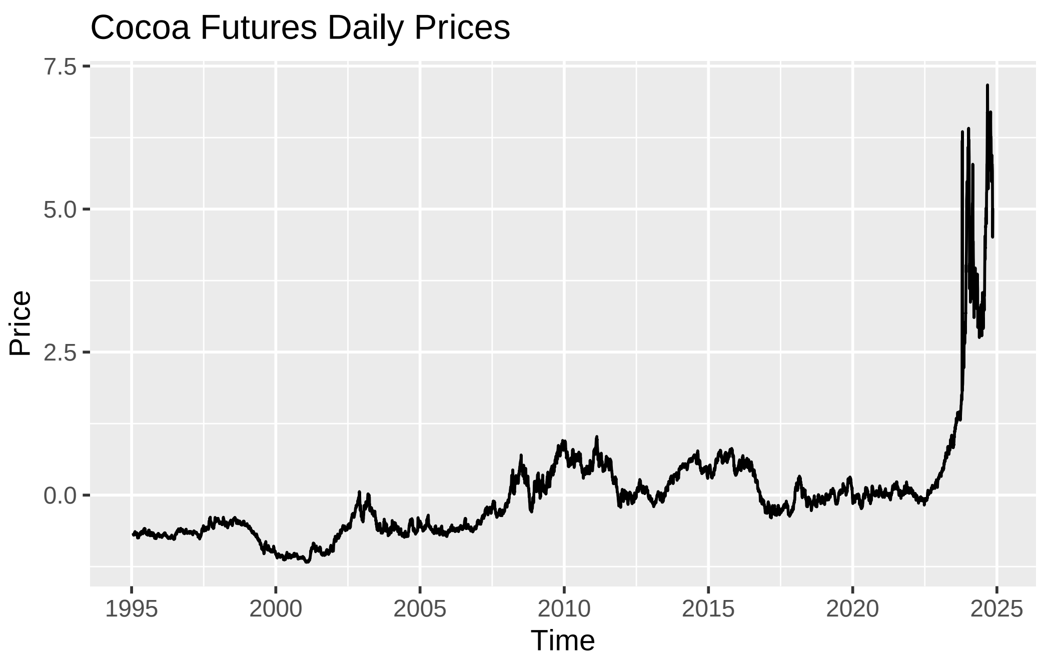
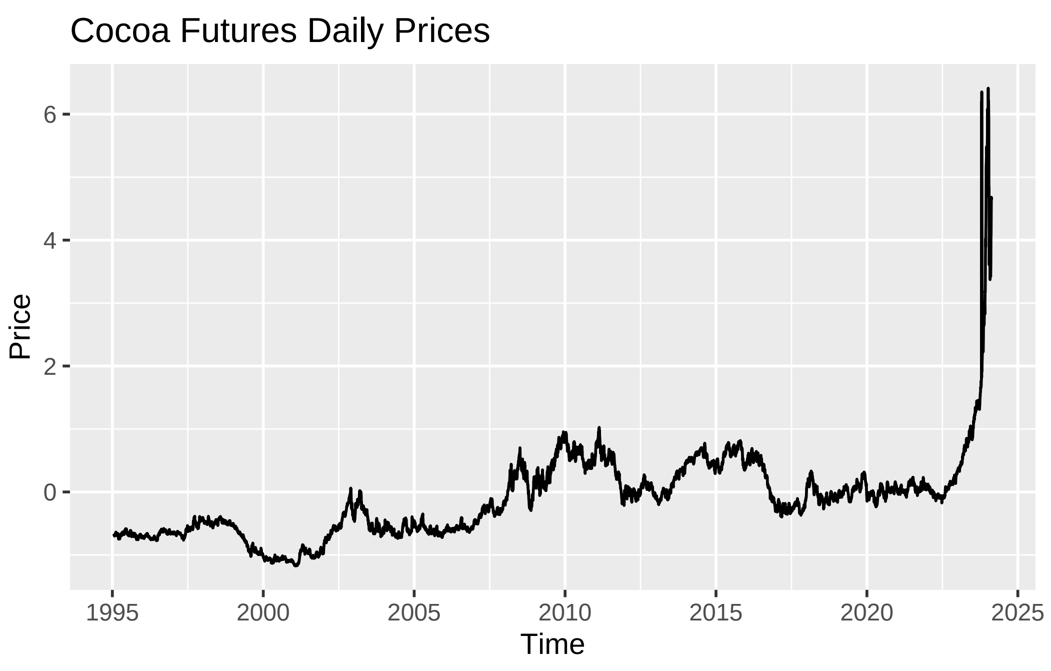
457finaltwodifffrequency

```r
# Load libraries
library(readr)
library(lubridate)
library(forecast)
library(ggplot2)
library(tseries)
library(tsoutliers)
library(vars)
```

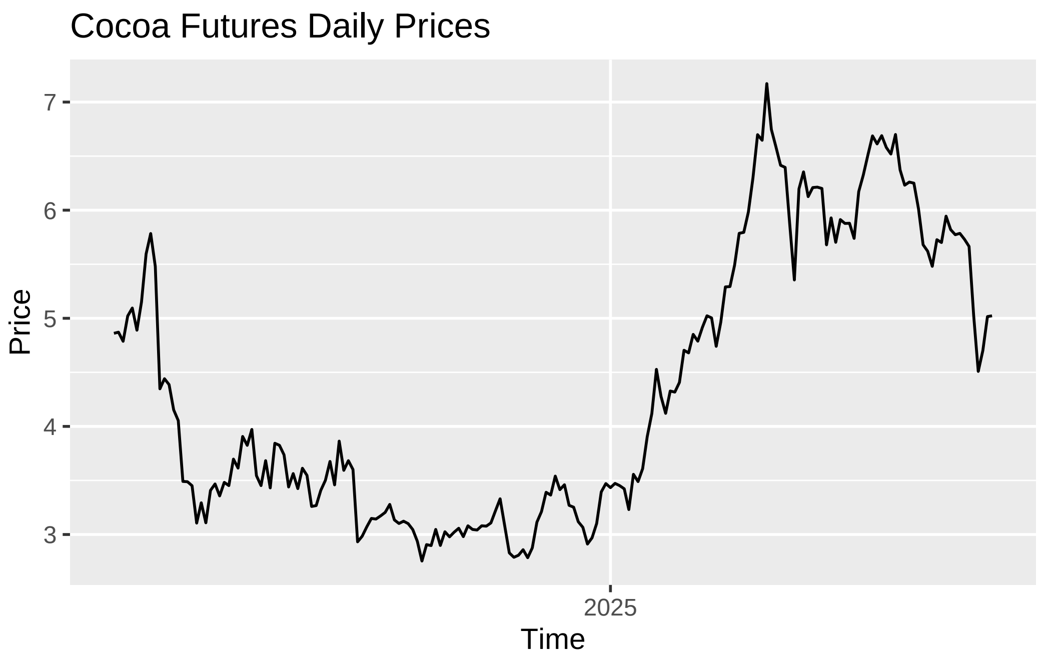
```r
# 1. Data Import and Preprocessing

# Read the CSV file (ensure your working directory is set correctly)
data <- read_csv("Daily Prices_ICCO (1).csv", show_col_types = FALSE)
data$Date <- as.Date(data$Date, format = "%d/%m/%Y")

# Make sure the data is in ascending order by Date
data <- data[order(data$Date), ]

# Scale the price and add it as a new column
data$ScaledPrice <- scale(data$`ICCO daily price (US$/tonne)`, center = TRUE, scale = TRUE)

# Calculate start_year and start_doy before creating the time series object
start_year <- year(min(data$Date))
start_doy  <- yday(min(data$Date))

# Create a time series object for daily data
price_ts <- ts(data$ScaledPrice, frequency = 262, start = c(start_year, start_doy))

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



```
# 2. Split the Data by Date (Training: Before 2020-01-01, Test: From 2020-01-01 Onward)

# Create training and test subsets from the original data frame
train_data <- data[data$Date < as.Date("2024-06-01"), ]
test_data  <- data[data$Date >= as.Date("2024-06-01"), ]

# Convert the training subset into a time series
train_start_year <- year(min(train_data$Date))
train_start_doy  <- yday(min(train_data$Date))
train_ts <- ts(train_data$ScaledPrice, frequency = 262, start = c(train_start_year, train_sta
autoplot(train_ts)+
  ggtitle("Cocoa Futures Daily Prices") +
  xlab("Time") +
  ylab("Price")
```



```
# Convert the test subset into a time series
test_start_year <- year(min(test_data$Date))
test_start_doy  <- yday(min(test_data$Date))
test_ts <- ts(test_data$ScaledPrice, frequency = 262, start = c(test_start_year, test_start_

autoplot(test_ts)+
  ggtitle("Cocoa Futures Daily Prices") +
  xlab("Time") +
  ylab("Price")
```



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

```
Estimated Box-Cox Lambda: 0.4932815
```

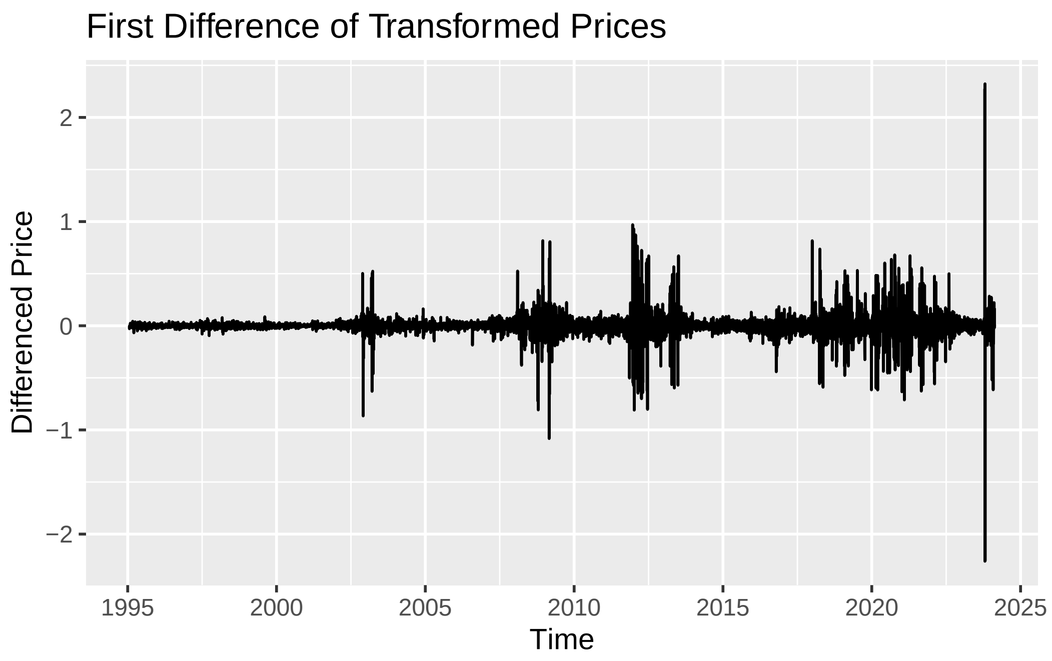
```r
# Apply Box-Cox transformation if lambda is not 1 (i.e., non-linear variance)
if(abs(lambda - 1) > 0.1){
  train_ts <- BoxCox(train_ts, lambda)
} else {
  train_ts <- train_ts
}
```

```r
# 3. Stationarity Check

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

```

    Augmented Dickey-Fuller Test
```

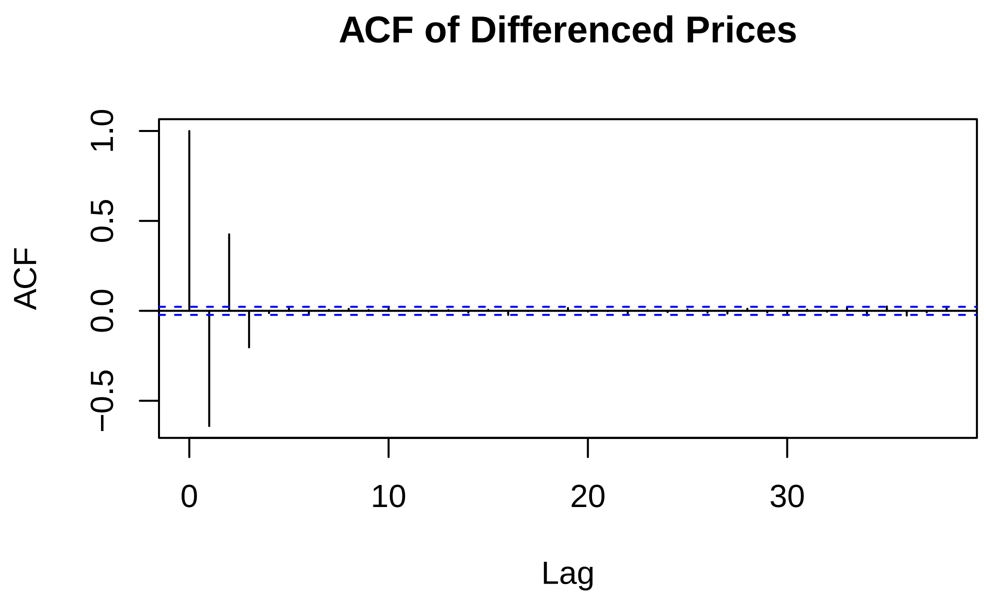
```
data:  train_ts
Dickey-Fuller = -3.1274, Lag order = 19, p-value = 0.1008
alternative hypothesis: stationary
```

```
# Compute the differenced series if the ADF test suggests non-stationarity
if(adf_result$p.value > 0.05){
  train_diff <- diff(train_ts)
} else {
  train_diff <- train_ts
}

autoplot(train_diff) +
  ggtitle("First Difference of Transformed Prices") +
  xlab("Time") +
  ylab("Differenced Price")
```

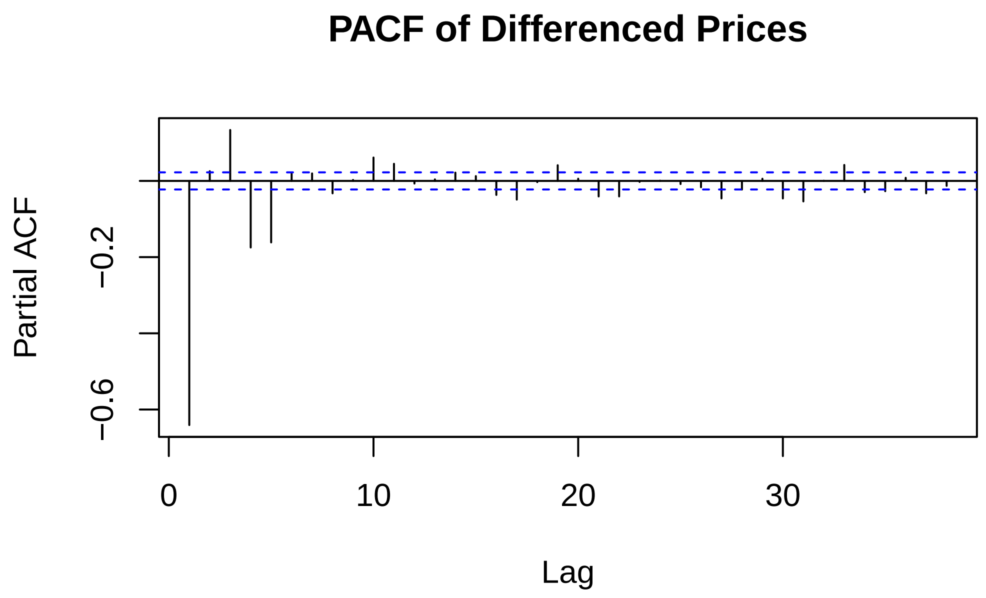


```
train_ts_acf <- ts(train_data$ScaledPrice, frequency = 1, start = c(train_start_year, train_s
if(adf_result$p.value > 0.05){
  train_diff_acf <- diff(train_ts_acf)
} else {
  train_diff_acf <- train_ts_acf
}
```

```r
acf(train_diff_acf, main = "ACF of Differenced Prices")
```



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

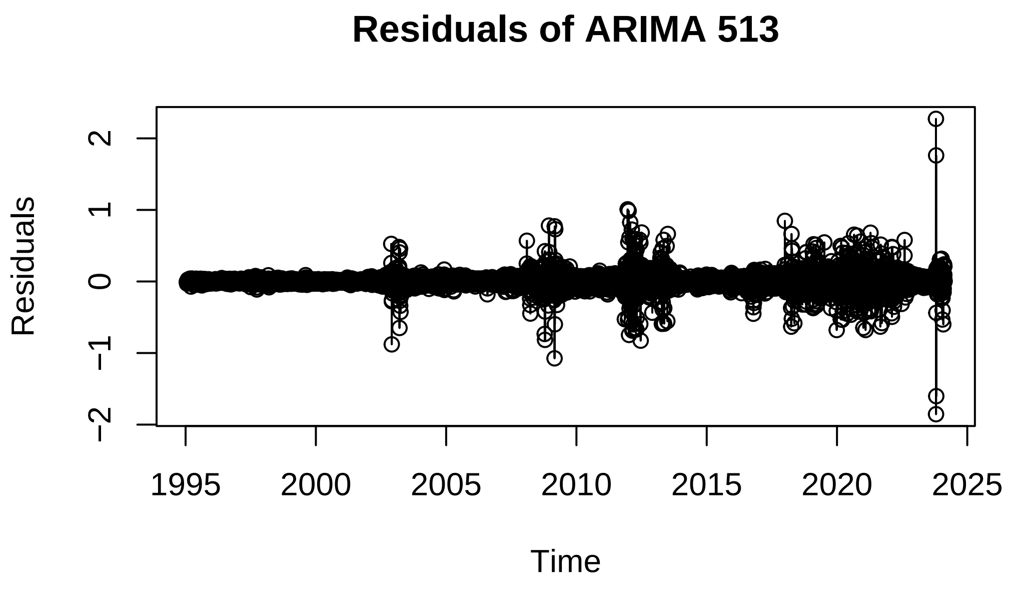


```r
# Manual candidate models based on ACF/PACF interpretation

candidate_513 <- arima(train_ts, order = c(5, 1, 3))
candidate_514 <- arima(train_ts, order = c(5, 1, 4))
candidate_415 <- arima(train_ts, order = c(4, 1, 5))
candidate_413 <- arima(train_ts, order = c(4, 1, 3))
candidate_414 <- arima(train_ts, order = c(4, 1, 4))
candidate_315 <- arima(train_ts, order = c(3, 1, 5))
candidate_314 <- arima(train_ts, order = c(3, 1, 4))
candidate_311 <- arima(train_ts, order = c(3, 1, 1))
candidate_312 <- arima(train_ts, order = c(3, 1, 2))
candidate_313 <- arima(train_ts, order = c(3, 1, 3))
model_comparison <- data.frame(
  Model = c("ARIMA(513)", "ARIMA(514)", "ARIMA(415)", "ARIMA(413)", "ARIMA(414)", "ARIMA(315
  AIC = c(AIC(candidate_513), AIC(candidate_514), AIC(candidate_415), AIC(candidate_413), AI
)
print(model_comparison)
```

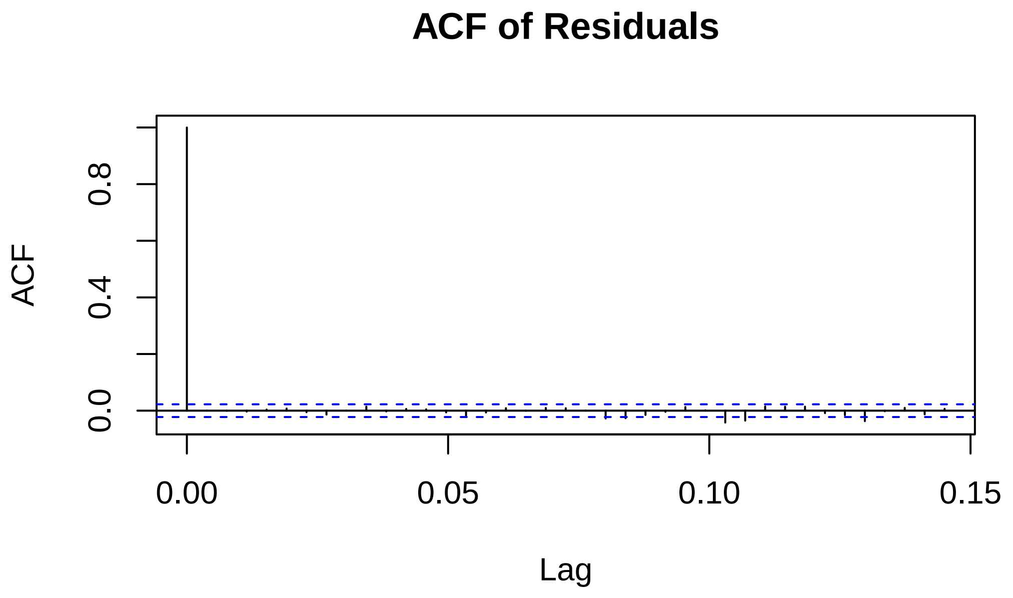
```
        Model       AIC
1  ARIMA(513) -12526.13
2  ARIMA(514) -12524.63
3  ARIMA(415) -12517.01
4  ARIMA(413) -12518.28
5  ARIMA(414) -12516.78
6  ARIMA(315) -12522.52
7  ARIMA(314) -12518.23
8  ARIMA(311) -12493.57
9  ARIMA(312) -12501.67
10 ARIMA(313) -12520.09
```

```r
best_model <- candidate_513
```

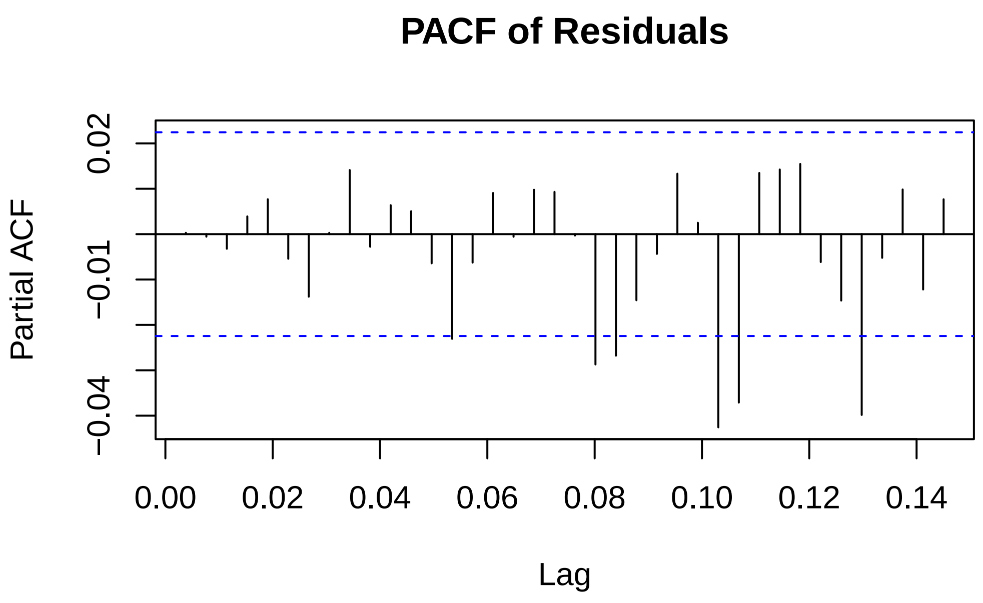
```r
plot(residuals(candidate_513), type = "o",
     main = "Residuals of ARIMA 513",
     xlab = "Time", ylab = "Residuals")
```



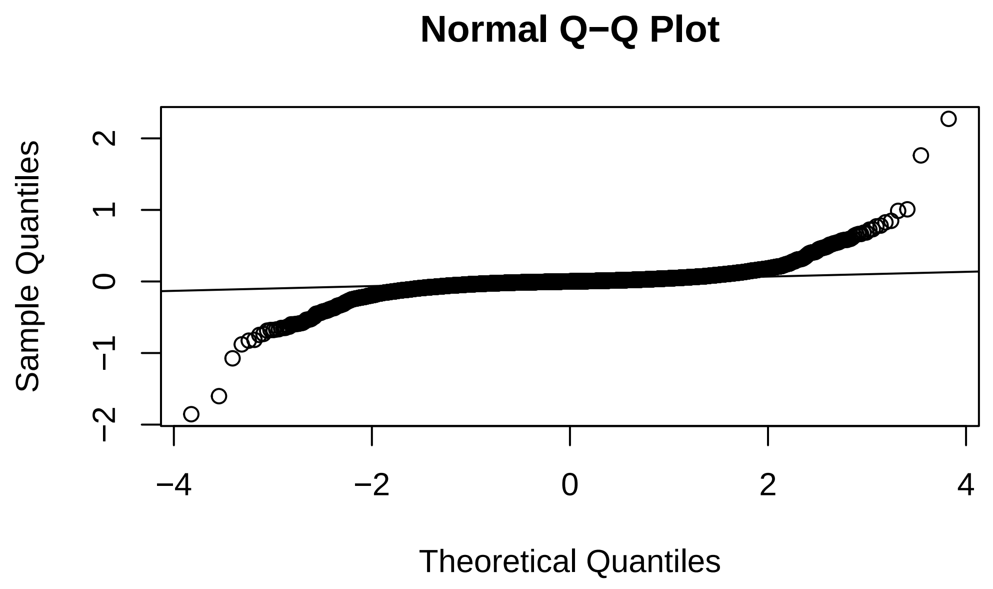
```r
acf(residuals(candidate_513), main = "ACF of Residuals")
```



```
pacf(residuals(candidate_513), main = "PACF of Residuals")
```



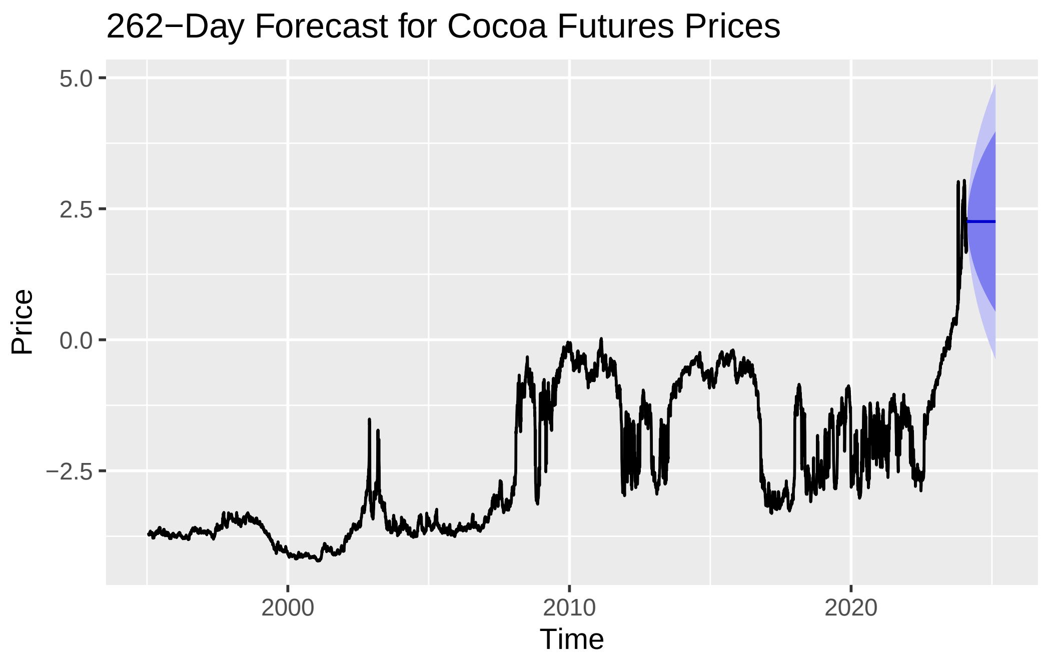
```
qqnorm(residuals(candidate_513))
qqline(residuals(candidate_513))
```



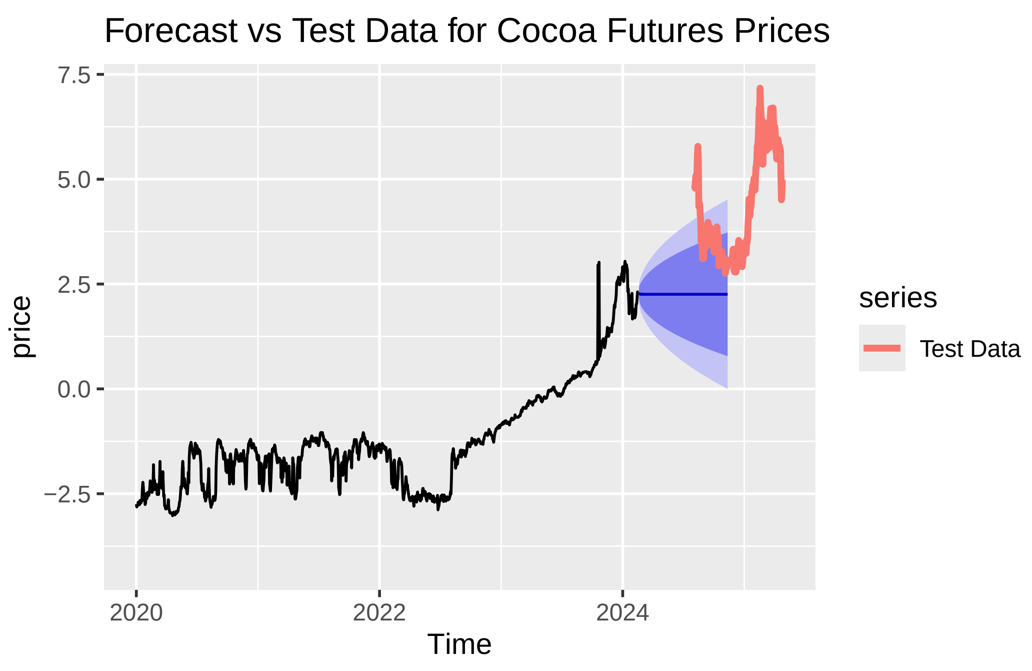
```
lg_test <- Box.test(residuals(candidate_311), lag = 10, type = "Ljung-Box")
print(lg_test)
```

```

    Box-Ljung test

data:  residuals(candidate_311)
X-squared = 38.967, df = 10, p-value = 2.573e-05
```

```
forecast_horizon <- 262
fc <- forecast(candidate_513, h = forecast_horizon)
autoplot(fc)+
  ggtitle("262-Day Forecast for Cocoa Futures Prices")+
  xlab("Time")+
  ylab("Price")
```



```r
horizon <- length(test_ts)
fc <- forecast(candidate_513, h = horizon)
autoplot(fc, PI = TRUE) +
  autolayer(test_ts, series = "Test Data", size = 1.2) +
  ggtitle("Forecast vs Test Data for Cocoa Futures Prices")+
  xlab("Time") +
  ylab("price")+
  scale_x_continuous(limits = c(2020, NA))
```



```r
# 1) Check that both have the same length
length(fc$mean)   # horizon of the forecast
```

```
[1] 192
```

```r
length(test_ts)   # length of the test set
```

```
[1] 192
```

```r
# 2) Convert both to numeric
fc_vec   <- as.numeric(fc$mean)
test_vec <- as.numeric(test_ts)

# 3) Confirm they match in length
if(length(fc_vec) != length(test_vec)){
  stop("Forecast length != Test set length. Adjust horizon or test split.")
}

# 4) Now do accuracy on numeric vectors
accuracy(fc_vec, test_vec)
```

```
                ME     RMSE      MAE     MPE    MAPE
Test set 2.097244 2.448376 2.097244 43.9061 43.9061
```