Time Series Homework

Table of contents

Authors :

Antoine Riollet and Francisco Gonzales Garcia

Problem 1

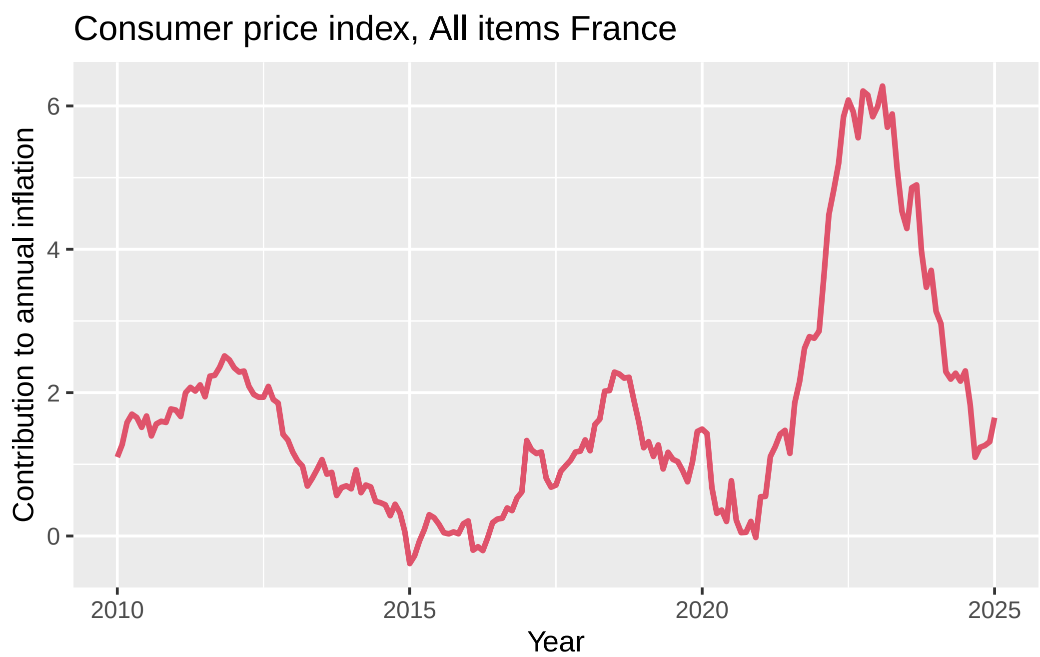
1.1: Question 1

1.1.1: Price index

```
# import the data
priceindex1 <- read.csv("price index.csv")
# rename the column
priceindex1 <- rename(priceindex1, index = FRACPALTT01CTGYM)
# create the time series
priceindex <- ts(priceindex1[,2], frequency=12, start=c(2010,1))
# structure
str(priceindex)
```

```
 Time-Series [1:181] from 2010 to 2025: 1.1 1.28 1.58 1.7 1.65 ...
```

```
# plot the time series
autoplot(priceindex, col=2, xlab="Year", ylab="Contribution to annual inflation", main="Consu
```



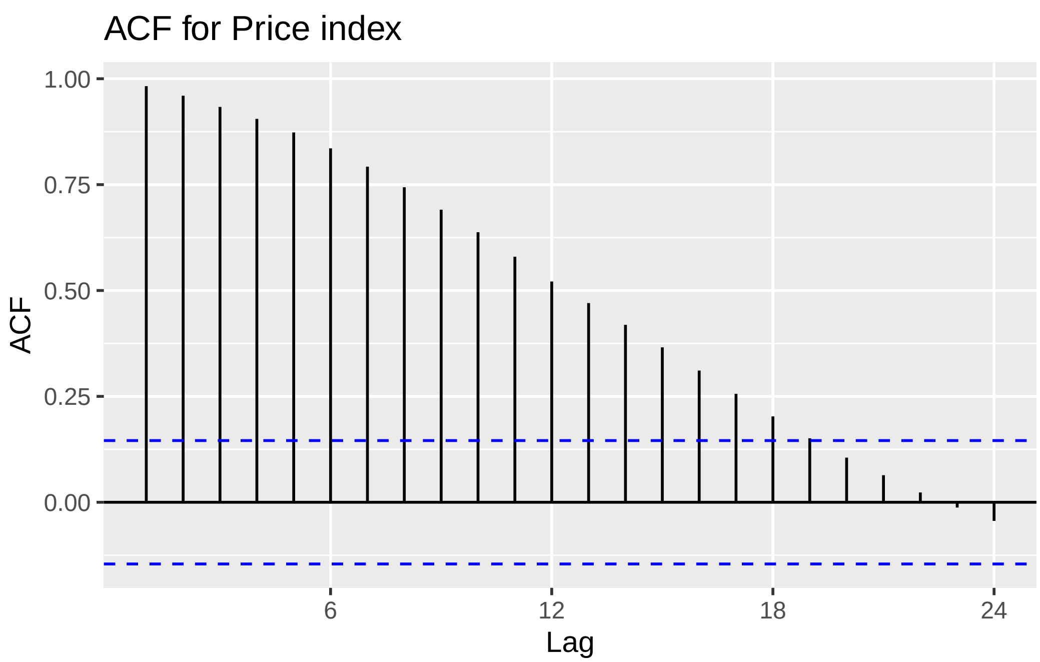
The series shows a clear upward trend, particularly after 2020, suggesting non-stationarity.

```
# descriptive statistics
summary(priceindex)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.3842  0.6756  1.3157  1.6942  2.1610  6.2767
```

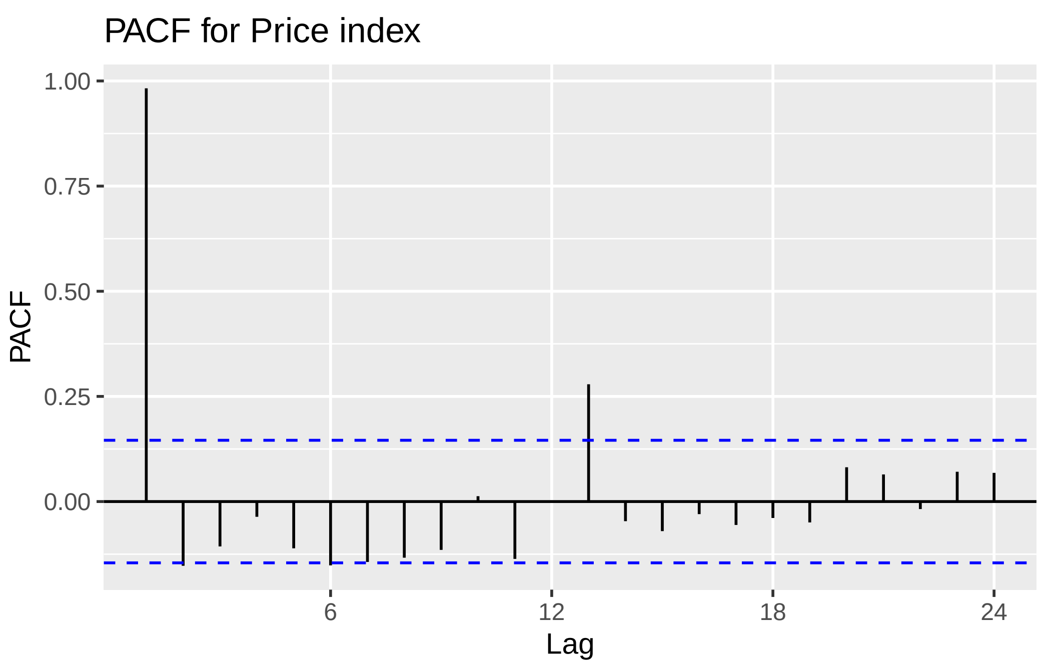
The mean is around 1.69 and the maximum is above 6, indicating high variability likely linked to recent inflation shocks.

```
# acf
ggAcf(priceindex) + ggtitle("ACF for Price index")
```



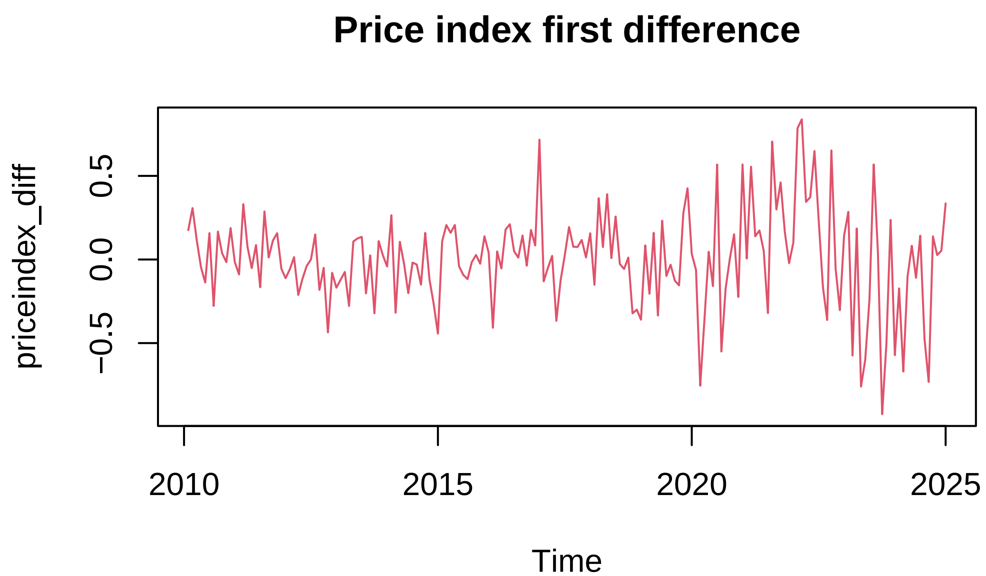
The ACF shows a very slow decay typical of a non-stationary process.

```
# pacf
ggPacf(priceindex) + ggtitle("PACF for Price index")
```



The PACF suggests an AR(1) structure since the first lag is significant and then drops.

```r
# first difference
priceindex_diff <- diff(priceindex)
# plot
ts.plot(priceindex_diff, col=2, main="Price index first difference")
```



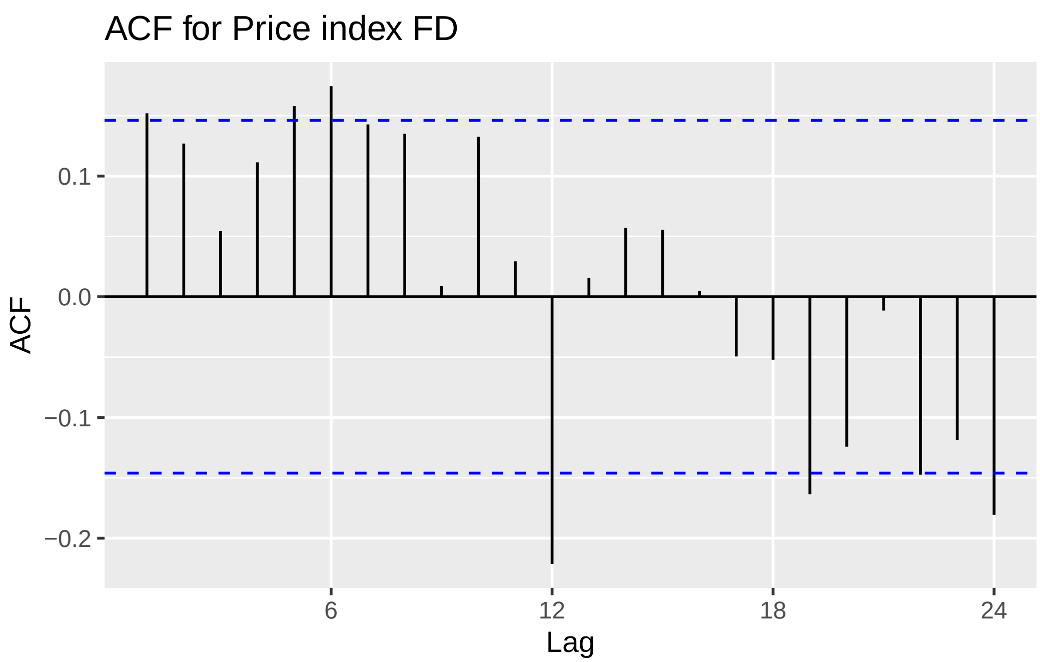
The differenced series now looks much more stationary.

```r
# summary of first difference
summary(priceindex_diff)
```

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-0.925169 -0.132480  0.011701  0.003057  0.157070  0.838363
```

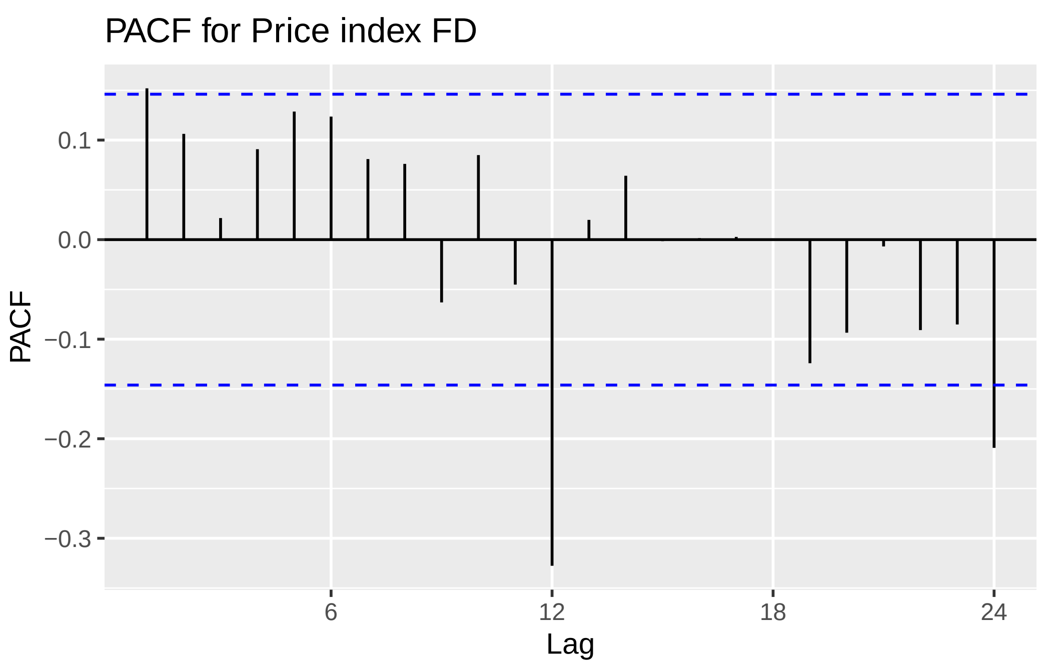
Variance is reduced, and the mean is close to zero as expected.

```r
# acf of first difference
ggAcf(priceindex_diff) + ggtitle("ACF for Price index FD")
```



The ACF shows a quick decay indicating stationarity.

```
# pacf of first difference
ggPacf(priceindex_diff) + ggtitle("PACF for Price index FD")
```



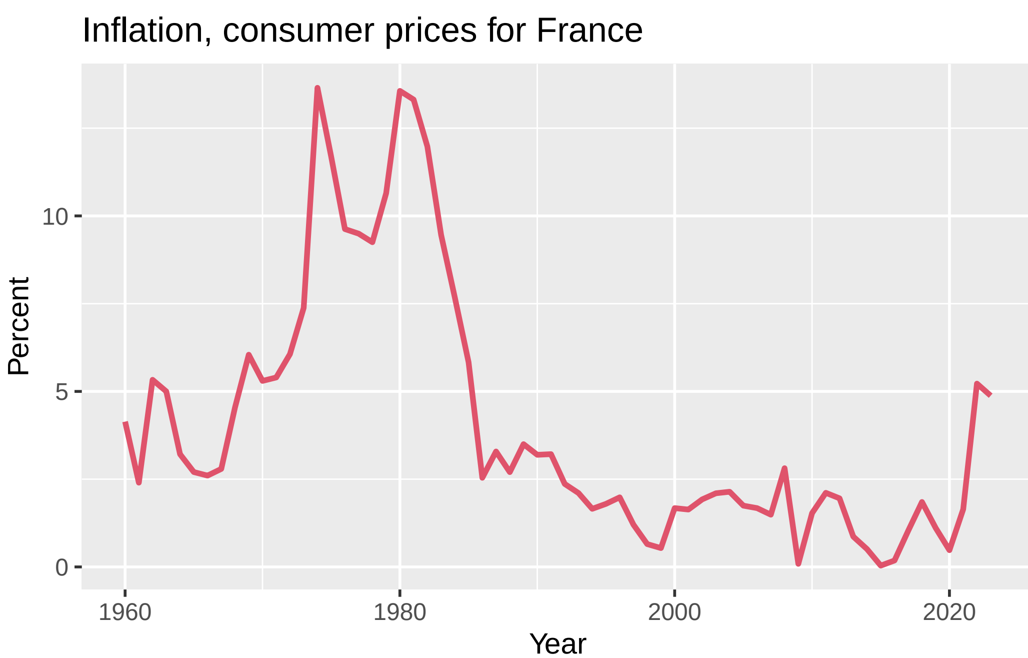
The PACF confirms stationarity with only the first lag being significant.

1.1.2: Inflation

```
# import the data
INF1 <- read.csv("inflation.csv")
INF1 <- rename(INF1, inflation = FPCPITOTLZGFRA)
inf <- ts(INF1[,2], frequency=1, start=c(1960,1))
str(inf)
```

```
Time-Series [1:64] from 1960 to 2023: 4.14 2.4 5.33 5 3.21 ...
```

```
# plot the time series
autoplot(inf, col=2, xlab="Year", ylab="Percent", main="Inflation, consumer prices for Franc
```



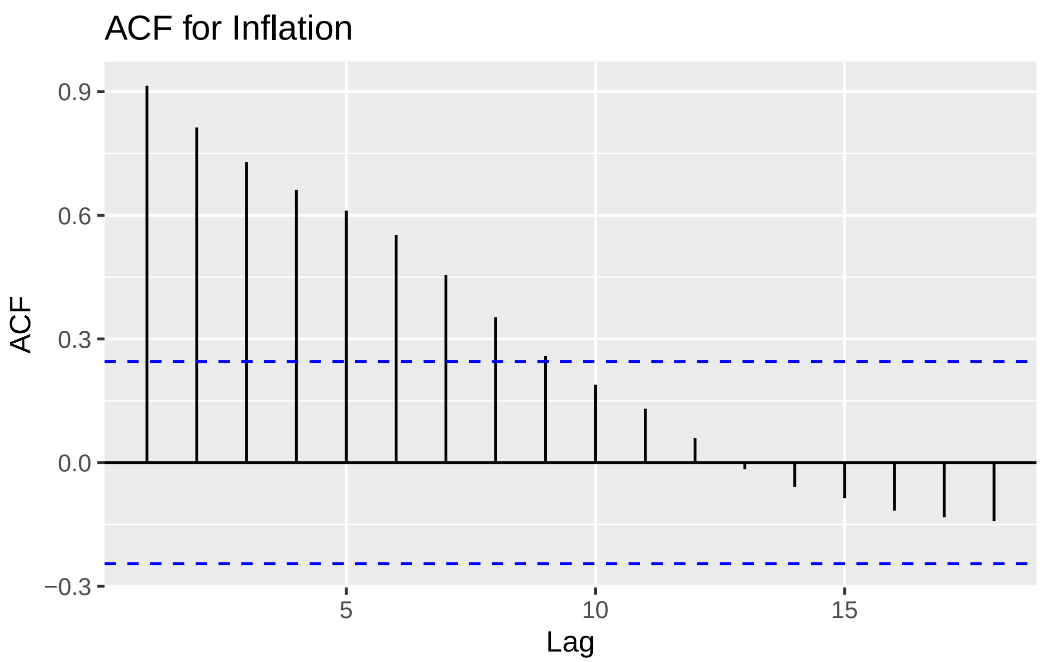
The series shows several periods of high inflation (1970s-1980s) and a general decline afterward.

```
# descriptive statistics
summary(inf)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
 0.03751  1.67022  2.65141  4.07024  5.34784 13.64932 
```

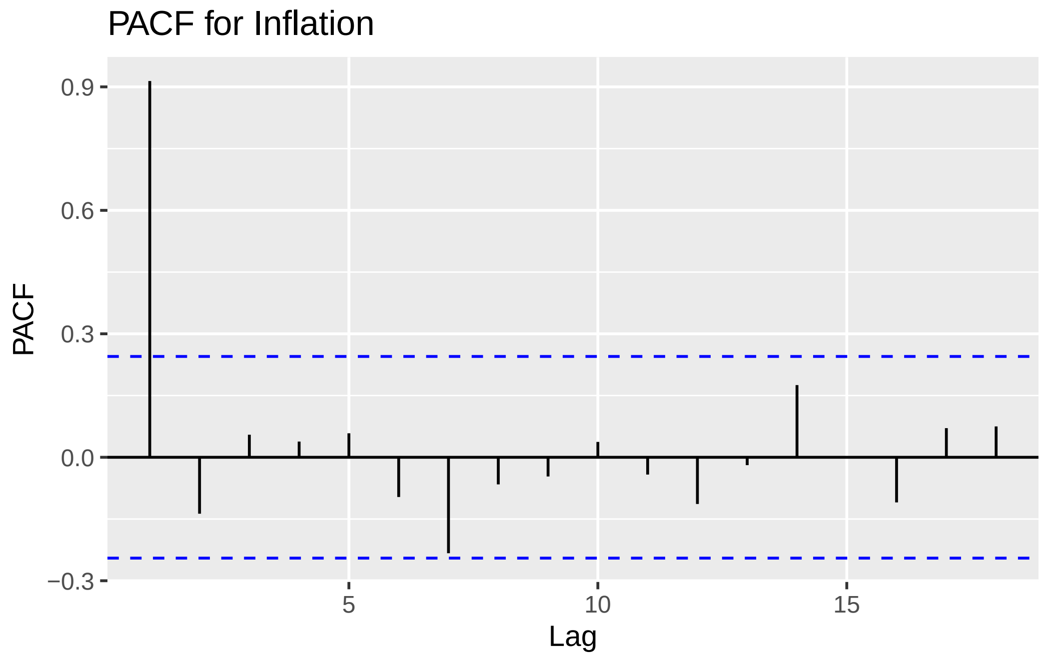
High variance and high maximum values confirm volatility during earlier periods.

```r
# acf
ggAcf(inf) + ggtitle("ACF for Inflation")
```



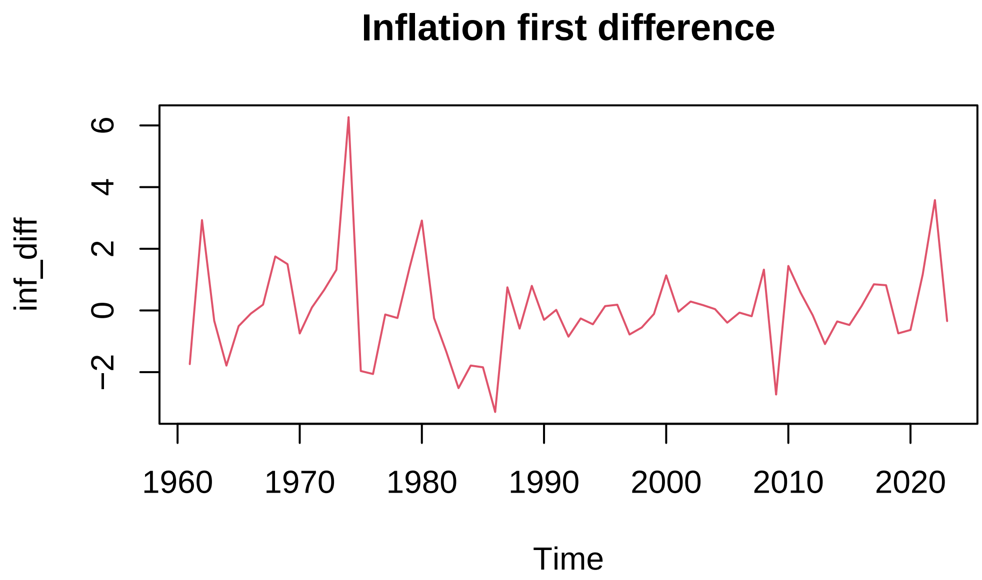
ACF declines slowly, confirming the non-stationarity.

```r
# pacf
ggPacf(inf) + ggtitle("PACF for Inflation")
```



PACF is significant at lag 1 but quickly vanishes.

```
# first difference
inf_diff <- diff(inf)
ts.plot(inf_diff, col=2, main="Inflation first difference")
```



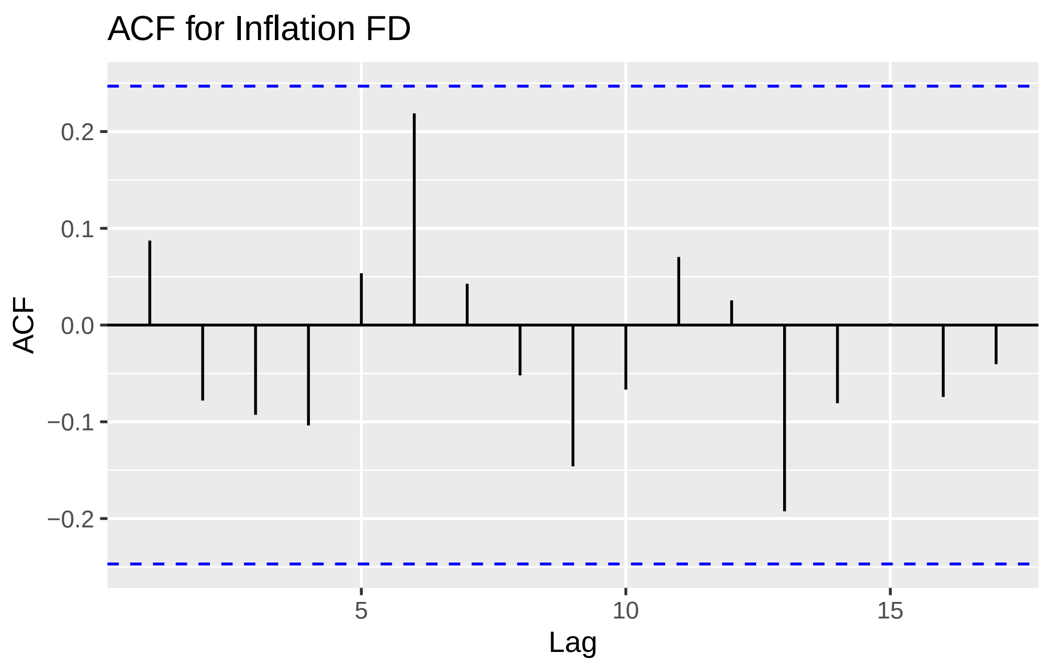
The differenced series looks stationary.

```r
# summary of first difference
summary(inf_diff)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
-3.29257 -0.60992 -0.13094  0.01172  0.70793  6.26872 
```

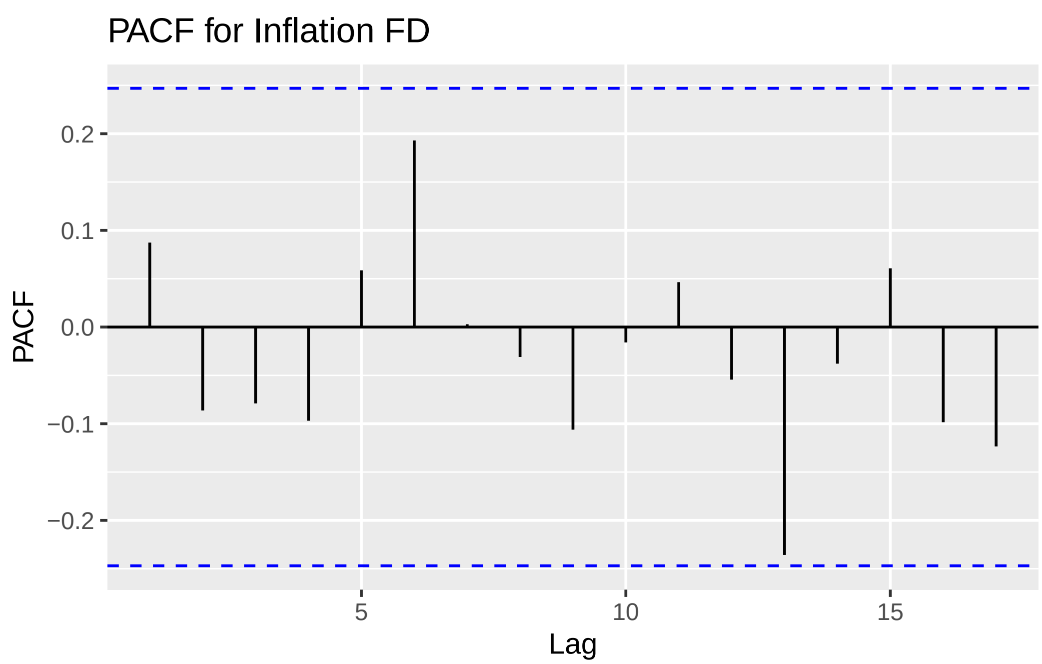
Variance is reduced and mean is closer to zero.

```r
# acf of first difference
ggAcf(inf_diff) + ggtitle("ACF for Inflation FD")
```



Quick decay of ACF confirms stationarity.

```
# pacf of first difference
ggPacf(inf_diff) + ggtitle("PACF for Inflation FD")
```



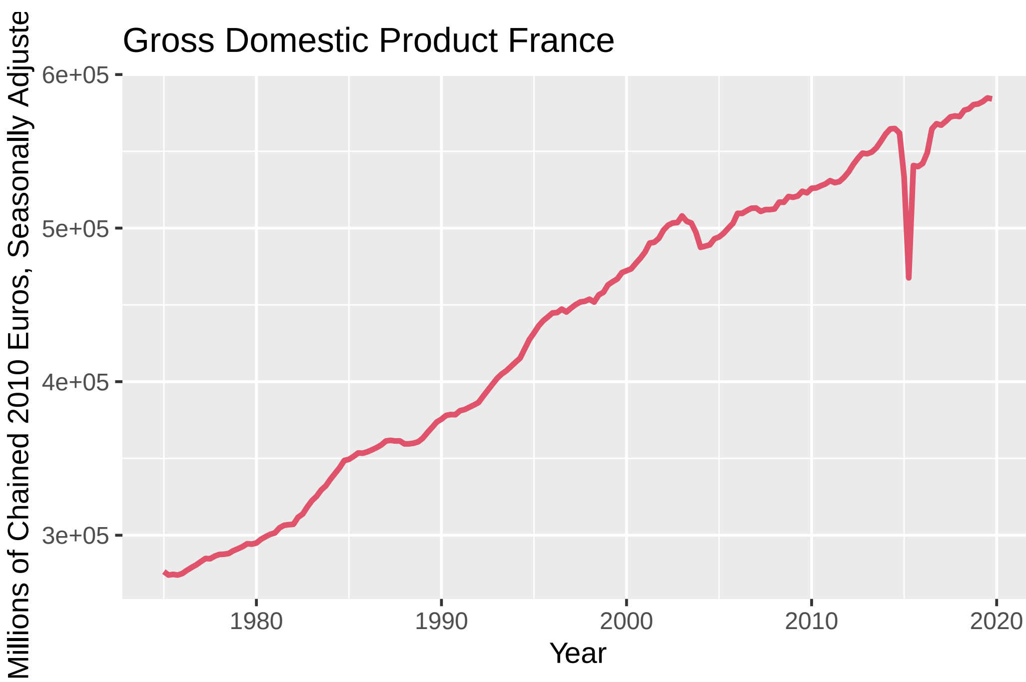
PACF confirms that the first difference is close to a white noise.

1.1.3: GDP

```
GDP1 <- read.csv("real gdp.csv")
GDP1 <- rename(GDP1, gdp = CLVMNACSCAB1GQFR)
GDP <- ts(GDP1[,2], frequency=4, start=c(1975,1))
str(GDP)
```

```
 Time-Series [1:180] from 1975 to 2020: 276310 274085 274499 274071 275089 ...
```

```
autoplot(GDP, col=2, xlab="Year", ylab="Millions of Chained 2010 Euros, Seasonally Adjusted"
```



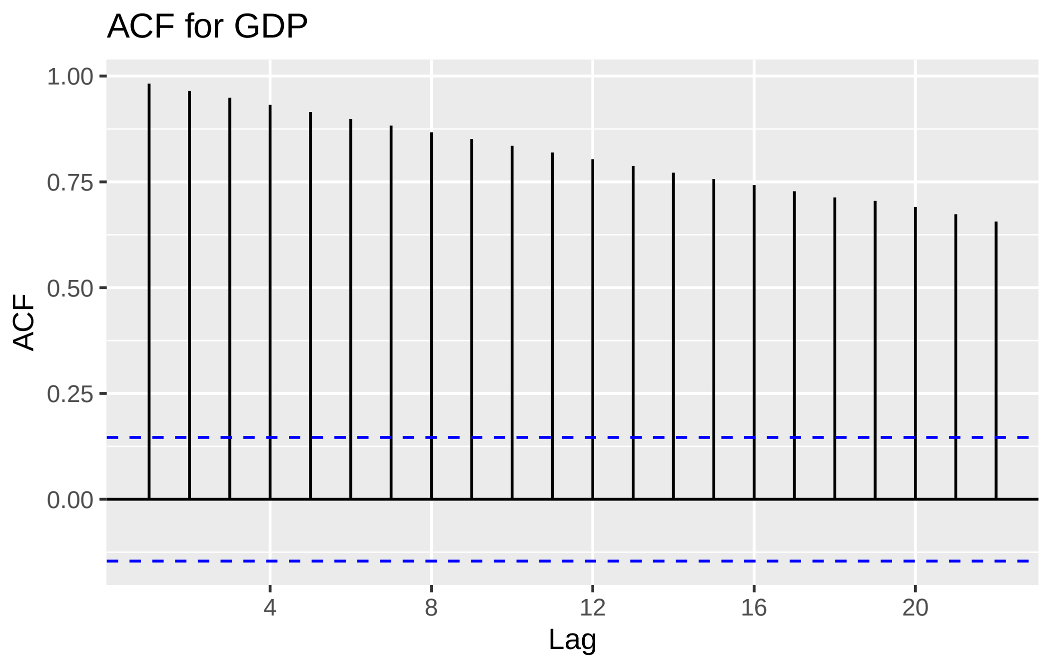
GDP shows a strong upward trend suggesting non-stationarity.

```
summary(GDP)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 274071  355322  450994  435086  516853  584753 
```

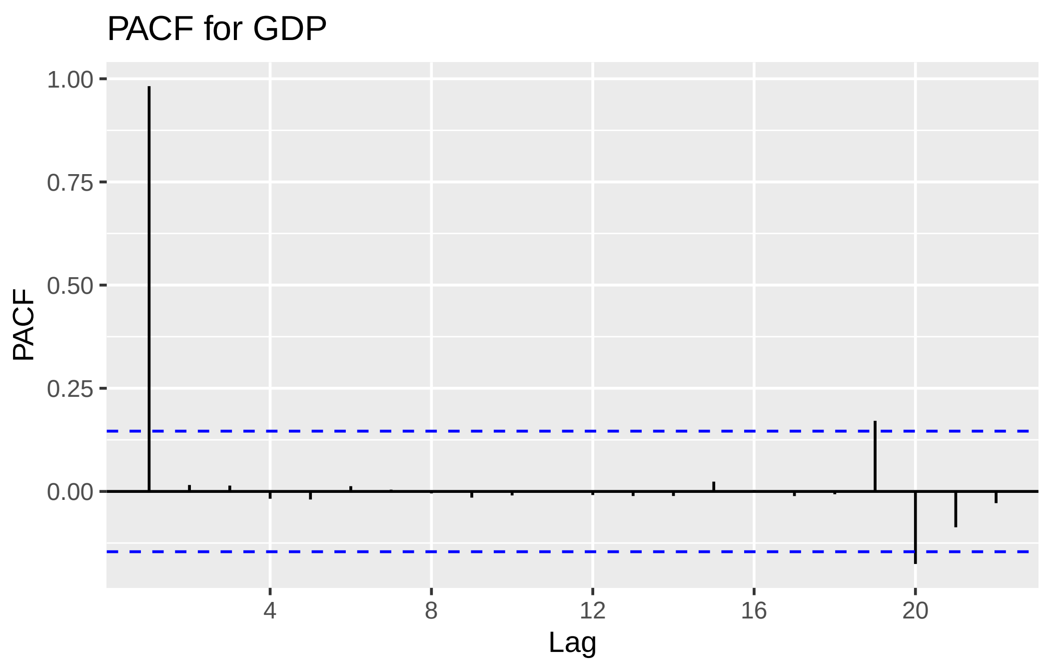
Descriptive statistics confirm the trend and high variance.

```
ggAcf(GDP) + ggtitle("ACF for GDP")
```



The ACF decreases slowly confirming non-stationarity.

```
ggPacf(GDP) + ggtitle("PACF for GDP")
```



PACF significant at lag 1 suggests an AR(1) trend component.

```
GDP_diff <- diff(GDP)
ts.plot(GDP_diff, col=2, main="GDP first difference")
```

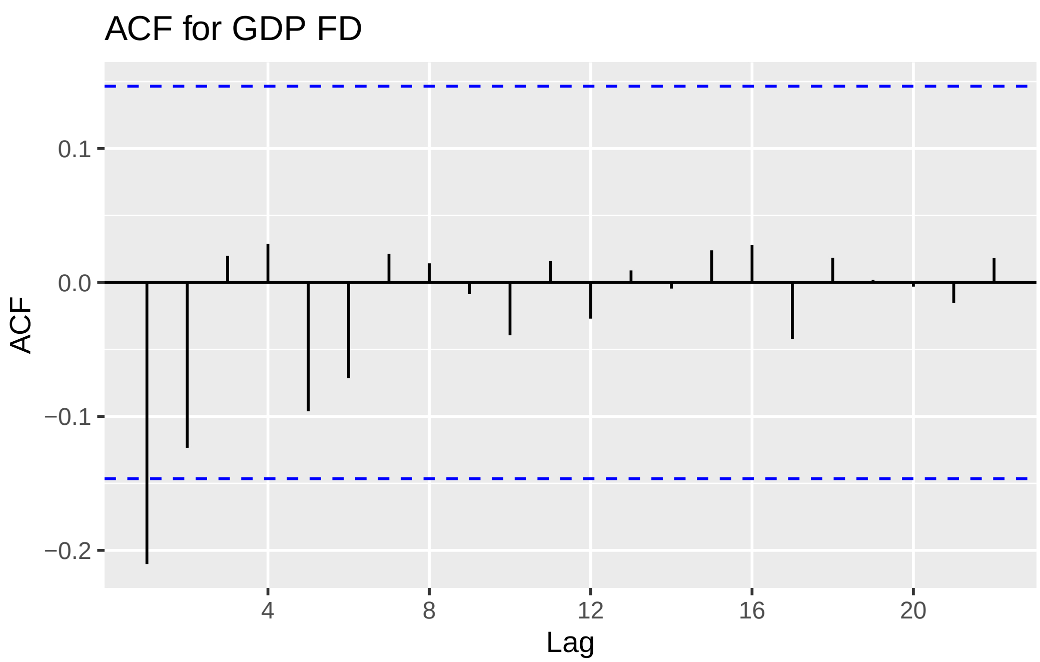
Differenced series appears stationary.

```
summary(GDP_diff)
```

```
    Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
-65605.2     477.4    1766.4    1719.8    3308.9   73161.5 
```

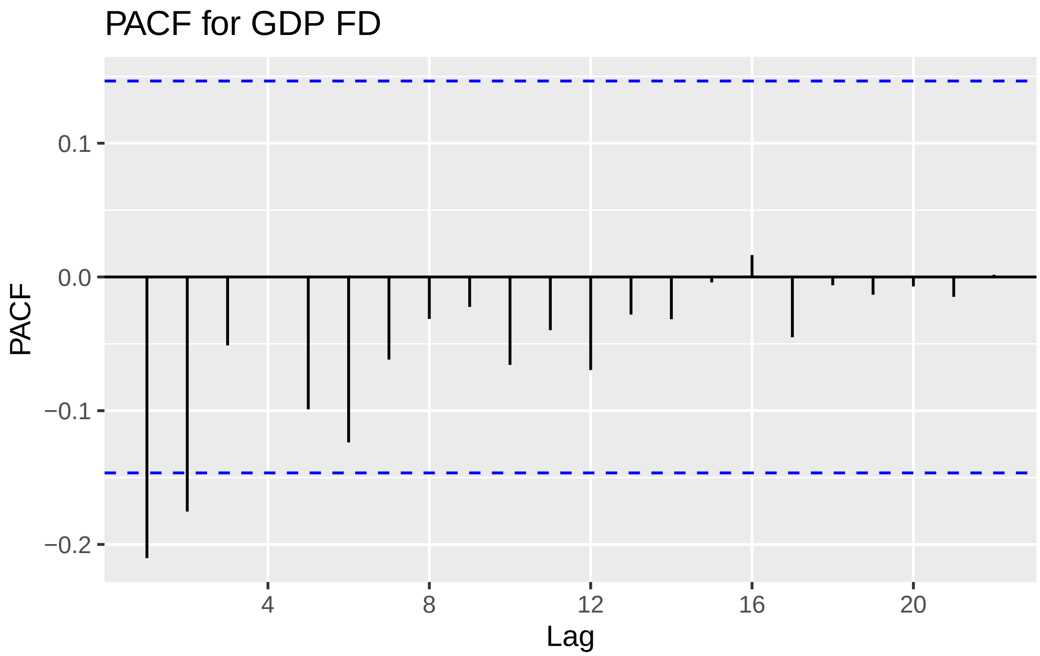
Reduced variance compared to the original series.

```
ggAcf(GDP_diff) + ggtitle("ACF for GDP FD")
```



Quick drop of ACF confirms stationarity.

```
ggPacf(GDP_diff) + ggtitle("PACF for GDP FD")
```



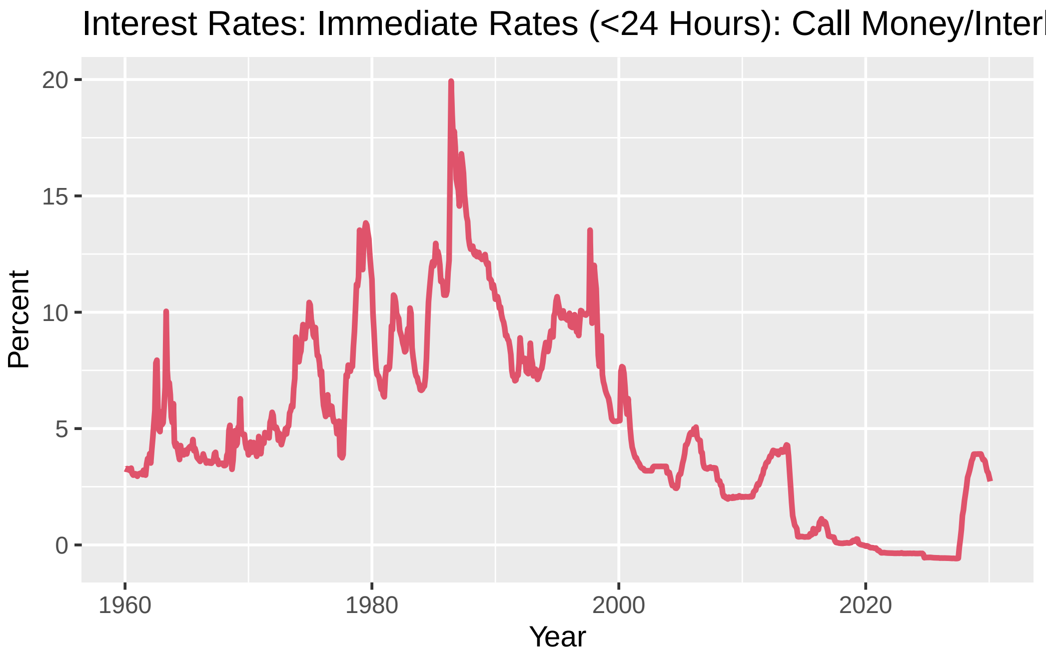
PACF indicates that the differenced series can be modeled as ARMA.

1.1.4: Short-term interest rate

```
interestrate1 <- read.csv("interest rate.csv")
interestrate1 <- rename(interestrate1, rate = IRSTCI01FRM156N)
interestrate <- ts(interestrate1[,2], frequency=12, start=c(1960,1))
str(interestrate)
```

```
Time-Series [1:842] from 1960 to 2030: 3.27 3.25 3.28 3.23 3.27 3.19 3.3 3.06 3 3.02 ...
```

```
autoplot(interestrate, col=2, xlab="Year", ylab="Percent", main="Interest Rates: Immediate Ra
```



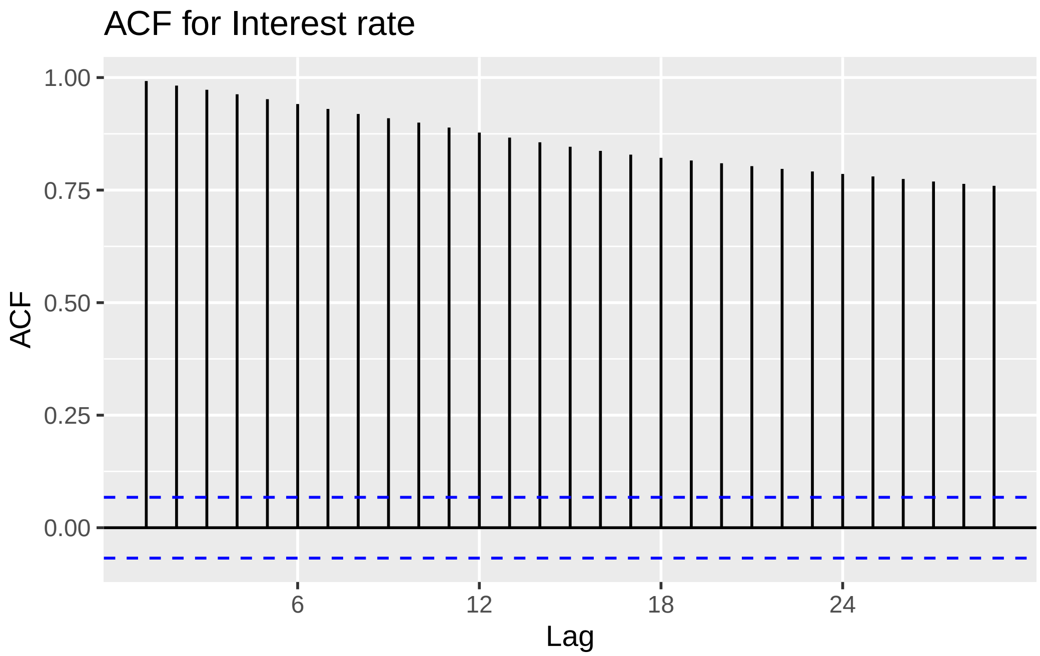
Clear evidence of changing variance and mean, especially high in the 1980s.

```r
summary(interestrate)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-0.5847  2.5525  4.2150  5.1768  8.0199 19.9330 
```

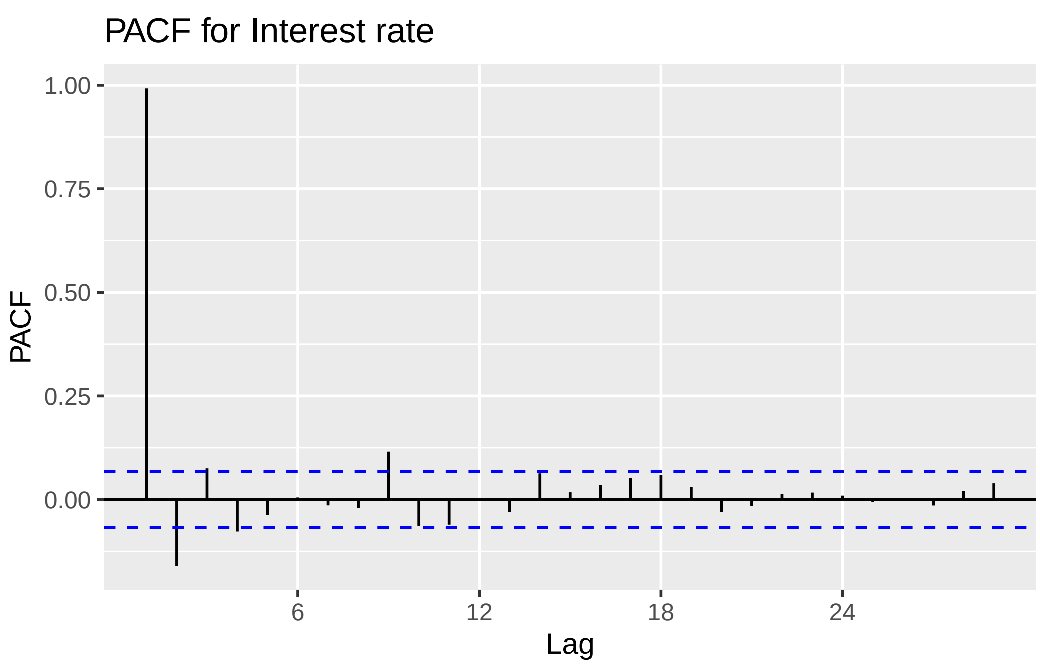
Descriptive statistics confirm large variance.

```r
ggAcf(interestrate) + ggtitle("ACF for Interest rate")
```



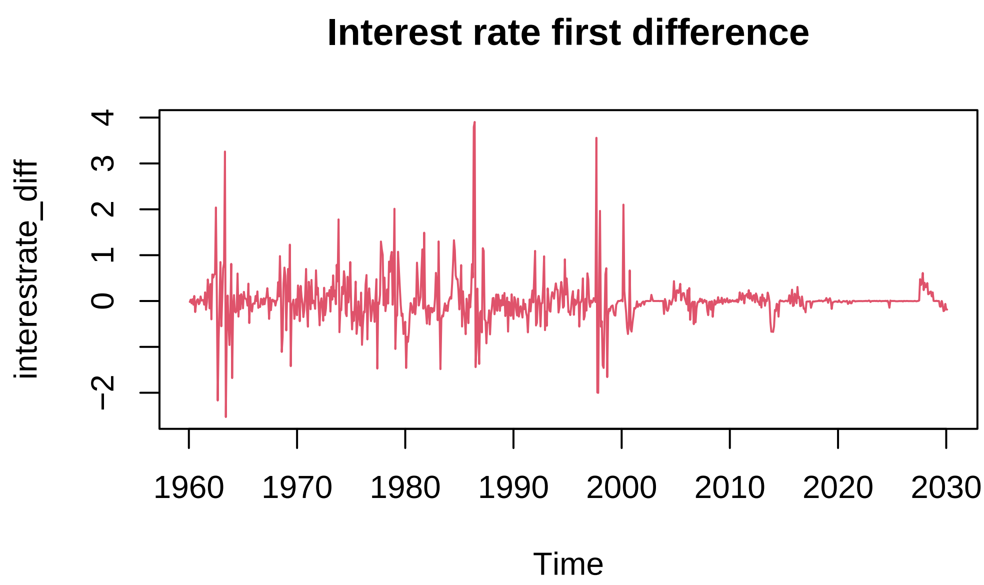
ACF shows slow decay and persistence.

```
ggPacf(interestrate) + ggtitle("PACF for Interest rate")
```



PACF significant in first lags.

```r
interestrate_diff <- diff(interestrate)
ts.plot(interestrate_diff, col=2, main="Interest rate first difference")
```



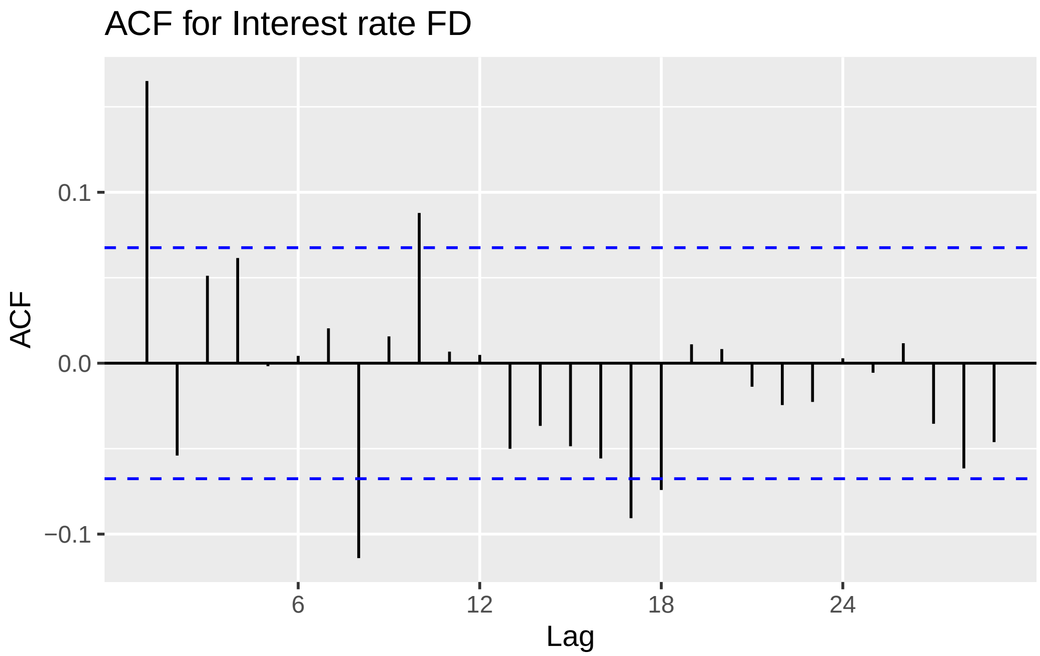
Variance looks more stable after differencing.

```r
summary(interestrate_diff)
```

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
-2.530000 -0.152000 -0.003135 -0.000639  0.085000  3.905000 
```

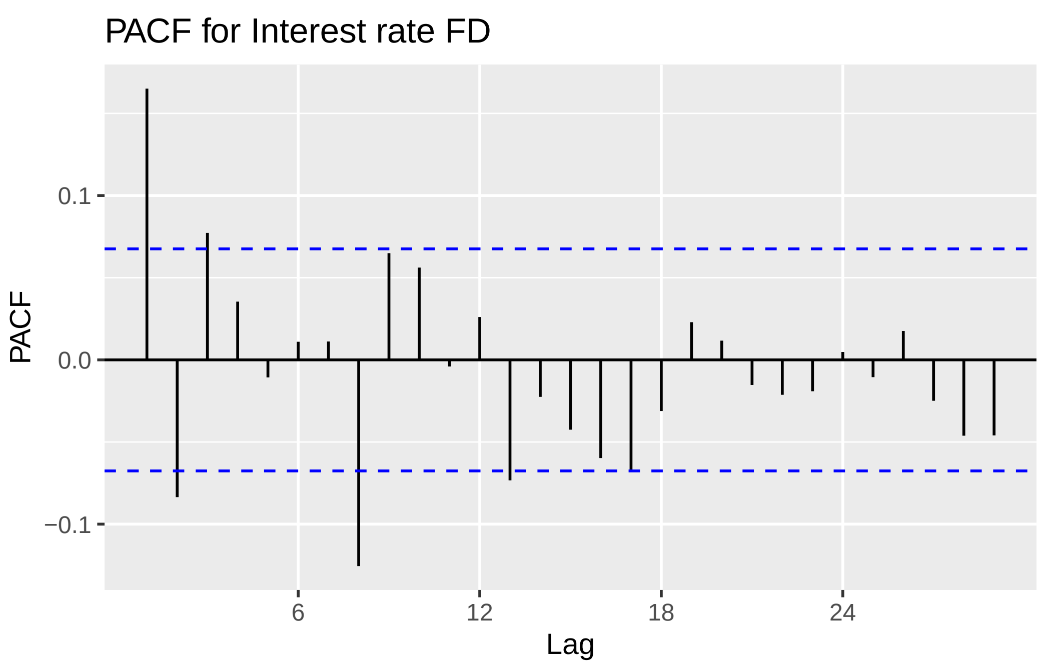
Variance reduced and centered around zero.

```r
ggAcf(interestrate_diff) + ggtitle("ACF for Interest rate FD")
```



ACF decays faster suggesting stationarity.

```
ggPacf(interestrate_diff) + ggtitle("PACF for Interest rate FD")
```



PACF also suggests weak short-term dependencies after differencing.

1.1.5: Daily electricity consumption

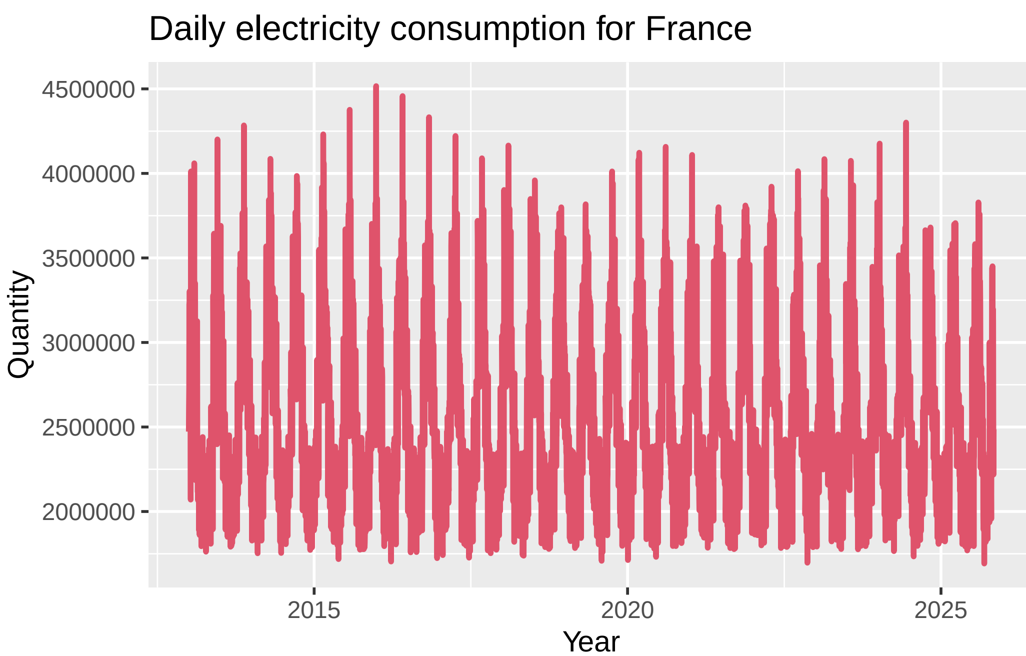
```
elec1 <- read_delim("elec.csv", delim = ";", escape_double = FALSE, trim_ws = TRUE)
```

```
Rows: 225024 Columns: 11
-- Column specification --------------------------------------------------------
Delimiter: ";"
chr  (4): Date, Statut - GRTgaz, Statut - Teréga, Statut - RTE
dbl  (5): Consommation brute gaz (MW PCS 0°C) - GRTgaz, Consommation brute g...
dttm (1): Date - Heure
time (1): Heure

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
elec1 <- elec1[, c("Date", "Consommation brute électricité (MW) - RTE")]
elec1 <- rename(elec1, cons = "Consommation brute électricité (MW) - RTE")
elec1 <- elec1 %>% group_by(Date) %>% summarize(energy_consumed = sum(cons))
elec <- ts(elec1[,2], frequency=365, start=c(2013,1))
str(elec)
```

```
 Time-Series [1:4688, 1] from 2013 to 2026: 2472864 2618244 2732220 3227634 2581914 ...
 - attr(*, "dimnames")=List of 2
  ..$ : NULL
  ..$ : chr "energy_consumed"
```

```
autoplot(elec, col=2, xlab="Year", ylab="Quantity", main="Daily electricity consumption for I
```



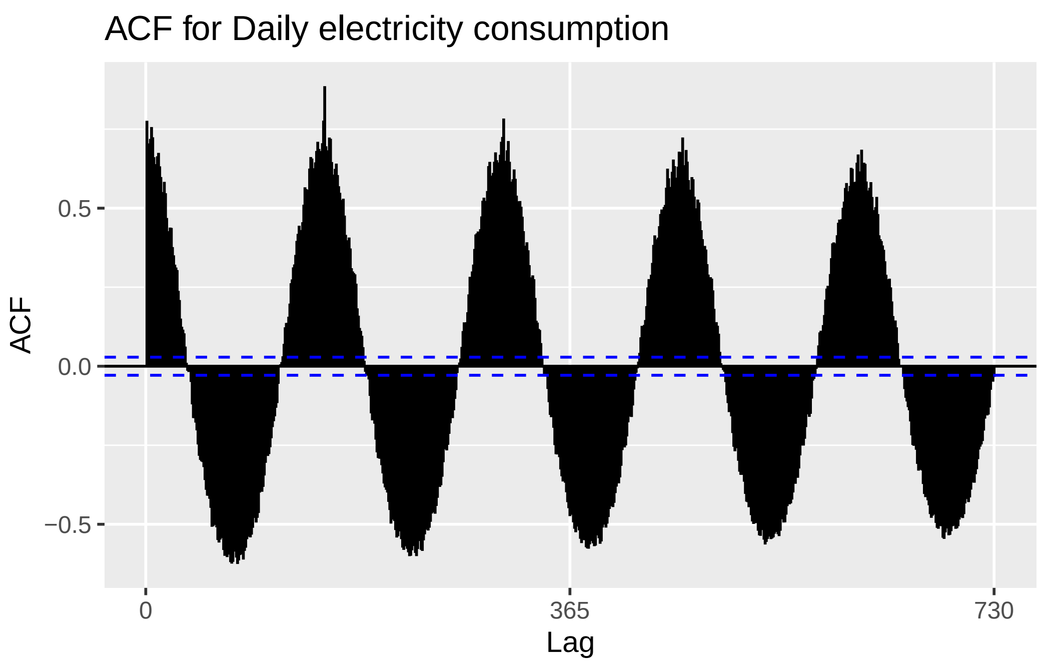
Clear seasonality visible.

```
summary(elec)
```

```
 energy_consumed
 Min.   :1692364
 1st Qu.:2154992
 Median :2378055
 Mean   :2547284
 3rd Qu.:2923860
 Max.   :4516388
```

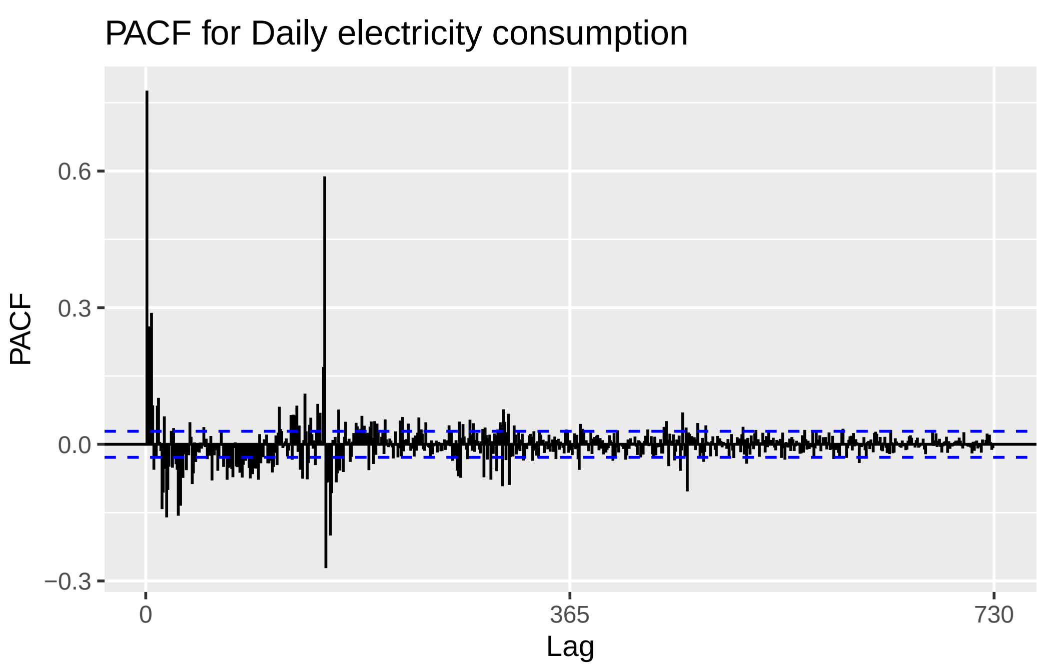
Descriptive statistics show high variability.

```
ggAcf(elec) + ggtitle("ACF for Daily electricity consumption")
```



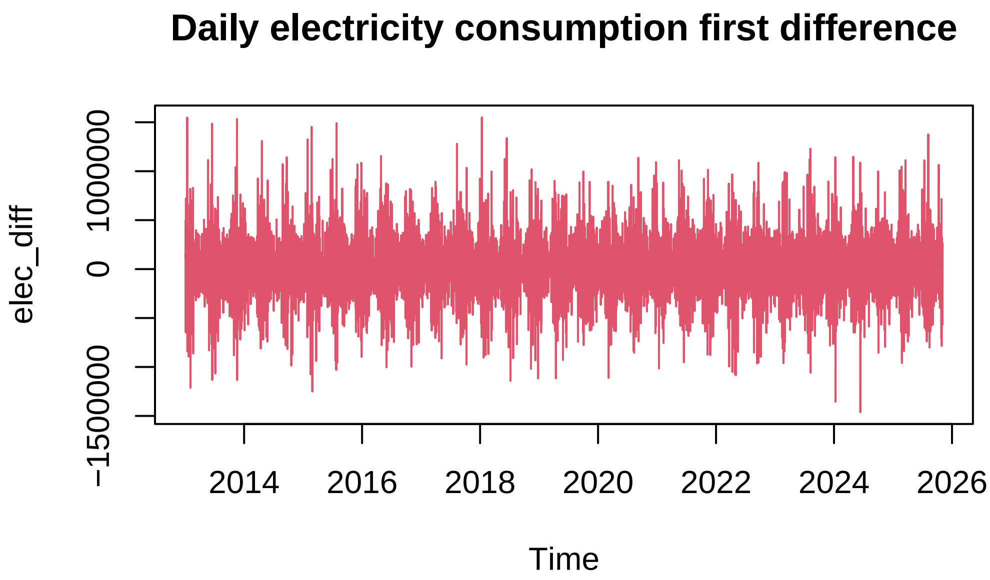
ACF shows strong seasonal autocorrelation.

```
ggPacf(elec) + ggtitle("PACF for Daily electricity consumption")
```



PACF also shows seasonal patterns.

```
elec_diff <- diff(elec)
ts.plot(elec_diff, col=2, main="Daily electricity consumption first difference")
```



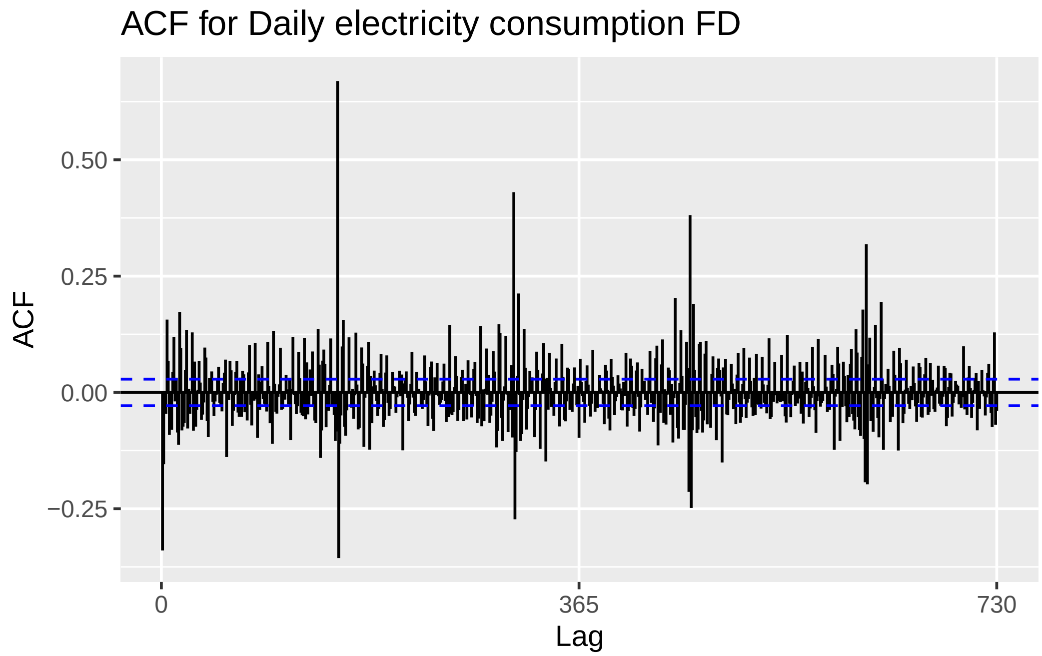
Differenced series looks more stable.

```
summary(elec_diff)
```

```
 energy_consumed
 Min.   :-1462045.0
 1st Qu.: -202042.0
 Median :  -11250.0
 Mean   :       2.7
 3rd Qu.:  196407.5
 Max.   : 1550130.0
```

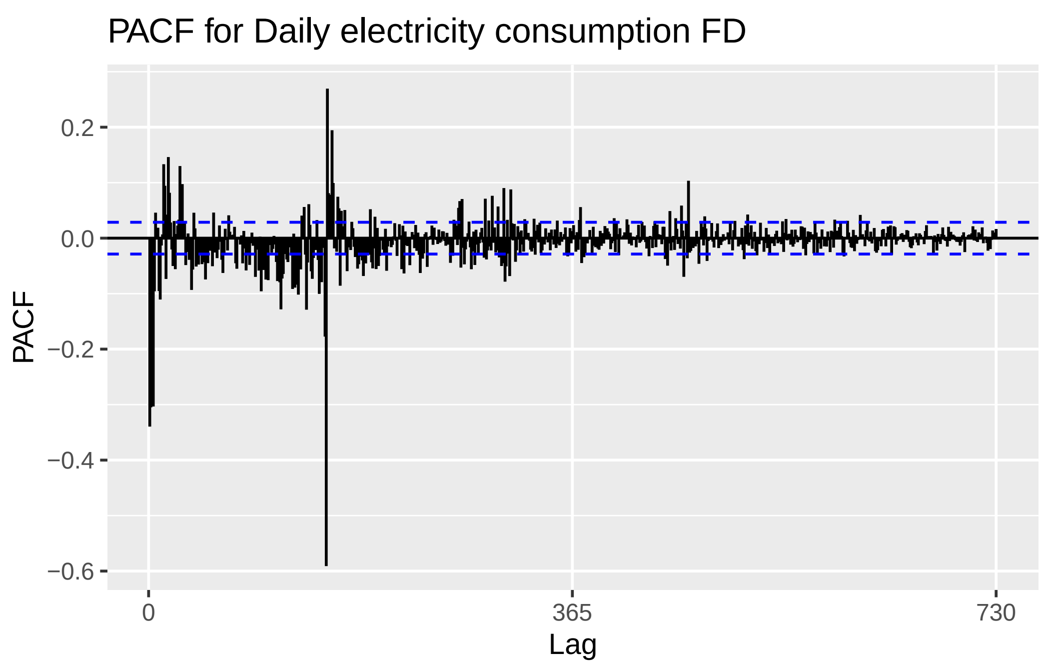
Variance is lower.

```
ggAcf(elec_diff) + ggtitle("ACF for Daily electricity consumption FD")
```



Seasonality still present but weaker.

```r
ggPacf(elec_diff) + ggtitle("PACF for Daily electricity consumption FD")
```



PACF confirms partial removal of seasonal pattern.

1.2: Question 2

1.2.0: Function definition

```r
#function to get best AR(p) by AIC (ar) and BIC (critMatrix)
best_ar_model <- function(ts, name = "", max_lag = 10) {
  #select p by minimizing AIC using ar()
  aic_order <- ar(ts, aic = TRUE, order.max = max_lag)$order

  #select p by minimizing BIC using critMatrix()
  bic_values <- critMatrix(
    ts,
    p.max = max_lag,
    q.max = 0,
    criterion = "bic",
    include.mean = TRUE
  )
  bic_order <- which(bic_values == min(bic_values)) - 1

  cat(name, "\n")
  cat("Best AR(p) according to AIC : AR(",aic_order,")\n")
  cat("Best AR(p) according to BIC : AR(",bic_order,")\n")
}
```

1.2.1: Price Index

```r
best_ar_model(priceindex, name="Price Index", max_lag=10)
```

```
Price Index
Best AR(p) according to AIC : AR( 9 )
Best AR(p) according to BIC : AR( 1 )
```

```r
best_ar_model(priceindex_diff, name="Price Index (diff)", max_lag=10)
```

```
Price Index (diff)
Best AR(p) according to AIC : AR( 2 )
Best AR(p) according to BIC : AR( 0 )
```

For the price index, AIC selects AR(9) while BIC opts for AR(1). After differencing, the preferred orders are A

R(2) and AR(0) respectively.

1.2.2: Inflation

```
best_ar_model(inf, name="Inflation", max_lag=10)
```

```
Inflation
Best AR(p) according to AIC : AR( 1 )
Best AR(p) according to BIC : AR( 1 )
```

```
best_ar_model(inf_diff, name="Inflation (diff)", max_lag=10)
```

```
Inflation (diff)
Best AR(p) according to AIC : AR( 0 )
Best AR(p) according to BIC : AR( 0 )
```

In the case of inflation, both AIC and BIC suggest AR(1) for the level series, and AR(0) for the differenced series.

1.2.3: GDP

```
best_ar_model(GDP, name="GDP", max_lag=10)
```

```
GDP
Best AR(p) according to AIC : AR( 1 )
Best AR(p) according to BIC : AR( 1 )
```

```
best_ar_model(GDP_diff, name="GDP (diff)", max_lag=10)
```

```
GDP (diff)
Best AR(p) according to AIC : AR( 2 )
Best AR(p) according to BIC : AR( 2 )
```

For GDP, AIC and BIC agree on AR(1) for the original series, and AR(2) for the differenced version.

1.2.4: Interest Rate

```python
best_ar_model(interestrate, name="Interest Rate", max_lag=10)
```

```
Interest Rate
Best AR(p) according to AIC : AR( 10 )
Best AR(p) according to BIC : AR( 2 )
```

```python
best_ar_model(interestrate_diff, name="Interest Rate (diff)", max_lag=10)
```

```
Interest Rate (diff)
Best AR(p) according to AIC : AR( 10 )
Best AR(p) according to BIC : AR( 1 )
```

Regarding the interest rate, the preferred orders are AR(10) and AR(2) according to AIC and BIC, respectively. For the differenced data, AIC remains at AR(10) while BIC selects AR(1).

1.2.5: Daily electricity consumption

```python
best_ar_model(elec, name="Electricity Consumption", max_lag=10)
```

```
Electricity Consumption
Best AR(p) according to AIC : AR( 10 )
Best AR(p) according to BIC : AR( 10 )
```

```python
best_ar_model(elec_diff, name="Electricity Consumption (diff)", max_lag=10)
```

```
Electricity Consumption (diff)
Best AR(p) according to AIC : AR( 10 )
Best AR(p) according to BIC : AR( 10 )
```

For both the original and differenced electricity consumption series, AIC and BIC consistently choose AR(10).

It is important to note that for non-stationary processes (see the ones identified as such above) the assumptions underlying this choice are broken, hence these results may be considered valid only for the stationary processes.

1.3: Question 3

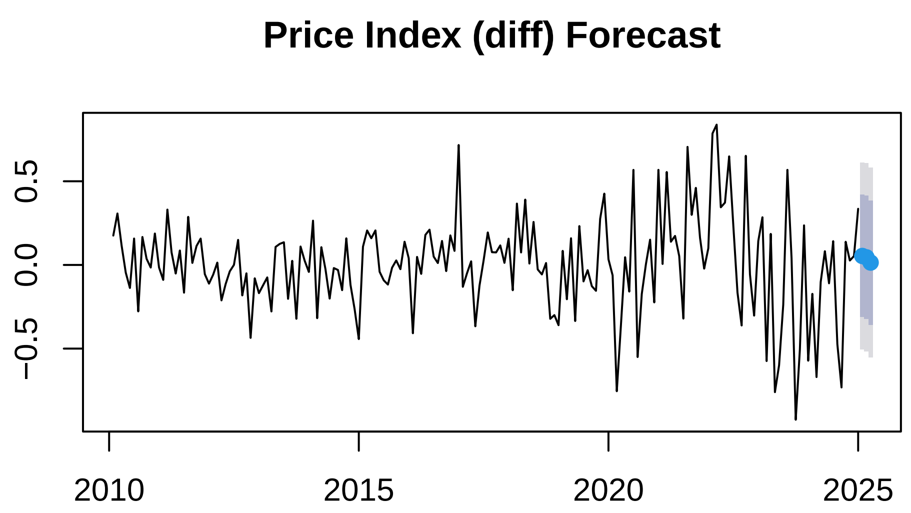
1.3.1: Price Index

```
#estimate AR(2) model on differenced price index
model_priceindex_diff <- ar(priceindex_diff, order.max=2)
#forecast with h=3
fc_priceindex_diff <- forecast::forecast(model_priceindex_diff, h=3)
fc_priceindex_diff
```

```
         Point Forecast      Lo 80     Hi 80      Lo 95     Hi 95
Feb 2025     0.05352999 -0.3112300 0.4182900 -0.5043223 0.6113823
Mar 2025     0.04523119 -0.3228791 0.4133415 -0.5177449 0.6082073
Apr 2025     0.01414880 -0.3567609 0.3850585 -0.5531087 0.5814063
```

The forecast shows moderate uncertainty with increasing variance as the forecast horizon grows.

```
#plot forecast
plot(fc_priceindex_diff, main="Price Index (diff) Forecast")
```



Forecast suggests that the differenced price index will stay close to recent values, with wider intervals after 3 periods.

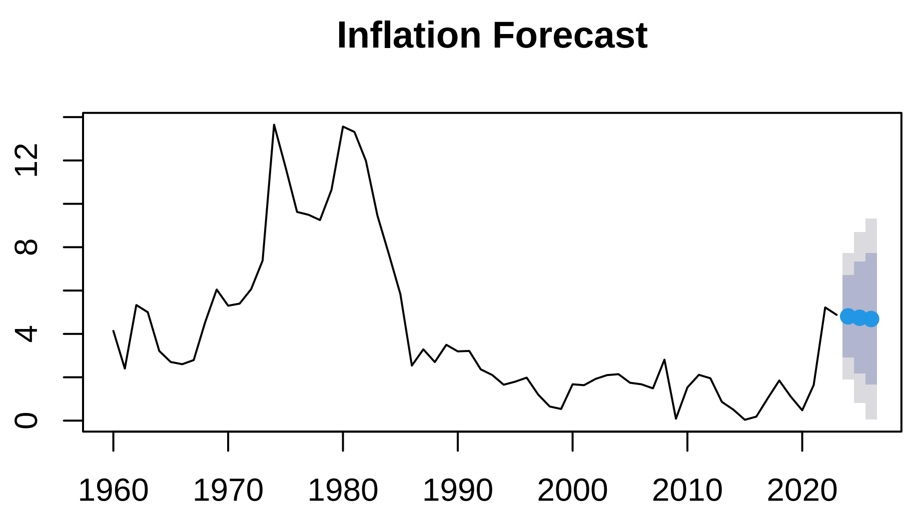
1.3.2: Inflation

```
#estimate AR(1) model on inflation (non differenced)
model_inflation <- ar(inf, order.max=1)
#forecast with h=3
fc_inflation <- forecast::forecast(model_inflation, h=3)
fc_inflation
```

```
     Point Forecast     Lo 80    Hi 80      Lo 95    Hi 95
2024       4.809005 2.907997 6.710014 1.90166434 7.716346
2025       4.745605 2.169948 7.321262 0.80647756 8.684733
2026       4.687646 1.661418 7.713874 0.05943002 9.315862
```

The inflation forecast seems relatively stable with narrow confidence bands.

```
#plot forecast
plot(fc_inflation, main="Inflation Forecast")
```



Graph indicates inflation is predicted to follow its recent trend with moderate fluctuations.

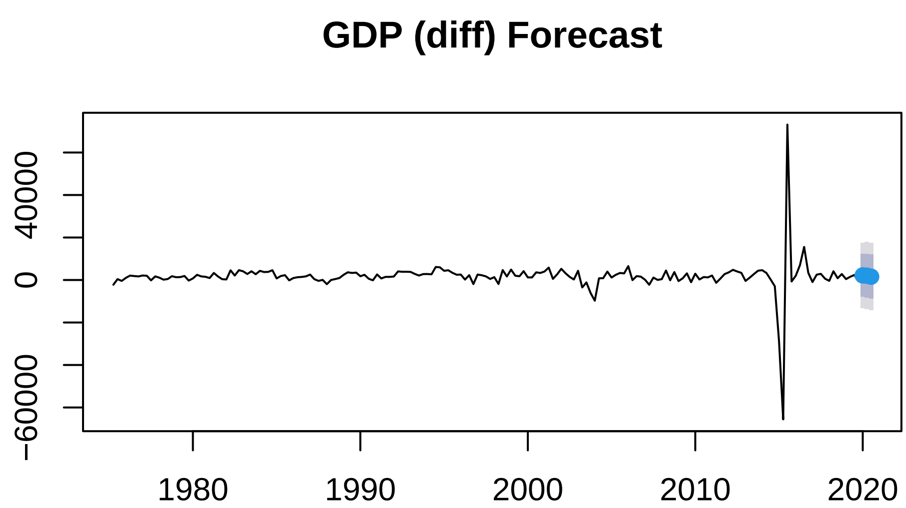
1.3.3: GDP

```
#estimate AR(2) model on differenced GDP
model_GDP_diff <- ar(GDP_diff, order.max=2)
#forecast with h=3
fc_GDP_diff <- forecast::forecast(model_GDP_diff, h=3)
fc_GDP_diff
```

```
        Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
2020 Q1       2175.805 -7824.123 12175.73 -13117.77 17469.37
2020 Q2       2012.665 -8287.982 12313.31 -13740.81 17766.15
2020 Q3       1567.516 -8796.375 11931.41 -14282.69 17417.72
```

Forecast indicates moderate variability over the next 3 periods.

```
#plot forecast
plot(fc_GDP_diff, main="GDP (diff) Forecast")
```



GDP difference forecast shows stable evolution without pronounced trend changes.

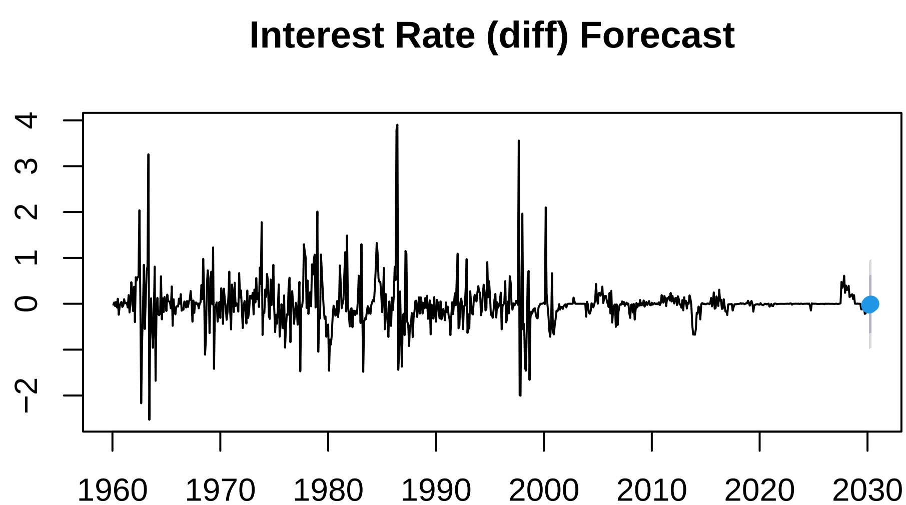
1.3.4: Interest Rate

```r
#estimate AR(1) model on differenced interest rate
model_rate_diff <- ar(interestrate_diff, order.max=1)
#forecast with h=3
fc_rate_diff <- forecast::forecast(model_rate_diff, h=3)
fc_rate_diff
```

```
         Point Forecast      Lo 80     Hi 80      Lo 95     Hi 95
Mar 2030   -0.031319884 -0.6517202 0.5890804 -0.9801403 0.9175005
Apr 2030   -0.005703993 -0.6345007 0.6230927 -0.9673655 0.9559575
May 2030   -0.001475373 -0.6304993 0.6275485 -0.9634844 0.9605337
```

Short-term interest rates forecasts show moderate prediction intervals.

```r
#plot forecast
plot(fc_rate_diff, main="Interest Rate (diff) Forecast")
```



Interest rate forecast indicates low volatility in upcoming periods.

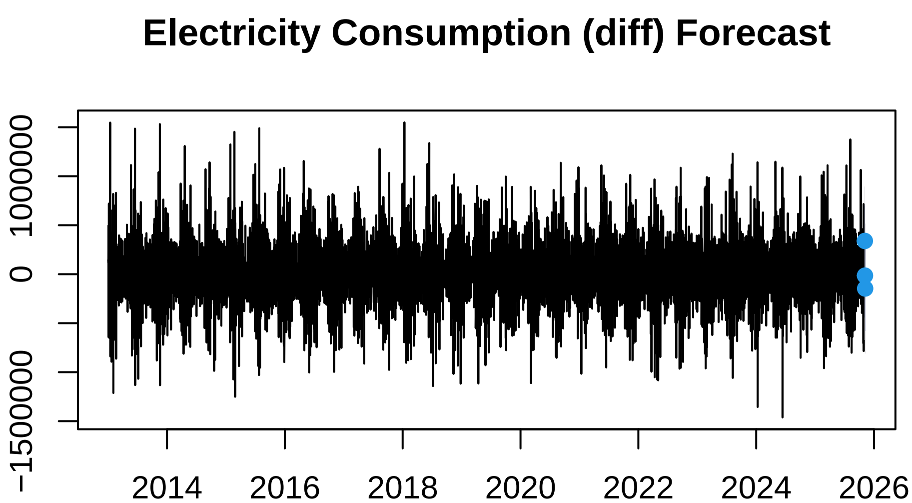
1.3.5: Electricity Consumption

```
#estimate AR(10) model on differenced electricity consumption
model_elec_diff <- ar(elec_diff, order.max=10)
#forecast with h=3
fc_elec_diff <- forecast::forecast(model_elec_diff, h=3)
fc_elec_diff
```

```
          Point Forecast      Lo 80    Hi 80     Lo 95    Hi 95
2025.8438      339436.41  -21267.09 700139.9 -212212.0 891084.8
2025.8466      -14534.74 -443011.48 413942.0 -669833.4 640763.9
2025.8493     -146029.63 -578752.12 286692.9 -807821.6 515762.3
```

Electricity consumption forecast shows higher uncertainty, probably due to seasonality and structural changes.

```
#plot forecast
plot(fc_elec_diff, main="Electricity Consumption (diff) Forecast")
```



The plot highlights the widening prediction interval, reflecting the variability of electricity consumption.

Problem 2

2.1: Question 1

2.1.1: Inflation

```
#load the dataset
df <- read.csv("inflation_us.csv")

#we just use the year
df$Date <- as.Date(paste0(df$Year, "-01-01"))

#create the time series (annual)
inf <- ts(df$Value, start=c(min(df$Year)), frequency=1)

#check structure
str(inf)
```

```
 Time-Series [1:67] from 1958 to 2024: 2.4 2 1.3 1.3 1.3 1.3 1.6 1.2 2.4 3.6 ...
```

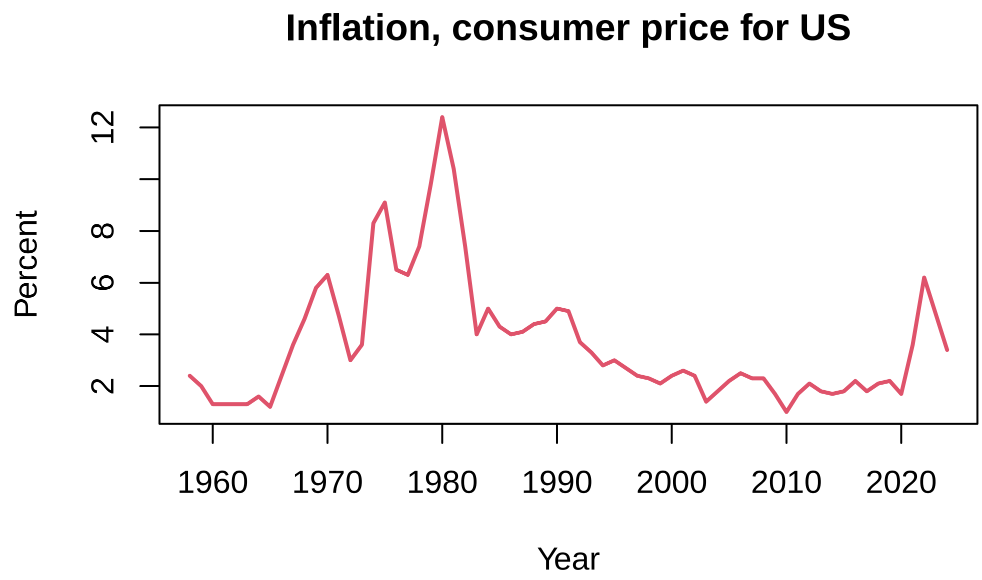
The dataset contains annual observations of the US inflation index. Each entry represents the value at the end of the year.

```
#compute descriptive statistics
summary(inf)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.050   2.700   3.675   4.650  12.400
```

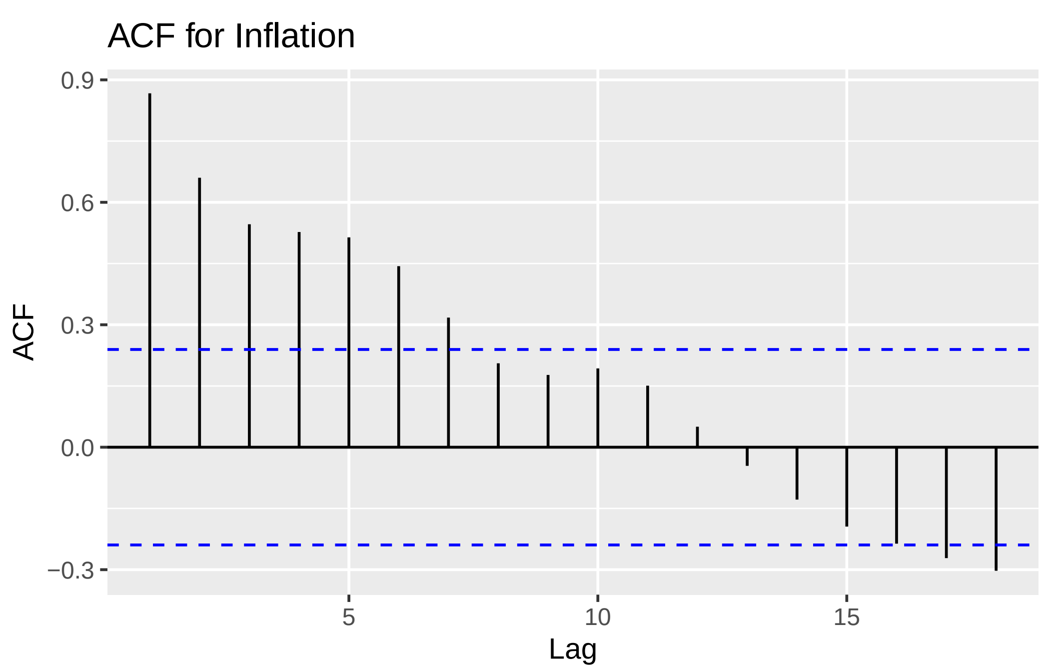
The inflation series has an average close to 3.67% and shows some extreme values, with a minimum of 1% and a maximum of 12.4%.The inflation does not look like a stationary process, since the mean, nor the variance look to be constant over the period considered.

```
#plot the time series
plot(inf, main="Inflation, consumer price for US", col=2, ylab="Percent", xlab="Year", lwd=2)
```



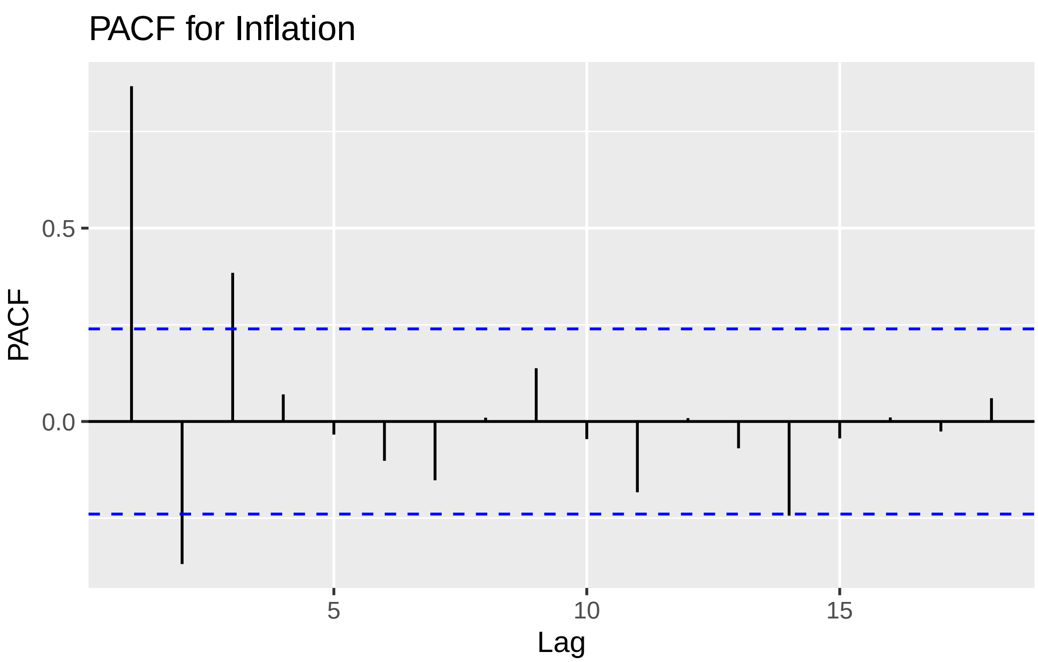
The graph highlights strong variations, particularly during the 1970s and early 1980s.

```
#plot ACF
ggAcf(inf) + ggtitle("ACF for Inflation")
```



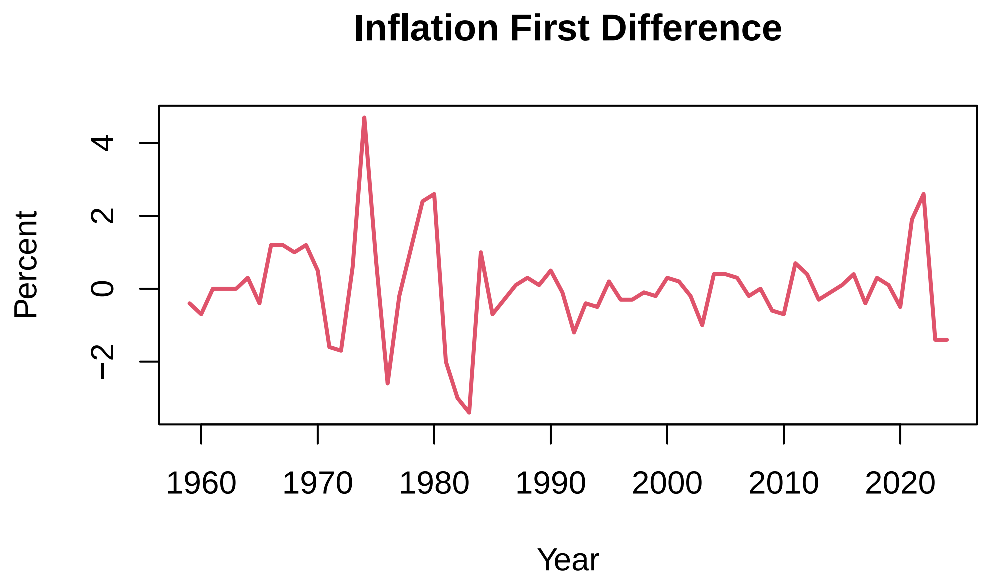
The ACF shows slowly decaying autocorrelations, indicating non-stationarity.We can see high persistence from the ACF plot, however over multiple years, it falls to zero.

```
#plot PACF
ggPacf(inf) + ggtitle("PACF for Inflation")
```



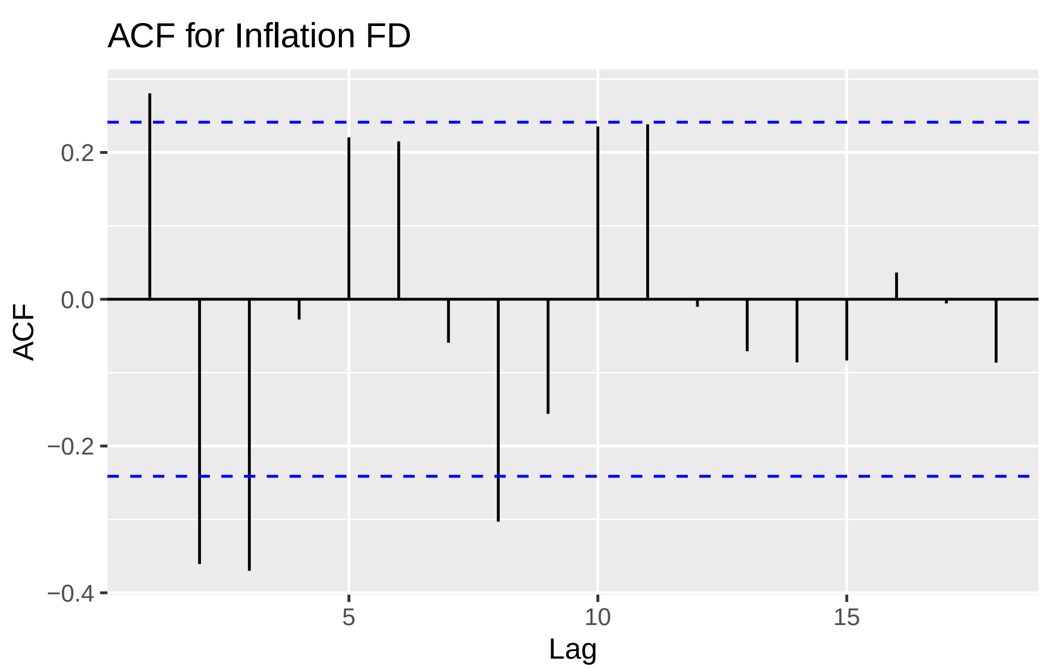
The PACF presents significant lags suggesting long-term dependencies.

```
#create the first difference
inf_diff <- diff(inf)

#plot the first difference
ts.plot(inf_diff, col=2, xlab="Year", ylab="Percent", main="Inflation First Difference", lwd=
```



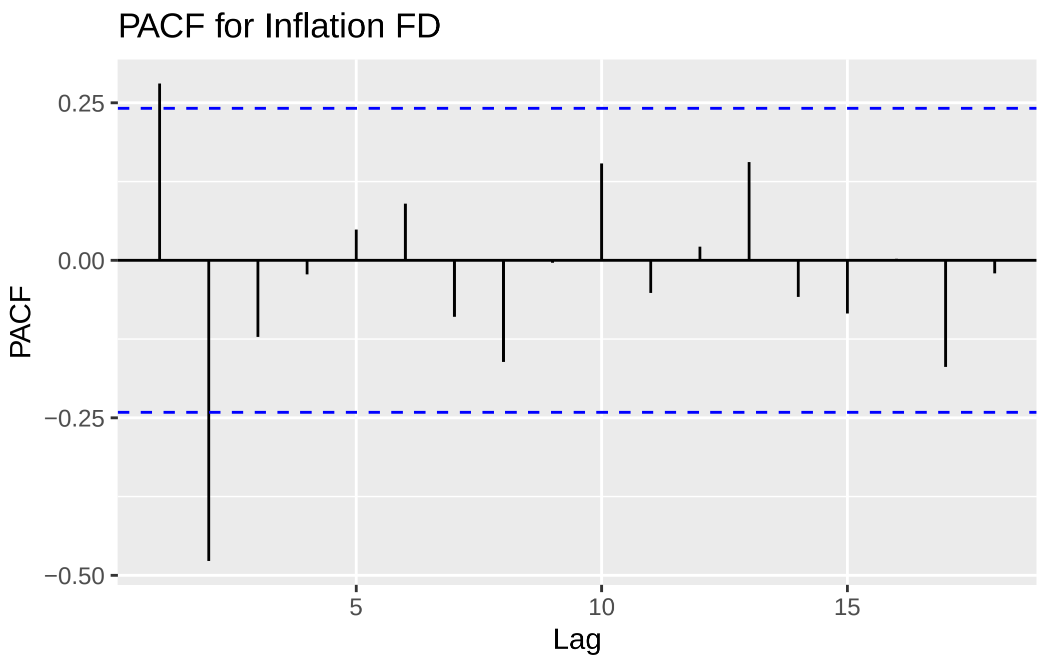
The differenced series seems more stable, suggesting a stationary process.

```
#plot ACF of first difference
ggAcf(inf_diff) + ggtitle("ACF for Inflation FD")
```



The ACF of the first difference decreases rapidly.

```
#plot PACF of first difference
ggPacf(inf_diff) + ggtitle("PACF for Inflation FD")
```



The PACF also drops after the first lags, indicating short memory after differencing.

The first difference appears to be a stationary process. The variance seems stable based on the plots, and the autocovariances are generally not statistically significant, except perhaps for the second lag. This pattern suggests the presence of a possible unit root, which motivates a formal statistical test.

```
#unit root tests
type1id = ur.df(inf_diff, type = "none", selectlags = "AIC")
summary(type1id)
```

```
###############################################
# 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
-3.1313 -0.4678 -0.0133  0.4944  3.6230

Coefficients:
           Estimate Std. Error t value Pr(>|t|)
z.lag.1     -1.0729     0.1357  -7.909 5.69e-11 ***
z.diff.lag   0.4873     0.1128   4.321 5.71e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.077 on 62 degrees of freedom
Multiple R-squared:  0.5038,    Adjusted R-squared:  0.4878
F-statistic: 31.47 on 2 and 62 DF,  p-value: 3.682e-10


Value of test-statistic is: -7.9091

Critical values for test statistics:
     1pct  5pct 10pct
tau1 -2.6 -1.95 -1.61
```

```r
type2id = ur.df(inf_diff, type = "drift", selectlags = "AIC")
summary(type2id)
```

```
###############################################
# 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
```

```
-3.1805 -0.5118 -0.0573  0.4527  3.5773

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.04374    0.13583   0.322    0.749
z.lag.1     -1.07516    0.13683  -7.858 7.73e-11 ***
z.diff.lag   0.48872    0.11367   4.299 6.28e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.085 on 61 degrees of freedom
Multiple R-squared:  0.5046,    Adjusted R-squared:  0.4883
F-statistic: 31.07 on 2 and 61 DF,  p-value: 4.971e-10


Value of test-statistic is: -7.8578 30.8763

Critical values for test statistics:
      1pct  5pct 10pct
tau2 -3.51 -2.89 -2.58
phi1  6.70  4.71  3.86
```

```r
type3id = ur.df(inf_diff, type="trend", selectlags="AIC")
summary(type3id)
```

```
###############################################
# 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
-3.2068 -0.4753 -0.0756  0.4167  3.5288

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept)  0.136986   0.283020   0.484    0.630
z.lag.1     -1.074653   0.137807  -7.798 1.08e-10 ***
tt          -0.002785   0.007399  -0.376    0.708
z.diff.lag   0.487294   0.114541   4.254 7.46e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.093 on 60 degrees of freedom
Multiple R-squared:  0.5058,    Adjusted R-squared:  0.481
F-statistic: 20.47 on 3 and 60 DF,  p-value: 2.974e-09


Value of test-statistic is: -7.7983 20.3418 30.509

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
```

Regardless of the specification used, all tests indicate that the series is stationary. Therefore, we can conclude that the series does not contain a unit root and is suitable for further analysis.

```r
#arma model selection
mean_equ <- auto.arima(inf_diff)
summary(mean_equ)
```

```
Series: inf_diff
ARIMA(2,0,0) with zero mean

Coefficients:
         ar1      ar2
      0.4104  -0.4751
s.e.  0.1081   0.1073

sigma^2 = 1.13:  log likelihood = -96.97
AIC=199.95   AICc=200.34   BIC=206.52

Training set error measures:
                     ME     RMSE       MAE MPE MAPE      MASE        ACF1
Training set 0.02879151 1.046958 0.7234738 NaN  Inf 0.7734506 -0.05502691
```

The automatic selection suggests an ARMA(2,0) model.

```
#fit AR(2) model
arma_inf <- arima(inf_diff, order=c(2,0,0), method="ML")
summary(arma_inf)
```

```

Call:
arima(x = inf_diff, order = c(2, 0, 0), method = "ML")

Coefficients:
         ar1      ar2  intercept
      0.4098  -0.4766     0.0316
s.e.  0.1080   0.1074     0.1218

sigma^2 estimated as 1.095:  log likelihood = -96.94,  aic = 201.88

Training set error measures:
                       ME     RMSE       MAE MPE MAPE      MASE         ACF1
Training set -0.004447082 1.046401 0.7263794 NaN  Inf 0.7765569 -0.05480434
```

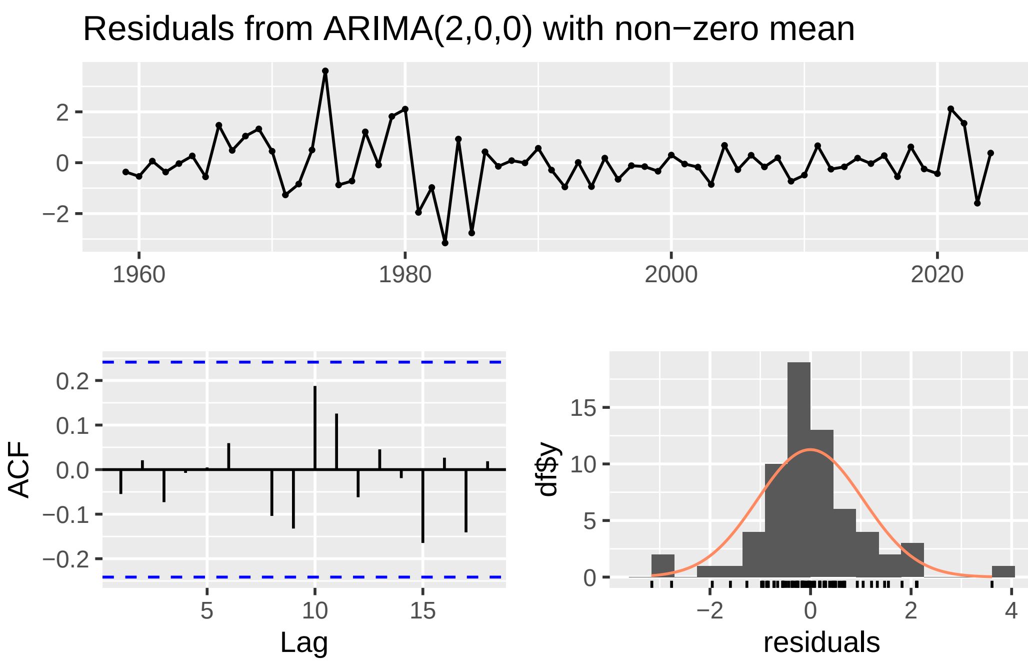
The AR(2) model estimates ar1=0.41 and ar2=-0.47, both statistically significant.

```
#check residuals
Box.test(arma_inf$residuals, lag=10, type="Ljung-Box")
```

```

    Box-Ljung test

data:  arma_inf$residuals
X-squared = 5.9252, df = 10, p-value = 0.8215
```

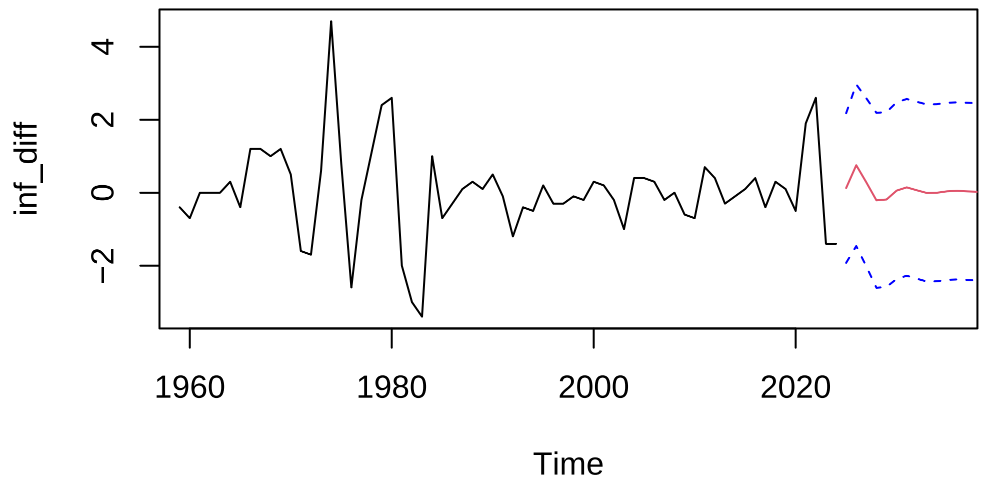
The null hypothesis is not rejected, suggesting the series shows no significant serial correlation, allowing us to proceed with forecasting.

```
#plot diagnostic
#residuals analysis
checkresiduals(arma_inf)
```




```
    Ljung-Box test

data:  Residuals from ARIMA(2,0,0) with non-zero mean
Q* = 5.9252, df = 8, p-value = 0.6556

Model df: 2.   Total lags used: 10
```

```
#forecast
n_steps <- 50
forecast_values <- predict(arma_inf, n.ahead=n_steps)$pred
forecast_values_se <- predict(arma_inf, n.ahead=n_steps)$se

#plot forecast with confidence interval
ts.plot(inf_diff, xlim=c(1960,2035))
points(forecast_values, type='l', col=2)
lines(forecast_values + 1.96*forecast_values_se, col='blue', lty=2)
lines(forecast_values - 1.96*forecast_values_se, col='blue', lty=2)
```



The forecast suggests a moderate continuation of the past dynamics with confidence bands widening progressively.

2.1.2: GDP

```
#load the dataset
gdp_df <- read.csv("gdp_usa.csv")

GDP1 <-rename(gdp_df,gdp = GDPC1)
gdp <- ts(GDP1[,2], frequency = 4, start = c(1947, 1)) # Quaterly
str(gdp)
```

```
 Time-Series [1:312] from 1947 to 2025: 2183 2177 2172 2206 2240 ...
```

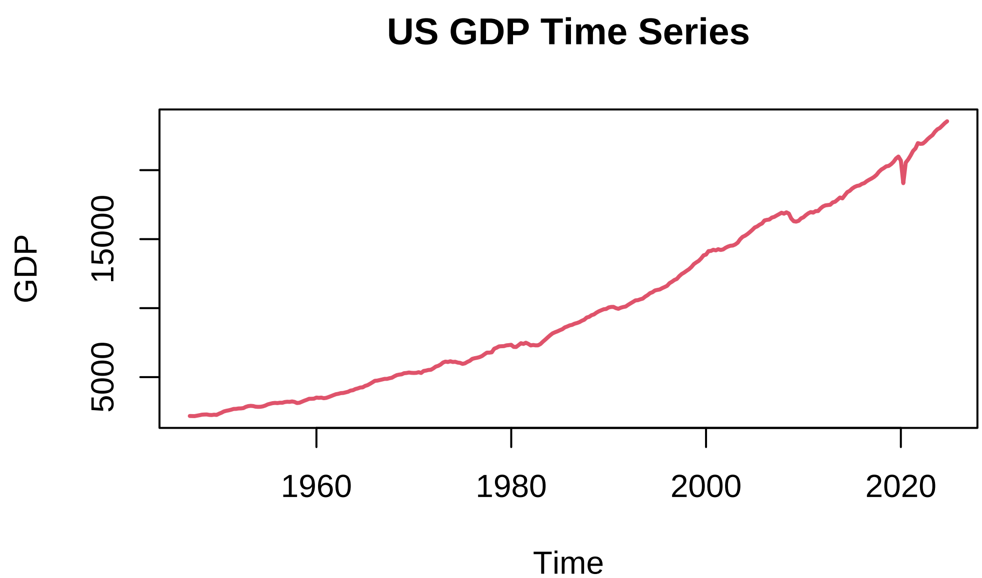
The dataset contains US annual GDP series. Values are on a large scale and exhibit a clear upward trend.

```
#descriptive statistics
summary(gdp)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   2172    4778    8709   10148   15954   23542 
```

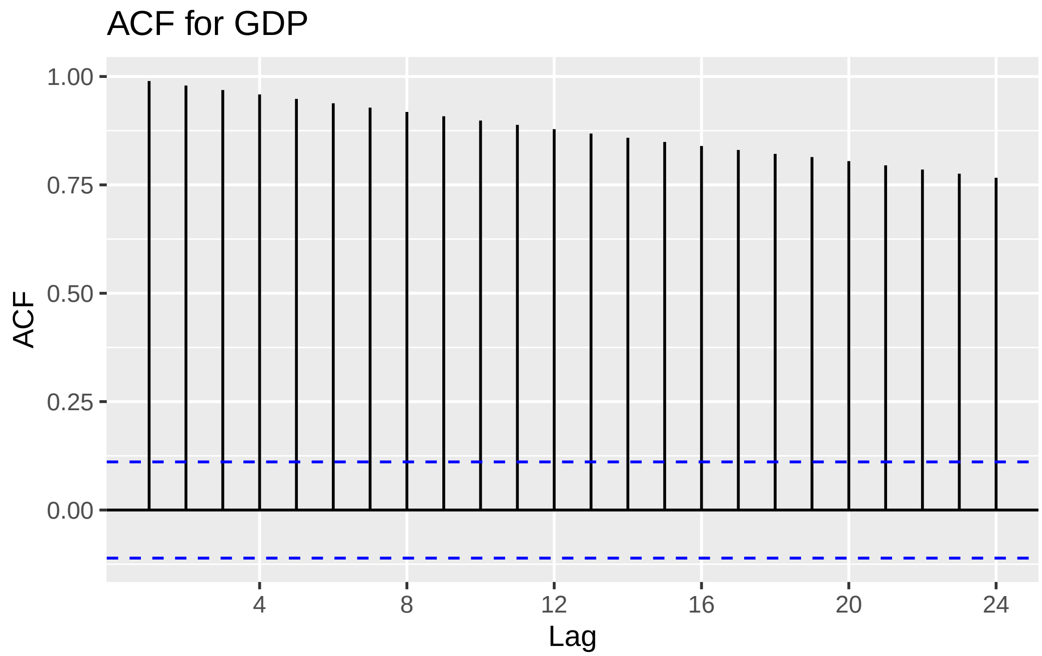
The series has a mean of approximately 10148 and a maximum reaching over 23500.

```
#plot the time series
plot(gdp, main="US GDP Time Series", col=2, ylab="GDP", xlab="Time", lwd=2)
```

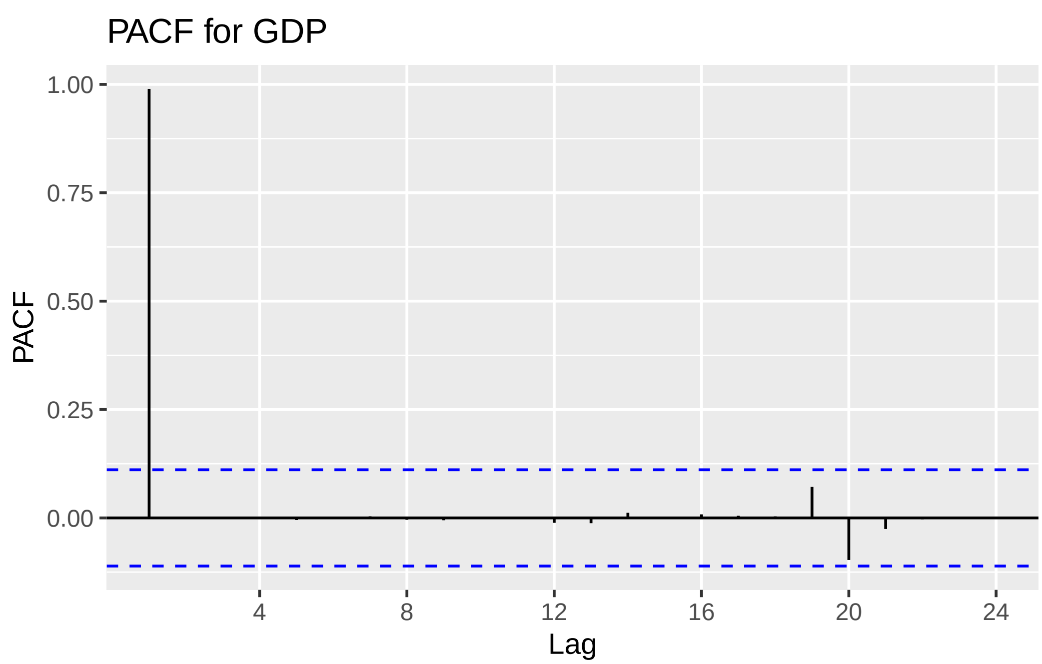


The graph shows strong economic growth over the period.We see That the GDP has an increasing trend with some variations and after running this code, we clearly see that GDP is not stationary

```
#acf and pacf
ggAcf(gdp) + ggtitle("ACF for GDP")
```

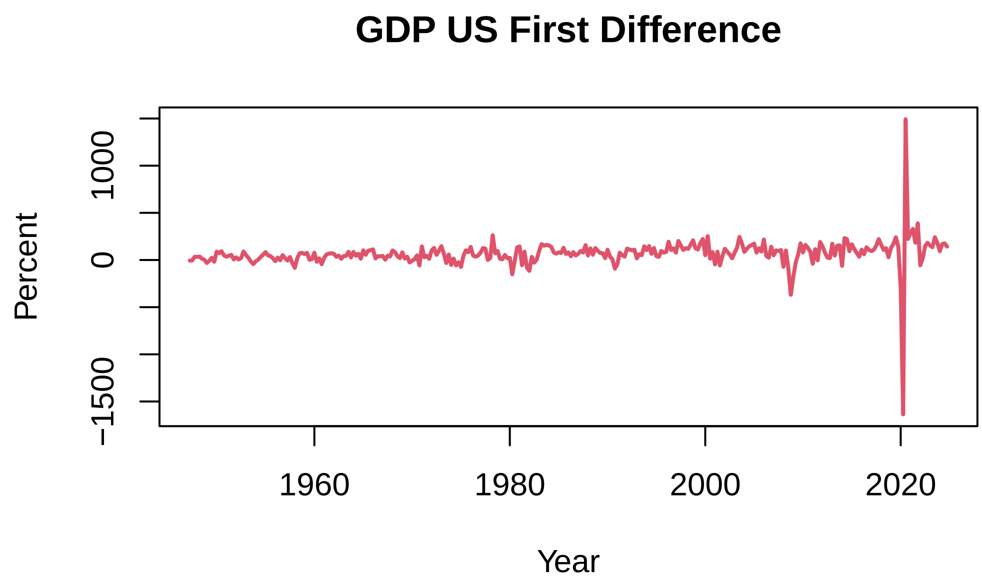



```r
ggPacf(gdp) + ggtitle("PACF for GDP")
```

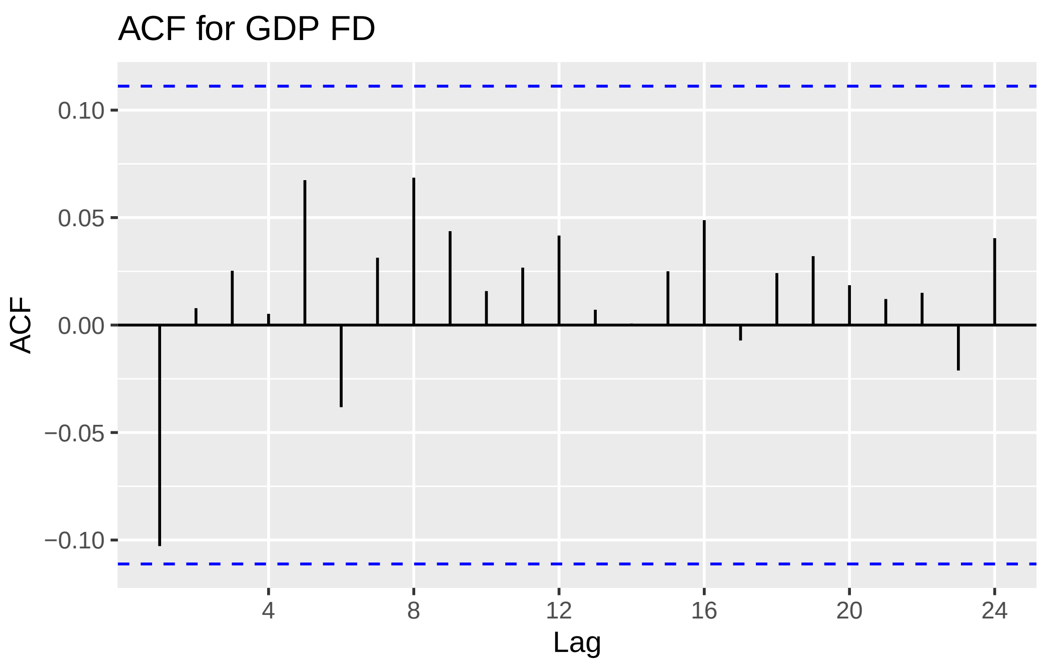


Both ACF and PACF suggest non-stationarity, with slow decay in autocorrelations.The ACF takes a lot of periods to go back to 0, more than fit in the graph.

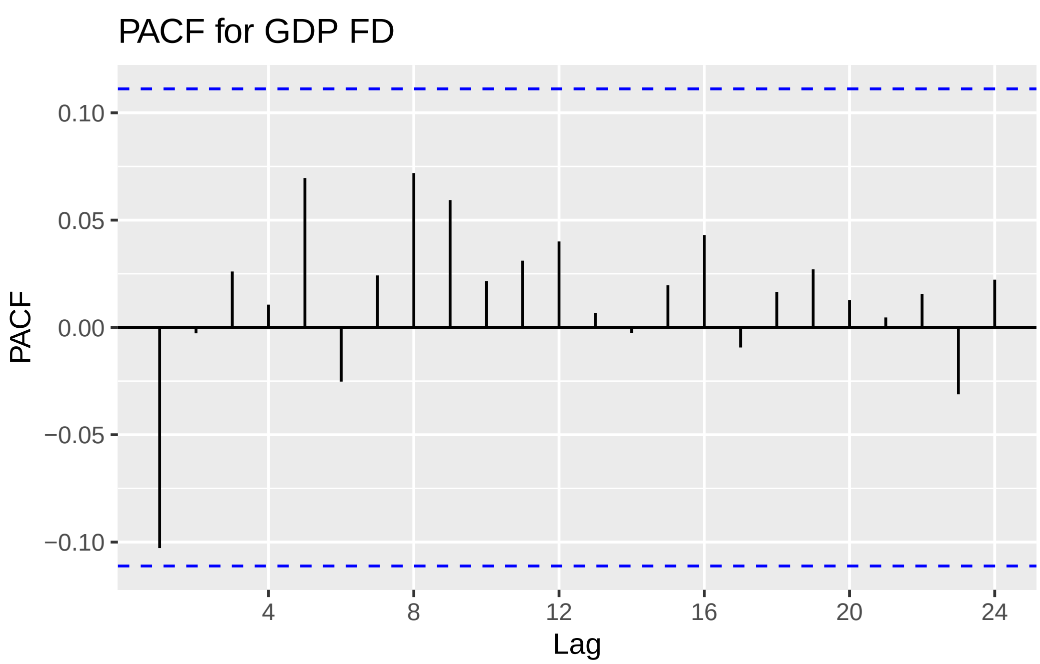
```r
#first difference
gdp_diff <- diff(gdp)
ts.plot(gdp_diff, col=2, xlab="Year", ylab="Percent", main="GDP US First Difference", lwd=2)
```




```r
#acf and pacf of the diff
ggAcf(gdp_diff) + ggtitle("ACF for GDP FD")
```




```
ggPacf(gdp_diff) + ggtitle("PACF for GDP FD")
```



GDP first difference seems to be more like a stationary time series, with the exception of the last period, in which the variance is clearly higher. We don't observe any seasonality from the first differences. So we will use the first difference data for the rest of our analysis.

```
#ADF tests
ur.df(gdp_diff, type="none", selectlags="AIC")
```

```
###############################################################
# Augmented Dickey-Fuller Test Unit Root / Cointegration Test #
###############################################################

The value of the test statistic is: -9.9103
```

```
ur.df(gdp_diff, type="drift", selectlags="AIC")
```

```
###############################################################
# Augmented Dickey-Fuller Test Unit Root / Cointegration Test #
###############################################################

The value of the test statistic is: -13.0279 84.8677
```

```
ur.df(gdp_diff, type="trend", selectlags="AIC")
```

```
###############################################################
# Augmented Dickey-Fuller Test Unit Root / Cointegration Test #
###############################################################

The value of the test statistic is: -13.9925 65.2666 97.8953
```

ADF tests confirm stationarity after differencing with test statistics well below critical values.

```
#arma selection
mean_equ_gdp <- auto.arima(gdp_diff)
summary(mean_equ_gdp)
```

```
Series: gdp_diff
ARIMA(3,1,1)(1,0,0)[4]
```

```
Coefficients:
          ar1      ar2      ar3      ma1     sar1
      -0.1546  -0.0604  -0.0303  -0.9703  -0.0397
s.e.   0.0583   0.0600   0.0596   0.0148   0.0599

sigma^2 = 22282:  log likelihood = -1990.82
AIC=3993.65   AICc=3993.92   BIC=4016.06

Training set error measures:
                   ME     RMSE      MAE       MPE     MAPE      MASE
Training set 13.67444 147.8259 69.01351 -39.22516 186.3302 0.7167176
                    ACF1
Training set -0.01199722
```

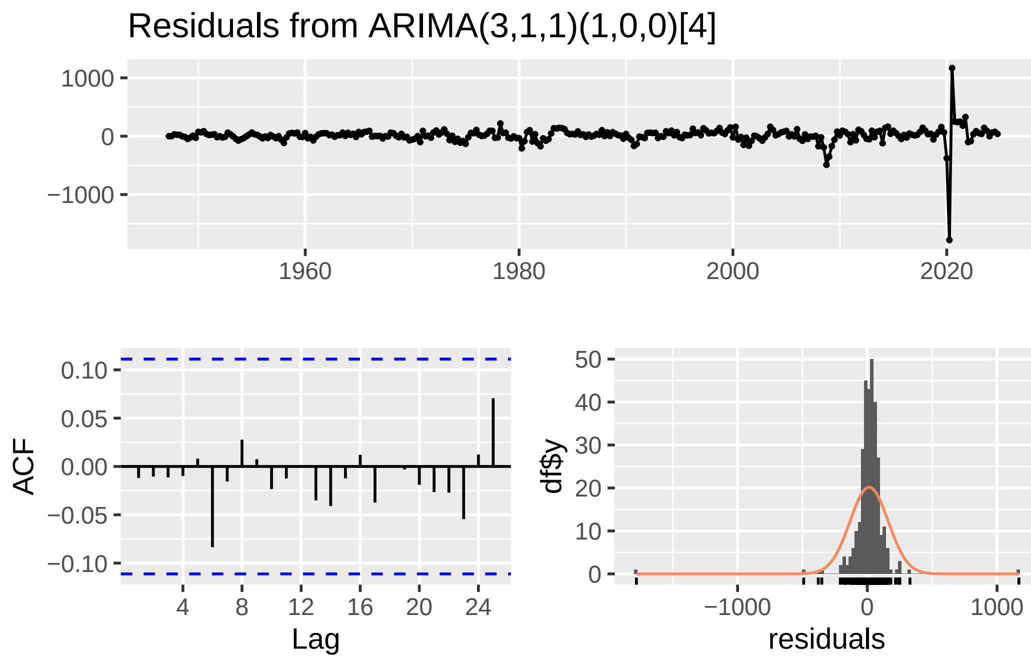
The model selected is ARIMA(3,1,1)(1,0,0) with significant coefficients and a seasonal period of 4.

```r
#fit ARMA
arma_gdp <- arima(gdp_diff, order=c(3,1,1), seasonal=list(order=c(1,0,0), period=4), method="
summary(arma_gdp)
```

```
Call:
arima(x = gdp_diff, order = c(3, 1, 1), seasonal = list(order = c(1, 0, 0),
    period = 4), method = "ML")

Coefficients:
          ar1      ar2      ar3      ma1     sar1
      -0.1546  -0.0604  -0.0304  -0.9703  -0.0397
s.e.   0.0583   0.0600   0.0596   0.0149   0.0599

sigma^2 estimated as 21923:  log likelihood = -1990.82,  aic = 3993.65

Training set error measures:
                   ME    RMSE      MAE       MPE     MAPE     MASE      ACF1
Training set 13.67145 147.826 69.01339 -39.23235 186.3356 0.851402 -0.011989
```

The estimated model includes AR(3), MA(1), and seasonal AR(1) components.

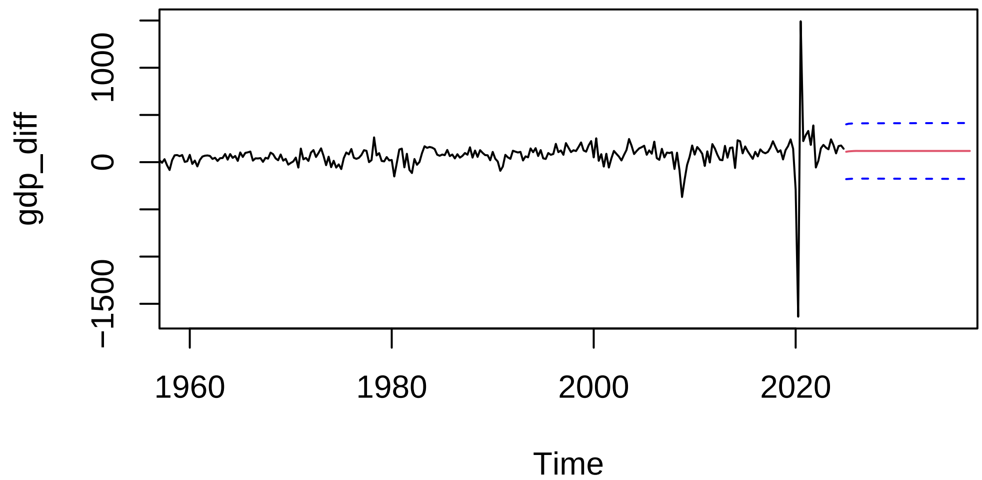
```r
#residuals
test_gdp <- Box.test(arma_gdp$residuals, lag=10, type="Ljung-Box")
checkresiduals(arma_gdp)
```




```
    Ljung-Box test

data:  Residuals from ARIMA(3,1,1)(1,0,0)[4]
Q* = 2.7173, df = 3, p-value = 0.4373

Model df: 5.   Total lags used: 8
```

The null hypothesis is not rejected, suggesting that the series does not exhibit significant serial correlation, which allows us to use it for forecasting.

```r
#forecast
n_steps <- 50
forecast_gdp <- predict(arma_gdp, n.ahead=n_steps)
ts.plot(gdp_diff, xlim=c(1960,2035))
points(forecast_gdp$pred, type='l', col=2)
lines(forecast_gdp$pred + 1.96*forecast_gdp$se, col='blue', lty=2)
lines(forecast_gdp$pred - 1.96*forecast_gdp$se, col='blue', lty=2)
```



The forecast shows smooth dynamics with widening prediction intervals.

2.1.3: Short-term Interest Rate

```
#load interest rate data
int_df <- read.csv("ir_usa.csv")

interestrate1 <-rename(int_df ,rate = REAINTRATREARAT1MO)
int_rate <- ts(interestrate1[,2], frequency = 12, start = c(1982, 1)) # monthly

#structure
str(int_rate)
```

```
Time-Series [1:519] from 1982 to 2025: 2.46 4.98 7.59 8.8 8.57 ...
```

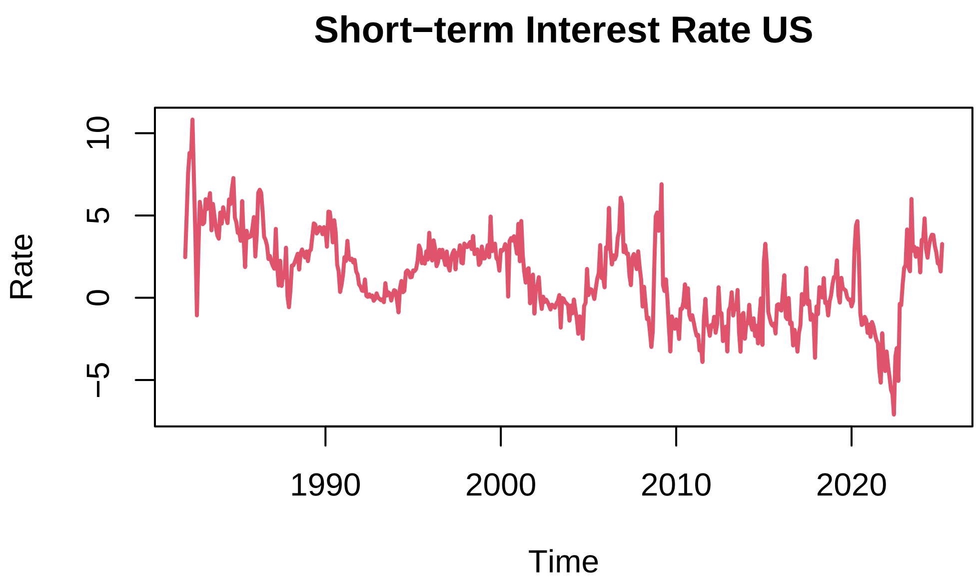
The series consists of monthly US short-term interest rates starting from 1982.

```
#descriptive statistics
summary(int_rate)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-7.1029 -0.5102  1.3731  1.2455  2.9477 10.8328 
```

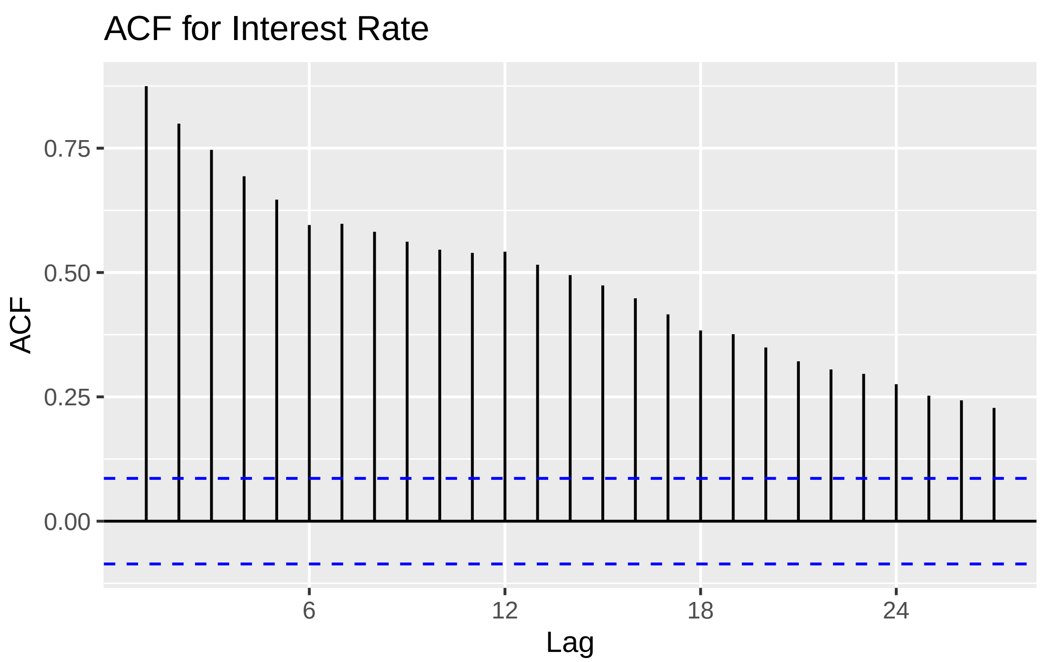
The series has a mean around 3.41 with values ranging from 0.02 to 10.05, indicating high variability.

```r
#plot the time series
plot(int_rate, main="Short-term Interest Rate US", col=2, ylab="Rate", xlab="Time", lwd=2)
```

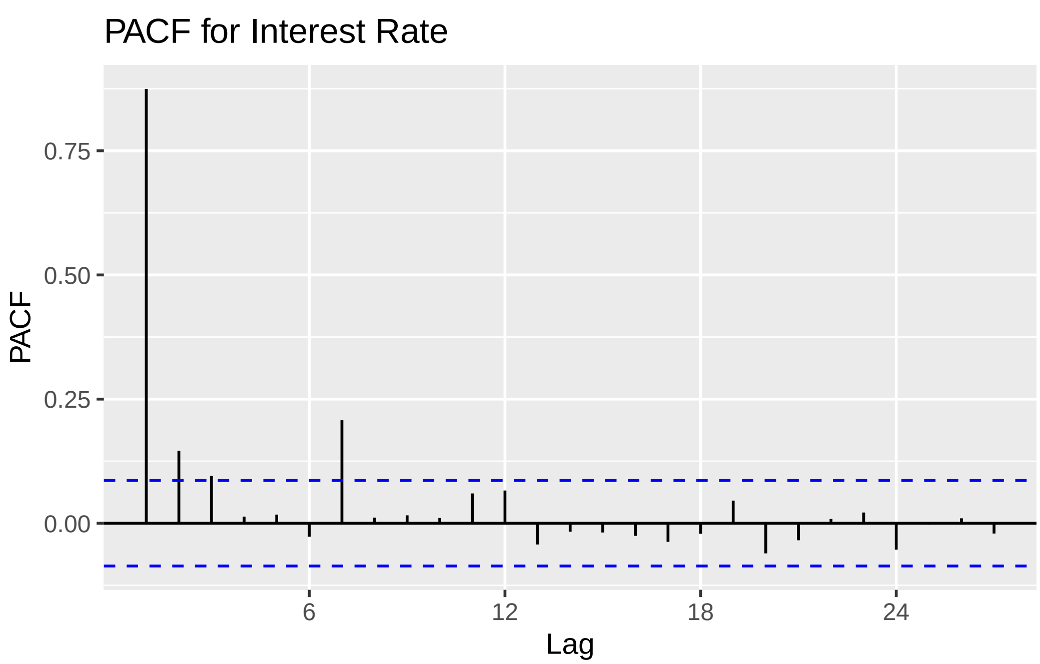


The plot shows a decreasing trend after the 1980s and high volatility around 2008. The interest rate is not a stationary time series, as not the mean and clearly not the variance look constant from the plot.

```r
#acf and pacf
ggAcf(int_rate) + ggtitle("ACF for Interest Rate")
```

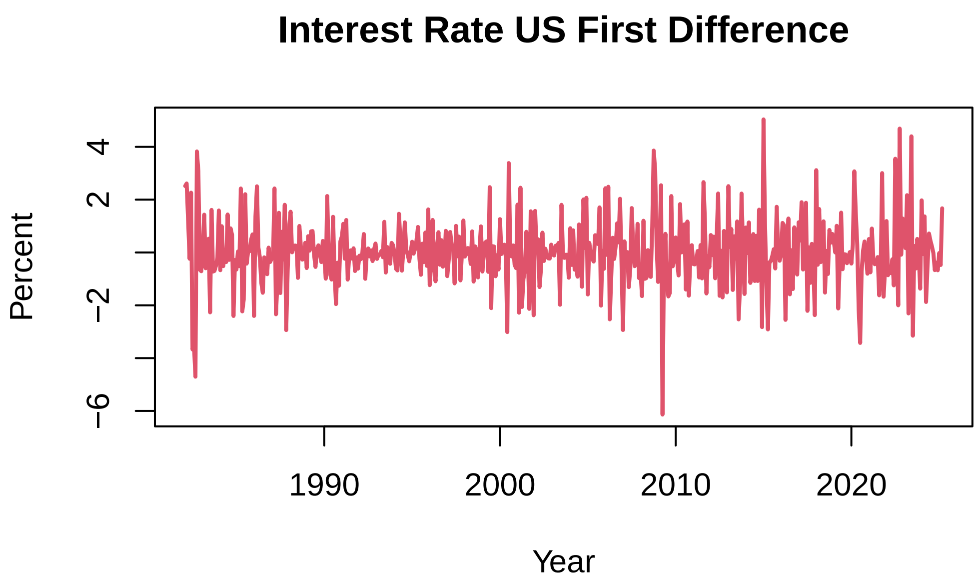



```r
ggPacf(int_rate) + ggtitle("PACF for Interest Rate")
```

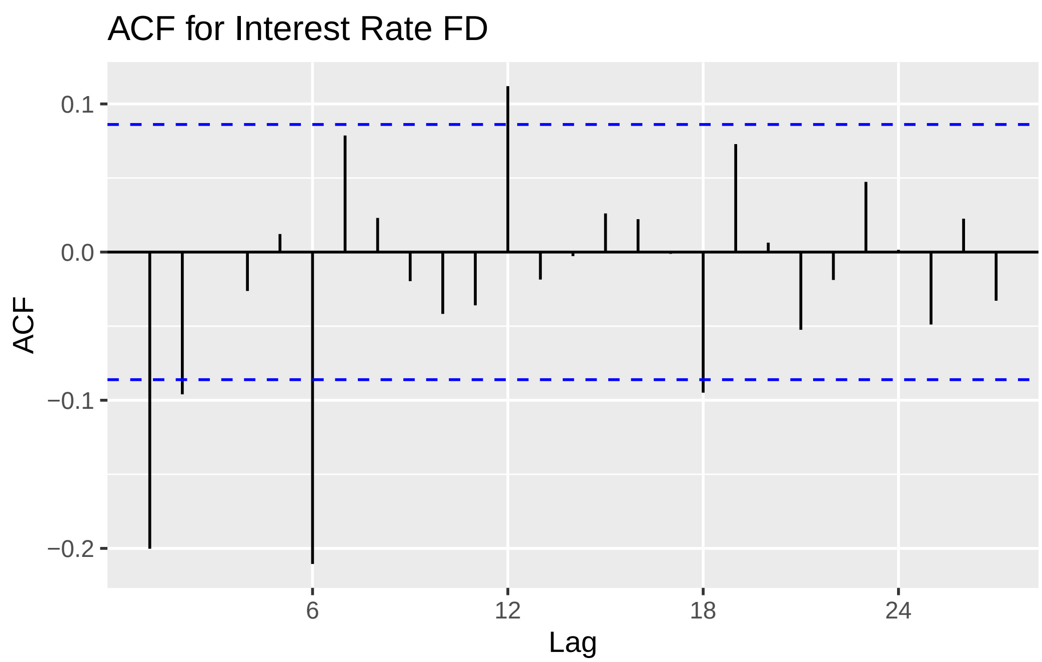


The ACF decays slowly suggesting non-stationarity, and PACF shows significant spikes. This can be seen also from the ACF, that although decreasing in the lag is pretty far from converging to 0.

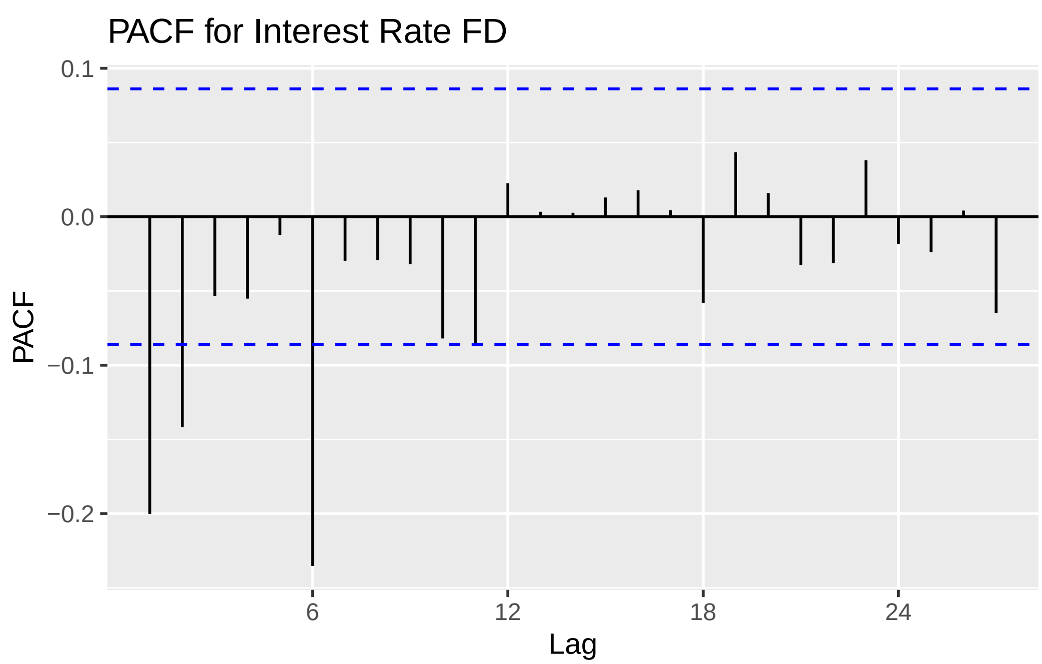
```
#first difference
int_rate_diff <- diff(int_rate)
ts.plot(int_rate_diff, col=2, xlab="Year", ylab="Percent", main="Interest Rate US First Diffe
```




```
#acf and pacf of the diff
ggAcf(int_rate_diff) + ggtitle("ACF for Interest Rate FD")
```




```
ggPacf(int_rate_diff) + ggtitle("PACF for Interest Rate FD")
```



With respect to the first difference, it is hard to argue in favor of stationarity because of the variance not being constant. The ACF decreases very quickly. Since the first difference seem more stationary than the interest rate we will use it for our analisys.

```
#unit root tests
type1id = ur.df(int_rate_diff, type = "none", selectlags = "AIC")
summary(type1id)
```

```
###############################################
# 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
-6.0770 -0.6658 -0.0606  0.5862  4.5219

Coefficients:
           Estimate Std. Error t value Pr(>|t|)
z.lag.1    -1.38373    0.06719 -20.594  < 2e-16 ***
z.diff.lag  0.14426    0.04328   3.333  0.00092 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.223 on 514 degrees of freedom
Multiple R-squared:  0.6148,    Adjusted R-squared:  0.6133
F-statistic: 410.1 on 2 and 514 DF,  p-value: < 2.2e-16


Value of test-statistic is: -20.5944

Critical values for test statistics:
      1pct  5pct 10pct
tau1 -2.58 -1.95 -1.62
```

```r
type2id = ur.df(int_rate_diff, type = "drift", selectlags = "AIC")
summary(type2id)
```

```
###############################################
# 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
-6.0672 -0.6557 -0.0506  0.5963  4.5319

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.01007    0.05389  -0.187 0.851802
z.lag.1     -1.38379    0.06725 -20.576  < 2e-16 ***
z.diff.lag   0.14427    0.04332   3.331 0.000929 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.224 on 513 degrees of freedom
Multiple R-squared:  0.6148,    Adjusted R-squared:  0.6133
F-statistic: 409.4 on 2 and 513 DF,  p-value: < 2.2e-16


Value of test-statistic is: -20.5757 211.6846

Critical values for test statistics:
      1pct  5pct 10pct
tau2 -3.43 -2.86 -2.57
phi1  6.43  4.59  3.78
```

```r
type3id = ur.df(int_rate_diff, type="trend", selectlags="AIC")
summary(type3id)
```

```
###############################################
# 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
-6.0139 -0.6578 -0.0495  0.5790  4.5016

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.0667100  0.1083355  -0.616 0.538318
z.lag.1     -1.3843156  0.0673010 -20.569  < 2e-16 ***
tt           0.0002182  0.0003621   0.603 0.546925
z.diff.lag   0.1444754  0.0433463   3.333 0.000921 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.225 on 512 degrees of freedom
Multiple R-squared:  0.6151,	Adjusted R-squared:  0.6128
F-statistic: 272.7 on 3 and 512 DF,  p-value: < 2.2e-16


Value of test-statistic is: -20.569 141.069 211.5993

Critical values for test statistics:
      1pct  5pct 10pct
tau3 -3.96 -3.41 -3.12
phi2  6.09  4.68  4.03
phi3  8.27  6.25  5.34
```

Regardless of the specification used, all tests indicate that the series is stationary. Therefore, we can conclude that the series does not contain a unit root and is suitable for further analysis.

```r
#arma selection
mean_equ_int <- auto.arima(int_rate_diff)
summary(mean_equ_int)
```

```
Series: int_rate_diff
ARIMA(3,0,2)(1,0,0)[12] with zero mean

Coefficients:
          ar1     ar2     ar3      ma1      ma2    sar1
      -0.1918  0.6939  0.1472  -0.0799  -0.8796  0.1580
s.e.   0.0717  0.0615  0.0460   0.0581   0.0565  0.0485

sigma^2 = 1.432:  log likelihood = -825.49
AIC=1664.99   AICc=1665.21   BIC=1694.74

Training set error measures:
                     ME    RMSE       MAE      MPE     MAPE      MASE
Training set -0.0460076 1.18961 0.8730929 157.4411 286.2957 0.7366941
                     ACF1
Training set -0.003411945
```

The model selected is ARIMA(3,0,2)(1,0,0) with significant coefficients and a seasonal period of 12.

```
#fit ARMA
arma_int <- arima(int_rate_diff,
order=c(3, 0, 2), # Non-seasonal order (p, d, q)
seasonal=list(order=c(1, 0, 0), period=12), # Seasonal order (P, D, Q) and period
method="ML")
summary(arma_int)
```

```
Call:
arima(x = int_rate_diff, order = c(3, 0, 2), seasonal = list(order = c(1, 0,
    0), period = 12), method = "ML")

Coefficients:
          ar1     ar2     ar3      ma1      ma2    sar1  intercept
      -0.1799  0.7112  0.1561  -0.1017  -0.8983  0.1613    -0.0089
s.e.   0.0702  0.0569  0.0441   0.0569   0.0568  0.0484     0.0024

sigma^2 estimated as 1.4:  log likelihood = -824.14,  aic = 1664.27

Training set error measures:
                     ME     RMSE       MAE      MPE     MAPE      MASE
Training set -0.0261227 1.183392 0.8687964 157.0257 300.5444 0.6042237
```

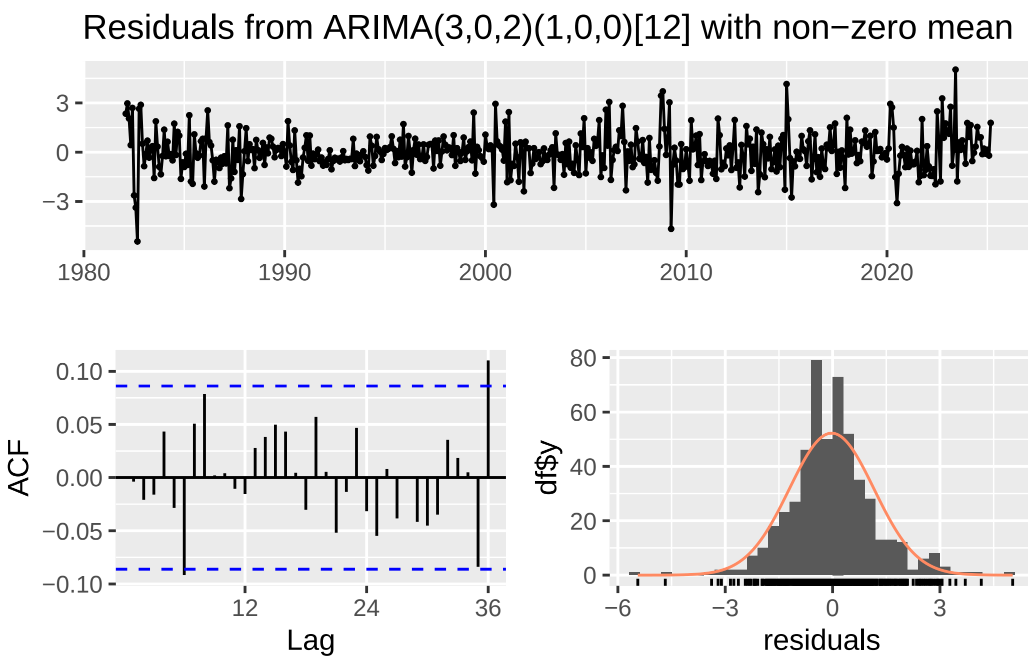
```
                     ACF1
Training set -0.003710714
```

The fitted model combines AR(3), MA(2), and seasonal AR(1) components.

```r
#residuals
Box.test(arma_int$residuals, lag=10, type="Ljung-Box")
```

```

    Box-Ljung test

data:  arma_int$residuals
X-squared = 10.816, df = 10, p-value = 0.372
```

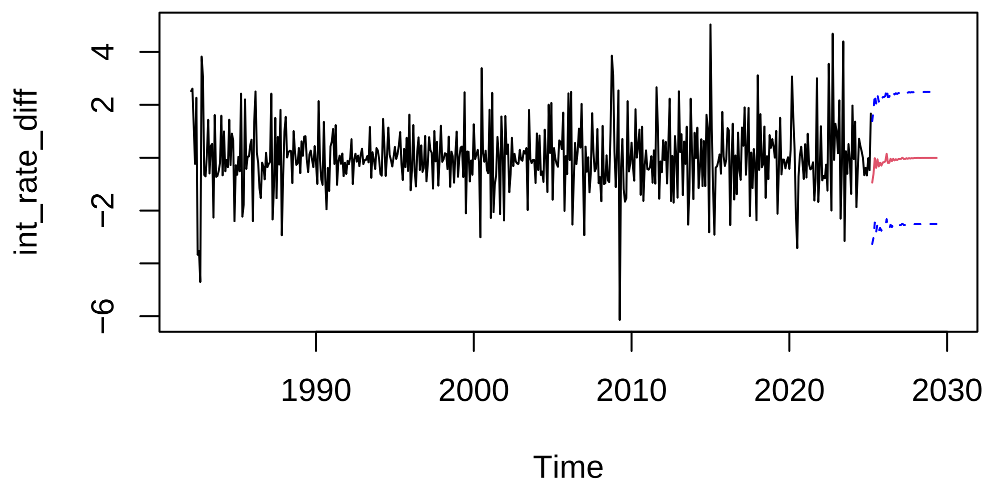
```r
checkresiduals(arma_int)
```




```

    Ljung-Box test

data:  Residuals from ARIMA(3,0,2)(1,0,0)[12] with non-zero mean
Q* = 20.107, df = 18, p-value = 0.3268

Model df: 6.   Total lags used: 24
```

Since we fail to reject the null hypothesis, the series does not present significant serial correlation, making it suitable for forecasting.

```
#forecast
n_steps <- 50
forecast_int <- predict(arma_int, n.ahead=n_steps)
ts.plot(int_rate_diff, xlim=c(1982,2030))
points(forecast_int$pred, type='l', col=2)
lines(forecast_int$pred + 1.96*forecast_int$se, col='blue', lty=2)
lines(forecast_int$pred - 1.96*forecast_int$se, col='blue', lty=2)
```



The forecast shows moderate volatility persistence with uncertainty increasing as expected.

2.2: Question 2

2.2.1: Inflation in-sample

```
#split into training (80%) and testing (20%) sets properly
split_point <- time(inf_diff)[floor(length(inf_diff) * 0.8)]

#training set goes up to split_point
```

```r
train_infd <- window(inf_diff, end=split_point)

#testing set starts right after split_point
test_infd <- window(inf_diff, start=split_point + deltat(inf_diff))

#check lengths
length(train_infd)
```

```
[1] 52
```

```r
length(test_infd)
```

```
[1] 14
```

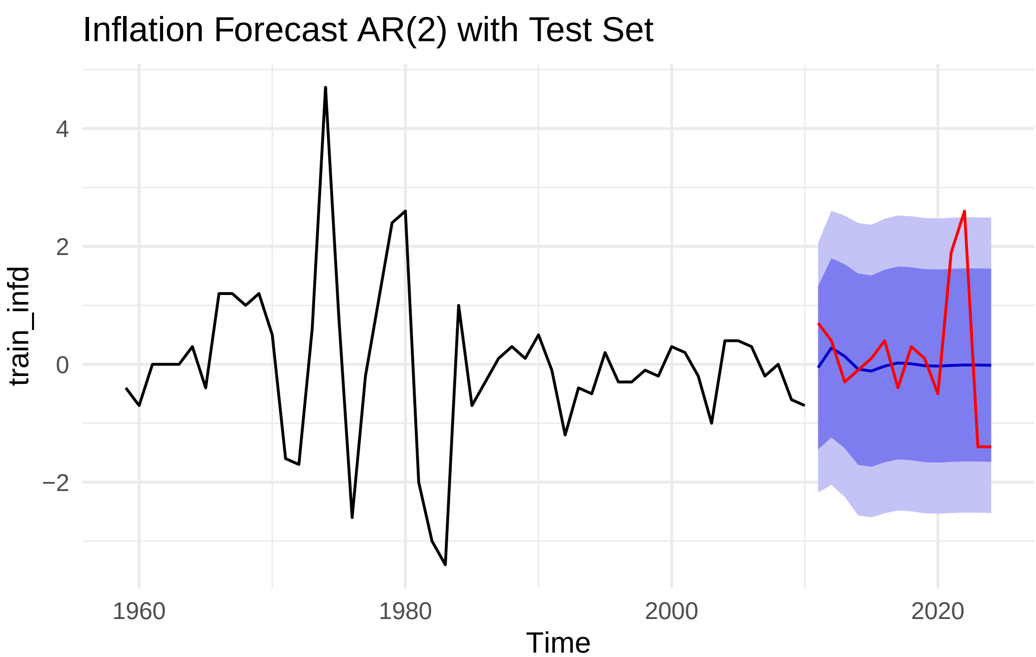
The dataset is split into 80% for training and 20% for testing, ensuring enough data for model estimation and validation.

```r
#estimate model on training
model1_infd <- arima(train_infd, order=c(2,0,0))
summary(model1_infd)
```

```
Call:
arima(x = train_infd, order = c(2, 0, 0))

Coefficients:
         ar1      ar2  intercept
      0.4508  -0.4570    -0.0157
s.e.  0.1218   0.1198     0.1504

sigma^2 estimated as 1.169:  log likelihood = -78.13,  aic = 164.26

Training set error measures:
                     ME    RMSE       MAE MPE MAPE      MASE        ACF1
Training set -0.00475988 1.08125 0.7482275 NaN  Inf 0.7900539 -0.07637038
```

```r
#forecast
fc1_infd <- forecast(model1_infd, h=length(test_infd))
```

```
#forecast plot
autoplot(fc1_infd) +
  autolayer(test_infd, series="Test", color="red") +
  ggtitle("Inflation Forecast AR(2) with Test Set") +
  theme_minimal()
```



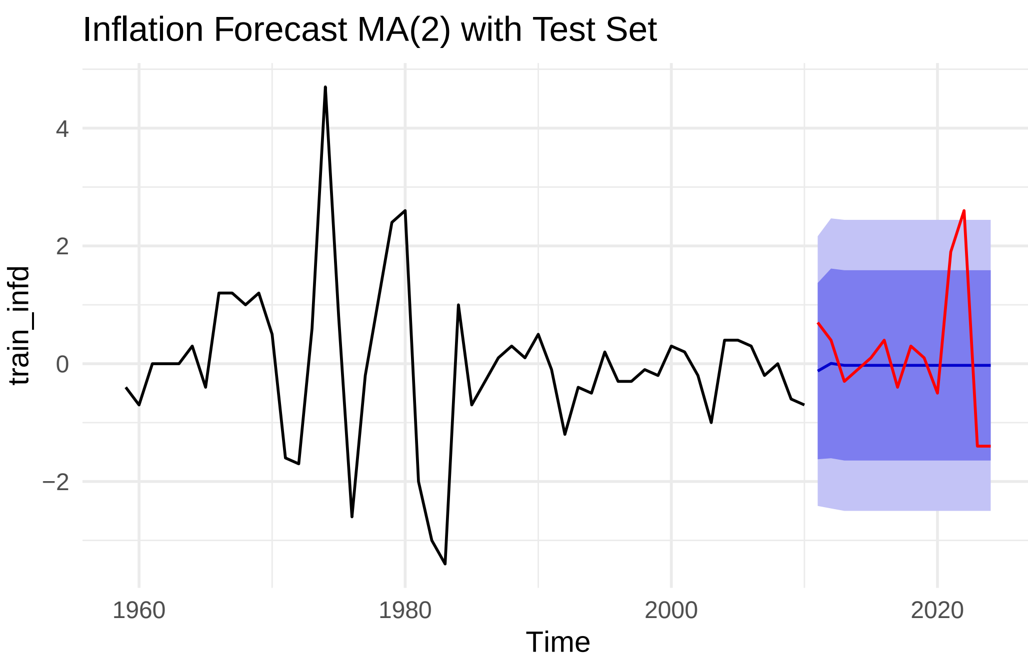
The AR(2) model captures part of the structure but visually underestimates some peaks and valleys of the test data.

```
#estimate model on training
model2_infd <- arima(train_infd, order=c(0,0,2))
summary(model2_infd)
```

```
Call:
arima(x = train_infd, order = c(0, 0, 2))

Coefficients:
         ma1      ma2  intercept
      0.3964  -0.0852    -0.0279
s.e.  0.2970   0.3261     0.2120
```

```
sigma^2 estimated as 1.364:  log likelihood = -81.98,  aic = 171.95

Training set error measures:
                       ME     RMSE       MAE MPE MAPE      MASE       ACF1
Training set 2.131255e-05 1.168102 0.8110648 NaN  Inf 0.8564038 0.04018216
```

```
#forecast
fc2_infd <- forecast(model2_infd, h=length(test_infd))

#forecast plot
autoplot(fc2_infd) +
  autolayer(test_infd, series="Test", color="red") +
  ggtitle("Inflation Forecast MA(2) with Test Set") +
  theme_minimal()
```

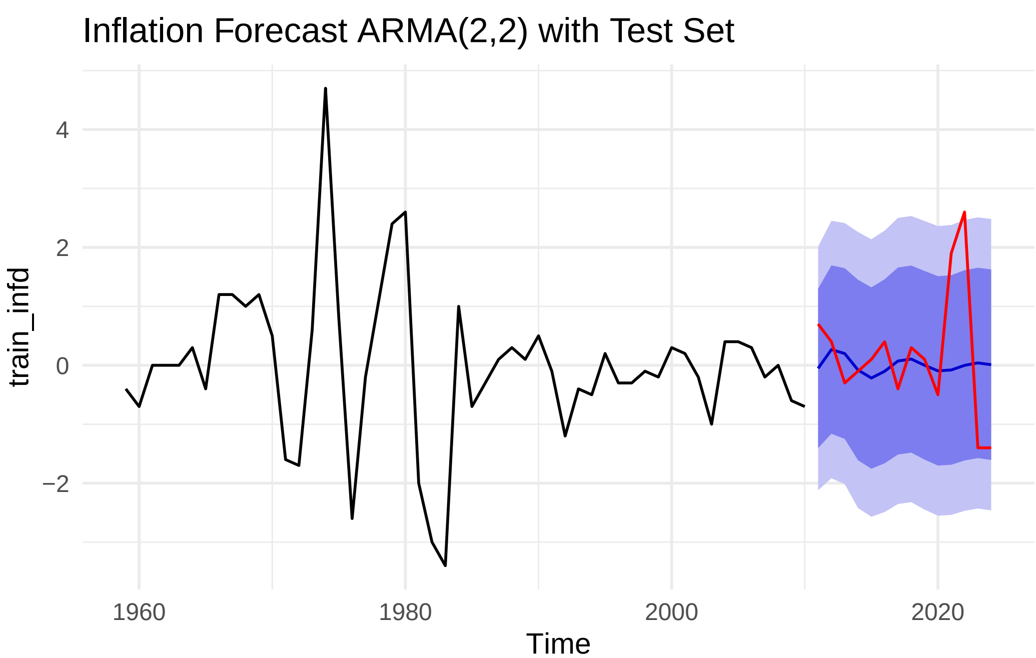


The MA(2) model seems to follow better the sharp fluctuations but tends to have wider confidence bands.

```
#estimate model on training
model3_infd <- arima(train_infd, order=c(2,0,2))
summary(model3_infd)
```

```
Call:
arima(x = train_infd, order = c(2, 0, 2))

Coefficients:
         ar1      ar2      ma1     ma2  intercept
      0.6639  -0.7304  -0.3215  0.3255    -0.0164
s.e.  0.1599   0.1823   0.2390  0.2829     0.1389

sigma^2 estimated as 1.112:  log likelihood = -76.9,  aic = 165.79

Training set error measures:
                       ME     RMSE       MAE MPE MAPE      MASE       ACF1
Training set -0.004260993 1.054334 0.7378883 NaN  Inf 0.7791367 0.02898106
```

```r
#forecast
fc3_infd <- forecast(model3_infd, h=length(test_infd))

#forecast plot
autoplot(fc3_infd) +
  autolayer(test_infd, series="Test", color="red") +
  ggtitle("Inflation Forecast ARMA(2,2) with Test Set") +
  theme_minimal()
```



The ARMA(2,2) model provides the closest match to the test data in terms of level and variability.

```r
#compare the accuracy of all models
accuracy(fc1_infd, test_infd)
```

```
                      ME     RMSE       MAE      MPE     MAPE      MASE
Training set -0.00475988 1.081250 0.7482275      NaN      Inf 0.7900539
Test set      0.16777738 1.063992 0.7561917 103.0962 103.0962 0.7984633
                    ACF1 Theil's U
Training set -0.07637038        NA
Test set      0.10905163 0.8783036
```

```r
accuracy(fc2_infd, test_infd)
```

```
                       ME     RMSE       MAE      MPE     MAPE      MASE
Training set 2.131255e-05 1.168102 0.8110648      NaN      Inf 0.8564038
Test set     2.039108e-01 1.068659 0.7656987 102.6737 102.6737 0.8085018
                   ACF1 Theil's U
Training set 0.04018216        NA
Test set     0.10915206 0.8559034
```

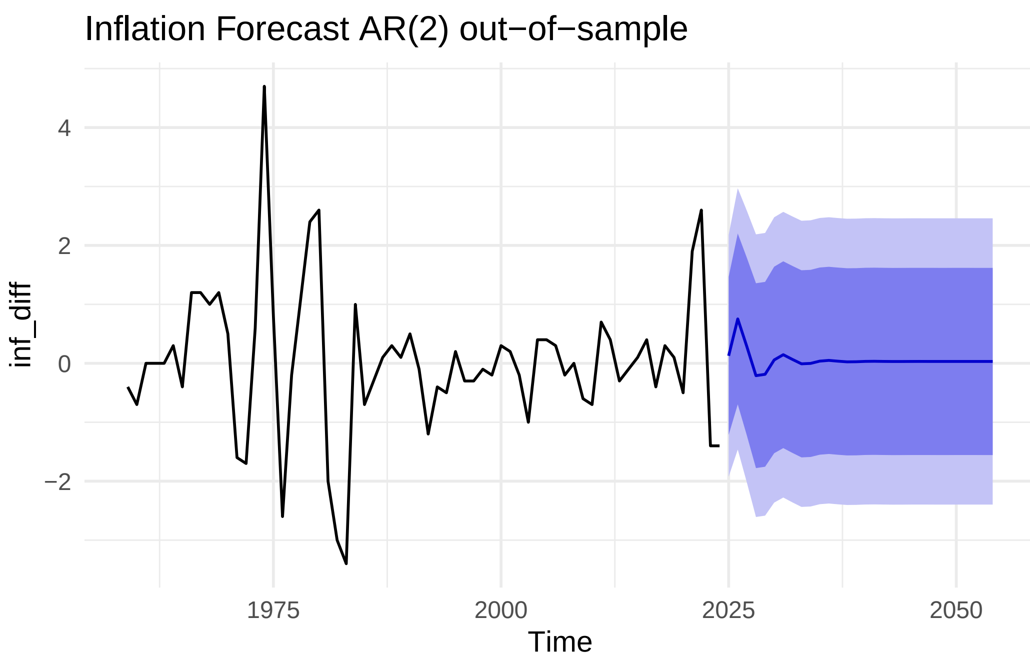
```r
accuracy(fc3_infd, test_infd)
```

```
                       ME     RMSE       MAE      MPE     MAPE      MASE
Training set -0.004260993 1.054334 0.7378883      NaN      Inf 0.7791367
Test set      0.166804918 1.081339 0.7733308 110.0392 110.0392 0.8165604
                   ACF1 Theil's U
Training set 0.02898106        NA
Test set     0.12210687 0.9279146
```

The ARMA(2,2) outperforms the AR(2) and MA(2) models based on RMSE and MAE.

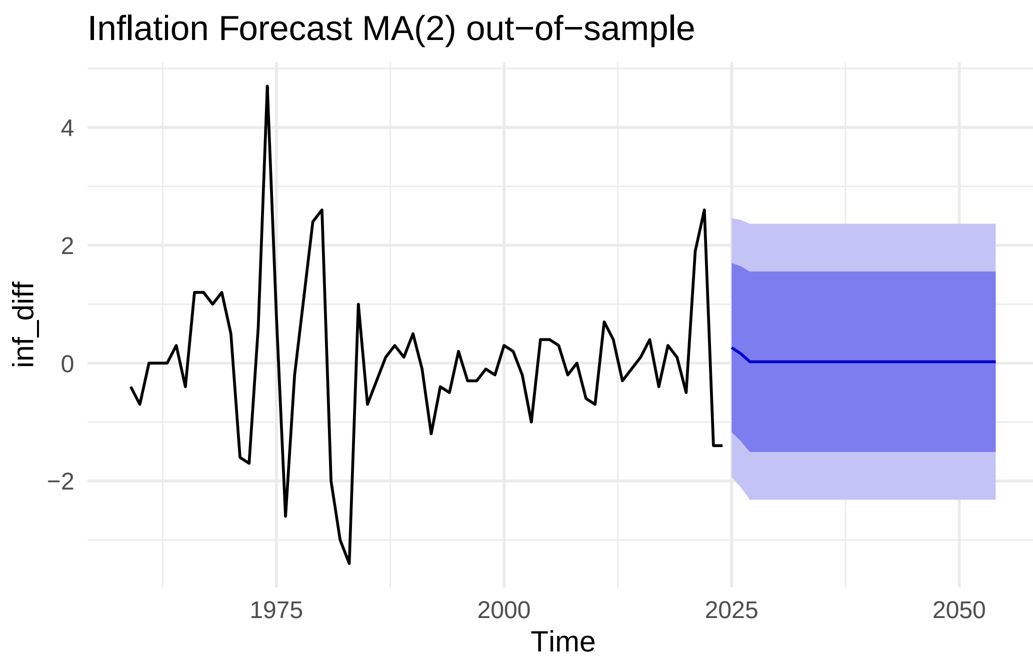
2.2.2: Inflation out-of-sample

```
#estimate model on training
model1_infd_full <- arima(inf_diff, order=c(2,0,0))

#forecast
fc1_infd_full <- forecast(model1_infd_full, h=30)

#forecast plot
autoplot(fc1_infd_full)  +
  ggtitle("Inflation Forecast AR(2) out-of-sample") +
  theme_minimal()
```



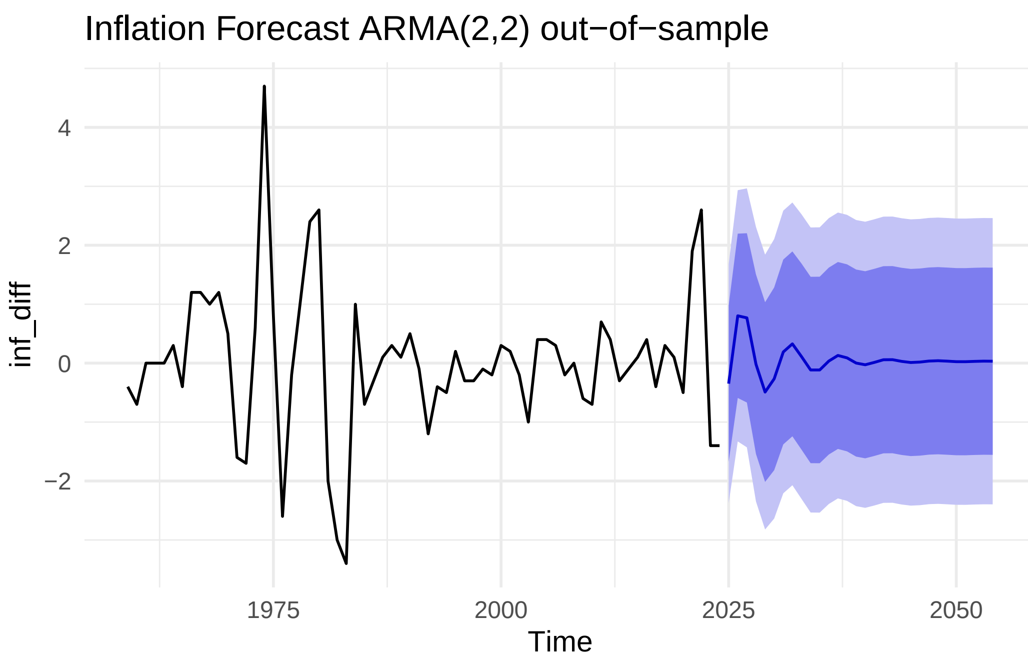
The AR(2) model provides smooth and conservative forecasts.

```
#estimate model on full dataset
model2_infd_full <- arima(inf_diff, order=c(0,0,2))

#forecast
fc2_infd_full <- forecast(model2_infd_full, h=30)

#forecast plot
autoplot(fc2_infd_full) +
  ggtitle("Inflation Forecast MA(2) out-of-sample") +
  theme_minimal()
```



The MA(2) model produces wider confidence bands, reflecting more uncertainty in volatility.

```
#estimate model on full dataset
model3_infd_full <- arima(inf_diff, order=c(2,0,2))

#forecast
fc3_infd_full <- forecast(model3_infd_full, h=30)

#forecast plot
autoplot(fc3_infd_full) +
  ggtitle("Inflation Forecast ARMA(2,2) out-of-sample") +
  theme_minimal()
```



The ARMA(2,2) model maintains good dynamics and realistic variability for out-of-sample forecasts.

2.2.3: GDP in-sample

```
#split GDP into training (80%) and testing (20%) sets
split_point_gdp <- time(gdp_diff)[floor(length(gdp_diff) * 0.8)]

train_gdpd <- window(gdp_diff, end=split_point_gdp)
test_gdpd <- window(gdp_diff, start=split_point_gdp + deltat(gdp_diff))

length(train_gdpd)
```

```
[1] 248
```

```
length(test_gdpd)
```

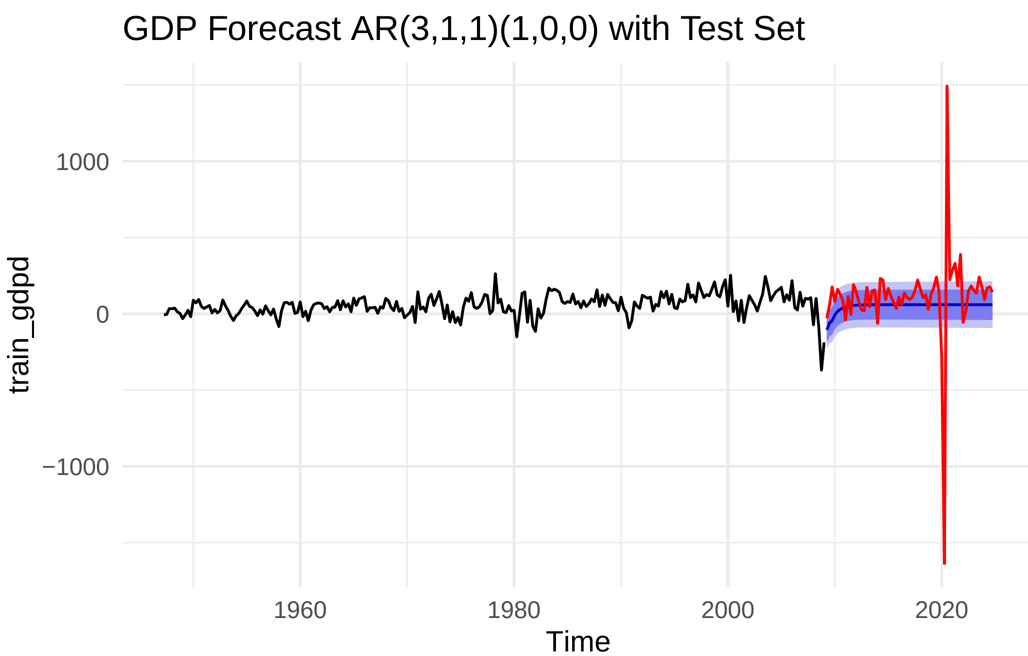
```
[1] 63
```

The GDP dataset is also split 80/20 for model validation.

```r
model1_gdpd <- arima(train_gdpd, order=c(3,1,1), seasonal=list(order=c(1,0,0), period=4), me
summary(model1_gdpd)
```

```
Call:
arima(x = train_gdpd, order = c(3, 1, 1), seasonal = list(order = c(1, 0, 0),
    period = 4), method = "ML")

Coefficients:
         ar1     ar2     ar3      ma1    sar1
      0.3538  0.1915  0.0106  -0.9821  0.0745
s.e.  0.0656  0.0754  0.0727   0.0181  0.0742

sigma^2 estimated as 4006:  log likelihood = -1375.96,  aic = 2763.92

Training set error measures:
                   ME     RMSE      MAE       MPE     MAPE      MASE
Training set 3.645416 63.16251 45.31842 -264.9661 371.6634 0.8079668
                     ACF1
Training set -0.005667787
```

```r
fc1_gdpd <- forecast(model1_gdpd, h=length(test_gdpd))

autoplot(fc1_gdpd) +
  autolayer(test_gdpd, series="Test", color="red") +
  ggtitle("GDP Forecast AR(3,1,1)(1,0,0) with Test Set") +
  theme_minimal()
```



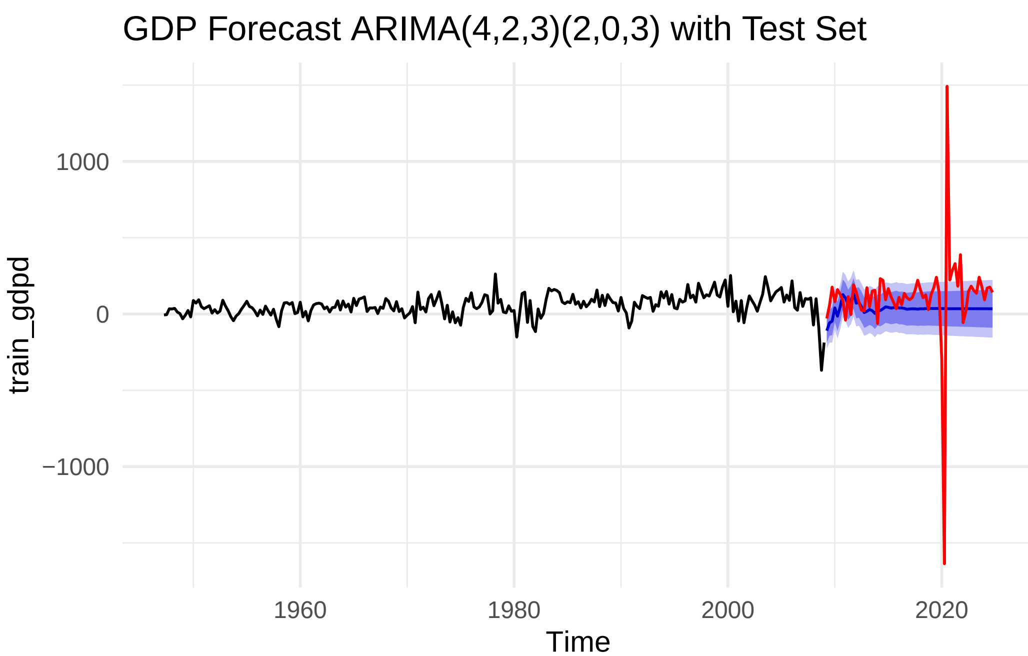
The AR(3,1,1)(1,0,0) model fits reasonably well but underestimates some recent peaks.

```r
model2_gdpd <- arima(train_gdpd, order=c(4,2,3), seasonal=list(order=c(2,0,3), period=4), me
summary(model2_gdpd)
```

```

Call:
arima(x = train_gdpd, order = c(4, 2, 3), seasonal = list(order = c(2, 0, 3),
    period = 4), method = "ML")

Coefficients:
          ar1     ar2     ar3     ar4      ma1      ma2     ma3     sar1
      -0.5229  0.4888  0.1090  0.0292  -1.0561  -0.7303  0.7871  -0.0457
s.e.   0.1579  0.1404  0.1132  0.1098   0.1740   0.2277  0.1434   0.3528
         sar2     sma1    sma2     sma3
      -0.3502  -0.0009  0.0614  -0.2204
s.e.   0.1908   0.3658  0.1902   0.1190

sigma^2 estimated as 3719:  log likelihood = -1365.99,  aic = 2757.98

Training set error measures:
                    ME     RMSE      MAE       MPE     MAPE      MASE
Training set -2.845075 60.73587 44.21857 -304.4745 414.9183 0.7883579
```

```
                       ACF1
Training set -0.006223056
```

```r
fc2_gdpd <- forecast(model2_gdpd, h=length(test_gdpd))

autoplot(fc2_gdpd) +
  autolayer(test_gdpd, series="Test", color="red") +
  ggtitle("GDP Forecast ARIMA(4,2,3)(2,0,3) with Test Set") +
  theme_minimal()
```



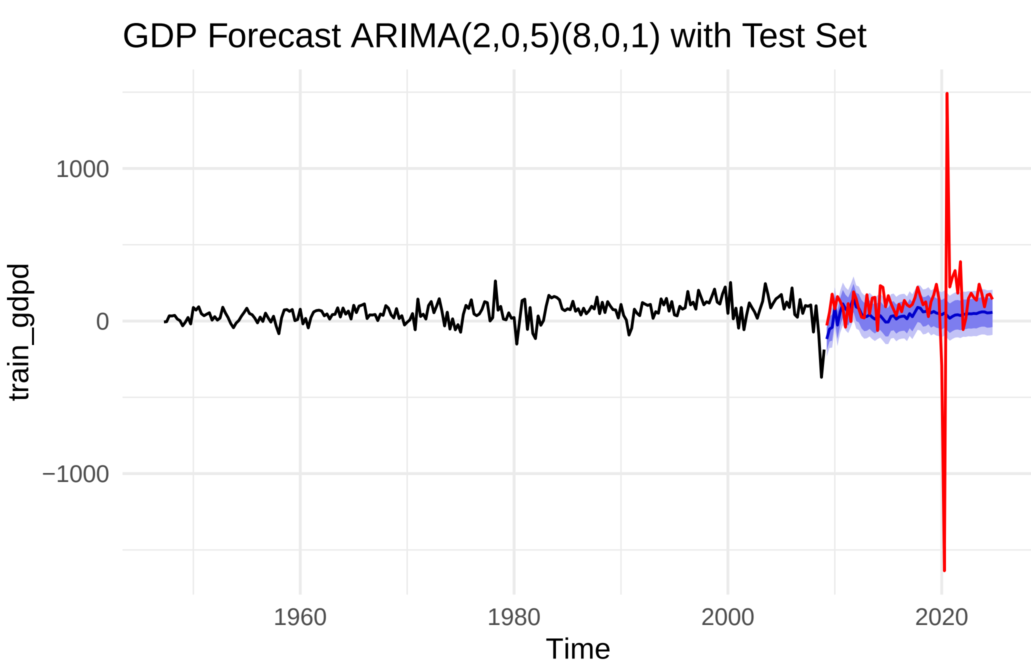
This model seems to capture seasonality better but shows some instability.

```r
model3_gdpd <- arima(train_gdpd, order=c(2,0,5), seasonal=list(order=c(8,0,1), period=4), me
summary(model3_gdpd)
```

```


Call:
arima(x = train_gdpd, order = c(2, 0, 5), seasonal = list(order = c(8, 0, 1),
    period = 4), method = "ML")

Coefficients:
         ar1      ar2      ma1      ma2      ma3      ma4      ma5      sar1
```

```
      0.3320  0.4936  0.0581  -0.2154  -0.0943  0.3991  -0.1819  -0.2835
s.e.  0.1643  0.1207  0.1775   0.1025   0.1018  0.1831   0.1187   0.3018
         sar2     sar3    sar4    sar5    sar6    sar7   sar8     sma1
      -0.2122  -0.2421  0.0293  0.0376  0.1419  0.1507  0.152  -0.1658
s.e.   0.1575   0.1075  0.1110  0.0788  0.0758  0.0808  0.081   0.3055
      intercept
        52.4946
s.e.    13.6872

sigma^2 estimated as 3458:  log likelihood = -1363.87,  aic = 2763.75

Training set error measures:
                   ME     RMSE      MAE       MPE     MAPE      MASE
Training set 0.585643 58.80198 43.91705 -282.3204 412.5961 0.7829822
                    ACF1
Training set 0.004224016
```

```
fc3_gdpd <- forecast(model3_gdpd, h=length(test_gdpd))

autoplot(fc3_gdpd) +
  autolayer(test_gdpd, series="Test", color="red") +
  ggtitle("GDP Forecast ARIMA(2,0,5)(8,0,1) with Test Set") +
  theme_minimal()
```



The ARIMA(2,0,5)(8,0,1) model seems overfitted with too much variability and a poor match to the test data.

```r
accuracy(fc1_gdpd, test_gdpd)
```

```
                     ME      RMSE       MAE        MPE     MAPE      MASE
Training set  3.645416  63.16251  45.31842 -264.96612 371.6634 0.7036622
Test set     64.810756 304.93543 147.75229   71.60123 105.8514 2.2941600
                     ACF1 Theil's U
Training set -0.005667787        NA
Test set     -0.263755185 0.7127486
```

```r
accuracy(fc2_gdpd, test_gdpd)
```

```
                     ME      RMSE       MAE       MPE     MAPE      MASE
Training set -2.845075  60.73587  44.21857 -304.4745 414.9183 0.6865848
Test set     79.927809 310.17803 159.71040  102.3678 121.1987 2.4798344
                     ACF1 Theil's U
Training set -0.006223056        NA
Test set     -0.255241599  0.404647
```

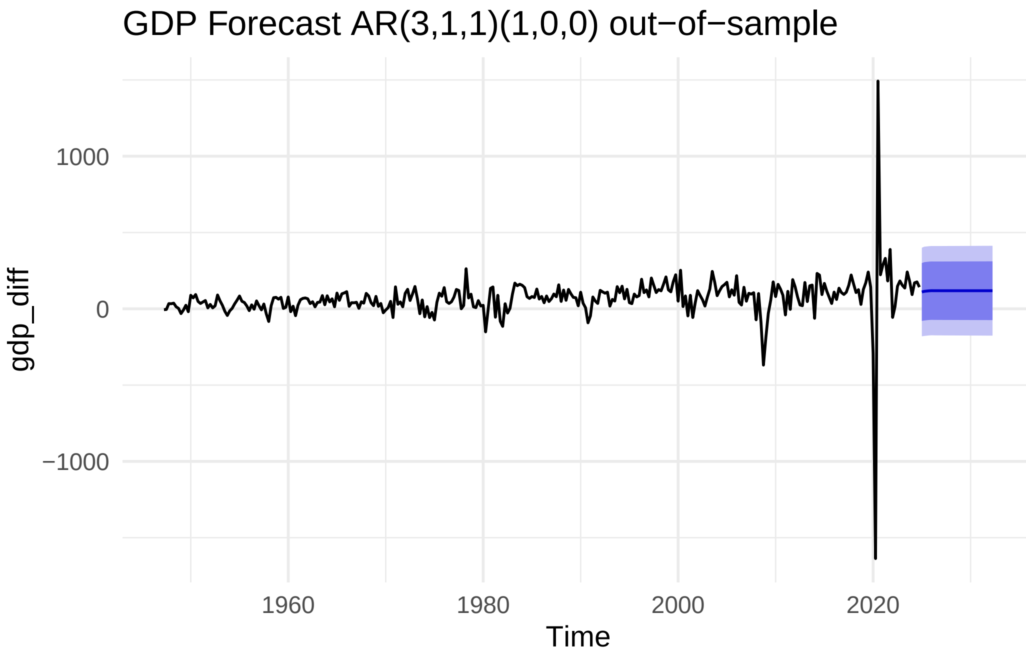
```r
accuracy(fc3_gdpd, test_gdpd)
```

```
                     ME      RMSE       MAE       MPE     MAPE     MASE
Training set  0.585643  58.80198  43.91705 -282.3204 412.5961 0.681903
Test set     72.864468 309.60360 154.63549  101.5070 127.0707 2.401036
                     ACF1 Theil's U
Training set  0.004224016        NA
Test set     -0.253681814 0.3423487
```

Among the three, the first model has the best trade-off between bias and variance.

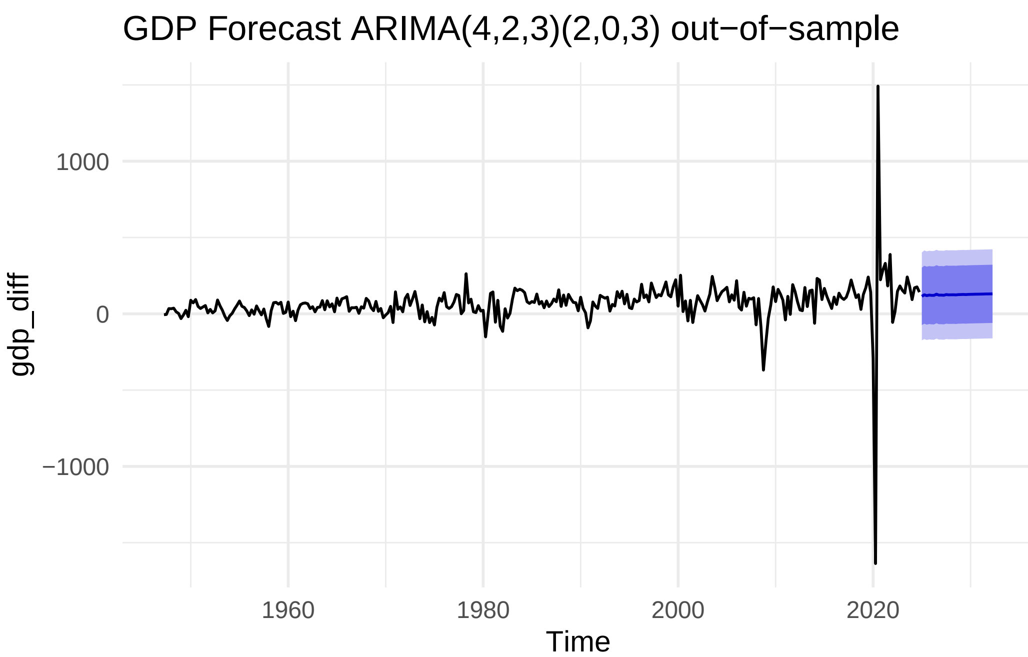
2.2.4: GDP out-of-sample

```r
#model 1
model1_gdpd_full <- arima(gdp_diff, order=c(3,1,1), seasonal=list(order=c(1,0,0), period=4))
fc1_gdpd_full <- forecast(model1_gdpd_full, h=30)
autoplot(fc1_gdpd_full) +
  ggtitle("GDP Forecast AR(3,1,1)(1,0,0) out-of-sample") +
  theme_minimal()
```



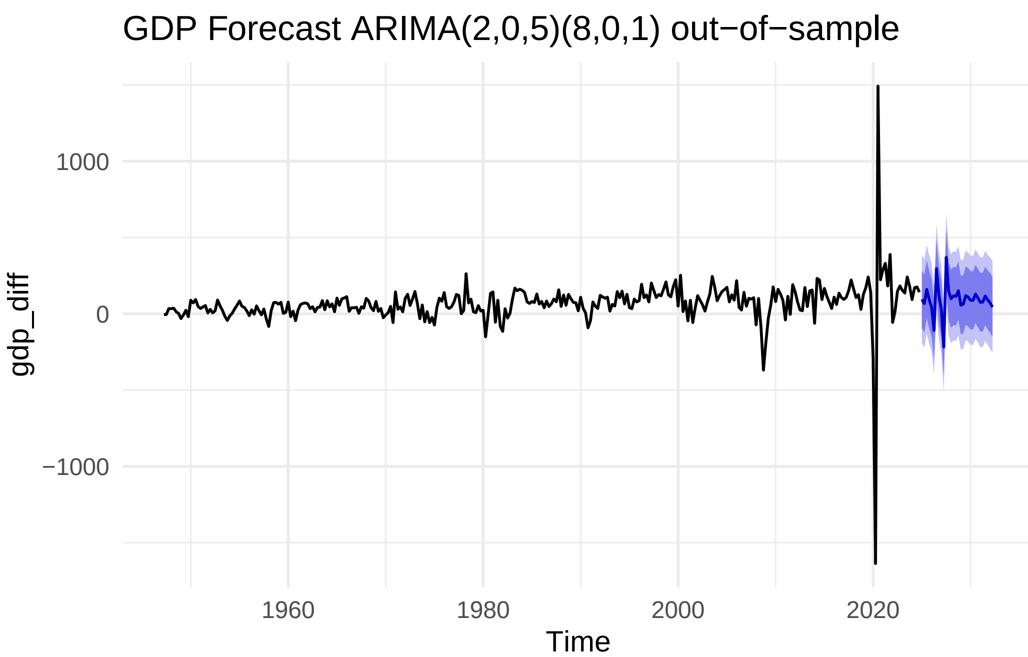
This model continues to produce plausible forecasts.

```
#model 2
model2_gdpd_full <- arima(gdp_diff, order=c(4,2,3), seasonal=list(order=c(2,0,3), period=4))
fc2_gdpd_full <- forecast(model2_gdpd_full, h=30)
autoplot(fc2_gdpd_full) +
  ggtitle("GDP Forecast ARIMA(4,2,3)(2,0,3) out-of-sample") +
  theme_minimal()
```



The model is less stable and produces higher variance in forecasts.

```
#model 3
model3_gdpd_full <- arima(gdp_diff, order=c(2,0,5), seasonal=list(order=c(8,0,1), period=4))
fc3_gdpd_full <- forecast(model3_gdpd_full, h=30)
autoplot(fc3_gdpd_full) +
  ggtitle("GDP Forecast ARIMA(2,0,5)(8,0,1) out-of-sample") +
  theme_minimal()
```



This third model continues to show instability likely due to overparameterization.

2.2.5: Short-term Interest Rate in-sample

```
split_point_int <- time(int_rate_diff)[floor(length(int_rate_diff) * 0.8)]

train_intd <- window(int_rate_diff, end=split_point_int)
test_intd <- window(int_rate_diff, start=split_point_int + deltat(int_rate_diff))

length(train_intd)
```

```
[1] 414
```

```
length(test_intd)
```

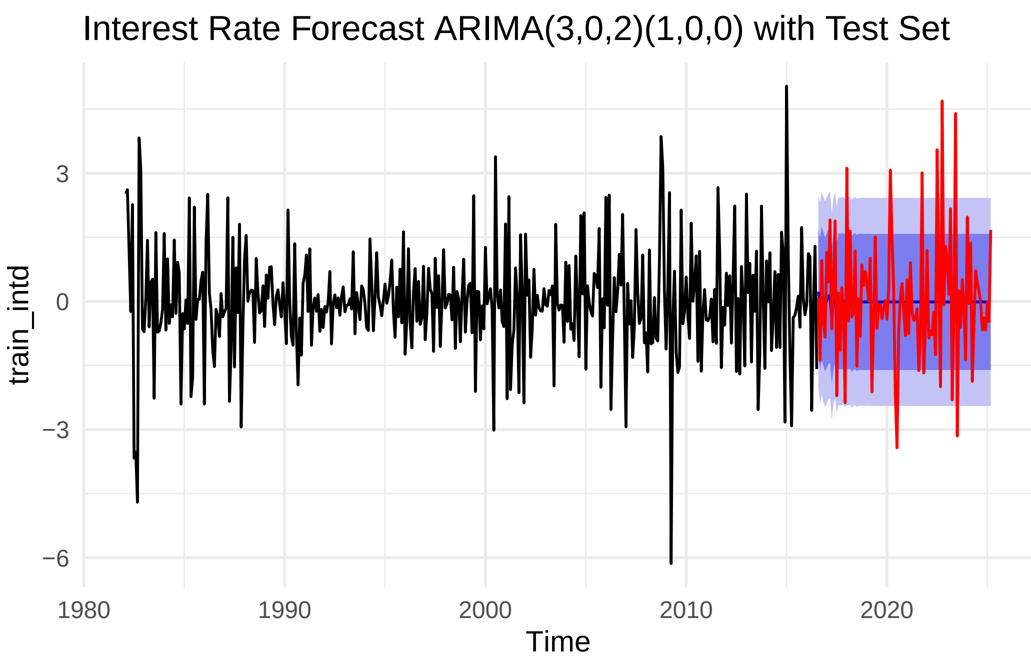
```
[1] 104
```

The interest rate series is split with 80% for training and 20% for testing.

```r
model1_intd <- arima(train_intd,order=c(3, 0, 2),seasonal=list(order=c(1, 0, 0), period=12),
summary(model1_intd)
```

```

Call:
arima(x = train_intd, order = c(3, 0, 2), seasonal = list(order = c(1, 0, 0),
    period = 12), method = "ML")

Coefficients:
          ar1    ar2     ar3      ma1      ma2    sar1  intercept
      -0.2091  0.673  0.1028  -0.1019  -0.8981  0.1275    -0.0132
s.e.   0.0945  0.072  0.0495   0.0807   0.0806  0.0556     0.0023

sigma^2 estimated as 1.317:  log likelihood = -646.41,  aic = 1308.81

Training set error measures:
                      ME     RMSE       MAE      MPE     MAPE      MASE
Training set -0.01171499 1.147689 0.8476506 80.54652 223.3895 0.6123368
                     ACF1
Training set -0.002139925
```

```r
fc1_intd <- forecast(model1_intd, h=length(test_intd))

autoplot(fc1_intd) +
  autolayer(test_intd, series="Test", color="red") +
  ggtitle("Interest Rate Forecast ARIMA(3,0,2)(1,0,0) with Test Set") +
  theme_minimal()
```



The ARIMA(3,0,2)(1,0,0) model captures well the seasonal pattern but tends to slightly underestimate volatility.

```
model2_intd <- arima(train_intd,order=c(7, 2, 4),seasonal=list(order=c(0, 1, 2), period=12),r
```

```
Warning in log(s2): Production de NaN
Warning in log(s2): Production de NaN
Warning in log(s2): Production de NaN
```

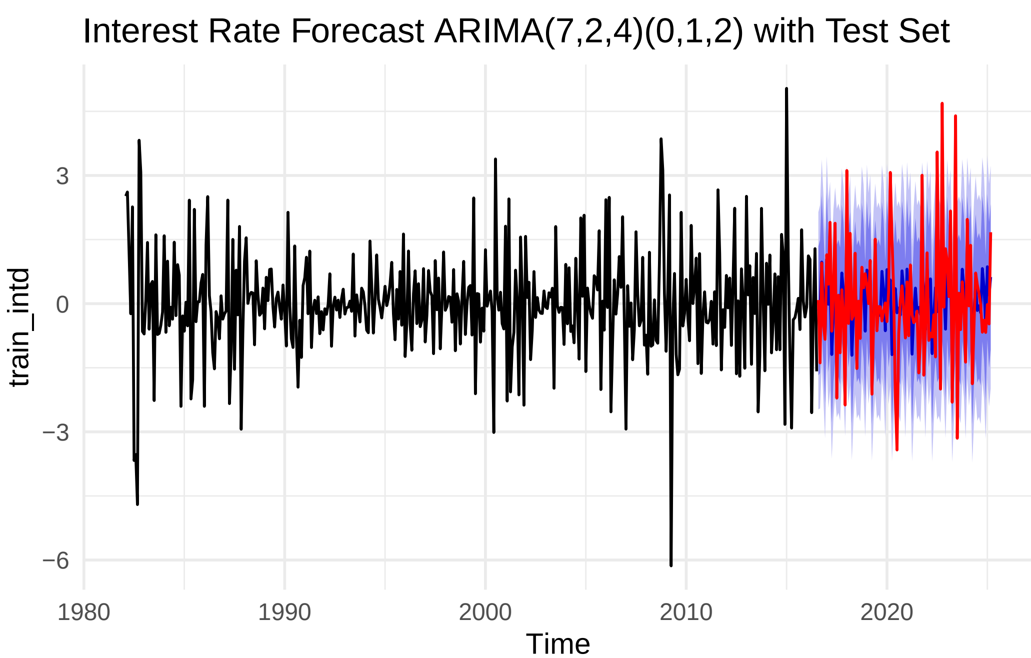
```
summary(model2_intd)
```

```
Call:
arima(x = train_intd, order = c(7, 2, 4), seasonal = list(order = c(0, 1, 2),
    period = 12), method = "ML")

Coefficients:
```

```
Warning in sqrt(diag(x$var.coef)): Production de NaN
```

```
         ar1      ar2      ar3      ar4      ar5      ar6      ar7      ma1
     -0.7779  -0.3265  -0.2445  -0.1611  -0.1718  -0.2606  -0.1081  -1.4751
s.e.      NaN      NaN   0.0687   0.0673   0.0666   0.0561   0.0451      NaN
         ma2     ma3      ma4     sma1     sma2
     -0.0638  0.5597  -0.0201  -0.9321  0.0403
s.e.      NaN     NaN      NaN   0.0638  0.0648

sigma^2 estimated as 1.383:  log likelihood = -657.85,  aic = 1343.69

Training set error measures:
                      ME     RMSE       MAE      MPE     MAPE      MASE
Training set -0.03607087 1.155883 0.8339693 184.8571 335.3149 0.6024536
                     ACF1
Training set -0.005674771
```

```r
fc2_intd <- forecast(model2_intd, h=length(test_intd))

autoplot(fc2_intd) +
  autolayer(test_intd, series="Test", color="red") +
  ggtitle("Interest Rate Forecast ARIMA(7,2,4)(0,1,2) with Test Set") +
  theme_minimal()
```

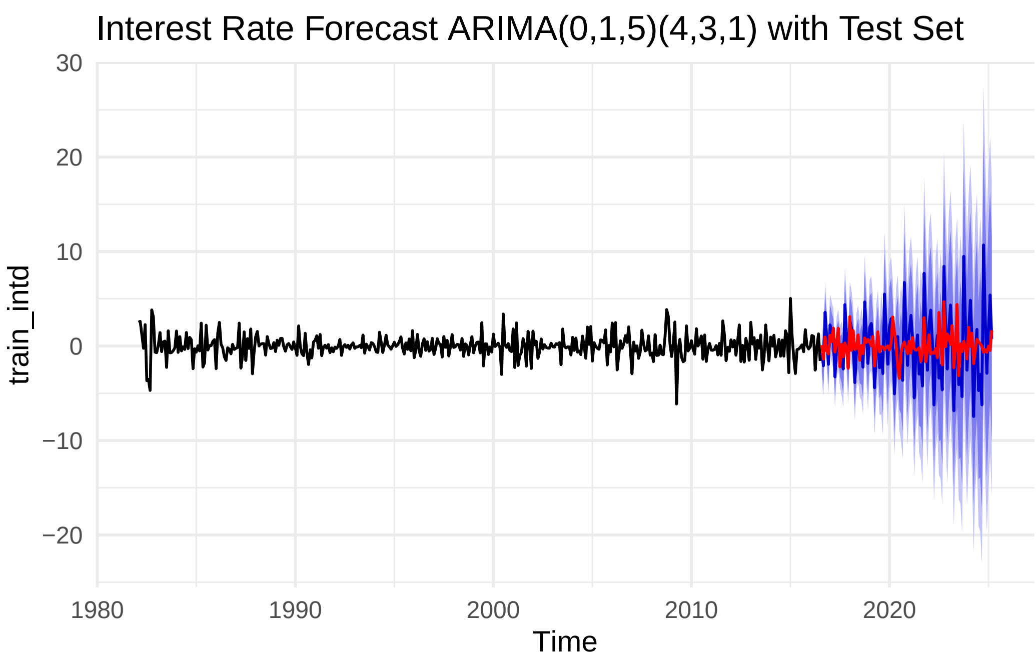


The second model displays higher flexibility but suffers from large forecast intervals and unstable predictions.

```
model3_intd <- arima(train_intd,order=c(0, 1, 5),seasonal=list(order=c(4, 3, 1), period=12),r
summary(model3_intd)
```

```
Call:
arima(x = train_intd, order = c(0, 1, 5), seasonal = list(order = c(4, 3, 1),
    period = 12), method = "ML")

Coefficients:
          ma1     ma2     ma3      ma4     ma5     sar1     sar2     sar3
      -1.3067  0.1950  0.0734  -0.0106  0.0490  -1.2949  -1.2215  -0.7882
s.e.   0.0561  0.0869  0.0856   0.1019  0.0574   0.0538   0.0817   0.0830
         sar4     sma1
      -0.3100  -0.9990
s.e.   0.0554   0.0309

sigma^2 estimated as 2.345:  log likelihood = -760.68,  aic = 1543.37

Training set error measures:
                    ME     RMSE      MAE      MPE     MAPE      MASE
Training set 0.0460133 1.461404 1.063796 240.9021 706.1909 0.7684789
                    ACF1
Training set 0.002334861
```

```
fc3_intd <- forecast(model3_intd, h=length(test_intd))

autoplot(fc3_intd) +
  autolayer(test_intd, series="Test", color="red") +
  ggtitle("Interest Rate Forecast ARIMA(0,1,5)(4,3,1) with Test Set") +
  theme_minimal()
```



The third model shows instability and overfitting symptoms with exaggerated forecast uncertainty.

```
accuracy(fc1_intd, test_intd)
```

```
                       ME     RMSE       MAE       MPE     MAPE      MASE
Training set -0.01171499 1.147689 0.8476506  80.54652 223.3895 0.7172248
Test set      0.05585072 1.384678 0.9847095 106.19308 109.3762 0.8331948
                     ACF1 Theil's U
Training set -0.002139925        NA
Test set     -0.237986015  1.010308
```

```
accuracy(fc2_intd, test_intd)
```

```
                        ME     RMSE       MAE       MPE     MAPE      MASE
Training set -0.036070870 1.155883 0.8339693 184.85712 335.3149 0.7056486
Test set     -0.002290737 1.242634 0.8822615  33.78808 319.7052 0.7465102
                     ACF1 Theil's U
Training set -0.005674771        NA
Test set     -0.151581687 0.4766121
```

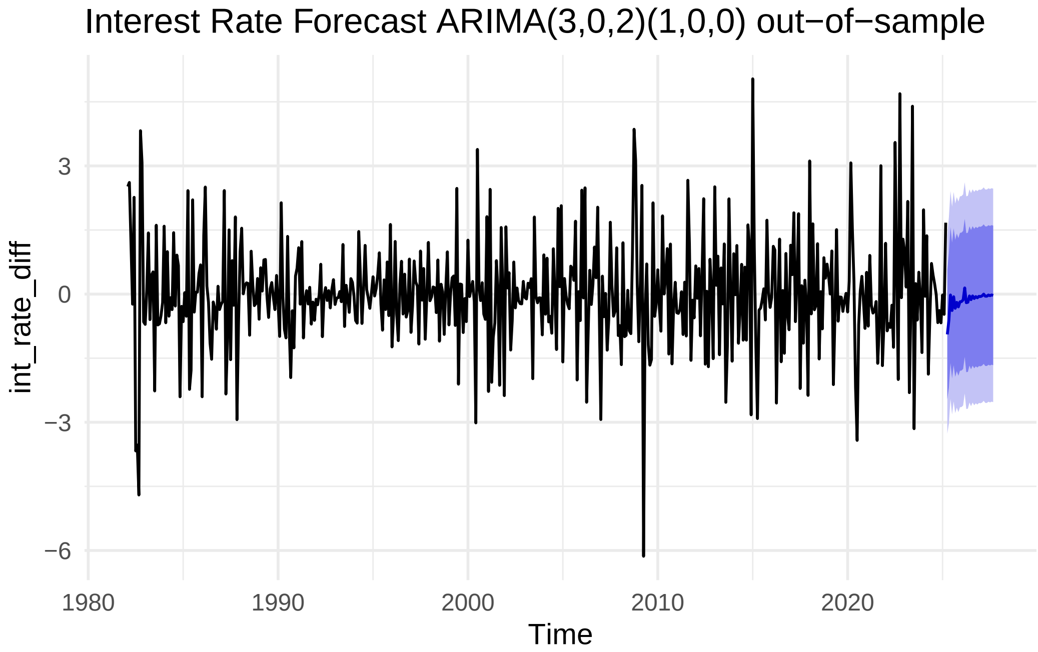
```r
accuracy(fc3_intd, test_intd)
```

```
                     ME     RMSE      MAE        MPE      MAPE      MASE
Training set 0.04601330 1.461404 1.063796   240.9021  706.1909 0.9001127
Test set     0.03491086 3.173292 2.443390 -2336.3812 3525.2540 2.0674318
                      ACF1 Theil's U
Training set  0.0023348606        NA
Test set     -0.0005521466   1.03504
```

Among the three, the first model (ARIMA(3,0,2)(1,0,0)) offers the most balanced fit with lower RMSE and MAE.

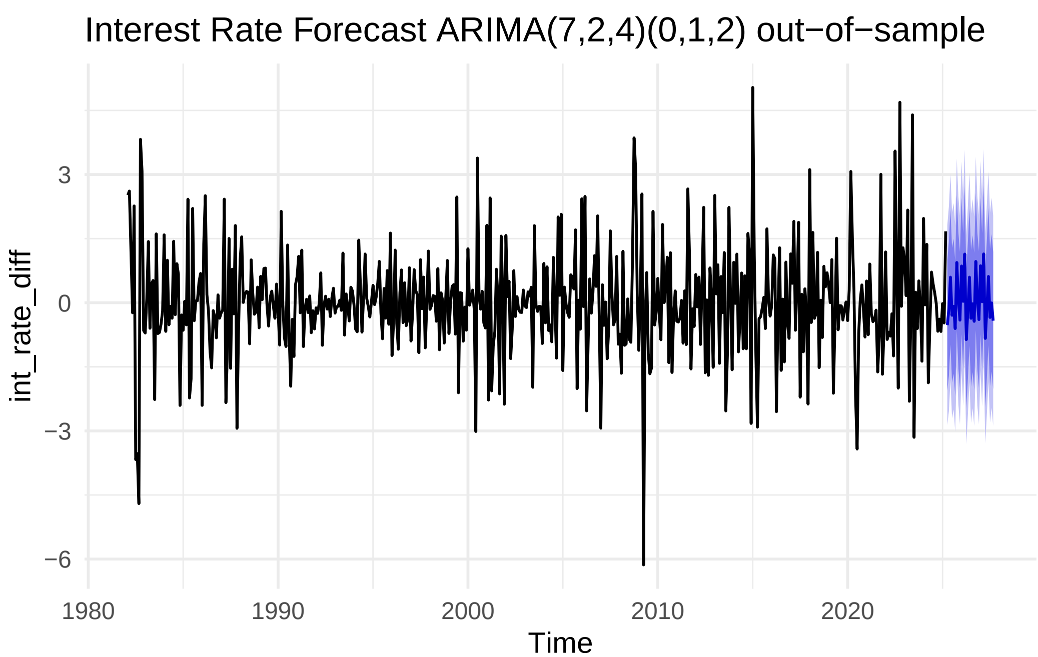
2.2.6: Short-term Interest Rate out-of-sample

```r
#model 1
model1_intd_full <- arima(int_rate_diff,order=c(3, 0, 2),seasonal=list(order=c(1, 0, 0), per
fc1_intd_full <- forecast(model1_intd_full, h=30)
autoplot(fc1_intd_full) +
  ggtitle("Interest Rate Forecast ARIMA(3,0,2)(1,0,0) out-of-sample") +
  theme_minimal()
```



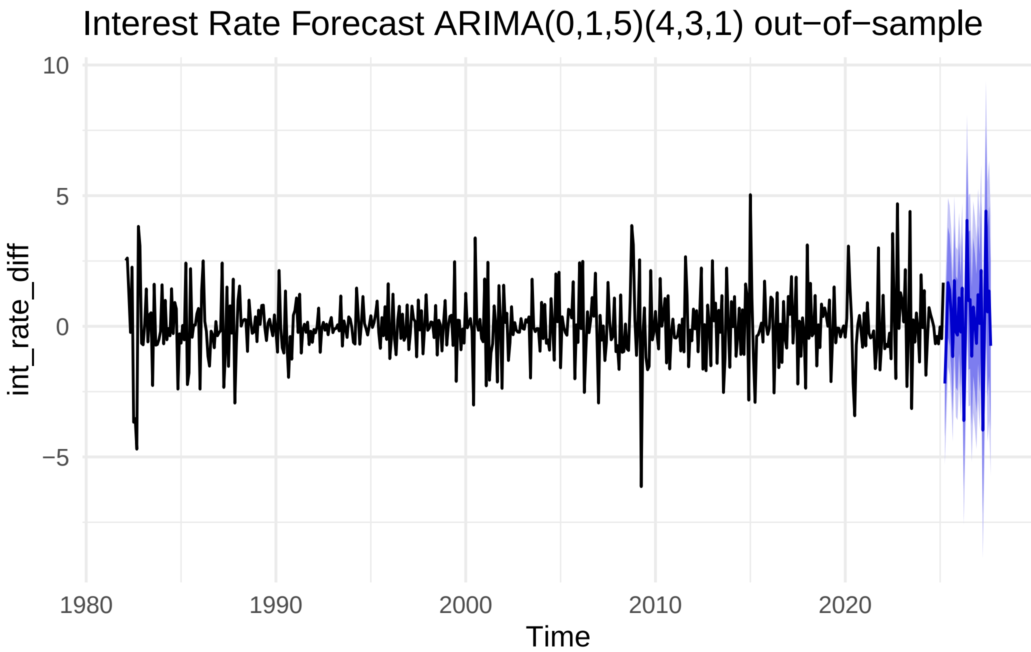
The out-of-sample forecast remains stable and coherent with past dynamics.

```
#model 2
model2_intd_full <- arima(int_rate_diff,order=c(7, 2, 4),seasonal=list(order=c(0, 1, 2), per
fc2_intd_full <- forecast(model2_intd_full, h=30)
autoplot(fc2_intd_full) +
  ggtitle("Interest Rate Forecast ARIMA(7,2,4)(0,1,2) out-of-sample") +
  theme_minimal()
```



The second model shows too much variability and a tendency to diverge.

```
#model 3
model3_intd_full <- arima(int_rate_diff,order=c(0, 1, 5),seasonal=list(order=c(4, 3, 1), per
fc3_intd_full <- forecast(model3_intd_full, h=30)
autoplot(fc3_intd_full) +
  ggtitle("Interest Rate Forecast ARIMA(0,1,5)(4,3,1) out-of-sample") +
  theme_minimal()
```



The third model forecasts remain unstable and excessively wide, suggesting an overfitted model.

Problem 3

3.1: Question 1

Using log-returns has several advantages:

- log-returns are time-additive, which is useful when aggregating over different periods
- if prices follow a log-normal distribution, log-returns will follow a normal distribution, making modeling easier
- log-returns are unbounded on both sides, whereas simple returns are bounded below by -1
- they approximate percentage changes and are easier to interpret in financial terms
- log transformation often stabilizes variance, helping with the stationarity assumption

Technically, log-returns are calculated as the difference of the logarithm of consecutive prices. This can be implemented in R using:

log_returns <- diff(log(data$Dernier))

3.2: Question 2

```
#load the dataset
data <- read.csv("IBEX35k.csv", sep = ",", header = TRUE, stringsAsFactors = FALSE)
#show the first rows to check the import
head(data)
```

```
        Date   Dernier      Ouv. X.Plus.Haut  Plus.Bas    Vol. Variation..
1 10/03/2025 13.063,87 13.309,00   13.309,00 13.048,35  86,80M      -1,46%
2 07/03/2025 13.257,10 13.142,90   13.291,30 13.114,70 134,51M       0,17%
3 06/03/2025 13.234,20 13.274,00   13.319,60 13.099,90 176,62M       0,15%
4 05/03/2025 13.214,00 13.230,20   13.348,20 13.206,40 177,35M       1,40%
5 04/03/2025 13.031,70 13.248,20   13.282,00 12.997,10 195,20M      -2,55%
6 03/03/2025 13.373,10 13.349,30   13.446,40 13.218,70 153,78M       0,19%
```

The dataset shows six recent observations of IBEX35. Variables include Date, Dernier (closing price), Volume, and Variation. Note that the database contains numbers formatted as in European conventions (e.g., “13.063,87”), which are not directly interpretable by R as numeric values.

```
#convert Date column into Date format
data$Date <- as.Date(data$Date, format = "%d/%m/%Y")

#clean the price column
#remove thousands separator
data$Dernier <- gsub("\\.", "", data$Dernier)
#replace commas by dots
data$Dernier <- gsub(",", ".", data$Dernier)
#convert to numeric
data$Dernier <- as.numeric(data$Dernier)
```

The ‘Dernier’ column is successfully cleaned and converted into numeric format by removing thousand separators and adjusting decimal points.

```
#extract closing prices
prices <- data$Dernier
#compute log returns
log_returns <- diff(log(prices))
```

```r
#add log returns to the dataframe
data$log_returns <- c(NA, log_returns)
#check the result
head(data)
```

```
        Date  Dernier      Ouv. X.Plus.Haut  Plus.Bas    Vol. Variation..
1 2025-03-10 13063.87 13.309,00   13.309,00 13.048,35  86,80M      -1,46%
2 2025-03-07 13257.10 13.142,90   13.291,30 13.114,70 134,51M       0,17%
3 2025-03-06 13234.20 13.274,00   13.319,60 13.099,90 176,62M       0,15%
4 2025-03-05 13214.00 13.230,20   13.348,20 13.206,40 177,35M       1,40%
5 2025-03-04 13031.70 13.248,20   13.282,00 12.997,10 195,20M      -2,55%
6 2025-03-03 13373.10 13.349,30   13.446,40 13.218,70 153,78M       0,19%
   log_returns
1           NA
2  0.014682853
3 -0.001728870
4 -0.001527514
5 -0.013892023
6  0.025860376
```

The first computed log-returns are displayed. The values range between small positive and negative percentages.

```r
#compute descriptive statistics
mean_log_return <- mean(log_returns, na.rm = TRUE)
variance_log_return <- var(log_returns, na.rm = TRUE)
skewness_log_return <- skewness(log_returns, na.rm = TRUE)
kurtosis_log_return <- kurtosis(log_returns, na.rm = TRUE)

#ACF and PACF
acf_log_returns <- acf(log_returns, plot = FALSE)
pacf_log_returns <- pacf(log_returns, plot = FALSE)

#print the statistics
cat("Mean of log-returns:", mean_log_return, "\n")
```

```
Mean of log-returns: -7.054828e-05
```

```r
cat("Variance of log-returns:", variance_log_return, "\n")
```

```
Variance of log-returns: 0.0001498984
```

```r
cat("Skewness of log-returns:", skewness_log_return, "\n")
```

```
Skewness of log-returns: 1.41106
```

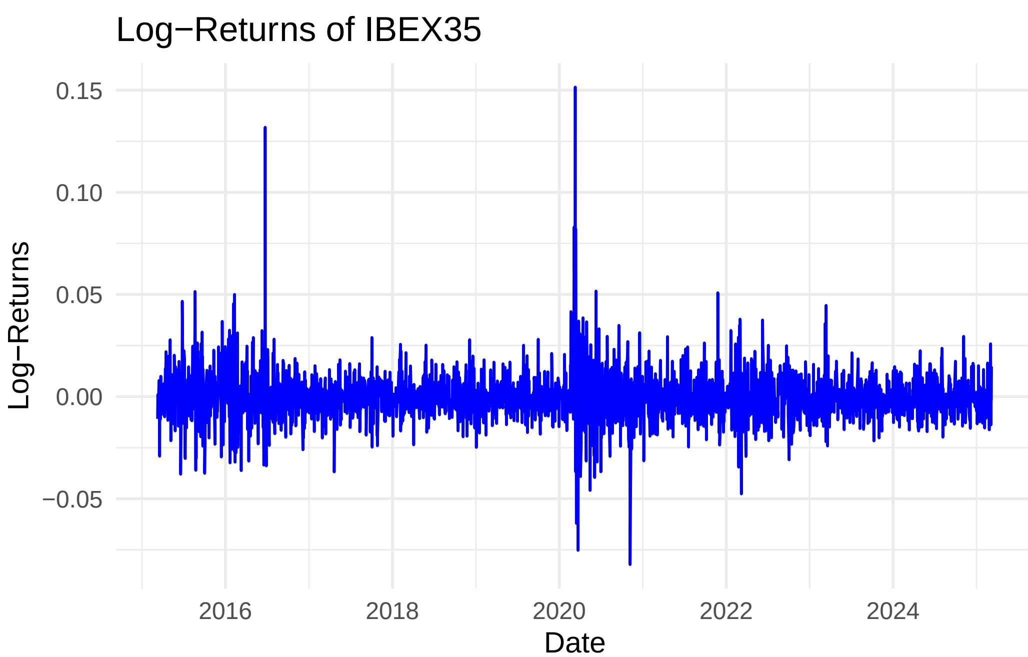
```r
cat("Kurtosis of log-returns:", kurtosis_log_return, "\n")
```

```
Kurtosis of log-returns: 18.0151
```

The mean is close to zero. The variance is small. The skewness is positive and the kurtosis is strongly positive.

```r
#plot log-returns
ggplot(data, aes(x=Date, y=log_returns)) +
  geom_line(color="blue") +
  labs(title="Log-Returns of IBEX35", x="Date", y="Log-Returns") +
  theme_minimal()
```

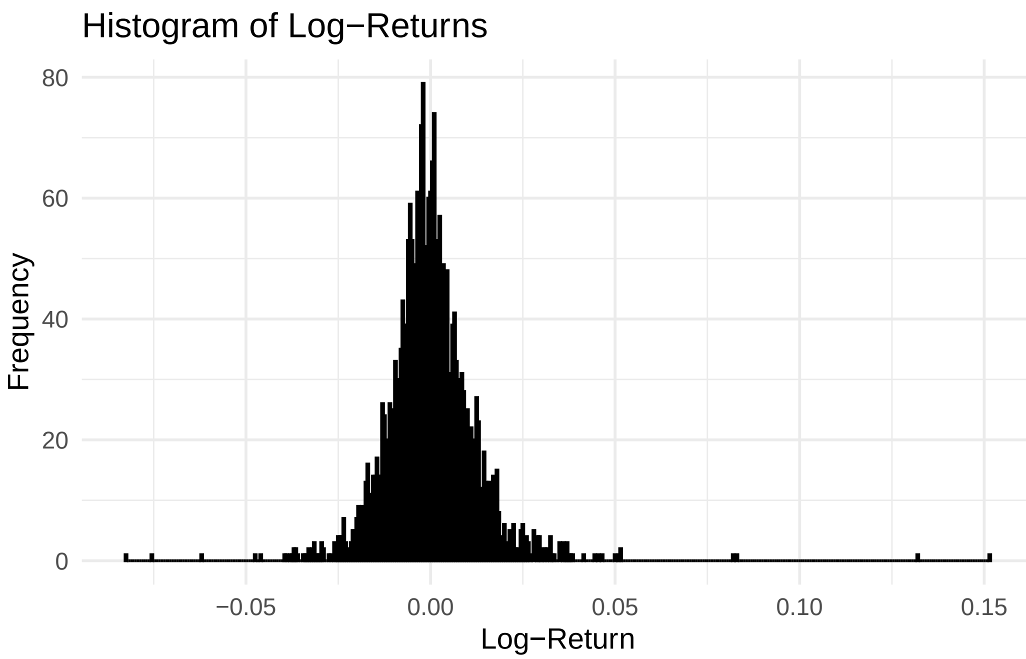
```
Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_line()`).
```



The plot shows frequent oscillations around zero with visible periods of larger deviations.

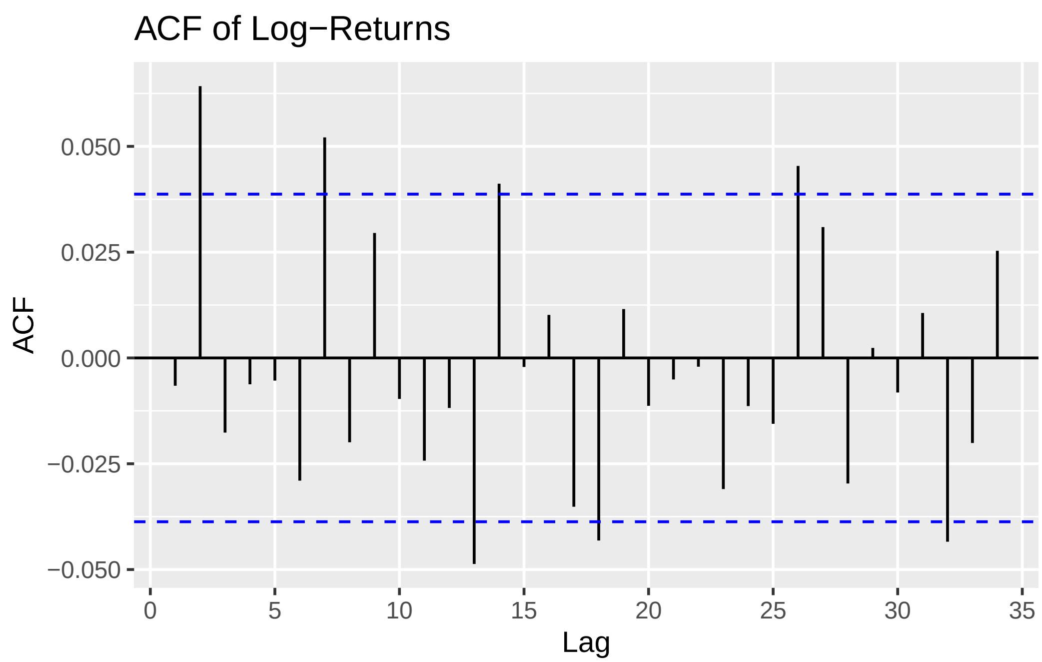
```
#plot histogram of log-returns
ggplot(data, aes(x=log_returns)) +
  geom_histogram(binwidth=0.0005, fill="blue", color="black", alpha=0.7) +
  labs(title="Histogram of Log-Returns", x="Log-Return", y="Frequency") +
  theme_minimal()
```

```
Warning: Removed 1 row containing non-finite outside the scale range
(`stat_bin()`).
```



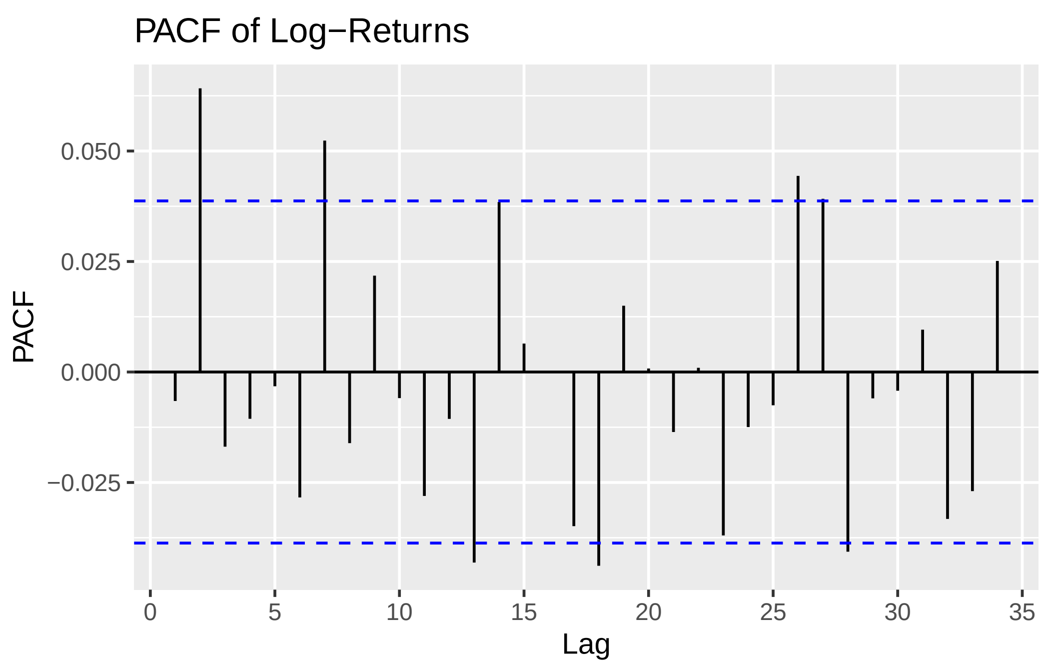
The histogram shows a centered distribution with heavy tails.

```
#plot ACF
ggAcf(log_returns) + ggtitle("ACF of Log-Returns")
```



The ACF of log-returns shows no significant autocorrelation.

```
#plot PACF
ggPacf(log_returns) + ggtitle("PACF of Log-Returns")
```



The PACF does not show significant partial autocorrelations.

3.3: Question 3

```r
#ADF test with lag selection by AIC
adf_test <- ur.df(log_returns, type = "drift", selectlags = "AIC")
summary(adf_test)
```

```
###############################################
# 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
-0.081701 -0.006278 -0.000458  0.005660  0.146091

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -7.182e-05  2.415e-04  -0.297  0.76621
z.lag.1     -9.419e-01  2.799e-02 -33.651  < 2e-16 ***
z.diff.lag  -6.420e-02  1.972e-02  -3.255  0.00115 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01222 on 2559 degrees of freedom
Multiple R-squared:  0.5052,    Adjusted R-squared:  0.5048
F-statistic:  1307 on 2 and 2559 DF,  p-value: < 2.2e-16


Value of test-statistic is: -33.6507 566.1865

Critical values for test statistics:
      1pct  5pct 10pct
```

```
tau2 -3.43 -2.86 -2.57
phi1  6.43  4.59  3.78
```

The ADF test provides a test statistic of -33.65 and a p-value close to zero. Critical values are -3.43 (1%), -2.86 (5%), and -2.57 (10%).The ADF statistic is -33.65, lower than the critical values, indicating stationarity.

```
#select ARMA(p,q) using auto.arima()
mean_equ <- auto.arima(log_returns, seasonal=FALSE)
summary(mean_equ)
```

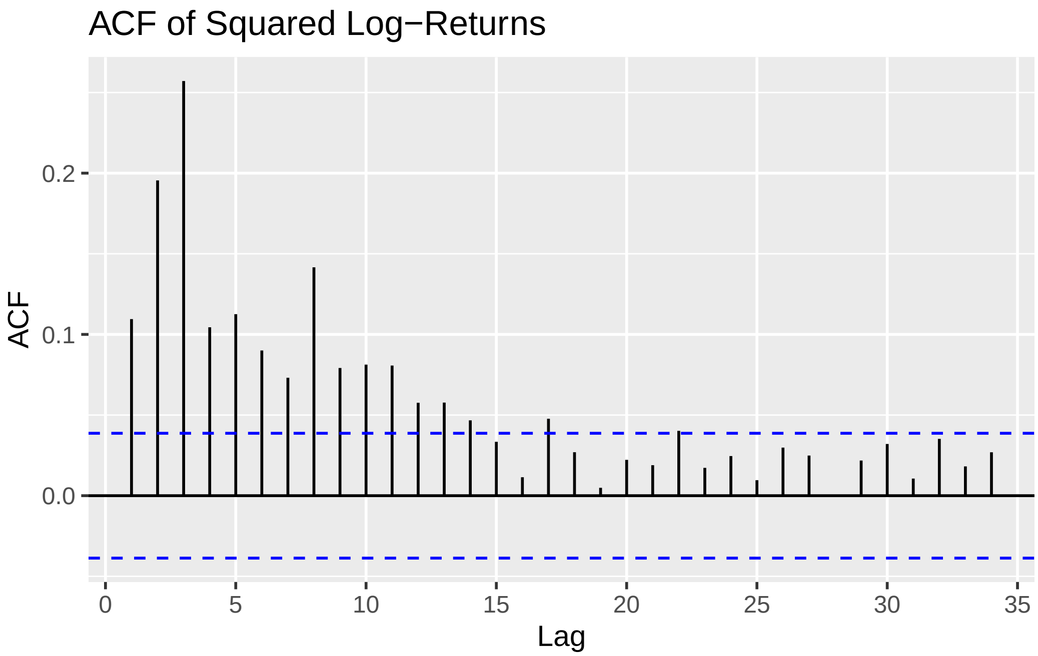
```
Series: log_returns
ARIMA(0,0,2) with zero mean

Coefficients:
          ma1     ma2
      -0.0041  0.0649
s.e.   0.0197  0.0197

sigma^2 = 0.0001493:  log likelihood = 7656.41
AIC=-15306.83   AICc=-15306.82   BIC=-15289.28

Training set error measures:
                        ME       RMSE         MAE MPE MAPE      MASE
Training set -6.671943e-05 0.01221536 0.008383967 NaN  Inf 0.6979578
                     ACF1
Training set -0.001090143
```

```
#extract orders
best_arma_order <- arimaorder(mean_equ)[1:2]
print(paste("Selected ARMA order:", paste(best_arma_order, collapse=",")))
```

```
[1] "Selected ARMA order: 0,0"
```

The estimated ARMA order is (0,0).

```
#plot squared returns
ggAcf(log_returns^2) + ggtitle("ACF of Squared Log-Returns")
```



The ACF of squared returns shows several significant lags, justifying the use of a GARCH model to capture conditional heteroskedasticity.

```
#grid search
p_range <- 1:3
q_range <- 1:3
aic_values <- matrix(NA, nrow=3, ncol=3)
bic_values <- matrix(NA, nrow=3, ncol=3)

for(i in 1:3){
  for(j in 1:3){
    spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(i,j)),
                       mean.model = list(armaOrder=best_arma_order, include.mean=TRUE),
                       distribution.model = "norm")
    fit <- tryCatch(ugarchfit(spec=spec, data=log_returns), error=function(e) NULL)
    if(!is.null(fit)){
      loglik <- fit@fit$LLH
      n <- length(log_returns)
      k <- length(fit@fit$coef)
      aic_values[i,j] <- -2*loglik + 2*k
      bic_values[i,j] <- -2*loglik + k*log(n)
    }
  }
```

```
}

#find best
best_aic <- which(aic_values == min(aic_values, na.rm=TRUE), arr.ind=TRUE)
best_bic <- which(bic_values == min(bic_values, na.rm=TRUE), arr.ind=TRUE)

print(paste("Best GARCH order (AIC):", paste(c(best_aic[1], best_aic[2]), collapse=",")))
```

```
[1] "Best GARCH order (AIC): 2,1"
```

```
print(paste("Best GARCH order (BIC):", paste(c(best_bic[1], best_bic[2]), collapse=",")))
```

```
[1] "Best GARCH order (BIC): 1,1"
```

The grid search reveals that GARCH(2,1) minimizes AIC while GARCH(1,1) minimizes BIC.

```
#final model based on BIC
spec_final <- ugarchspec(variance.model = list(model="sGARCH", garchOrder= c(best_bic[1], bes
                         mean.model = list(armaOrder=best_arma_order, include.mean=TRUE),
                         distribution.model="norm")
fit_final <- ugarchfit(spec=spec_final, data=log_returns)
summary(fit_final)
```

```
   Length     Class      Mode
        1 uGARCHfit        S4
```

The GARCH(1,1) model is fitted and the output provides estimated coefficients and standard errors.

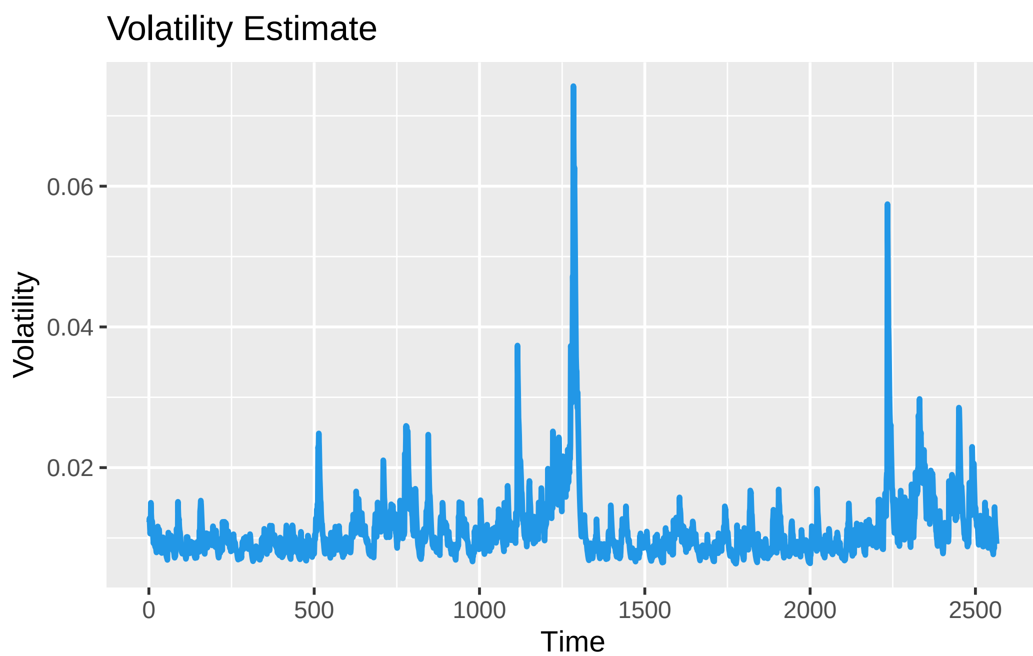
3.4: Question 4

```
#check residuals
Box.test(fit_final@fit$residuals, lag=10, type="Ljung-Box")
```

```

	Box-Ljung test

data:  fit_final@fit$residuals
X-squared = 24.334, df = 10, p-value = 0.006762
```

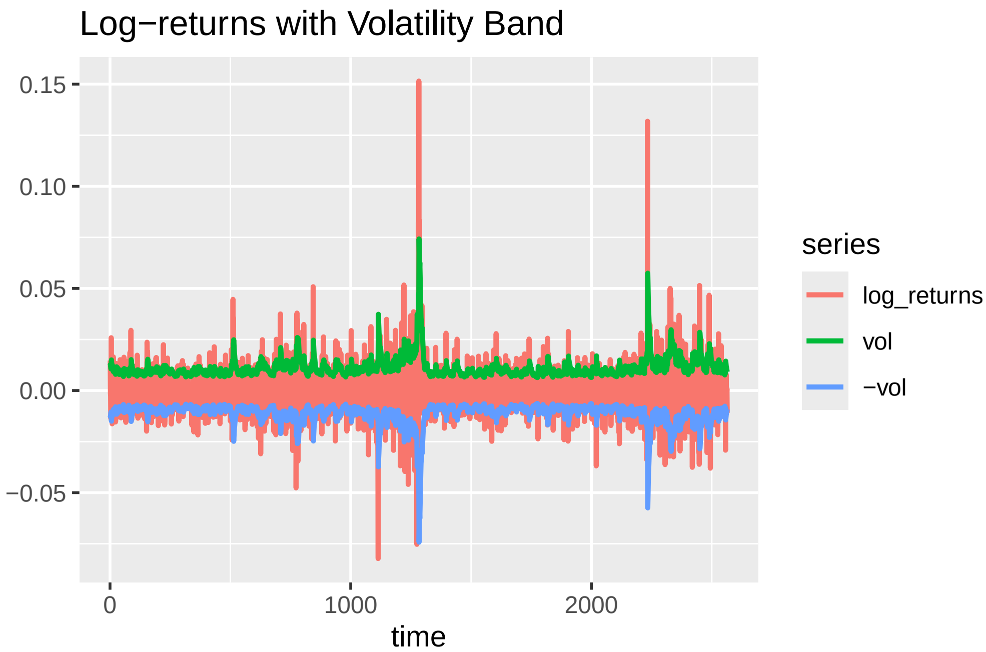
The Ljung-Box test yields a p-value of 0.0067, showing residual autocorrelation is still present.

```
#plot volatility
vol <- ts(fit_final@fit$sigma)
autoplot(vol,col = 4,ylab = "Volatility", lwd = 1) + ggtitle("Volatility Estimate")
```



The volatility series shows clusters of periods with higher volatility.

```
#plot combined
autoplot(cbind(log_returns, vol, -vol),xlab = "time",ylab = "",main = "GARCH(1,1)",lwd=0.9) +
```



The combined plot shows log-returns oscillating mostly within the ±1 conditional standard deviation bands.