GDP_forecasting

JAHNAVI GANGU

2024-09-10

```
# Load libraries
library(tidyverse)
```

```
## Warning: package 'dplyr' was built under R version 4.2.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(fredr)
library(forecast)
```

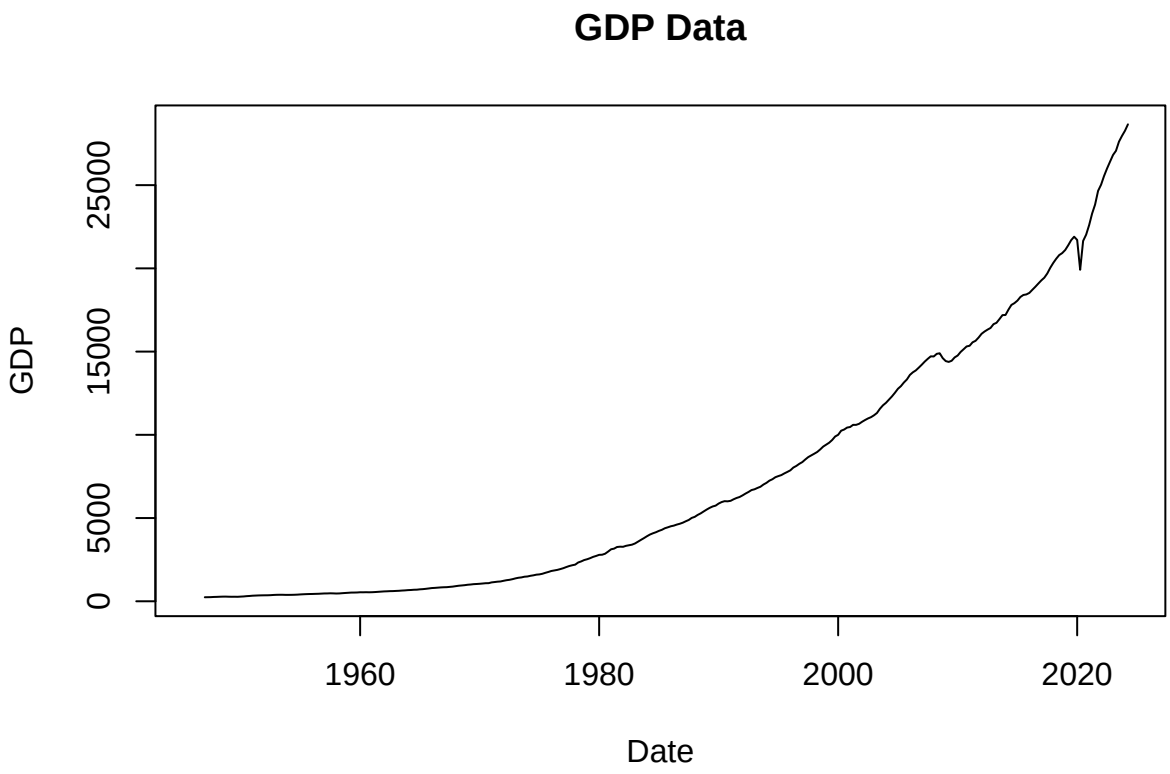
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(imputeTS)
```

```
# Set API key for FRED data access
fredr_set_key("c387f7cbc3f36a5a52a03d391d17e253")
```

```
# Retrieve GDP data from FRED database
GDP_data <- fredr(series_id = "GDP")
```

```
# Plot GDP data
plot(GDP_data$date, GDP_data$value, type = "l", xlab = "Date", ylab = "GDP", main = "GDP Data")
```




```
# Display structure and first few rows of GDP data
head(GDP_data)
```

```
## # A tibble: 6 x 5
##   date       series_id value realtime_start realtime_end
##   <date>     <chr>     <dbl> <date>         <date>
## 1 1946-01-01 GDP         NA  2024-08-29     2024-08-29
## 2 1946-04-01 GDP         NA  2024-08-29     2024-08-29
## 3 1946-07-01 GDP         NA  2024-08-29     2024-08-29
## 4 1946-10-01 GDP         NA  2024-08-29     2024-08-29
## 5 1947-01-01 GDP        243. 2024-08-29     2024-08-29
## 6 1947-04-01 GDP        246. 2024-08-29     2024-08-29
```

```
str(GDP_data)
```

```
## tibble [314 x 5] (S3: tbl_df/tbl/data.frame)
##  $ date          : Date[1:314], format: "1946-01-01" "1946-04-01" ...
##  $ series_id     : chr [1:314] "GDP" "GDP" "GDP" "GDP" ...
##  $ value         : num [1:314] NA NA NA NA 243 ...
##  $ realtime_start: Date[1:314], format: "2024-08-29" "2024-08-29" ...
##  $ realtime_end  : Date[1:314], format: "2024-08-29" "2024-08-29" ...
```

```
# Extract numeric GDP values
numeric_GDP_data <- GDP_data %>%
  select(value)
```

```
# Check for missing values
missing_values <- sum(is.na(numeric_GDP_data$value))
```

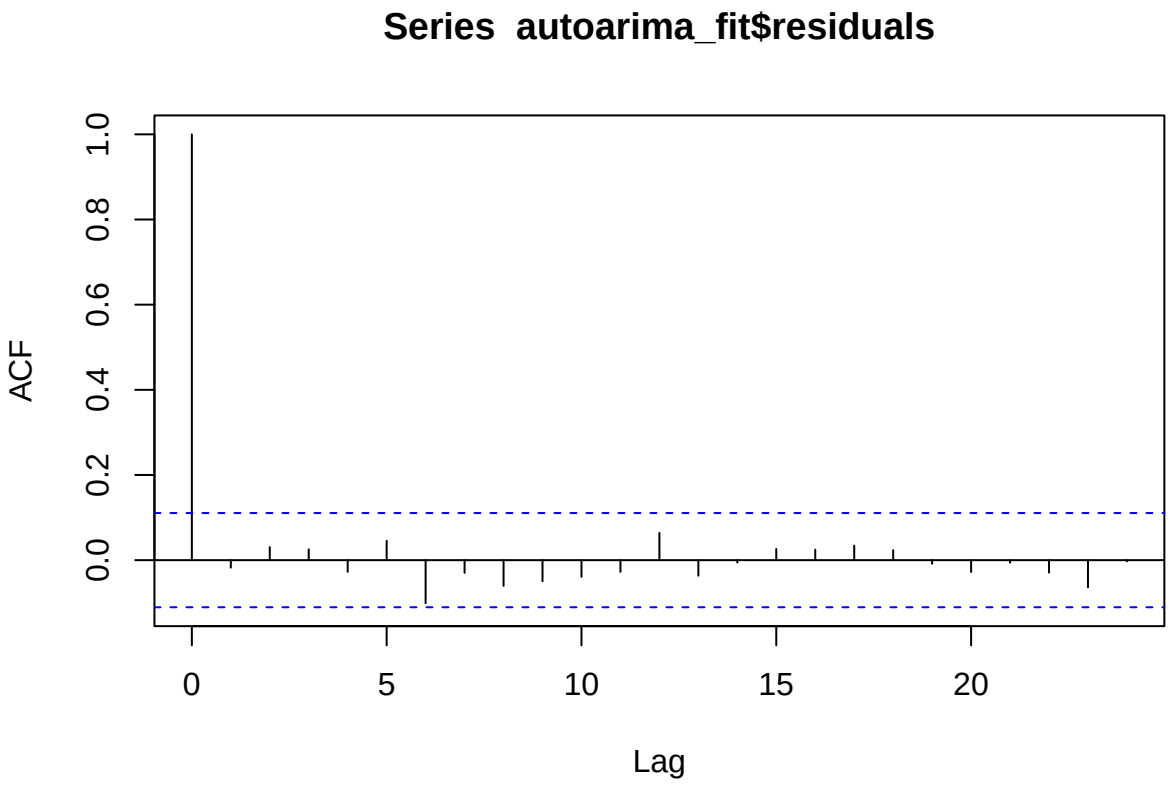
```r
# Impute missing values if any
if (missing_values > 0) {
  numeric_GDP_data$value <- na_interpolation(numeric_GDP_data$value)
}
```

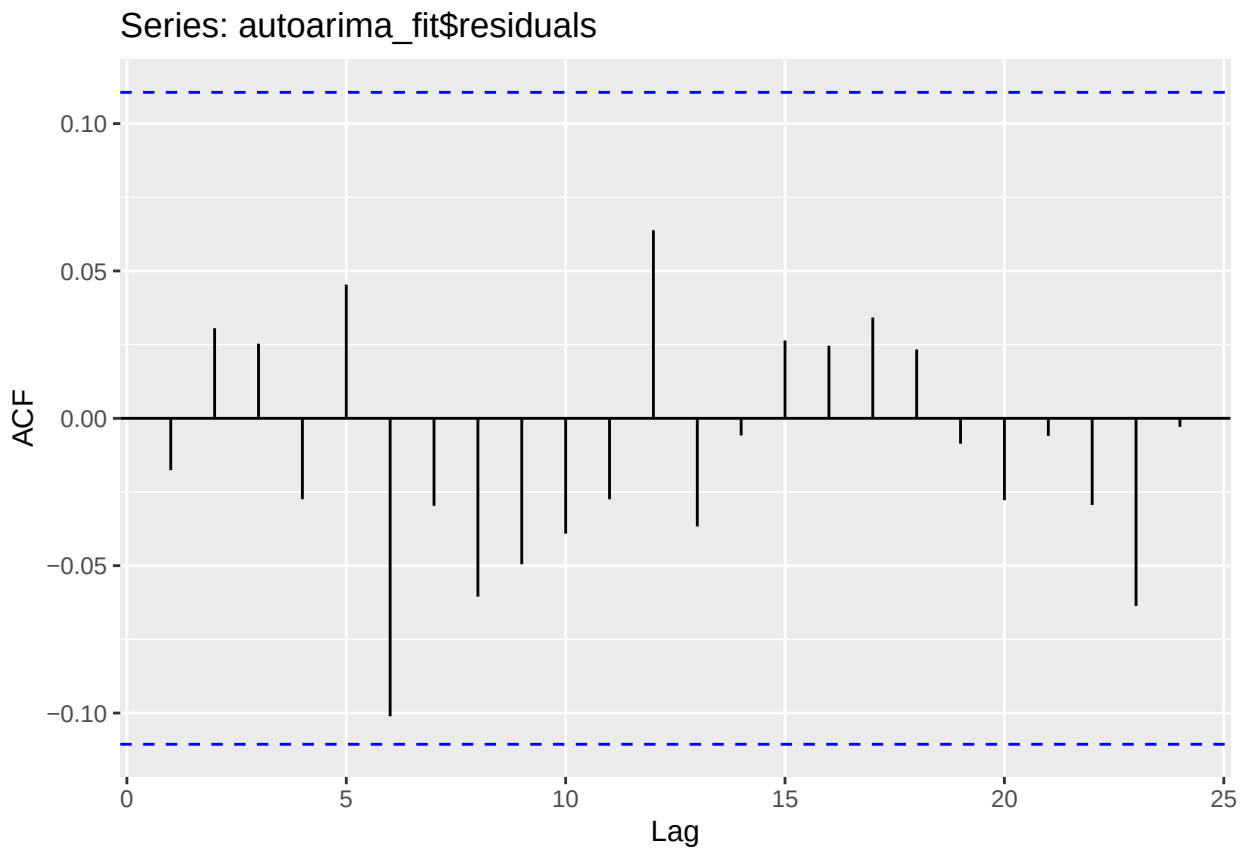
```r
# Define target variable
target_variable <- numeric_GDP_data$value
```

```r
# Fit AutoARIMA model
autoarima_fit <- auto.arima(target_variable)

# Print model summary
summary(autoarima_fit)
```

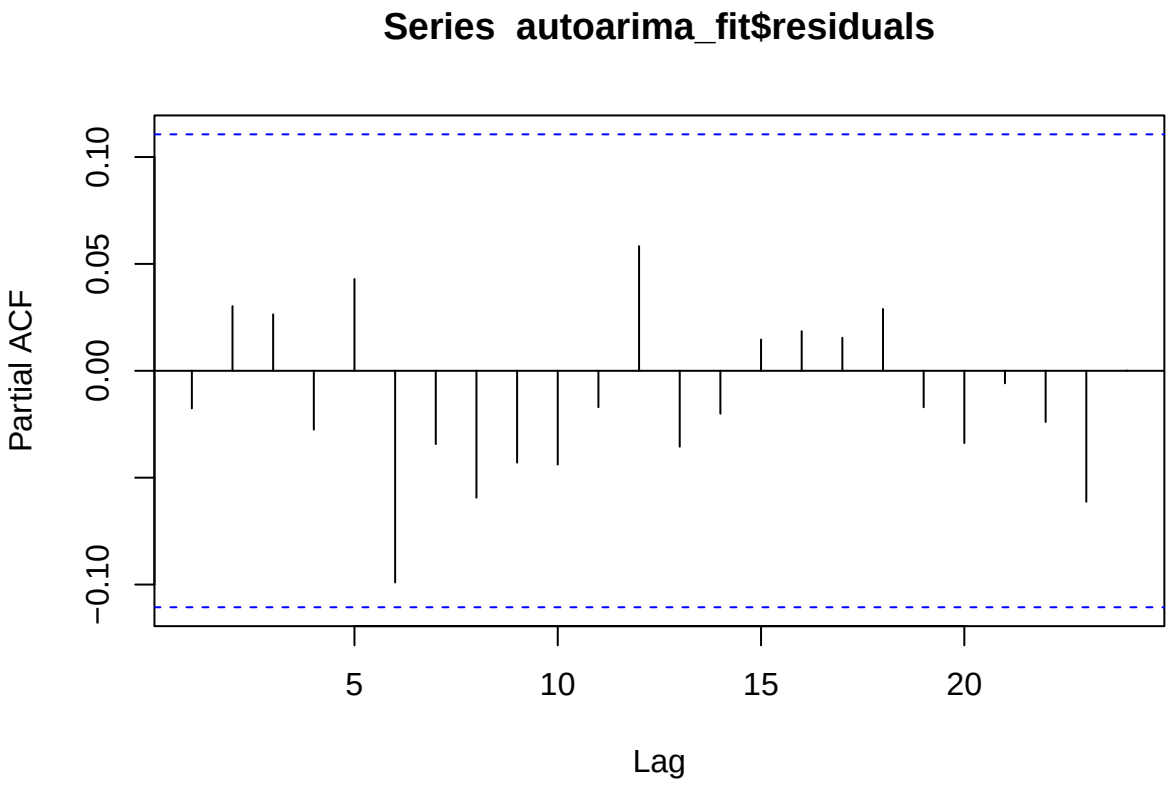
```
## Series: target_variable
## ARIMA(0,2,2)
##
## Coefficients:
##           ma1     ma2
##       -1.0433  0.1215
## s.e.   0.0536  0.0532
##
## sigma^2 = 28025:  log likelihood = -2040.29
## AIC=4086.59   AICc=4086.67   BIC=4097.82
##
## Training set error measures:
##                     ME     RMSE     MAE       MPE      MAPE      MASE
## Training set 14.87591 166.3359 52.7672 0.2869568 0.8292295 0.4922959
##                      ACF1
## Training set -0.01759396
```

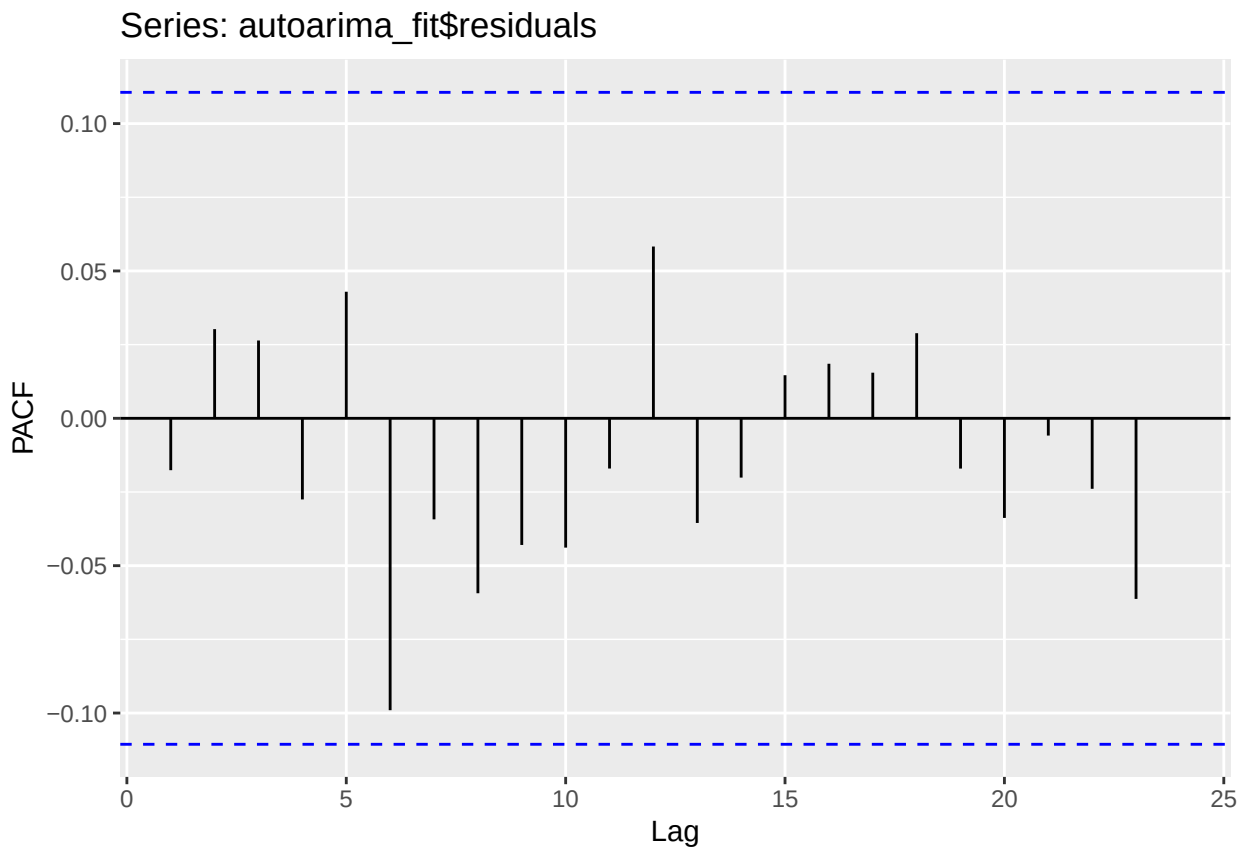
```r
# Plot ACF and PACF of AutoARIMA model residuals
autoplot(acf(autoarima_fit$residuals))
```


Series autoarima_fit$residuals
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
Lag

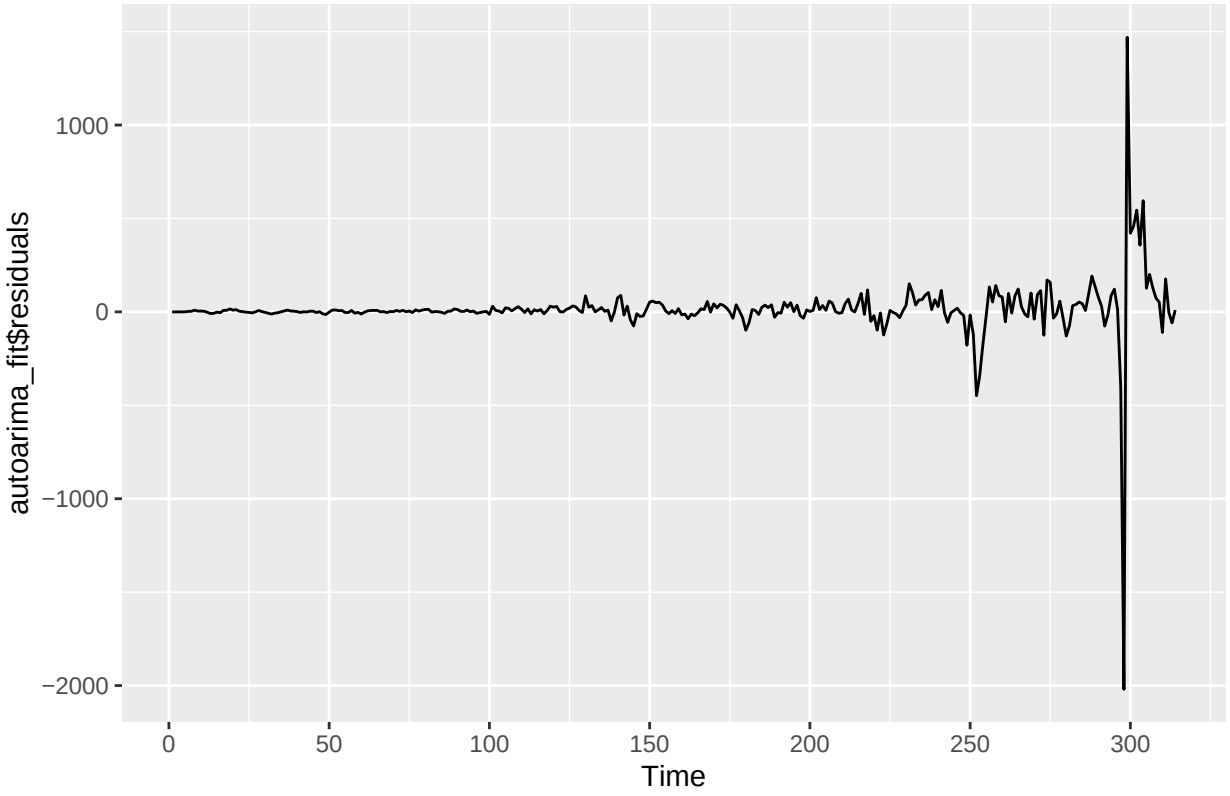



```
autoplot(pacf(autoarima_fit$residuals))
```


Series autoarima_fit$residuals
Partial ACF
0.10
0.05
0.00
−0.10
5
10
15
20
Lag




```
autoplot(autoarima_fit$residuals)
```

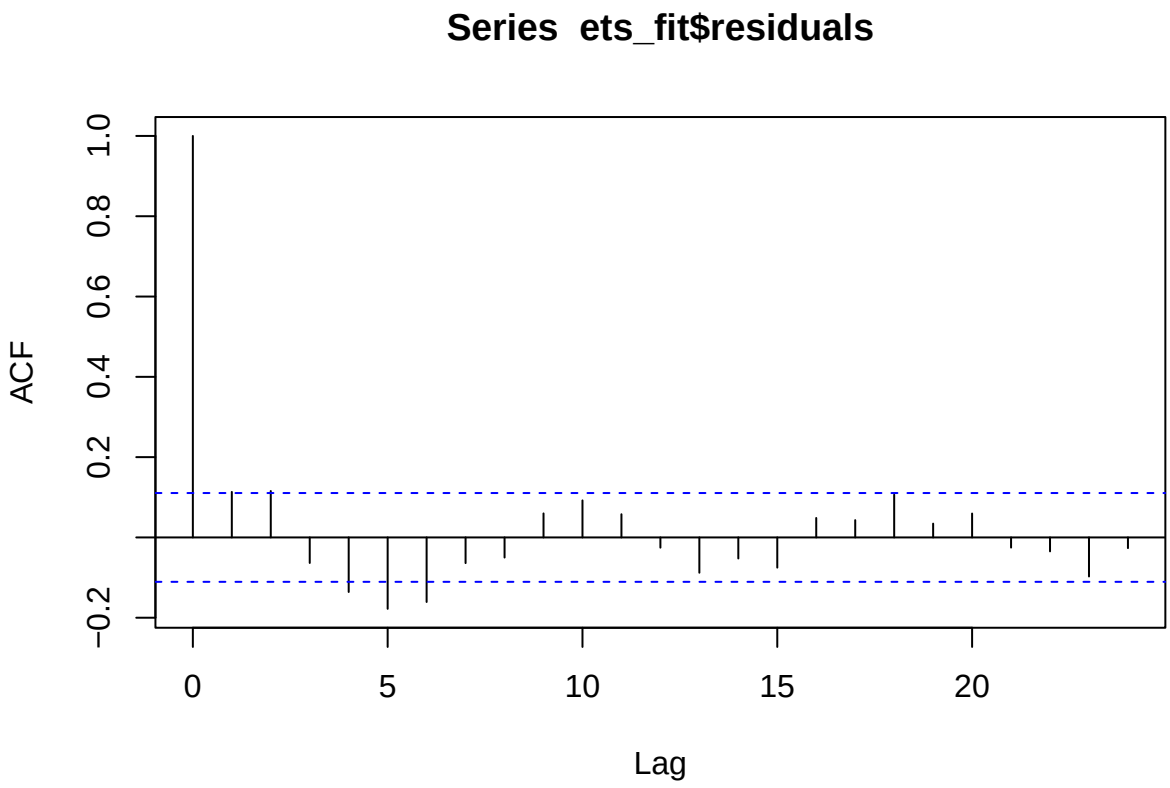


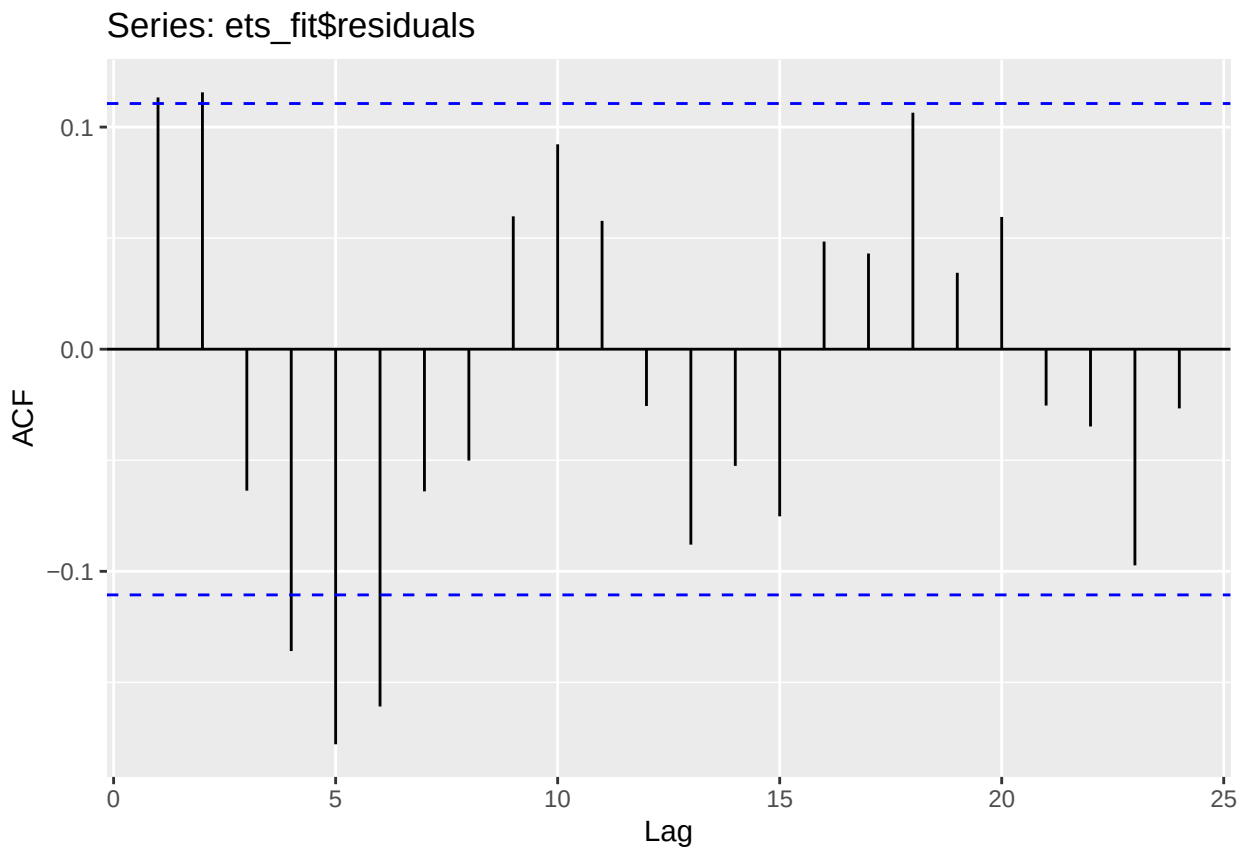

```r
# Fit ETS model
ets_fit <- ets(target_variable)

# Print model summary
summary(ets_fit)
```

```
## ETS(M,A,N)
##
## Call:
##  ets(y = target_variable)
##
##   Smoothing parameters:
##     alpha = 0.9999
##     beta  = 0.1279
##
##   Initial states:
##     l = 233.1786
##     b = 3.1984
##
##   sigma:  0.0127
##
##      AIC     AICc      BIC
## 4129.038 4129.233 4147.785
##
## Training set error measures:
##                    ME    RMSE      MAE       MPE      MAPE      MASE       ACF1
```

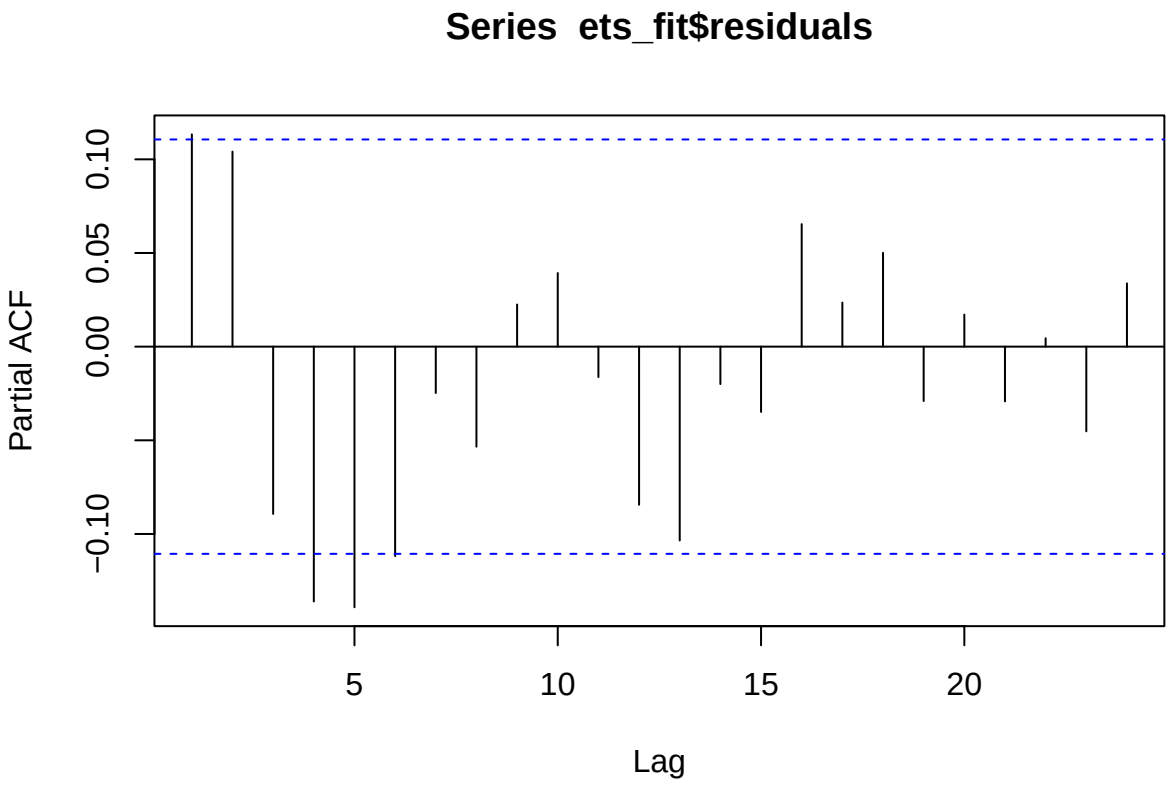
```
## Training set 9.581936 169.271 49.38476 0.1722281 0.7850069 0.4607392 -0.1760449
```

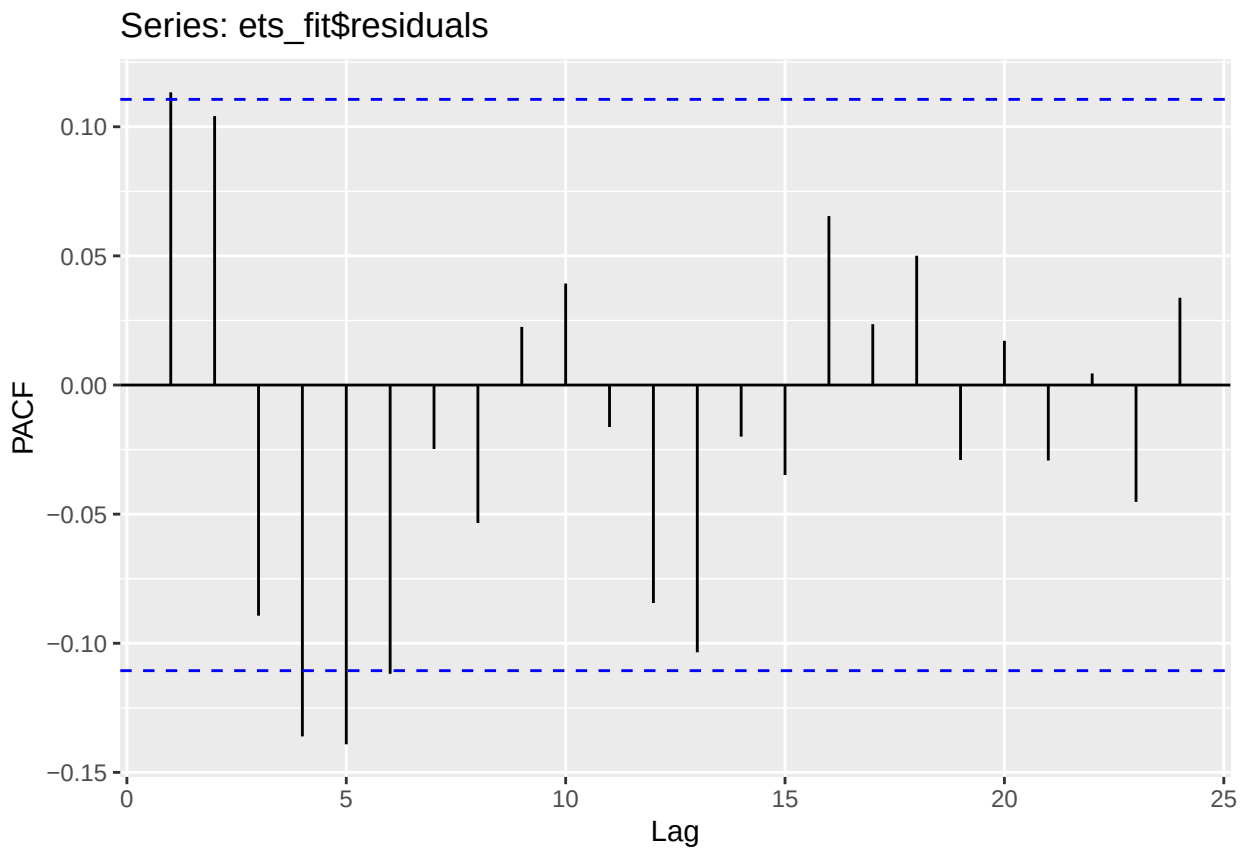
```r
# Plot ACF and PACF of ETS model residuals
autoplot(acf(ets_fit$residuals))
```



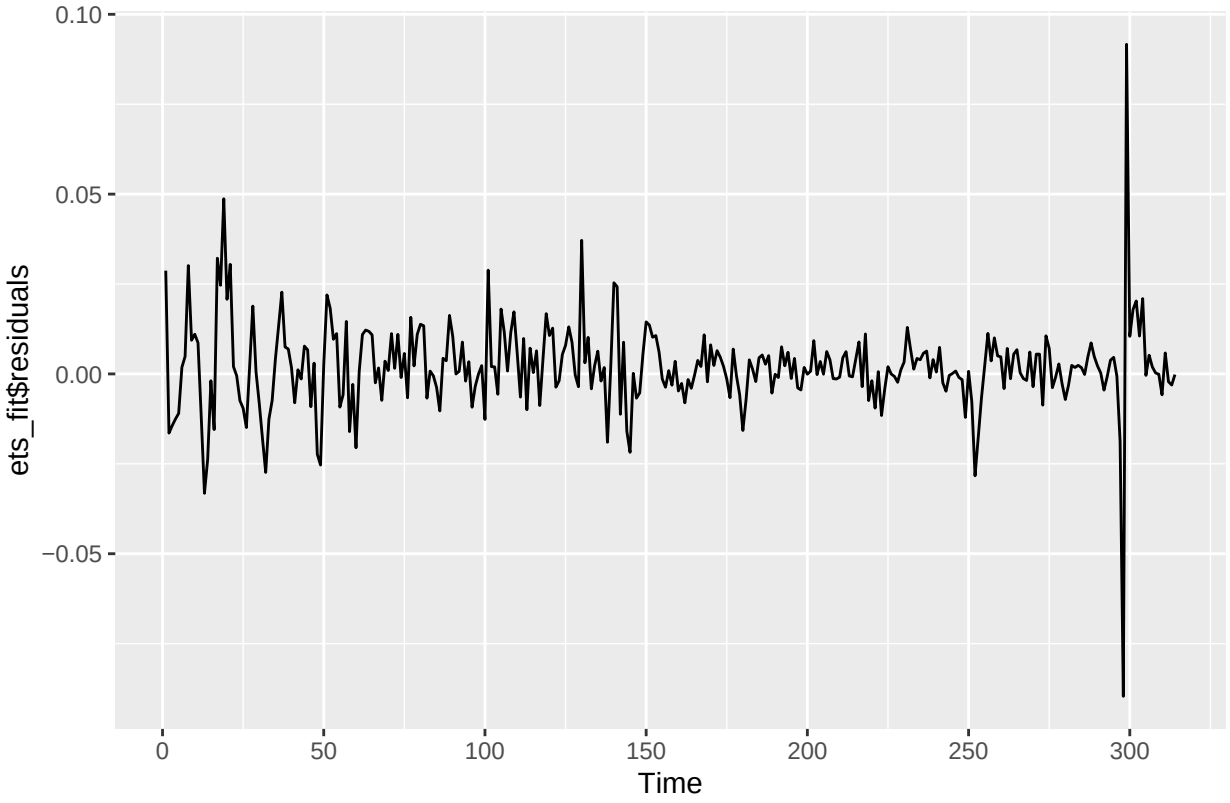



```
autoplot(pacf(ets_fit$residuals))
```


Series ets_fit$residuals
Partial ACF
0.10
0.05
0.00
−0.10
5
10
15
20
Lag

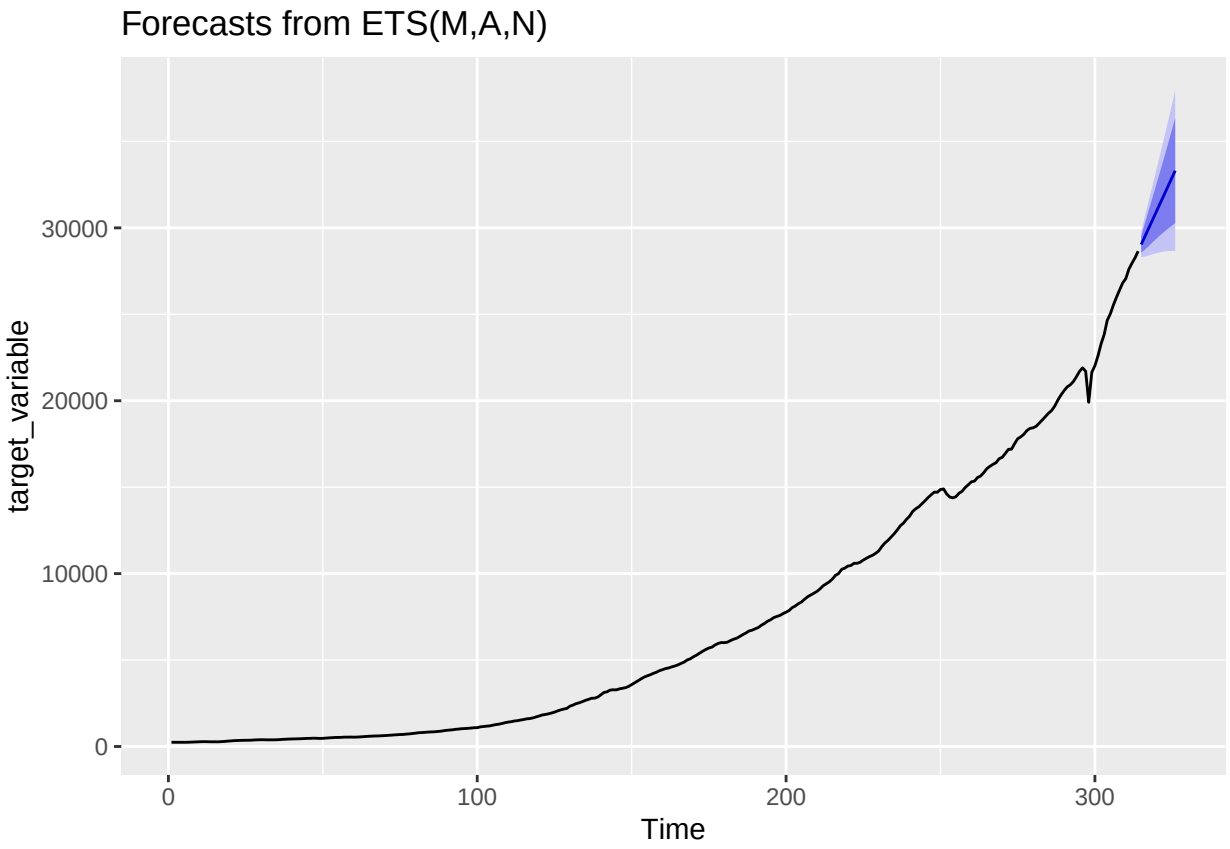



```
autoplot(ets_fit$residuals)
```

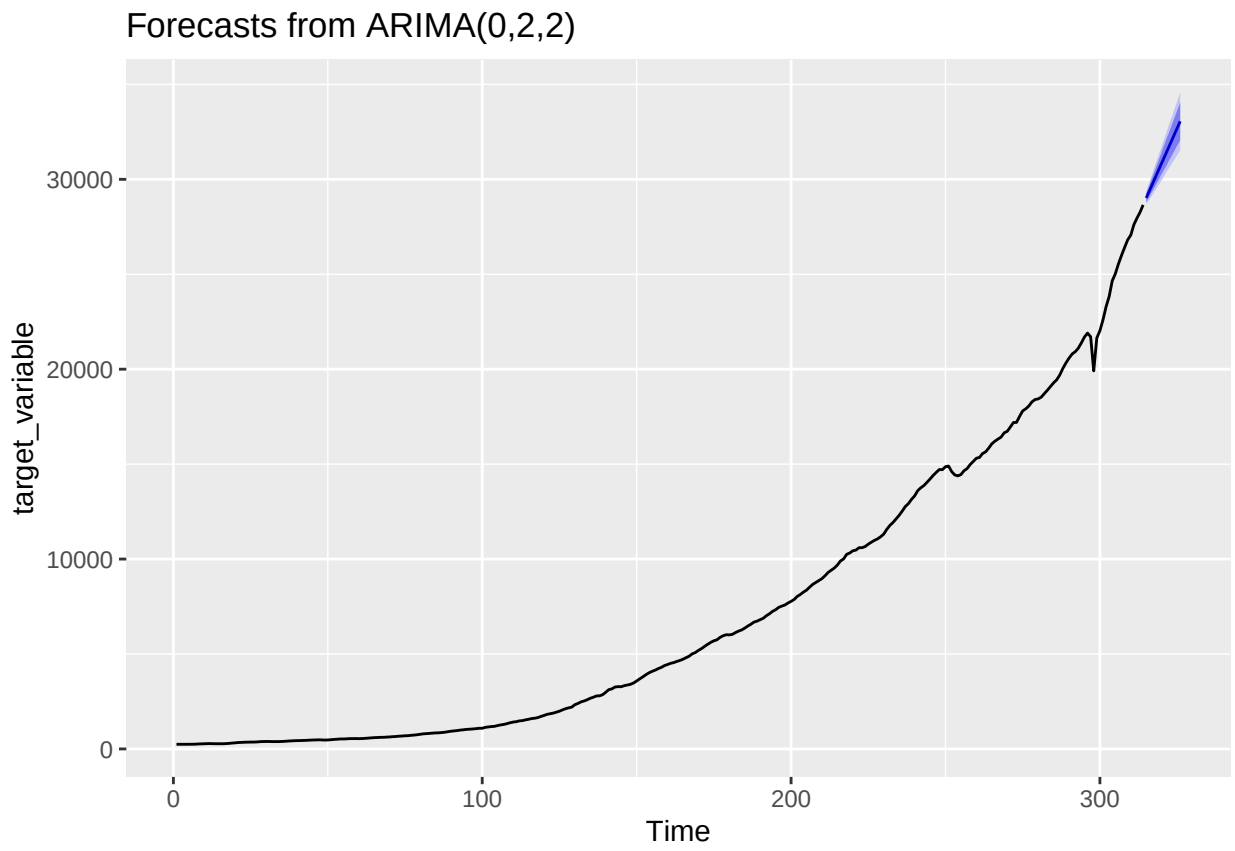



```
# Define forecast horizon
forecast_horizon <- 12
```

```
# Forecast with ETS model
ets_forecast <- forecast(ets_fit, h = forecast_horizon)

# Plot ETS forecast
autoplot(ets_forecast)
```




```
# Forecast with AutoARIMA model
autoarima_forecast <- forecast(autoarima_fit, h = forecast_horizon)

# Plot AutoARIMA forecast
autoplot(autoarima_forecast)
```




```r
# Define function for K-Fold Cross-Validation
kfold_cv <- function(model_fit, data, forecast_horizon, k = 10) {
  folds <- cut(1:nrow(data), breaks = k, labels = FALSE)
  rmse_errors <- rep(NA, k)
  mae_errors <- rep(NA, k)
  mape_errors <- rep(NA, k)
  for (i in 1:k) {
    training_set <- data[folds != i, ]
    testing_set <- data[folds == i, ]

    # Remove missing values from training and testing sets
    training_set <- na.omit(training_set)
    testing_set <- na.omit(testing_set)

    model <- model_fit(training_set)
    forecast <- forecast(model, h = forecast_horizon)
    forecast_values <- forecast$mean
    actual_values <- head(testing_set$value, length(forecast_values))
    rmse_errors[i] <- sqrt(mean((actual_values - forecast_values)^2))
    mae_errors[i] <- mean(abs(actual_values - forecast_values))
    mape_errors[i] <- mean(abs((actual_values - forecast_values) / actual_values) * 100)
  }
  data.frame(
    Fold = 1:k,
    RMSE = rmse_errors,
    MAE = mae_errors,
```

```
    MAPE = mape_errors
  )
}
```

```
# Apply K-Fold CV to AutoARIMA model
autoarima_cv <- kfold_cv(function(data) auto.arima(data$value), GDP_data, forecast_horizon)
```

```
# Apply K-Fold CV to ETS model
ets_cv <- kfold_cv(function(data) ets(ts(data$value, frequency = 4)), GDP_data, forecast_horizon)
```

```
# Print CV results for each model
print("AutoARIMA K-Fold CV Results:")
```

```
## [1] "AutoARIMA K-Fold CV Results:"
```

```
print(summary(autoarima_cv))
```

```
##       Fold            RMSE             MAE               MAPE
##  Min.   : 1.00   Min.   :  591.5   Min.   :  493.1   Min.   :    2.389
##  1st Qu.: 3.25   1st Qu.:20604.1   1st Qu.:20592.2   1st Qu.:  214.698
##  Median : 5.50   Median :27203.3   Median :27188.7   Median :  853.280
##  Mean   : 5.50   Mean   :23581.0   Mean   :23554.0   Mean   : 2864.941
##  3rd Qu.: 7.75   3rd Qu.:30290.5   3rd Qu.:30266.2   3rd Qu.: 4272.713
##  Max.   :10.00   Max.   :30798.9   Max.   :30773.2   Max.   :11597.362
```

```
print("ETS K-Fold CV Results:")
```

```
## [1] "ETS K-Fold CV Results:"
```

```
print(summary(ets_cv))
```

```
##       Fold            RMSE            MAE               MAPE
##  Min.   : 1.00   Min.   :  606   Min.   :  508.2   Min.   :    2.463
##  1st Qu.: 3.25   1st Qu.:20908   1st Qu.:20890.2   1st Qu.:  217.664
##  Median : 5.50   Median :27544   Median :27523.7   Median :  863.916
##  Mean   : 5.50   Mean   :23710   Mean   :23680.9   Mean   : 2866.285
##  3rd Qu.: 7.75   3rd Qu.:30250   3rd Qu.:30229.6   3rd Qu.: 4269.510
##  Max.   :10.00   Max.   :30937   Max.   :30908.8   Max.   :11647.489
```

```
# One-Step Ahead Point Forecasts
autoarima_forecast_1 <- forecast(autoarima_fit, h = 1)$mean
ets_forecast_1 <- forecast(ets_fit, h = 1)$mean
```

```
# Print forecasts
print("AutoARIMA One-Step Ahead Forecast:")
```

```
## [1] "AutoARIMA One-Step Ahead Forecast:"
```

```
print(autoarima_forecast_1)
```

```
## Time Series:
## Start = 315
## End = 315
## Frequency = 1
## [1] 29018.13
```

```
print("ETS One-Step Ahead Forecast:")
```

```
## [1] "ETS One-Step Ahead Forecast:"
```

```r
print(ets_forecast_1)
```

```
## Time Series:
## Start = 315
## End = 315
## Frequency = 1
## [1] 29040.24
```