# Code PDF

Tomás Gonçalves

2024-08-06

# 1 - Import Libraries

```
library(tseries)
```

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

```
library(forecast)
library(urca)
library(xts)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(fpp3)
```

```
## Registered S3 method overwritten by 'tsibble':
##   method               from
##   as_tibble.grouped_df dplyr
```

```
## -- Attaching packages -------------------------------------------- fpp3 1.0.0 --
```

```
## v tibble      3.2.1     v tsibble     1.1.5
## v dplyr       1.1.4     v tsibbledata 0.4.1
## v tidyr       1.3.1     v feasts      0.3.2
## v lubridate   1.9.3     v fable       0.3.4
## v ggplot2     3.5.1     v fabletools  0.4.2
```

```
## -- Conflicts ------------------------------------------------- fpp3_conflicts --
## x lubridate::date()    masks base::date()
## x dplyr::filter()      masks stats::filter()
## x dplyr::first()       masks xts::first()
## x tsibble::index()     masks zoo::index()
## x tsibble::intersect() masks base::intersect()
## x tsibble::interval()  masks lubridate::interval()
## x dplyr::lag()         masks stats::lag()
## x dplyr::last()        masks xts::last()
## x tsibble::setdiff()   masks base::setdiff()
## x tsibble::union()     masks base::union()
```

```r
library(AER)
```

```
## Loading required package: car

## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## Loading required package: lmtest

## Loading required package: sandwich

## Loading required package: survival
```

```r
library(strucchange)
library(tsibble)
library(dplyr)
library(ggplot2)
library(feasts)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats 1.0.0     v readr   2.1.5
## v purrr   1.0.2     v stringr 1.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x stringr::boundary() masks strucchange::boundary()
## x dplyr::filter()     masks stats::filter()
## x dplyr::first()      masks xts::first()
## x tsibble::interval() masks lubridate::interval()
## x dplyr::lag()        masks stats::lag()
## x dplyr::last()       masks xts::last()
## x car::recode()       masks dplyr::recode()
## x purrr::some()       masks car::some()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(gets)
```

```
## Loading required package: parallel
##
## Attaching package: 'gets'
##
## The following object is masked from 'package:car':
##
##     logit
```

```r
library(fable)


## Set the seed for reproducibility
set.seed(1234)

## Set path:
path <- 'C:\\Users\\tomas\\OneDrive\\Desktop\\CBS\\2nd Semester\\Predictive Analytics\\FinalExam2'
setwd(path)
getwd()
```

```
## [1] "C:/Users/tomas/OneDrive/Desktop/CBS/2nd Semester/Predictive Analytics/FinalExam2"
```

# 2 - Data Exploration

## 2.1 - Load Data

```r
pt <- read.csv("PortugalRealGDP.csv", sep = ",")
pt <- pt %>%
  mutate(DATE = as.Date(DATE, format="%m/%d/%Y"),
         date = yearquarter(DATE)) %>%
  select(-DATE) %>%  # Keep 'date' and remove 'DATE'
  mutate(gdp = GDP) %>%  # Create 'gdp' column from 'GDP'
  select(-GDP) %>%  # Remove the original 'GDP' column
  as_tsibble(index = date)
```

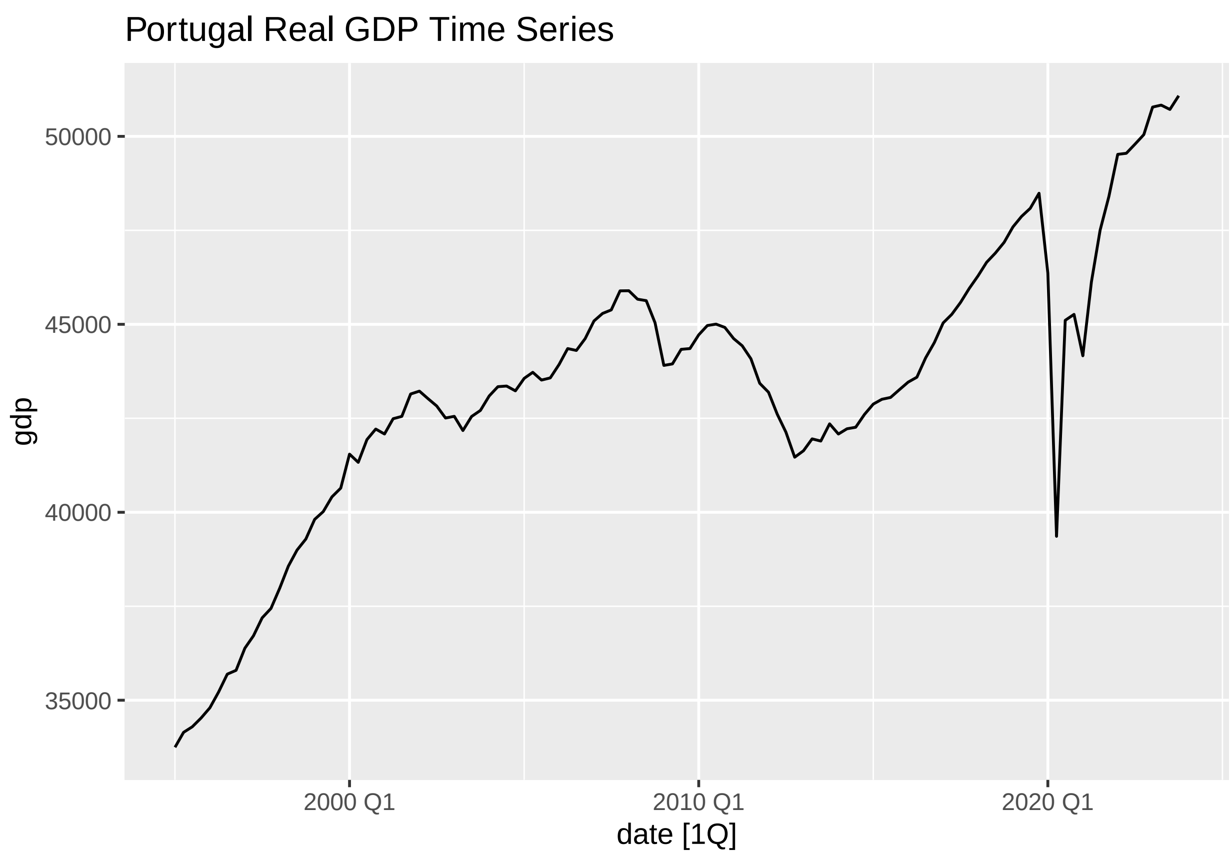
## 2.2 - Check Data

```r
print(pt)
```

```
## # A tsibble: 116 x 2 [1Q]
##       date    gdp
##      <qtr>  <dbl>
##  1 1995 Q1 33747.
##  2 1995 Q2 34145.
##  3 1995 Q3 34294.
##  4 1995 Q4 34526.
##  5 1996 Q1 34799.
##  6 1996 Q2 35216
##  7 1996 Q3 35694.
##  8 1996 Q4 35795.
##  9 1997 Q1 36383.
## 10 1997 Q2 36714.
## # i 106 more rows
```

### 2.3 - Check Time Series

```r
autoplot(pt, gdp) + ggtitle("Portugal Real GDP Time Series")
```



### 2.4 - Verify Missing Values

```
print(sum(is.na(pt)))
```

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

### 2.5 - Verify Outliers

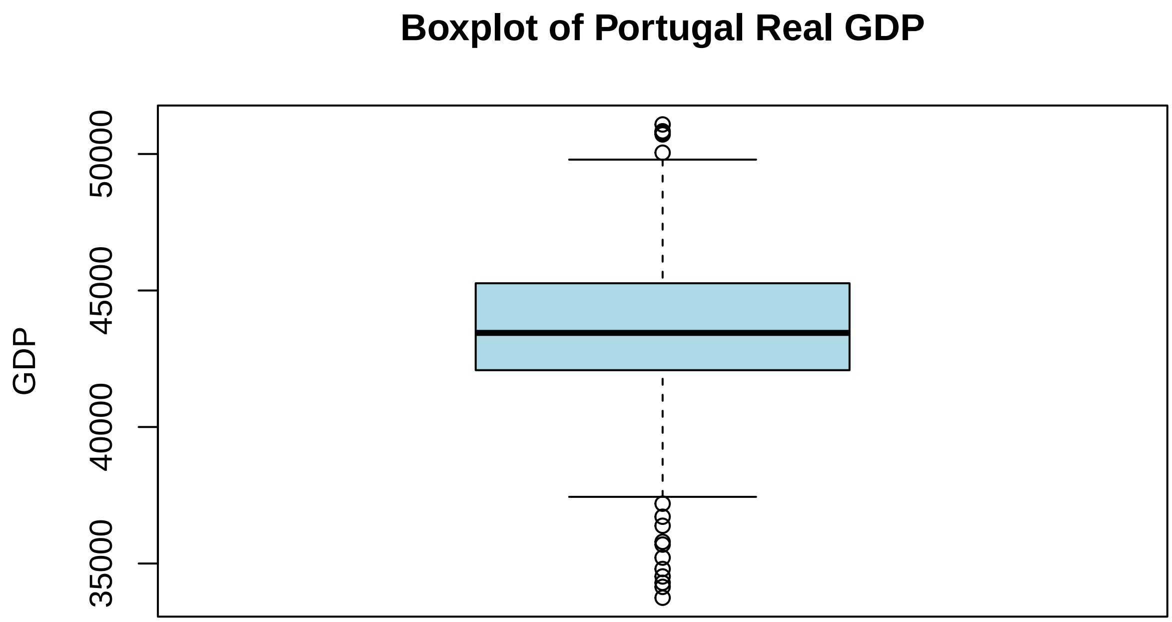
```
Q1 <- quantile(pt$gdp, 0.25)
Q3 <- quantile(pt$gdp, 0.75)
IQR <- Q3 - Q1

# Boundaries
lower_bound <- Q1 - 1.5 * IQR
upper_bound <- Q3 + 1.5 * IQR

# Identify outliers
outliers <- pt %>%
  filter(gdp < lower_bound | gdp > upper_bound)

cat("Number of outliers: ", nrow(outliers), "\n")
```

```
## Number of outliers:  16
```

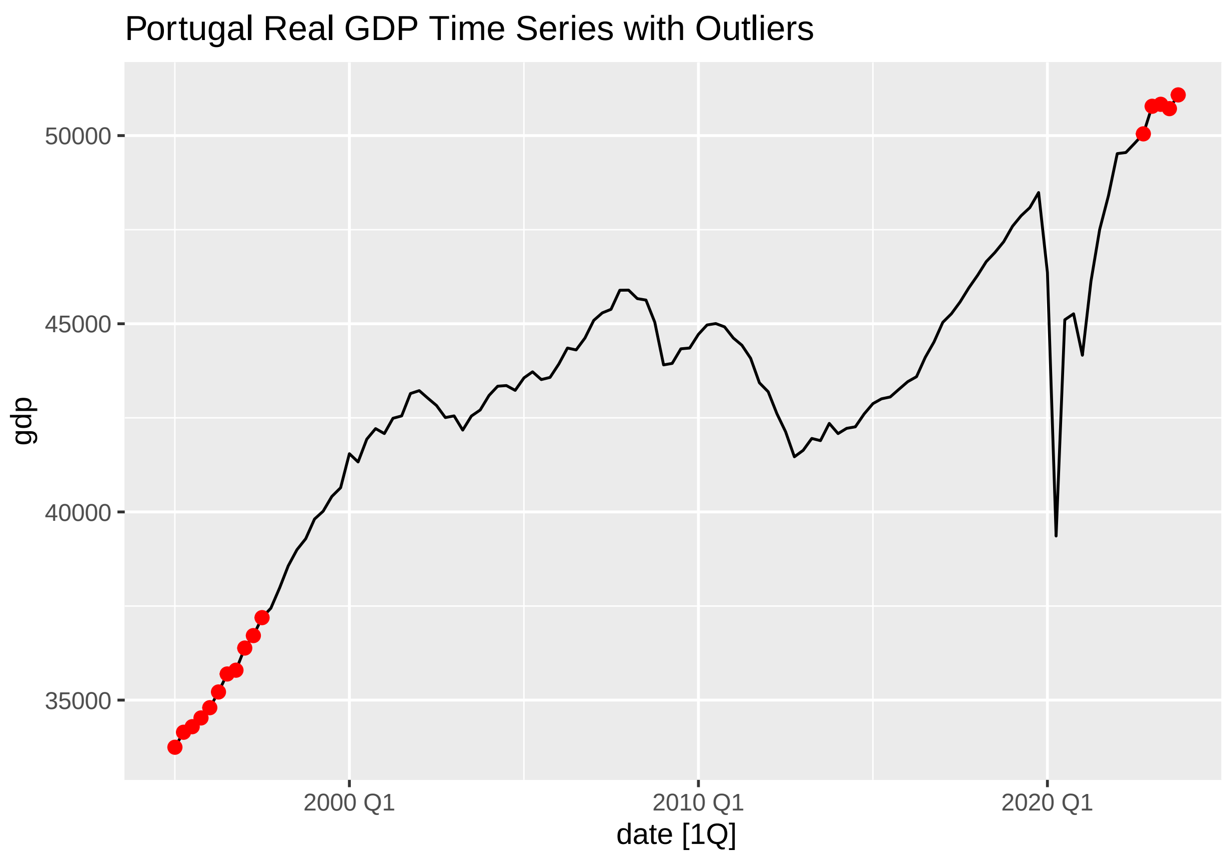
```
print(outliers)
```

```
## # A tsibble: 16 x 2 [1Q]
##       date    gdp
##      <qtr>  <dbl>
##  1 1995 Q1 33747.
##  2 1995 Q2 34145.
##  3 1995 Q3 34294.
##  4 1995 Q4 34526.
##  5 1996 Q1 34799.
##  6 1996 Q2 35216
##  7 1996 Q3 35694.
##  8 1996 Q4 35795.
##  9 1997 Q1 36383.
## 10 1997 Q2 36714.
## 11 1997 Q3 37192.
## 12 2022 Q4 50046.
## 13 2023 Q1 50779.
## 14 2023 Q2 50832.
## 15 2023 Q3 50716.
## 16 2023 Q4 51082.
```

```
# Visualize outliers using boxplot
boxplot(pt$gdp, main = "Boxplot of Portugal Real GDP", ylab = "GDP", col = "lightblue")
```

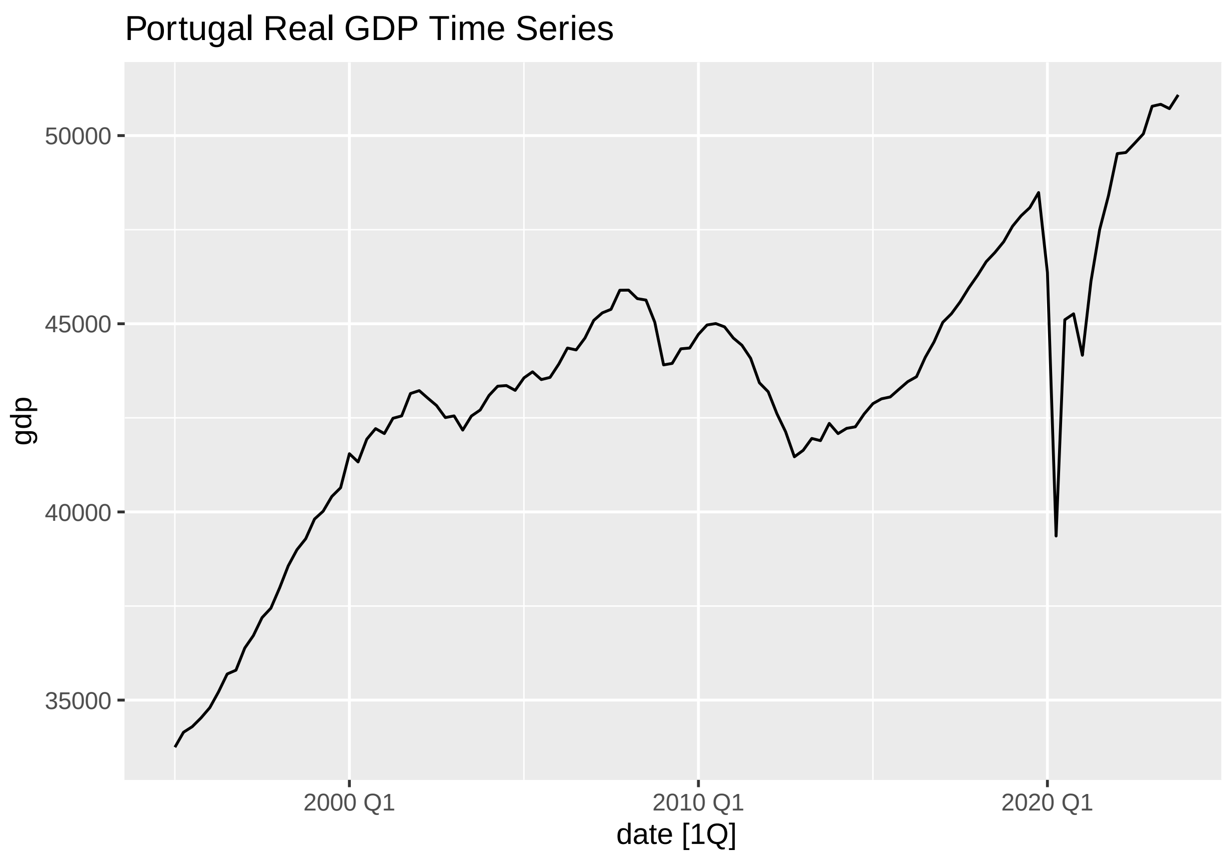



```r
# Mark outliers on the time series plot
autoplot(pt, gdp) +
  geom_point(data = outliers, aes(x = date, y = gdp), color = "red", size = 2) +
  ggtitle("Portugal Real GDP Time Series with Outliers")
```

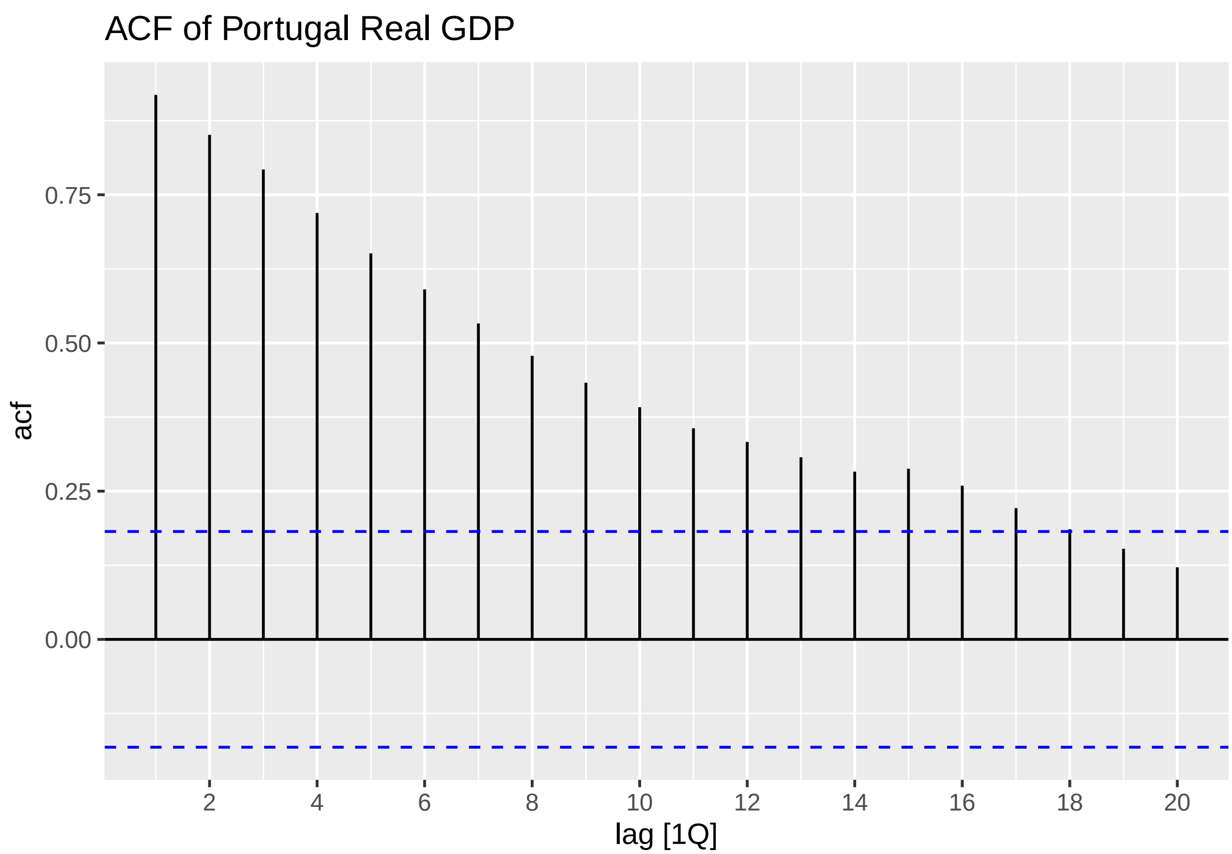



## 3 - Plots for Preliminary Analysis

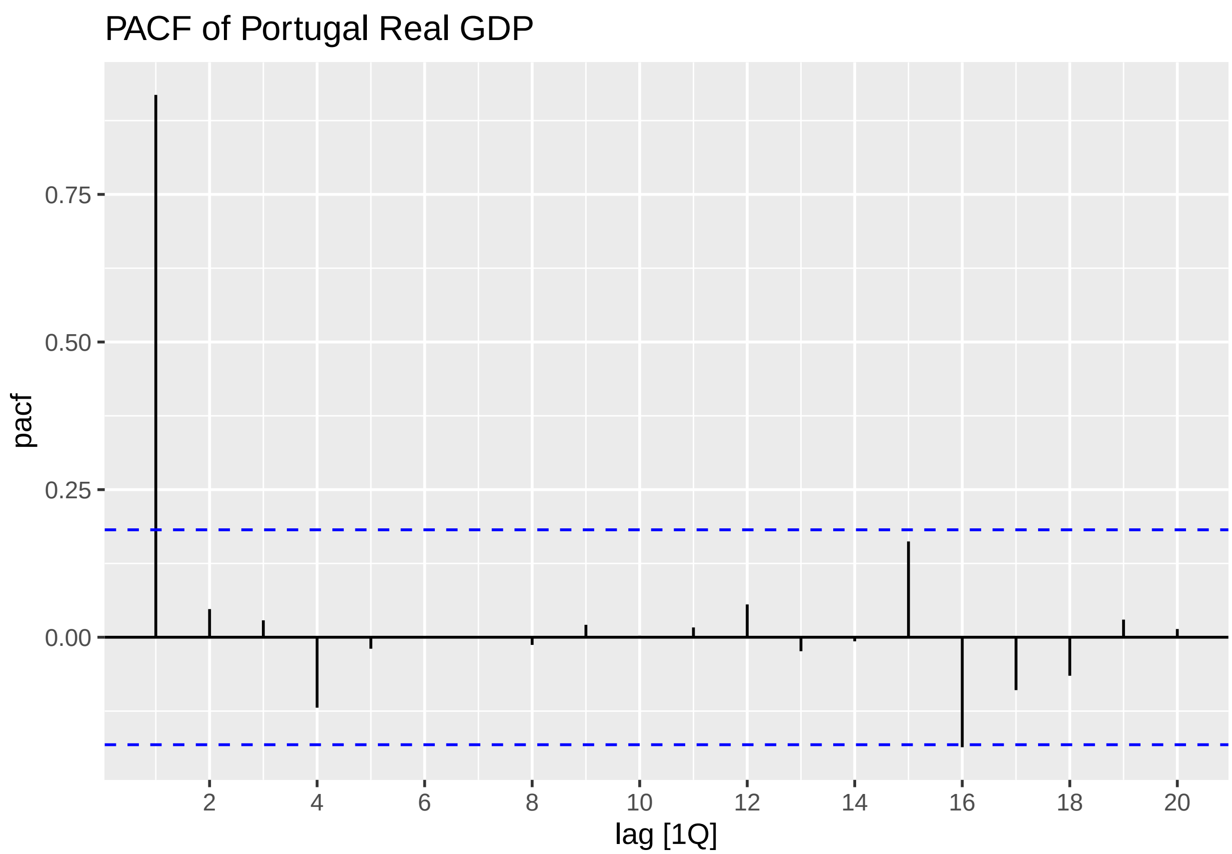
```
# Plot 1: Original time series
autoplot(pt, gdp) + ggtitle("Portugal Real GDP Time Series")
```



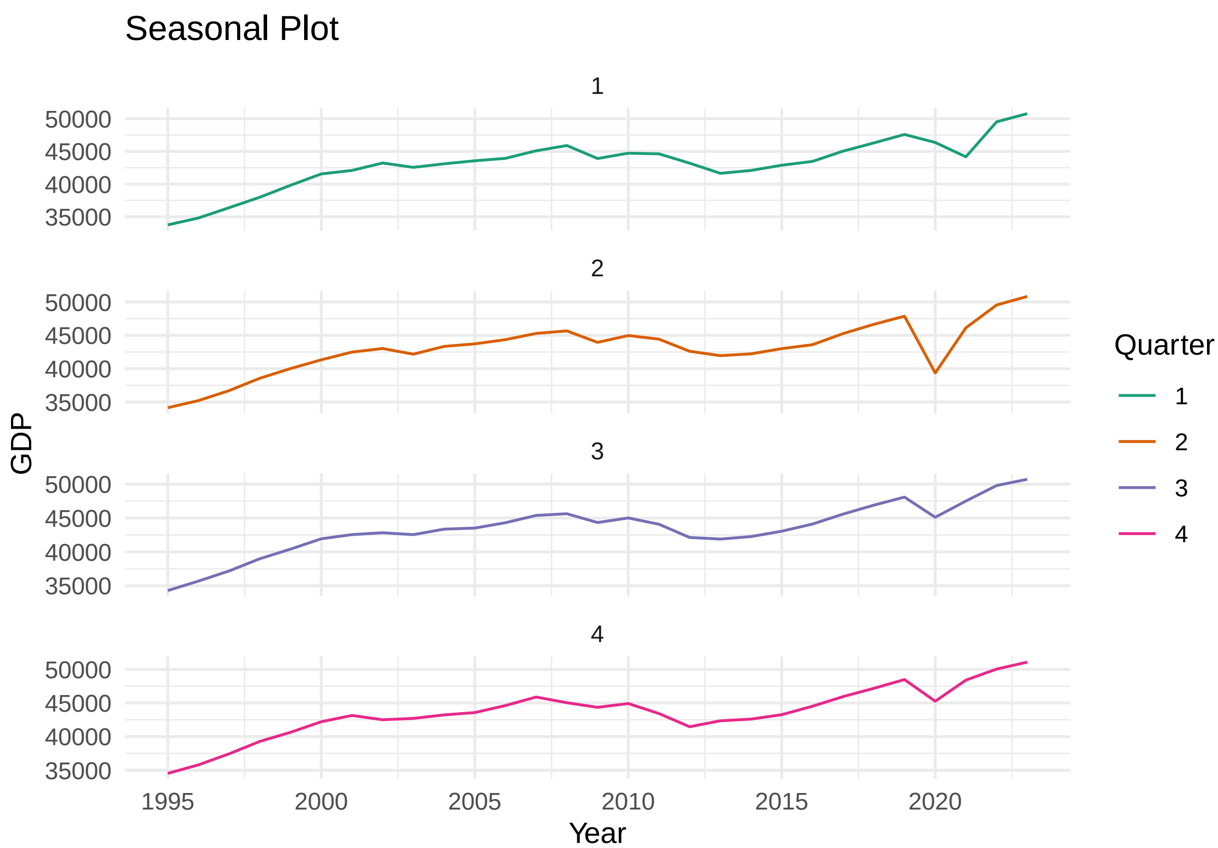

```r
# Plot 2: ACF of the original data
pt %>% ACF(gdp) %>% autoplot() + ggtitle("ACF of Portugal Real GDP")
```



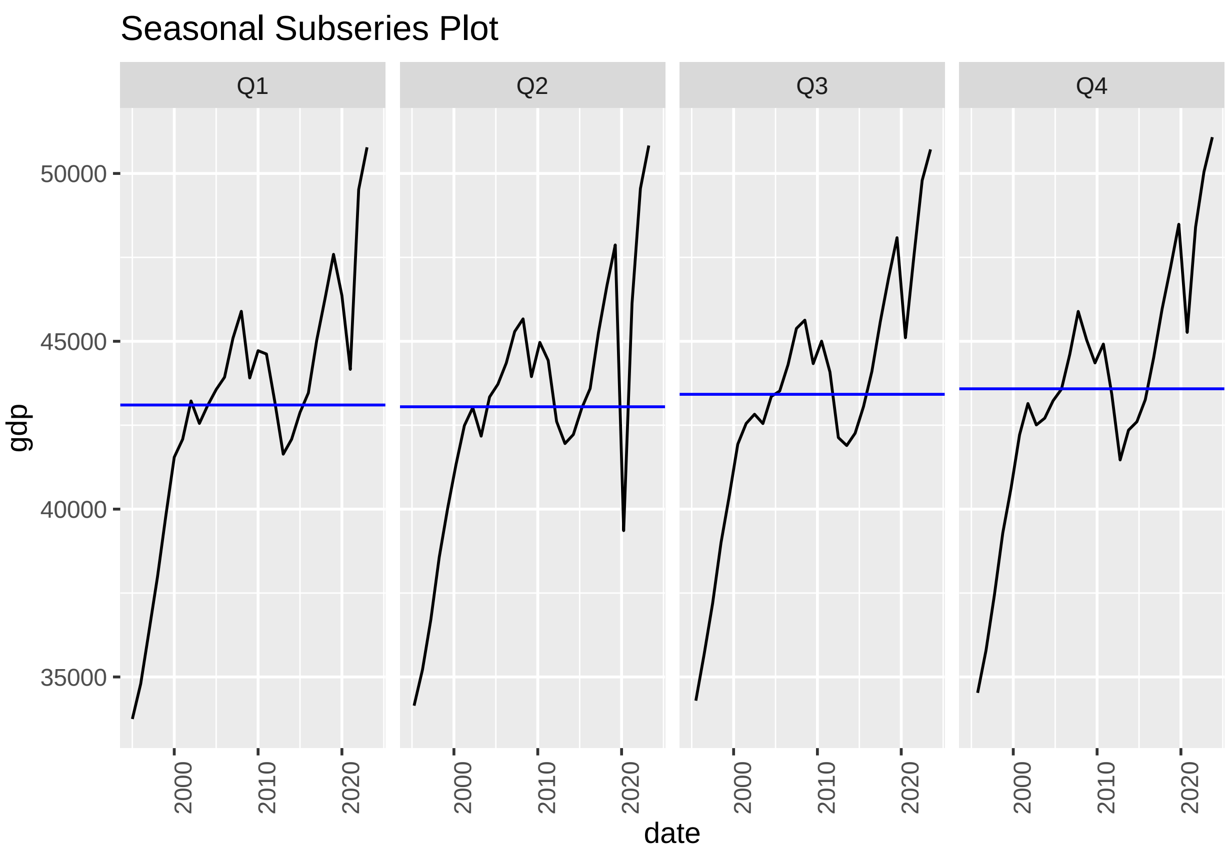

```
# Plot 3: PACF of the original data
pt %>% PACF(gdp) %>% autoplot() + ggtitle("PACF of Portugal Real GDP")
```




```r
# Plot 4: Seasonal plot
pt %>%
  mutate(quarter = quarter(date)) %>%
  ggplot(aes(x = year(date), y = gdp, color = factor(quarter))) +
  geom_line() +
  labs(title = "Seasonal Plot", x = "Year", y = "GDP", color = "Quarter") +
  theme_minimal() +
  scale_color_brewer(palette = "Dark2") +
  facet_wrap(~ quarter, scales = "free_y", ncol = 1)
```

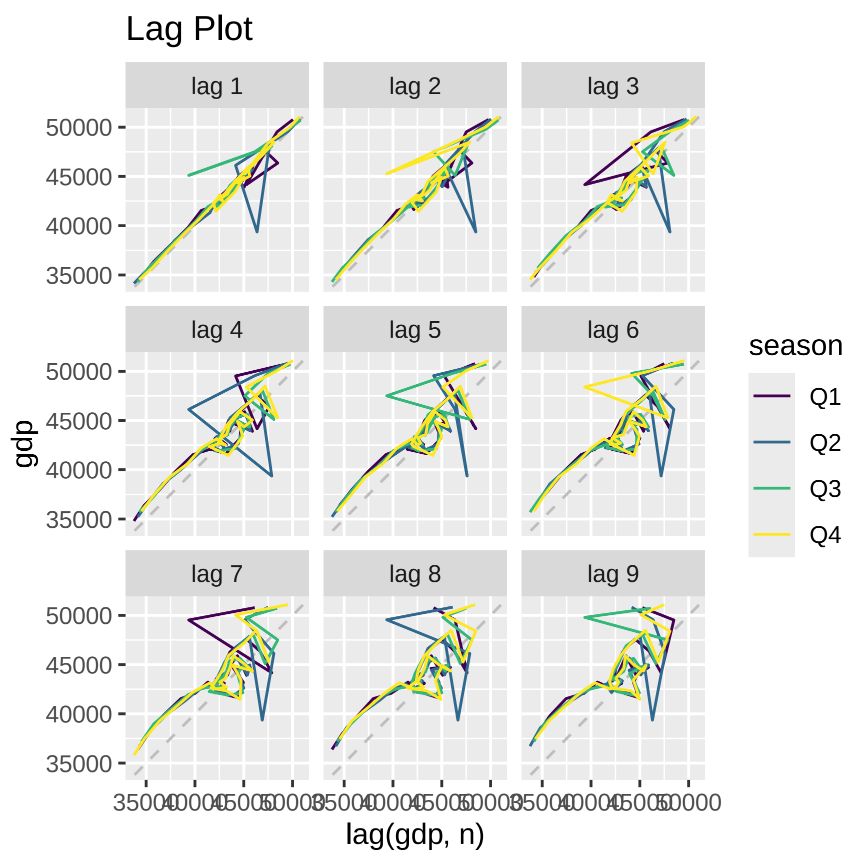



```
# Plot 5: Seasonal Subseries Plot
gg_subseries(pt, y = gdp) + ggtitle("Seasonal Subseries Plot")
```

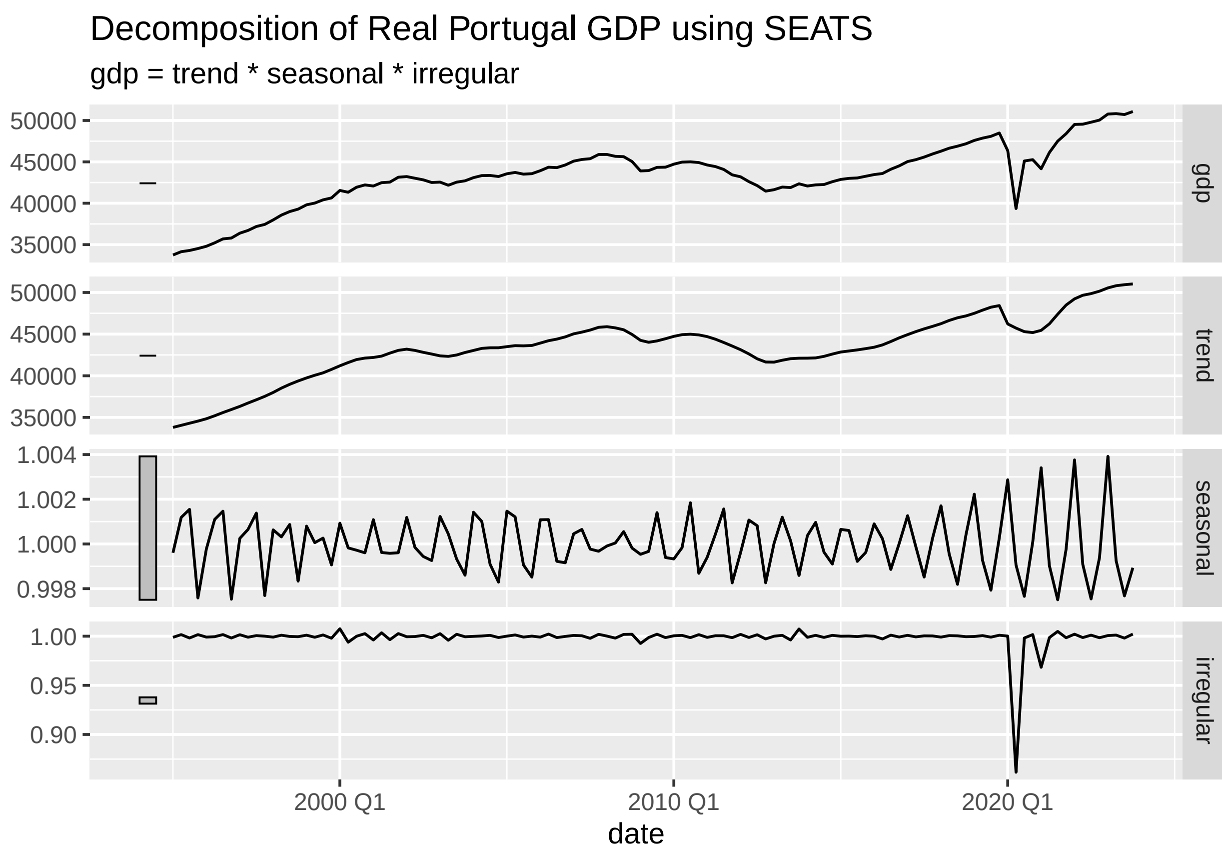



```r
# Plot 6: Lag plot of the original data
gg_lag(pt, gdp, do.lines = FALSE) + ggtitle("Lag Plot")
```

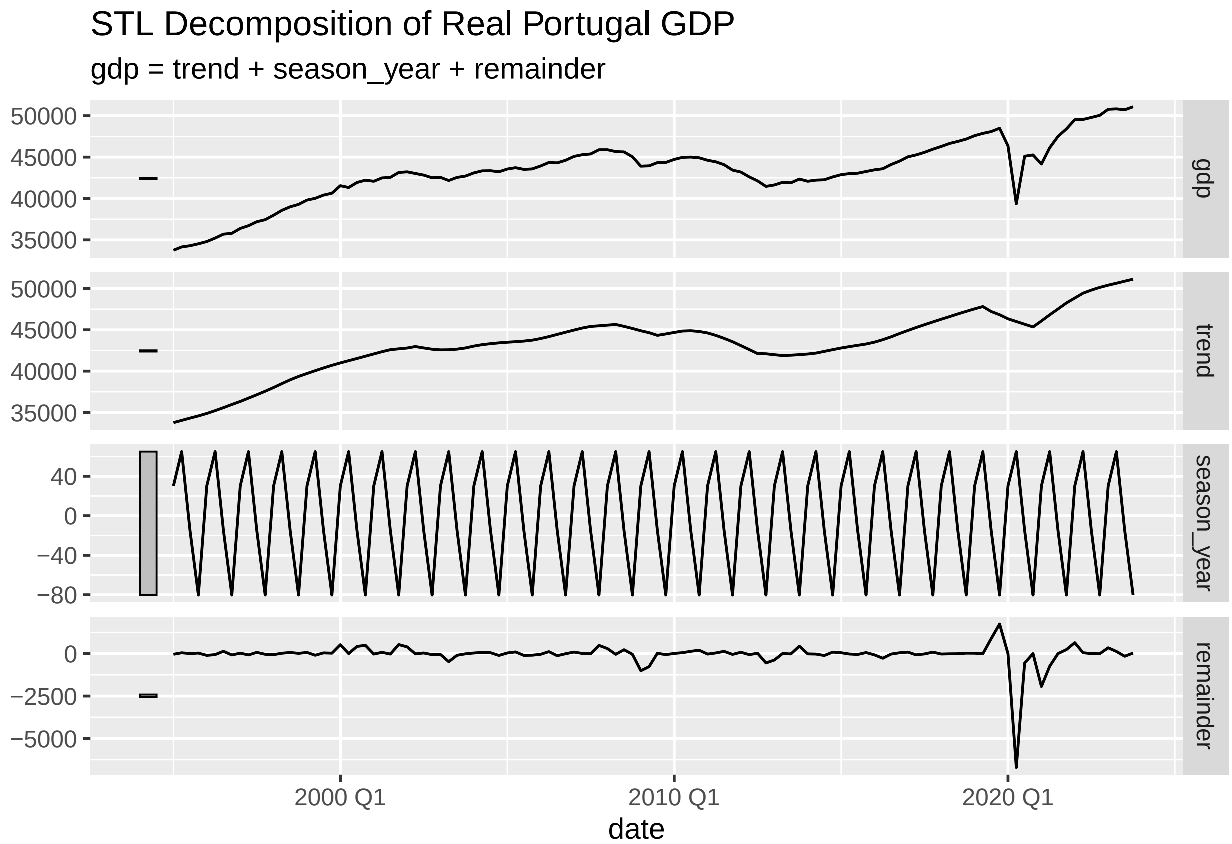
```
## Warning in lag_geom(..., arrow = arrow): Ignoring unknown parameters:
## ‘do.lines‘
```




```r
# Plot 7: SEATS Decomposition
seats_dcmp <- pt %>%
  model(seats = X_13ARIMA_SEATS(gdp ~ x11())) %>%
  components()
autoplot(seats_dcmp) +
  labs(title = "Decomposition of Real Portugal GDP using SEATS")
```




```r
# Plot 8: STL Decomposition
stl_dcmp <- pt %>%
  model(
    STL(gdp ~ trend(window = 7) +
          season(window = "periodic"),
        robust = TRUE)
  ) %>%
  components()

autoplot(stl_dcmp) +
  labs(title = "STL Decomposition of Real Portugal GDP")
```




# 4 - Box-Cox Transformation

```
lambda <- pt |>
  features(gdp, features = guerrero) |>
  pull(lambda_guerrero)

print(lambda)
```

```
## [1] -0.8999268
```

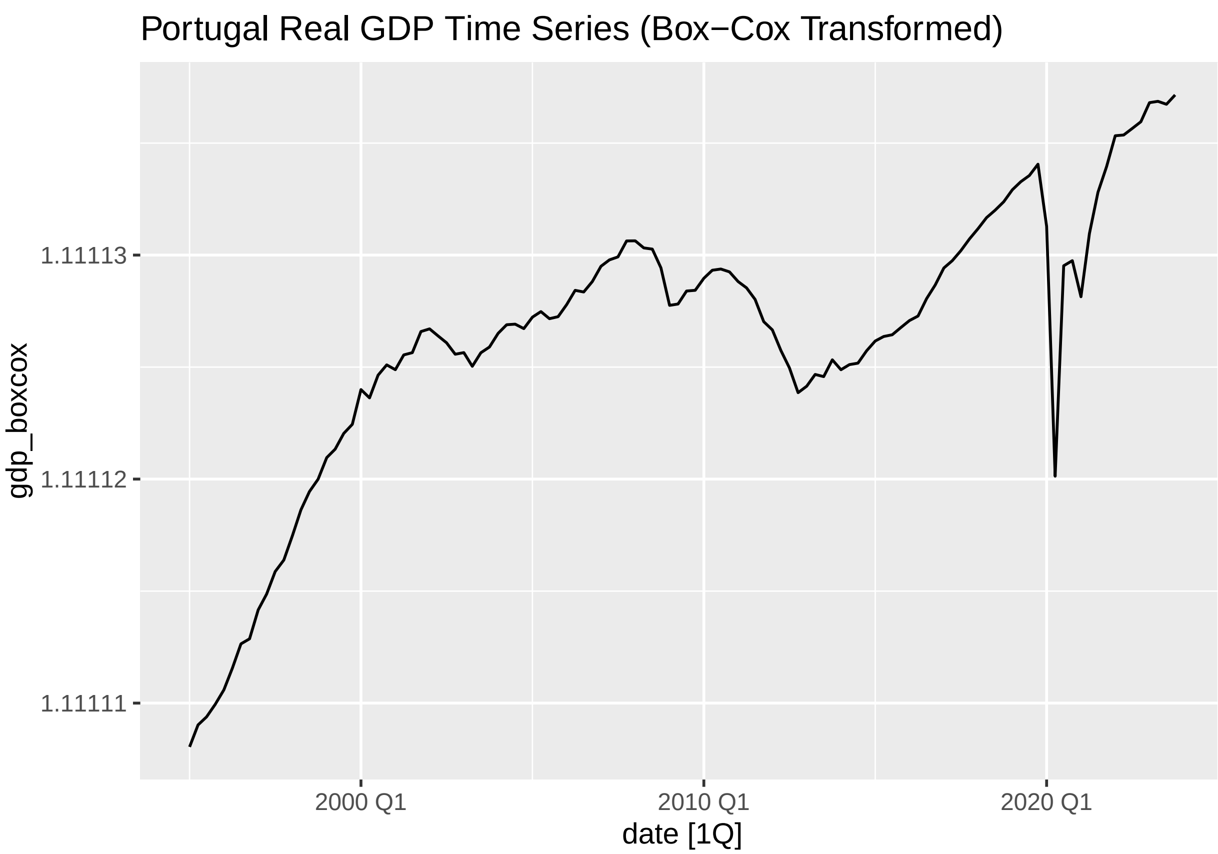
```
pt <- pt %>%
  mutate(gdp_boxcox = box_cox(gdp, lambda = lambda))

print(pt)
```

```
## # A tsibble: 116 x 3 [1Q]
##       date    gdp gdp_boxcox
##      <qtr>  <dbl>      <dbl>
##  1 1995 Q1 33747.       1.11
##  2 1995 Q2 34145.       1.11
##  3 1995 Q3 34294.       1.11
##  4 1995 Q4 34526.       1.11
```

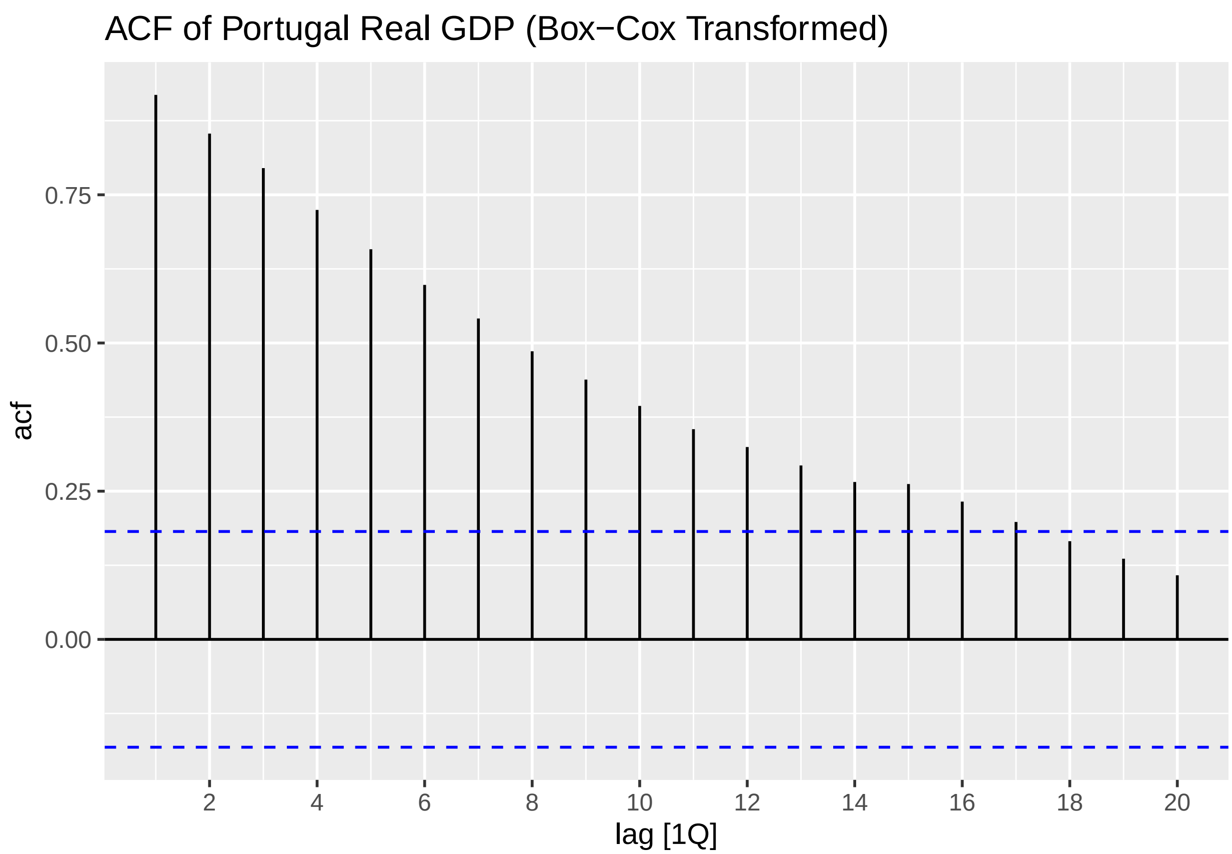
```
##   5 1996 Q1 34799.        1.11
##   6 1996 Q2 35216         1.11
##   7 1996 Q3 35694.        1.11
##   8 1996 Q4 35795.        1.11
##   9 1997 Q1 36383.        1.11
##  10 1997 Q2 36714.        1.11
## # i 106 more rows
```

## 5 - Plots for Preliminary Analysis with Box-Cox Transformation

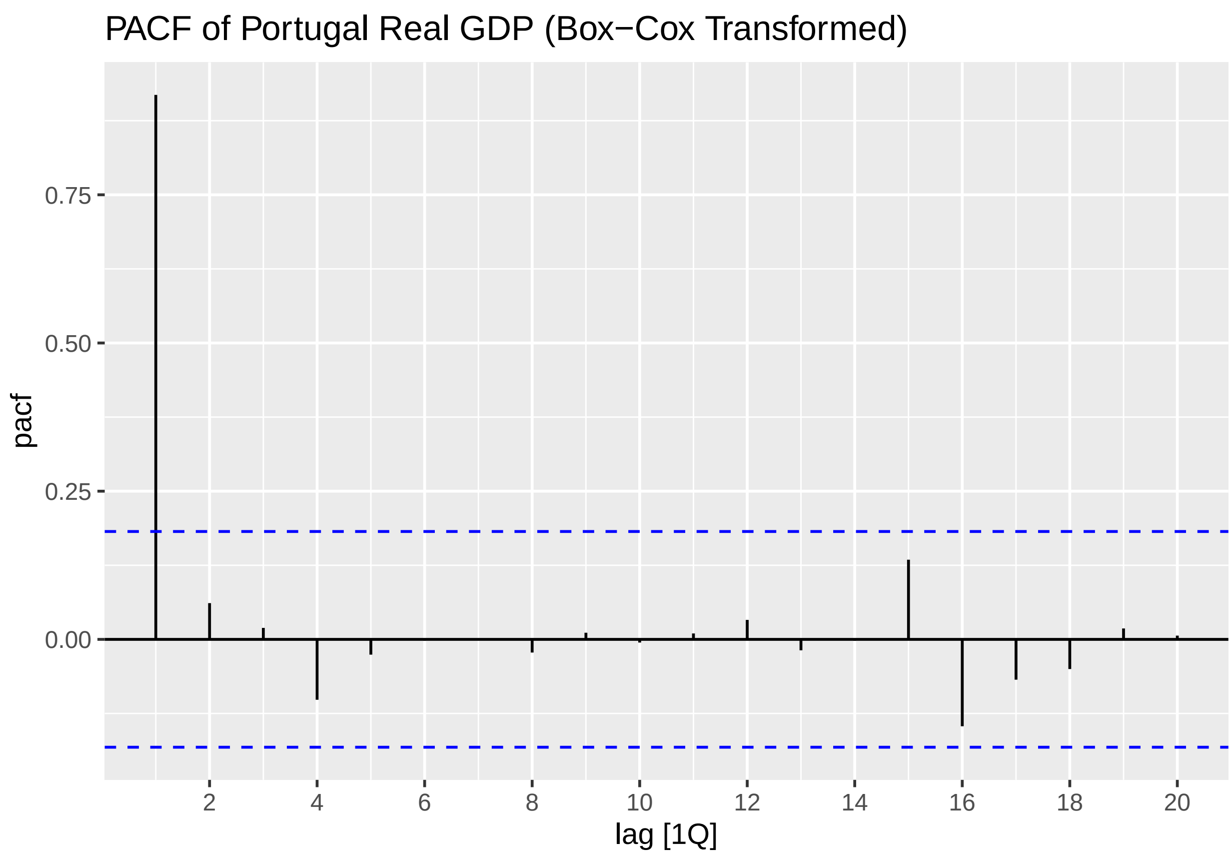
```r
# Plot 1: Original time series (Box-Cox Transformed)
autoplot(pt, gdp_boxcox) + ggtitle("Portugal Real GDP Time Series (Box-Cox Transformed)")
```



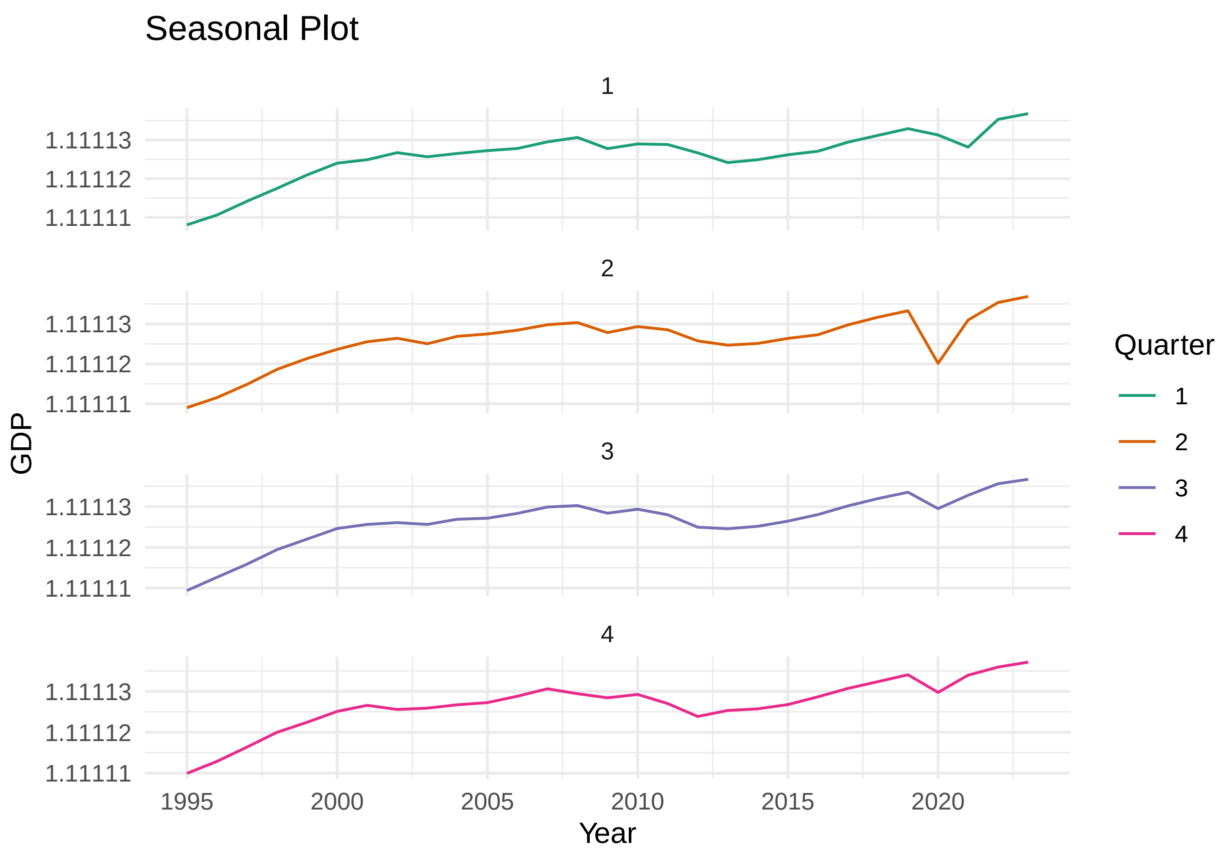

```r
# Plot 2: ACF of the original data (Box-Cox Transformed)
pt %>% ACF(gdp_boxcox) %>% autoplot() + ggtitle("ACF of Portugal Real GDP (Box-Cox Transformed)")
```



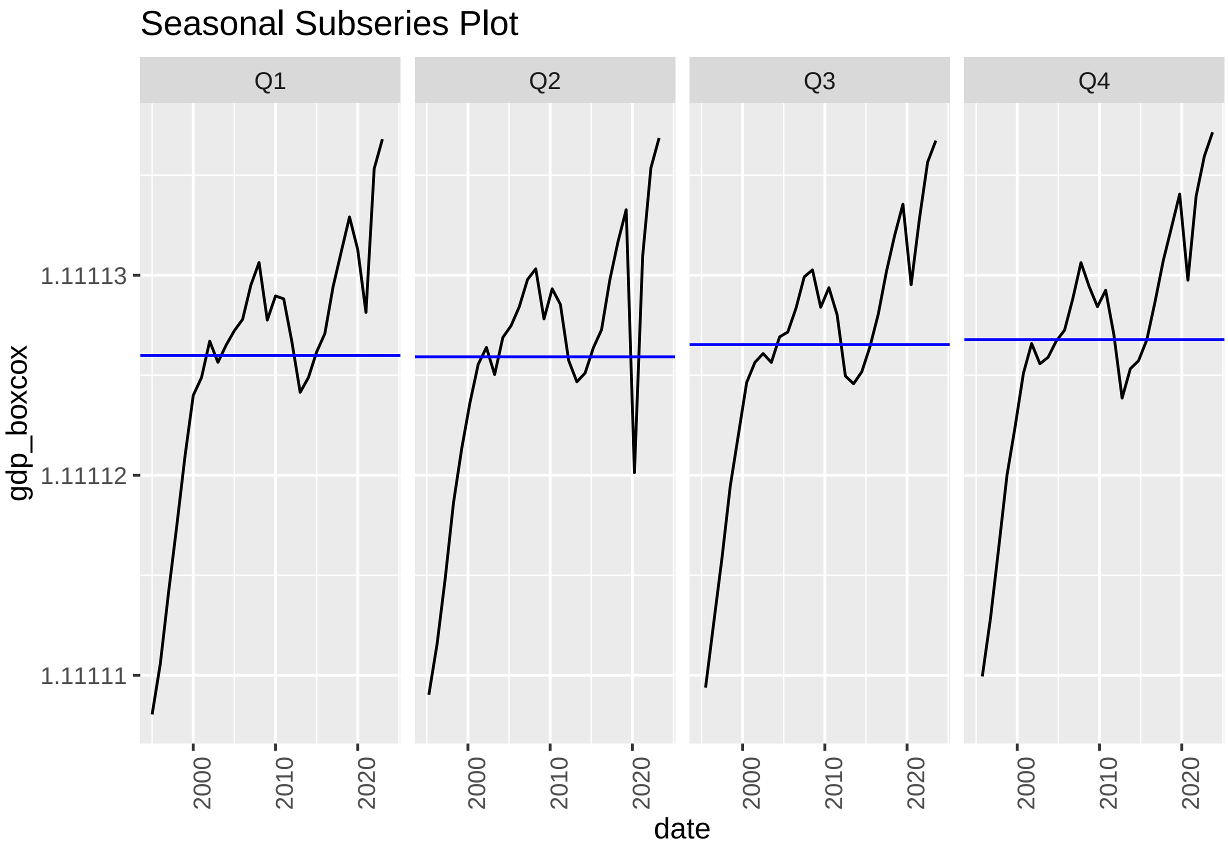

```r
# Plot 3: PACF of the original data (Box-Cox Transformed)
pt %>% PACF(gdp_boxcox) %>% autoplot() + ggtitle("PACF of Portugal Real GDP (Box-Cox Transformed)")
```




```r
# Plot 4: Seasonal plot (Box-Cox Transformed)
pt %>%
  mutate(quarter = quarter(date)) %>%
  ggplot(aes(x = year(date), y = gdp_boxcox, color = factor(quarter))) +
  geom_line() +
  labs(title = "Seasonal Plot", x = "Year", y = "GDP", color = "Quarter") +
  theme_minimal() +
  scale_color_brewer(palette = "Dark2") +
  facet_wrap(~ quarter, scales = "free_y", ncol = 1)
```

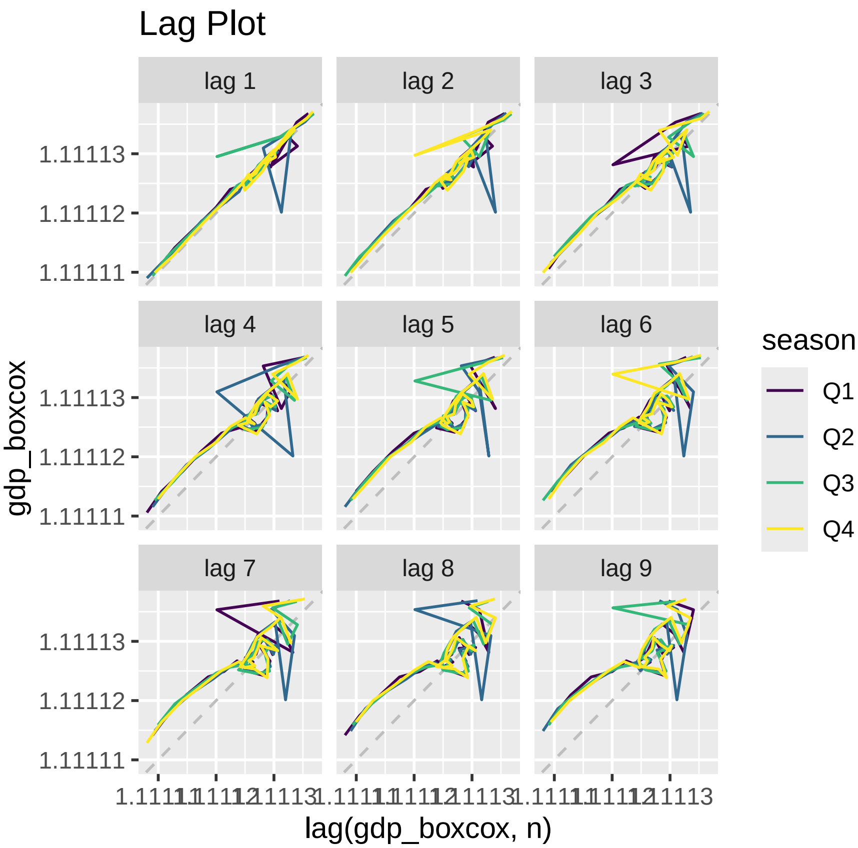



```r
# Plot 5: Seasonal Subseries Plot (Box-Cox Transformed)
gg_subseries(pt, y = gdp_boxcox) +
  ggtitle("Seasonal Subseries Plot")
```

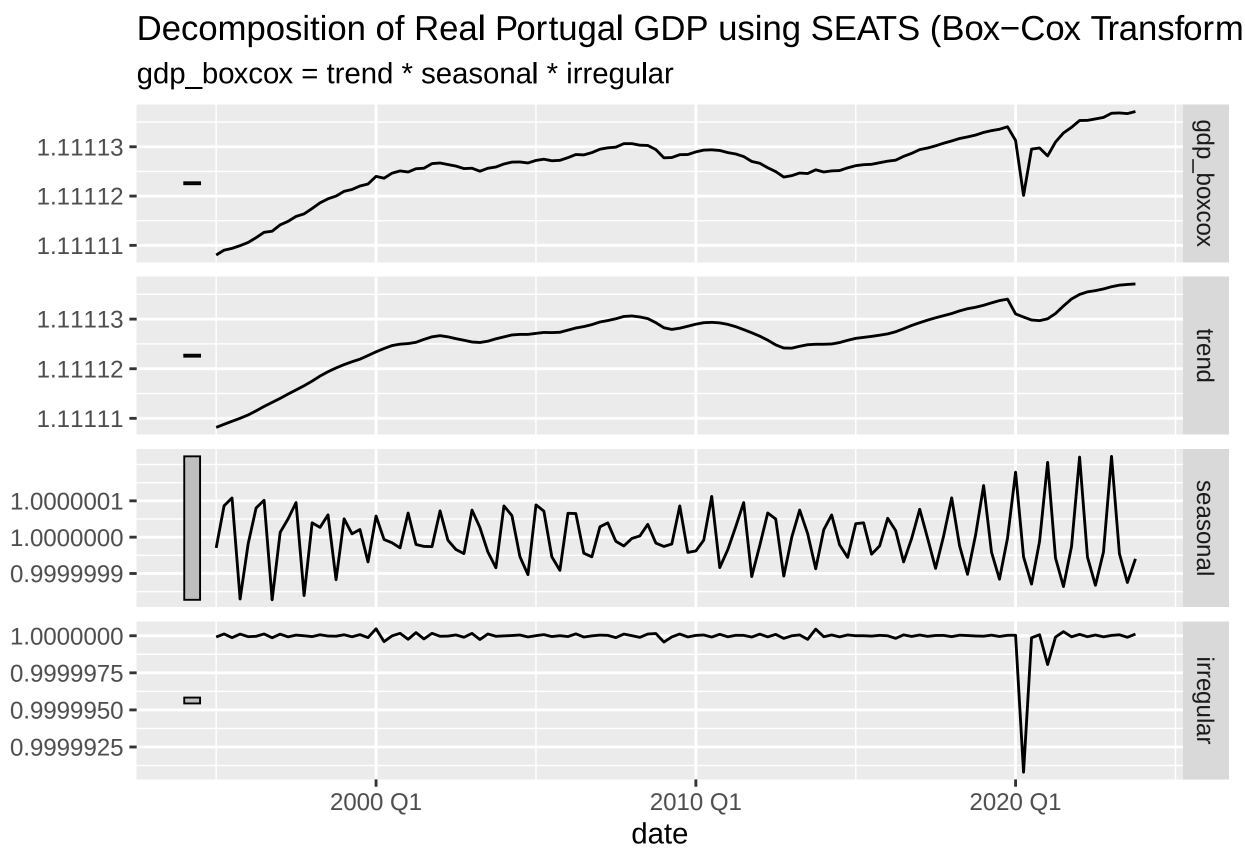



```
# Plot 6: Lag plot of the original data (Box-Cox Transformed)
gg_lag(pt, gdp_boxcox, do.lines = FALSE) +
  ggtitle("Lag Plot")
```

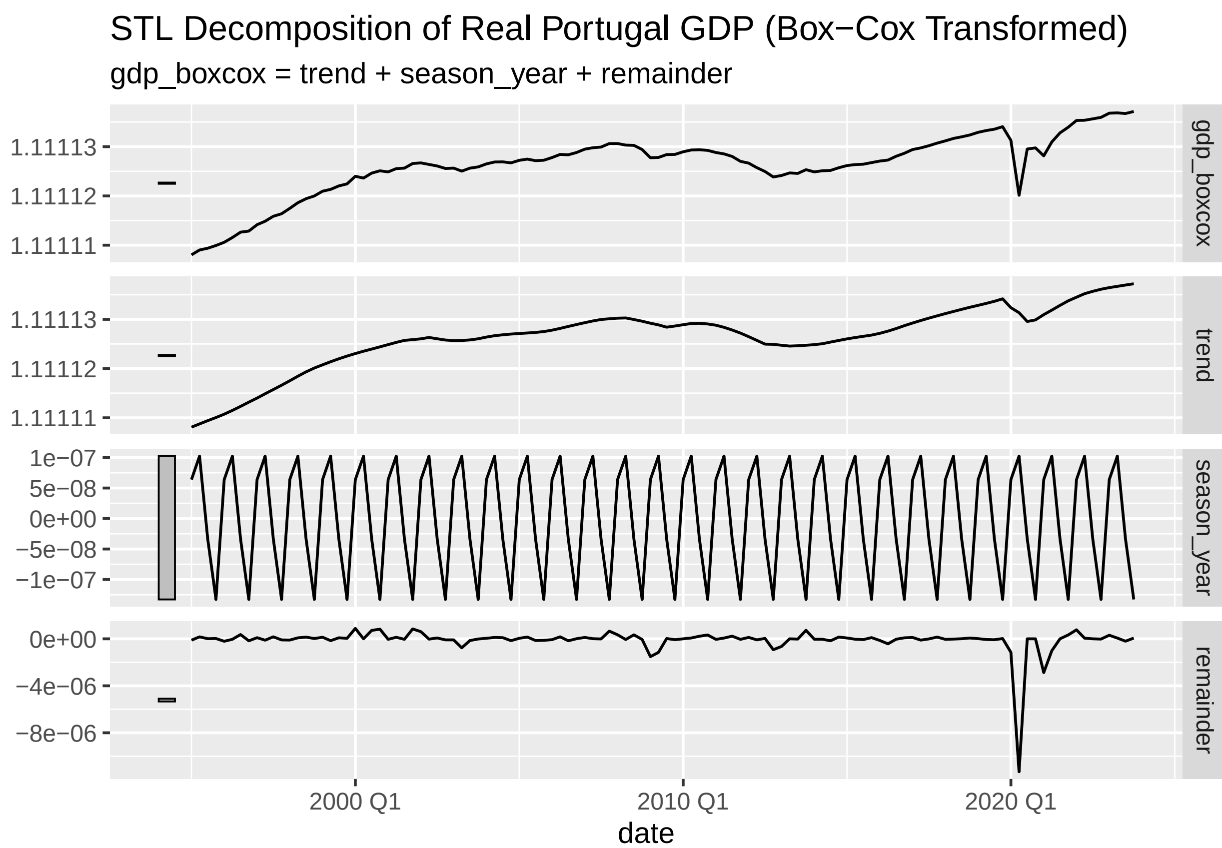
```
## Warning in lag_geom(..., arrow = arrow): Ignoring unknown parameters:
## ‘do.lines‘
```




```
# Plot 7: SEATS Decomposition (Box-Cox Transformed)
seats_dcmp <- pt %>%
  model(seats = X_13ARIMA_SEATS(gdp_boxcox ~ x11())) %>%
  components()
autoplot(seats_dcmp) +
  labs(title = "Decomposition of Real Portugal GDP using SEATS (Box-Cox Transformed)")
```




```r
# Plot 8: STL Decomposition (Box-Cox Transformed)
stl_dcmp <- pt %>%
  model(
    STL(gdp_boxcox ~ trend(window = 7) +
          season(window = "periodic"),
        robust = TRUE)
  ) %>%
  components()

autoplot(stl_dcmp) +
  labs(title = "STL Decomposition of Real Portugal GDP (Box-Cox Transformed)")
```




## 6 - Unit Root Tests on Box-Cox Transformed Data

```r
# Test 1: Augmented Dickey-Fuller test with a trend
summary(ur.df(as.ts(pt$gdp_boxcox), type = 'trend', lag = 24, selectlags = 'AIC'))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.183e-05 -3.402e-07  2.370e-07  4.729e-07  5.326e-06
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.544e-01  8.107e-02   1.904   0.0603 .
```

```
## z.lag.1     -1.389e-01  7.296e-02  -1.904   0.0603 .
## tt           1.418e-08  8.154e-09   1.738   0.0857 .
## z.diff.lag1 -1.677e-01  1.191e-01  -1.408   0.1628
## z.diff.lag2 -1.175e-01  1.157e-01  -1.015   0.3128
## z.diff.lag3  1.674e-01  1.096e-01   1.528   0.1303
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.595e-06 on 85 degrees of freedom
## Multiple R-squared:  0.1584, Adjusted R-squared:  0.1089
## F-statistic: 3.199 on 5 and 85 DF,  p-value: 0.01076
##
##
## Value of test-statistic is: -1.9043 1.7441 2.0671
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.99 -3.43 -3.13
## phi2  6.22  4.75  4.07
## phi3  8.43  6.49  5.47
```

```r
# Test 2: KPSS test with a trend
summary(ur.kpss(as.ts(pt$gdp_boxcox), type = 'tau'))
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: tau with 4 lags.
##
## Value of test-statistic is: 0.3061
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.119 0.146  0.176 0.216
```

```r
# Test 3: Augmented Dickey-Fuller test with a drift
summary(ur.df(as.ts(pt$gdp_boxcox), type = 'drift', lag = 24, selectlags = 'AIC'))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
```

```
## -1.187e-05 -2.952e-07  1.402e-07  5.139e-07  5.352e-06
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.04482    0.06243   0.718   0.4747
## z.lag.1     -0.04034    0.05618  -0.718   0.4747
## z.diff.lag1 -0.26308    0.11198  -2.349   0.0211 *
## z.diff.lag2 -0.20505    0.10850  -1.890   0.0621 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.62e-06 on 87 degrees of freedom
## Multiple R-squared:  0.1108, Adjusted R-squared:  0.08012
## F-statistic: 3.613 on 3 and 87 DF,  p-value: 0.01638
##
##
## Value of test-statistic is: -0.718 0.9586
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.46 -2.88 -2.57
## phi1  6.52  4.63  3.81
```

```
# Test 4: KPSS test with a level
summary(ur.kpss(as.ts(pt$gdp_boxcox), type = 'mu'))
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 4 lags.
##
## Value of test-statistic is: 1.5521
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

```
# Test 5: Augmented Dickey-Fuller test with none
summary(ur.df(as.ts(pt$gdp_boxcox), type = 'none', lag = 24, selectlags = 'AIC'))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
```

```
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.205e-05 -2.872e-07  1.894e-07  4.800e-07  5.298e-06
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## z.lag.1      1.824e-07  1.537e-07   1.187  0.23835
## z.diff.lag1 -2.927e-01  1.038e-01  -2.820  0.00594 **
## z.diff.lag2 -2.270e-01  1.038e-01  -2.187  0.03139 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.616e-06 on 88 degrees of freedom
## Multiple R-squared:  0.1112, Adjusted R-squared:  0.08094
## F-statistic: 3.671 on 3 and 88 DF,  p-value: 0.01521
##
##
## Value of test-statistic is: 1.1872
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

```r
#Not Stationary
```

# 7 - Perform Differencing on the Box-Cox Transformed GDP Data

```r
pt <- pt %>%
  mutate(d_gdp_boxcox = difference(gdp_boxcox)) %>%
  filter(!is.na(d_gdp_boxcox))

print(pt)
```

```
## # A tsibble: 115 x 4 [1Q]
##       date    gdp gdp_boxcox d_gdp_boxcox
##      <qtr>  <dbl>      <dbl>        <dbl>
##  1 1995 Q2 34145.       1.11  0.000000981
##  2 1995 Q3 34294.       1.11  0.000000360
##  3 1995 Q4 34526.       1.11  0.000000558
##  4 1996 Q1 34799.       1.11  0.000000646
##  5 1996 Q2 35216        1.11  0.000000971
##  6 1996 Q3 35694.       1.11  0.00000108
##  7 1996 Q4 35795.       1.11  0.000000225
##  8 1997 Q1 36383.       1.11  0.00000129
##  9 1997 Q2 36714.       1.11  0.000000708
## 10 1997 Q3 37192.       1.11  0.00000100
## # i 105 more rows
```

# 8 - Unit Root Tests on Differenced Box-Cox Transformed Data

```r
# Test 1: Augmented Dickey-Fuller test with a trend
summary(ur.df(as.ts(pt$d_gdp_boxcox), type = 'trend', lag = 24, selectlags = 'AIC'))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.223e-05 -1.775e-07  1.912e-07  4.310e-07  5.050e-06
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.908e-07  4.901e-07  -0.389   0.6980
## z.lag.1     -1.532e+00  1.651e-01  -9.276 1.35e-14 ***
## tt           5.605e-09  6.626e-09   0.846   0.3999
## z.diff.lag   2.337e-01  1.048e-01   2.231   0.0283 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.627e-06 on 86 degrees of freedom
## Multiple R-squared:  0.6419, Adjusted R-squared:  0.6294
## F-statistic: 51.38 on 3 and 86 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -9.2765 28.6855 43.0282
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.99 -3.43 -3.13
## phi2  6.22  4.75  4.07
## phi3  8.43  6.49  5.47
```

```r
# Test 2: KPSS test with a trend
summary(ur.kpss(as.ts(pt$d_gdp_boxcox), type = 'tau'))
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: tau with 4 lags.
##
```

```
## Value of test-statistic is: 0.1397
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.119 0.146  0.176 0.216
```

```r
# Test 3: Augmented Dickey-Fuller test with a drift
summary(ur.df(as.ts(pt$d_gdp_boxcox), type = 'drift', lag = 24, selectlags = 'AIC'))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.205e-05 -2.912e-07  1.942e-07  4.788e-07  5.309e-06
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.972e-07  1.726e-07   1.142   0.2564
## z.lag.1     -1.520e+00  1.642e-01  -9.253 1.36e-14 ***
## z.diff.lag   2.275e-01  1.044e-01   2.180   0.0319 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.624e-06 on 87 degrees of freedom
## Multiple R-squared:  0.6389, Adjusted R-squared:  0.6306
## F-statistic: 76.96 on 2 and 87 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -9.2531 42.8103
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.46 -2.88 -2.57
## phi1  6.52  4.63  3.81
```

```r
# Test 4: KPSS test with a level
summary(ur.kpss(as.ts(pt$d_gdp_boxcox), type = 'mu'))
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 4 lags.
```

```
##
## Value of test-statistic is: 0.2253
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

```r
# Test 5: Augmented Dickey-Fuller test with none
summary(ur.df(as.ts(pt$d_gdp_boxcox), type = 'none', lag = 24, selectlags = 'AIC'))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.182e-05 -1.008e-07  3.881e-07  6.660e-07  5.669e-06
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## z.lag.1     -1.4962     0.1632  -9.166 1.86e-14 ***
## z.diff.lag   0.2158     0.1040   2.075   0.0409 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.627e-06 on 88 degrees of freedom
## Multiple R-squared:  0.6335, Adjusted R-squared:  0.6251
## F-statistic: 76.04 on 2 and 88 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -9.1665
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

```r
#Stationary -> We can proceed
```

# 9 - Seasonality Analysis

```r
# STL Decomposition on Box-Cox transformed and differenced data
stl_decomp <- pt %>% model(STL(d_gdp_boxcox ~ season(window = "periodic")))
components <- components(stl_decomp)
```

```r
# Extract seasonal and remainder components
S_t <- components$season_year
R_t <- components$remainder

# Calculate the variance of seasonal and remainder components
var_Rt <- var(R_t, na.rm = TRUE)
var_St_Rt <- var(S_t + R_t, na.rm = TRUE)

# Calculate seasonal strength F_s
F_s <- max(0, 1 - var_Rt / var_St_Rt)

#Print
print(F_s)
```

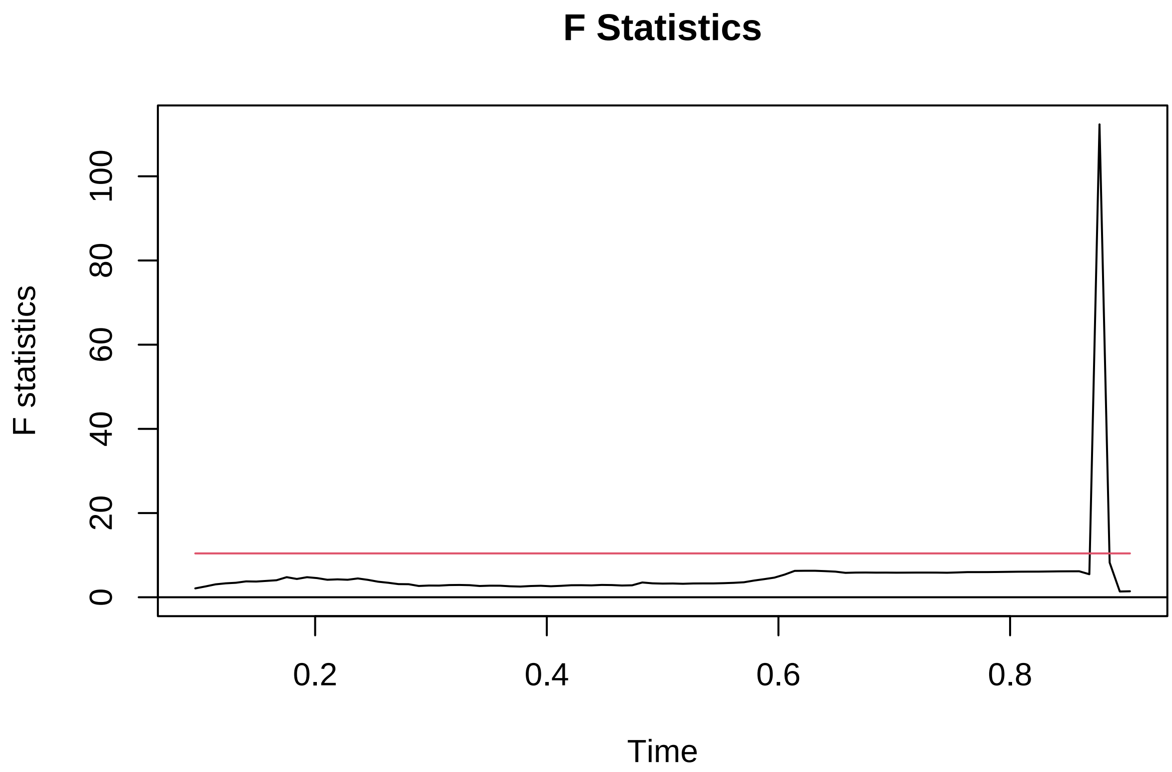
```
## [1] 0.02987415
```

```r
#Not necessary to seasonally difference
```

# 10 - Structural Break Test

## 10.1 - QLR Test (Chow Test)

```r
# Prepare the lagged values
pt.lag <- cbind(
  Lag0 = pt %>% select(d_gdp_boxcox) %>% filter(!is.na(d_gdp_boxcox)) %>% as.ts(),
  Lag1 = stats::lag(pt %>% select(d_gdp_boxcox) %>% filter(!is.na(d_gdp_boxcox)) %>% as.ts())
)

qlr <- Fstats(Lag0 ~ 1 + Lag1, data = pt.lag, from = 0.10)
plot(qlr, alpha = 0.1, main = "F Statistics")
```




```r
test <- sctest(qlr, type = "supF")
print(test)
```

```
##
##  supF test
##
## data:  qlr
## sup.F = 112.34, p-value < 2.2e-16
```

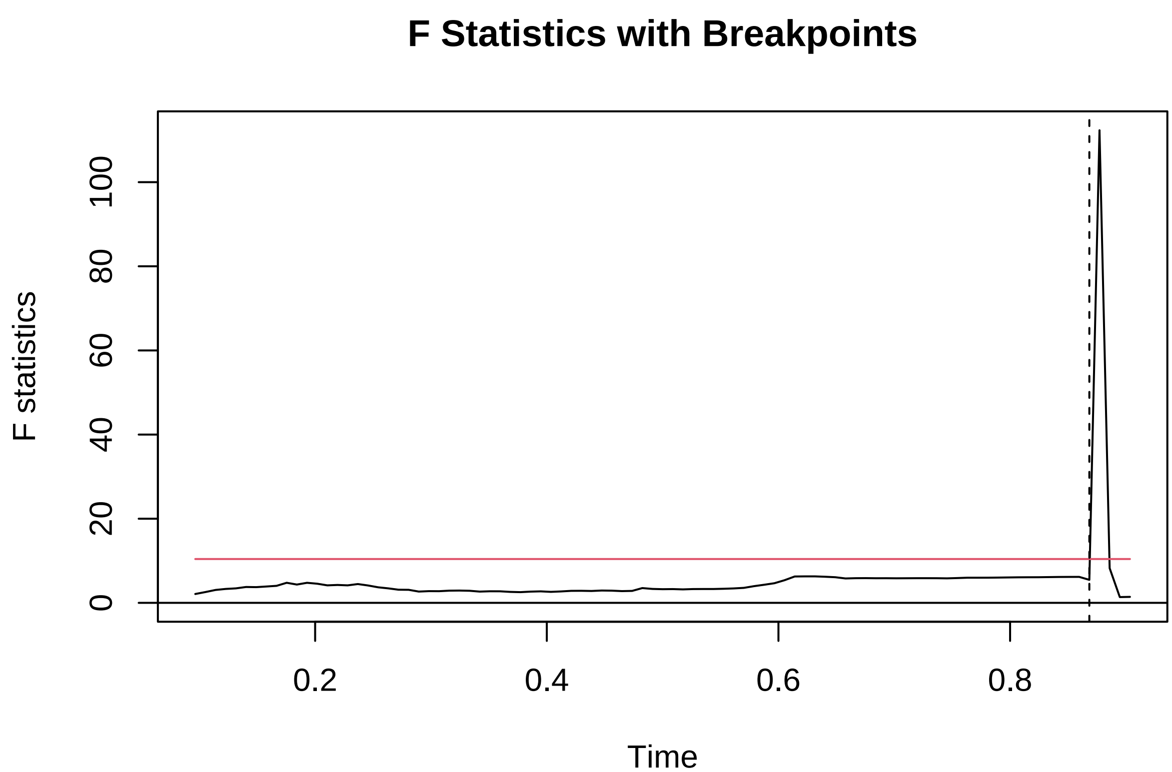
```r
breaks <- breakpoints(qlr, alpha = 0.01)
print(breaks)
```

```
##
##   Optimal 2-segment partition:
##
## Call:
## breakpoints.Fstats(obj = qlr, alpha = 0.01)
##
## Breakpoints at observation number:
## 100
##
## Corresponding to breakdates:
## 0.8684211
```

```r
breakpoint_dates <- pt$date[breaks$breakpoints]
print(breakpoint_dates)
```

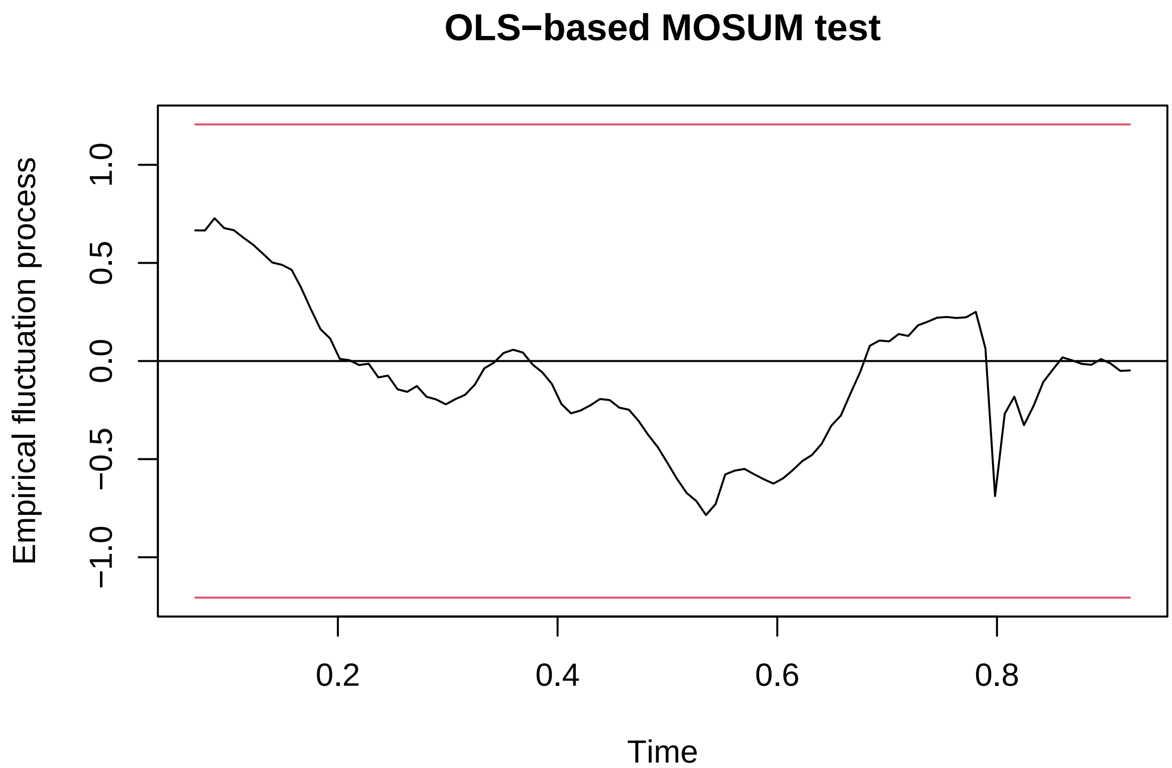
```
## <yearquarter[1]>
## [1] "2020 Q1"
## # Year starts on: January
```

```r
plot(qlr, alpha = 0.1, main = "F Statistics with Breakpoints")
lines(breakpoints(qlr))
```



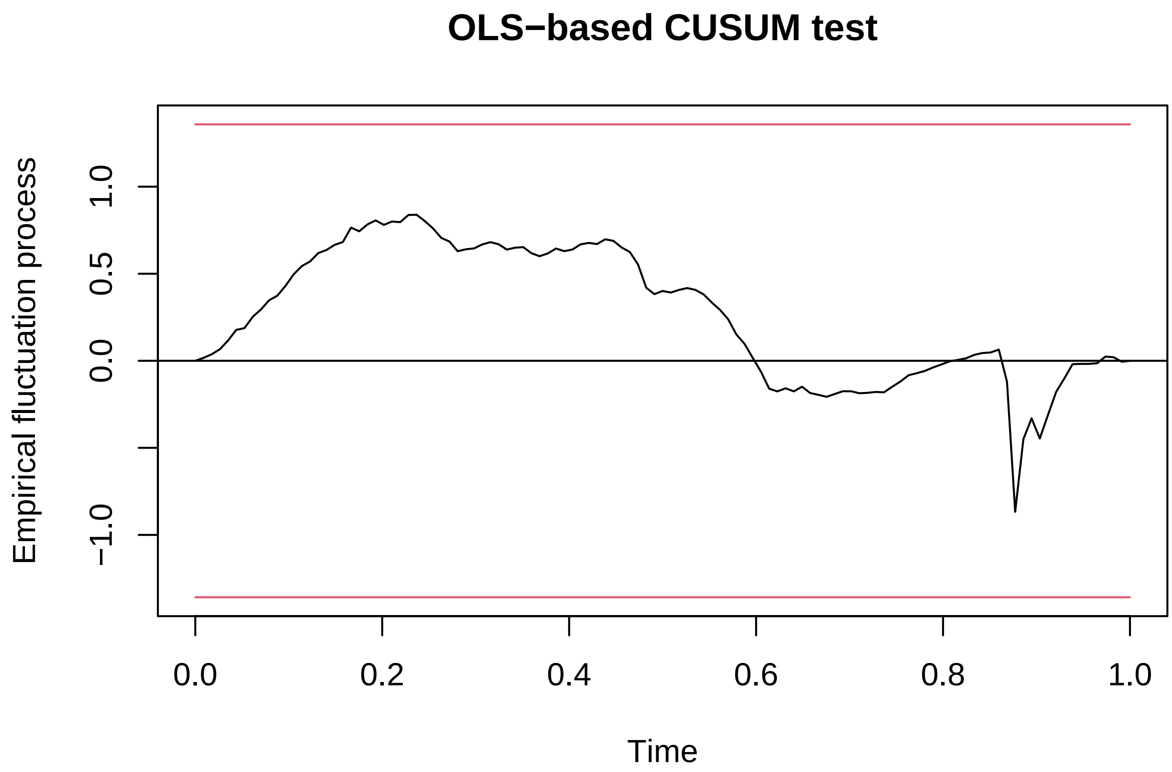

### 10.2 - OLS-MOSUM Test

```r
# Create a data frame with d_gdp_boxcox and its lag
pt.df <- pt %>%
  mutate(Lag1 = lag(d_gdp_boxcox)) %>%
  filter(!is.na(Lag1)) %>%
  as.data.frame()

mosum_test <- efp(d_gdp_boxcox ~ Lag1, type = "OLS-MOSUM", data = pt.df)
plot(mosum_test, main = "OLS-based MOSUM test")
```




### 10.3 - OLS-CUSUM Test

```
cusum_test <- efp(d_gdp_boxcox ~ Lag1, type = "OLS-CUSUM", data = pt.df)
plot(cusum_test, main = "OLS-based CUSUM test")
```




### 10.4 - SIS

```
sis <- isat(pt.df$d_gdp_boxcox, t.pval = 0.1)
```

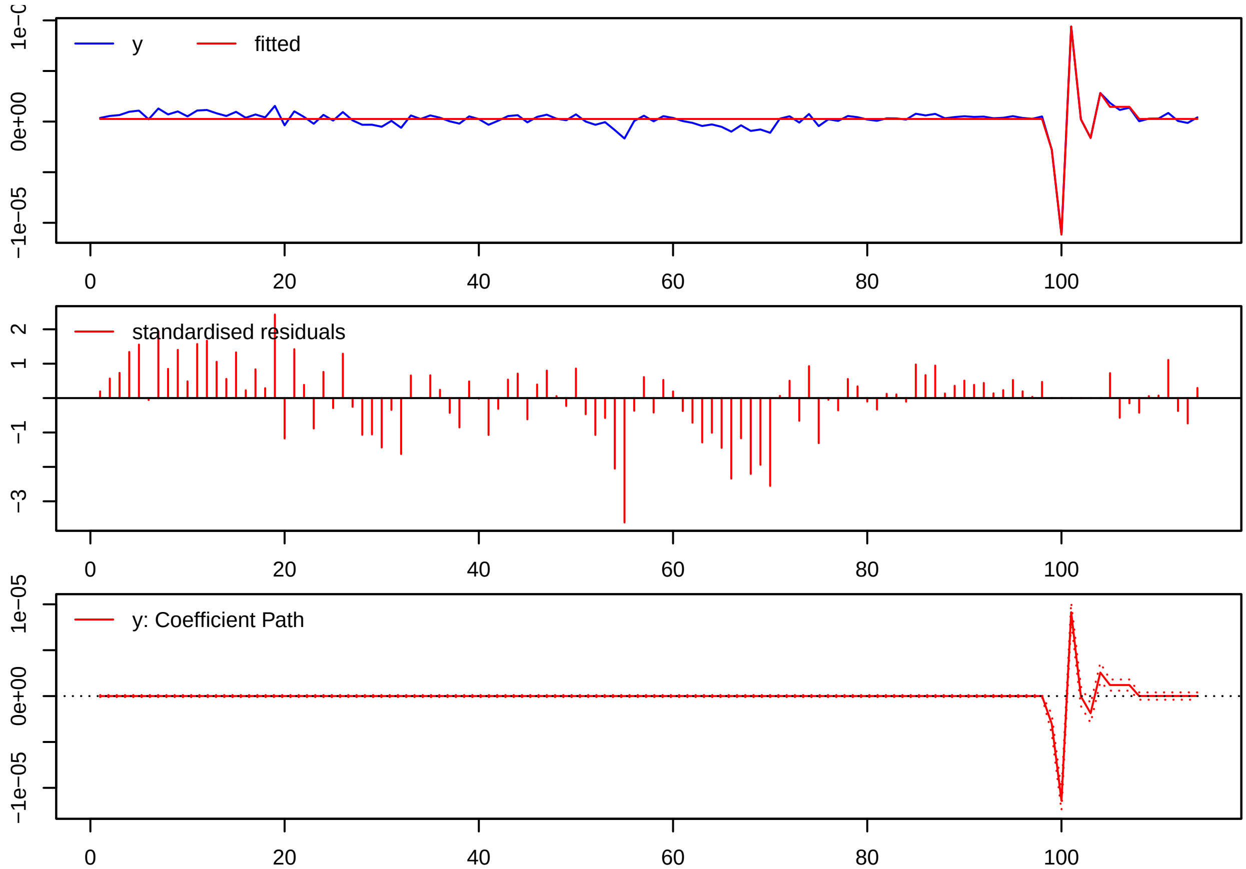
```
##

## SIS block 1 of 4:

## 29 path(s) to search

## Searching: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
##
## SIS block 2 of 4:
## 29 path(s) to search
## Searching: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
##
## SIS block 3 of 4:
## 29 path(s) to search
## Searching: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
##
## SIS block 4 of 4:
## 20 path(s) to search
## Searching: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

```
##
## GETS of union of retained SIS variables...
##
## GETS of union of ALL retained variables...
```

```r
print(sis)
```

```
##
## Date: Wed Aug 14 11:35:08 2024
## Dependent var.: y
## Method: Ordinary Least Squares (OLS)
## Variance-Covariance: Ordinary
## No. of observations (mean eq.): 114
## Sample: 1 to 114
##
## SPECIFIC mean equation:
##
##                coef   std.error   t-stat   p-value
## mconst  2.5543e-07  5.3853e-08   4.7432 6.657e-06 ***
## sis99  -3.0297e-06  5.3583e-07  -5.6543 1.359e-07 ***
## sis100 -8.3803e-06  7.5394e-07 -11.1155 < 2.2e-16 ***
## sis101  2.0551e-05  7.5394e-07  27.2577 < 2.2e-16 ***
## sis102 -9.1694e-06  7.5394e-07 -12.1620 < 2.2e-16 ***
## sis103 -1.8368e-06  7.5394e-07  -2.4362  0.016524 *
## sis104  4.4302e-06  7.5394e-07   5.8761 5.007e-08 ***
## sis105 -1.3644e-06  6.1559e-07  -2.2164  0.028818 *
## sis108 -1.1952e-06  3.6788e-07  -3.2489  0.001556 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostics and fit:
##
##                    Chi-sq df   p-value
## Ljung-Box AR(1)    19.765  1 8.757e-06 ***
## Ljung-Box ARCH(1)   6.659  1  0.009866 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## SE of regression    0.00000
## R-squared           0.88829
## Log-lik.(n=114)  1489.41777
```

```r
plot(sis, main = "SIS without additional regressors (2)")
```




## 11 - Create Train and Test Sets

```r
# Identify the break date
break_date <- yearquarter("2020 Q1")

# Split the data
train <- pt %>% filter(date < break_date)
test <- pt %>% filter(date >= break_date)

# Verify splits
print(train)
```

```
## # A tsibble: 99 x 4 [1Q]
##       date    gdp gdp_boxcox d_gdp_boxcox
##      <qtr>  <dbl>      <dbl>        <dbl>
##  1 1995 Q2 34145.       1.11  0.000000981
##  2 1995 Q3 34294.       1.11  0.000000360
##  3 1995 Q4 34526.       1.11  0.000000558
##  4 1996 Q1 34799.       1.11  0.000000646
##  5 1996 Q2 35216        1.11  0.000000971
##  6 1996 Q3 35694.       1.11  0.00000108
##  7 1996 Q4 35795.       1.11  0.000000225
##  8 1997 Q1 36383.       1.11  0.00000129
```

```
## 9 1997 Q2 36714.       1.11  0.000000708
## 10 1997 Q3 37192.       1.11  0.00000100
## # i 89 more rows
```

```r
print(test)
```

```
## # A tsibble: 16 x 4 [1Q]
##        date    gdp gdp_boxcox  d_gdp_boxcox
##       <qtr>  <dbl>      <dbl>         <dbl>
##  1 2020 Q1 46364.       1.11 -0.00000277
##  2 2020 Q2 39359.       1.11 -0.0000112
##  3 2020 Q3 45107.       1.11  0.00000940
##  4 2020 Q4 45265        1.11  0.000000226
##  5 2021 Q1 44163.       1.11 -0.00000161
##  6 2021 Q2 46128.       1.11  0.00000282
##  7 2021 Q3 47504.       1.11  0.00000184
##  8 2021 Q4 48403.       1.11  0.00000115
##  9 2022 Q1 49522.       1.11  0.00000137
## 10 2022 Q2 49550.       1.11  0.0000000339
## 11 2022 Q3 49793.       1.11  0.000000291
## 12 2022 Q4 50046.       1.11  0.000000300
## 13 2023 Q1 50779.       1.11  0.000000852
## 14 2023 Q2 50832.       1.11  0.0000000603
## 15 2023 Q3 50716.       1.11 -0.000000133
## 16 2023 Q4 51082.       1.11  0.000000417
```

# 12 - Unit Root Tests on Training Set

```r
# Test 1: Augmented Dickey-Fuller test with a trend
summary(ur.df(as.ts(train$d_gdp_boxcox), type = 'trend', lag = 24, selectlags = 'AIC'))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.403e-06 -2.526e-07  1.278e-08  2.760e-07  8.362e-07
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -9.830e-08  1.599e-07  -0.615 0.540784
## z.lag.1     -4.836e-01  1.252e-01  -3.863 0.000247 ***
```

```
## tt            2.502e-09  2.484e-09   1.007 0.317174
## z.diff.lag  -1.572e-01  1.157e-01  -1.358 0.178763
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.504e-07 on 70 degrees of freedom
## Multiple R-squared:  0.3126, Adjusted R-squared:  0.2831
## F-statistic: 10.61 on 3 and 70 DF,  p-value: 7.805e-06
##
##
## Value of test-statistic is: -3.8629 5.0244 7.5365
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -4.04 -3.45 -3.15
## phi2  6.50  4.88  4.16
## phi3  8.73  6.49  5.47
```

```
# Test 2: KPSS test with a trend
summary(ur.kpss(as.ts(train$d_gdp_boxcox), type = 'tau'))
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: tau with 3 lags.
##
## Value of test-statistic is: 0.3163
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.119 0.146  0.176 0.216
```

```
# Test 3: Augmented Dickey-Fuller test with a drift
summary(ur.df(as.ts(train$d_gdp_boxcox), type = 'drift', lag = 24, selectlags = 'AIC'))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.408e-06 -2.912e-07  3.383e-08  2.898e-07  7.865e-07
##
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.328e-08  5.422e-08   0.983 0.329082
## z.lag.1     -4.632e-01  1.236e-01  -3.749 0.000359 ***
## z.diff.lag  -1.681e-01  1.153e-01  -1.458 0.149131
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.504e-07 on 71 degrees of freedom
## Multiple R-squared:  0.3026, Adjusted R-squared:  0.283
## F-statistic:  15.4 on 2 and 71 DF,  p-value: 2.777e-06
##
##
## Value of test-statistic is: -3.749 7.0275
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.51 -2.89 -2.58
## phi1  6.70  4.71  3.86
```

```r
# Test 4: KPSS test with a level
summary(ur.kpss(as.ts(train$d_gdp_boxcox), type = 'mu'))
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 3 lags.
##
## Value of test-statistic is: 0.6272
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

```r
# Test 5: Augmented Dickey-Fuller test with none
summary(ur.df(as.ts(train$d_gdp_boxcox), type = 'none', lag = 24, selectlags = 'AIC'))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.340e-06 -2.548e-07  8.415e-08  3.328e-07  8.631e-07
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## z.lag.1     -0.4317     0.1193  -3.619 0.000547 ***
## z.diff.lag  -0.1826     0.1143  -1.598 0.114363
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.503e-07 on 72 degrees of freedom
## Multiple R-squared:  0.2931, Adjusted R-squared:  0.2735
## F-statistic: 14.93 on 2 and 72 DF,  p-value: 3.77e-06
##
##
## Value of test-statistic is: -3.6188
##
## Critical values for test statistics:
##      1pct  5pct 10pct
## tau1 -2.6 -1.95 -1.61
```

```
#Not Stationary
```

# 13 - Apply Second Differencing On Training Set

```
# Apply second differencing on the training set
train <- train %>%
  mutate(dd_gdp_boxcox = difference(d_gdp_boxcox)) %>%
  filter(!is.na(dd_gdp_boxcox))

# Print the train dataset to verify the changes
print(train)
```

```
## # A tsibble: 98 x 5 [1Q]
##       date    gdp gdp_boxcox d_gdp_boxcox dd_gdp_boxcox
##      <qtr>  <dbl>      <dbl>        <dbl>         <dbl>
##  1 1995 Q3 34294.       1.11  0.000000360 -0.000000622
##  2 1995 Q4 34526.       1.11  0.000000558  0.000000199
##  3 1996 Q1 34799.       1.11  0.000000646  0.0000000878
##  4 1996 Q2 35216        1.11  0.000000971  0.000000324
##  5 1996 Q3 35694.       1.11  0.00000108   0.000000114
##  6 1996 Q4 35795.       1.11  0.000000225 -0.000000859
##  7 1997 Q1 36383.       1.11  0.00000129   0.00000107
##  8 1997 Q2 36714.       1.11  0.000000708 -0.000000584
##  9 1997 Q3 37192.       1.11  0.00000100   0.000000294
## 10 1997 Q4 37442.       1.11  0.000000515 -0.000000487
## # i 88 more rows
```

## 14 - Unit Root Tests on Second Differenced Box-Cox Transformed Data

```r
# Test 1: Augmented Dickey-Fuller Test with a Trend
adf_trend <- summary(ur.df(as.ts(train$dd_gdp_boxcox), type = 'trend', lag = 24, selectlags = 'AIC'))

# Test 2: KPSS Test with a Trend
kpss_trend <- summary(ur.kpss(as.ts(train$dd_gdp_boxcox), type = 'tau'))

# Test 3: Augmented Dickey-Fuller Test with a Drift
adf_drift <- summary(ur.df(as.ts(train$dd_gdp_boxcox), type = 'drift', lag = 24, selectlags = 'AIC'))

# Test 4: KPSS Test with a Level
kpss_level <- summary(ur.kpss(as.ts(train$dd_gdp_boxcox), type = 'mu'))

# Test 5: Augmented Dickey-Fuller Test with None
adf_none <- summary(ur.df(as.ts(train$dd_gdp_boxcox), type = 'none', lag = 24, selectlags = 'AIC'))

# Print results
print(adf_trend)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.236e-06 -2.381e-07 -3.239e-08  3.058e-07  1.079e-06
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.070e-07  1.691e-07  -0.633  0.52928
## z.lag.1     -2.775e+00  4.749e-01  -5.843  1.8e-07 ***
## tt           1.783e-09  2.621e-09   0.680  0.49867
## z.diff.lag1  1.253e+00  4.288e-01   2.922  0.00478 **
## z.diff.lag2  9.276e-01  3.636e-01   2.551  0.01310 *
## z.diff.lag3  7.548e-01  2.912e-01   2.592  0.01178 *
## z.diff.lag4  5.681e-01  2.040e-01   2.785  0.00701 **
## z.diff.lag5  3.132e-01  1.054e-01   2.970  0.00417 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.672e-07 on 65 degrees of freedom
## Multiple R-squared:  0.751,  Adjusted R-squared:  0.7242
## F-statistic: 28.01 on 7 and 65 DF,  p-value: < 2.2e-16
```

```
##
##
## Value of test-statistic is: -5.8432 11.3839 17.0736
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -4.04 -3.45 -3.15
## phi2  6.50  4.88  4.16
## phi3  8.73  6.49  5.47
```

```r
print(kpss_trend)
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: tau with 3 lags.
##
## Value of test-statistic is: 0.0209
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.119 0.146  0.176 0.216
```

```r
print(adf_drift)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.240e-06 -2.297e-07 -2.967e-08  2.974e-07  1.099e-06
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.910e-09  5.449e-08   0.035  0.97214
## z.lag.1     -2.734e+00  4.692e-01  -5.828 1.84e-07 ***
## z.diff.lag1  1.216e+00  4.236e-01   2.871  0.00550 **
## z.diff.lag2  8.962e-01  3.592e-01   2.495  0.01511 *
## z.diff.lag3  7.313e-01  2.880e-01   2.539  0.01347 *
## z.diff.lag4  5.543e-01  2.022e-01   2.742  0.00785 **
## z.diff.lag5  3.062e-01  1.045e-01   2.930  0.00465 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 4.653e-07 on 66 degrees of freedom
## Multiple R-squared:  0.7493, Adjusted R-squared:  0.7265
## F-statistic: 32.87 on 6 and 66 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -5.8276 16.9826
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.51 -2.89 -2.58
## phi1  6.70  4.71  3.86
```

```r
print(kpss_level)
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 3 lags.
##
## Value of test-statistic is: 0.0497
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

```r
print(adf_none)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.238e-06 -2.278e-07 -2.776e-08  2.992e-07  1.101e-06
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## z.lag.1      -2.7341     0.4656  -5.872 1.48e-07 ***
## z.diff.lag1   1.2162     0.4204   2.893  0.00515 **
## z.diff.lag2   0.8963     0.3565   2.514  0.01434 *
## z.diff.lag3   0.7315     0.2858   2.559  0.01275 *
## z.diff.lag4   0.5544     0.2006   2.763  0.00738 **
## z.diff.lag5   0.3063     0.1037   2.953  0.00433 **
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.618e-07 on 67 degrees of freedom
## Multiple R-squared:  0.7493, Adjusted R-squared:  0.7268
## F-statistic: 33.37 on 6 and 67 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -5.8718
##
## Critical values for test statistics:
##      1pct  5pct 10pct
## tau1 -2.6 -1.95 -1.61
```

```
#Stationary -> We can proceed
```

# 15 - Model Selection ARIMA

## 15.1 - Auto ARIMA Drift

```
# tsibble format
pt <- pt %>%
  as_tsibble(index = date)

auto_arima_model_drift <- train %>%
  model(auto_arima = ARIMA(gdp ~ 1))

# Report the model
report(auto_arima_model_drift)
```

```
## Series: gdp
## Model: ARIMA(1,1,1) w/ drift
##
## Coefficients:
##          ar1      ma1  constant
##       0.8375  -0.5351   25.8207
## s.e.  0.0965   0.1487   13.2117
##
## sigma^2 estimated as 86351:  log likelihood=-687.54
## AIC=1383.09   AICc=1383.52   BIC=1393.39
```

```
# Print the report
print(auto_arima_model_drift)
```

```
## # A mable: 1 x 1
##                auto_arima
##                   <model>
## 1 <ARIMA(1,1,1) w/ drift>
```

```r
#output: ARIMA(1,1,1)
```

### 15.2 - Auto ARIMA with Drift

```r
# Auto ARIMA model with drift
auto_arima_model_drift <- train %>%
  model(auto_arima = ARIMA(gdp ~ 1))

# Report the model
report(auto_arima_model_drift)
```

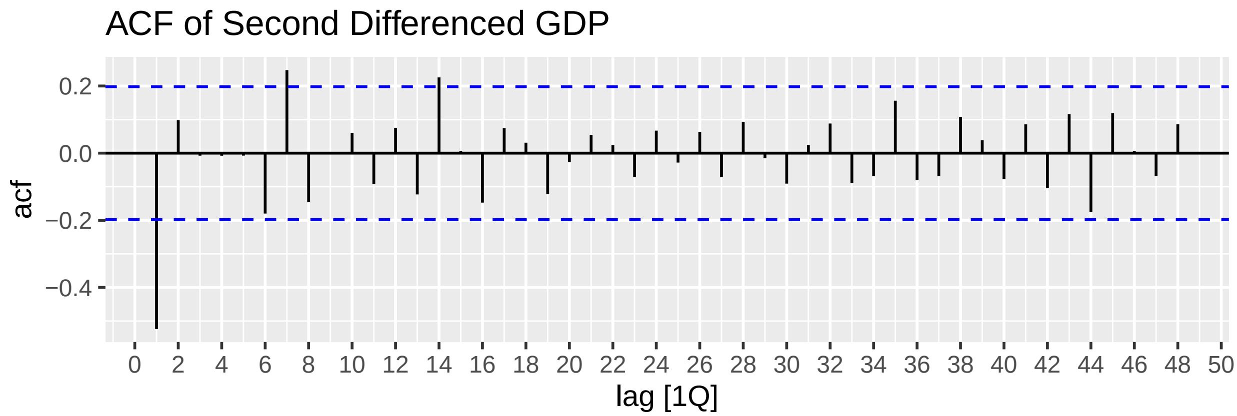
```
## Series: gdp
## Model: ARIMA(1,1,1) w/ drift
##
## Coefficients:
##          ar1      ma1  constant
##       0.8375  -0.5351   25.8207
## s.e.  0.0965   0.1487   13.2117
##
## sigma^2 estimated as 86351:  log likelihood=-687.54
## AIC=1383.09   AICc=1383.52   BIC=1393.39
```

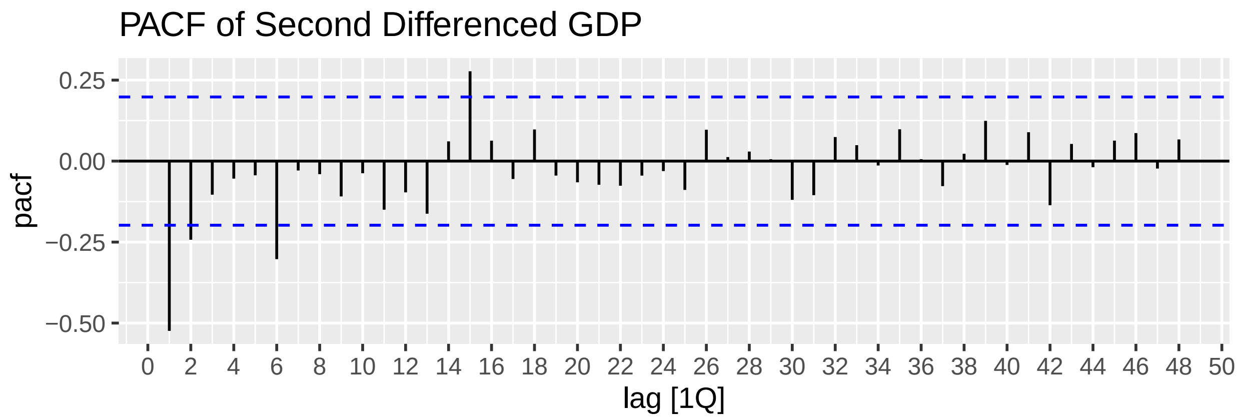
```r
print(auto_arima_model_drift)
```

```
## # A mable: 1 x 1
##                auto_arima
##                   <model>
## 1 <ARIMA(1,1,1) w/ drift>
```

## 15.3 - ACF and PACF

```r
ACF_fd <- train %>%
  ACF(dd_gdp_boxcox, lag_max = 48) %>%
  autoplot() + labs(title = "ACF of Second Differenced GDP")

PACF_fd <- train %>%
  PACF(dd_gdp_boxcox, lag_max = 48) %>%
  autoplot() + labs(title = "PACF of Second Differenced GDP")

# Plot ACF and PACF using grid.arrange
grid.arrange(ACF_fd, PACF_fd, ncol=1)
```




## 16 - ARIMA

```
modelsg <- train %>%
  model(
    auto_arima_model_drift = ARIMA(gdp ~ 1),
    guessed_arima_1 = ARIMA(gdp ~ pdq(2, 2, 1) + PDQ(1, 0, 1, 4)),
    guessed_arima_2 = ARIMA(gdp ~ pdq(1, 2, 1) + PDQ(1, 0, 1, 4))
  )


# Model Summary
model_summary <- glance(modelsg) %>%
  arrange(AICc) %>%
  select(.model, AIC, AICc, BIC) %>%
  mutate(across(AIC:BIC, ~format(round(.x, 2), nsmall = 2)))

print(model_summary)
```

```
## # A tibble: 3 x 4
##   .model                 AIC     AICc    BIC
##   <chr>                  <chr>   <chr>   <chr>
## 1 guessed_arima_2        1373.62 1374.28 1386.44
## 2 guessed_arima_1        1373.92 1374.87 1389.31
```

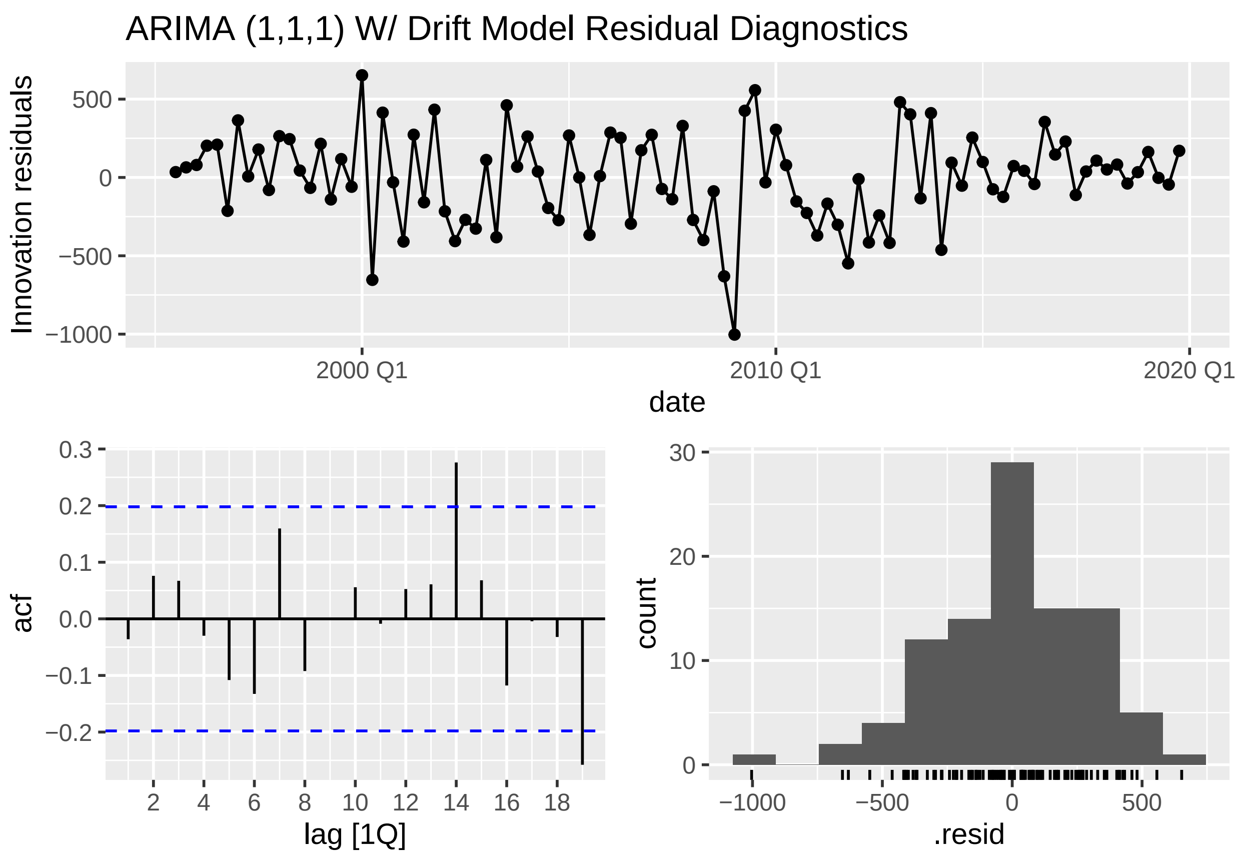
```
## 3 auto_arima_model_drift 1383.09 1383.52 1393.39
```

```r
# Accuracy
accuracy_metrics_arima <- bind_rows(
  train %>%
    model(auto_arima_model_drift = ARIMA(gdp ~ 1)) %>%
    forecast(test) %>%
    fabletools::accuracy(test %>% select(gdp)),

  train %>%
    model(guessed_arima_1 = ARIMA(gdp ~ pdq(2, 2, 1) + PDQ(1, 0, 1, 4))) %>%
    forecast(test) %>%
    fabletools::accuracy(test %>% select(gdp)),

  train %>%
    model(guessed_arima_2 = ARIMA(gdp ~ pdq(1, 2, 1) + PDQ(1, 0, 1, 4))) %>%
    forecast(test) %>%
    fabletools::accuracy(test %>% select(gdp))
) %>%
  select(.model, RMSE, MAE, MAPE) %>%
  mutate(across(RMSE:MAPE, ~format(round(.x, 2), nsmall = 2)))

print(accuracy_metrics_arima)
```

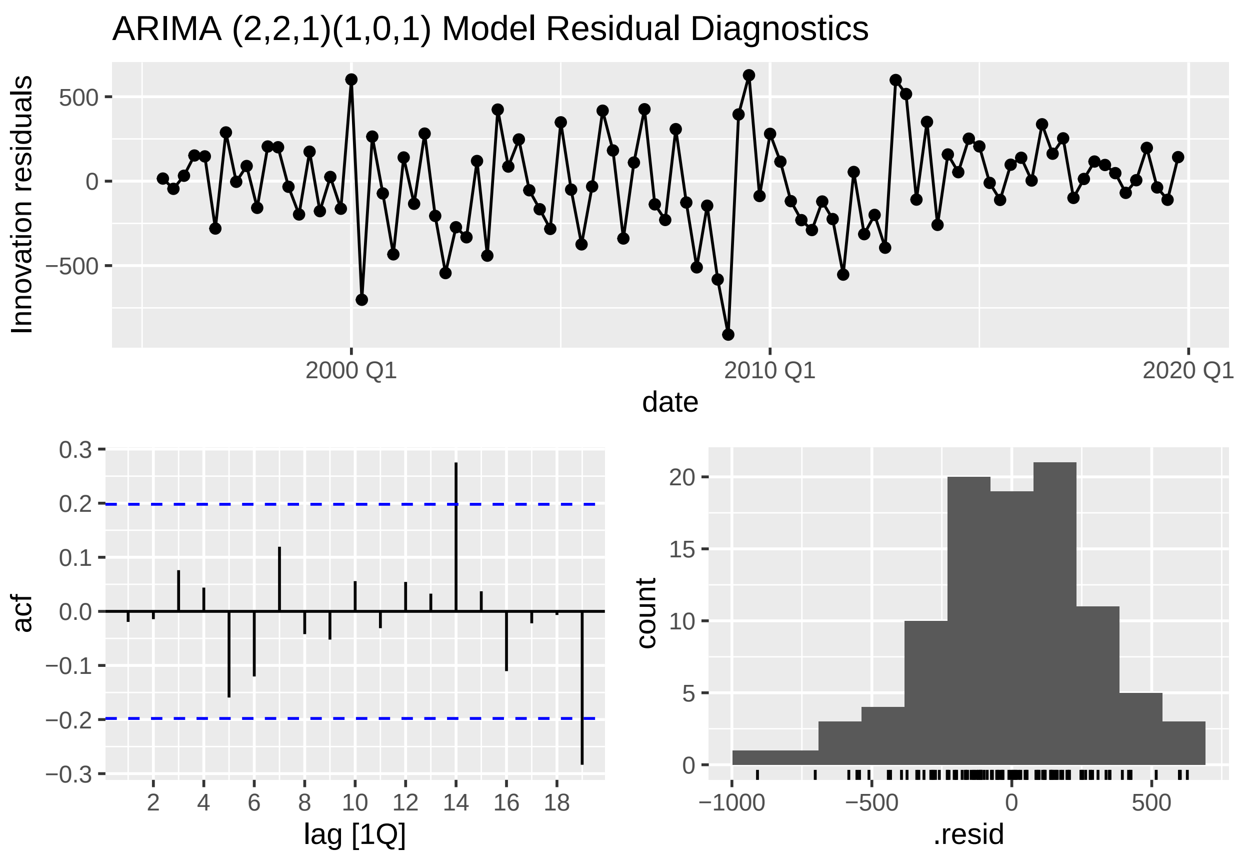
```
## # A tibble: 3 x 4
##   .model                 RMSE    MAE     MAPE
##   <chr>                  <chr>   <chr>   <chr>
## 1 auto_arima_model_drift 3482.22 2528.03 5.67
## 2 guessed_arima_1        3644.55 2913.46 6.45
## 3 guessed_arima_2        3692.08 2990.81 6.60
```

# 17 - Residuals ARIMA

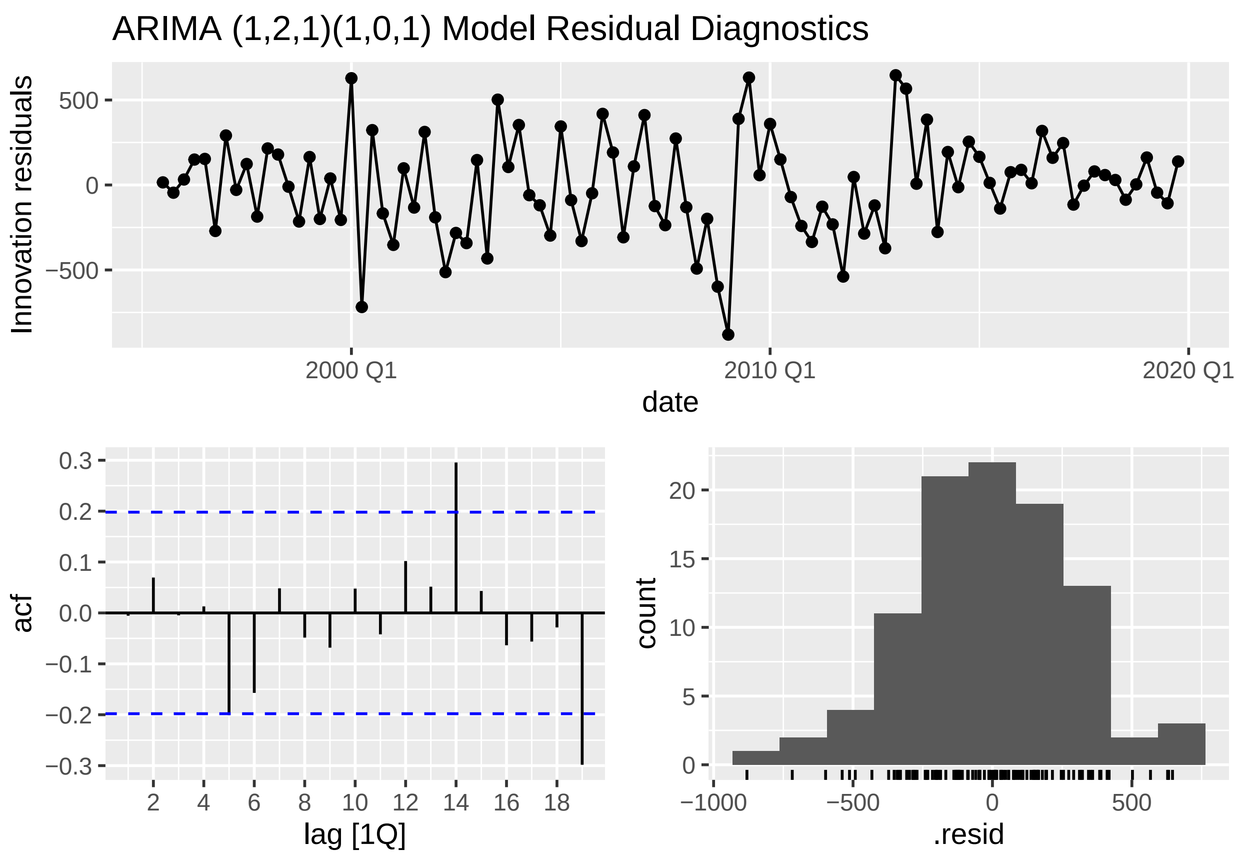
```r
# Residual diagnostics for auto ARIMA W/ Drift model
modelsg %>%
  select(auto_arima_model_drift) %>%
  gg_tsresiduals(type = "innovation") +
  ggtitle("ARIMA (1,1,1) W/ Drift Model Residual Diagnostics")
```




```r
# Residual diagnostics for Guessed ARIMA Model 1
modelsg %>%
  select(guessed_arima_1) %>%
  gg_tsresiduals(type = "innovation") +
  ggtitle("ARIMA (2,2,1)(1,0,1) Model Residual Diagnostics")
```




```
# Residual diagnostics for Guessed ARIMA Model 2
modelsg %>%
  select(guessed_arima_2) %>%
  gg_tsresiduals(type = "innovation") +
  ggtitle("ARIMA (1,2,1)(1,0,1) Model Residual Diagnostics")
```




## 18 - Ljung-Box Test ARIMA

```
# Ljung-Box test for auto ARIMA W/ Drift model
lb_test_auto_arima_drift <- augment(modelsg %>% select(auto_arima_model_drift)) %>%
  features(.resid, ljung_box, lag = 10, dof = 2)

# Ljung-Box test for Guessed ARIMA Model 1
lb_test_guessed_arima_1 <- augment(modelsg %>% select(guessed_arima_1)) %>%
  features(.resid, ljung_box, lag =10, dof = 2)

# Ljung-Box test for Guessed ARIMA Model 2
lb_test_guessed_arima_2 <- augment(modelsg %>% select(guessed_arima_2)) %>%
  features(.resid, ljung_box, lag = 10, dof = 2)

# Display Ljung-Box test results
print(lb_test_auto_arima_drift)
```

```
## # A tibble: 1 x 3
##   .model                   lb_stat lb_pvalue
##   <chr>                      <dbl>     <dbl>
## 1 auto_arima_model_drift     8.40      0.395
```

```r
print(lb_test_guessed_arima_1)
```

```
## # A tibble: 1 x 3
##   .model           lb_stat lb_pvalue
##   <chr>              <dbl>     <dbl>
## 1 guessed_arima_1     7.45     0.489
```

```r
print(lb_test_guessed_arima_2)
```

```
## # A tibble: 1 x 3
##   .model           lb_stat lb_pvalue
##   <chr>              <dbl>     <dbl>
## 1 guessed_arima_2     8.66     0.372
```

## 19 - Shapiro-Wilk Test ARIMA

```r
shapiro_test_auto_arima_drift <- shapiro.test(modelsg %>%
                                                select(auto_arima_model_drift) %>%
                                                residuals() %>%
                                                select(.resid) %>%
                                                as.ts())

shapiro_test_guessed_arima_1 <- shapiro.test(modelsg %>%
                                               select(guessed_arima_1) %>%
                                               residuals() %>%
                                               select(.resid) %>%
                                               as.ts())

shapiro_test_guessed_arima_2 <- shapiro.test(modelsg %>%
                                               select(guessed_arima_2) %>%
                                               residuals() %>%
                                               select(.resid) %>%
                                               as.ts())

# Print the results
print(shapiro_test_auto_arima_drift)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  modelsg %>% select(auto_arima_model_drift) %>% residuals() %>% select(.resid) %>% as.ts()
## W = 0.98539, p-value = 0.3525
```

```r
print(shapiro_test_guessed_arima_1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  modelsg %>% select(guessed_arima_1) %>% residuals() %>% select(.resid) %>% as.ts()
## W = 0.99114, p-value = 0.7673
```

```r
print(shapiro_test_guessed_arima_2)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  modelsg %>% select(guessed_arima_2) %>% residuals() %>% select(.resid) %>% as.ts()
## W = 0.99188, p-value = 0.8222
```

# 20 - Forecasting ARIMA

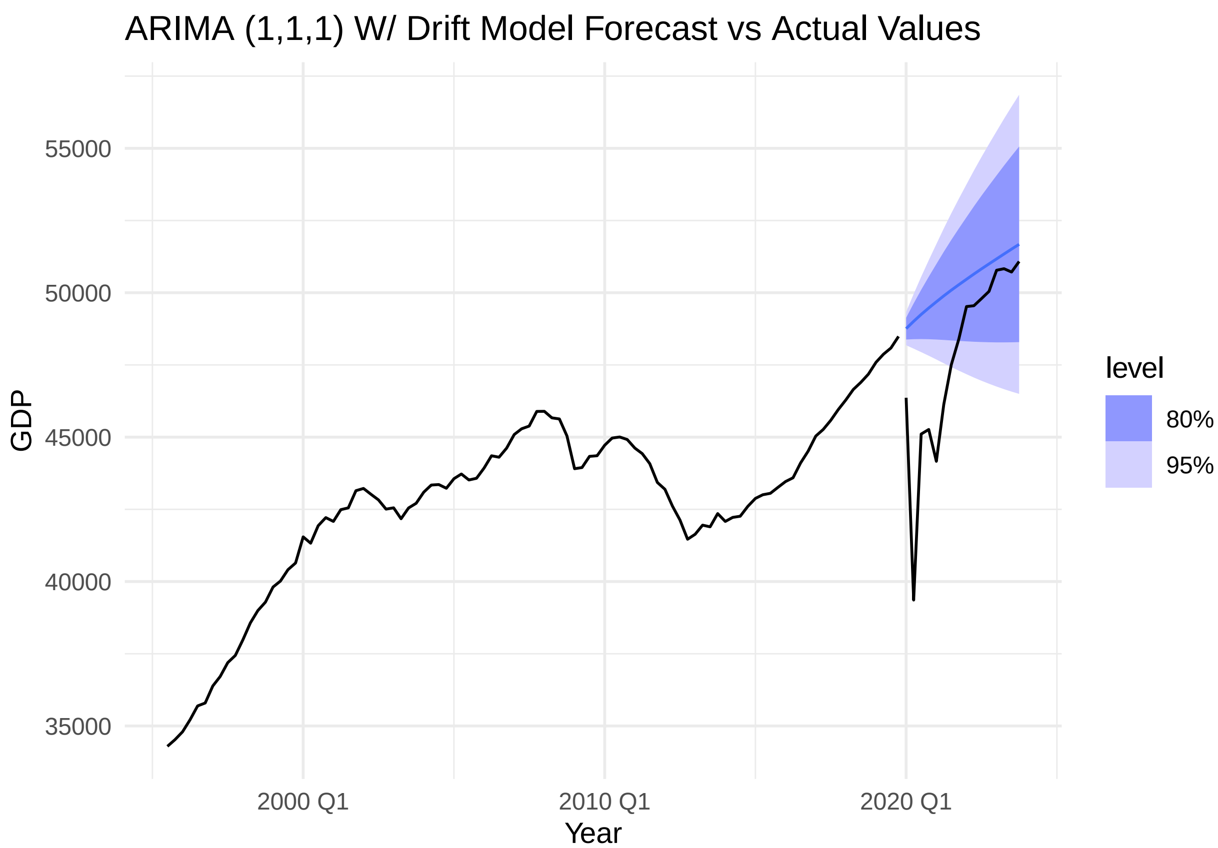
```r
auto_arima_drift_forecast <- modelsg %>%
  select(auto_arima_model_drift) %>%
  forecast(h = 16)

guessed_arima_1_forecast <- modelsg %>%
  select(guessed_arima_1) %>%
  forecast(h = 16)

guessed_arima_2_forecast <- modelsg %>%
  select(guessed_arima_2) %>%
  forecast(h = 16)


# Plotting the forecasts along with actual values in the test set
autoplot(train, gdp) +
  autolayer(auto_arima_drift_forecast, series = "Auto ARIMA Forecast") +
  autolayer(test, series = "Actual Values") +
  ggtitle("ARIMA (1,1,1) W/ Drift Model Forecast vs Actual Values") +
  labs(x = "Year", y = "GDP") +
  theme_minimal()
```

```
## Warning in ggdist::geom_lineribbon(without(intvl_mapping, "colour_ramp"), :
## Ignoring unknown parameters: `series`
```

```
## Warning in geom_line(mapping = without(mapping, "shape"), data =
## unpack_data(object[single_row[["FALSE"]], : Ignoring unknown parameters:
## `series`
```

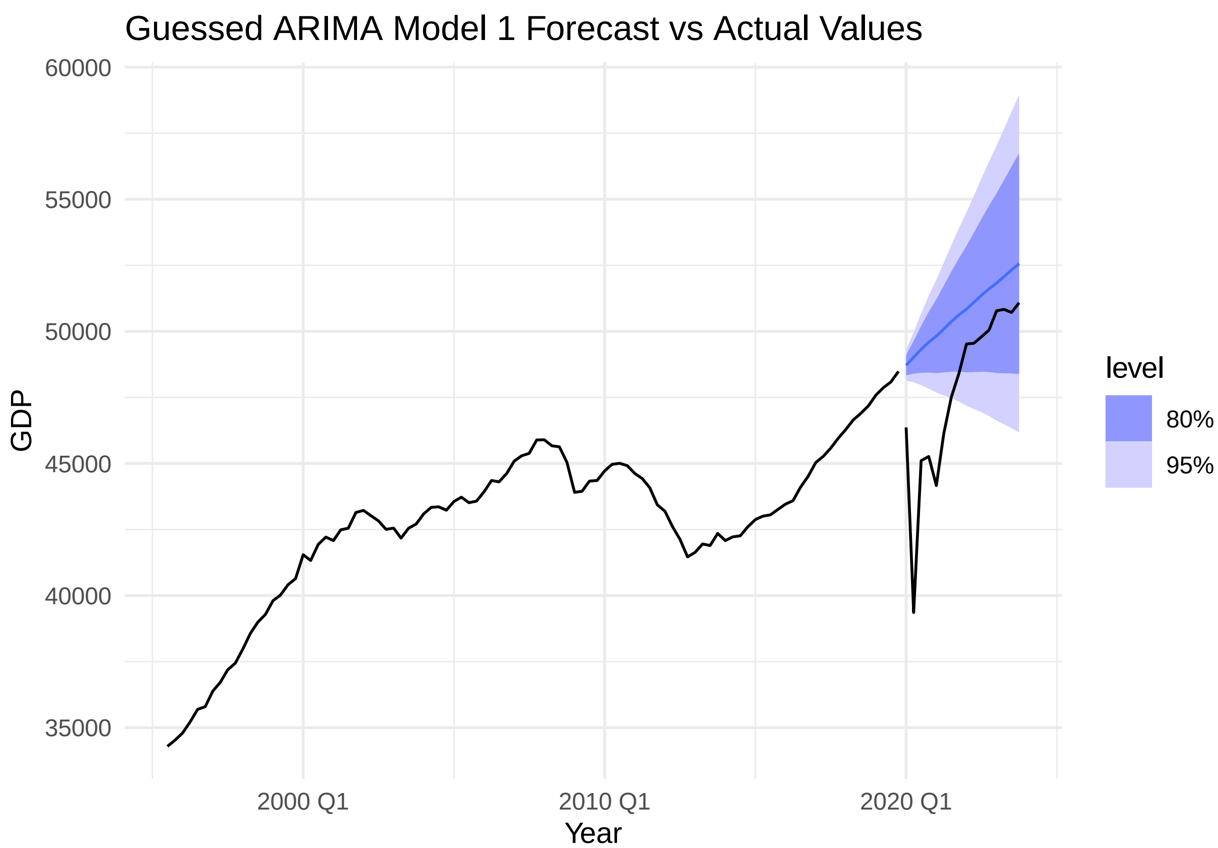
```
## Plot variable not specified, automatically selected `.vars = gdp`
```

```
## Warning in geom_line(eval_tidy(expr(aes(!!!aes_spec))), data = object, ..., :
## Ignoring unknown parameters: `series`
```



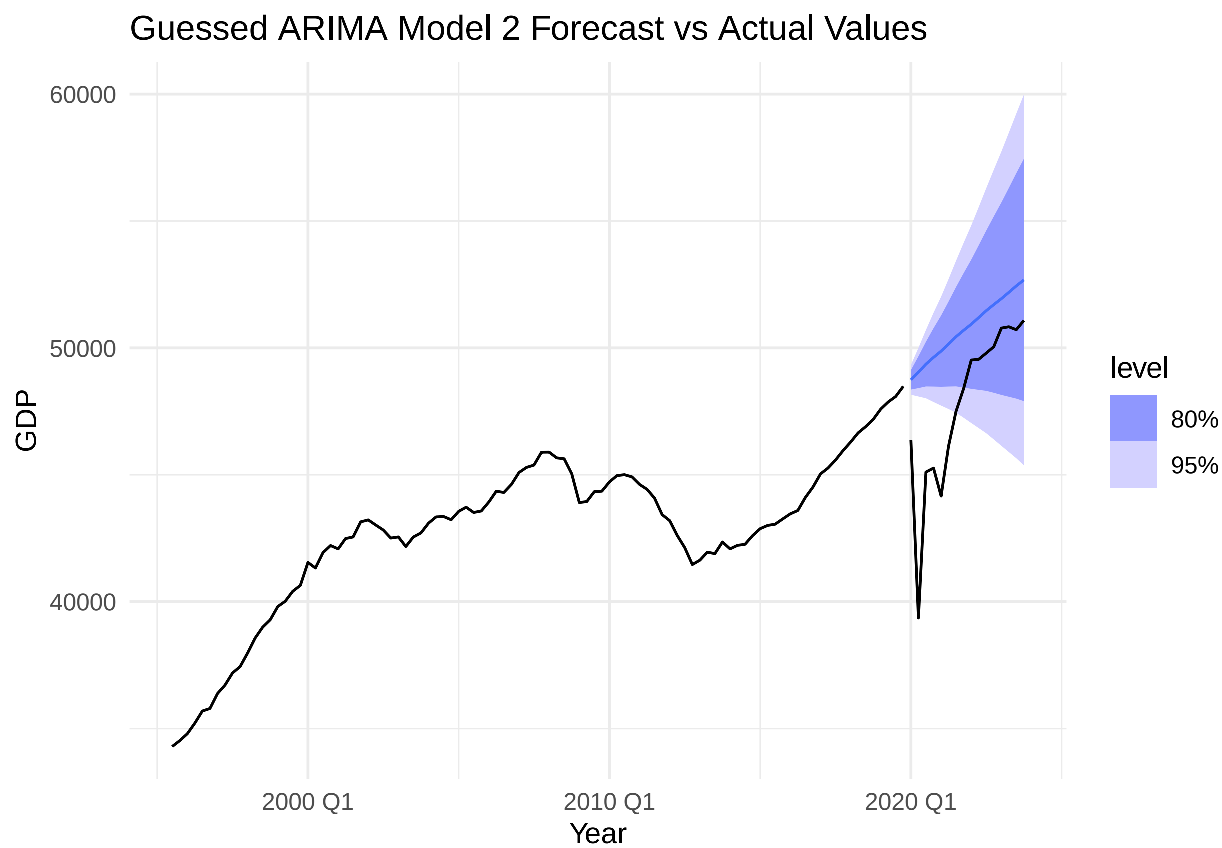

```r
autoplot(train, gdp) +
  autolayer(guessed_arima_1_forecast, series = "Guessed ARIMA 1 Forecast") +
  autolayer(test, series = "Actual Values") +
  ggtitle("Guessed ARIMA Model 1 Forecast vs Actual Values") +
  labs(x = "Year", y = "GDP") +
  theme_minimal()
```

```
## Warning in ggdist::geom_lineribbon(without(intvl_mapping, "colour_ramp"), : Ignoring unknown paramete
## Ignoring unknown parameters: ‘series‘

## Plot variable not specified, automatically selected ‘.vars = gdp‘

## Warning in geom_line(eval_tidy(expr(aes(!!!aes_spec))), data = object, ..., :
## Ignoring unknown parameters: ‘series‘
```




```r
autoplot(train, gdp) +
  autolayer(guessed_arima_2_forecast, series = "Guessed ARIMA 2 Forecast") +
  autolayer(test, series = "Actual Values") +
  ggtitle("Guessed ARIMA Model 2 Forecast vs Actual Values") +
  labs(x = "Year", y = "GDP") +
  theme_minimal()
```

```
## Warning in ggdist::geom_lineribbon(without(intvl_mapping, "colour_ramp"), : Ignoring unknown paramete
## Ignoring unknown parameters: `series`

## Plot variable not specified, automatically selected `.vars = gdp`

## Warning in geom_line(eval_tidy(expr(aes(!!!aes_spec))), data = object, ..., :
## Ignoring unknown parameters: `series`
```




```r
# Print the forecast values for each model
print(auto_arima_drift_forecast)
```

```
## # A fable: 16 x 4 [1Q]
## # Key:     .model [1]
##    .model                    date              gdp  .mean
##    <chr>                    <qtr>           <dist>  <dbl>
##  1 auto_arima_model_drift 2020 Q1   N(48758, 86351) 48758.
##  2 auto_arima_model_drift 2020 Q2  N(49010, 232818) 49010.
##  3 auto_arima_model_drift 2020 Q3  N(49247, 441781) 49247.
##  4 auto_arima_model_drift 2020 Q4  N(49472, 711604) 49472.
##  5 auto_arima_model_drift 2021 Q1   N(49685, 1e+06) 49685.
##  6 auto_arima_model_drift 2021 Q2 N(49890, 1417027) 49890.
##  7 auto_arima_model_drift 2021 Q3 N(50088, 1842073) 50088.
##  8 auto_arima_model_drift 2021 Q4 N(50279, 2308035) 50279.
##  9 auto_arima_model_drift 2022 Q1 N(50465, 2809708) 50465.
## 10 auto_arima_model_drift 2022 Q2 N(50646, 3342305) 50646.
## 11 auto_arima_model_drift 2022 Q3 N(50824, 3901510) 50824.
## 12 auto_arima_model_drift 2022 Q4 N(50999, 4483500) 50999.
## 13 auto_arima_model_drift 2023 Q1 N(51171, 5084920) 51171.
## 14 auto_arima_model_drift 2023 Q2 N(51341, 5702859) 51341.
## 15 auto_arima_model_drift 2023 Q3 N(51509, 6334805) 51509.
## 16 auto_arima_model_drift 2023 Q4   N(51676, 7e+06) 51676.
```

```r
print(guessed_arima_1_forecast)
```

```
## # A fable: 16 x 4 [1Q]
## # Key:     .model [1]
##    .model             date                gdp  .mean
##    <chr>             <qtr>             <dist>  <dbl>
##  1 guessed_arima_1 2020 Q1   N(48717, 87769) 48717.
##  2 guessed_arima_1 2020 Q2  N(49020, 234538) 49020.
##  3 guessed_arima_1 2020 Q3  N(49319, 479880) 49319.
##  4 guessed_arima_1 2020 Q4  N(49591, 807310) 49591.
##  5 guessed_arima_1 2021 Q1 N(49818, 1186042) 49818.
##  6 guessed_arima_1 2021 Q2 N(50086, 1642435) 50086.
##  7 guessed_arima_1 2021 Q3 N(50366, 2178044) 50366.
##  8 guessed_arima_1 2021 Q4 N(50621, 2803427) 50621.
##  9 guessed_arima_1 2022 Q1 N(50842, 3478953) 50842.
## 10 guessed_arima_1 2022 Q2 N(51097, 4240819) 51097.
## 11 guessed_arima_1 2022 Q3 N(51363, 5089844) 51363.
## 12 guessed_arima_1 2022 Q4   N(51608, 6e+06) 51608.
## 13 guessed_arima_1 2023 Q1   N(51825, 7e+06) 51825.
## 14 guessed_arima_1 2023 Q2 N(52070, 8142974) 52070.
## 15 guessed_arima_1 2023 Q3 N(52324, 9337447) 52324.
## 16 guessed_arima_1 2023 Q4 N(52561, 1.1e+07) 52561.
```

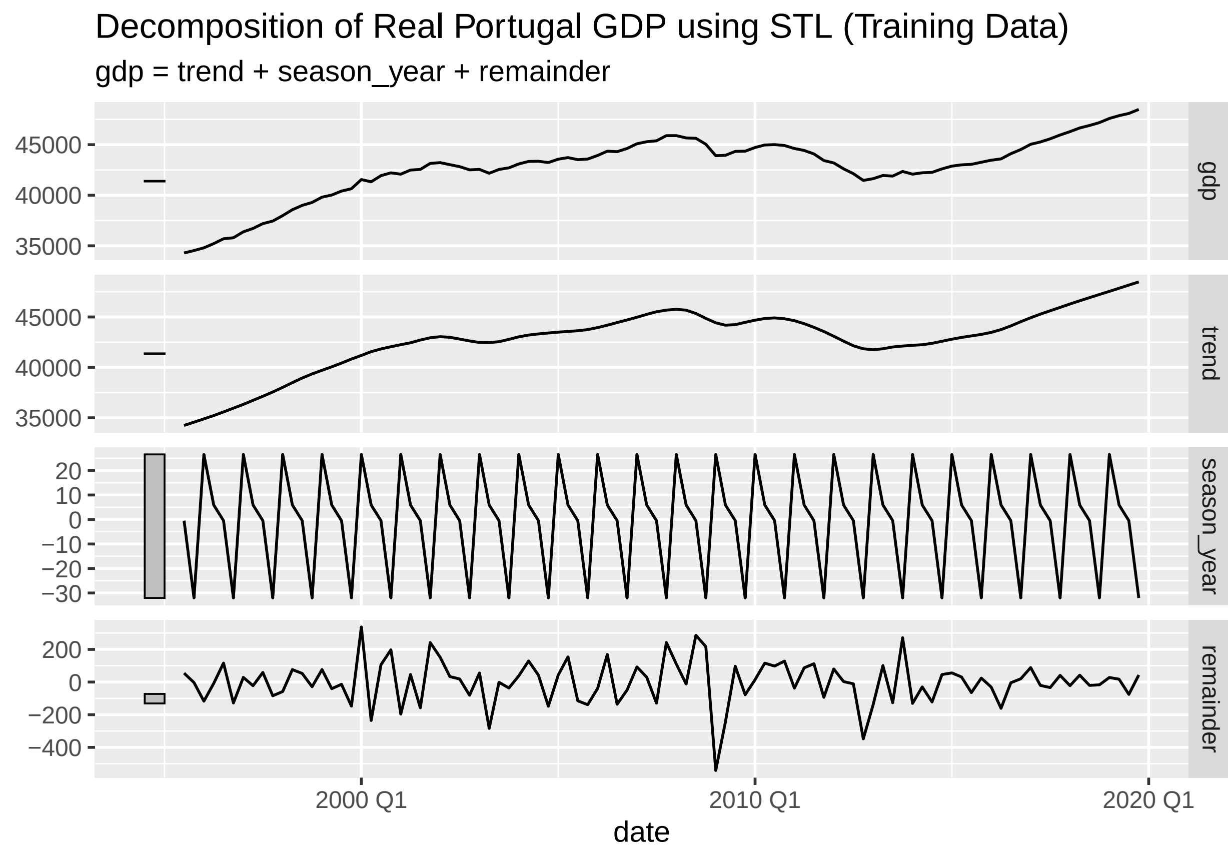
```r
print(guessed_arima_2_forecast)
```

```
## # A fable: 16 x 4 [1Q]
## # Key:     .model [1]
##    .model             date                gdp  .mean
##    <chr>             <qtr>             <dist>  <dbl>
##  1 guessed_arima_2 2020 Q1   N(48740, 88776) 48740.
##  2 guessed_arima_2 2020 Q2  N(49042, 237979) 49042.
##  3 guessed_arima_2 2020 Q3  N(49360, 470183) 49360.
##  4 guessed_arima_2 2020 Q4   N(49626, 8e+05) 49626.
##  5 guessed_arima_2 2021 Q1 N(49872, 1198070) 49872.
##  6 guessed_arima_2 2021 Q2 N(50150, 1702819) 50150.
##  7 guessed_arima_2 2021 Q3 N(50441, 2329426) 50441.
##  8 guessed_arima_2 2021 Q4 N(50694, 3091133) 50694.
##  9 guessed_arima_2 2022 Q1 N(50932, 3940121) 50932.
## 10 guessed_arima_2 2022 Q2 N(51193, 4926034) 51193.
## 11 guessed_arima_2 2022 Q3 N(51463, 6057409) 51463.
## 12 guessed_arima_2 2022 Q4 N(51706, 7344327) 51706.
## 13 guessed_arima_2 2023 Q1 N(51938, 8739895) 51938.
## 14 guessed_arima_2 2023 Q2   N(52187, 1e+07) 52187.
## 15 guessed_arima_2 2023 Q3 N(52442, 1.2e+07) 52442.
## 16 guessed_arima_2 2023 Q4 N(52678, 1.4e+07) 52678.
```

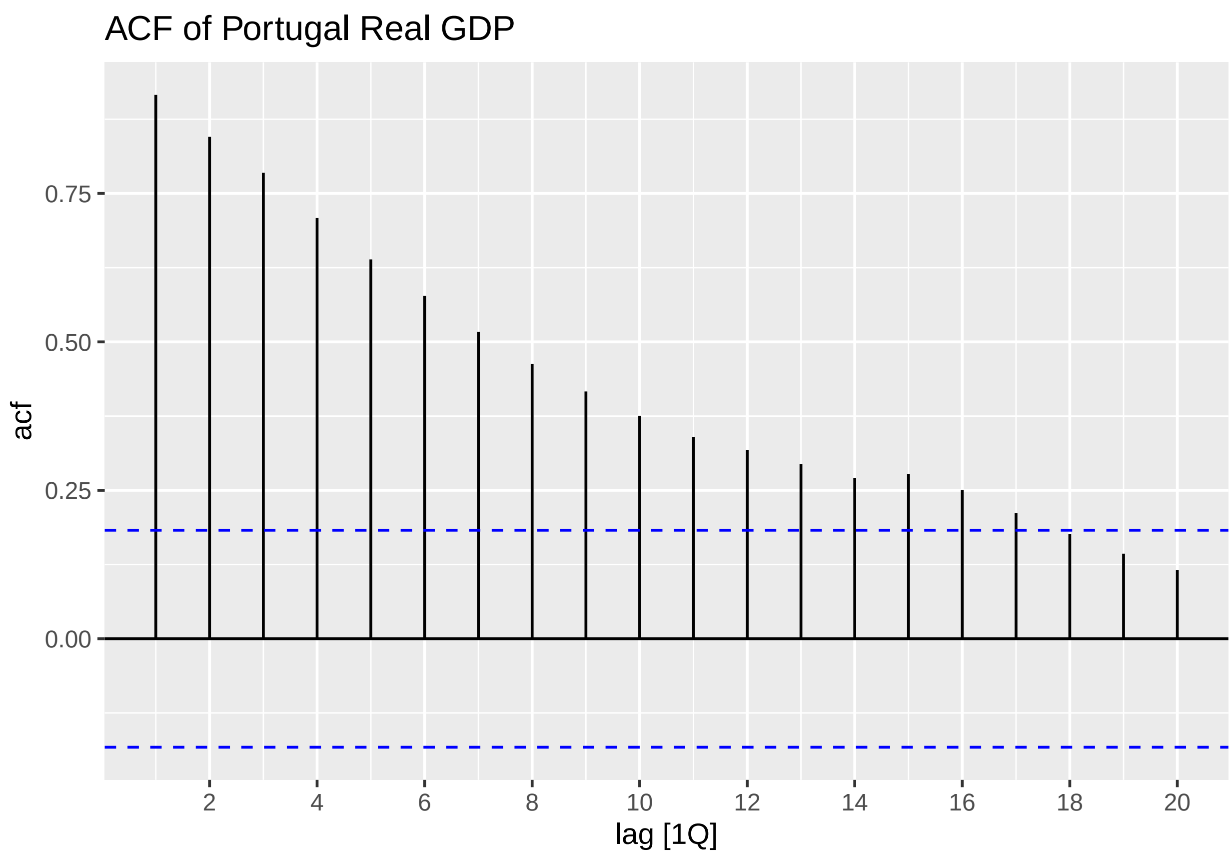
# 21 - Decomposition / Model Selection ETS

```r
### 21.1 - STL
stl_dcmp <- train %>%
```

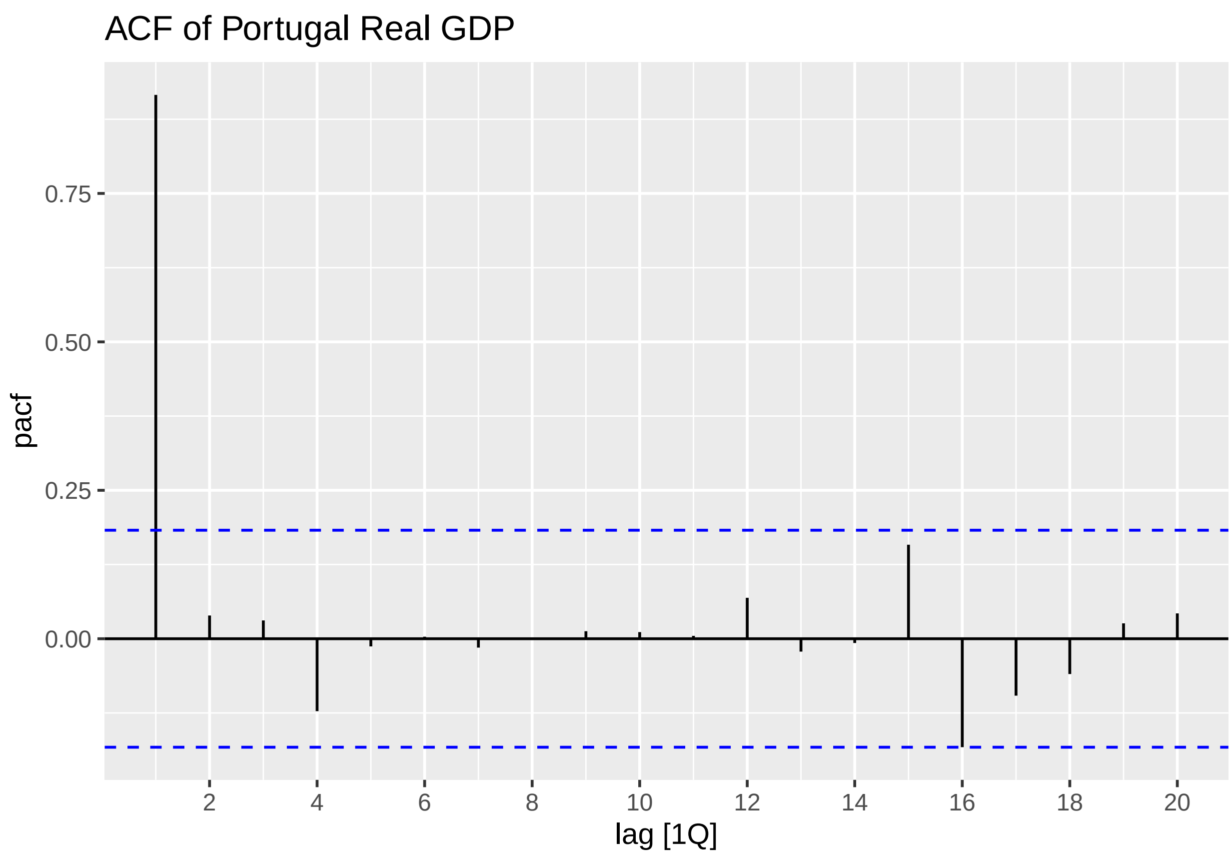
```r
  model(stl = STL(gdp ~ season(window = "periodic"))) %>%
  components()

autoplot(stl_dcmp) +
  labs(title = "Decomposition of Real Portugal GDP using STL (Training Data)")
```




```r
### 21.2 - STL on Training Set
pt %>% ACF(gdp) %>% autoplot() + ggtitle("ACF of Portugal Real GDP")
```




```r
### 21.3 - STL on Training Set
pt %>% PACF(gdp) %>% autoplot() + ggtitle("ACF of Portugal Real GDP")
```




## 22 - ETS

```
# Fit different ETS models to the training data
ets_models <- train %>%
  model(
    auto_ets = ETS(gdp),
    ets_AAN = ETS(gdp ~ error("A") + trend("A") + season("N")),
    ets_AAdN = ETS(gdp ~ error("A") + trend("Ad") + season("N")),
    ets_MAN = ETS(gdp ~ error("M") + trend("A") + season("N"))
  )

# Model Summary
model_summary_ets <- glance(ets_models) %>%
  arrange(AICc) %>%
  select(.model, AIC, AICc, BIC) %>%
  mutate(across(AIC:BIC, ~format(round(.x, 2), nsmall = 2)))

print(model_summary_ets)
```

```
## # A tibble: 4 x 4
##   .model   AIC     AICc    BIC
##   <chr>    <chr>   <chr>   <chr>
## 1 auto_ets 1571.87 1572.79 1587.38
```

```
## 2 ets_AAdN 1572.36 1573.28 1587.87
## 3 ets_MAN  1572.75 1573.40 1585.67
## 4 ets_AAN  1574.07 1574.72 1586.99
```

```r
# Accuracy
accuracy_metrics_ets <- bind_rows(
  train %>%
    model(auto_ets = ETS(gdp)) %>%
    forecast(test) %>%
    fabletools::accuracy(test %>% select(gdp)),

  train %>%
    model(ets_AAN = ETS(gdp ~ error("A") + trend("A") + season("N"))) %>%
    forecast(test) %>%
    fabletools::accuracy(test %>% select(gdp)),

  train %>%
    model(ets_AAdN = ETS(gdp ~ error("A") + trend("Ad") + season("N"))) %>%
    forecast(test) %>%
    fabletools::accuracy(test %>% select(gdp)),

  train %>%
    model(ets_MAN = ETS(gdp ~ error("M") + trend("A") + season("N"))) %>%
    forecast(test) %>%
    fabletools::accuracy(test %>% select(gdp))
) %>%
  select(.model, RMSE, MAE, MAPE) %>%
  mutate(across(RMSE:MAPE, ~format(round(.x, 2), nsmall = 2)))

print(accuracy_metrics_ets)
```

```
## # A tibble: 4 x 4
##   .model    RMSE    MAE     MAPE
##   <chr>     <chr>   <chr>   <chr>
## 1 auto_ets 3350.93 2253.67 5.10
## 2 ets_AAN  3964.91 3431.72 7.50
## 3 ets_AAdN 3325.70 2256.58 5.10
## 4 ets_MAN  3958.59 3422.67 7.48
```

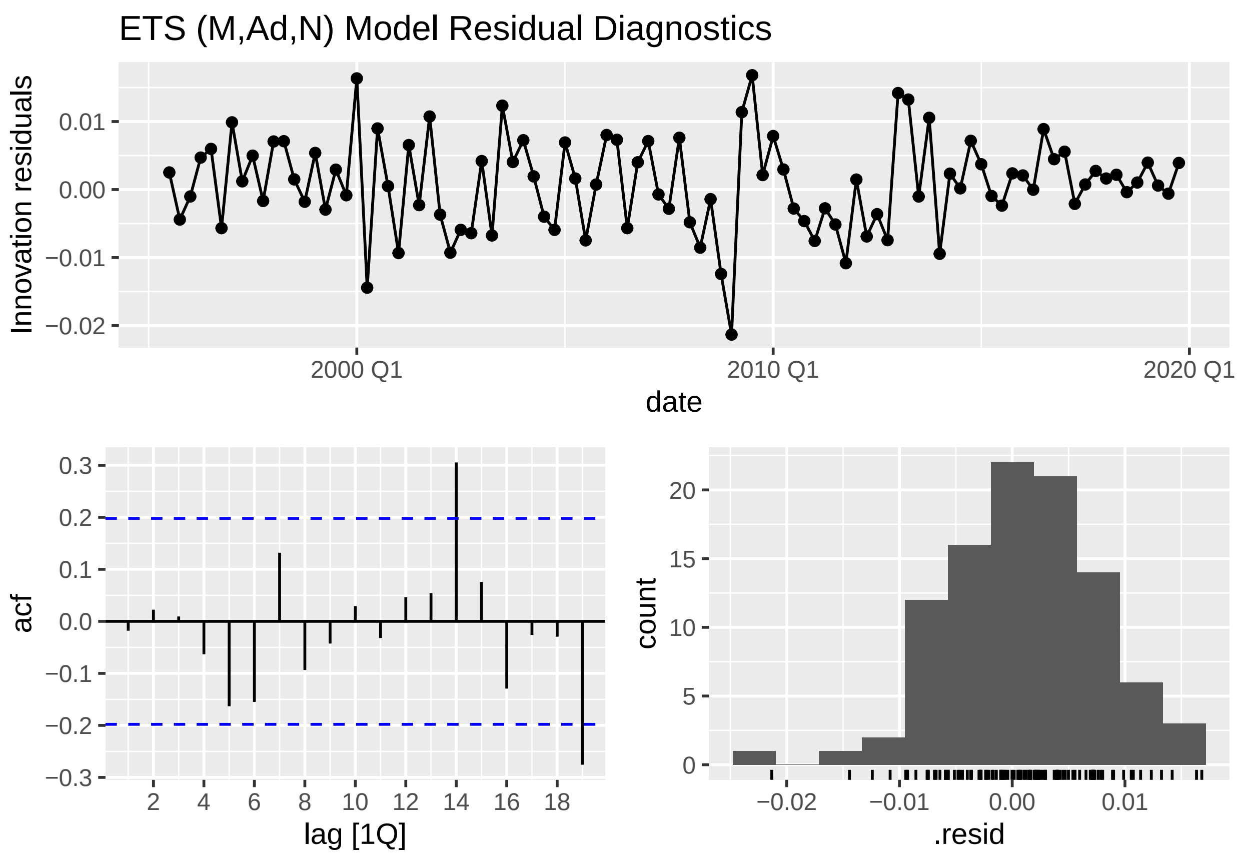
```r
# Fit the Auto ETS model
auto_ets_model <- train %>%
  model(auto_ets = ETS(gdp))

# Print the report
print(auto_ets_model)
```

```
## # A mable: 1 x 1
##        auto_ets
##         <model>
## 1 <ETS(M,Ad,N)>
```

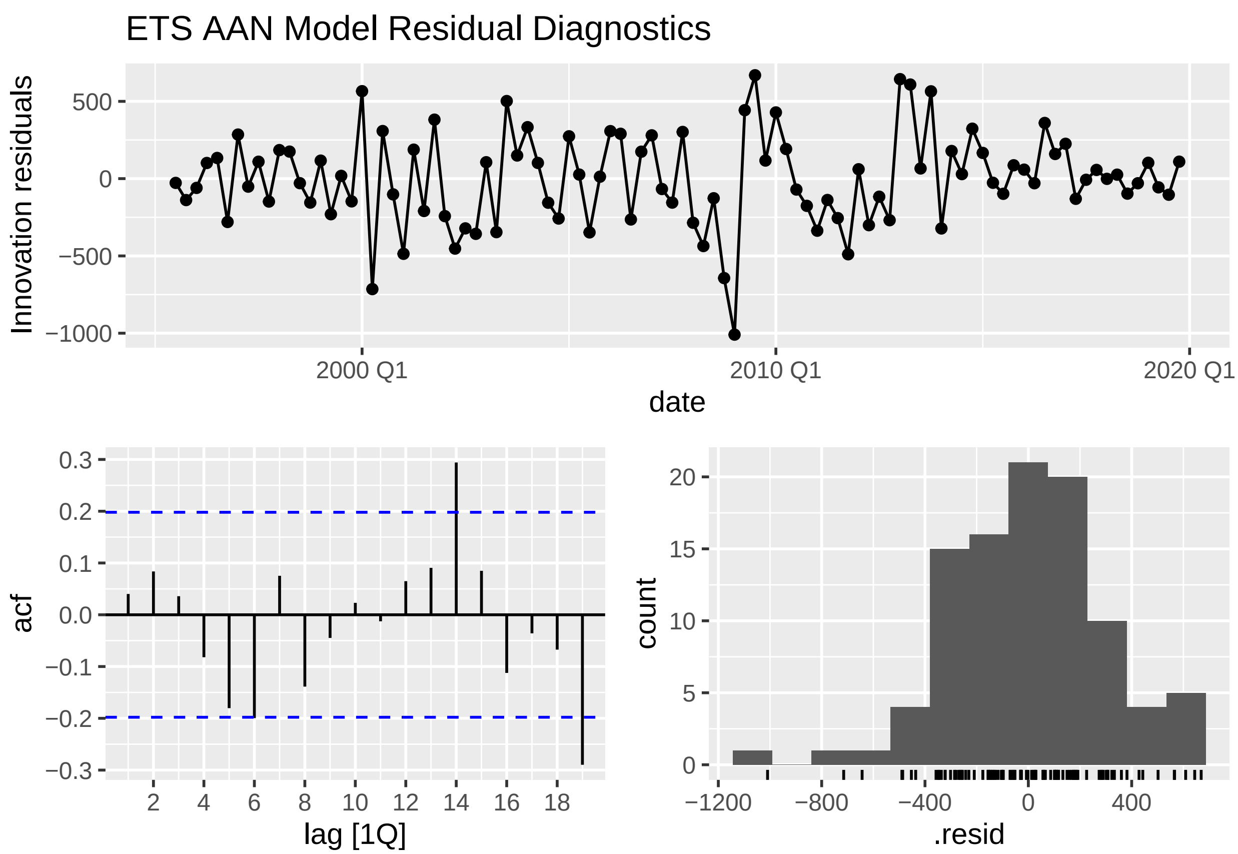
```
#(M,Ad,N)
```

## 23 - Residuals ETS

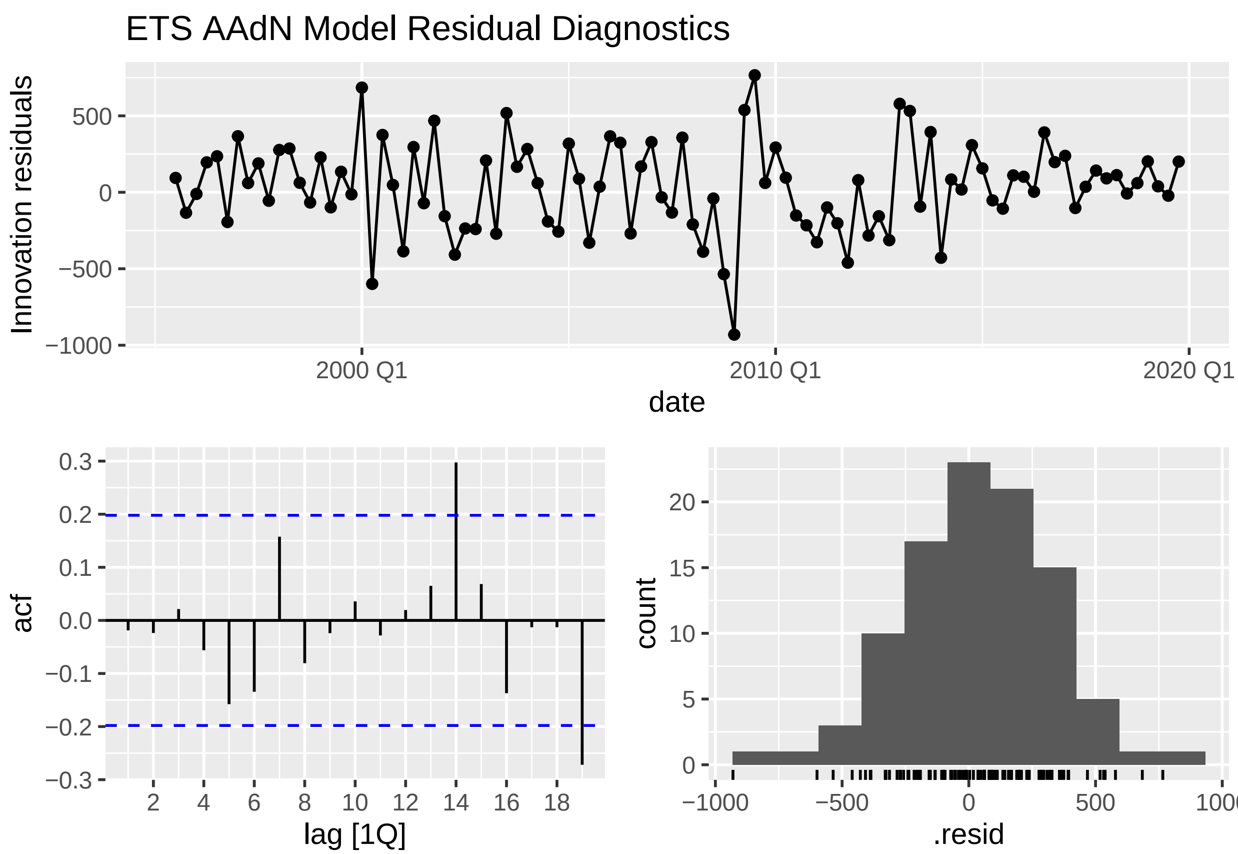
```
ets_models %>%
  select(auto_ets) %>%
  gg_tsresiduals(type = "innovation") +
  ggtitle("ETS (M,Ad,N) Model Residual Diagnostics")
```



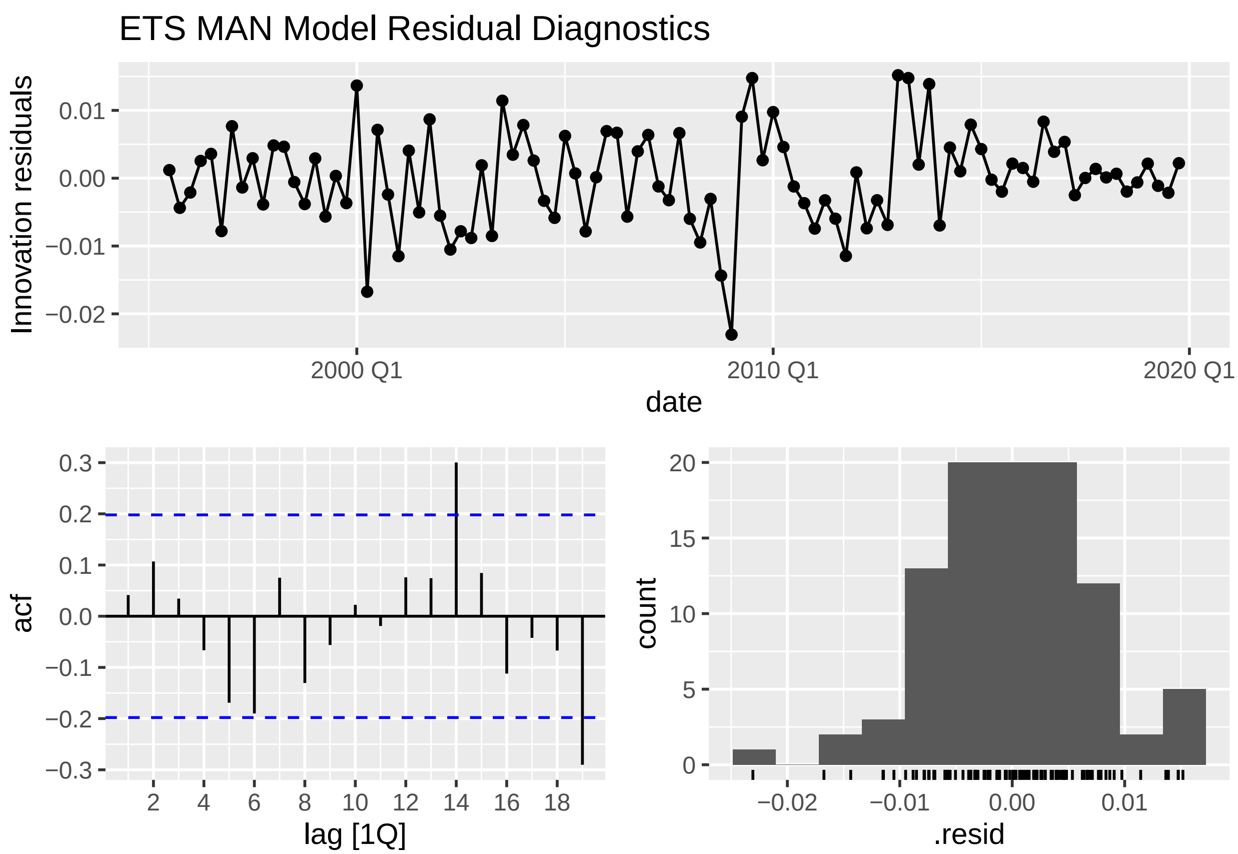

```
ets_models %>%
  select(ets_AAN) %>%
  gg_tsresiduals(type = "innovation") +
  ggtitle("ETS AAN Model Residual Diagnostics")
```




```r
ets_models %>%
  select(ets_AAdN) %>%
  gg_tsresiduals(type = "innovation") +
  ggtitle("ETS AAdN Model Residual Diagnostics")
```




```r
ets_models %>%
  select(ets_MAN) %>%
  gg_tsresiduals(type = "innovation") +
  ggtitle("ETS MAN Model Residual Diagnostics")
```




## 24 - Ljung-Box Test ETS

```
# Perform the Ljung-Box test for the Auto ETS model
lb_test_auto_ets <- augment(ets_models %>% select(auto_ets)) %>%
  features(.resid, ljung_box, lag = 10, dof = 2)

# Perform the Ljung-Box test for the ETS AAN model
lb_test_ets_AAN <- augment(ets_models %>% select(ets_AAN)) %>%
  features(.resid, ljung_box, lag = 10, dof = 2)

# Perform the Ljung-Box test for the ETS AAdN model
lb_test_ets_AAdN <- augment(ets_models %>% select(ets_AAdN)) %>%
  features(.resid, ljung_box, lag = 10, dof = 2)

# Perform the Ljung-Box test for the ETS MAN model
lb_test_ets_MAN <- augment(ets_models %>% select(ets_MAN)) %>%
  features(.resid, ljung_box, lag = 10, dof = 2)

# Display Ljung-Box test results
print(lb_test_auto_ets)
```

```
## # A tibble: 1 x 3
##   .model   lb_stat lb_pvalue
```

```
##   <chr>       <dbl>     <dbl>
## 1 auto_ets    9.53     0.300
```

```
print(lb_test_ets_AAN)
```

```
## # A tibble: 1 x 3
##   .model  lb_stat lb_pvalue
##   <chr>     <dbl>     <dbl>
## 1 ets_AAN    12.4     0.135
```

```
print(lb_test_ets_AAdN)
```

```
## # A tibble: 1 x 3
##   .model   lb_stat lb_pvalue
##   <chr>      <dbl>     <dbl>
## 1 ets_AAdN    8.61     0.376
```

```
print(lb_test_ets_MAN)
```

```
## # A tibble: 1 x 3
##   .model  lb_stat lb_pvalue
##   <chr>     <dbl>     <dbl>
## 1 ets_MAN    12.6     0.128
```

# 25 - Shapiro-Wilk Test ETS

```
shapiro_test_auto_ets <- shapiro.test(ets_models %>%
                                        select(auto_ets) %>%
                                        residuals() %>%
                                        select(.resid) %>%
                                        as.ts())

shapiro_test_ets_AAN <- shapiro.test(ets_models %>%
                                       select(ets_AAN) %>%
                                       residuals() %>%
                                       select(.resid) %>%
                                       as.ts())

shapiro_test_ets_AAdN <- shapiro.test(ets_models %>%
                                        select(ets_AAdN) %>%
                                        residuals() %>%
                                        select(.resid) %>%
                                        as.ts())

shapiro_test_ets_MAN <- shapiro.test(ets_models %>%
                                       select(ets_MAN) %>%
                                       residuals() %>%
                                       select(.resid) %>%
                                       as.ts())
```

```r
# Print the results
print(shapiro_test_auto_ets)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ets_models %>% select(auto_ets) %>% residuals() %>% select(.resid) %>% as.ts()
## W = 0.99296, p-value = 0.8924
```

```r
print(shapiro_test_ets_AAN)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ets_models %>% select(ets_AAN) %>% residuals() %>% select(.resid) %>% as.ts()
## W = 0.98664, p-value = 0.4287
```

```r
print(shapiro_test_ets_AAdN)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ets_models %>% select(ets_AAdN) %>% residuals() %>% select(.resid) %>% as.ts()
## W = 0.99256, p-value = 0.8681
```

```r
print(shapiro_test_ets_MAN)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ets_models %>% select(ets_MAN) %>% residuals() %>% select(.resid) %>% as.ts()
## W = 0.98738, p-value = 0.4788
```

# 26 - Smoothing Parameters ETS

```r
# Auto ETS Model
report(ets_models %>%
         select(auto_ets))
```

```
## Series: gdp
## Model: ETS(M,Ad,N)
##   Smoothing parameters:
##     alpha = 0.9143283
##     beta  = 0.3969956
##     phi   = 0.8882289
##
##   Initial states:
```

```
##      l[0]     b[0]
##  33799.42 459.3424
##
##   sigma^2:  0
##
##      AIC     AICc      BIC
## 1571.866 1572.790 1587.376
```

```
# ETS AAN Model
report(ets_models %>%
         select(ets_AAN))
```

```
## Series: gdp
## Model: ETS(A,A,N)
##   Smoothing parameters:
##     alpha = 0.9998999
##     beta  = 0.2770357
##
##   Initial states:
##      l[0]     b[0]
##  33942.41 378.6373
##
##   sigma^2:  90814.94
##
##      AIC     AICc      BIC
## 1574.068 1574.720 1586.992
```

```
# ETS AAdN Model
report(ets_models %>%
         select(ets_AAdN))
```

```
## Series: gdp
## Model: ETS(A,Ad,N)
##   Smoothing parameters:
##     alpha = 0.8632823
##     beta  = 0.4967614
##     phi   = 0.8476961
##
##   Initial states:
##      l[0]     b[0]
##  33799.03 472.4967
##
##   sigma^2:  88382.93
##
##      AIC     AICc      BIC
## 1572.359 1573.282 1587.869
```

```
# ETS MAN Model
report(ets_models %>%
         select(ets_MAN))
```

```
## Series: gdp
```

```
## Model: ETS(M,A,N)
##   Smoothing parameters:
##     alpha = 0.9998996
##     beta  = 0.2497498
##
##   Initial states:
##      l[0]     b[0]
##  33878.59 373.9652
##
##   sigma^2:  0
##
##      AIC     AICc      BIC
## 1572.745 1573.398 1585.670
```

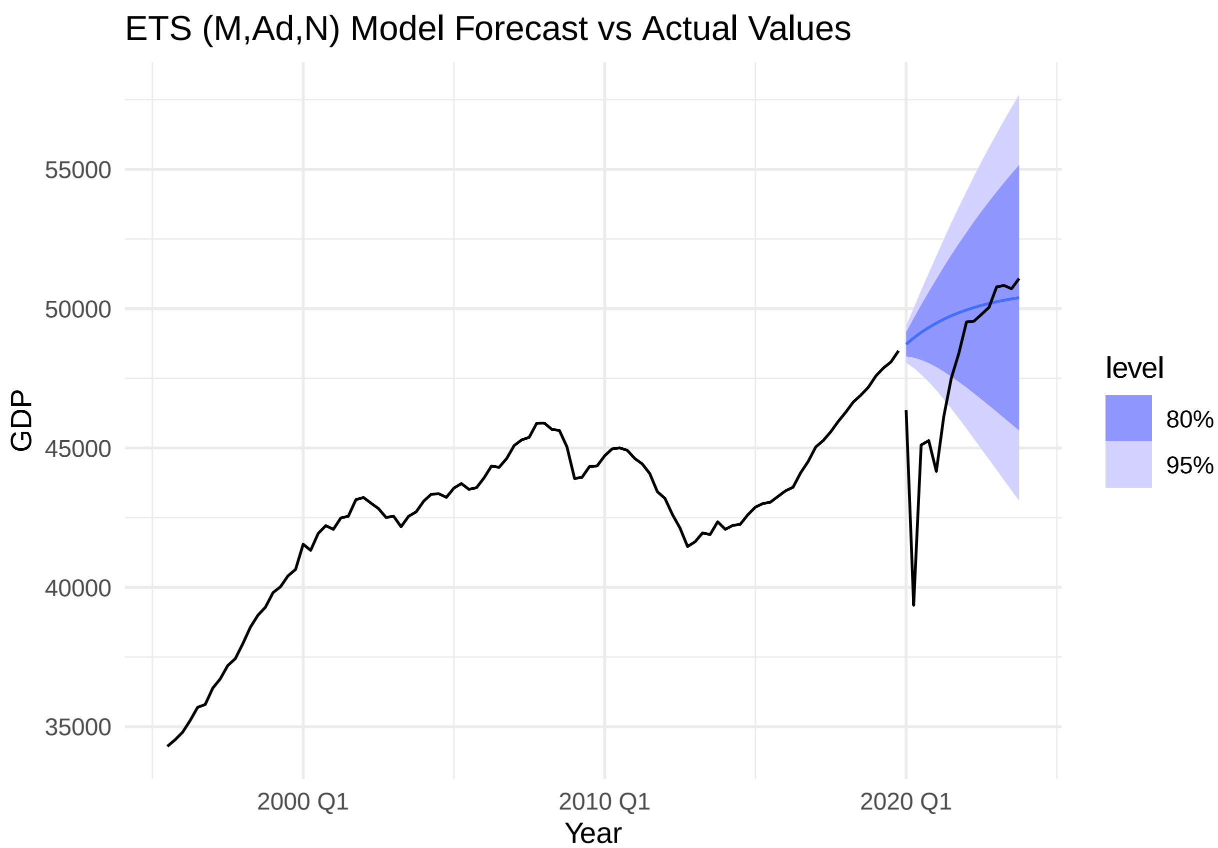
## 27 - Forecasting ETS

```
# Forecast the ETS models on the test data
ets_auto_forecast <- ets_models %>%
  select(auto_ets) %>%
  forecast(h = 16)

ets_AAN_forecast <- ets_models %>%
  select(ets_AAN) %>%
  forecast(h = 16)

ets_AAdN_forecast <- ets_models %>%
  select(ets_AAdN) %>%
  forecast(h = 16)

ets_MAN_forecast <- ets_models %>%
  select(ets_MAN) %>%
  forecast(h = 16)

# Plotting the forecasts along with actual values in the test set
autoplot(train, gdp) +
  autolayer(ets_auto_forecast, .mean, series = "Auto ETS Forecast") +
  autolayer(test, gdp, series = "Actual Values") +
  ggtitle("ETS (M,Ad,N) Model Forecast vs Actual Values") +
  labs(x = "Year", y = "GDP") +
  theme_minimal()
```

```
## Warning in ggdist::geom_lineribbon(without(intvl_mapping, "colour_ramp"), :
## Ignoring unknown parameters: `series`
```

```
## Warning in geom_line(mapping = without(mapping, "shape"), data =
## unpack_data(object[single_row[["FALSE"]], : Ignoring unknown parameters:
## `series`
```

```
## Warning in geom_line(eval_tidy(expr(aes(!!!aes_spec))), data = object, ..., :
## Ignoring unknown parameters: `series`
```

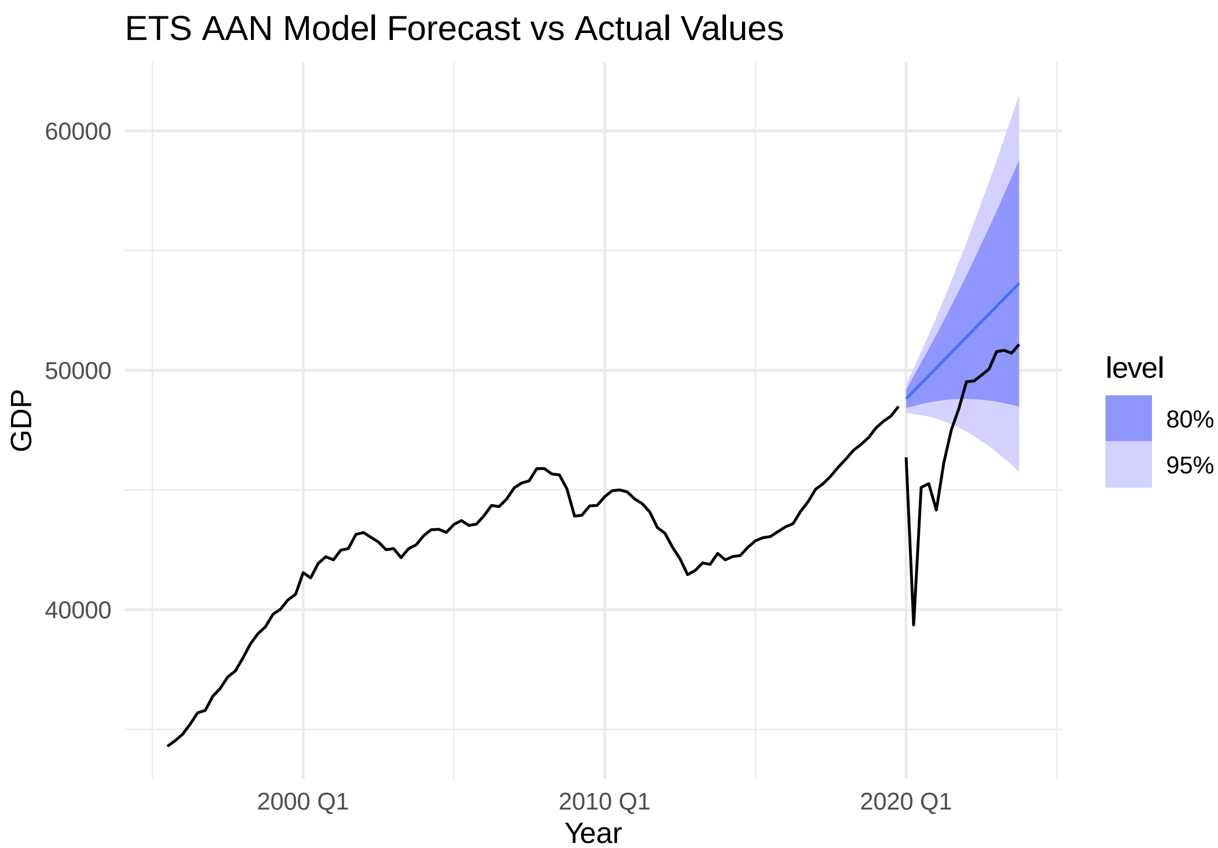



```r
autoplot(train, gdp) +
  autolayer(ets_AAN_forecast, .mean, series = "ETS AAN Forecast") +
  autolayer(test, gdp, series = "Actual Values") +
  ggtitle("ETS AAN Model Forecast vs Actual Values") +
  labs(x = "Year", y = "GDP") +
  theme_minimal()
```

```
## Warning in ggdist::geom_lineribbon(without(intvl_mapping, "colour_ramp"), :
## Ignoring unknown parameters: `series`
```

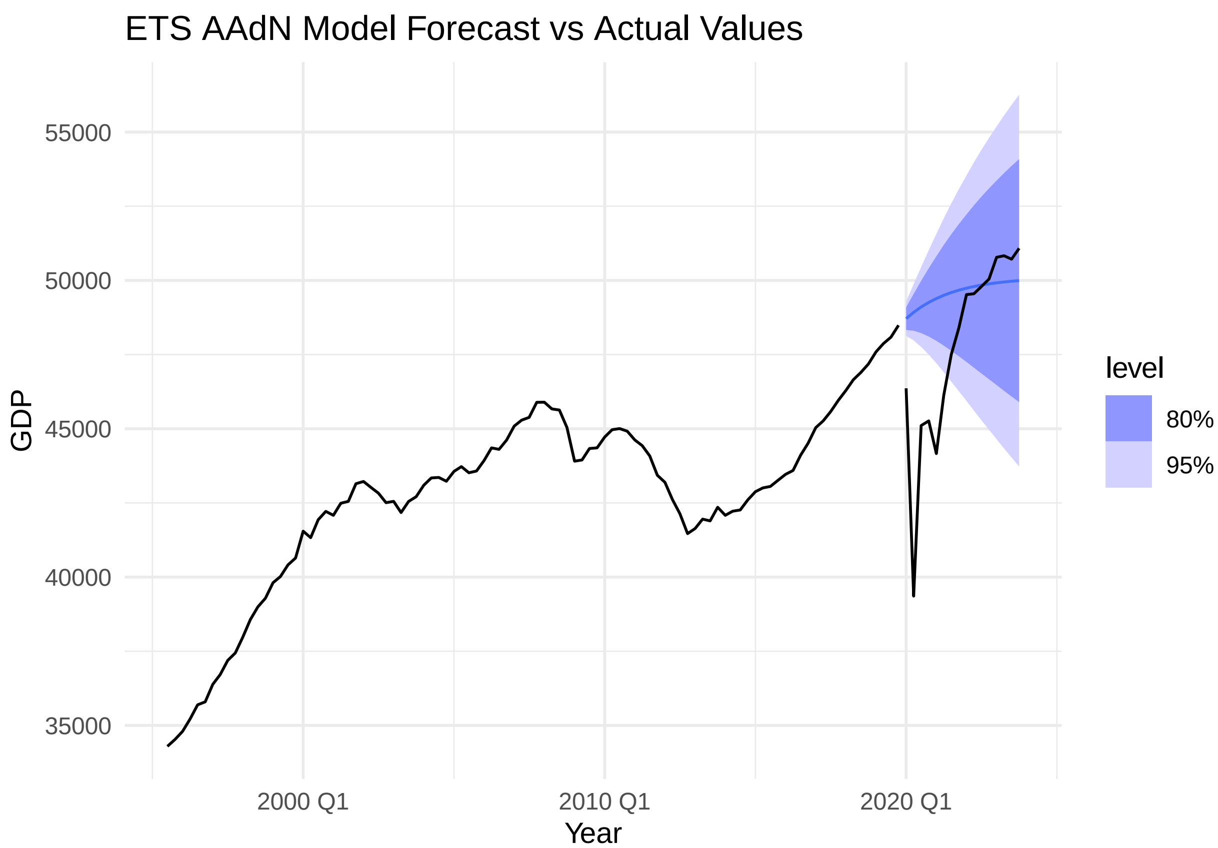
```
## Warning in geom_line(mapping = without(mapping, "shape"), data =
## unpack_data(object[single_row[["FALSE"]], : Ignoring unknown parameters:
## `series`
```

```
## Warning in geom_line(eval_tidy(expr(aes(!!!aes_spec))), data = object, ..., :
## Ignoring unknown parameters: `series`
```



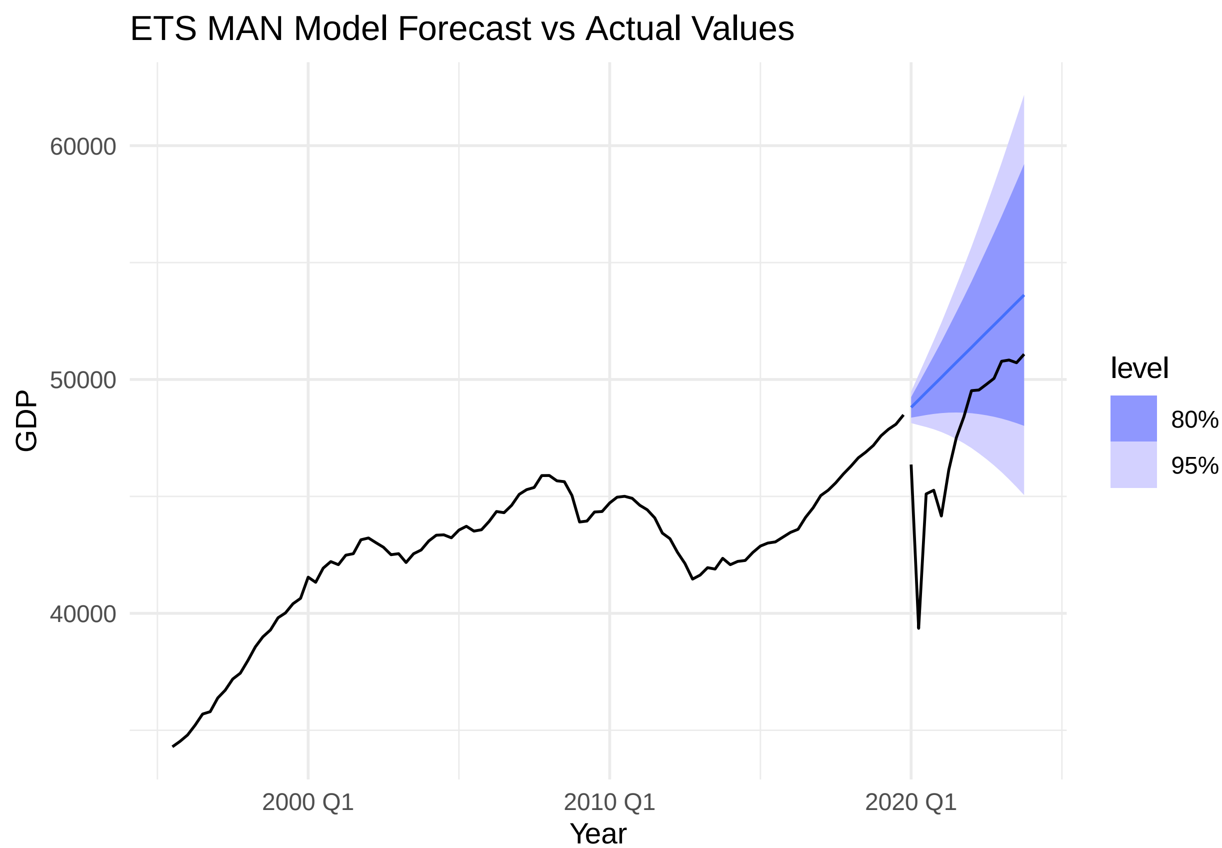

```r
autoplot(train, gdp) +
  autolayer(ets_AAdN_forecast, .mean, series = "ETS AAdN Forecast") +
  autolayer(test, gdp, series = "Actual Values") +
  ggtitle("ETS AAdN Model Forecast vs Actual Values") +
  labs(x = "Year", y = "GDP") +
  theme_minimal()
```

```
## Warning in ggdist::geom_lineribbon(without(intvl_mapping, "colour_ramp"), :
## Ignoring unknown parameters: ‘series‘

## Warning in geom_line(mapping = without(mapping, "shape"), data =
## unpack_data(object[single_row[["FALSE"]], : Ignoring unknown parameters:
## ‘series‘

## Warning in geom_line(eval_tidy(expr(aes(!!!aes_spec))), data = object, ..., :
## Ignoring unknown parameters: ‘series‘
```




```r
autoplot(train, gdp) +
  autolayer(ets_MAN_forecast, .mean, series = "ETS MAN Forecast") +
  autolayer(test, gdp, series = "Actual Values") +
  ggtitle("ETS MAN Model Forecast vs Actual Values") +
  labs(x = "Year", y = "GDP") +
  theme_minimal()
```

```
## Warning in ggdist::geom_lineribbon(without(intvl_mapping, "colour_ramp"), :
## Ignoring unknown parameters: ‘series‘

## Warning in geom_line(mapping = without(mapping, "shape"), data =
## unpack_data(object[single_row[["FALSE"]], : Ignoring unknown parameters:
## ‘series‘

## Warning in geom_line(eval_tidy(expr(aes(!!!aes_spec))), data = object, ..., :
## Ignoring unknown parameters: ‘series‘
```




```r
# Print the forecast values for each model
print(ets_auto_forecast)
```

```
## # A fable: 16 x 4 [1Q]
## # Key:     .model [1]
##    .model      date                gdp  .mean
##    <chr>      <qtr>             <dist>  <dbl>
##  1 auto_ets 2020 Q1  N(48723, 115072) 48723.
##  2 auto_ets 2020 Q2   N(48948, 3e+05) 48948.
##  3 auto_ets 2020 Q3  N(49147, 590858) 49147.
##  4 auto_ets 2020 Q4  N(49324, 993343) 49324.
##  5 auto_ets 2021 Q1 N(49481, 1511724) 49481.
##  6 auto_ets 2021 Q2 N(49621, 2145917) 49621.
##  7 auto_ets 2021 Q3 N(49745, 2893361) 49745.
##  8 auto_ets 2021 Q4 N(49855, 3749803) 49855.
##  9 auto_ets 2022 Q1 N(49953, 4709879) 49953.
## 10 auto_ets 2022 Q2 N(50040, 5767562) 50040.
## 11 auto_ets 2022 Q3 N(50117, 6916481) 50117.
## 12 auto_ets 2022 Q4 N(50186, 8150166) 50186.
## 13 auto_ets 2023 Q1 N(50247, 9462217) 50247.
## 14 auto_ets 2023 Q2 N(50301, 1.1e+07) 50301.
## 15 auto_ets 2023 Q3 N(50349, 1.2e+07) 50349.
## 16 auto_ets 2023 Q4 N(50392, 1.4e+07) 50392.
```

```r
print(ets_AAN_forecast)
```

```
## # A fable: 16 x 4 [1Q]
## # Key:     .model [1]
##    .model     date                 gdp  .mean
##    <chr>     <qtr>              <dist>  <dbl>
##  1 ets_AAN 2020 Q1   N(48809, 90815) 48809.
##  2 ets_AAN 2020 Q2  N(49130, 238895) 49130.
##  3 ets_AAN 2020 Q3  N(49452, 458197) 49452.
##  4 ets_AAN 2020 Q4  N(49773, 762662) 49773.
##  5 ets_AAN 2021 Q1 N(50095, 1166229) 50095.
##  6 ets_AAN 2021 Q2 N(50416, 1682839) 50416.
##  7 ets_AAN 2021 Q3 N(50738, 2326431) 50738.
##  8 ets_AAN 2021 Q4 N(51059, 3110946) 51059.
##  9 ets_AAN 2022 Q1 N(51381, 4050322) 51381.
## 10 ets_AAN 2022 Q2 N(51702, 5158500) 51702.
## 11 ets_AAN 2022 Q3 N(52024, 6449420) 52024.
## 12 ets_AAN 2022 Q4 N(52345, 7937021) 52345.
## 13 ets_AAN 2023 Q1 N(52667, 9635244) 52667.
## 14 ets_AAN 2023 Q2 N(52988, 1.2e+07) 52988.
## 15 ets_AAN 2023 Q3 N(53310, 1.4e+07) 53310.
## 16 ets_AAN 2023 Q4 N(53631, 1.6e+07) 53631.
```

```r
print(ets_AAdN_forecast)
```

```
## # A fable: 16 x 4 [1Q]
## # Key:     .model [1]
##    .model      date                 gdp  .mean
##    <chr>      <qtr>              <dist>  <dbl>
##  1 ets_AAdN 2020 Q1   N(48711, 88383) 48711.
##  2 ets_AAdN 2020 Q2  N(48924, 234183) 48924.
##  3 ets_AAdN 2020 Q3  N(49105, 472290) 49105.
##  4 ets_AAdN 2020 Q4  N(49258, 806285) 49258.
##  5 ets_AAdN 2021 Q1 N(49388, 1234239) 49388.
##  6 ets_AAdN 2021 Q2 N(49498, 1750950) 49498.
##  7 ets_AAdN 2021 Q3 N(49592, 2349447) 49592.
##  8 ets_AAdN 2021 Q4   N(49671, 3e+06) 49671.
##  9 ets_AAdN 2022 Q1 N(49738, 3760641) 49738.
## 10 ets_AAdN 2022 Q2 N(49795, 4557799) 49795.
## 11 ets_AAdN 2022 Q3 N(49843, 5406287) 49843.
## 12 ets_AAdN 2022 Q4 N(49884, 6299540) 49884.
## 13 ets_AAdN 2023 Q1 N(49919, 7231641) 49919.
## 14 ets_AAdN 2023 Q2 N(49948, 8197323) 49948.
## 15 ets_AAdN 2023 Q3 N(49973, 9191934) 49973.
## 16 ets_AAdN 2023 Q4   N(49994, 1e+07) 49994.
```

```r
print(ets_MAN_forecast)
```

```
## # A fable: 16 x 4 [1Q]
## # Key:     .model [1]
##    .model     date                 gdp  .mean
##    <chr>     <qtr>              <dist>  <dbl>
```

```
##   1 ets_MAN 2020 Q1  N(48807, 117447) 48807.
##   2 ets_MAN 2020 Q2   N(49128, 3e+05) 49128.
##   3 ets_MAN 2020 Q3  N(49448, 570454) 49448.
##   4 ets_MAN 2020 Q4  N(49769, 937322) 49769.
##   5 ets_MAN 2021 Q1 N(50089, 1418961) 50089.
##   6 ets_MAN 2021 Q2   N(50410, 2e+06) 50410.
##   7 ets_MAN 2021 Q3 N(50730, 2791386) 50730.
##   8 ets_MAN 2021 Q4 N(51051, 3715116) 51051.
##   9 ets_MAN 2022 Q1 N(51371, 4819509) 51371.
##  10 ets_MAN 2022 Q2 N(51692, 6121576) 51692.
##  11 ets_MAN 2022 Q3 N(52012, 7638549) 52012.
##  12 ets_MAN 2022 Q4 N(52332, 9387883) 52332.
##  13 ets_MAN 2023 Q1 N(52653, 1.1e+07) 52653.
##  14 ets_MAN 2023 Q2 N(52973, 1.4e+07) 52973.
##  15 ets_MAN 2023 Q3 N(53294, 1.6e+07) 53294.
##  16 ets_MAN 2023 Q4 N(53614, 1.9e+07) 53614.
```

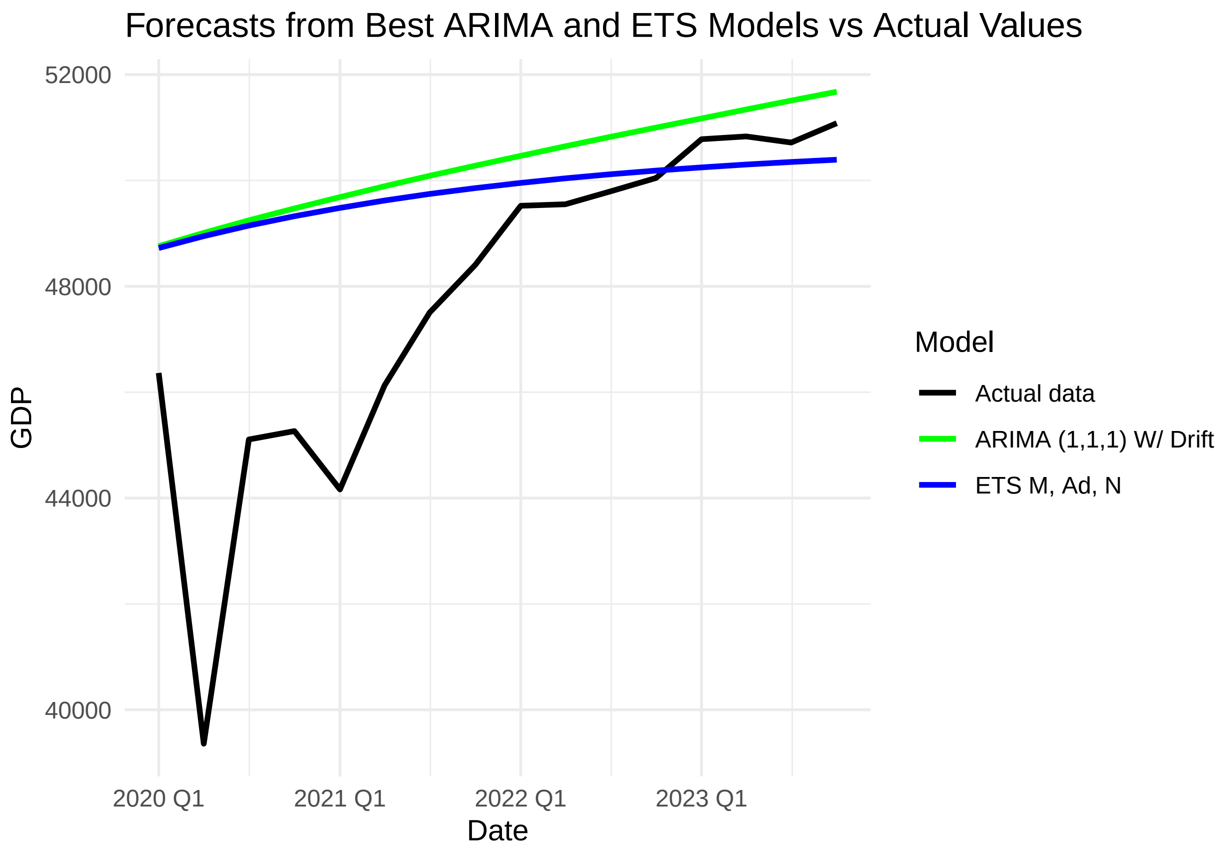
# 28 - ARIMA VS ETS

## 28.1 - Visualization

```r
# Convert forecasts to data frame and add Model column
arima_forecast <- auto_arima_drift_forecast %>%
  as_tibble() %>%
  mutate(Model = "ARIMA (1,1,1) W/ Drift")

ets_forecast <- ets_auto_forecast %>%
  as_tibble() %>%
  mutate(Model = "ETS M, Ad, N")

# Convert actual data to tibble and add Model column
actual_data <- test %>%
  as_tibble() %>%
  mutate(Model = "Actual data") %>%
  rename(.mean = gdp)

# Combine all data for plotting
combined_forecasts <- bind_rows(
  arima_forecast %>% select(date, .mean, Model),
  ets_forecast %>% select(date, .mean, Model),
  actual_data %>% select(date, .mean, Model)
)

# Create the plot with actual data in black
ggplot(combined_forecasts, aes(x = date, y = .mean, color = Model)) +
  geom_line(size = 1) +
  scale_color_manual(values = c("Actual data" = "black", "ARIMA (1,1,1) W/ Drift" = "green", "ETS M, Ad
  labs(title = "Forecasts from Best ARIMA and ETS Models vs Actual Values",
       x = "Date",
       y = "GDP") +
  theme_minimal() +
```

```
  theme(legend.position = "right")
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




### 28.2 - Accuracy on Filtered Test Dataset

```
test_filtered <- test %>%
  filter(date >= yearquarter("2022 Q1"))

# Generate forecasts for the entire test period
auto_arima_forecast <- train %>%
  model(auto_arima = ARIMA(gdp ~ 1)) %>%
  forecast(test)

auto_ets_forecast <- train %>%
  model(auto_ets = ETS(gdp)) %>%
  forecast(test)
```

```r
# Filter the forecasts to only include data from 2022-Q1 onwards
auto_arima_filtered <- auto_arima_forecast %>%
  filter(date >= yearquarter("2022 Q1"))

auto_ets_filtered <- auto_ets_forecast %>%
  filter(date >= yearquarter("2022 Q1"))

# Calculate accuracy metrics using the filtered forecast results
accuracy_metrics_filtered <- bind_rows(
  fabletools::accuracy(auto_arima_filtered, test %>% select(gdp)),
  fabletools::accuracy(auto_ets_filtered, test %>% select(gdp))
) %>%
  select(.model, RMSE, MAE, MAPE) %>%
  mutate(across(RMSE:MAPE, ~format(round(.x, 2), nsmall = 2)))

# Print the accuracy metrics
print(accuracy_metrics_filtered)
```

```
## # A tibble: 2 x 4
##   .model     RMSE   MAE    MAPE
##   <chr>      <chr>  <chr>  <chr>
## 1 auto_arima 826.01 789.05 1.57
## 2 auto_ets   464.62 438.23 0.87
```

## 28.3 - Visualization on Filtered Dataset

```r
# Convert forecasts to data frame and add Model column
arima_forecast_filtered <- auto_arima_filtered %>%
  as_tibble() %>%
  mutate(Model = "ARIMA (1,1,1) W/ Drift")

ets_forecast_filtered <- auto_ets_filtered %>%
  as_tibble() %>%
  mutate(Model = "ETS M, Ad, N")

# Convert actual filtered data to tibble and add Model column
actual_data_filtered <- test_filtered %>%
  as_tibble() %>%
  mutate(Model = "Actual data") %>%
  rename(.mean = gdp)

# Combine all data for plotting
combined_forecasts_filtered <- bind_rows(
  arima_forecast_filtered %>% select(date, .mean, Model),
  ets_forecast_filtered %>% select(date, .mean, Model),
  actual_data_filtered %>% select(date, .mean, Model)
)

# Create the plot with actual data in black
ggplot(combined_forecasts_filtered, aes(x = date, y = .mean, color = Model)) +
  geom_line(size = 1) +
```

```r
  scale_color_manual(values = c("Actual data" = "black", "ARIMA (1,1,1) W/ Drift" = "green", "ETS M, Ad
  labs(title = "Forecasts from Best ARIMA and ETS Models vs Actual Values (2022-Q1 Onwards)",
       x = "Date",
       y = "GDP") +
  theme_minimal() +
  theme(legend.position = "right")
```