bigger than the real value 13.14 ^oC.

residuals check

```r
checkresiduals(arima_Forecast_temp, lag = 12)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,1)(1,1,1)[12] with drift
## Q* = 8.5176, df = 7, p-value = 0.2892
##
## Model df: 5.   Total lags used: 12
```