Visionary Final

Justin DePue, John Rooney, and Tony Jiang

2023-04-10

library packages

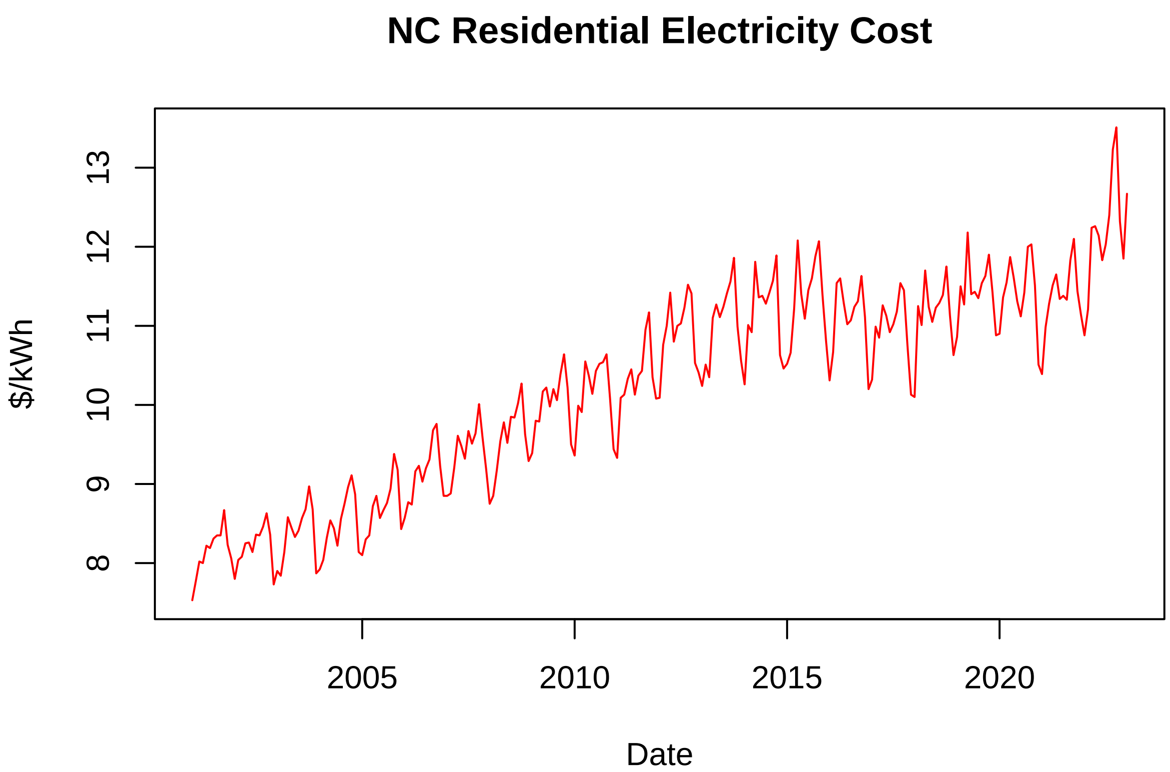
```
library(tseries)
library(lubridate)
library(here)
library(tidyverse)
library(outliers)
library(ggplot2)
library(forecast)
library(kableExtra)
```

load the data

```
# automatically set the working directory as personal local path to R project
# so the file path in read.csv can work for everyone
setwd(here())

# load the data
electricity_prices.df <- read.csv("./Data/Average_retail_price_of_electricity.csv", header = TRUE, skip=

nc_electricity.df<-electricity_prices.df[228,4:(ncol(electricity_prices.df))] %>%
  pivot_longer(cols=everything(), names_to = 'my_date', values_to = 'price_per_kWh')

# examine whether there is missing value or not
summary(nc_electricity.df)
```

```
##    my_date          price_per_kWh
##  Length:265         Length:265
##  Class :character   Class :character
##  Mode  :character   Mode  :character
```

```
# there is no missing value

nc_electricity.df$price_per_kWh<-as.numeric(nc_electricity.df$price_per_kWh)

ts_electricity<-ts(rev(nc_electricity.df[,2]), start=c(2001,1), frequency=12)

plot(ts_electricity, col="red", ylab="$/kWh", xlab="Date", main="NC Residential Electricity Cost")
```



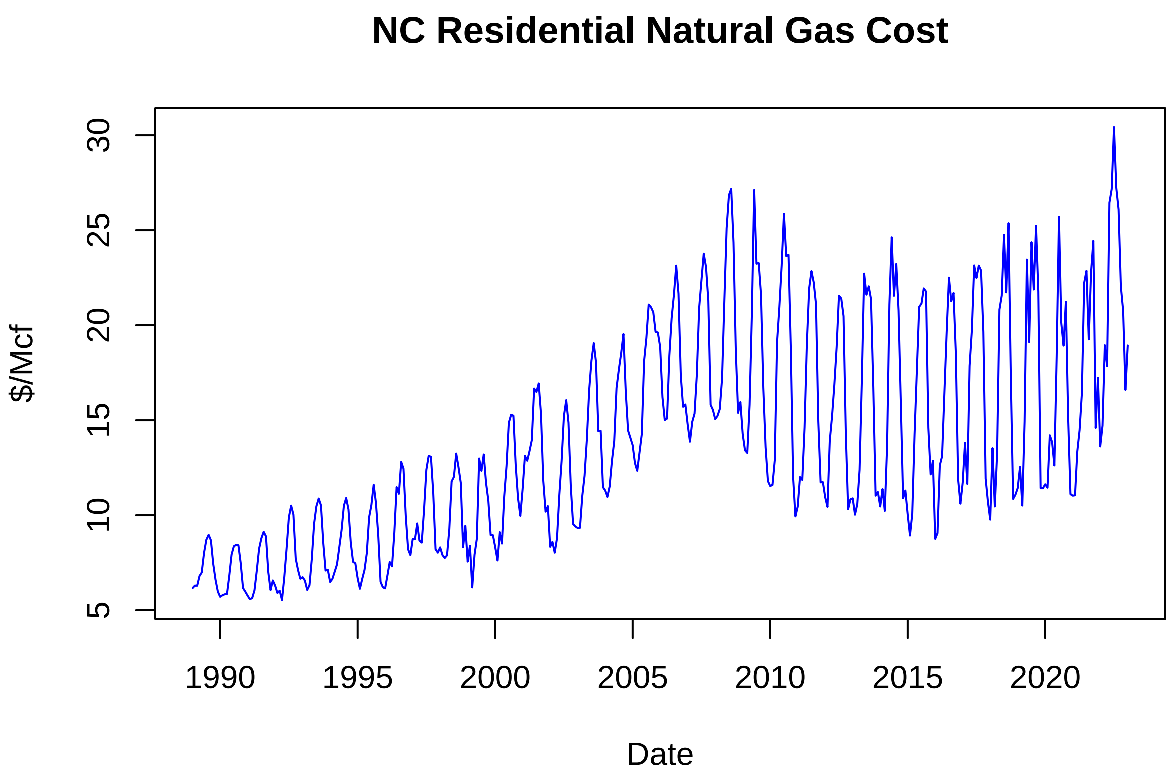

```
# Import Natural Gas data

natural_gas.df <- read.csv("./Data/NC_NaturalGas.csv", header = TRUE, skip=2,col.names = c("year", "pri

#Check for missing data. There is an extra row with an NA at the end of the data, so I removed it with 

summary(na.omit(natural_gas.df))
```

```
##      year               price      
##  Length:409         Min.   : 5.54  
##  Class :character   1st Qu.: 9.07  
##  Mode  :character   Median :12.54  
##                     Mean   :13.78  
##                     3rd Qu.:18.11  
##                     Max.   :30.43
```

```
str(natural_gas.df)
```

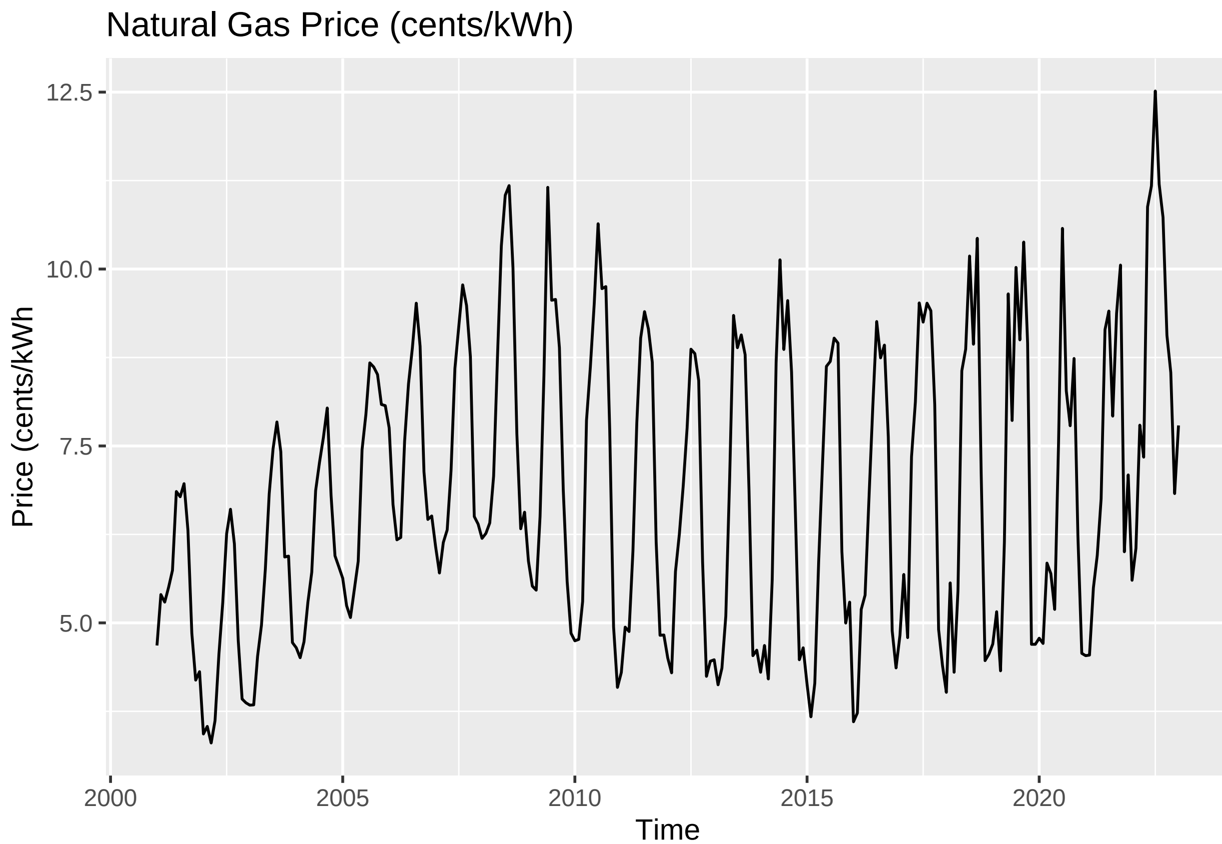
```
## 'data.frame':    410 obs. of  2 variables:
##  $ year : chr  "Jan-1989" "Feb-1989" "Mar-1989" "Apr-1989" ...
##  $ price: num  6.17 6.3 6.29 6.8 6.99 8.02 8.71 8.97 8.68 7.44 ...
```

```
#create timeseries for natural gas

ts_NG<-ts(na.omit(natural_gas.df[,2]), start=c(1989,1), frequency=12)

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

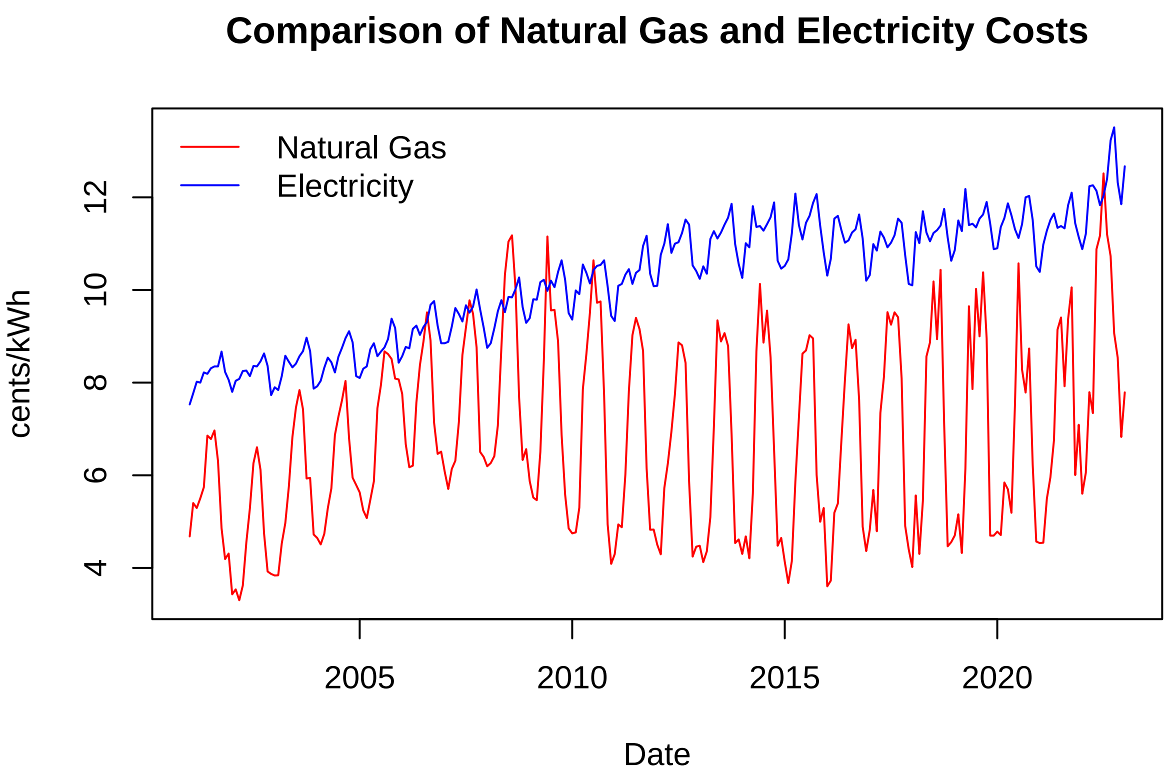


standardize the unit of natural gas data to make it comparable with electricity

```
# convert natural gas data from $/Mcf to $/kWh based on 80% heating efficiency
# $/kWh = [($/mcf/1.037)/293.07107]/0.9 = $/mcf/273.52


conversion <- 0.8*293.07107*1.037/100


natural_gas.df$kwh_equiv<-(natural_gas.df$price/conversion)


natural_gas.df<-na.omit(natural_gas.df)

ts_gas_equiv<-ts(natural_gas.df[,3], start=c(1989,1), frequency=12)

ts_gas_equiv<-window(ts_gas_equiv, start=c(2001, 1))

autoplot(ts_gas_equiv) +
  ylab("Price (cents/kWh") +
  ggtitle("Natural Gas Price (cents/kWh)")
```




```r
ts.plot(ts_gas_equiv, ts_electricity, gpars = list(col = c("red", "blue")),
        xlab="Date", ylab="cents/kWh",
        main="Comparison of Natural Gas and Electricity Costs")
legend("topleft", bty="n", lty=c(1,1), col=c("red","blue"),
       legend=c(" Natural Gas ", " Electricity "))
```

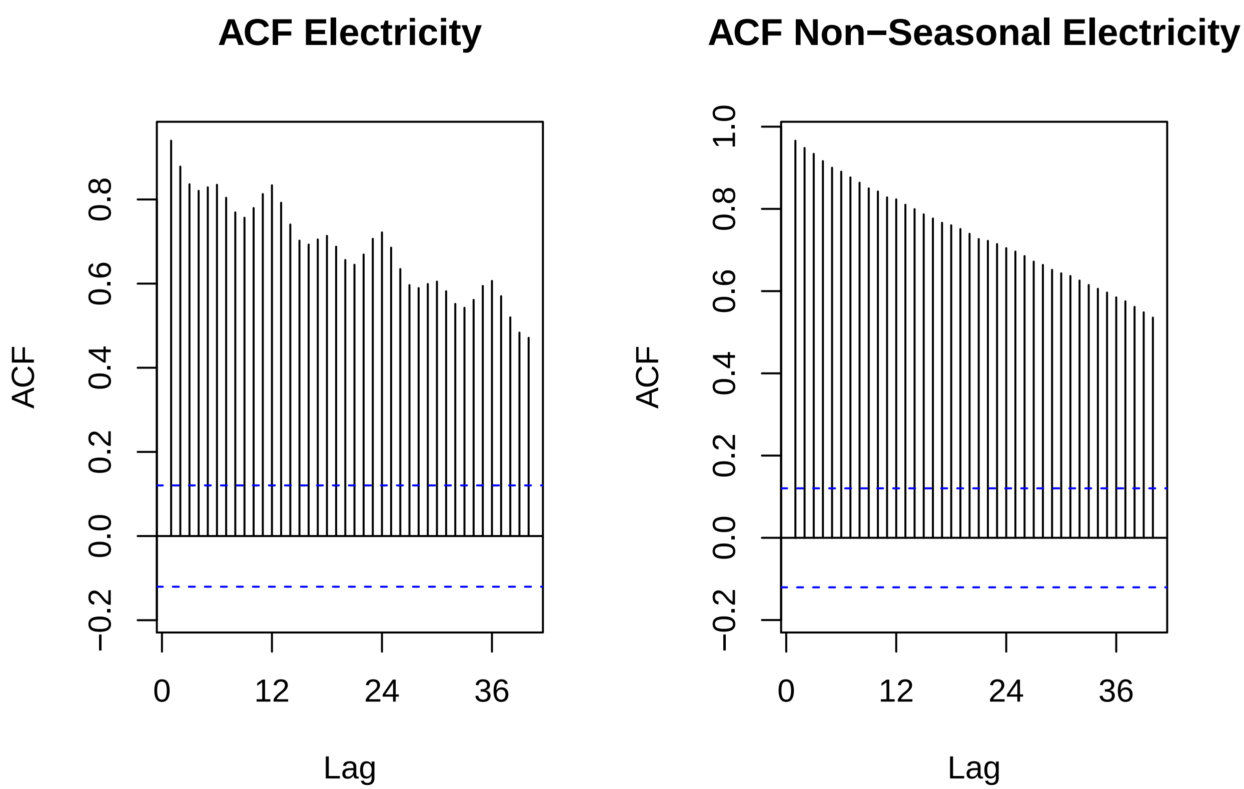



```
#Create Dataframe with both Gas & Electricity prices...
#create new date column
new_date<-seq(as.Date("2001/1/1"), by = "month", length.out = nrow(ts_electricity))

#bind date, gas, & electricity
nc_energy.df<-cbind.data.frame("Month_Date"=new_date,
                               "electricity"=nc_electricity.df$price_per_kWh,
                               "gas_equiv"=ts_gas_equiv)

#Subtract Gas price from Electricity price
nc_energy.df$cost_diff<-(nc_energy.df$electricity - nc_energy.df$gas_equiv)
```

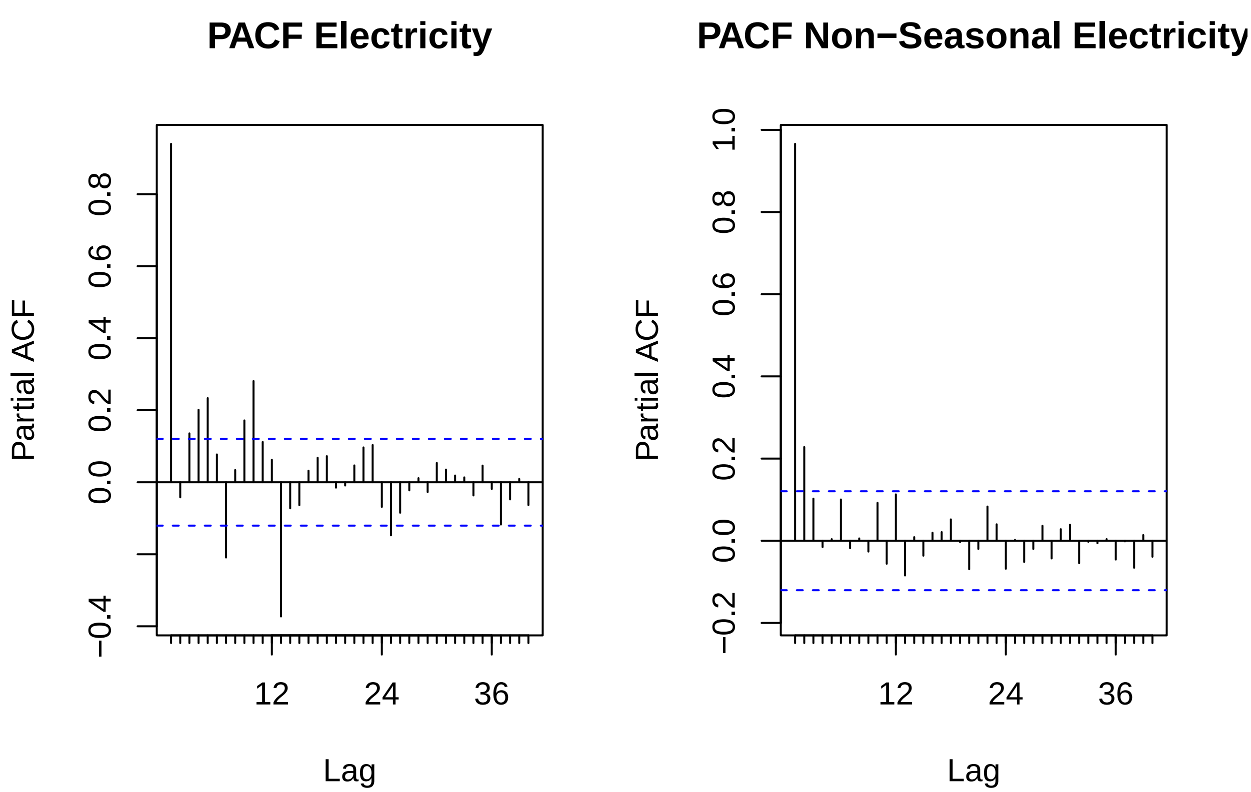
```
#Decompose TS

decomp_elec<-decompose(ts_electricity, type="additive")
decomp_gas<-decompose(ts_NG, type="multiplicative")

#remove seasonality
ts_elec_ns<-(ts_electricity-decomp_elec$seasonal)
ts_gas_ns<-(ts_NG-decomp_gas$seasonal)
```

plot ACF and PACF to see the general pattern of two series

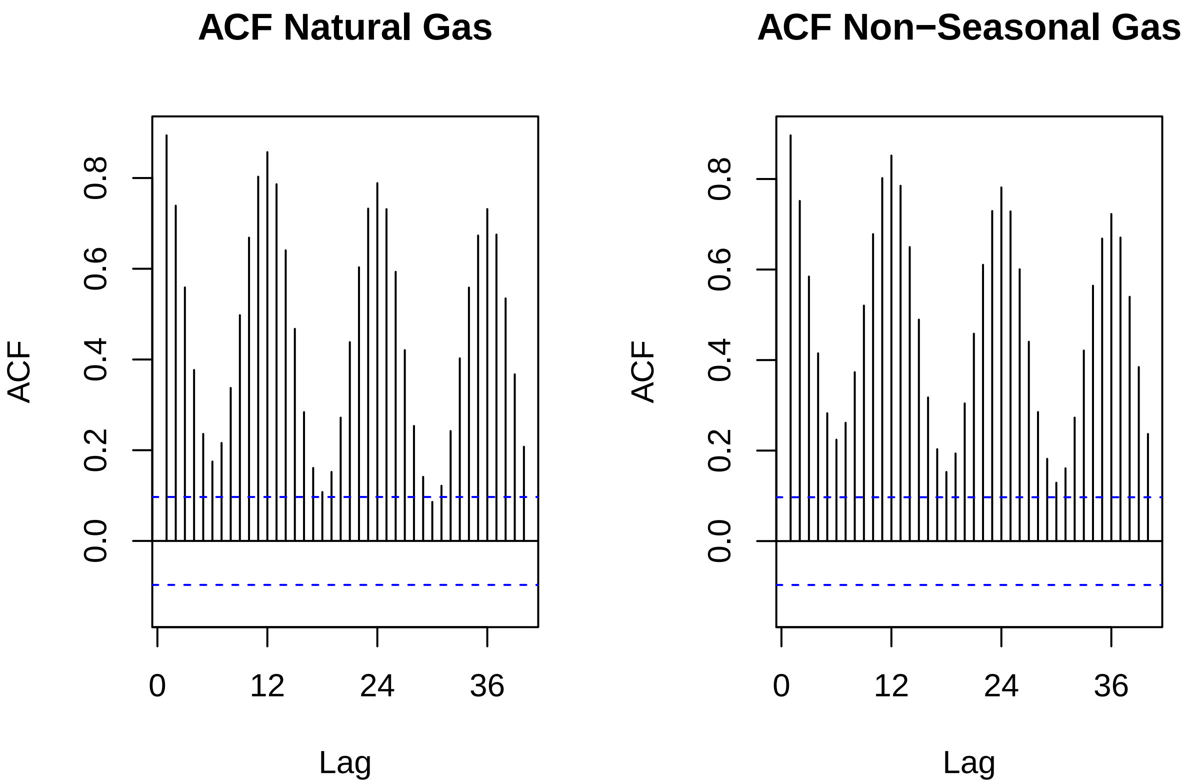
```
# ACF and PACF of electricity data
par(mfrow=c(1,2))
Acf(ts_electricity, lag.max=40, main="ACF Electricity")
Acf(ts_elec_ns, lag.max=40, main="ACF Non-Seasonal Electricity")
```



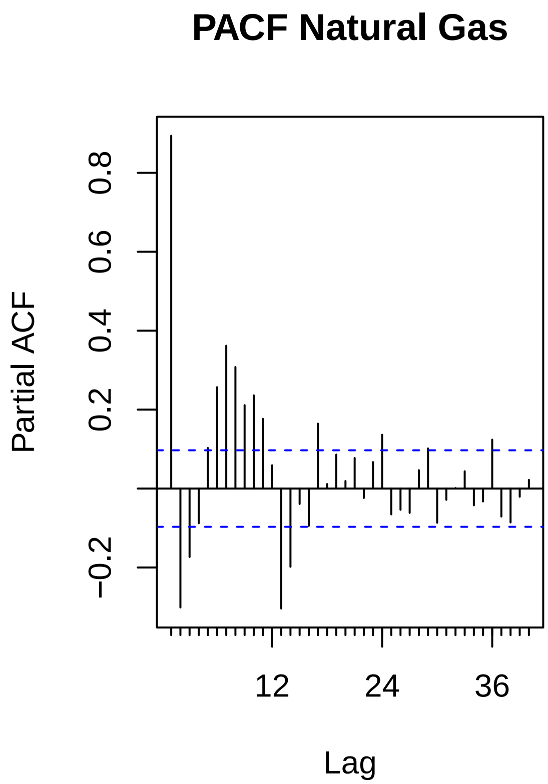

```
par(mfrow=c(1,2))
Pacf(ts_electricity, lag.max=40, main="PACF Electricity")
Pacf(ts_elec_ns, lag.max=40, main="PACF Non-Seasonal Electricity")
```

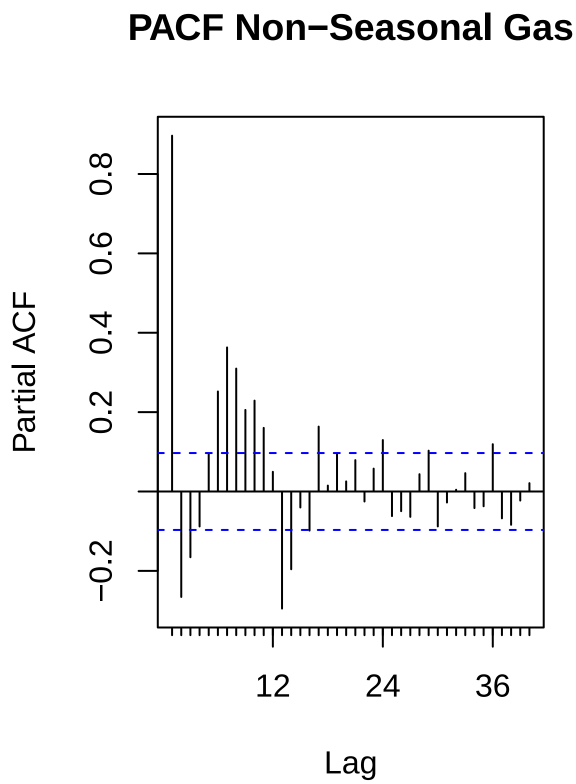


```r
# ACF and PACF of natural gas data
par(mfrow=c(1,2))
Acf(ts_NG, lag.max=40, main="ACF Natural Gas")
Acf(ts_gas_ns, lag.max=40, main="ACF Non-Seasonal Gas")
```



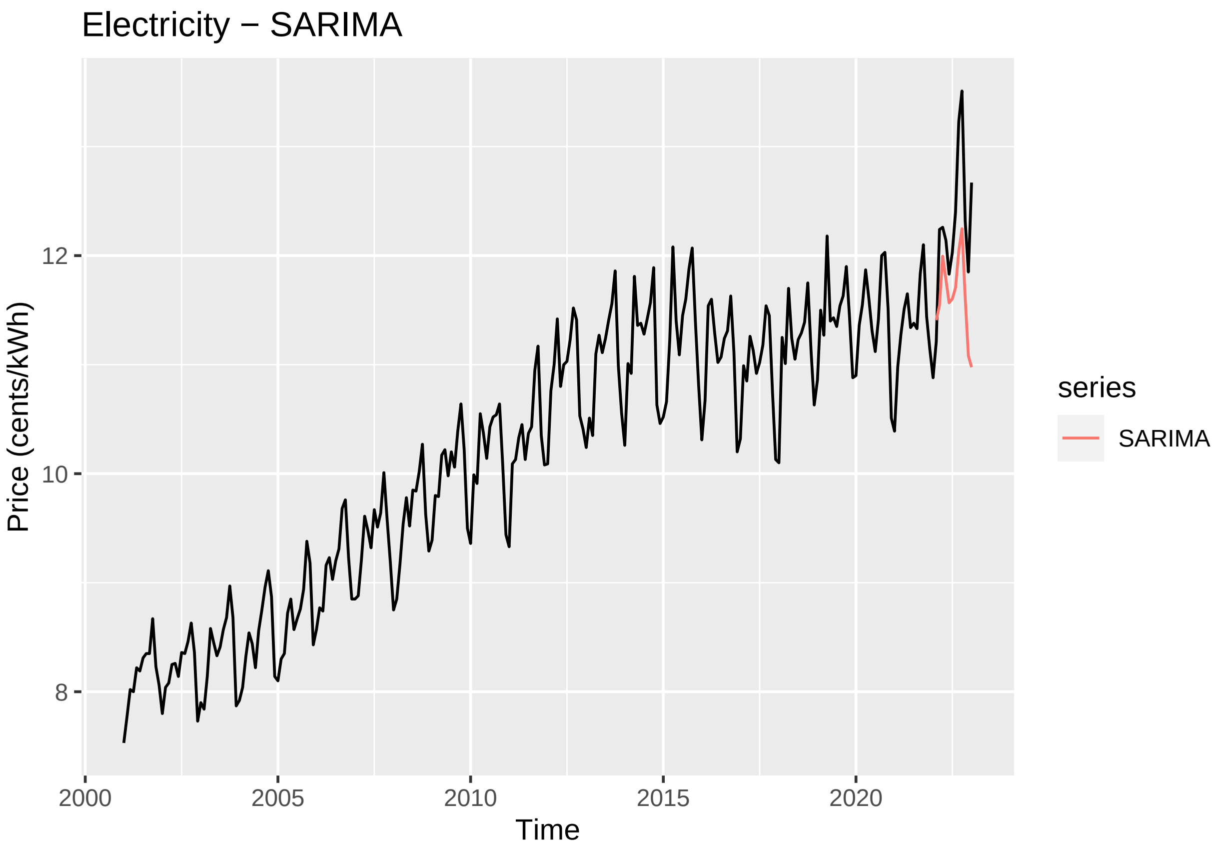

```r
par(mfrow=c(1,2))
Pacf(ts_NG, lag.max=40, main="PACF Natural Gas")
Pacf(ts_gas_ns, lag.max=40, main="PACF Non-Seasonal Gas")
```



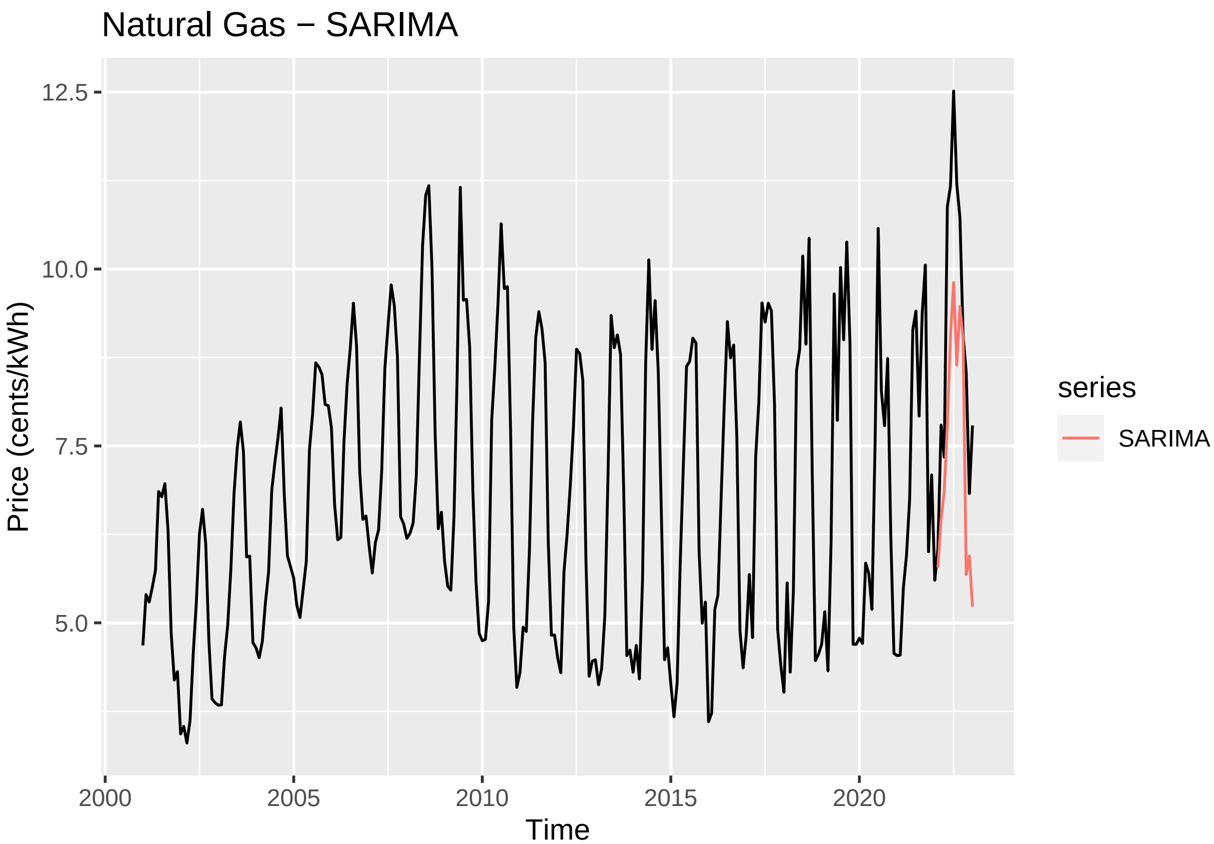


Use seasonal Arima to model eletricity and natural gas

```
#forecast Electricity SARIMA

arima.e.model<-auto.arima(window(ts_electricity, end=c(2022,1)))
arima.e.forecast<-forecast(arima.e.model, h=12)

autoplot(ts_electricity) +
  autolayer(arima.e.forecast$mean, series = "SARIMA") +
  ylab("Price (cents/kWh)") +
  ggtitle("Electricity - SARIMA")
```




```
  # performance is not that good

#Gas SARIMA forecast
arima.gas.model<-auto.arima(window(ts_gas_equiv, end=c(2022,1)))
arima.gas.forecast<-forecast(arima.gas.model, h=12)

autoplot(ts_gas_equiv)+
  autolayer(arima.gas.forecast$mean, series = "SARIMA") +
  ylab("Price (cents/kWh)") +
  ggtitle("Natural Gas - SARIMA")
```




```
# performance is not that good
```

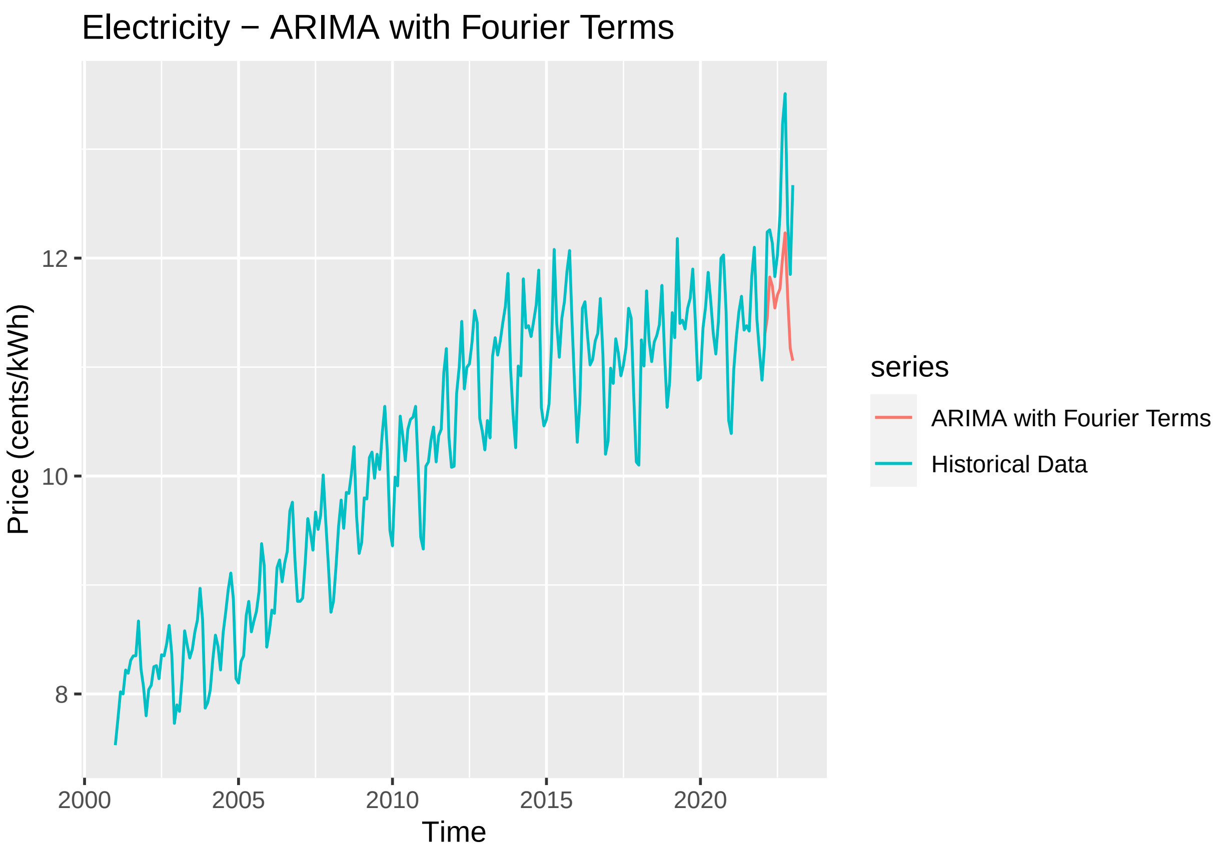
Examine seasonal Arima model's performance on electricity and NG data

```
# 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)
```

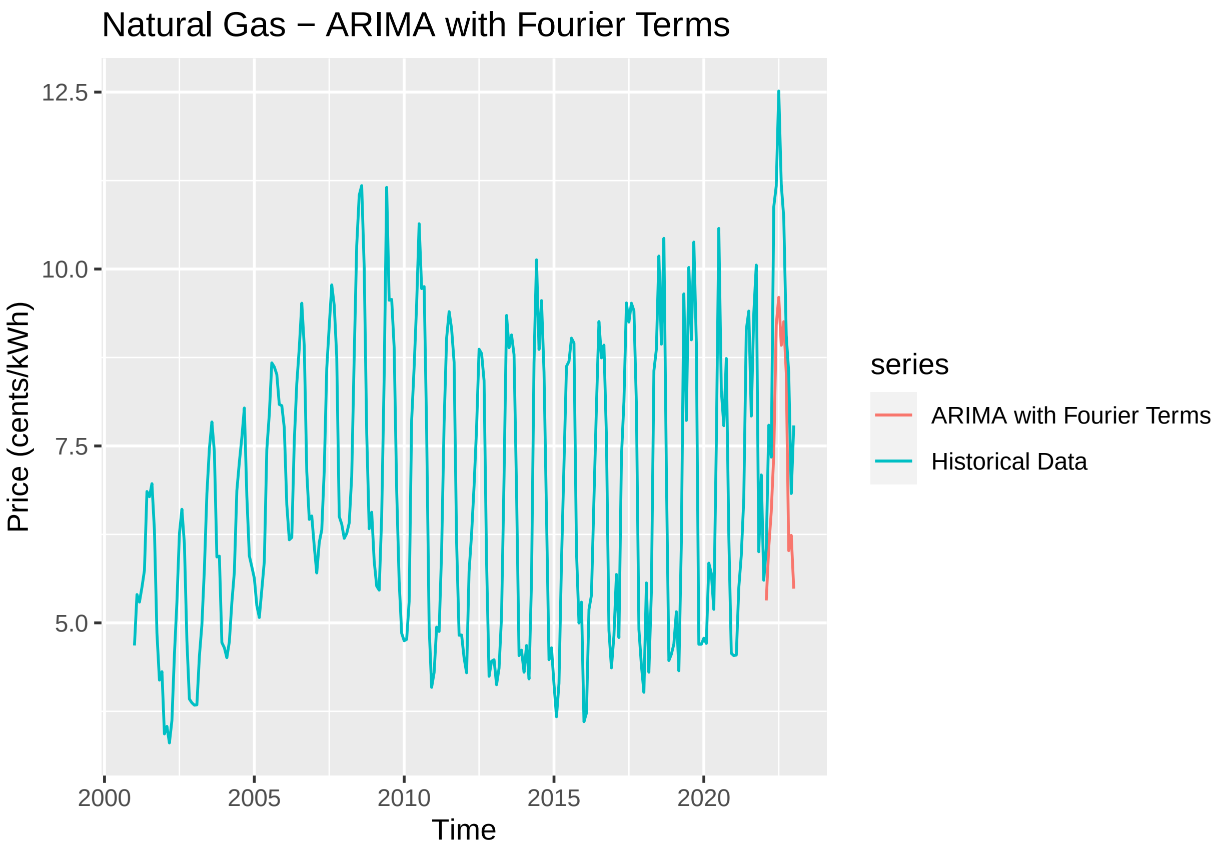
Use Arima with Fourier terms to model

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

```
                          K = 6,
                          h = 12),
           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(xreg = fourier(window(ts_gas_equiv, end = c(2022, 1)),
                            K = 6)
             ) %>%
  forecast(xreg = fourier(window(ts_gas_equiv,
                                 end = c(2022, 1)
                                 ),
                          K = 6,
                          h = 12),
           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)")
```

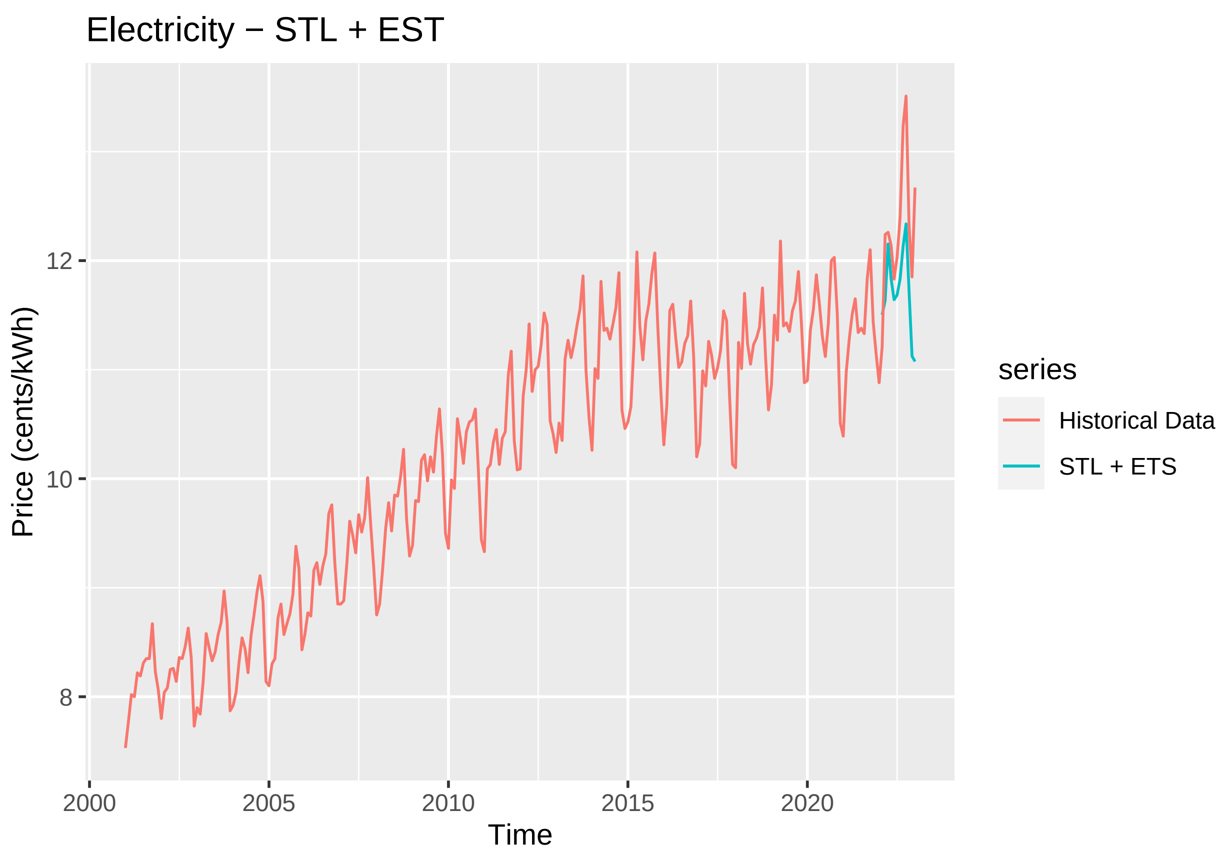


Examine Arima with fourier's performance on electricity and NG data

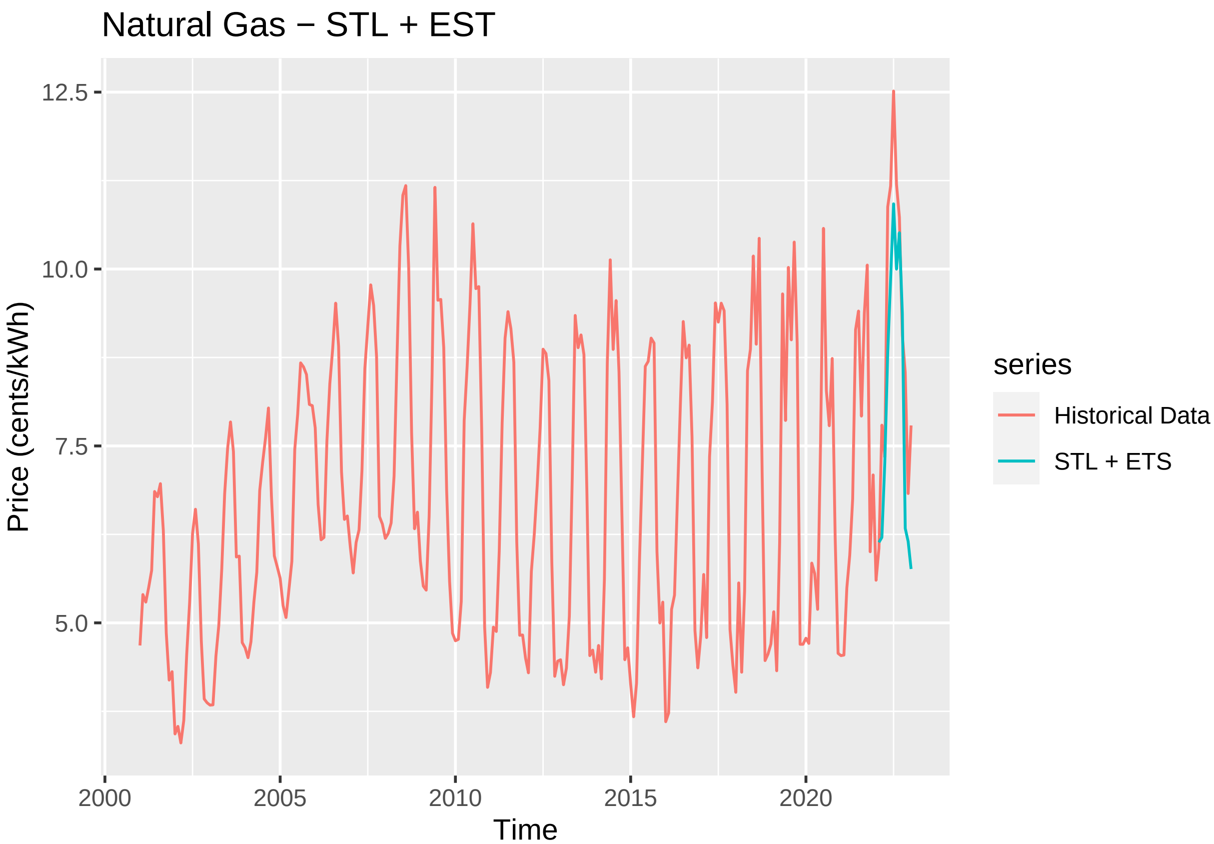
```
# model performance for electricity data
arima_four_e_perf <- ts_electricity %>%
  window(start = c(2022, 2)) %>%
  accuracy(arima.e.four.forecast$mean)

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

Use STL to model

```
# STL for eletricity
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)")
```




```r
# 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)")
```



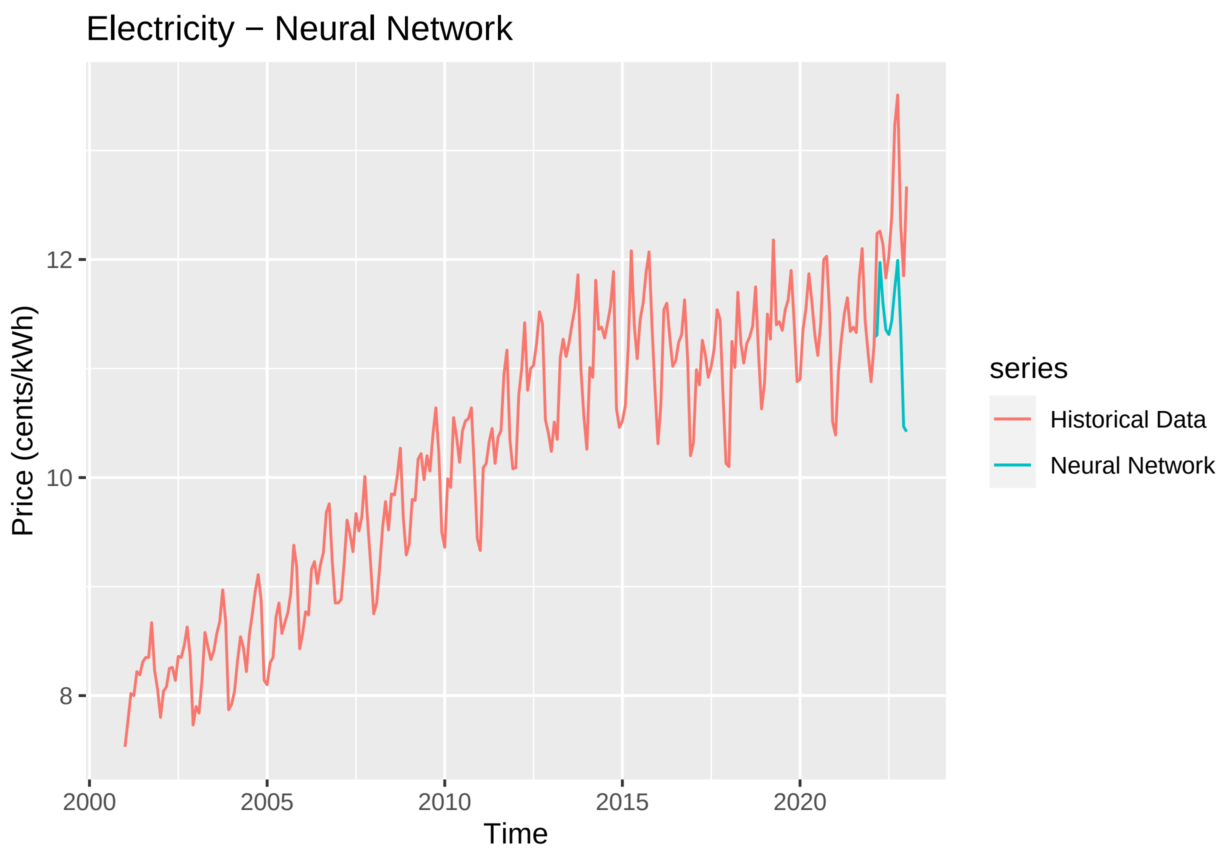
Examine STL's performance on electricity and NG data

```r
# 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)
```

Use Neural Network and fourier to model

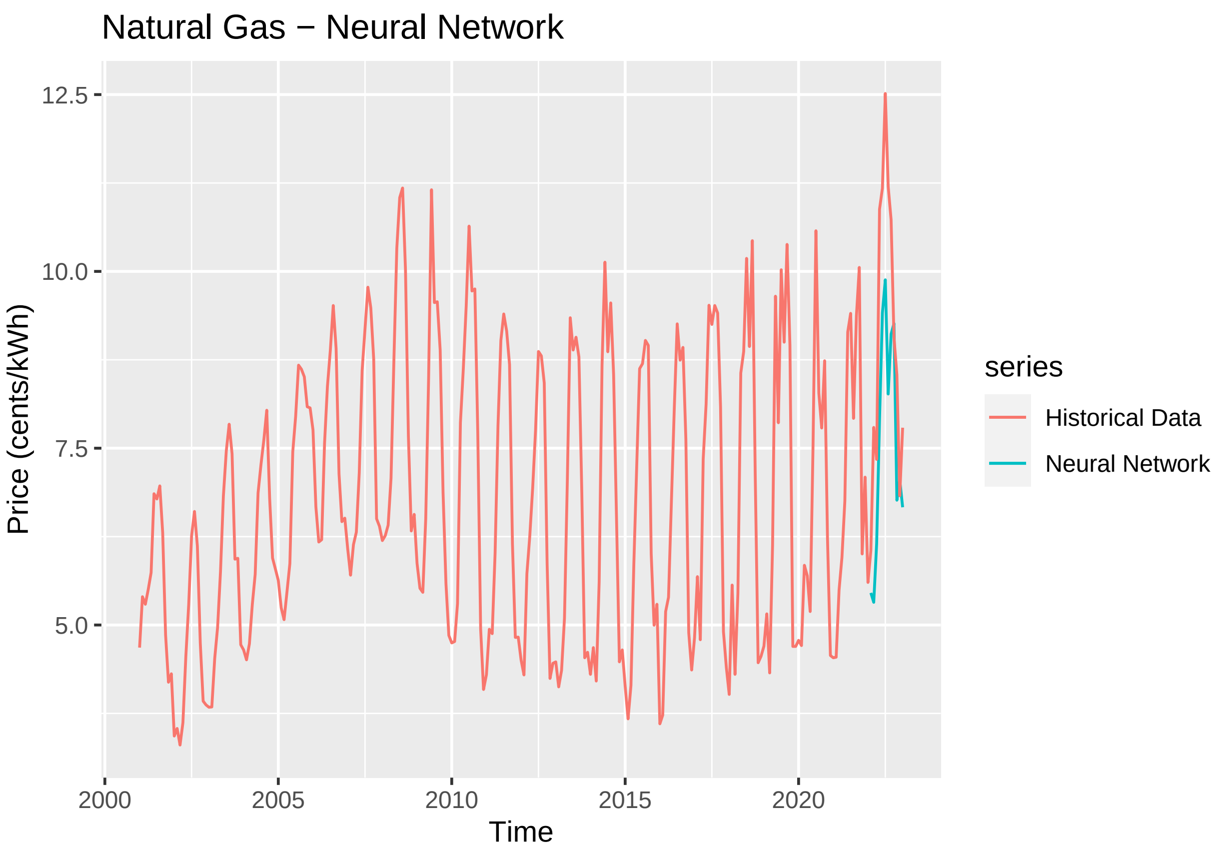
```r
# neural network forecast for electricity data
nn.e.forecast <- ts_electricity %>%
  window(end = c(2022 ,1)) %>%
  nnetar(p = 1, P = 1,
         xreg = fourier(window(ts_electricity,
                               end = c(2022, 1)
                               ),
                        K = 6)
         ) %>%
  forecast(xreg = fourier(window(ts_electricity,
                                 end = c(2022, 1)
                                 ),
                          K = 6, # a single K should be smaller than period/2
                          h = 12),
```

```
          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,
                                        end = c(2022, 1)
                                        ),
                          K = 6,
                          h = 12),
          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)")
```



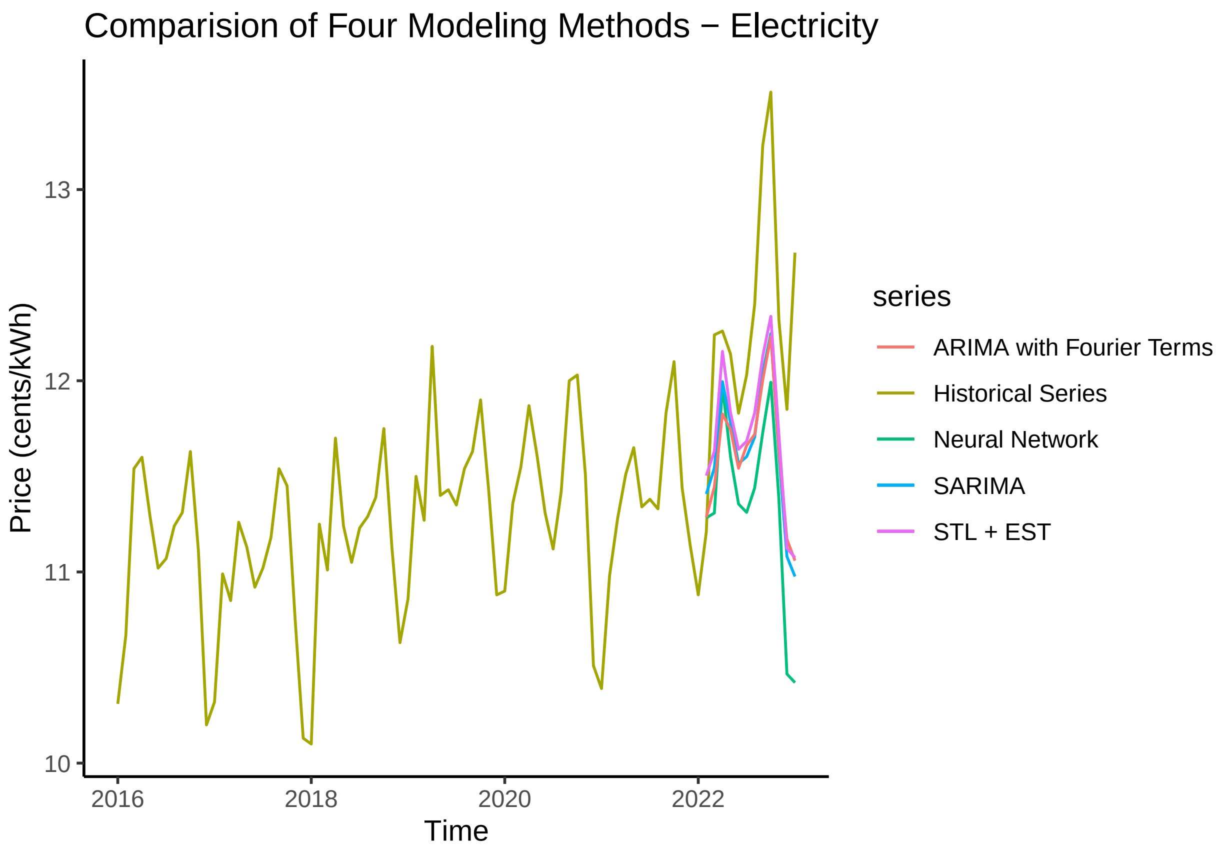
neutral network model performance

```
# 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)
```

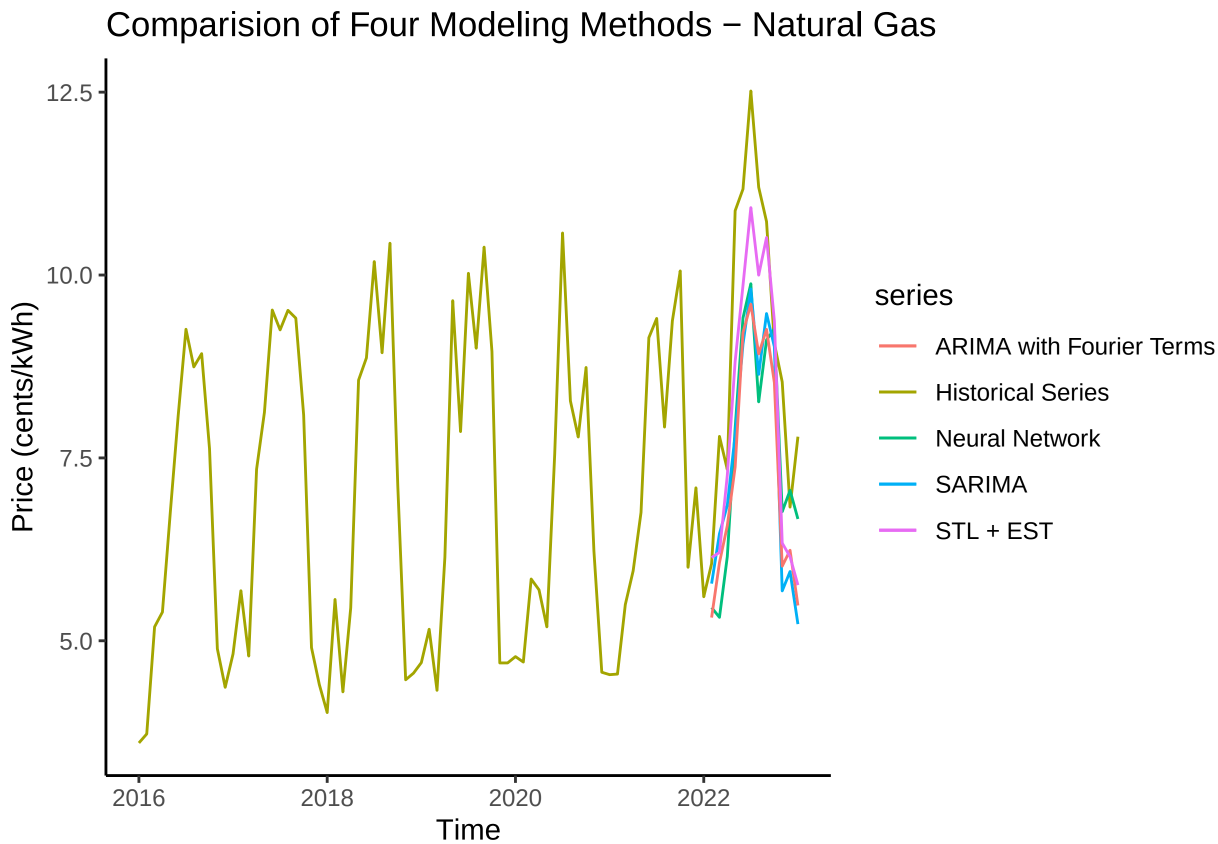
compare performance scores and generate tables for use

```
# 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") +
  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") +
  ylab("Price (cents/kWh)") +
  ggtitle("Comparision of Four Modeling Methods - Natural Gas") +
  theme_classic()
```




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

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

  # 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
```

```r
# 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	-0.69626	0.83237	0.70892	-5.99008	6.10231	0.20556	3.02159
STL	-0.58551	0.76708	0.63447	-5.01390	5.43949	0.25835	2.36056
Neural Network	-0.95201	1.12970	0.96419	-8.50215	8.61016	0.33335	2.84477

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

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

  # 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	-1.77629	2.00812	1.77629	-24.52203	24.52203	-0.26009	2.01593
STL	-1.04788	1.35599	1.11314	-14.03428	14.80904	-0.33348	1.28849
Neural Network	-1.55468	1.88210	1.62597	-20.89815	21.79634	0.01833	1.76125

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

```r
e.forecast.2324 <- ts_electricity %>%
  stlf(h = 12)

gas.forecast.2324 <- ts_gas_equiv %>%
  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, start = c(2017, 1)),
            series = "historical data for electricity") +
   autolayer(window(ts_gas_equiv, 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)")
```