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
# 1. Data Import and Preprocessing

# Read the CSV file (ensure the working directory is set correctly or provide full path)
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 structure (adjust column names if necessary)
str(data)
```

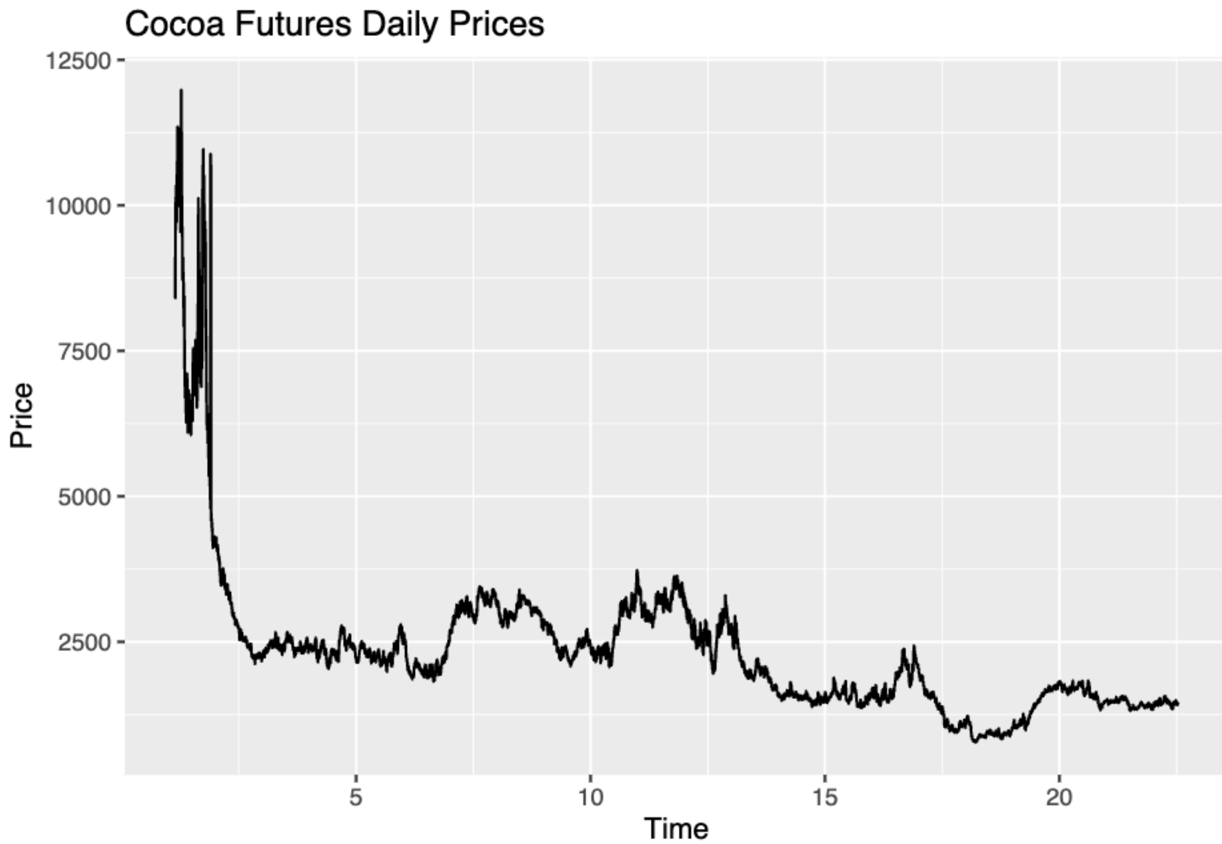
```
## spc_tbl_ [7,812 x 2] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Date                        : chr [1:7812] "27/02/2025" "26/02/2025" "25/02/2025" "24/02/2025" ..
##  $ ICCO daily price (US$/tonne): num [1:7812] 9100 9090 8669 8409 9106 ...
##  - attr(*, "spec")=
##   .. cols(
```

```
##   ..   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, set frequency = 365 (or use an appropriate frequency 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")
```



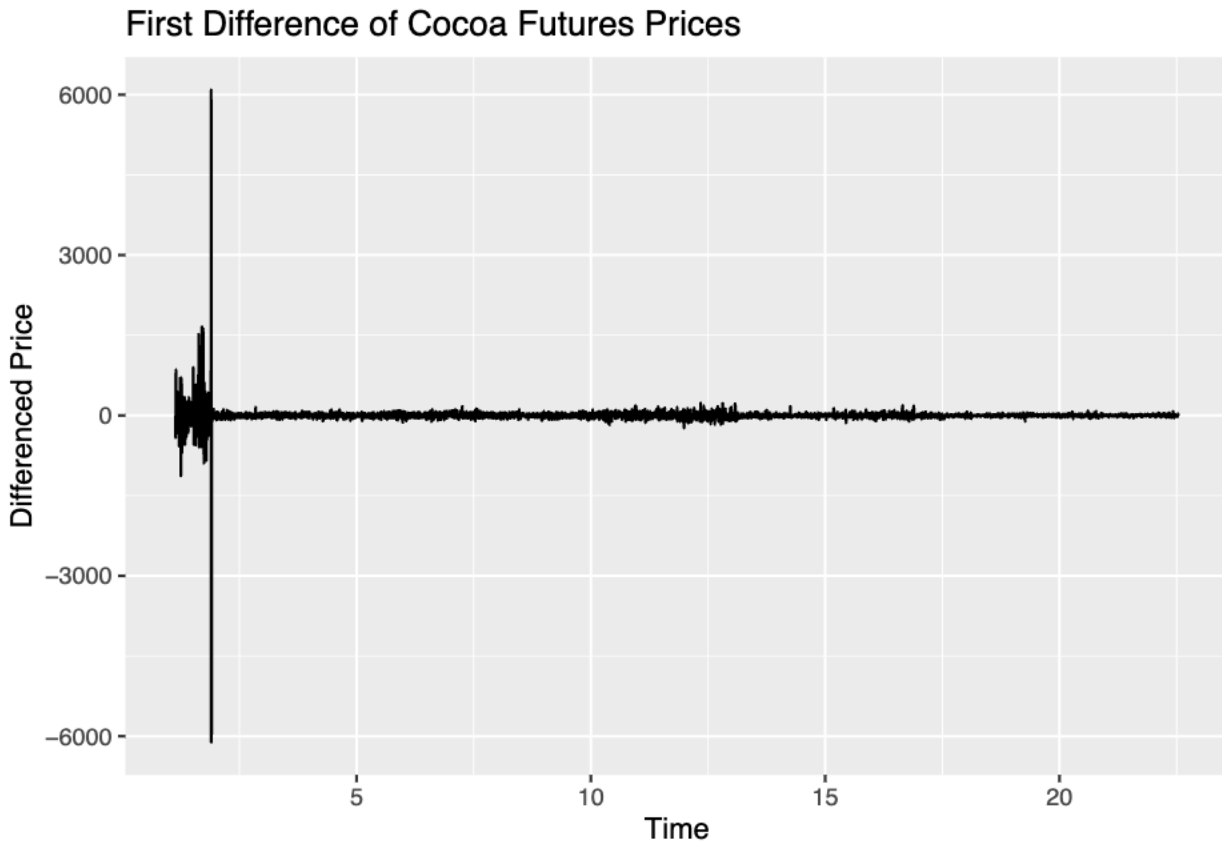

```r
# 2. Stationarity Check & Differencing

# Perform the Augmented Dickey-Fuller Test to check for stationarity
adf_result <- adf.test(price_ts)
```

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

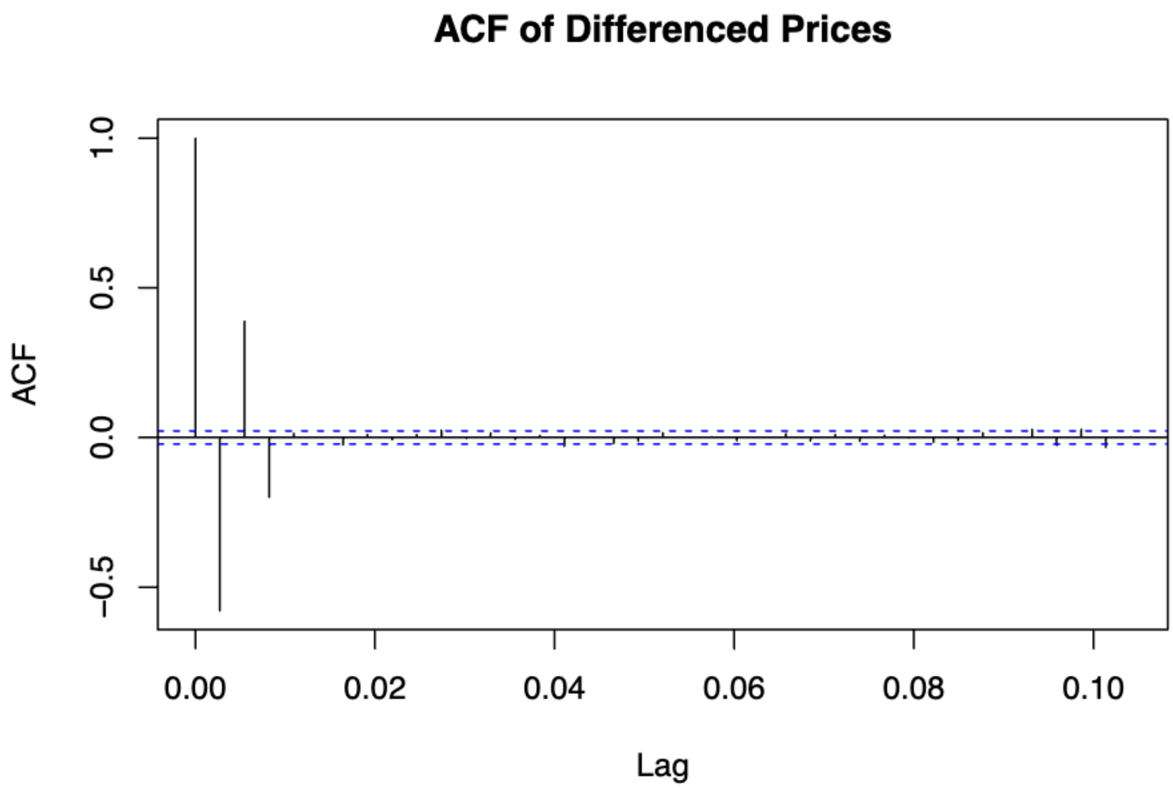
```r
print(adf_result)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  price_ts
## Dickey-Fuller = -6.6276, Lag order = 19, p-value = 0.01
## alternative hypothesis: stationary
```

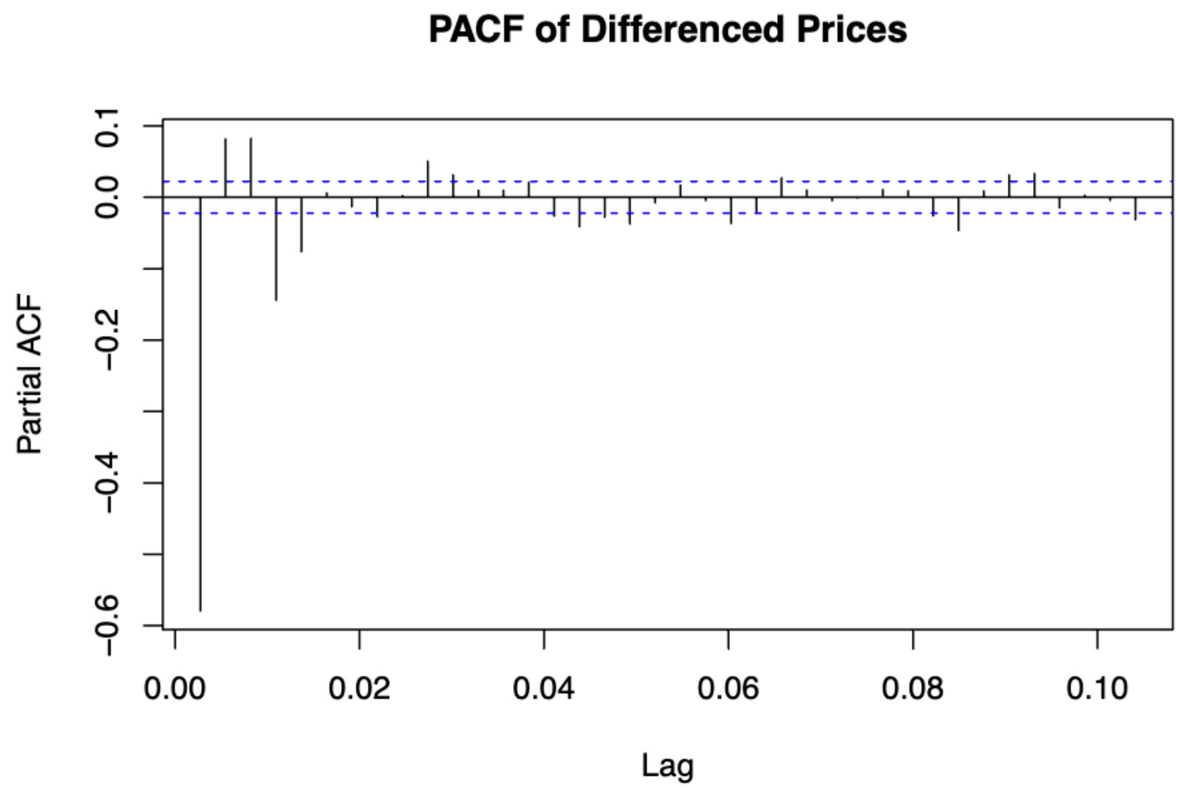
```r
# If the p-value is > 0.05, the series is likely non-stationary so we difference it
price_diff <- diff(price_ts)
autoplot(price_diff) +
  ggtitle("First Difference of Cocoa Futures Prices") +
  xlab("Time") +
  ylab("Differenced Price")
```




```r
# Plot ACF and PACF for the differenced series (use these plots to help decide on the ARIMA orders)
acf(price_diff, main="ACF of Differenced Prices")
```




```r
pacf(price_diff, main="PACF of Differenced Prices")
```




```r
# 3. Candidate ARIMA Models

# Based on your ACF/PACF interpretation (e.g., a spike at lag 1 in both ACF and PACF),
# try fitting ARIMA(1,1,0), ARIMA(0,1,1), and ARIMA(1,1,1).
arima110 <- arima(price_ts, order = c(1, 1, 0))
arima011 <- arima(price_ts, order = c(0, 1, 1))
arima111 <- arima(price_ts, order = c(1, 1, 1))

# Summaries of the models to inspect parameter estimates and diagnostic metrics
summary(arima110)
```

```
##
## Call:
## arima(x = price_ts, order = c(1, 1, 0))
##
## Coefficients:
##           ar1
##       -0.5790
## s.e.   0.0092
##
## sigma^2 estimated as 15857:  log likelihood = -48854.98,  aic = 97713.97
##
## Training set error measures:
##                     ME     RMSE      MAE         MPE     MAPE     MASE
## Training set -1.541816 125.9154 39.69301 -0.07304487 1.456403 1.146831
##                     ACF1
```

```
## Training set 0.04704558
```

```r
summary(arima011)
```

```
##
## Call:
## arima(x = price_ts, order = c(0, 1, 1))
##
## Coefficients:
##           ma1
##       -0.4575
## s.e.   0.0087
##
## sigma^2 estimated as 17977:  log likelihood = -49344.95,  aic = 98693.89
##
## Training set error measures:
##                     ME     RMSE      MAE         MPE     MAPE     MASE
## Training set -1.799099 134.0682 39.65113 -0.08647213 1.433988 1.145621
##                    ACF1
## Training set -0.1287615
```

```r
summary(arima111)
```

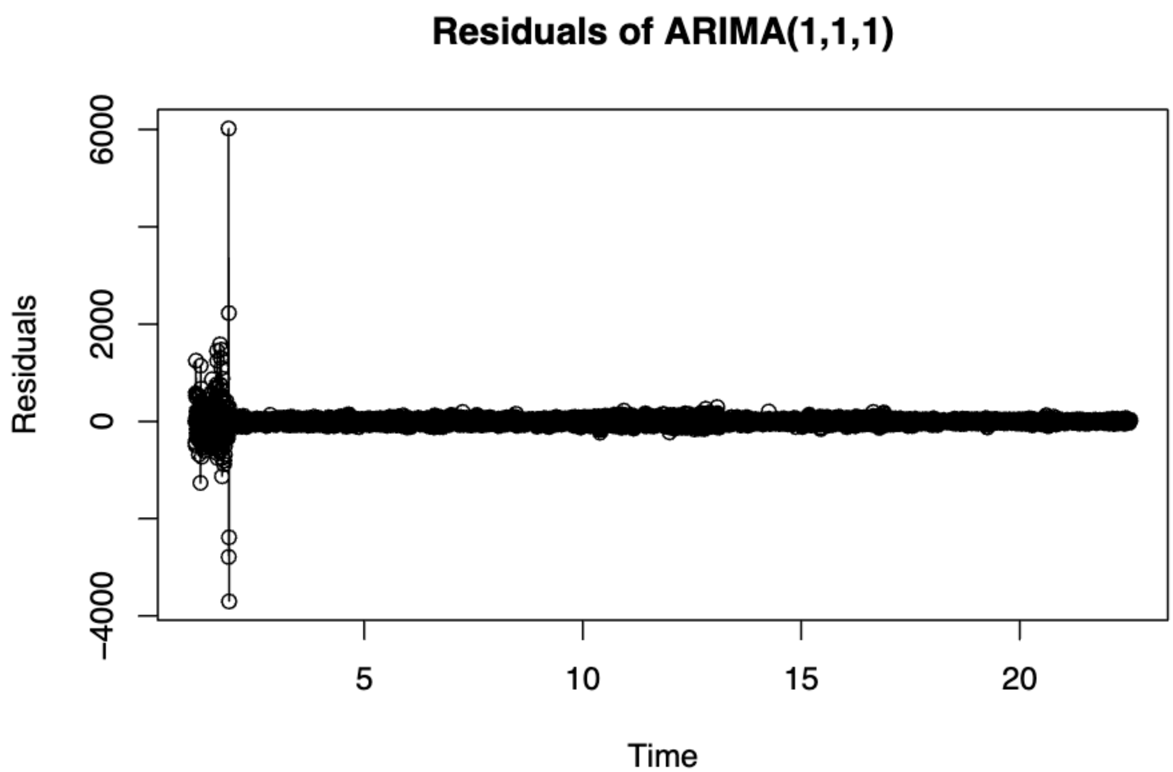
```
##
## Call:
## arima(x = price_ts, order = c(1, 1, 1))
##
## Coefficients:
##           ar1     ma1
##       -0.6476  0.1026
## s.e.   0.0132  0.0161
##
## sigma^2 estimated as 15778:  log likelihood = -48835.59,  aic = 97677.19
##
## Training set error measures:
##                     ME     RMSE      MAE         MPE     MAPE     MASE
## Training set -1.458705 125.6032 39.13152 -0.06952958 1.434941 1.130609
##                     ACF1
## Training set 0.007406219
```

```r
# 4. Model Comparison Using AIC

# Create a table with AIC values for comparison
model_comparison <- data.frame(
  Model = c("ARIMA(1,1,0)", "ARIMA(0,1,1)", "ARIMA(1,1,1)"),
  AIC = c(AIC(arima110), AIC(arima011), AIC(arima111))
)
print(model_comparison)
```

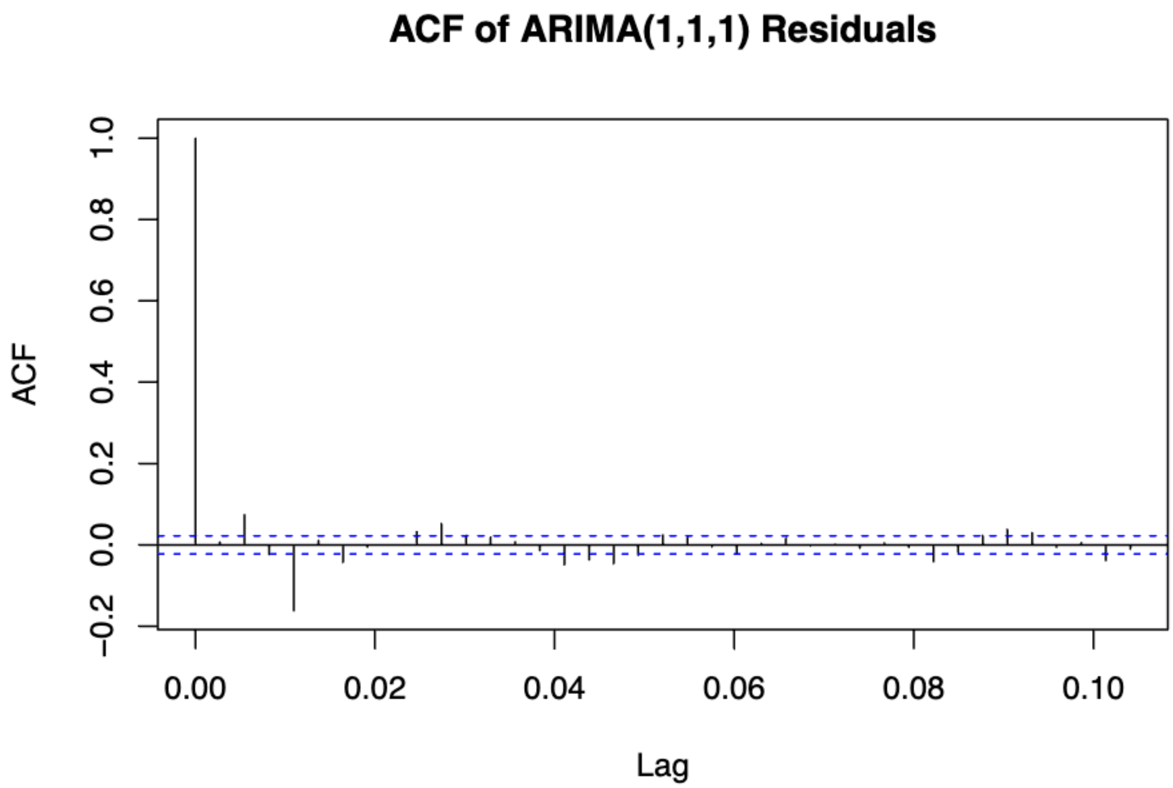
```
##          Model      AIC
## 1 ARIMA(1,1,0) 97713.97
## 2 ARIMA(0,1,1) 98693.89
## 3 ARIMA(1,1,1) 97677.19
```

```r
# 5. Residual Diagnostics for the Chosen Model
# Suppose the AIC values suggest that ARIMA(1,1,1) is best; then check its residuals:
```

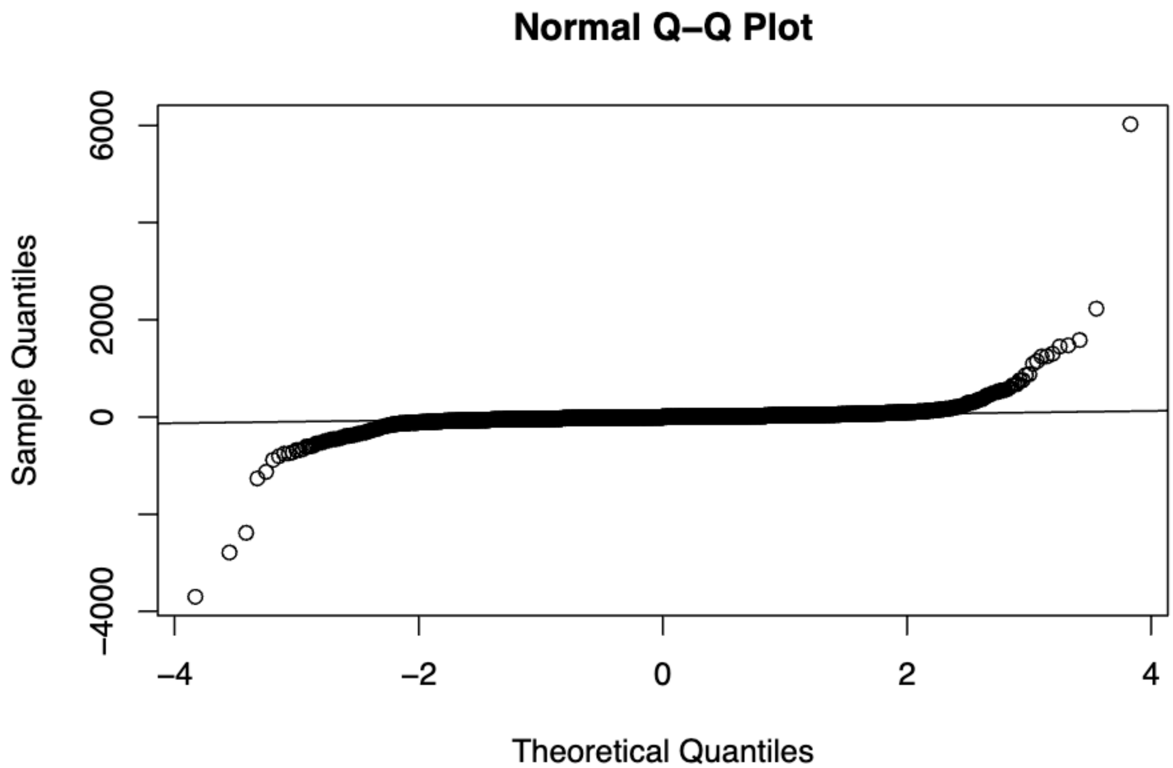
```r
# Residual plot over time
plot(residuals(arima111), type = "o",
     main = "Residuals of ARIMA(1,1,1)",
     xlab = "Time", ylab = "Residuals")
```




```r
# ACF of residuals to check for autocorrelation
acf(residuals(arima111), main="ACF of ARIMA(1,1,1) Residuals")
```



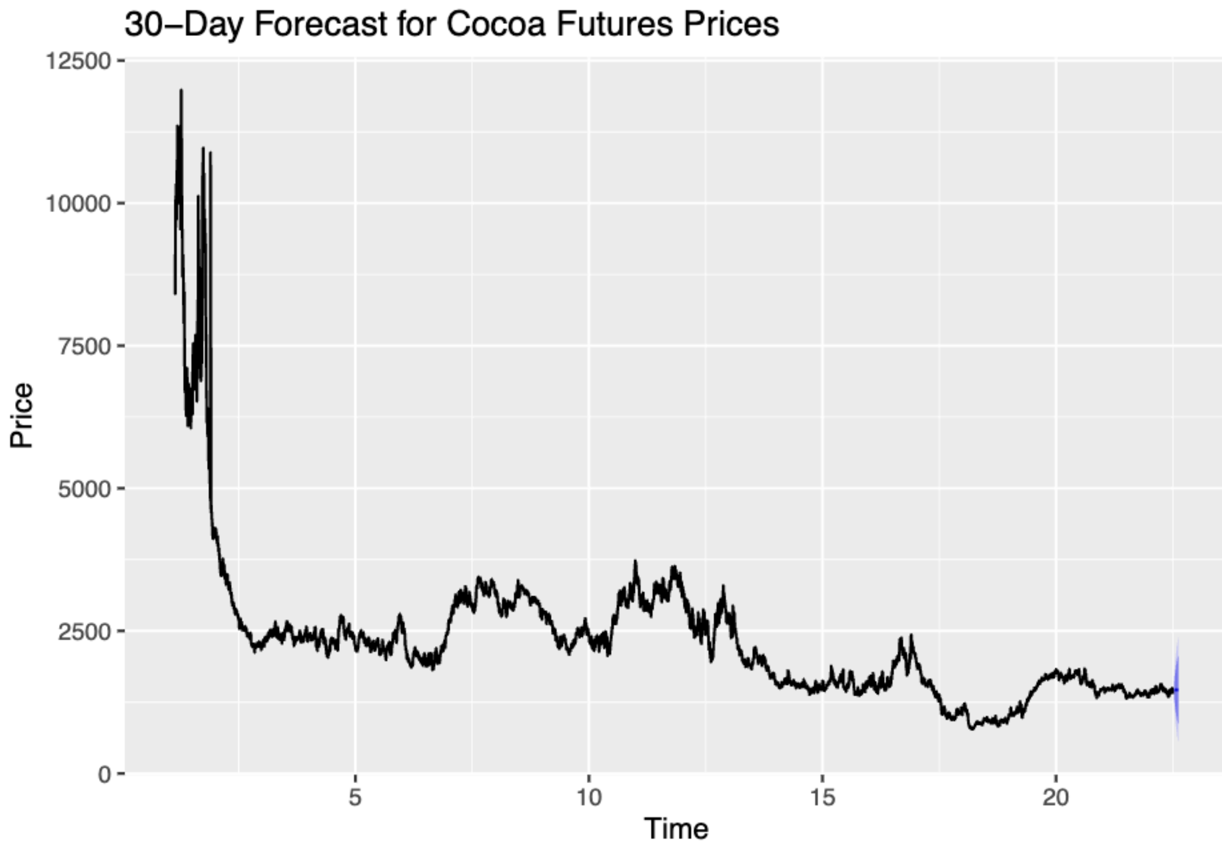

```
# Q-Q plot for normality of residuals
qqnorm(residuals(arima111))
qqline(residuals(arima111))
```




```r
# Ljung-Box test to statistically test for autocorrelation in residuals
lb_test <- Box.test(residuals(arima111), lag = 10, type = "Ljung-Box")
print(lb_test)
```

```
##
##  Box-Ljung test
##
## data:  residuals(arima111)
## X-squared = 300.18, df = 10, p-value < 2.2e-16
```

```r
# 6. Forecasting
# Forecast the next 30 days (or adjust as needed) using the selected model
forecast_horizon <- 30
fc <- forecast(arima111, h = forecast_horizon)
autoplot(fc) +
  ggtitle("30-Day Forecast for Cocoa Futures Prices") +
  xlab("Time") +
  ylab("Price")
```




```
# End of script
```