Visionary Final

Justin DePue, John Rooney, and Tony Jiang

2023-04-27

Contents

- Knitting commands in code chunks:
  - `include = FALSE` - code is run, but neither code nor results appear in knitted file
  - `echo = FALSE` - code not included in knitted file, but results are
  - `eval = FALSE` - code is not run in the knitted file
  - `message = FALSE` - messages do not appear in knitted file
  - `warning = FALSE` - warnings do not appear...
  - `fig.cap = "..."` - adds a caption to graphical results

##Introduction This study was born out of both curiosity and necessity. One of our teammates faced higher than expected energy bills this winter and was suddenly faced with the question of whether to shiver to save money or spend it on heating bills and be forced to eat Spaghetti-O's as sustenance. While spring has sprung and summer is on the horizon, we know that winter is coming and that we should prepare now in order that history not repeat itself.

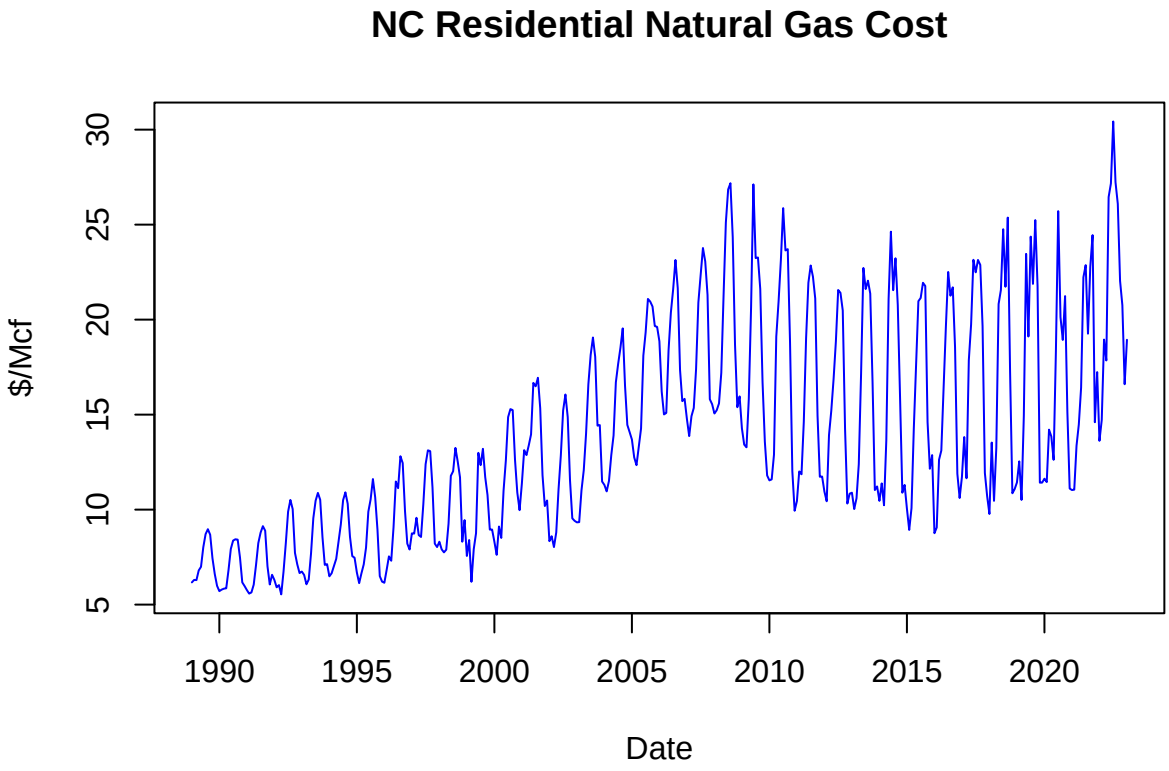
As students recently armed with the tools to conduct time series analysis and forecasting, we realized we could complete our final project while aiding our teammate with knowledge for the future. Our study question thus emerged: would it be more cost-effective to heat an apartment in North Carolina using electricity or a natural-gas powered heater?

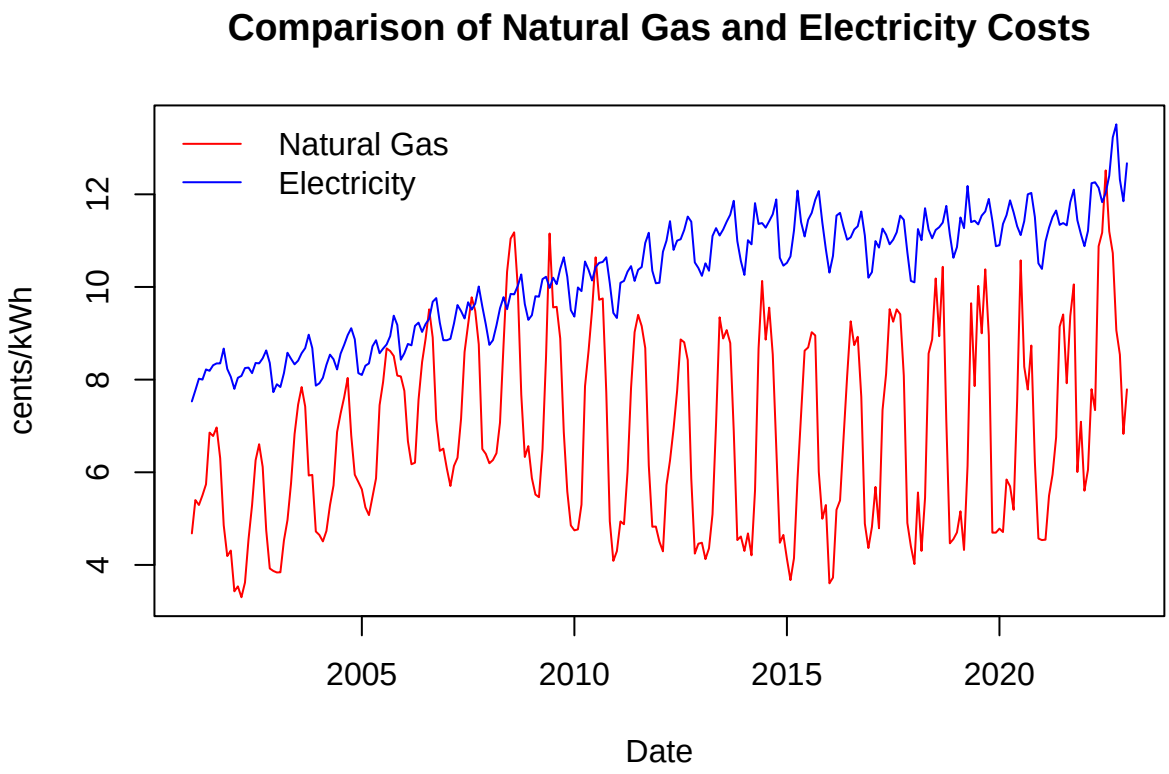
Full of optimism that we could complete a class requirement while doing some good for the world (for as Marvel taught us, when you help someone, you help everyone), we set out to find the data that would lead us to the answer we sought. Our journey led us to that great repository of energy knowledge, the US Energy Information Administration. There we found two datasets we felt confident would help us help our teammate: "North Carolina Price of Natural Gas Delivered to Residential Customers (Dollars Per Thousand Cubic Feet)", which contained monthly data from January 1989 through January 2023, and "Average Retail Price of Electricity by State and Sector" which contained monthly data from January 2001 through January 2023.

##Data

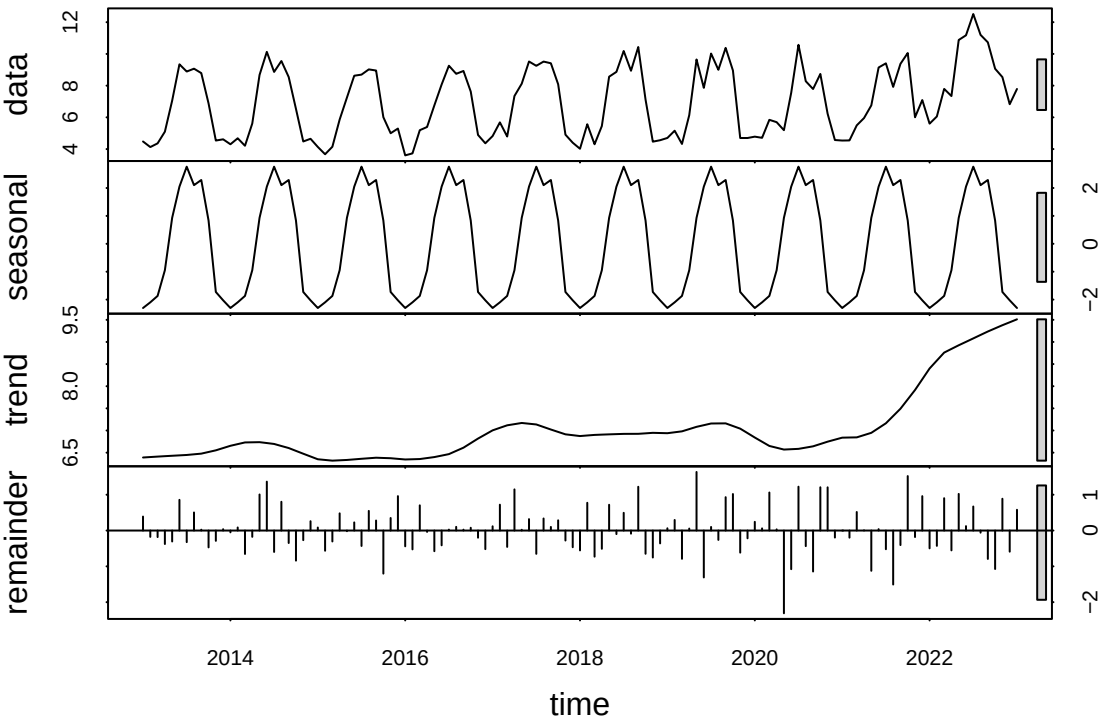
##Analysis We first created initial time series objects and plotted them, along with ACF and PACF plots to gain an initial sense of what the series looked like and what seasonality they may have.

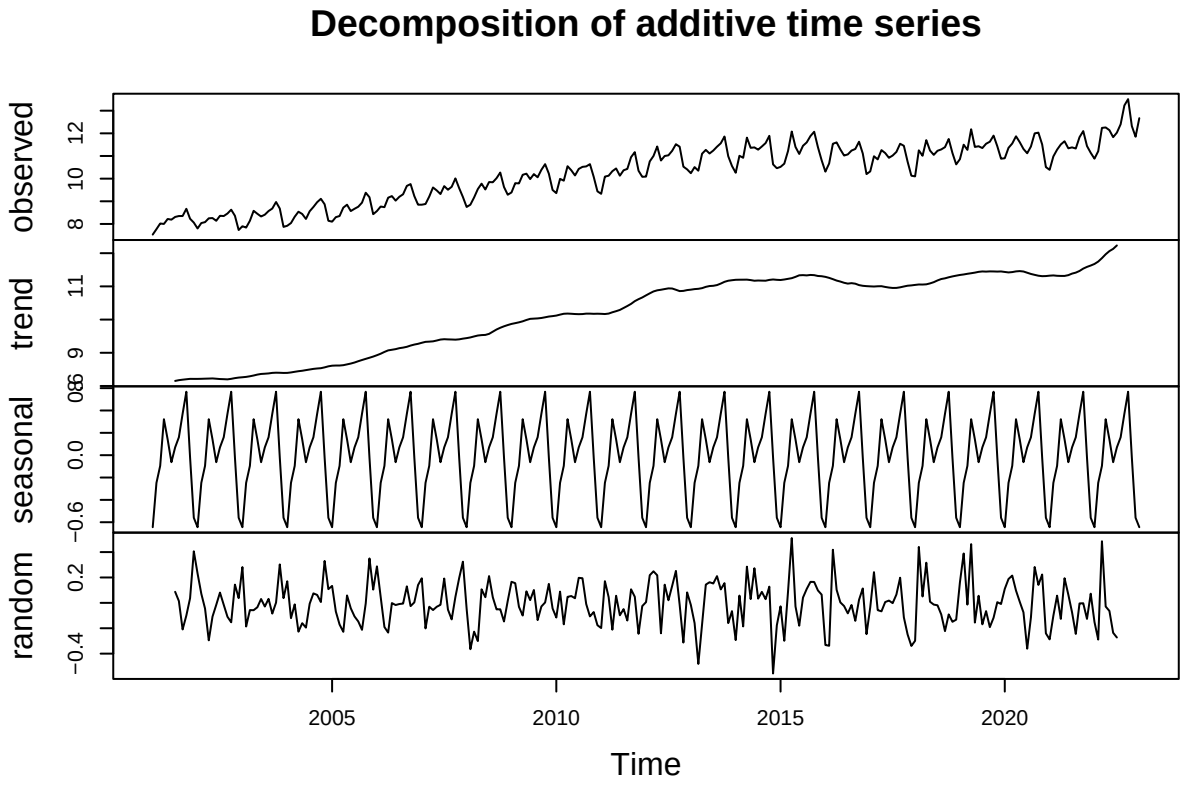
```r
# plot raw time series
plot(ts_NG, col="blue", ylab="$/Mcf", xlab="Date", main="NC Residential Natural Gas Cost")
```

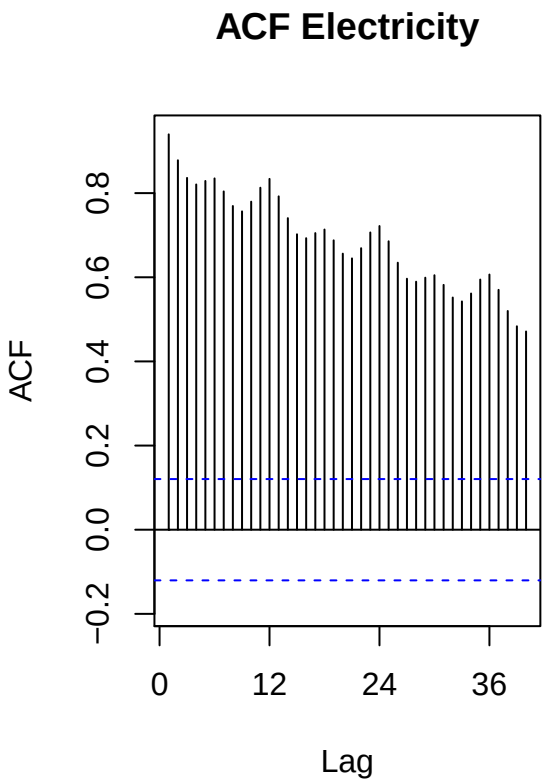


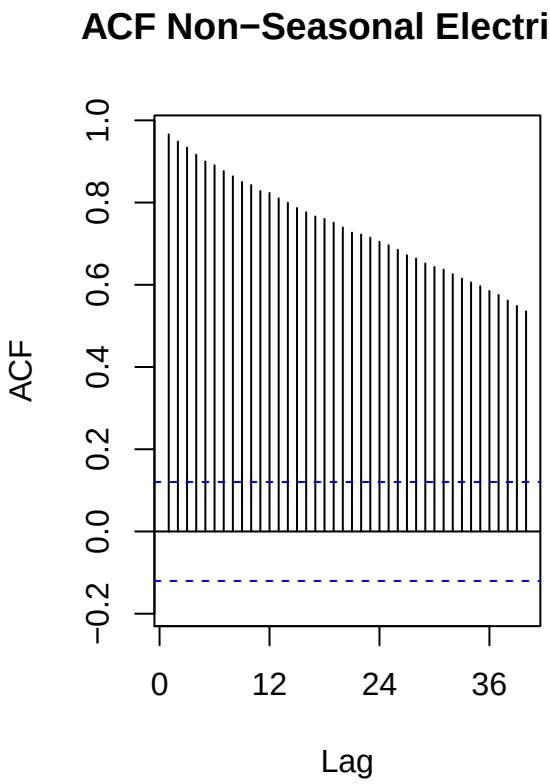


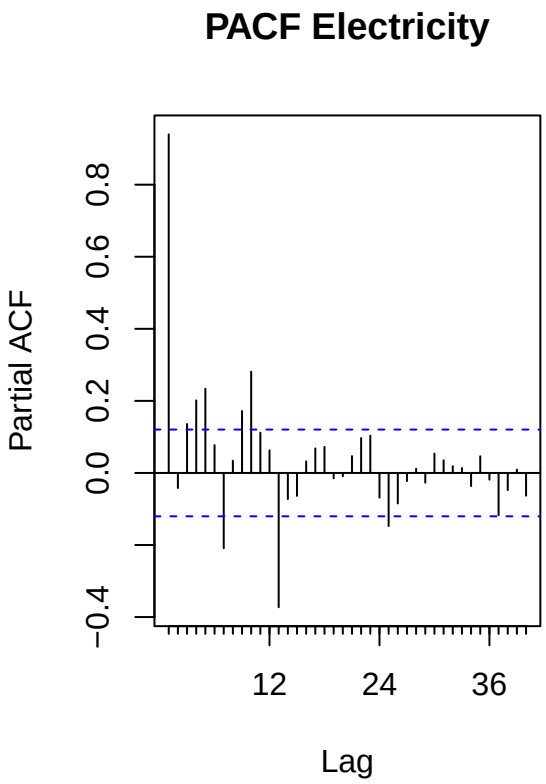
We then decomposed the time series objects for further analysis.

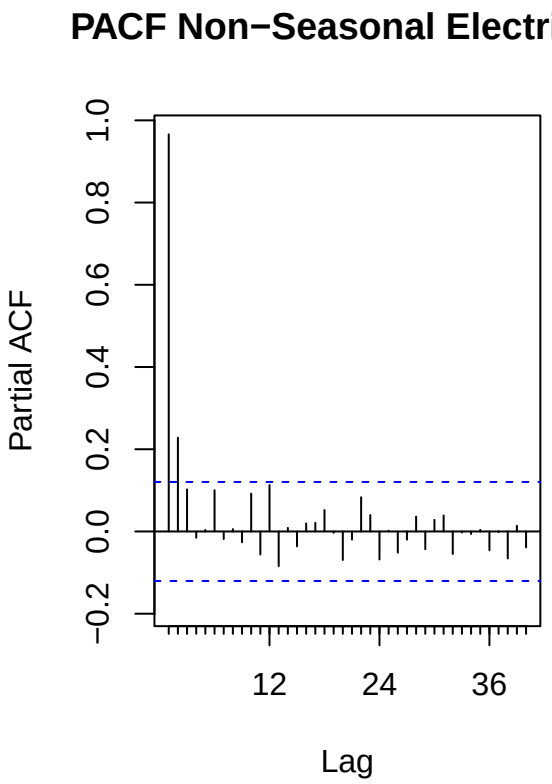

data
seasonal
trend
remainder
4
6
8
12
9.5
8.0
6.5
2
0
−2
1
0
−2
2014
2016
2018
2020
2022
time

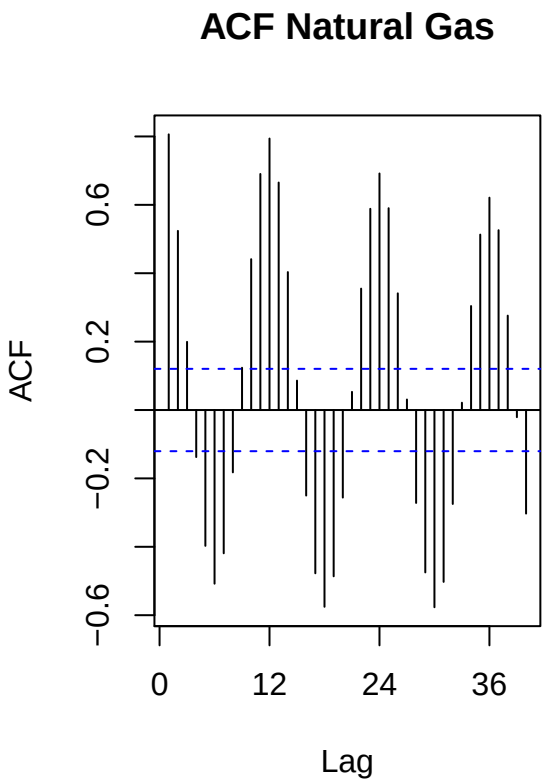

Decomposition of additive time series
observed
trend
seasonal
random
8
10
12
9
11
0.0
−0.6
0.2
−0.4
2005
2010
2015
2020
Time

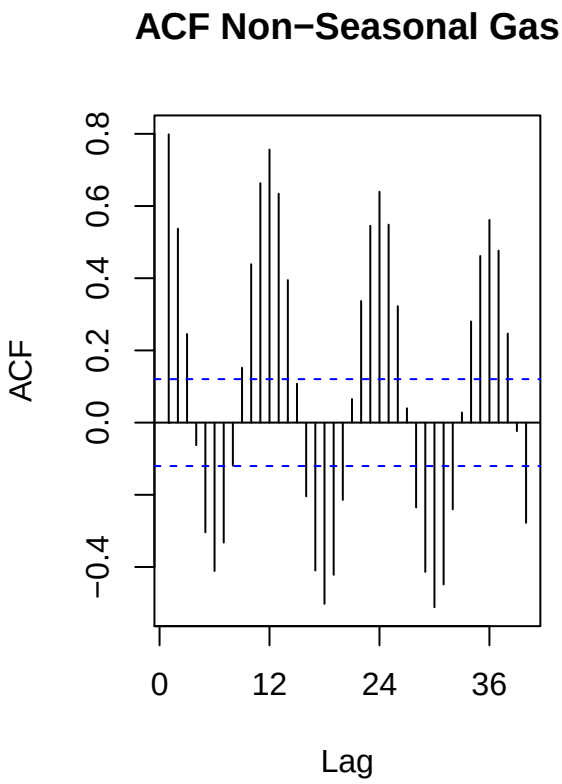

ACF Electricity
ACF
−0.2
0.0
0.2
0.4
0.6
0.8
0
12
24
36
Lag

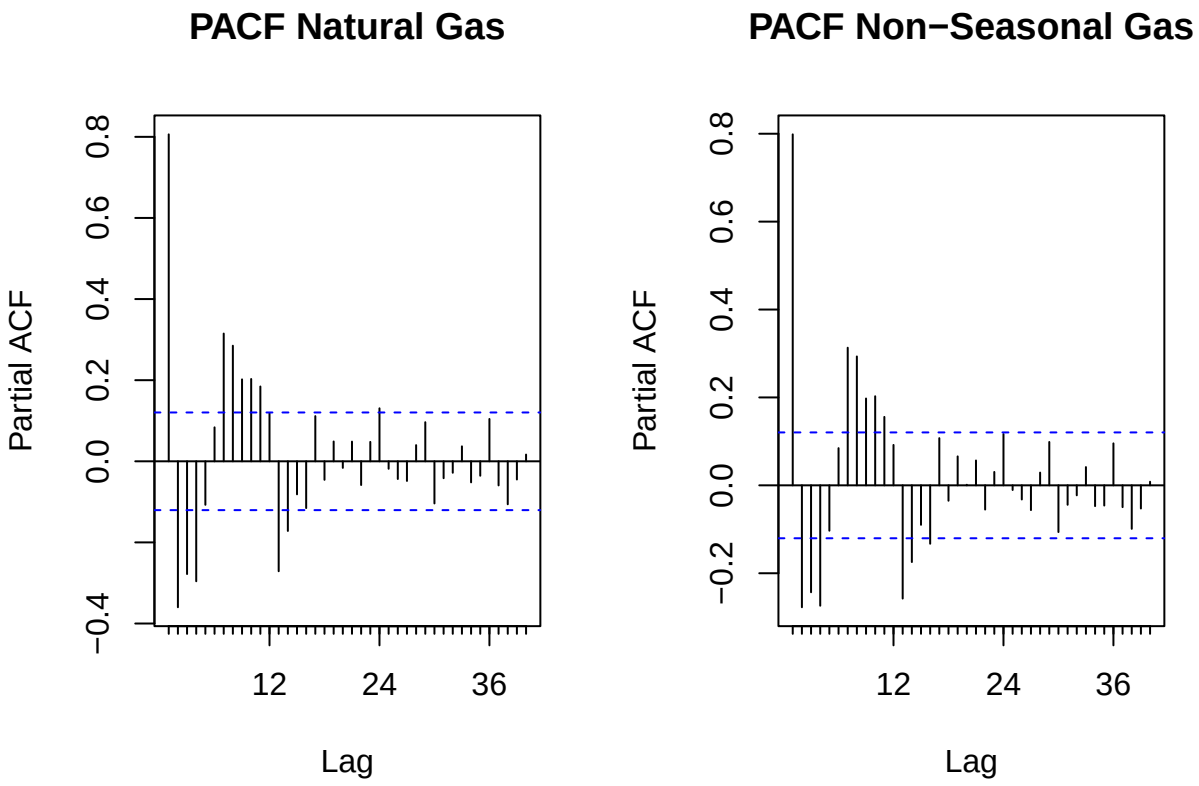

ACF Non−Seasonal Electricity
ACF
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
0
12
24
36
Lag


PACF Electricity
Partial ACF
−0.4
0.0
0.2
0.4
0.6
0.8
12
24
36
Lag


PACF Non−Seasonal Electricity
Partial ACF
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
12
24
36
Lag


ACF Natural Gas
ACF
0.6
0.2
−0.2
−0.6
0
12
24
36
Lag

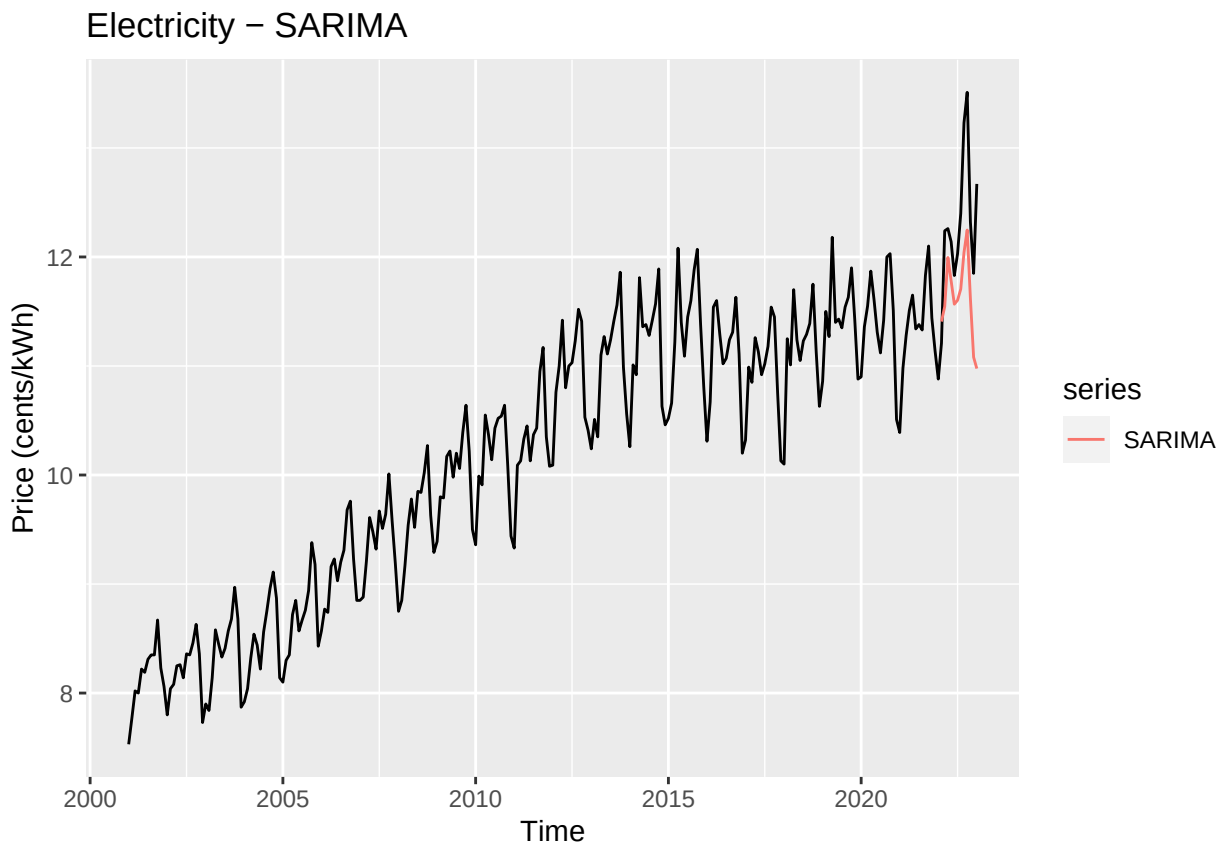

ACF Non−Seasonal Gas
ACF
0.8
0.6
0.4
0.2
0.0
−0.4
0
12
24
36
Lag

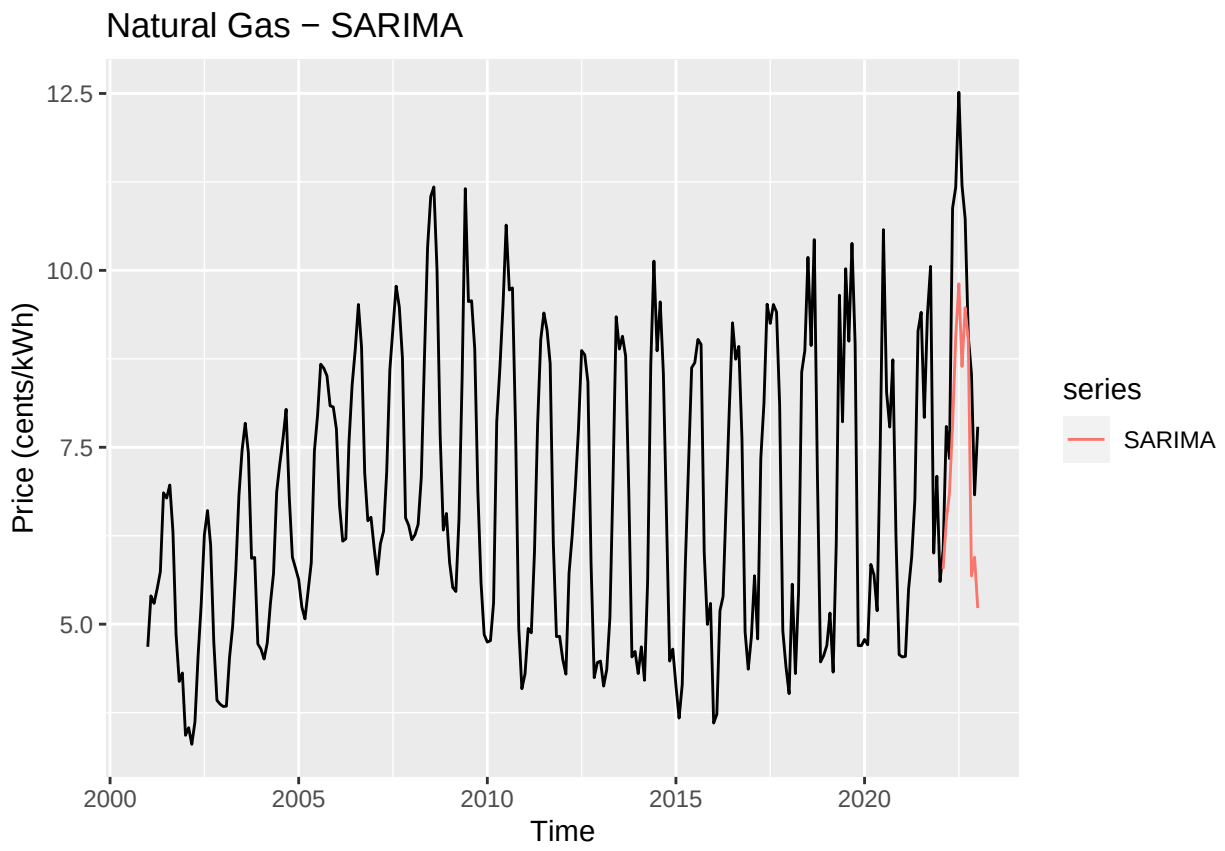


Several models were developed and tested to determine what would best fit the data we had and what may lead to the best forecast to determine whether heating via electricity or natural gas would be most cost-efficient in the upcoming winter. The five that were used were the Seasonal ARIMA, ARIMA with Fourier terms, Neural Networks, TBATS, and STL + EST.

Each of these tests used functions from the "forecast" library and required the use of time series data which we created using the "ts" function from the "tseries" library.

While the Seasonal ARIMA was a logical first model to try, we quickly felt that the performance was not particularly strong.


Electricity − SARIMA
Price (cents/kWh)
12
10
8
2000
2005
2010
2015
2020
Time
series
SARIMA

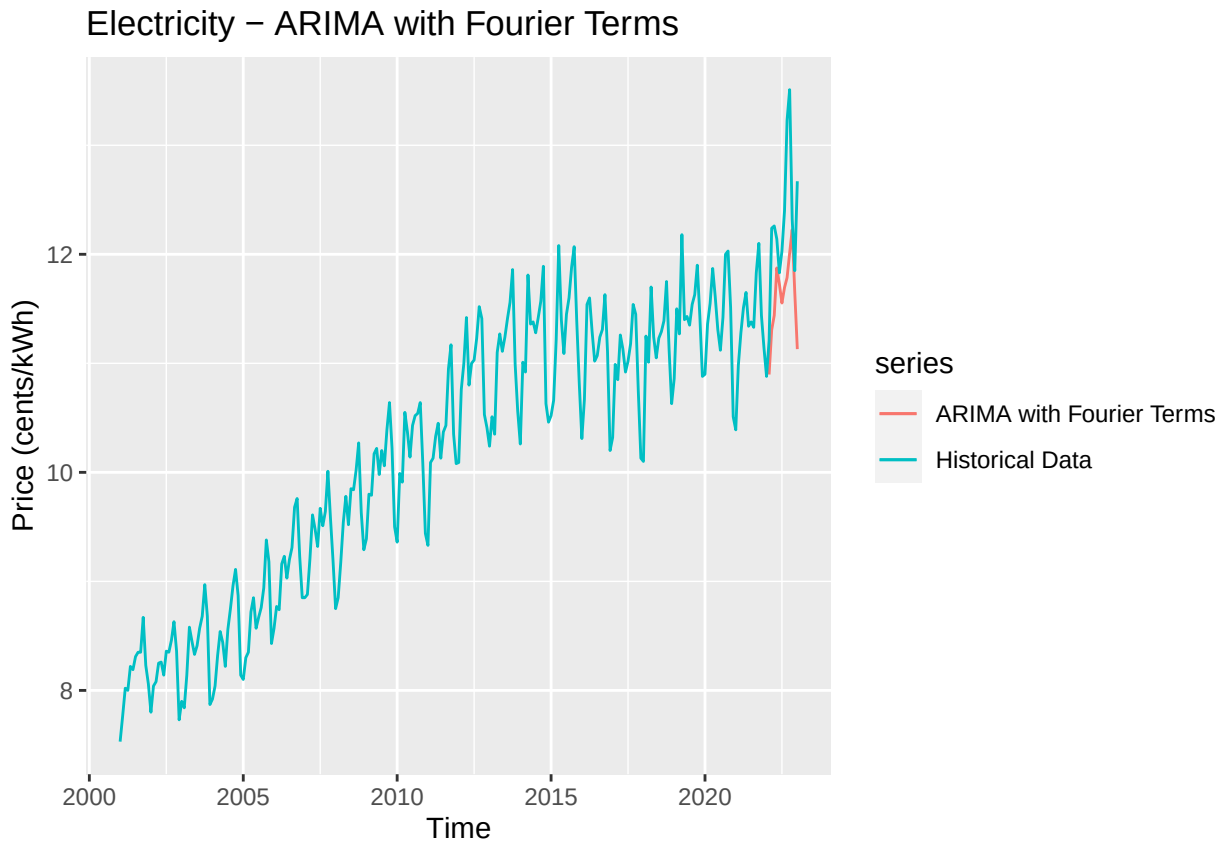



```
# model performance for electricity data
sarima_e_perf <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(arima.e.forecast$mean)


# model performance for gas data
sarima_gas_perf <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(arima.gas.forecast$mean)
```

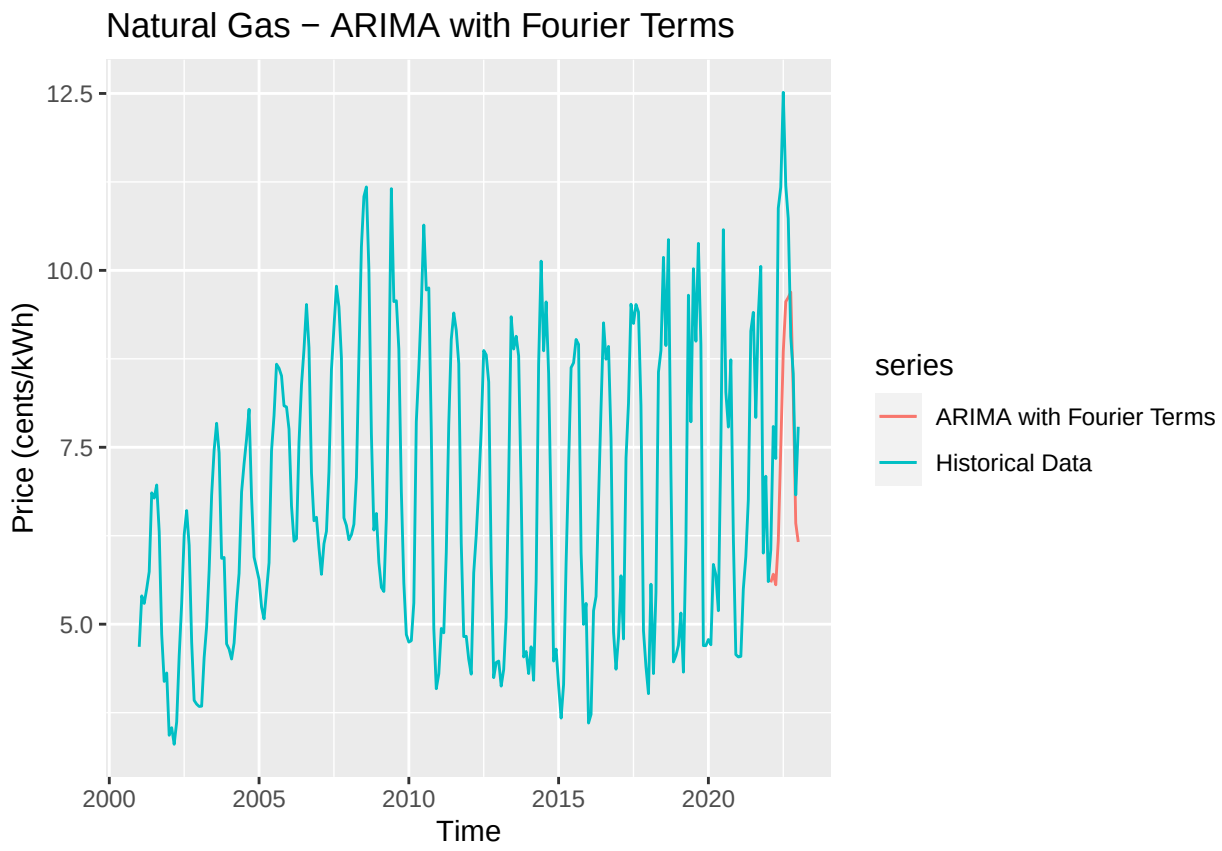
We decided to explore more advanced models that could handle the ARIMA with Fourier Terms model, Neural Networks, TBATS, and STL + EST. ARIMA with Fourier terms is known as a dynamic harmonic regression model with an ARMA error structure, using the "fourier" function from package "forecast" to find terms that model seasonal components.

```
# arima with fourier terms for eletricity
arima.e.four.forecast <- ts_electricity %>%
  window(end = c(2022, 1)) %>%
  auto.arima(seasonal = FALSE,
             xreg = fourier(window(ts_electricity, end = c(2022, 1)),
                            K = 6)) %>%
  forecast(xreg = fourier(window(ts_electricity,
                                 start = c(2022,2)
                                 ),
```

```
                            K = 6),
            h = 12)

autoplot(arima.e.four.forecast$mean, series = "ARIMA with Fourier Terms") +
  autolayer(ts_electricity, series = "Historical Data") +
  ggtitle("Electricity - ARIMA with Fourier Terms") +
  ylab("Price (cents/kWh)")
```



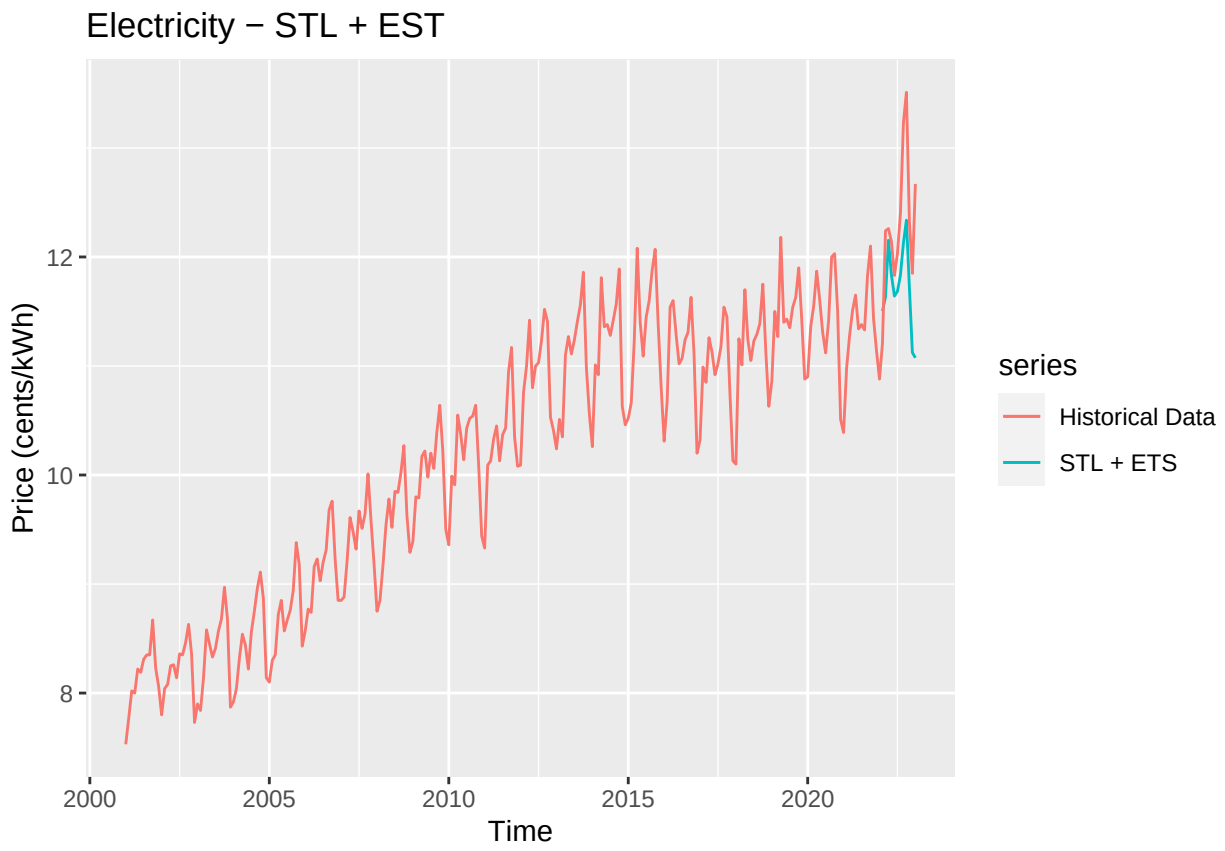

```
# arima with fourier terms for gas
arima.gas.four.forecast <- ts_gas_equiv %>%
  window(end = c(2022, 1)) %>%
  auto.arima(seasonal = FALSE,
             xreg = fourier(window(ts_gas_equiv, end = c(2022, 1)),
                            K = 6)
             ) %>%
  forecast(xreg = fourier(window(ts_gas_equiv,
                                 start = c(2022, 2)
                                 ),
                          K = 6),
           h = 12)

autoplot(arima.gas.four.forecast$mean, series = "ARIMA with Fourier Terms") +
  autolayer(ts_gas_equiv, series = "Historical Data") +
  ggtitle("Natural Gas - ARIMA with Fourier Terms") +
  ylab("Price (cents/kWh)")
```



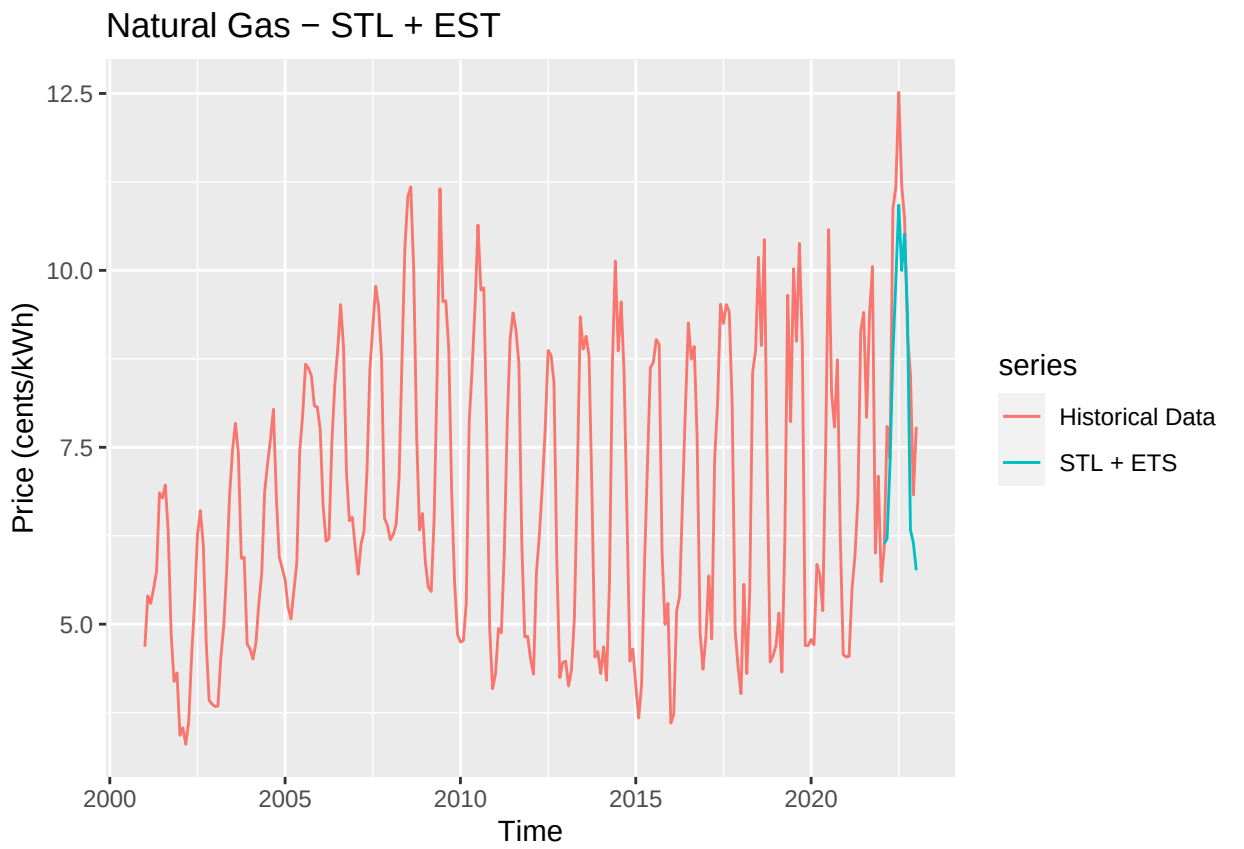

```r
# model performance for electricity data
arima_four_e_perf <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(arima.e.four.forecast$mean)

# model performance for gas data
arima_four_gas_perf <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(arima.gas.four.forecast$mean)
```

Then we tried STL model

```r
# STL for electricity
stl.e.forecast <- ts_electricity %>%
  window(end = c(2022, 1)) %>%
  stlf(h = 12)

autoplot(stl.e.forecast$mean, series = "STL + ETS") +
  autolayer(ts_electricity, series = "Historical Data") +
  ggtitle("Electricity - STL + EST") +
  ylab("Price (cents/kWh)")
```




```
# STL for gas
stl.gas.forecast <- ts_gas_equiv %>%
  window(end = c(2022, 1)) %>%
  stlf(h = 12)

autoplot(ts_gas_equiv, series = "Historical Data") +
  autolayer(stl.gas.forecast$mean, series = "STL + ETS") +
  ggtitle("Natural Gas - STL + EST") +
  ylab("Price (cents/kWh)")
```




```
# model performance for electricity data
stl_e_perf <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(stl.e.forecast$mean)

# model performance for gas data
stl_gas_perf <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(stl.gas.forecast$mean)
```

Then we tried to use neural network model. We used nnetar() in forecast package. We figured out p and P arguments in nnetar() have significant impact on model performance. Therefore, we first tried to identify the optimal p and P conbination by trying different combinations.

```
# find the optimal p and P values for neural network without fourier for electricity series

## p = 1, P = 0
nn.e.forecast.10 <- ts_electricity %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 0, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.e.10.score <- ts_electricity %>%
```

```
  window(start = c(2022, 2)) %>%
  accuracy(nn.e.forecast.10$mean)

## p = 1, P = 1
nn.e.forecast.11 <- ts_electricity %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 1, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.e.11.score <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.e.forecast.11$mean)

## p = 2, P = 0
nn.e.forecast.20 <- ts_electricity %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 2, P = 0, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.e.20.score <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.e.forecast.20$mean)

## p = 2, P = 1
nn.e.forecast.21 <- ts_electricity %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 2, P = 1, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.e.21.score <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.e.forecast.21$mean)

## p = 2, P = 2
nn.e.forecast.22 <- ts_electricity %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 2, P = 2, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.e.22.score <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.e.forecast.22$mean)

## p = 1, P = 2
nn.e.forecast.12 <- ts_electricity %>%
```

```
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 2, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.e.12.score <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.e.forecast.12$mean)

## p = 3, P = 1
nn.e.forecast.31 <- ts_electricity %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 3, P = 1, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.e.31.score <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.e.forecast.31$mean)

nn.e.scores <- rbind(nn.e.10.score, nn.e.11.score, nn.e.20.score, nn.e.21.score,
                     nn.e.22.score, nn.e.12.score, nn.e.31.score)

row.names(nn.e.scores) <- c("10", "11", "20", "21", "22", "12", "31")

nn.e.scores ## 11 has the lowest RMSE and 12 has the lowest MAPE
```

```
##             ME      RMSE       MAE        MPE      MAPE      ACF1 Theil's U
## 10 -1.1711716 1.2950718 1.1711716 -10.501360 10.501360 0.2845607 45.328113
## 11 -0.8743250 0.9877212 0.8743250  -7.619533  7.619533 0.1905909  4.832336
## 20 -1.0195841 1.1506183 1.0195841  -9.012901  9.012901 0.2914592 18.696382
## 21 -0.8822289 0.9965406 0.8822289  -7.696598  7.696598 0.2066680  4.744910
## 22 -0.8722379 1.0001477 0.8722379  -7.615647  7.615647 0.2315309  4.571394
## 12 -0.8724708 0.9948040 0.8724708  -7.610827  7.610827 0.2059075  4.758609
## 31 -0.8925567 1.0096805 0.8925567  -7.798114  7.798114 0.2206365  4.711655
```

```
# we can do the same thing to natural gas. Do we want to??
```

```
# find the optimal p and P values for neural network without fourier for natural gas series

## p = 1, P = 0
nn.gas.forecast.10 <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 0, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.gas.10.score <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast.10$mean)
```

```r
## p = 1, P = 1
nn.gas.forecast.11 <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 1, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.gas.11.score <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast.11$mean)

## p = 2, P = 0
nn.gas.forecast.20 <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 2, P = 0, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.gas.20.score <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast.20$mean)

## p = 2, P = 1
nn.gas.forecast.21 <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 2, P = 1, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.gas.21.score <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast.21$mean)

## p = 2, P = 2
nn.gas.forecast.22 <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 2, P = 2, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.gas.22.score <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast.22$mean)

## p = 1, P = 2
nn.gas.forecast.12 <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 2, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
```

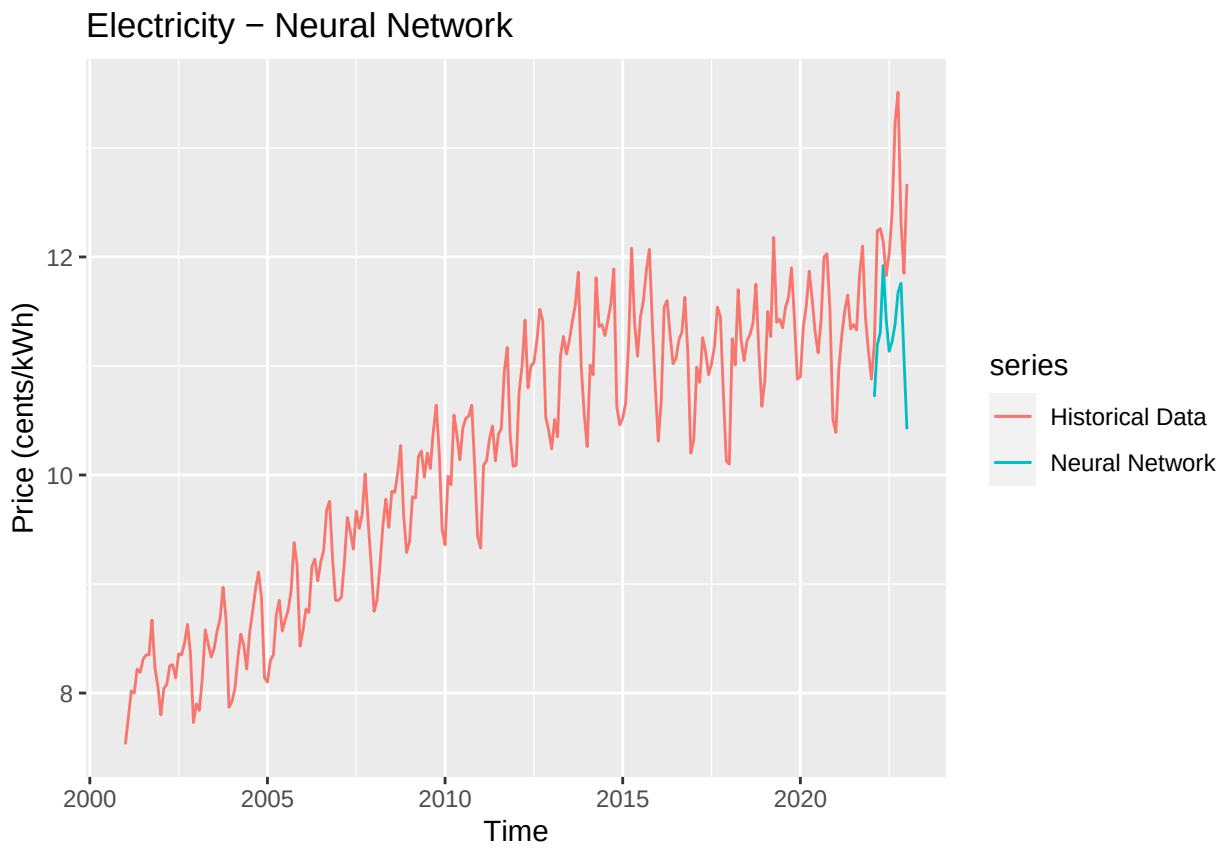
```
        ) %>%
  forecast(h = 12)

nn.gas.12.score <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast.12$mean)

## p = 3, P = 1
nn.gas.forecast.31 <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 3, P = 1, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.gas.31.score <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast.31$mean)

## p = 3, P = 2
nn.gas.forecast.32 <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 3, P = 1, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.gas.32.score <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast.32$mean)

## p = 3, P = 3
nn.gas.forecast.33 <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 3, P = 3, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         ) %>%
  forecast(h = 12)

nn.gas.33.score <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast.33$mean)

nn.gas.scores <- rbind(nn.gas.10.score, nn.gas.11.score, nn.gas.20.score, nn.gas.21.score,
                       nn.gas.22.score, nn.gas.12.score, nn.gas.31.score, nn.gas.32.score,
                       nn.gas.33.score)

row.names(nn.gas.scores) <- c("10", "11", "20", "21", "22", "12", "31", "32", "33")

nn.gas.scores ## 32 has the lowest RMSE and 11 has the lowest MAPE
```

```
##           ME     RMSE      MAE       MPE     MAPE      ACF1 Theil's U
## 10 -2.289927 3.002493 2.394917 -33.51945 34.92729 0.6285854 18.468526
```

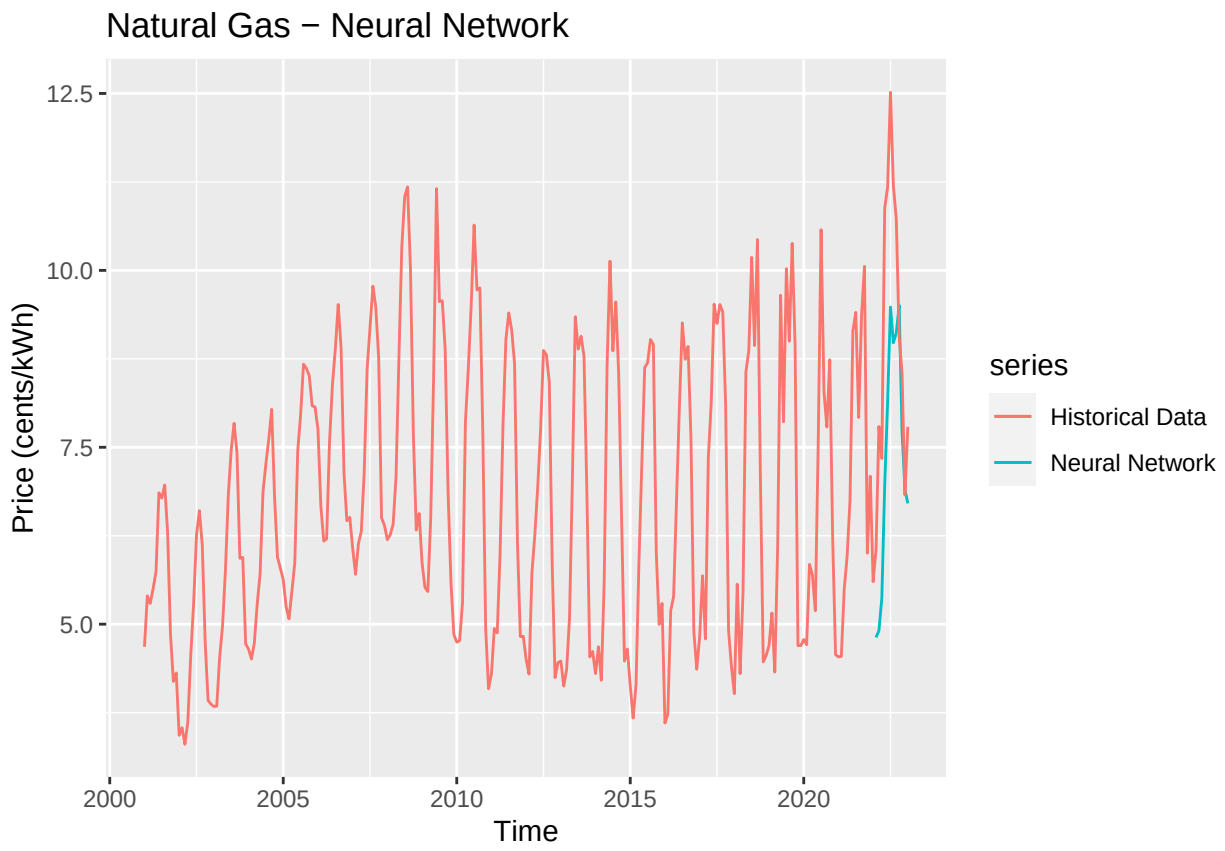
```
## 11 -1.790817 2.248296 1.977540 -25.62372 27.87065 0.2982629  2.463158
## 20 -2.541010 3.306837 2.702199 -40.81255 42.88047 0.6738659 12.271815
## 21 -1.877266 2.208028 1.949940 -27.13111 27.89601 0.2279845  2.516369
## 22 -2.014205 2.365200 2.060867 -30.18017 30.67949 0.2734526  2.665208
## 12 -1.922977 2.336096 1.993198 -28.06199 28.80222 0.2846316  2.608177
## 31 -1.906698 2.244569 1.966807 -28.06157 28.69929 0.2626046  2.541930
## 32 -1.892828 2.234236 1.949783 -27.89700 28.50247 0.2729119  2.582334
## 33 -1.975839 2.359902 2.074002 -29.53550 30.55231 0.3213647  2.644206
```

We found that the combination 11 (p = 1 and P = 1) has the best modeling performance for electricity series, and the combination 32 (p = 3 and P = 2) has the best modeling performance for natural gas series. Then we use this combination to run the neural network model with fourier terms for elecricity series.

```
# neural network forecast for electricity data
nn.e.forecast <- ts_electricity %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 1, # 2 and 1 are randomly decided. To find the optimal one,
         # we need to run a couple of combinations to find the best one
         xreg = fourier(window(ts_electricity,
                               end = c(2022, 1)
                               ),
                        K = 6)
         ) %>%
  forecast(xreg = fourier(window(ts_electricity,
                                 start = c(2022, 2)
                                 ),
                          K = 6),
           h = 12)

# neural network forecast for gas data
nn.gas.forecast <- ts_gas_equiv %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 1,
         xreg = fourier(window(ts_gas_equiv,
                               end = c(2022, 1)
                               ),
                        K = 6)
         ) %>%
  forecast(xreg = fourier(window(ts_gas_equiv,
                                 start = c(2022, 2)
                                 ),
                          K = 6),
           h = 12)


autoplot(nn.e.forecast$mean, series = "Neural Network") +
  autolayer(ts_electricity, series = "Historical Data") +
  ggtitle("Electricity - Neural Network") +
  ylab("Price (cents/kWh)")
```




```
autoplot(nn.gas.forecast$mean, series = "Neural Network") +
  autolayer(ts_gas_equiv, series = "Historical Data") +
  ggtitle("Natural Gas - Neural Network") +
  ylab("Price (cents/kWh)")
```




```
# neutral network model performance

# neural network model performance for electricity data
nn_e_perf <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.e.forecast$mean)

# neural network model performance for gas data
nn_gas_perf <- ts_gas_equiv %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.gas.forecast$mean)
```

Then we tried TBATS models for electricity and natural gas series

```
# TBATS model

# TBATS for electricity
tbats.e.forecast <- ts_electricity %>%
  window(end = c(2022, 1)) %>%
  tbats() %>%
  forecast(h = 12)

# TBATS for natural gas
tbats.gas.forecast <- ts_gas_equiv %>%
  window(end = c(2022, 1)) %>%
```

```
  tbats() %>%
  forecast(h = 12)
```

```
# TBATS model performance

# TBATS model performance for electricity data
tbats_e_perf <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(tbats.e.forecast$mean)

# TBATS model performance for gas data
tbats_gas_perf <- ts_gas_equiv%>%
  window(start = c(2022, 2)) %>%
  accuracy(tbats.gas.forecast$mean)
```

Then we think Ukraine War should have a significant impact on natural gas price and probably eletricity gas too. Also, temperature should be a good regressor to include since utility bills normally fluctuate in the same direction with temperature. Therefore, we created two covariates: UKRWAR and temperature. $UKRWAR$ is an indiciator variable with values of 0 and 1. Months before March 2022 have a value of 0, while months after and including March 2022 have a value of 1. The reason why we set the cutoff month at March 2002 despite the war started from last February is because the impact of the war on monthly natural gas price in February 2022 should be limited since the war started in late February. The temperature series is the monthly average temperature of Raleigh area. This is largest geographic level of historical temperature data.

After creating all the covariates, we repeated our modeling but with covariates to improve the accuracy of the models. First we incorporated covairates to neural network model.

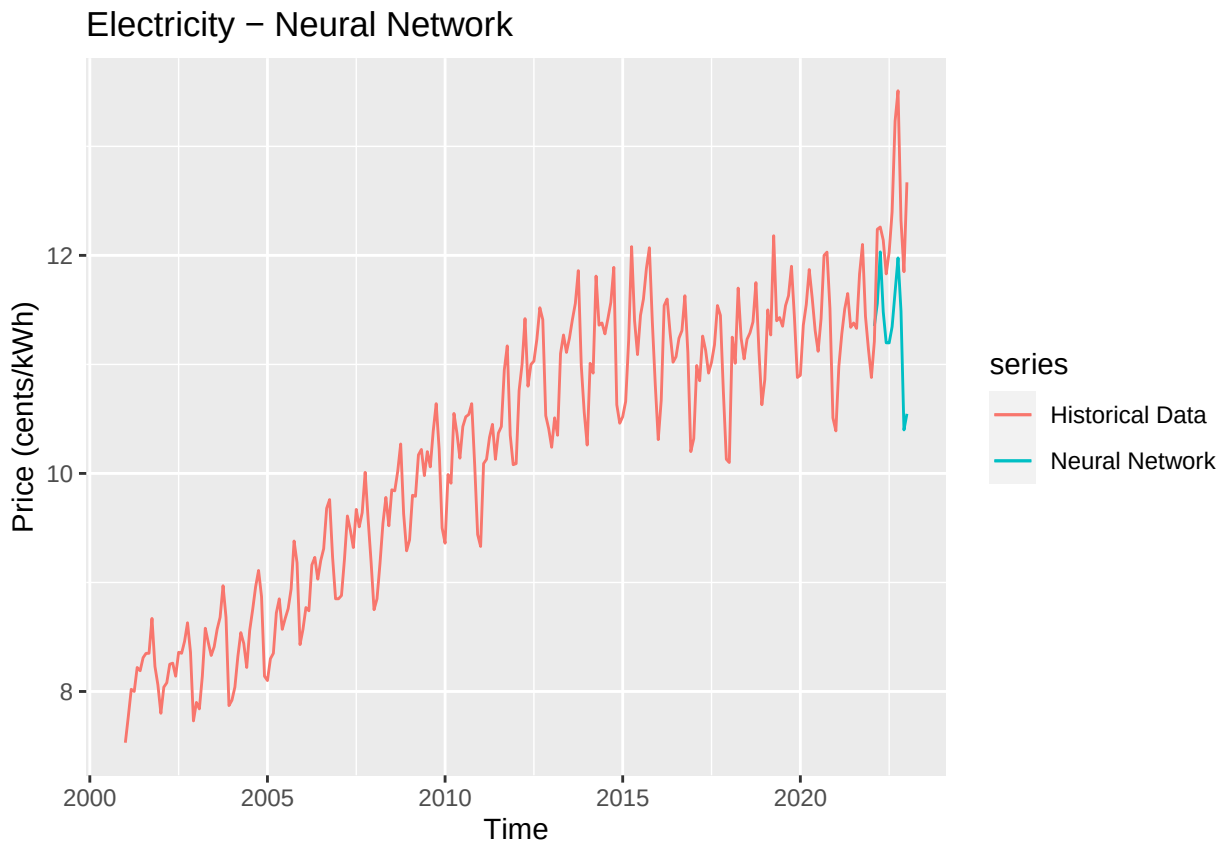
```
# Use Neural Network, temperature, UKRWAR, and fourier to model

# neural network forecast for electricity data
nn.cov.e.forecast <- ts_electricity_war[, 2] %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 1,
         xreg = covariates_train_e[, -1]) %>%
  forecast(xreg = window(covariates_full_e[, -1], start = c(2022, 2)),
           h = 12)
```

```
## Warning in forecast.nnetar(., xreg = window(covariates_full_e[, -1], start =
## c(2022, : xreg contains different column names from the xreg used in training.
## Please check that the regressors are in the same order.
```

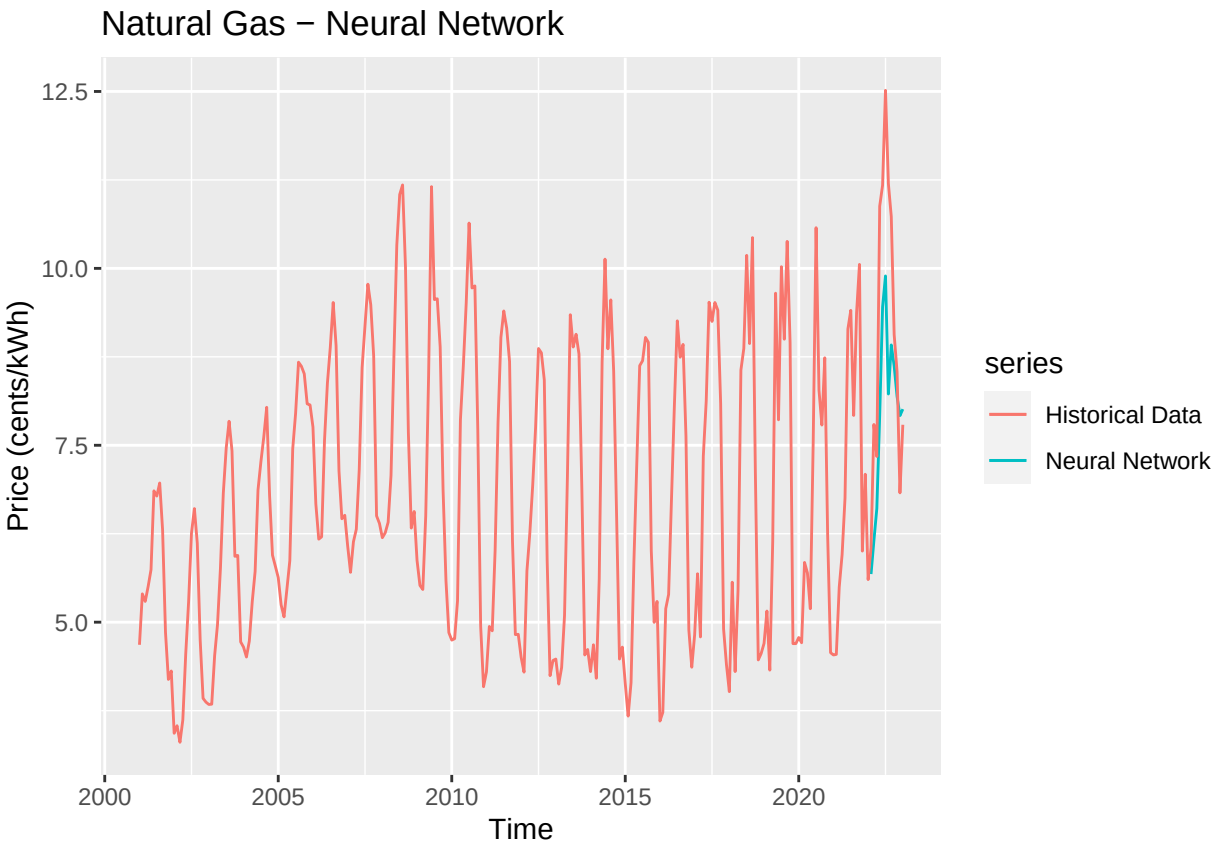
```
# neural network forecast for gas data
nn.cov.gas.forecast <- ts_gas_equiv_war[, 1] %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 1,
         xreg = covariates_train_gas[, -1]) %>%
  forecast(xreg = window(covariates_full_gas[, -1], start = c(2022, 2)),
           h = 12)
```

```
## Warning in forecast.nnetar(., xreg = window(covariates_full_gas[, -1], start =
## c(2022, : xreg contains different column names from the xreg used in training.
## Please check that the regressors are in the same order.
```

```
autoplot(nn.cov.e.forecast$mean, series = "Neural Network") +
  autolayer(ts_electricity_war[, 2], series = "Historical Data") +
  ggtitle("Electricity - Neural Network") +
  ylab("Price (cents/kWh)")
```




```
autoplot(nn.cov.gas.forecast$mean, series = "Neural Network") +
  autolayer(ts_gas_equiv_war[, 1], series = "Historical Data") +
  ggtitle("Natural Gas - Neural Network") +
  ylab("Price (cents/kWh)")
```




```
# neutral network model, temperature, fourier, UKRWAR performance

# neural network model performance for electricity data
nn_cov_e_perf <- ts_electricity_war[, 2] %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.cov.e.forecast$mean)

# neural network model performance for gas data
nn_cov_gas_perf <- ts_gas_equiv_war[, 1] %>%
  window(start = c(2022, 2)) %>%
  accuracy(nn.cov.gas.forecast$mean)
```

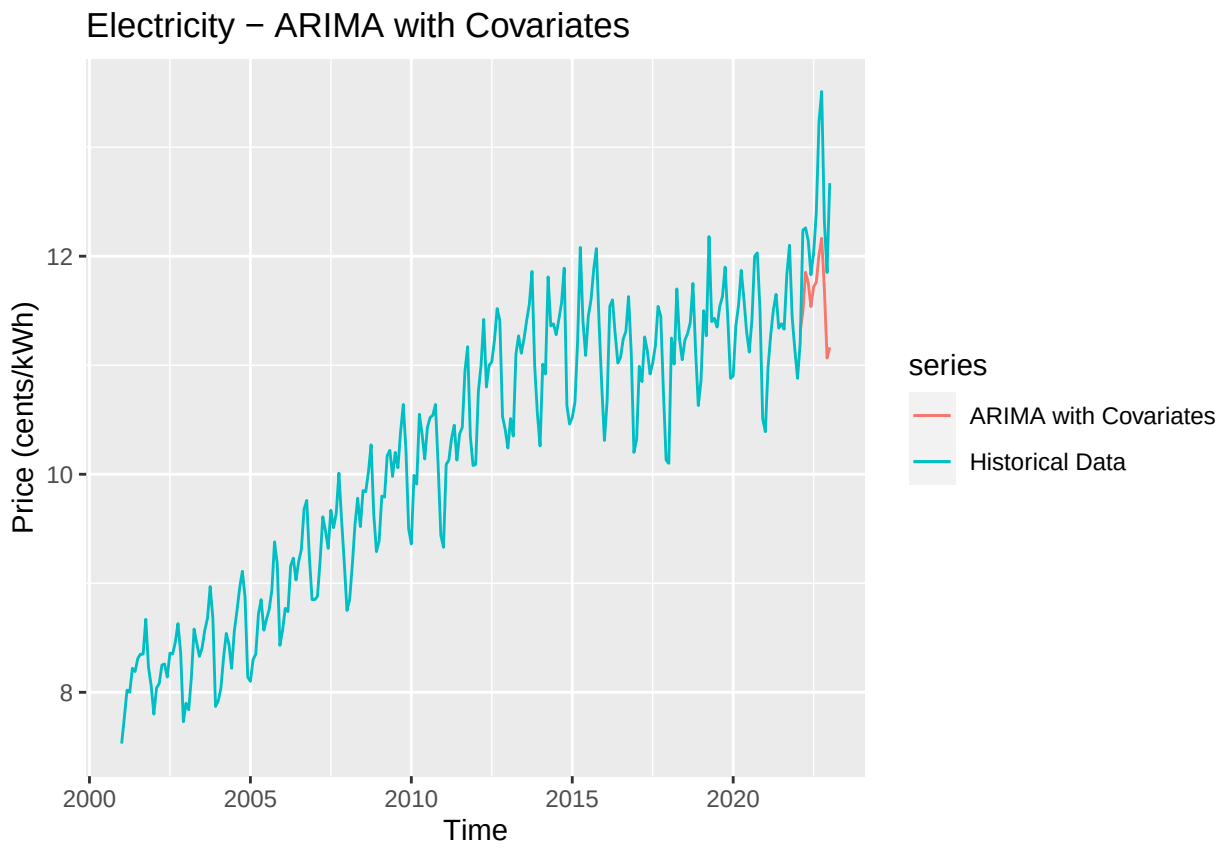
Then we use seasonal arima model with temperature and fourier terms to model two series. UKRWAR is excluded because R reports no suitable ARIMA model when UKRWAR is included. Function used: auto.arima(xreg)

```
# sarima with temperature and fourier terms for electricity

sarima.e.cov.forecast <- ts_electricity_war[, 2] %>%
  window(end = c(2022, 1)) %>%
  auto.arima(seasonal = FALSE,
             xreg = covariates_train_e[, -1]) %>%
  forecast(xreg = window(covariates_full_e[, -1], start = c(2022, 2)),
           h = 12)
```

```
## Warning in forecast.forecast_ARIMA(., xreg = window(covariates_full_e[, : xreg
```

```
## contains different column names from the xreg used in training. Please check
## that the regressors are in the same order.
```

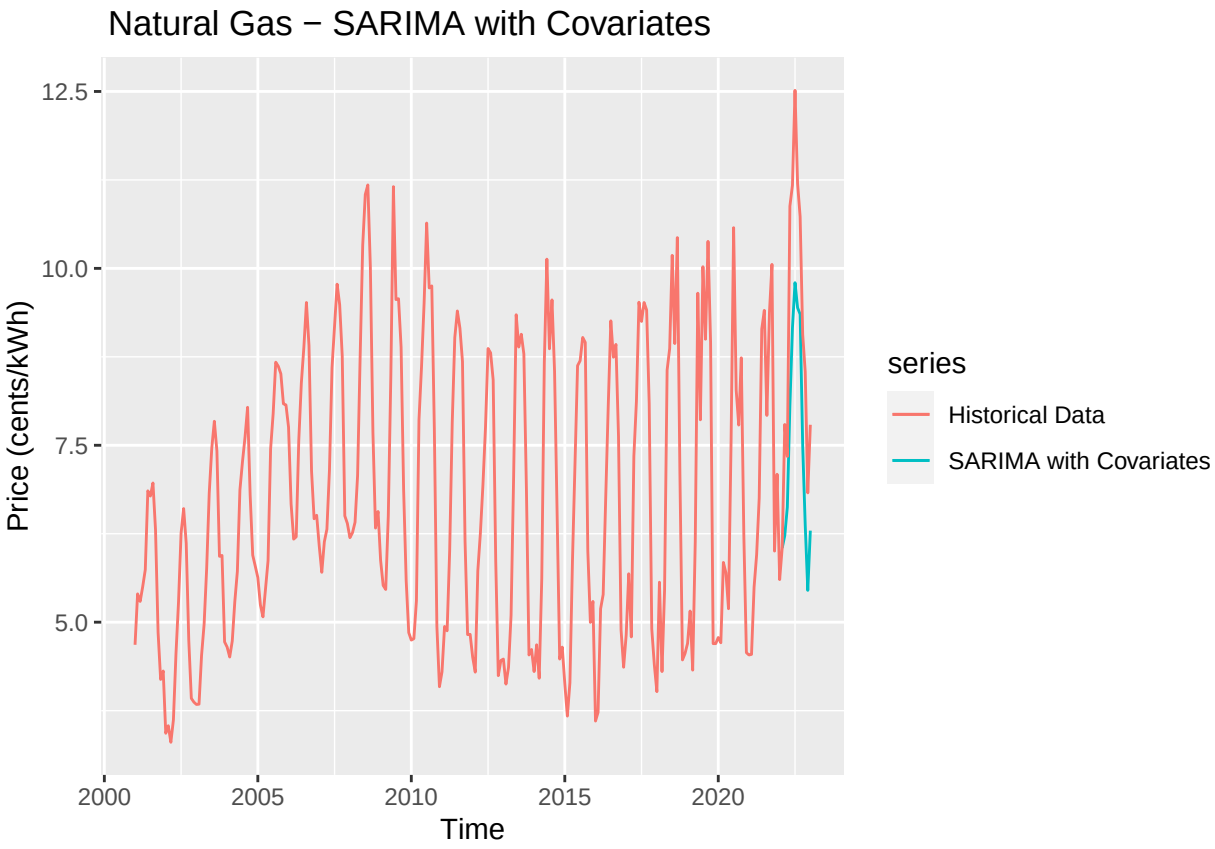
```
autoplot(sarima.e.cov.forecast$mean, series = "ARIMA with Covariates") +
  autolayer(ts_electricity_war[, 2], series = "Historical Data") +
  ggtitle("Electricity - ARIMA with Covariates") +
  ylab("Price (cents/kWh)")
```




```
# arima with covariates for gas
sarima.gas.cov.forecast <- ts_gas_equiv_war[, 1] %>%
  window(end = c(2022, 1)) %>%
  auto.arima(seasonal = FALSE,
             xreg = covariates_train_gas[, -1]) %>%
  forecast(xreg = window(covariates_full_gas[, -1], start = c(2022, 2)),
           h = 12)
```

```
## Warning in forecast.forecast_ARIMA(., xreg = window(covariates_full_gas[, : xreg
## contains different column names from the xreg used in training. Please check
## that the regressors are in the same order.
```

```
autoplot(sarima.gas.cov.forecast$mean, series = "SARIMA with Covariates") +
  autolayer(ts_gas_equiv_war[, 1], series = "Historical Data") +
  ylab("Price (cents/kWh)") +
  ggtitle("Natural Gas - SARIMA with Covariates") +
  theme(plot.title = element_text(hjust = 0.05))
```




```
#Examine Arima with fourier performance on electricity and NG data model performance for electricity da
sarima_cov_e_perf <- ts_electricity_war[, 2] %>%
  window(start = c(2022, 2)) %>%
  accuracy(sarima.e.cov.forecast$mean)

# model performance for gas data
sarima_cov_gas_perf <- ts_gas_equiv_war[, 1] %>%
  window(start = c(2022, 2)) %>%
  accuracy(sarima.gas.cov.forecast$mean)
```

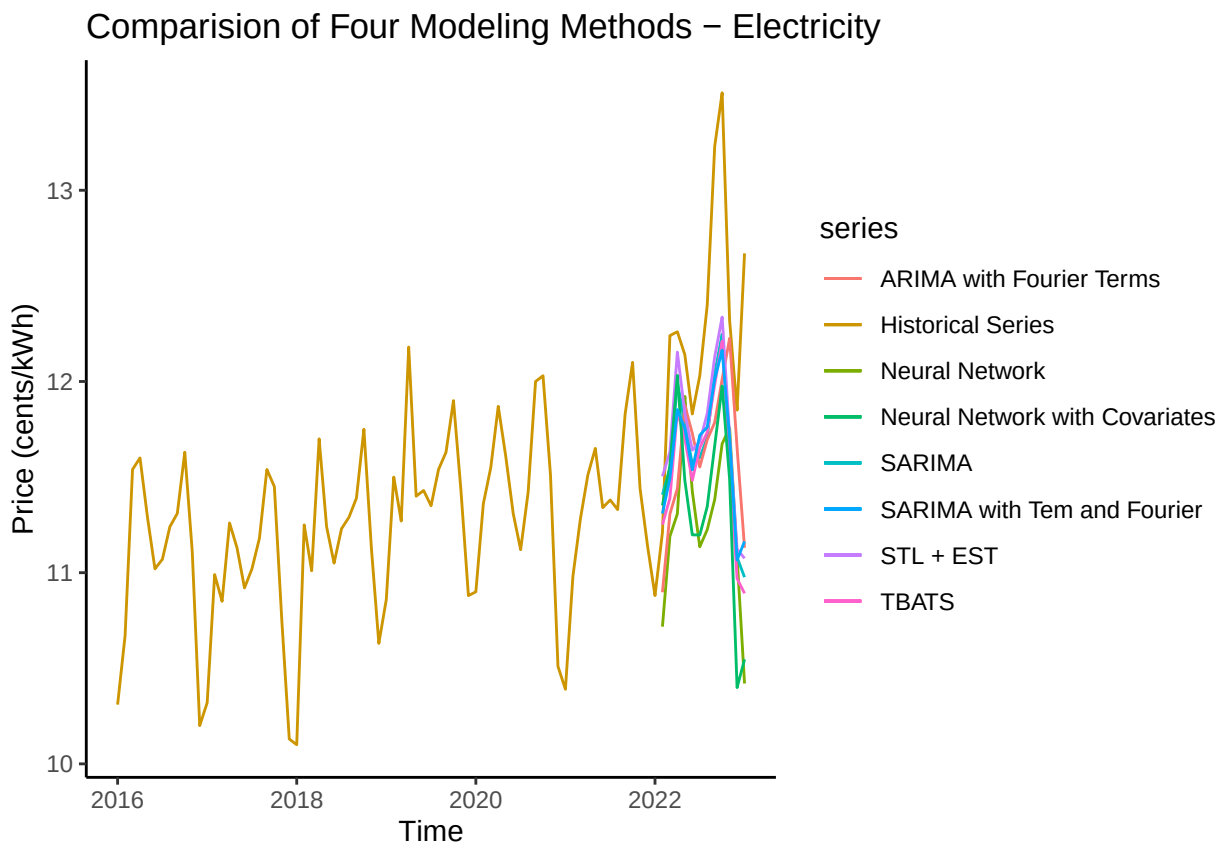
##Summary and Conclusions

Use ETS to model – not finished – I don't know how to do this, maybe we don't need to include this

```
# I think STL doesn't need fourier terms. So, I will get covariates dataframes without fourier terms
# because UKRWAR and temperature are the same across two data frames, we only need one
cov_train_nofour <- covariates_train_e[, 1:2]
colnames(cov_train_nofour) <- c("UKRWAR", "temperature")

cov_test_nofour <- covariates_full_e[, 1:2] %>%
  window(start = c(2022, 2))
colnames(cov_test_nofour) <- c("UKRWAR", "temperature")
```

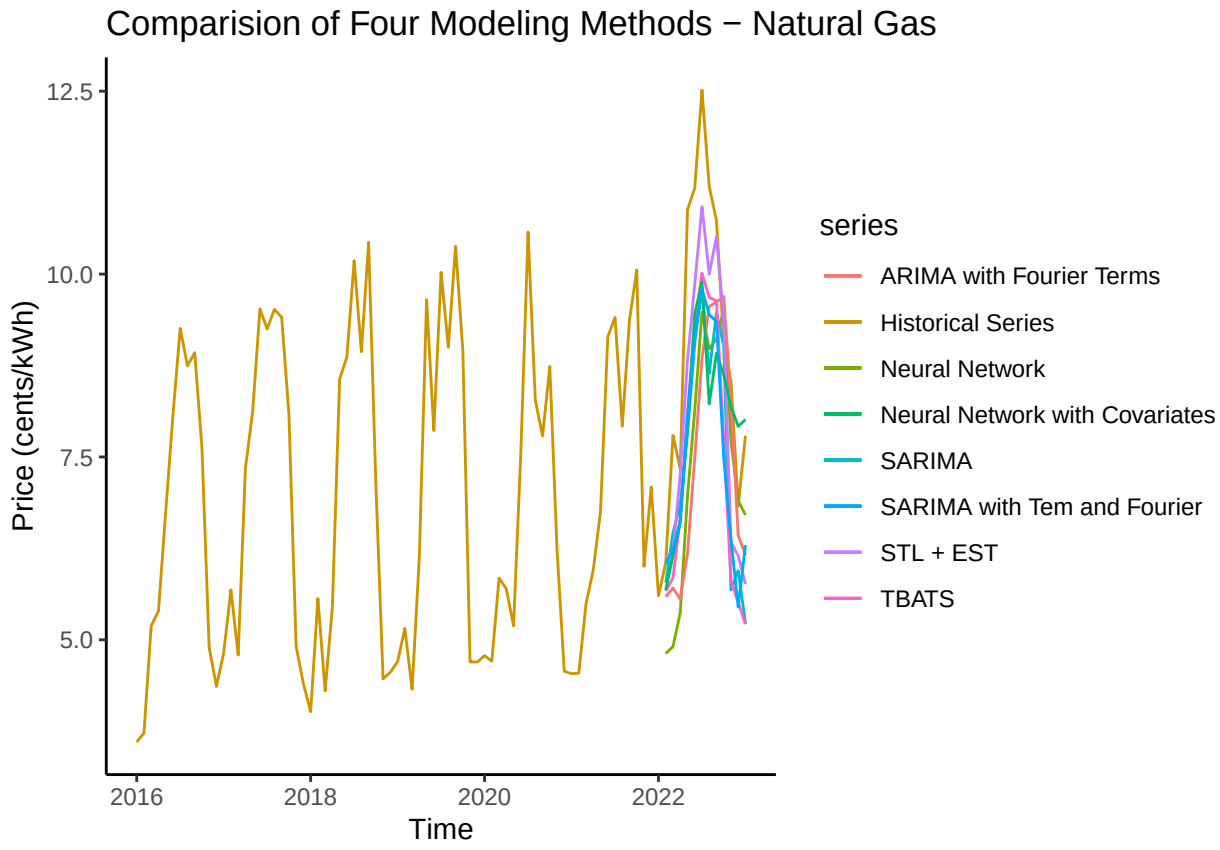
compare performance scores and generate tables for use

```
# plot these together
ts_electricity %>%
  window(start = c(2016, 1)) %>%
  autoplot(series = "Historical Series") +
  autolayer(nn.e.forecast$mean, series = "Neural Network") +
  autolayer(arima.e.forecast$mean, series = "SARIMA") +
  autolayer(arima.e.four.forecast$mean, series = "ARIMA with Fourier Terms") +
  autolayer(stl.e.forecast$mean, series = "STL + EST") +
  autolayer(tbats.e.forecast$mean, series = "TBATS") +
  autolayer(nn.cov.e.forecast$mean, series = "Neural Network with Covariates") +
  autolayer(sarima.e.cov.forecast$mean, series = "SARIMA with Tem and Fourier") +
  ylab("Price (cents/kWh)") +
  ggtitle("Comparision of Four Modeling Methods - Electricity") +
  theme_classic()
```




```
ts_gas_equiv %>%
  window(start = c(2016, 1)) %>%
  autoplot(series = "Historical Series") +
  autolayer(nn.gas.forecast$mean, series = "Neural Network") +
  autolayer(arima.gas.forecast$mean, series = "SARIMA") +
  autolayer(arima.gas.four.forecast$mean, series = "ARIMA with Fourier Terms") +
  autolayer(stl.gas.forecast$mean, series = "STL + EST") +
```

```
  autolayer(tbats.gas.forecast$mean, series = "TBATS") +
  autolayer(nn.cov.gas.forecast$mean,series = "Neural Network with Covariates") +
  autolayer(sarima.gas.cov.forecast$mean, series = "SARIMA with Tem and Fourier") +
  ylab("Price (cents/kWh)") +
  ggtitle("Comparision of Four Modeling Methods - Natural Gas") +
  theme_classic()
```




```
# scores for electricity
scores.e <- rbind(sarima_e_perf,
                  arima_four_e_perf,
                  stl_e_perf,
                  nn_e_perf,
                  tbats_e_perf,
                  nn_cov_e_perf,
                  sarima_cov_e_perf
                  ) %>%
  as.data.frame()

  # rename rows
row.names(scores.e) <- c("SARIMA",
                         "ARIMA with Fourier",
                         "STL",
                         "Neural Network",
                         "TBATS",
                         "Neural Network with Covariates",
```

```
                          "SARIMA with Tem and Fourier")

  # find the row index of the lowest RMSE
best.e.model <- scores.e$RMSE %>%
  which.min()

cat("The best model for electricity by RMSE is: ",
    row.names(scores.e[best.e.model, ]))
```

```
## The best model for electricity by RMSE is:  STL
```

```
# generate a visualized table to use in the report
kbl(scores.e,
    caption = "Forecast Accuracy for NC Residential Electricity Price",
    digits = array(5, ncol(scores.e))) %>%
  kable_styling(full_width = FALSE, position = "center",
                latex_options = "hold_position") %>%
  kable_styling(latex_options = "striped",
                stripe_index = best.e.model,
                stripe_color = "red")
```

Table 1: Forecast Accuracy for NC Residential Electricity Price

	ME	RMSE	MAE	MPE	MAPE	ACF1	Theil's U
SARIMA	-0.67702	0.83764	0.70998	-5.83726	6.12612	0.25988	2.73983
ARIMA with Fourier	-0.69901	0.87542	0.69901	-6.04750	6.04750	0.08222	2.82856
STL	-0.58551	0.76708	0.63447	-5.01390	5.43949	0.25835	2.36056
Neural Network	-1.03725	1.20383	1.03725	-9.29157	9.29157	0.18518	2.99352
TBATS	-0.74971	0.89063	0.75658	-6.50522	6.56626	0.23462	2.86546
Neural Network with Covariates	-0.95409	1.12846	0.97773	-8.51591	8.72410	0.39021	2.67299
SARIMA with Tem and Fourier	-0.67930	0.81795	0.69585	-5.83257	5.97890	0.26220	2.92127

```
# scores for gas
scores.gas <- rbind(sarima_gas_perf,
                    arima_four_gas_perf,
                    stl_gas_perf,
                    nn_gas_perf,
                    tbats_gas_perf,
                    nn_cov_gas_perf,
                    sarima_cov_gas_perf) %>%
  as.data.frame()

  # rename rows
row.names(scores.gas) <- c("SARIMA",
                           "ARIMA with Fourier",
                           "STL",
                           "Neural Network",
                           "TBATS",
                           "Neural Network with Covariates",
                           "SARIMA with Tem and Fourier")
```

```
  # find the row index of the lowest RMSE
best.gas.model <- scores.gas$RMSE %>%
  which.min()

cat("The best model for natural gas by RMSE is: ",
    row.names(scores.gas[best.gas.model, ]))
```

```
## The best model for natural gas by RMSE is:  STL
```

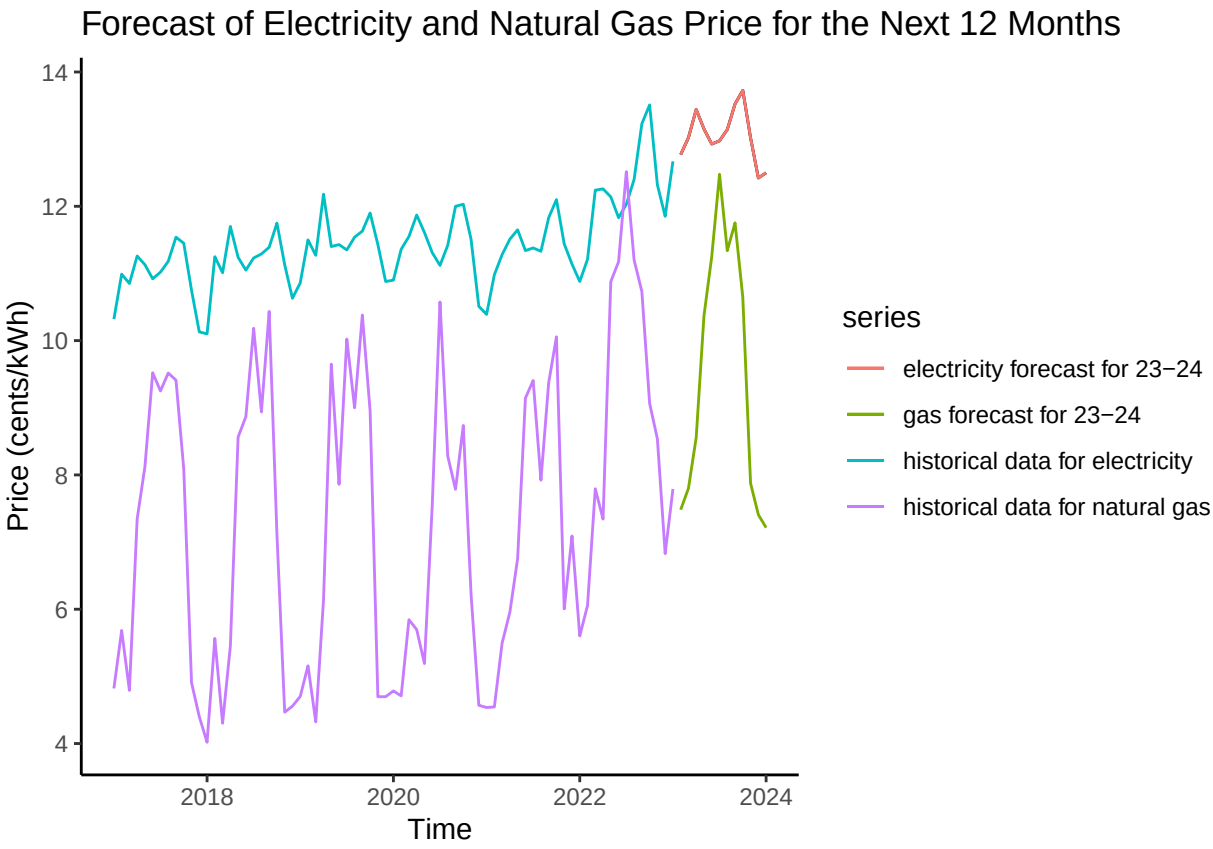
```
# generate a visualized table to use in the report
kbl(scores.gas,
    caption = "Forecast Accuracy for NC Residential Natural Gas Price",
    digits = array(5, ncol(scores.gas))) %>%
  kable_styling(full_width = FALSE, position = "center",
                latex_options = "hold_position") %>%
  kable_styling(latex_options = "striped",
                stripe_index = best.gas.model,
                stripe_color = "red")
```

Table 2: Forecast Accuracy for NC Residential Natural Gas Price

	ME	RMSE	MAE	MPE	MAPE	ACF1	Theil's U
SARIMA	-1.67902	1.97784	1.67902	-23.35174	23.35174	-0.15535	1.81675
ARIMA with Fourier	-1.73517	2.31199	1.83980	-25.17030	26.24979	0.57850	3.12540
STL	-1.04788	1.35599	1.11314	-14.03428	14.80904	-0.33348	1.28849
Neural Network	-1.77695	2.18274	1.86294	-25.88562	26.83971	0.53727	2.70687
TBATS	-1.72858	1.89245	1.72858	-24.53539	24.53539	-0.32231	2.08610
Neural Network with Covariates	-1.19246	1.72241	1.41105	-14.63497	17.39024	0.50070	2.10039
SARIMA with Tem and Fourier	-1.63275	1.79495	1.63275	-21.52013	21.52013	-0.01071	1.95239

Use STL to model electricity and natural gas for the next 12 month

```
e.forecast.2324 <- ts_electricity_war[, 2] %>%
  stlf(h = 12)

gas.forecast.2324 <- ts_gas_equiv_war[, 1] %>%
  stlf(h = 12)

e.forecast.2324$mean %>%
  autoplot() +
  autolayer(e.forecast.2324$mean, series = "electricity forecast for 23-24") +
  autolayer(gas.forecast.2324$mean, series = "gas forecast for 23-24") +
  autolayer(window(ts_electricity_war[, 2], start = c(2017, 1)),
            series = "historical data for electricity") +
   autolayer(window(ts_gas_equiv_war[, 1], start = c(2017, 1)),
            series = "historical data for natural gas") +
  theme_classic() +
  ggtitle("Forecast of Electricity and Natural Gas Price for the Next 12 Months") +
  ylab("Price (cents/kWh)")
```


Forecast of Electricity and Natural Gas Price for the Next 12 Months
Price (cents/kWh)
14
12
10
8
6
4
2018
2020
2022
2024
Time
series
electricity forecast for 23−24
gas forecast for 23−24
historical data for electricity
historical data for natural gas