457final.R

hyukjang

2025-03-25

```r
# Load required libraries
library(readr)      # For reading CSV files
library(lubridate)  # For date handling
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
library(forecast)   # For ARIMA modeling and forecasting
```

```
## Warning: package 'forecast' was built under R version 4.3.3

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

```r
library(ggplot2)    # For plotting
library(tseries)    # For stationarity tests
```

```
## Warning: package 'tseries' was built under R version 4.3.3
```

```r
library(tsoutliers) # For detecting and cleaning outliers

# 1. Data Import and Preprocessing

# Read the CSV file (ensure your working directory is set correctly)
data <- read_csv("Daily Prices_ICCO (1).csv")
```

```
## Rows: 7812 Columns: 2

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Date
## num (1): ICCO daily price (US$/tonne)
##
## 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.
```

```r
# Examine the structure and a preview of the data
str(data)
```

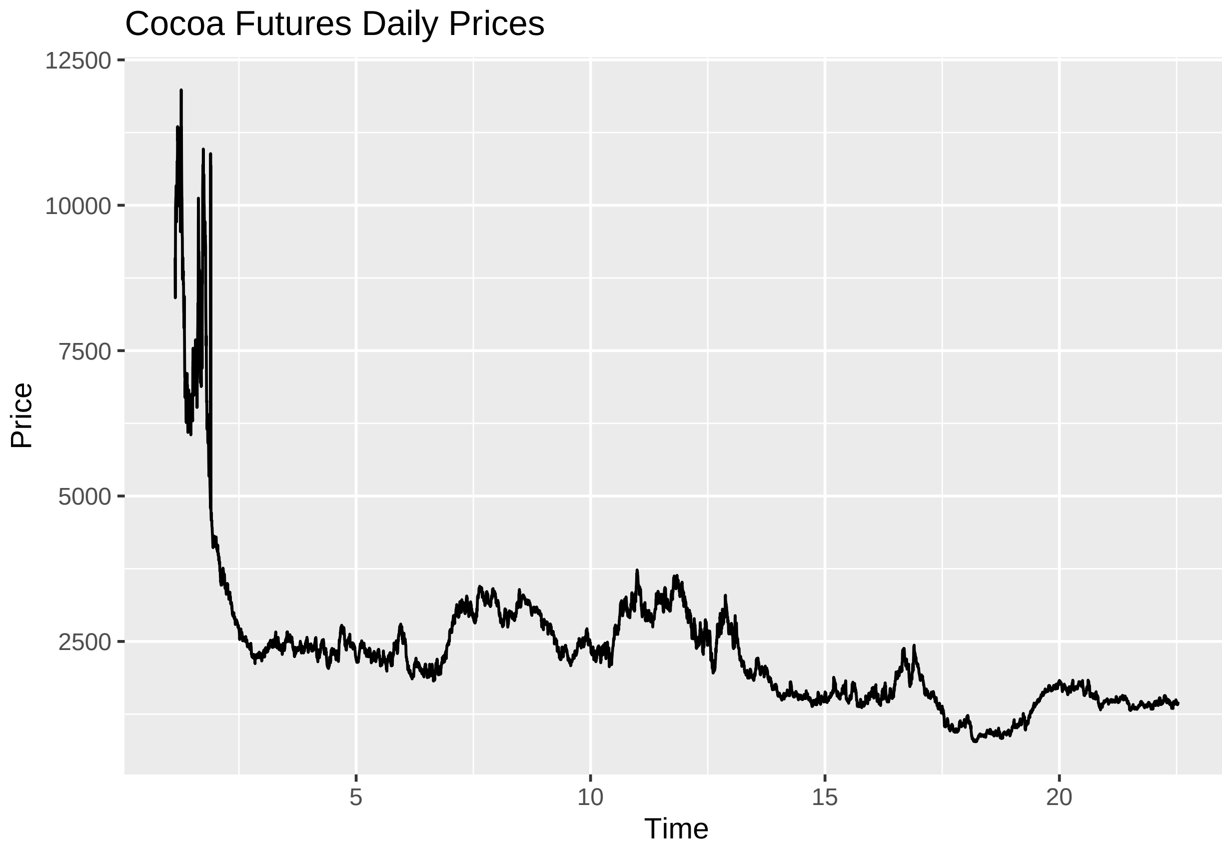
```
## spc_tbl_ [7,812 x 2] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Date                        : chr [1:7812] "27/02/2025" "26/02/2025" "25/02/2025" "24/02/2025" ..
##  $ ICCO daily price (US$/tonne): num [1:7812] 9100 9090 8669 8409 9106 ...
```

```
##  - attr(*, "spec")=
##   .. cols(
##   ..   Date = col_character(),
##   ..   `ICCO daily price (US$/tonne)` = col_number()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

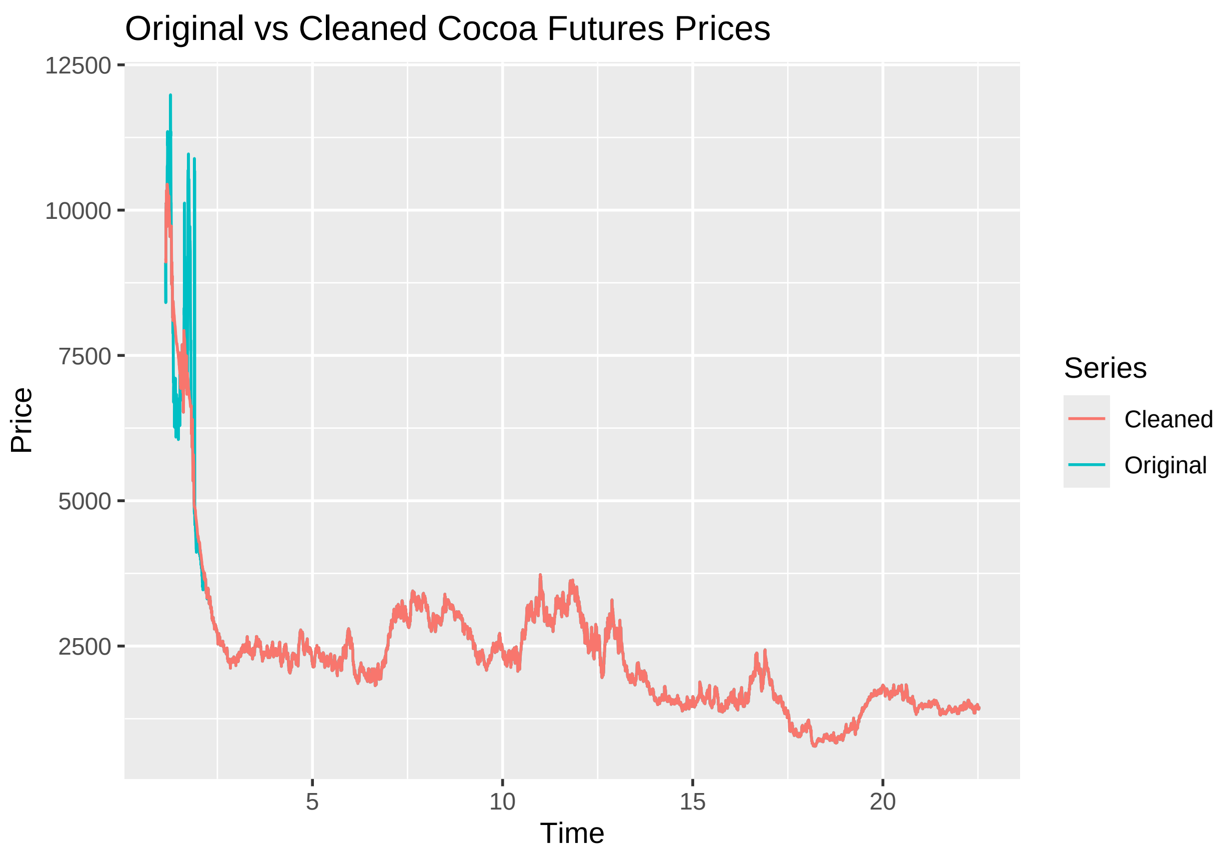
```r
head(data)
```

```
## # A tibble: 6 x 2
##   Date       `ICCO daily price (US$/tonne)`
##   <chr>                               <dbl>
## 1 27/02/2025                          9100.
## 2 26/02/2025                          9090.
## 3 25/02/2025                          8669.
## 4 24/02/2025                          8409.
## 5 21/02/2025                          9106.
## 6 20/02/2025                          9962.
```

```r
# Create a time series object.
# For daily data, we use frequency = 365 (adjust if weekends/holidays are removed)
start_year <- year(min(data$Date))
start_doy  <- yday(min(data$Date))
price_ts <- ts(data$`ICCO daily price (US$/tonne)`, frequency = 365, start = c(start_year, start_doy))

# Plot the raw time series
autoplot(price_ts) +
  ggtitle("Cocoa Futures Daily Prices") +
  xlab("Time") +
  ylab("Price")
```




```r
# 2. Data Cleaning and Transformation

# Remove potential outliers using tsoutliers package's tsclean function.
# This function detects and replaces outliers in the series.
price_ts_clean <- tsclean(price_ts)

# Plot the cleaned series vs the original
autoplot(price_ts, series = "Original") +
  autolayer(price_ts_clean, series = "Cleaned") +
  ggtitle("Original vs Cleaned Cocoa Futures Prices") +
  xlab("Time") +
  ylab("Price") +
  guides(colour = guide_legend(title = "Series"))
```




```
# Check for variance stabilization need using a Box-Cox transformation
lambda <- BoxCox.lambda(price_ts_clean)
cat("Estimated Box-Cox Lambda:", lambda, "\n")
```

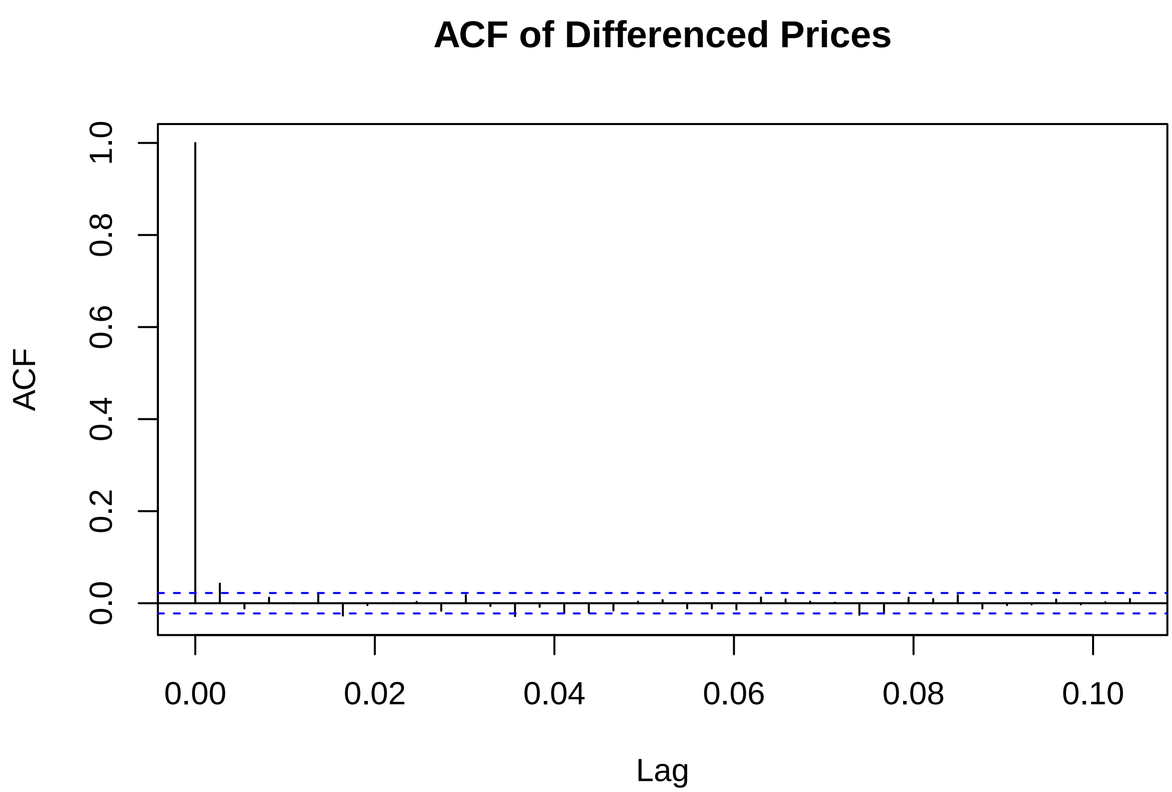
```
## Estimated Box-Cox Lambda: -0.504177
```

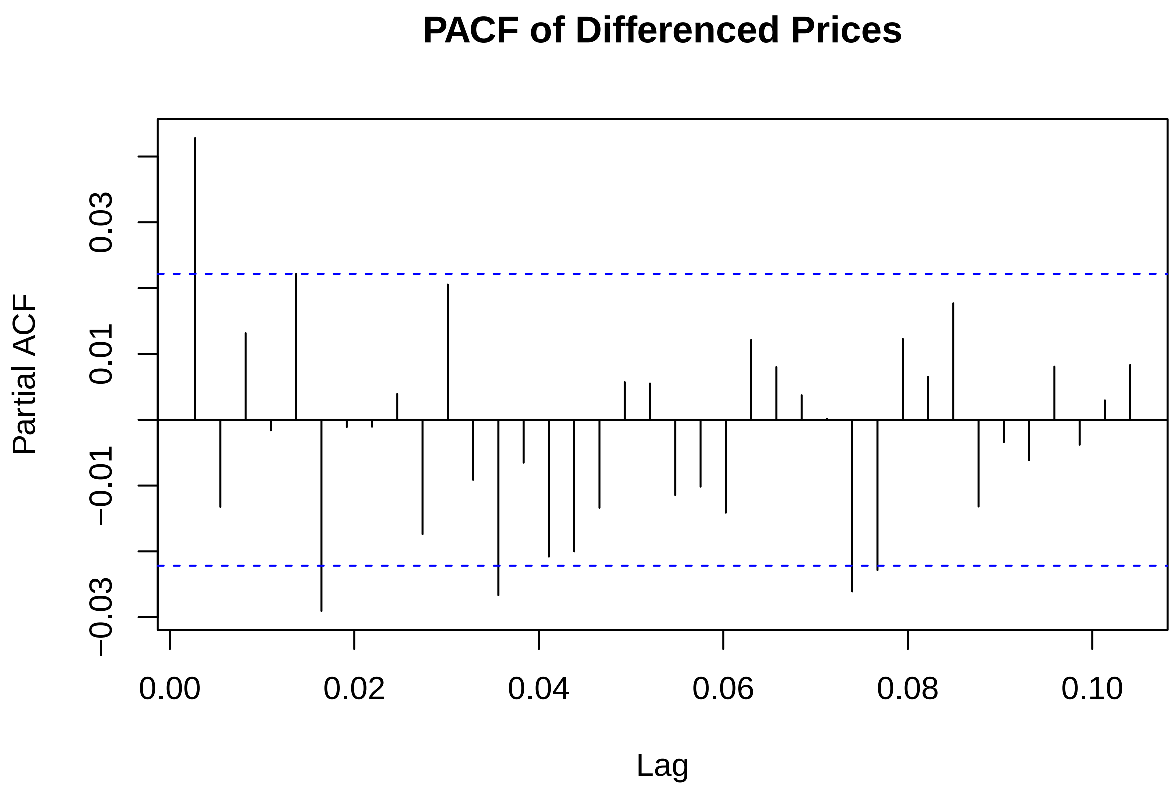
```
# Apply Box-Cox transformation if lambda is not 1 (i.e., non-linear variance)
if(abs(lambda - 1) > 0.1){
  price_ts_trans <- BoxCox(price_ts_clean, lambda)
} else {
  price_ts_trans <- price_ts_clean
}

# 3. Stationarity Check

# ADF Test on the transformed series
adf_result <- adf.test(price_ts_trans)
print(adf_result)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  price_ts_trans
## Dickey-Fuller = -2.885, Lag order = 19, p-value = 0.2035
## alternative hypothesis: stationary
```

```
# If non-stationary (p-value > 0.05), take first differences
if(adf_result$p.value > 0.05){
```

```
  price_diff <- diff(price_ts_trans)
  autoplot(price_diff) +
    ggtitle("First Difference of Transformed Prices") +
    xlab("Time") +
    ylab("Differenced Price")

  # Plot ACF and PACF for the differenced series
  acf(price_diff, main="ACF of Differenced Prices")
  pacf(price_diff, main="PACF of Differenced Prices")
} else {
  price_diff <- price_ts_trans
}
```






```r
# 4. ARIMA Model Selection and Fitting

# Option A: Manual candidate models based on ACF/PACF interpretation
# (e.g., ARIMA(1,1,0), ARIMA(0,1,1), ARIMA(1,1,1) if differencing was applied)
# Adjust orders if you believe differencing is needed.
candidate_110 <- arima(price_ts_trans, order = c(1, 1, 0))
candidate_011 <- arima(price_ts_trans, order = c(0, 1, 1))
candidate_111 <- arima(price_ts_trans, order = c(1, 1, 1))

# Option B: Let auto.arima select the best model.
# Note: Set lambda= if you applied a Box-Cox transformation above.
auto_fit <- auto.arima(price_ts_clean,
                       lambda = if(abs(lambda - 1) > 0.1) lambda else NULL,
                       biasadj = TRUE,
                       stepwise = FALSE,
                       approximation = FALSE)
summary(auto_fit)
```

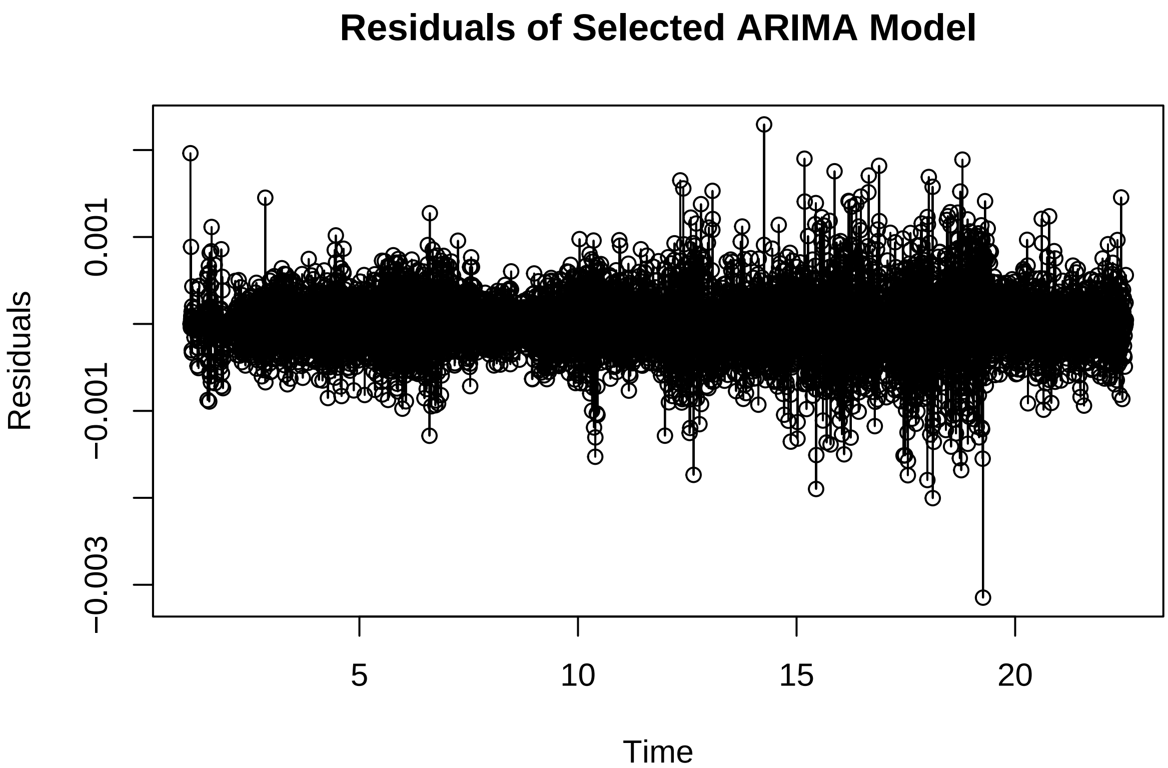
```
## Series: price_ts_clean
## ARIMA(1,1,1)
## Box Cox transformation: lambda= -0.504177
##
## Coefficients:
##           ar1     ma1
##       -0.7203  0.7560
## s.e.   0.1208  0.1147
```

```
##
## sigma^2 = 1.183e-07:  log likelihood = 51225.31
## AIC=-102444.6   AICc=-102444.6   BIC=-102423.7
##
## Training set error measures:
##                     ME     RMSE      MAE         MPE     MAPE       MASE
## Training set -1.457441 47.87373 26.21745 -0.05530376 1.127916 0.04389401
##                     ACF1
## Training set -0.07526354
```

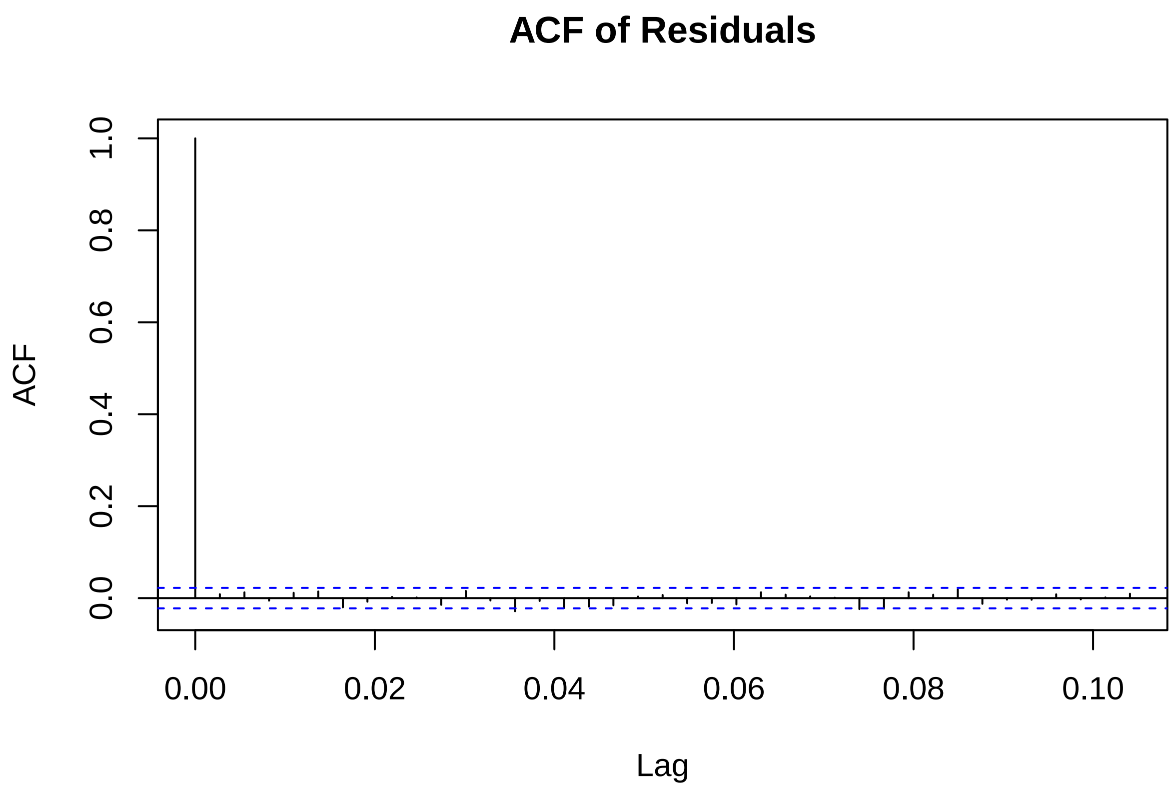
```r
# Compare AIC values among candidate models
model_comparison <- data.frame(
  Model = c("ARIMA(1,1,0)", "ARIMA(0,1,1)", "ARIMA(1,1,1)", "Auto ARIMA"),
  AIC = c(AIC(candidate_110), AIC(candidate_011), AIC(candidate_111), AIC(auto_fit))
)
print(model_comparison)
```

```
##          Model       AIC
## 1 ARIMA(1,1,0) -102440.7
## 2 ARIMA(0,1,1) -102441.0
## 3 ARIMA(1,1,1) -102444.6
## 4   Auto ARIMA -102444.6
```

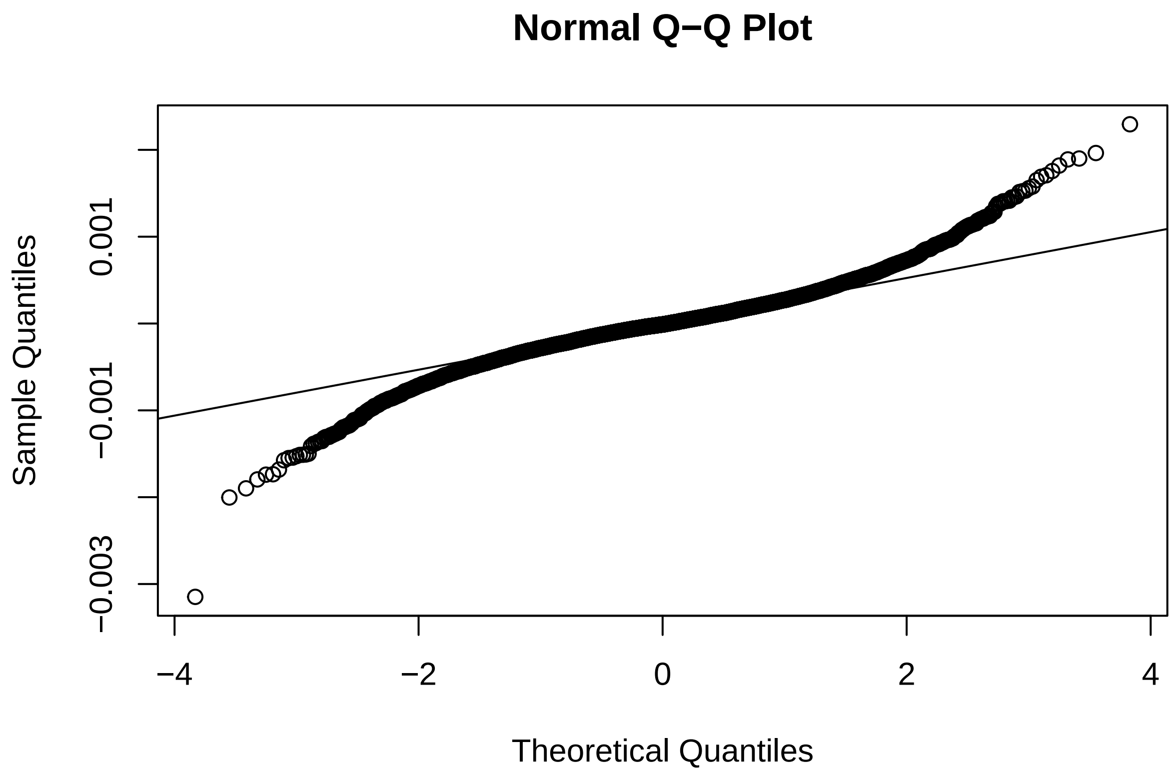
```r
# Choose the best model based on AIC. Here we assume auto_fit is the best candidate.
best_model <- auto_fit

# 5. Diagnostic Checks on the Selected Model

# Residual plots over time
plot(residuals(best_model), type = "o",
     main = "Residuals of Selected ARIMA Model",
     xlab = "Time", ylab = "Residuals")
```




```r
# ACF of residuals
acf(residuals(best_model), main="ACF of Residuals")
```



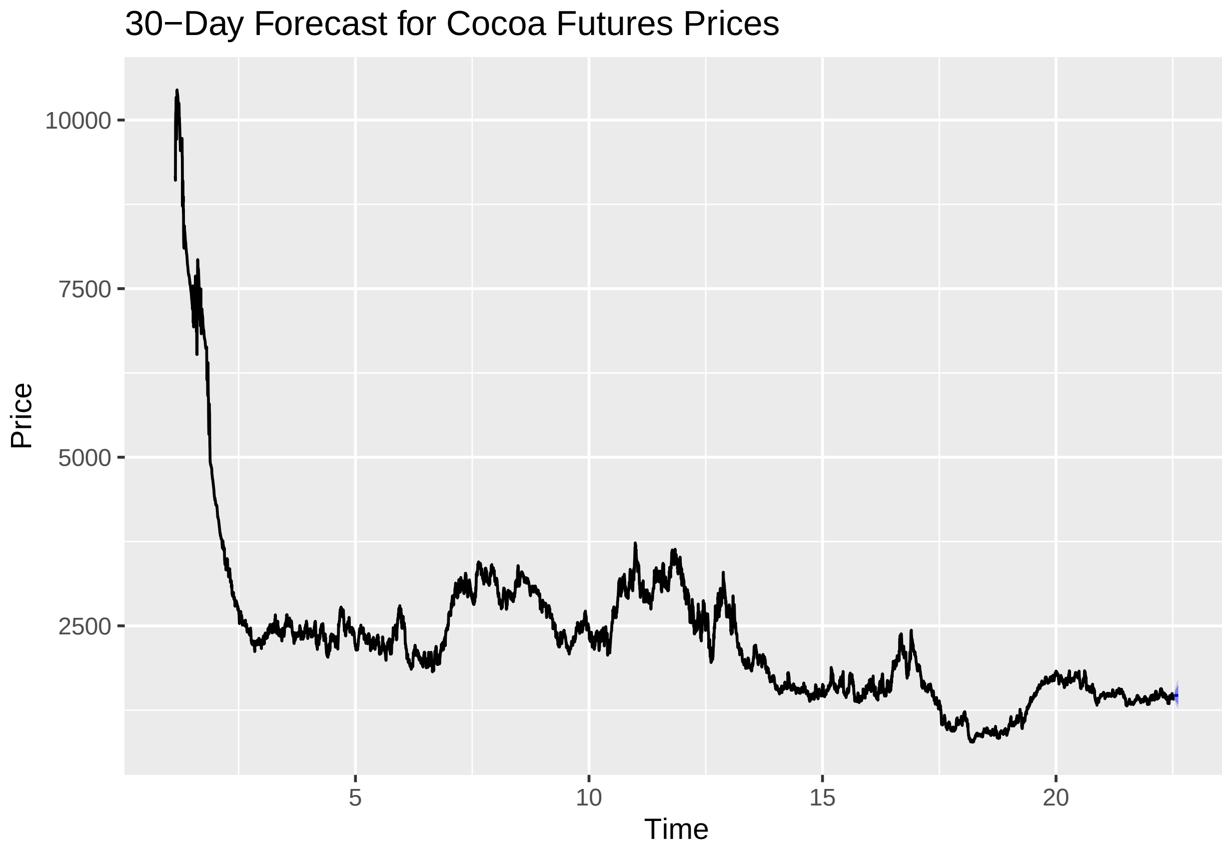

```r
# Q-Q plot for normality
qqnorm(residuals(best_model))
qqline(residuals(best_model))
```




```r
# Ljung-Box test for residual autocorrelation
lb_test <- Box.test(residuals(best_model), lag = 10, type = "Ljung-Box")
print(lb_test)
```

```
##
##  Box-Ljung test
##
## data:  residuals(best_model)
## X-squared = 10.067, df = 10, p-value = 0.4347
```

```r
# 6. Forecasting

forecast_horizon <- 30  # Forecast horizon (e.g., 30 days)
fc <- forecast(best_model, h = forecast_horizon)

# Plot the forecast
autoplot(fc) +
  ggtitle("30-Day Forecast for Cocoa Futures Prices") +
  xlab("Time") +
  ylab("Price")
```




```
# If a Box-Cox transformation was applied, the forecast function automatically back-transforms the pred
```