DSC 551 Final Project

“Ditching Coal: How the United States Is Moving Away from One Fossil Fuel”

Student Name: Mukila Rajasekar

```
library('fpp2')
```

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

```
## -- Attaching packages --------------------------------------------- fpp2 2.4 --
```

```
## v ggplot2   3.3.2     v fma       2.4
## v forecast  8.13      v expsmooth 2.3
```

```
##
```

```
library('readxl')
library('seasonal')
library('tseries')
```

Loading and Cleaning the dataset

The dataset is taken from the following source, https://www.eia.gov/totalenergy/data/monthly/ (https://www.eia.gov/totalenergy/data/monthly/) The dataset shows the U.S. National Monthly Net Electricity Generation from Coal in Million kWh (Jan 1995 - Jun 2020)

```
df <- read_excel("C:\\Users\\mr4060\\Desktop\\DSC551-Spatial Temporal Analysis\\Finals\\Electric
ity_Net_Generation__Total_(All_Sectors).xlsx")
head(df)
```

Month <S3: POSIXct>	Netelectricitygeneration_fromcoal <dbl>
1995-01-01	147220.5
1995-02-01	132900.2
1995-03-01	131429.8

Month <S3: POSIXct>	Netelectricitygeneration_fromcoal <dbl>
1995-04-01	123075.9
1995-05-01	130418.1
1995-06-01	142845.6

6 rows

Let's convert the 'Netelectricitygeneration_fromcoal' column into time series for our analysis.

```
coal <- ts(data = df$Netelectricitygeneration_fromcoal, start=c(1995,1),frequency=12)
tail(coal) #Time series dataset
```

```
##           Jan      Feb      Mar      Apr      May      Jun
## 2020 65170.39 56071.59 50585.69 40575.70 46488.02 65475.19
```

Exploring the time series

```
#Structure of the time series
str(coal)
```

```
##  Time-Series [1:306] from 1995 to 2020: 147220 132900 131430 123076 130418 ...
```

```
paste('Frequency:', frequency(coal))
```

```
## [1] "Frequency: 12"
```

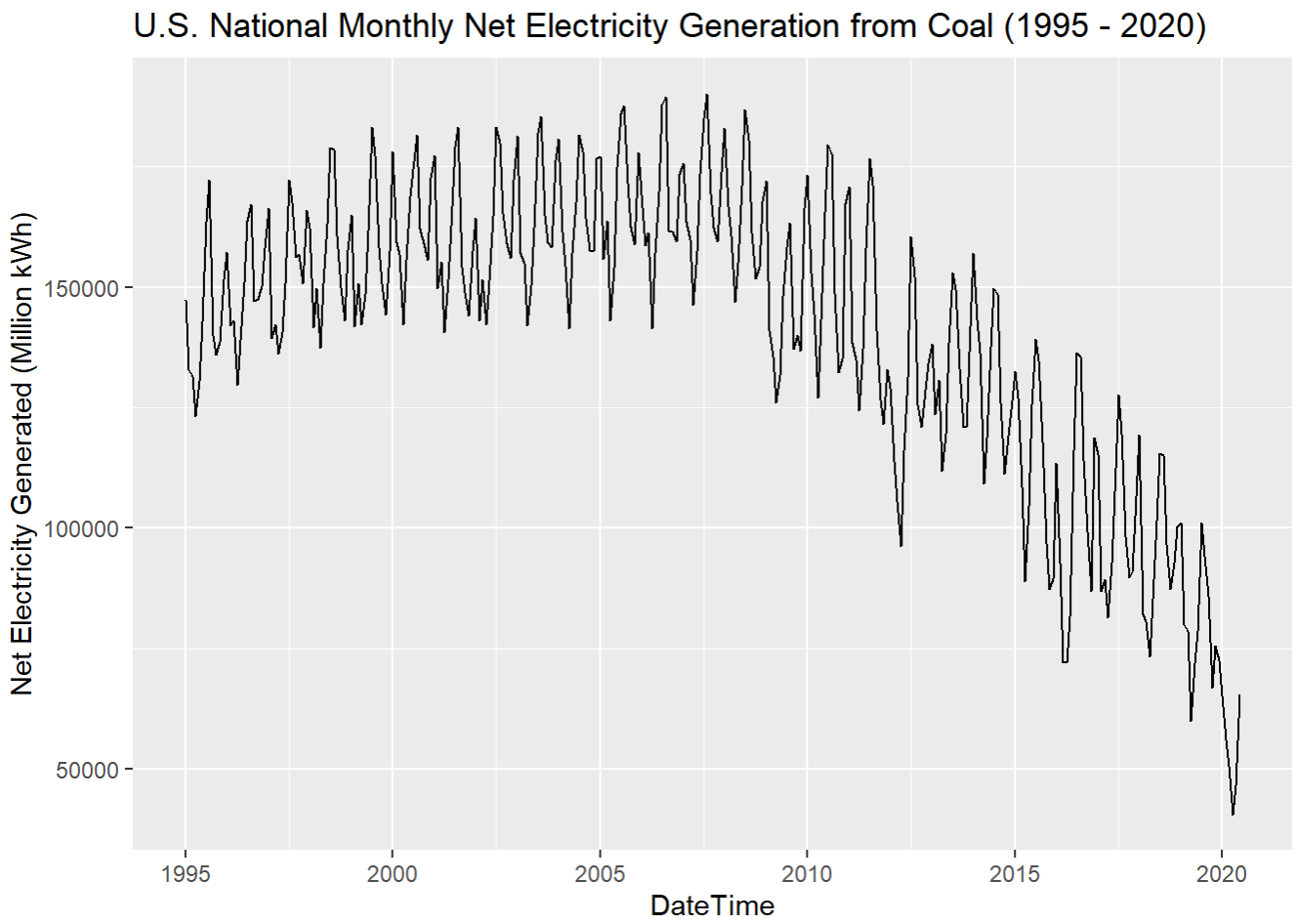
```
paste('Is it timeseries?',is.ts(coal))
```

```
## [1] "Is it timeseries? TRUE"
```

```
summary(coal)
```

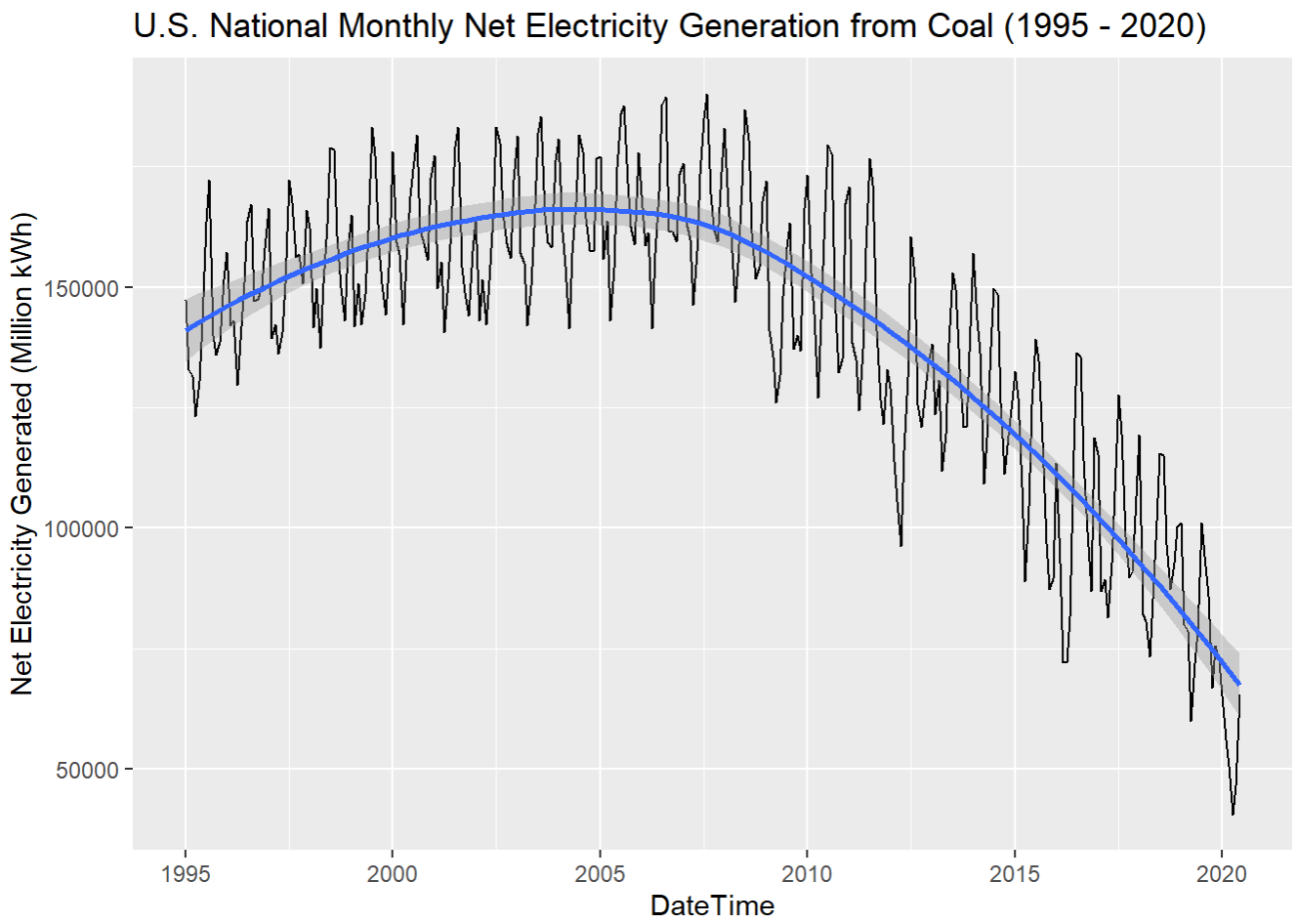
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   40576  125610  148494  140902  162399  190135
```

```
#Basic Plotting
autoplot(coal)+
  xlab("DateTime") + ylab("Net Electricity Generated (Million kWh)") +
  ggtitle("U.S. National Monthly Net Electricity Generation from Coal (1995 - 2020)")
```



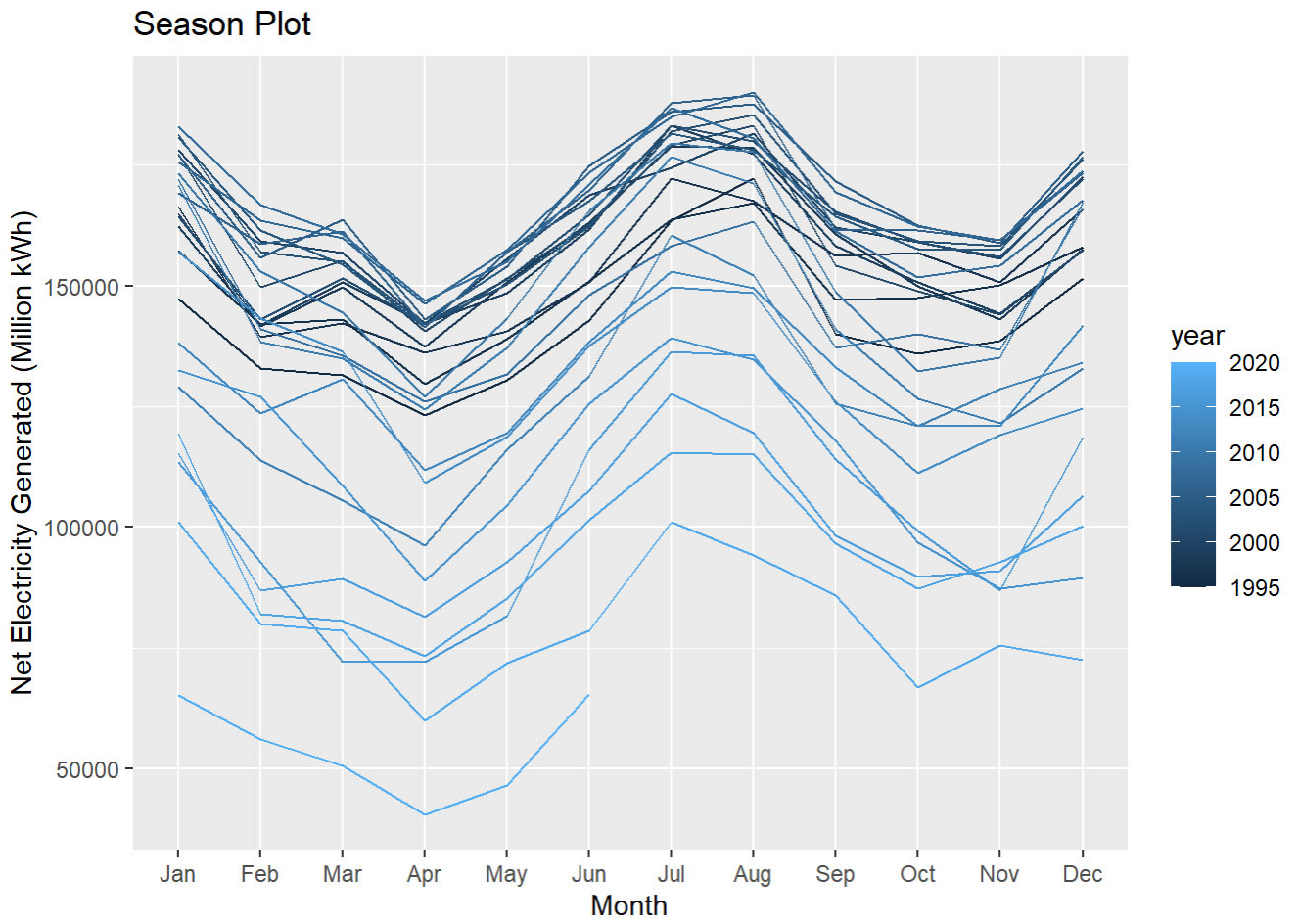

```
#Basic Plotting with smooth line
autoplot(coal)+geom_smooth() +
  xlab("DateTime") + ylab("Net Electricity Generated (Million kWh)") +
  ggtitle("U.S. National Monthly Net Electricity Generation from Coal (1995 - 2020)")
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



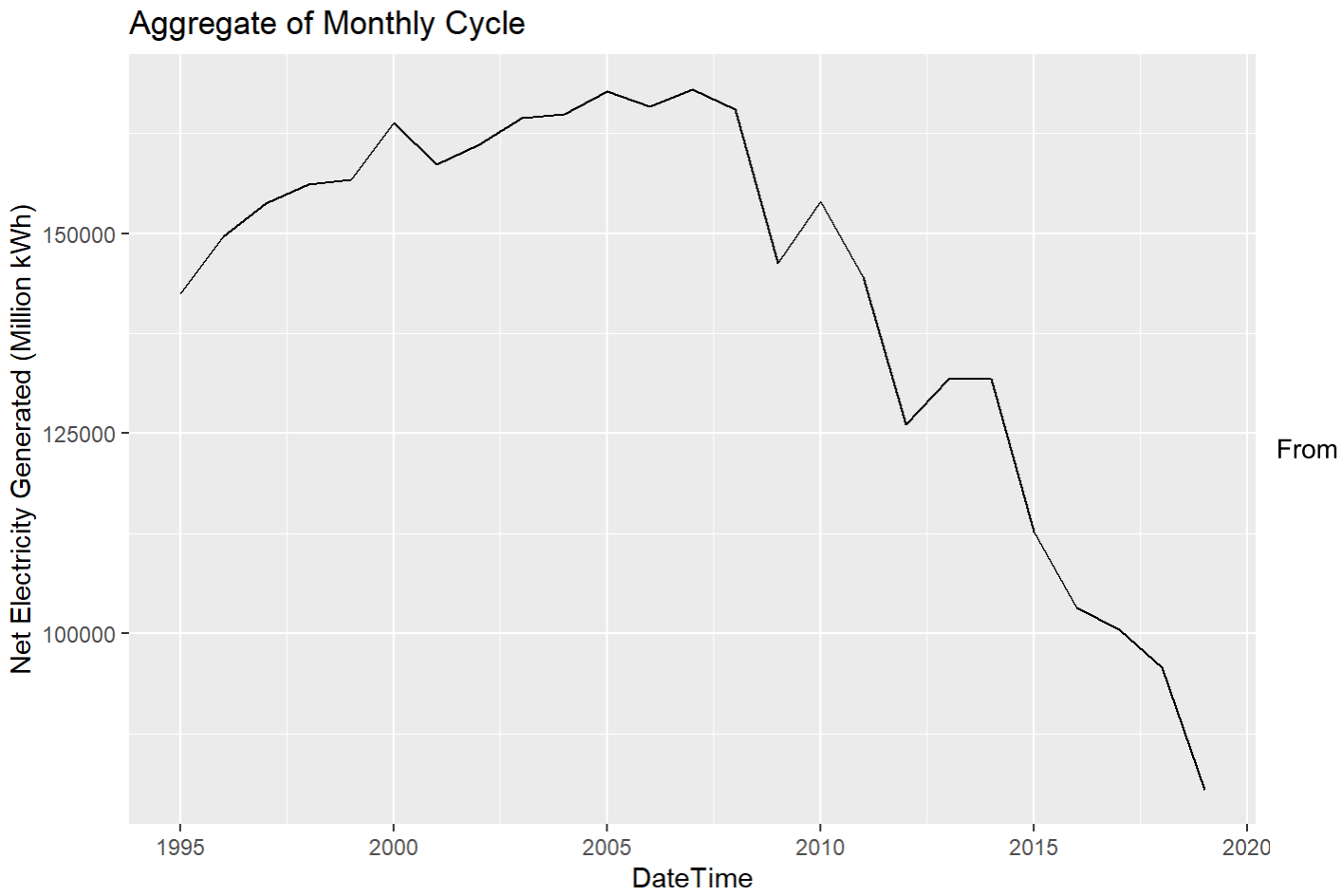
From these plots, it is clear that the electricity generation using coal has a clear trend. The overall electricity generation is decreasing rapidly in the last decade.

```
#Season Plot to explore monthly behavior
ggseasonplot(coal,continuous = TRUE)+
  xlab("Month") + ylab("Net Electricity Generated (Million kWh)") +
  ggtitle("Season Plot")
```



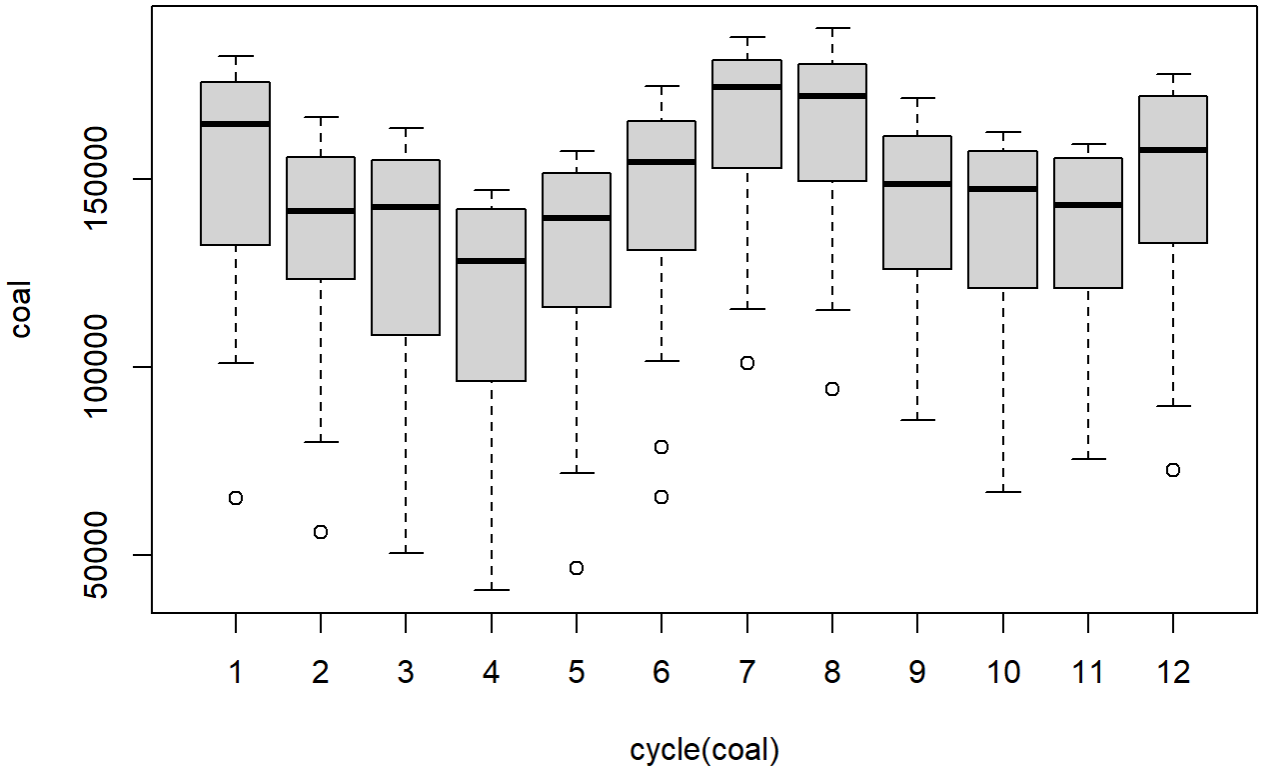
The season plot shows visible seasonality within each year. The electricity generation seems to increase during the summer and the winter months. Electricity generation is lowest during the spring months.

```
#This will aggregate the cycles and display a year on year trend
autoplot(aggregate(coal,FUN=mean))+
  xlab("DateTime") + ylab("Net Electricity Generated (Million kWh)") +
  ggtitle("Aggregate of Monthly Cycle")
```



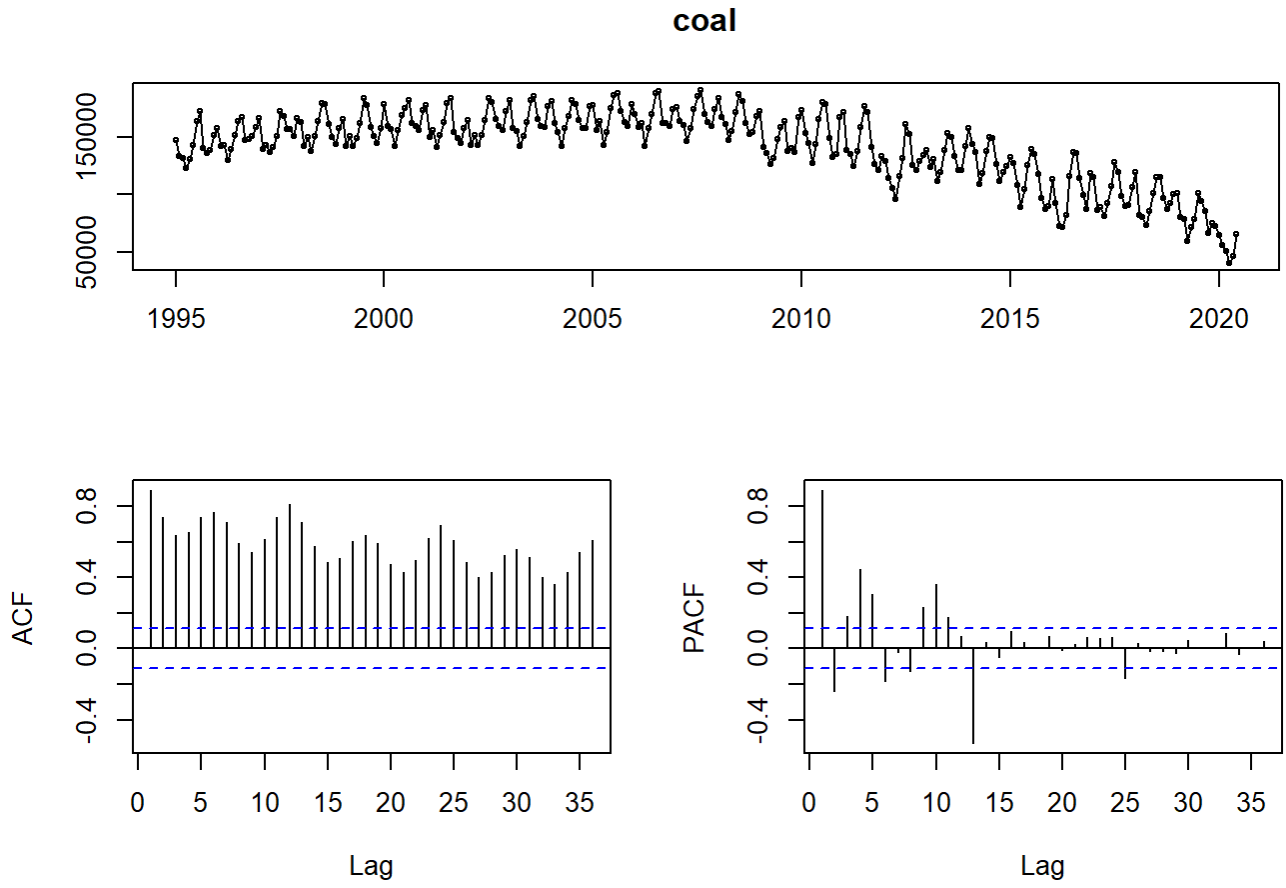
From this aggregate plot, we can see that electricity generation using coal was popular in the late 20th century and first few years in 21st century. Over the past decade, coal usage for electricity generation has decreased rapidly and continues to decrease even more in 2020.

```
boxplot(coal~cycle(coal))
```

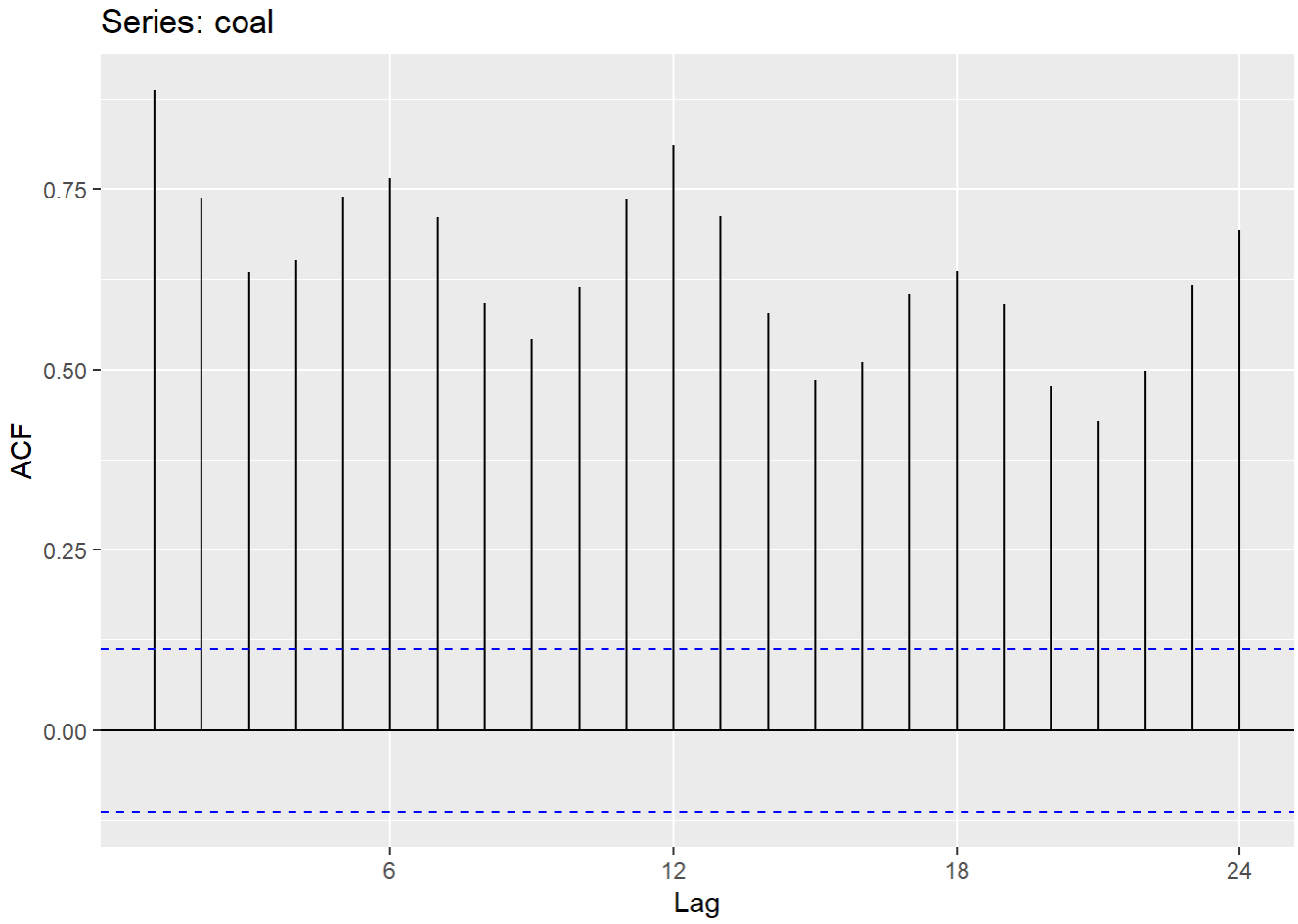


This box plot shows that the mean and variance change between months. July and August have the highest means and March and April have the biggest variance. This shows that our time series has a strong seasonal effect.

```
#ACF and PACF plots
tsdisplay(coal)
```


coal
150000
50000
1995
2000
2005
2010
2015
2020
ACF
0.8
0.4
0.0
-0.4
0
5
10
15
20
25
30
35
Lag
PACF
0.8
0.4
0.0
-0.4
0
5
10
15
20
25
30
35
Lag

```
ggAcf(coal)
```




```
#Boundary Value
T <- length(coal)
1.96/sqrt(T)
```

```
## [1] 0.1120457
```

```
paste('Mean:',mean(coal))
```

```
## [1] "Mean: 140902.440718954"
```

```
paste('Variance:',var(coal))
```

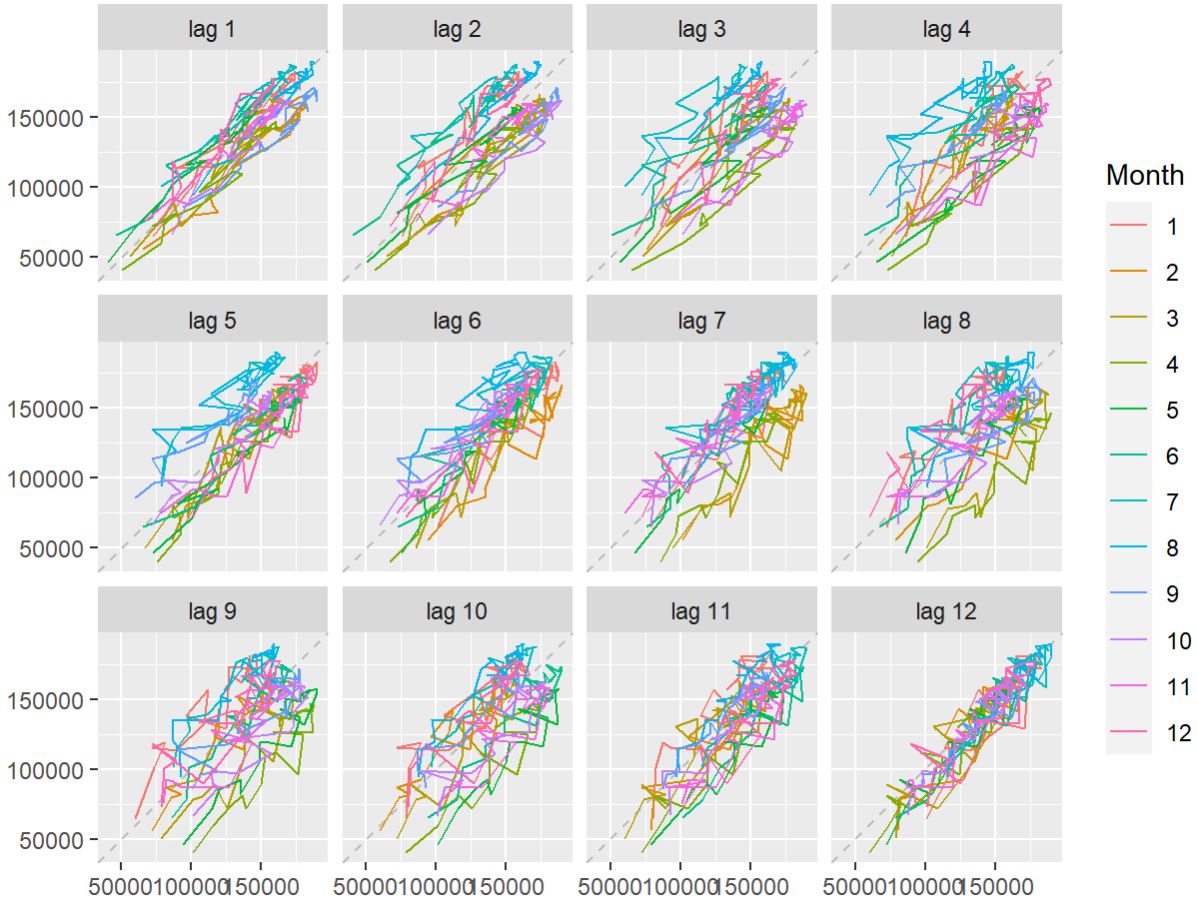
```
## [1] "Variance: 975070534.212082"
```

```
paste('Standard Deviation:',sd(coal))
```

```
## [1] "Standard Deviation: 31226.1194228819"
```

From the Acf plot, we can confirm that the series is not white noise since all the lag spikes lie outside ±1.96/√T which is + or - 0.1120457. ACF plot shows the time series has trend and seasonality.

```
gglagplot(coal,lags = 12)
```



gglagplot shows positive relationships at lag 12 reflecting strong seasonality in the data.

Basic Forecasting

For testing the forecasting methods, let us split the data into training set and testing set based on 80/20 method. i.e. 80% of the data is used for training and 20% is retained as testing set. We have about 25yrs in our dataset, so I am going to keep the last 5 years as a testing set and use the remaining data for training the forecast models.

```
coal_train <- subset(coal, end = length(coal)-60)
coal_test <- subset(coal, start = length(coal)-59)
str(coal_train)
```

```
##  Time-Series [1:246] from 1995 to 2015: 147220 132900 131430 123076 130418 ...
```

```
str(coal_test)
```

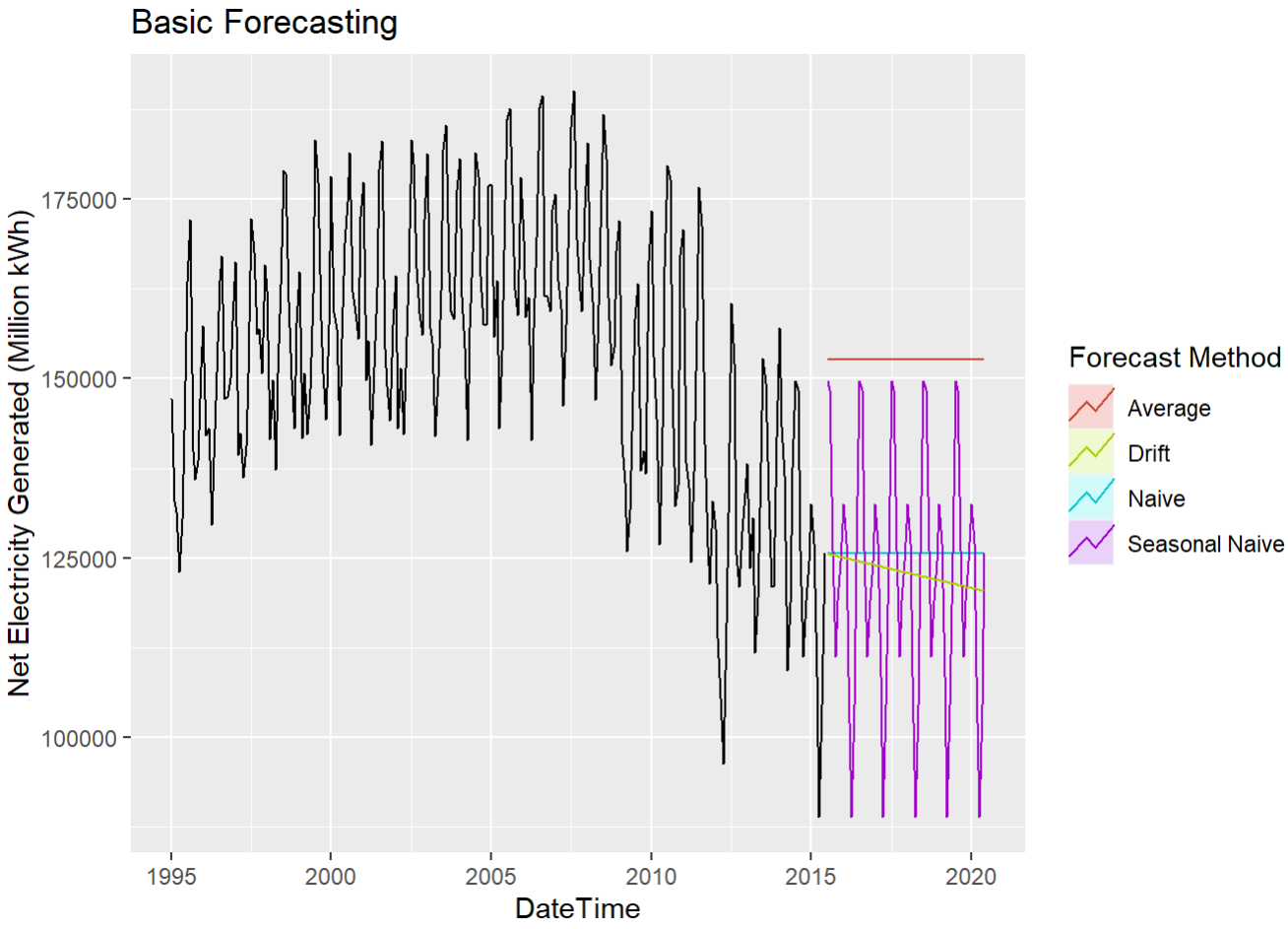
```
##  Time-Series [1:60] from 2016 to 2020: 139100 134670 117986 96759 87227 ...
```

Four Benchmark Forecasting Methods

Let's perform all 4 benchmark forecasting methods on the training set and test their accuracy on the testing set. - Naive Method -Seasonal Naive Method -Average Method -Drift Method

```
h=60
coal_train1 <- naive(coal_train, h)
coal_train2 <- snaive(coal_train, h)
coal_train3 <- meanf(coal_train, h)
coal_train4 <- rwf(coal_train, h, drift = TRUE)

autoplot(coal_train) +
  autolayer(coal_train1, PI = FALSE, series="Naive")+
  autolayer(coal_train2, PI = FALSE, series="Seasonal Naive")+
  autolayer(coal_train3, PI = FALSE, series="Average")+
  autolayer(coal_train4, PI = FALSE, series="Drift")+
  xlab("DateTime") + ylab("Net Electricity Generated (Million kWh)")+
  ggtitle("Basic Forecasting")+
  guides(colour=guide_legend(title="Forecast Method"))
```



```
#Comparing accuracy of forecasting methods
ar.naive <- accuracy(coal_train1, coal_test)
ar.snaive <- accuracy(coal_train2, coal_test)
ar.mean <- accuracy(coal_train3, coal_test)
ar.drift <- accuracy(coal_train4, coal_test)
RMSE=c(ar.mean["Test set","RMSE"],ar.naive["Test set","RMSE"],ar.snaive["Test set","RMSE"],ar.dr
ift["Test set","RMSE"])
MAE = c(ar.mean["Test set","MAE"],ar.naive["Test set","MAE"],ar.snaive["Test set","MAE"],ar.drif
t["Test set","MAE"])
MAPE = c(ar.mean["Test set","MAPE"],ar.naive["Test set","MAPE"],ar.snaive["Test set","MAPE"],ar.
drift["Test set","MAPE"])
MASE = c(ar.mean["Test set","MASE"],ar.naive["Test set","MASE"],ar.snaive["Test set","MASE"],ar.
drift["Test set","MASE"])
results <-data.frame(RMSE,MAE,MAPE,MASE, row.names = c("Mean","Naive","Seasonal Naive","Drift"))
results
```

	RMSE <dbl>	MAE <dbl>	MAPE <dbl>	MASE <dbl>
Mean	64256.42	60203.30	76.73612	7.869346
Naive	40058.20	34671.19	46.55070	4.531971
Seasonal Naive	33647.10	29695.81	37.90567	3.881625
Drift	37338.28	32150.00	43.25276	4.202419

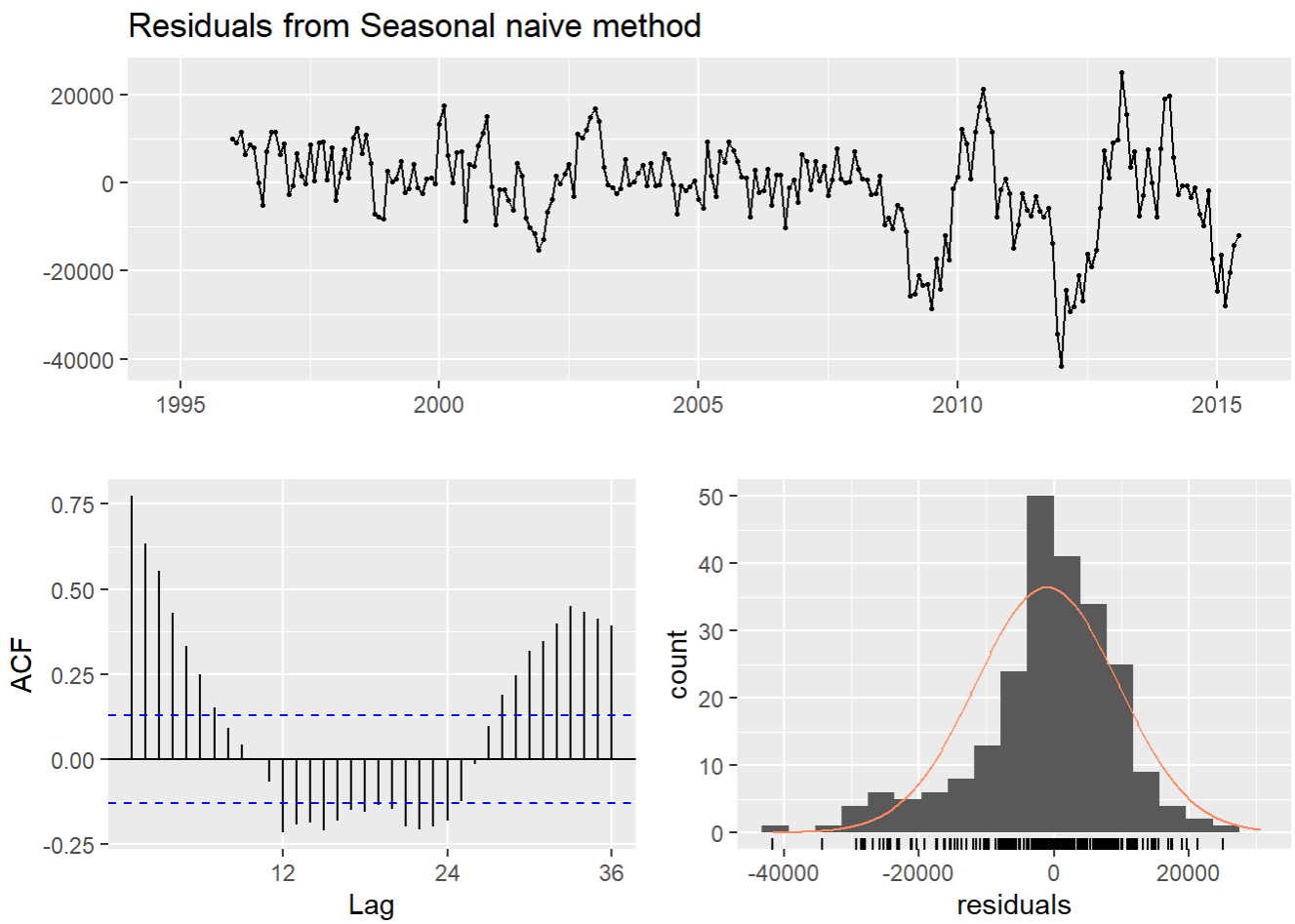
4 rows

From the above table, we can see that the **Seasonal Naive Method** performs better than the other methods.

Let’s compare the seasonal prediction with the actual test values.

```
autoplot(coal_train) + autolayer(coal_test,series = "Test set") + autolayer(coal_train2, PI=FALS
E, series = "Snaive") +
  xlab("DateTime") + ylab("Net Electricity Generated (Million kWh)")
```


```
checkresiduals(coal_train2)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 516.88, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

Again not white noise.

Let's repeat the above 4 forecasting methods using cross-validation and see seasonal method is still the best.

```
h=60
fd <- tsCV(coal, rwf, drift=TRUE, h)
paste("Drift RSME:", sqrt(mean(fd^2, na.rm=TRUE)))
```

```
## [1] "Drift RSME: 52372.7742751471"
```

```
fna <- tsCV(coal, naive, h)
paste("Naive RSME:",sqrt(mean(fna^2, na.rm=TRUE)))
```

```
## [1] "Naive RSME: 26082.6538591161"
```

```
fsna <- tsCV(coal, snaive, h)
paste("Snaive RSME:",sqrt(mean(fsna^2, na.rm=TRUE)))
```

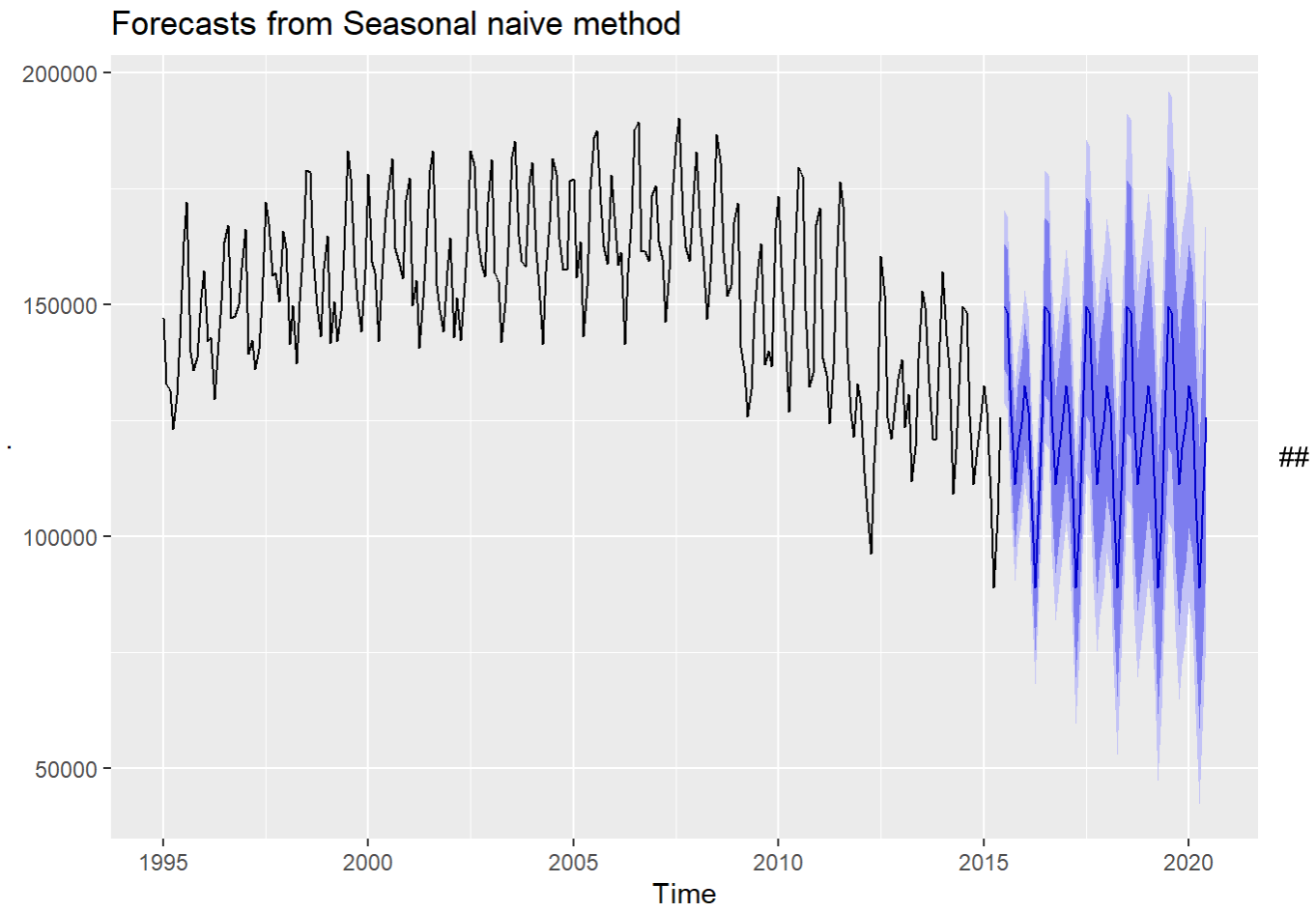
```
## [1] "Snaive RSME: 20076.8544210978"
```

```
fm <- tsCV(coal, meanf, h)
paste("Average RSME:",sqrt(mean(fm^2, na.rm=TRUE)))
```

```
## [1] "Average RSME: 35292.8015733204"
```

Cross-validation method also confirms that snaive method is the best out of the 4 basic methods.

```
coal_train %>% snaive(h=60) %>% autoplot()
```



##

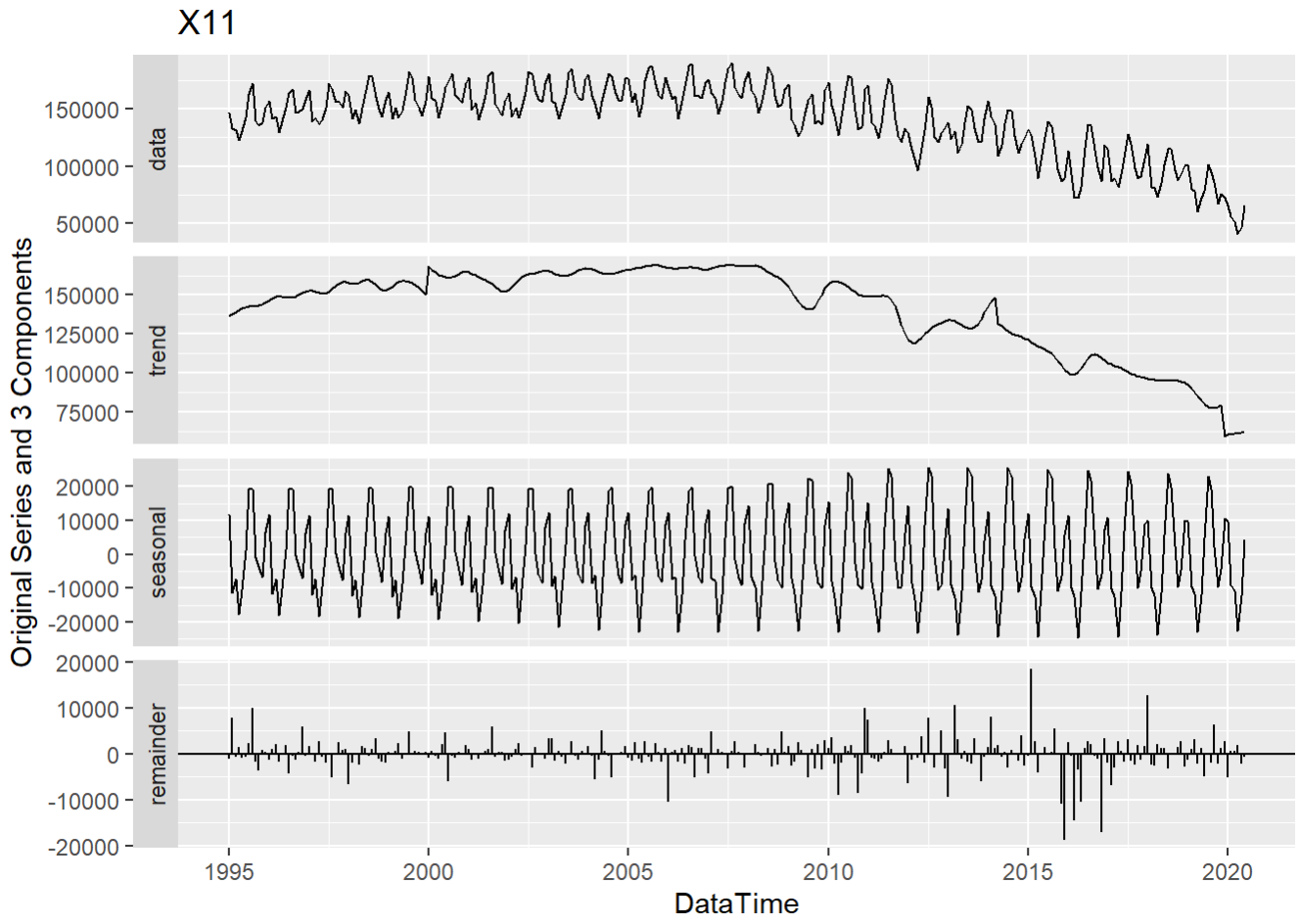
Decomposition

A seasonal time series consists of a trend component, a seasonal component and an irregular component. Decomposing the time series means separating the time series into these three components and estimating their values.

Electricity Generation using Coal has changed drastically over time in our time series and Classical Decomposition methods are unable to capture seasonal changes over time. X11, SEATS and STL methods might be best suited for our time series.

X11

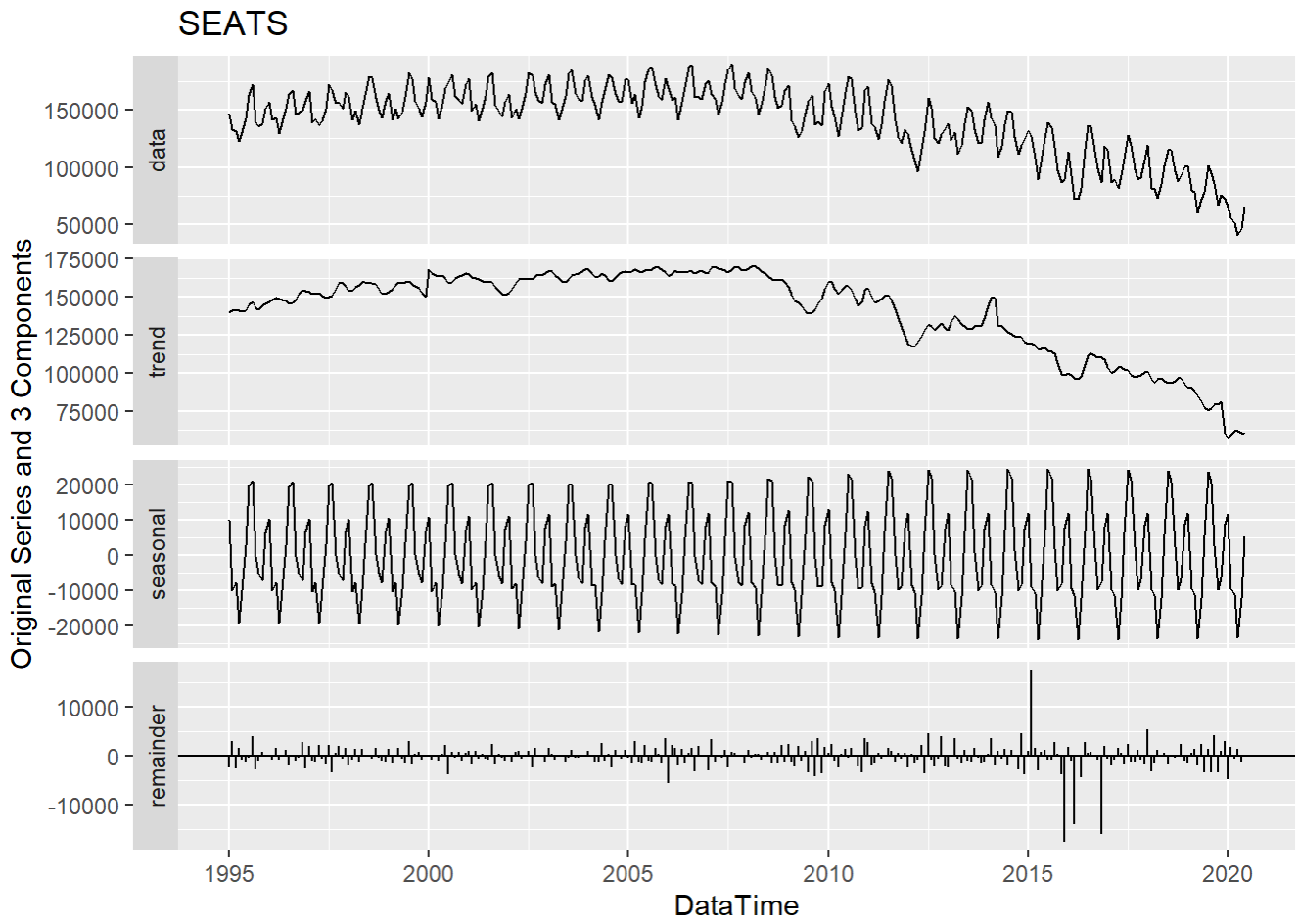
```
at.x11 <- seas(coal, x11="")
autoplot(at.x11) + xlab("DataTime") + ylab("Original Series and 3 Components") +
  ggtitle("X11")
```



###

SEATS

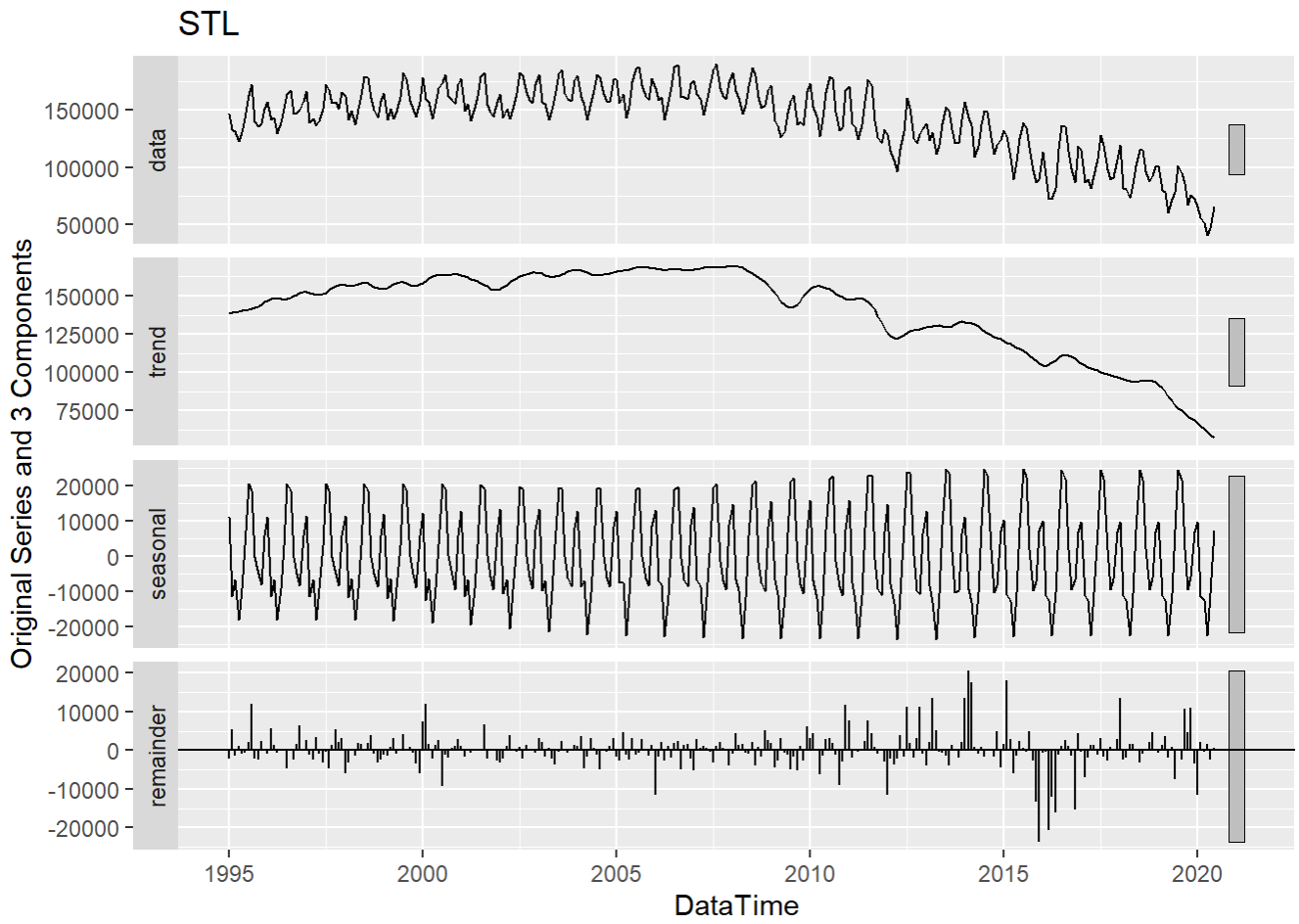
```
at.s <- seas(coal)
autoplot(at.s) + xlab("DataTime") + ylab("Original Series and 3 Components") +
  ggtitle("SEATS")
```



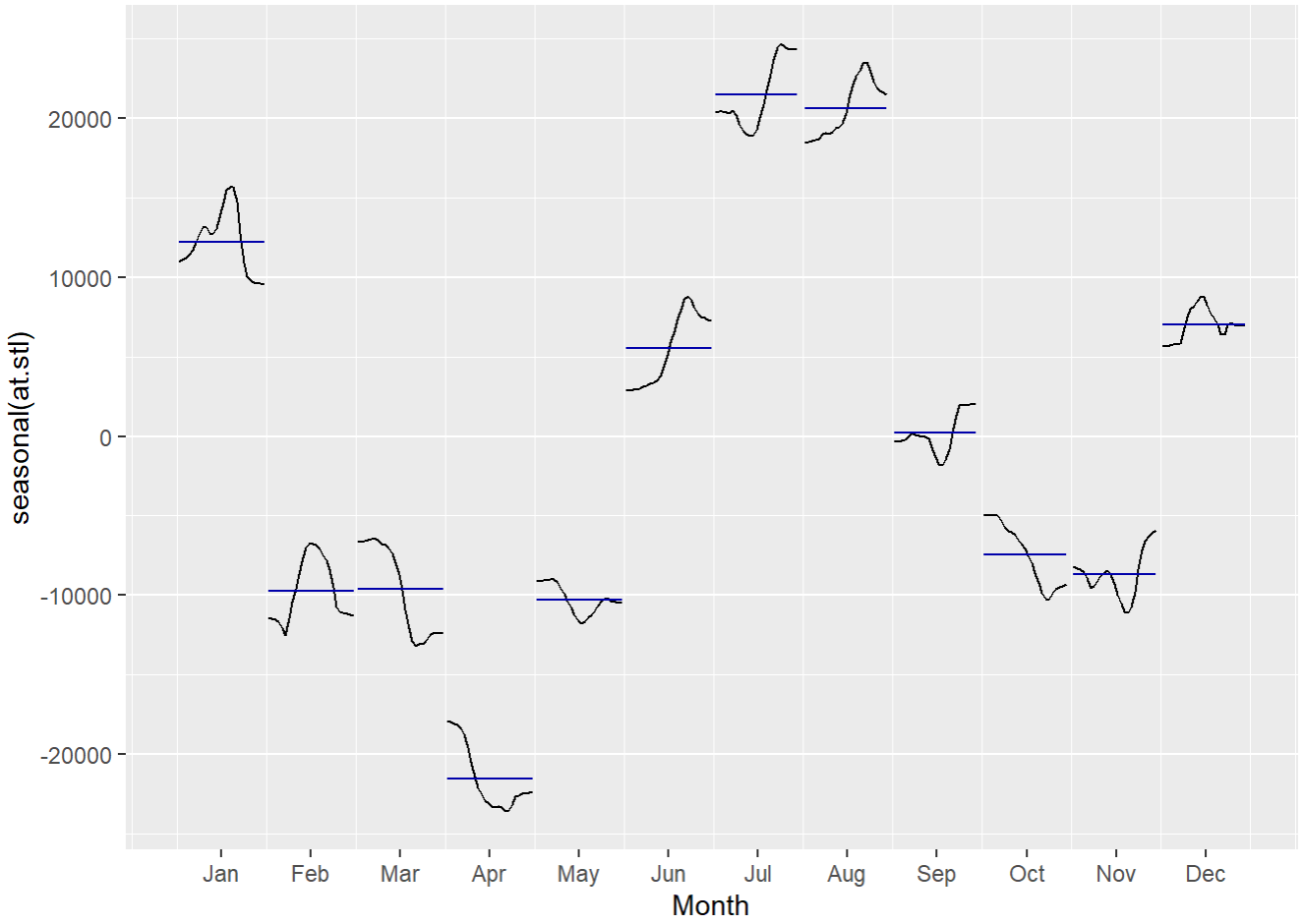
###

STL

```
at.stl <- stl(coal, s.window=10, t.window= 10, robust = TRUE)
autoplot(at.stl) + xlab("DataTime") + ylab("Original Series and 3 Components") +
  ggtitle("STL")
```




```
ggsubseriesplot(seasonal(at.stl))
```



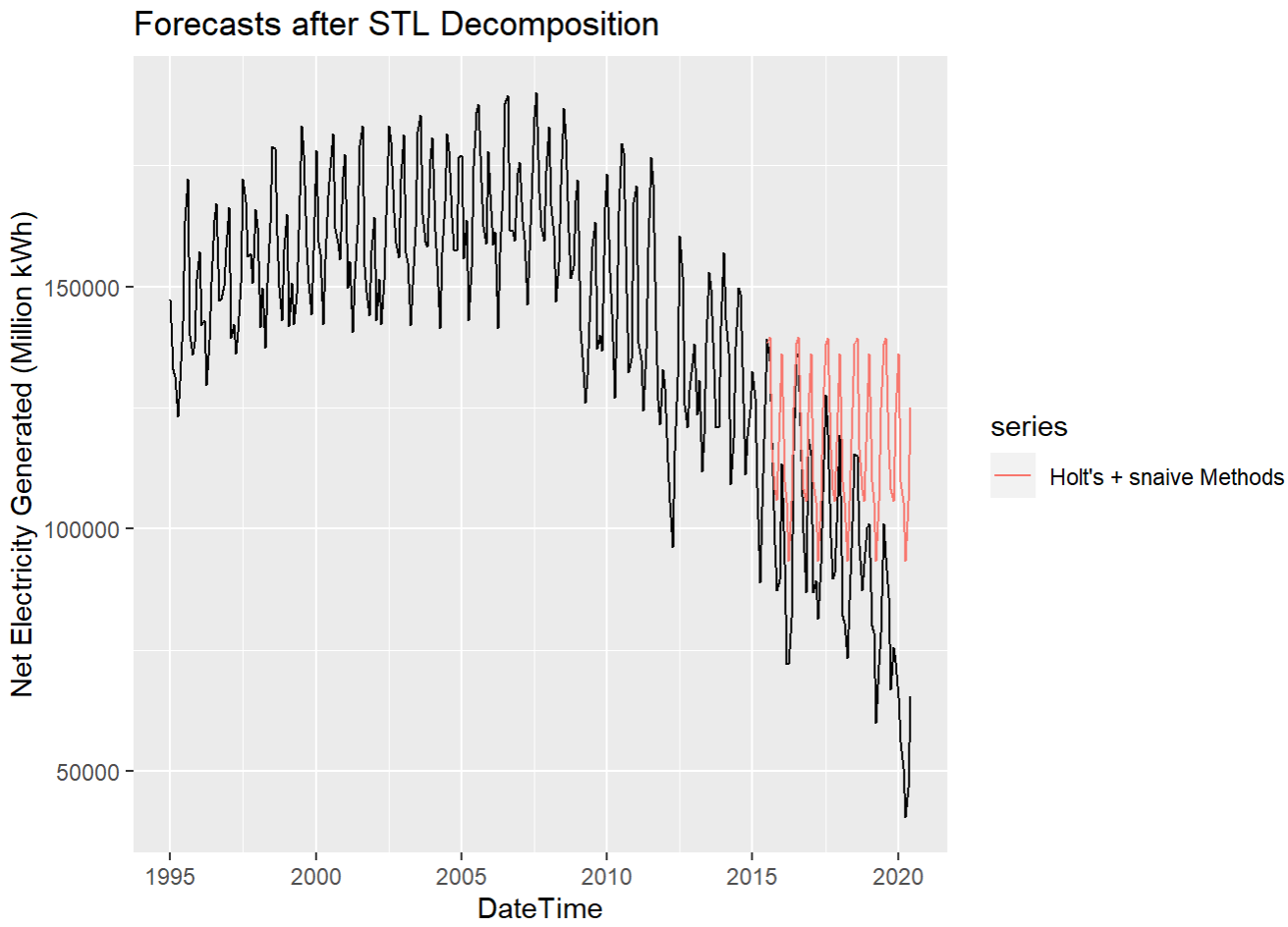
STL Decomposition method with s.window = 10 and t.window = 10 seems to capture the trend of the time series better than other methods. Let's use this stl method to do some forecasting. Holt' Linear Method extends simple exponential smoothing to allow the forecasting of data with a trend. I'm using Holt's Linear trend method and snaive method to forecast the seasonally adjusted series.

```
cstl <- at.stl <- stl(coal_train, s.window=10, t.window= 10, robust = TRUE)
cstl.noseas <- seasadj(cstl) #Seasonality removed
cstl.seas <- seasonal(cstl)

#Holt's method with damped trend on noseas data
cstl.noseas.fit <- holt(cstl.noseas, h=60, damped = TRUE)

#snaive on seas data
cstl.seas.fit <- snaive(cstl.seas, h=60)

#Plot
autoplot(coal) +
  autolayer(cstl.noseas.fit$mean+cstl.seas.fit$mean, series="Holt's + snaive Methods") +
  ggtitle("Forecasts after STL Decomposition") + xlab("DateTime") +
  ylab("Net Electricity Generated (Million kWh)")
```




```
#Accuracy
cstl.acc <- accuracy(cstl.noseas.fit$mean+cstl.seas.fit$mean,coal_test)

RMSE=c(ar.mean["Test set","RMSE"],ar.naive["Test set","RMSE"],ar.snaive["Test set","RMSE"],ar.dr
ift["Test set","RMSE"],cstl.acc["Test set","RMSE"])

MAE = c(ar.mean["Test set","MAE"],ar.naive["Test set","MAE"],ar.snaive["Test set","MAE"],ar.drif
t["Test set","MAE"],cstl.acc["Test set","MAE"])
results1 <-data.frame(RMSE,MAE, row.names = c("Mean","Naive","Seasonal Naive","Drift","STL on Ho
lt's + snaive"))
results1
```

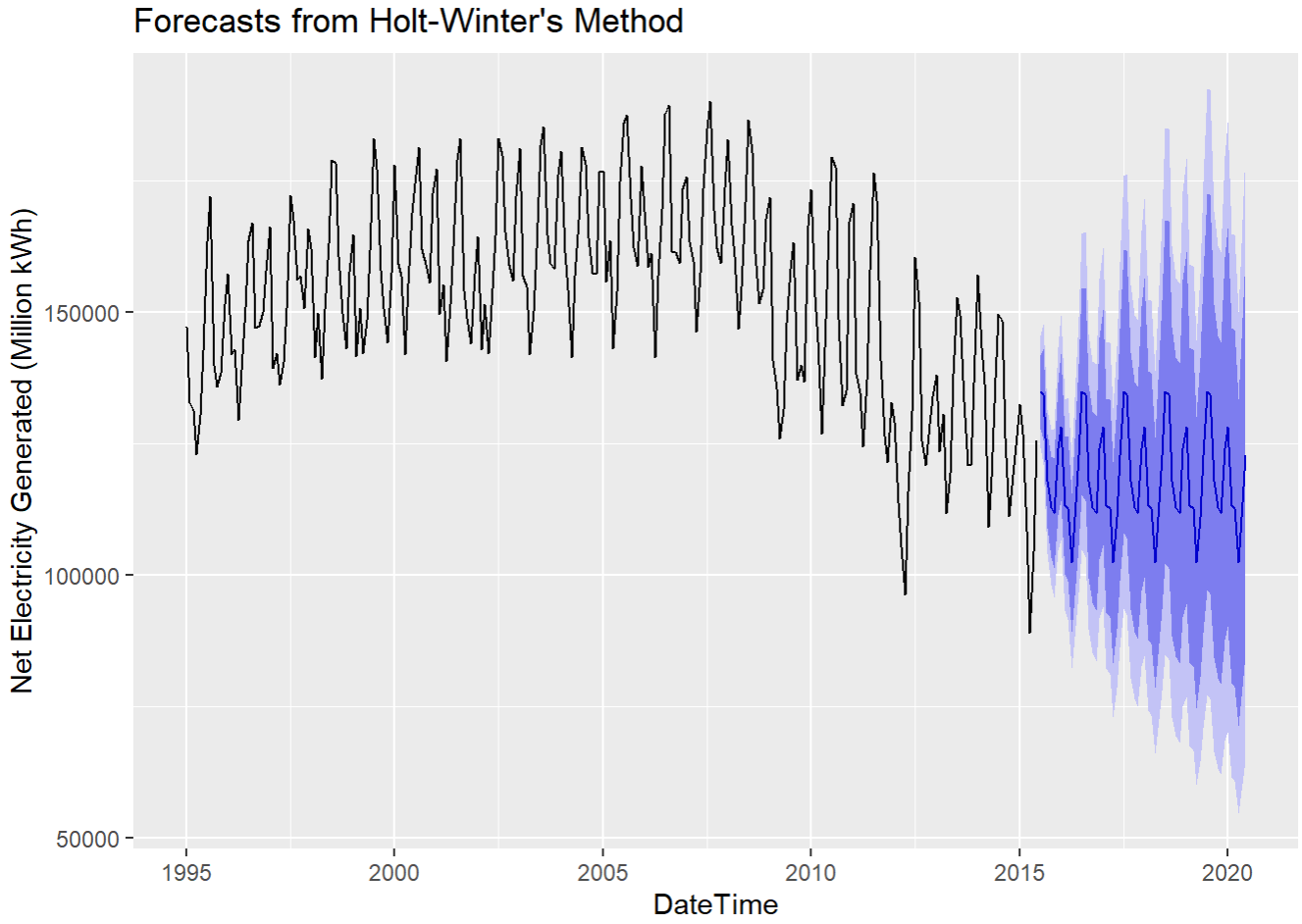
	RMSE <dbl>	MAE <dbl>
Mean	64256.42	60203.30
Naive	40058.20	34671.19
Seasonal Naive	33647.10	29695.81
Drift	37338.28	32150.00
STL on Holt's + snaive	29480.68	24886.05

5 rows

From this we can conclude that applying stl decomposition on Holt's Linear + snaive methods provides much lower errors than the forecasting methods before decomposition.

Holt-Winters Method extends Holt's method to capture seasonality. The Holt-Winters seasonal method comprises the forecast equation and three smoothing equations — level, trend and seasonal component with corresponding smoothing parameters $\alpha, \beta \, and \, \gamma$. The additive method is preferred when the seasonal variations are roughly constant through the series, while the multiplicative method is preferred when the seasonal variations are changing proportional to the level of the series.

```
cfitm <- hw(coal_train,seasonal="multiplicative",h=60,damped=TRUE)
autoplot(cfitm) +
  ggtitle("Forecasts from Holt-Winter's Method") + xlab("DateTime") +
  ylab("Net Electricity Generated (Million kWh)")
```



```
#Accuracy
cfitm.acc <- accuracy(cfitm,coal_test)

RMSE=c(ar.mean["Test set","RMSE"],ar.naive["Test set","RMSE"],ar.snaive["Test set","RMSE"],ar.dr
ift["Test set","RMSE"],cstl.acc["Test set","RMSE"],cfitm.acc["Test set","RMSE"])

MAE = c(ar.mean["Test set","MAE"],ar.naive["Test set","MAE"],ar.snaive["Test set","MAE"],ar.drif
t["Test set","MAE"],cstl.acc["Test set","MAE"],cfitm.acc["Test set","MAE"])

results2 <-data.frame(RMSE,MAE, row.names = c("Mean","Naive","Seasonal Naive","Drift","STL on Ho
lt's + snaive","Holt-Winters"))
results2
```

	RMSE <dbl>	MAE <dbl>
Mean	64256.42	60203.30
Naive	40058.20	34671.19
Seasonal Naive	33647.10	29695.81
Drift	37338.28	32150.00
STL on Holt's + snaive	29480.68	24886.05
Holt-Winters	31325.66	26627.93

6 rows

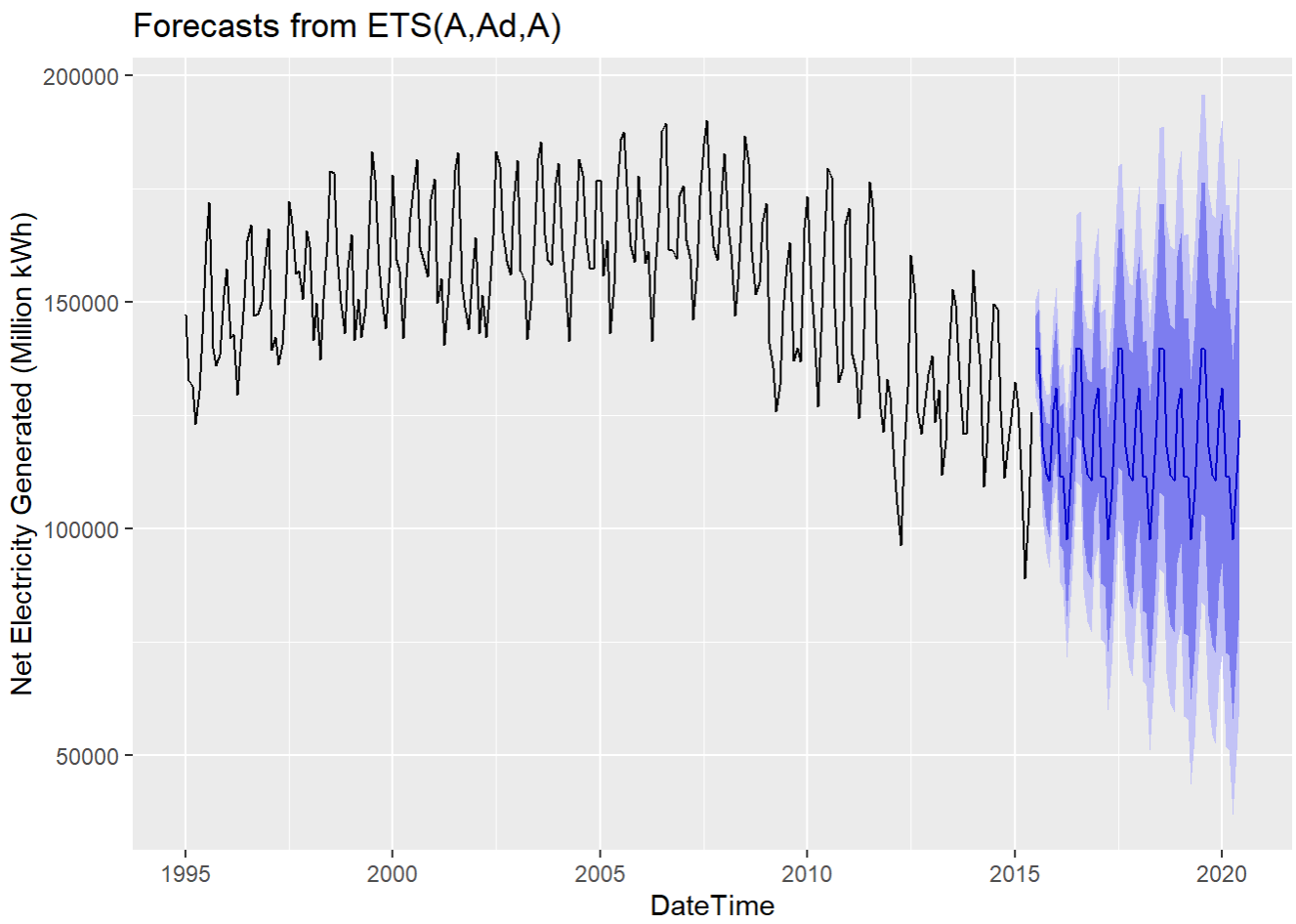
Forecasting using ETS

Error={A,M}, Trend={N,A,Ad}, and Seasonal={N,A,M}

```
ets1 <- ets(coal_train, ic = 'aic', damped = TRUE, restrict = FALSE) #Based on best aic value
ets1
```

```
## ETS(A,Ad,A)
##
## Call:
##  ets(y = coal_train, damped = TRUE, ic = "aic", restrict = FALSE)
##
##   Smoothing parameters:
##     alpha = 0.7256
##     beta  = 1e-04
##     gamma = 1e-04
##     phi   = 0.9749
##
##   Initial states:
##     l = 140159.9642
##     b = 607.7708
##     s = 7066.521 -8686.596 -7264.047 -1127.986 20405.34 20562.92
##            4622.358 -10544.96 -21585.94 -7705.628 -7596.084 11854.1
##
##   sigma:  5547.904
##
##      AIC     AICc      BIC
## 5614.314 5617.327 5677.410
```

```
ets1.fcst <- forecast(ets1,h=60)
ets1.acc <- accuracy(ets1.fcst,coal_test)
autoplot(ets1.fcst)+xlab("DateTime") +
  ylab("Net Electricity Generated (Million kWh)")
```




```
#Accuracy
RMSE=c(ar.mean["Test set","RMSE"],ar.naive["Test set","RMSE"],ar.snaive["Test set","RMSE"],ar.dr
ift["Test set","RMSE"],cstl.acc["Test set","RMSE"],cfitm.acc["Test set","RMSE"],ets1.acc["Test s
et","RMSE"])

MAE = c(ar.mean["Test set","MAE"],ar.naive["Test set","MAE"],ar.snaive["Test set","MAE"],ar.drif
t["Test set","MAE"],cstl.acc["Test set","MAE"],cfitm.acc["Test set","MAE"],ets1.acc["Test set",
"MAE"])

results3 <-data.frame(RMSE,MAE, row.names = c("Mean","Naive","Seasonal Naive","Drift","STL on Ho
lt's + snaive","Holt-Winters","ETS(A,Ad,A)"))
results3
```

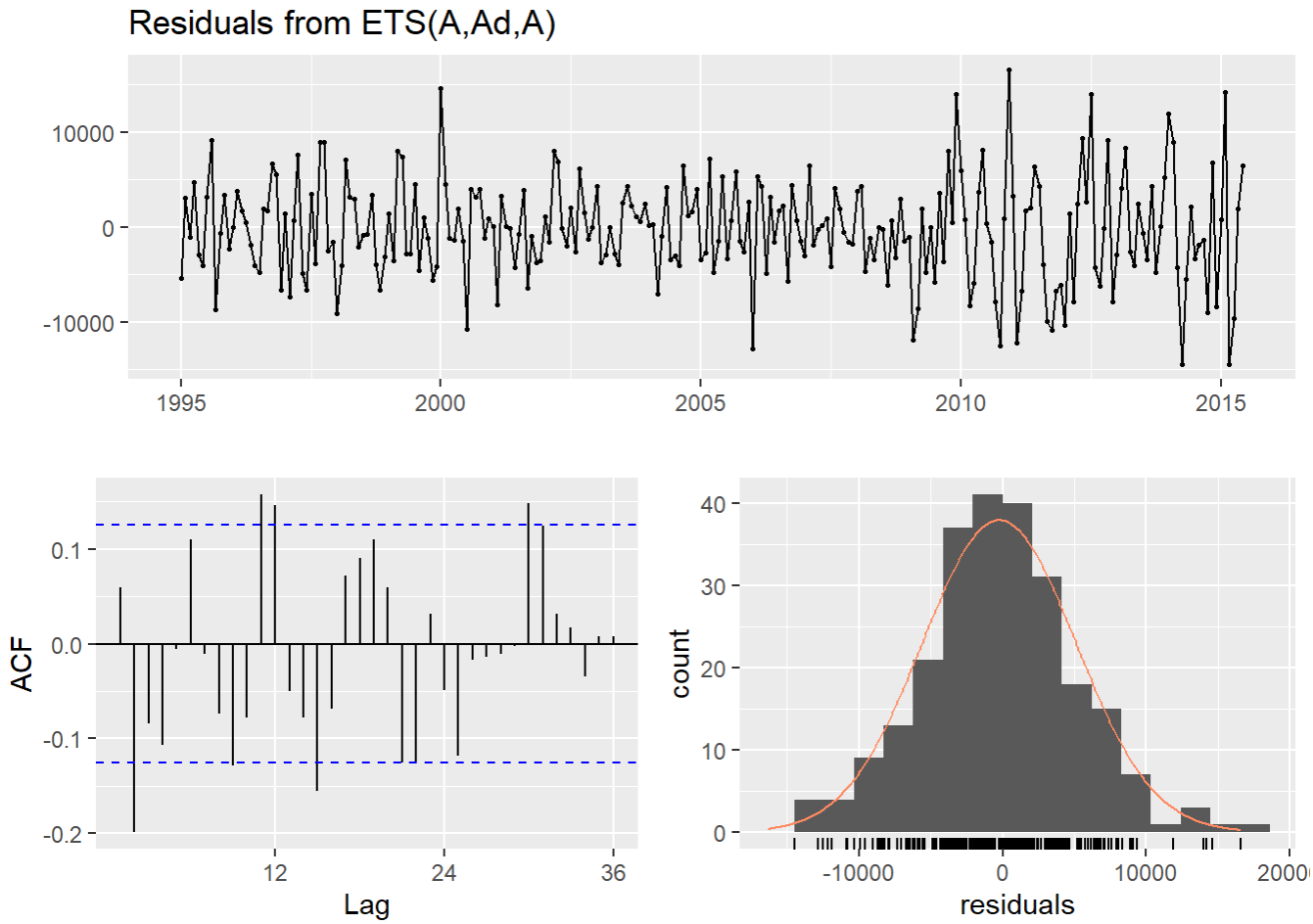
	RMSE <dbl>	MAE <dbl>
Mean	64256.42	60203.30
Naive	40058.20	34671.19
Seasonal Naive	33647.10	29695.81
Drift	37338.28	32150.00
STL on Holt's + snaive	29480.68	24886.05
Holt-Winters	31325.66	26627.93

	RMSE <dbl>	MAE <dbl>
ETS(A,Ad,A)	31134.06	26749.64

7 rows

Ets method with 'AAA' model and damped trend seems to perform better than Holt-Winters Multiplicative method. But they are both almost similar.

```
checkresiduals(ets1.fcst)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,Ad,A)
## Q* = 64.744, df = 7, p-value = 1.693e-11
##
## Model df: 17.   Total lags used: 24
```

ARIMA

While exponential smoothing models are based on a description of the trend and seasonality in the data, ARIMA models aim to describe the autocorrelations in the data.

Let's first check if the time series needs differencing or not.

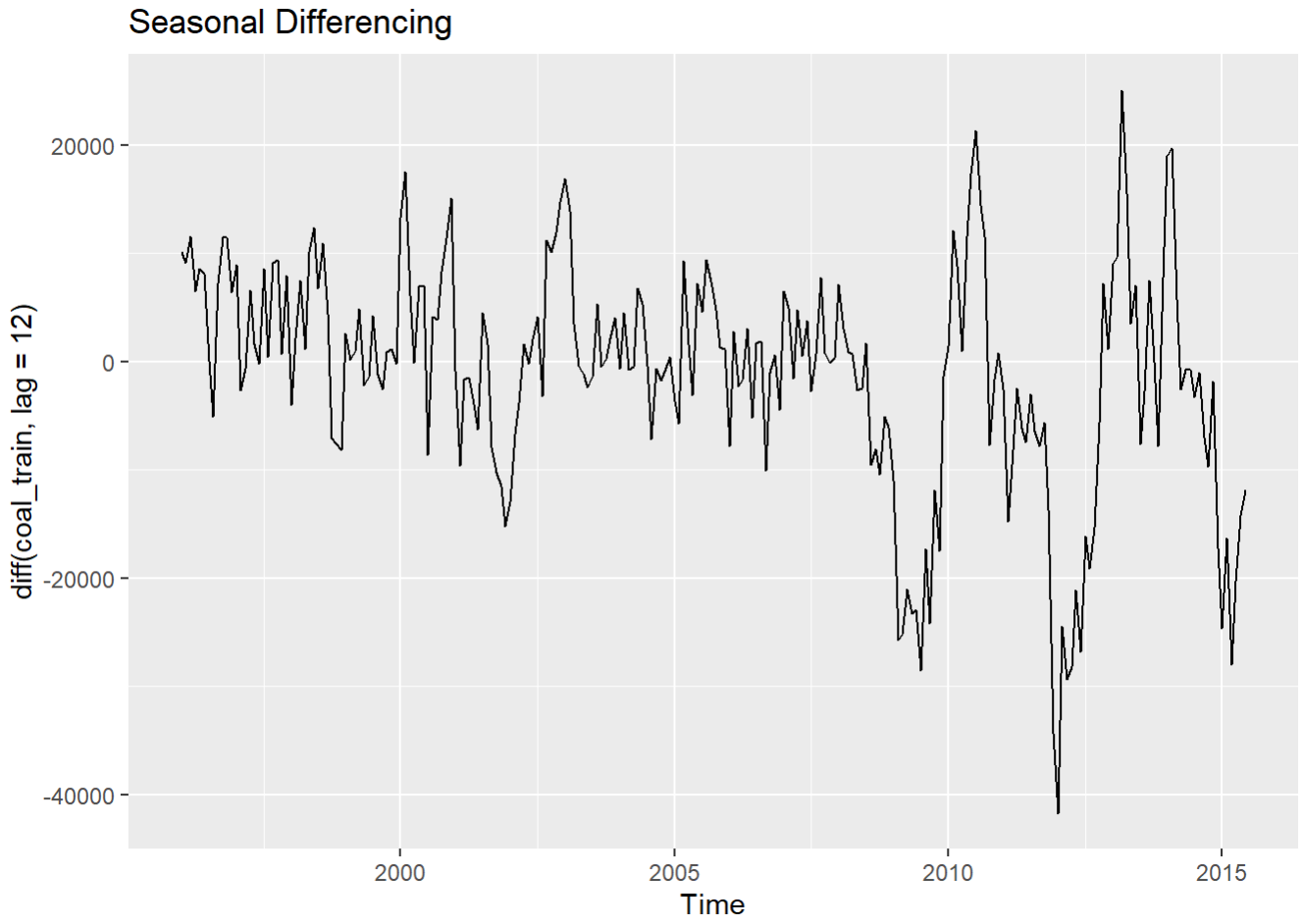
```
adf.test(coal_train)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  coal_train
## Dickey-Fuller = -2.5905, Lag order = 6, p-value = 0.3273
## alternative hypothesis: stationary
```

The p-value in ADF Test is not small enough so the data is not stationary. We need differencing.

Let's try seasonal differencing and repeat the above steps

```
autoplot(diff(coal_train,lag=12)) + ggtitle('Seasonal Differencing')
```



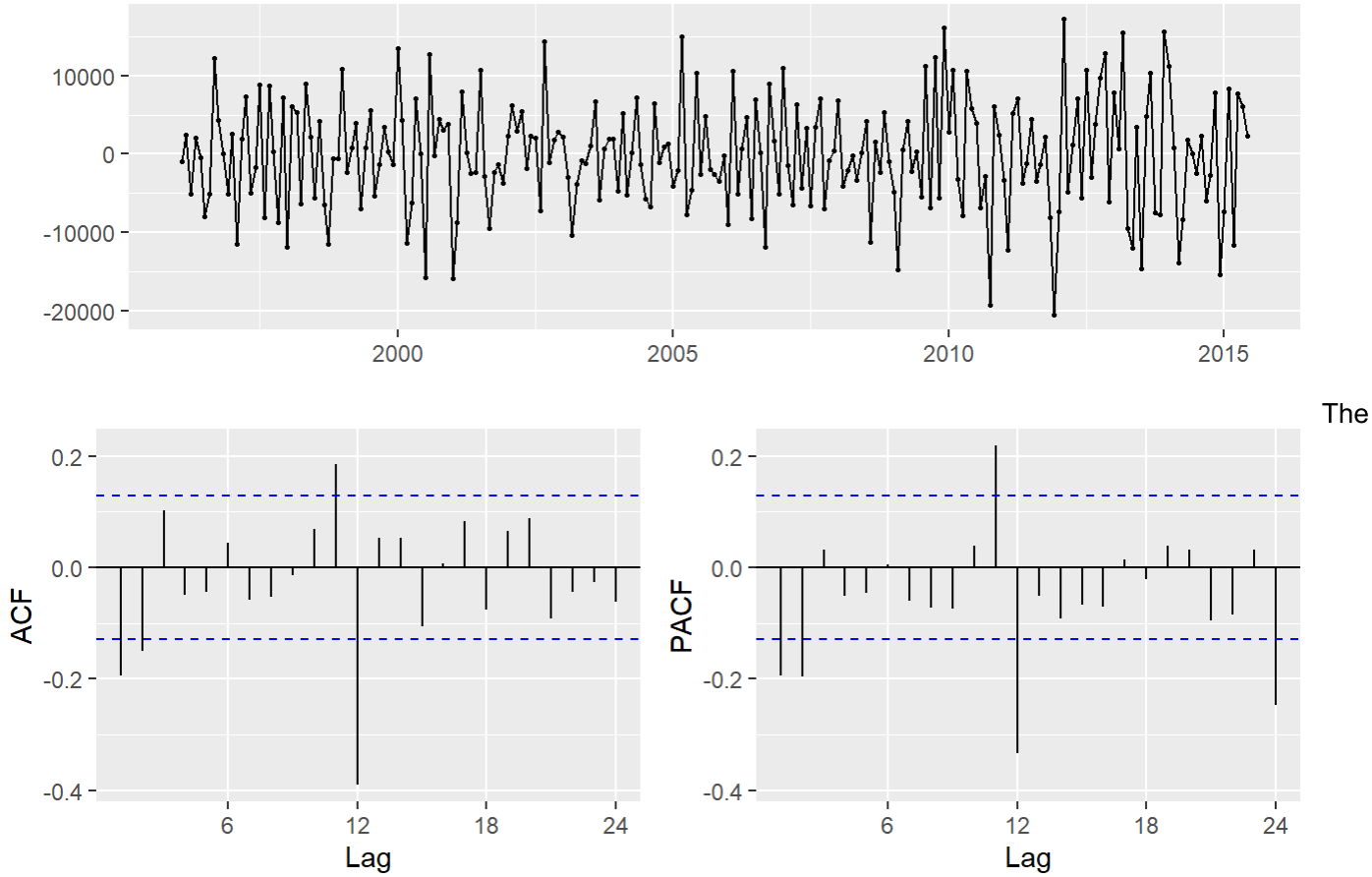

```
adf.test(diff(coal_train,lag=12))
```

```
## Warning in adf.test(diff(coal_train, lag = 12)): p-value smaller than printed p-
## value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(coal_train, lag = 12)
## Dickey-Fuller = -4.7211, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

This proves that the time series is not stationary by default and needs differencing to become stationary.

```
ggtsdisplay(diff(diff(coal_train,lag=12)),lag.max=24)
```



The ACF shows significant spike at lag 12 and PACF shows significant spikes at lag 12 and 24.

```
fit1 <- arima(coal_train,order = c(2,1,1),seasonal = c(1,1,2))
summary(fit1)
```

```
##
## Call:
## arima(x = coal_train, order = c(2, 1, 1), seasonal = c(1, 1, 2))
##
## Coefficients:
##          ar1     ar2      ma1     sar1     sma1     sma2
##       0.6358  0.0279  -0.8933  -0.4209  -0.2940  -0.4234
## s.e.  0.1040  0.0815   0.0839   0.3071   0.2906   0.2259
##
## sigma^2 estimated as 29790807:  log likelihood = -2341.86,  aic = 4697.71
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE    MAPE      MASE
## Training set -627.9415 5312.008 4035.419 -0.5243691 2.73357 0.3435129
##                     ACF1
## Training set -0.01720863
```

Let's try a few more arima models.

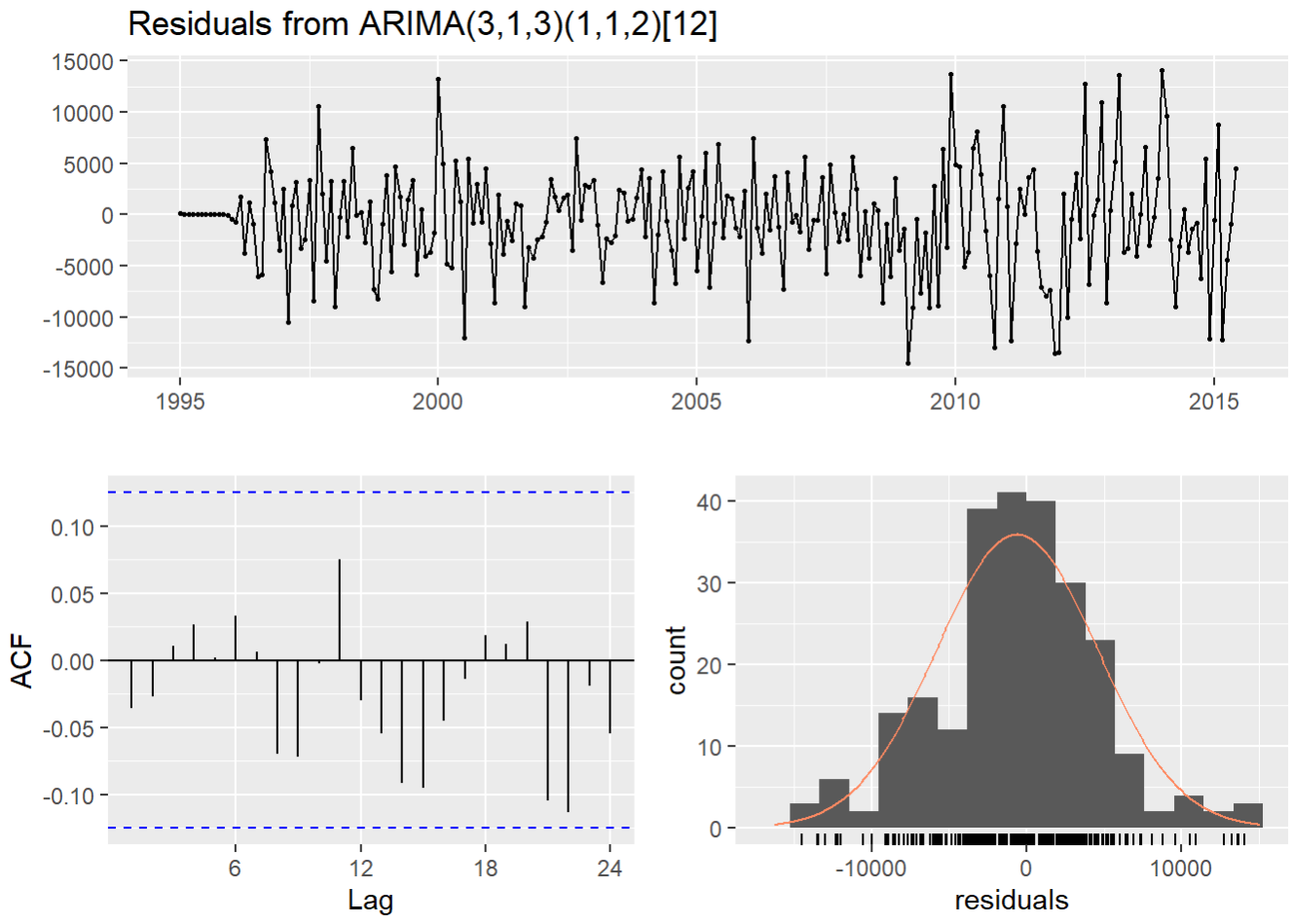
```
fit2 <- arima(coal_train,order = c(3,1,3),seasonal = c(1,1,2))
summary(fit2)
```

```
##
## Call:
## arima(x = coal_train, order = c(3, 1, 3), seasonal = c(1, 1, 2))
##
## Coefficients:
##           ar1      ar2     ar3     ma1      ma2      ma3     sar1     sma1
##       -0.5877  -0.0627  0.5986  0.3699  -0.2717  -0.7566  -0.3602  -0.3556
## s.e.   0.2276   0.1686  0.1363  0.2210   0.1287   0.1774   0.3330   0.3179
##          sma2
##       -0.3734
## s.e.   0.2471
##
## sigma^2 estimated as 29077532:  log likelihood = -2339.17,  aic = 4698.34
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -613.523 5248.033 3947.745 -0.5113172 2.665094 0.3360498
##                     ACF1
## Training set -0.03579695
```

```
fit3 <- arima(coal_train,order = c(1,1,0),seasonal = c(0,1,2))
summary(fit3)
```

```
##
## Call:
## arima(x = coal_train, order = c(1, 1, 0), seasonal = c(0, 1, 2))
##
## Coefficients:
##           ar1     sma1     sma2
##       -0.1763  -0.7012  -0.1099
## s.e.   0.0650   0.0740   0.0720
##
## sigma^2 estimated as 31783438:  log likelihood = -2349.06,  aic = 4706.12
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -279.1736 5486.78 4207.494 -0.2659637 2.856703 0.3581607
##                      ACF1
## Training set -0.04032498
```

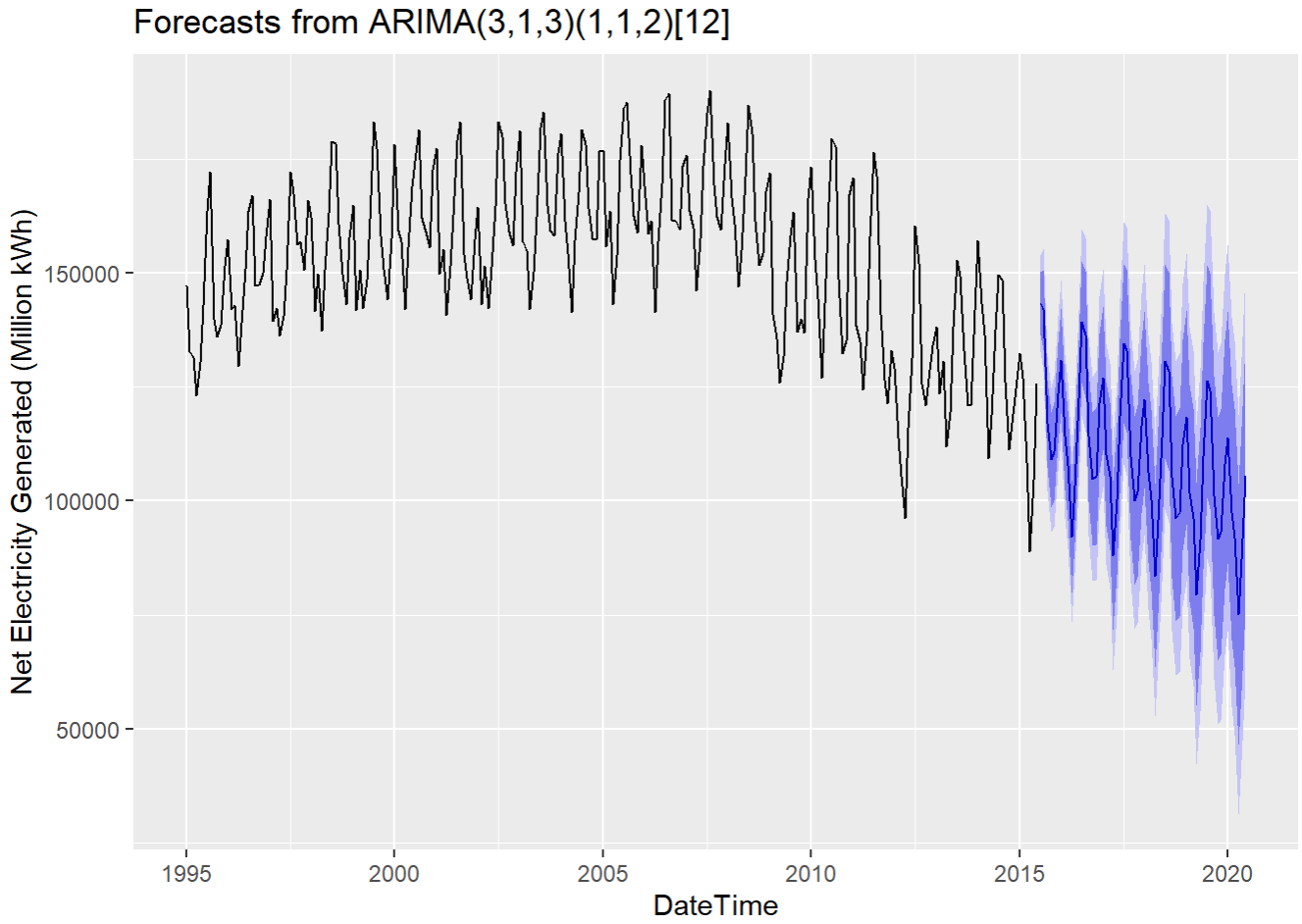
```
fit2 %>% checkresiduals(lag.max=24)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,1,3)(1,1,2)[12]
## Q* = 18.939, df = 15, p-value = 0.2165
##
## Model df: 9.   Total lags used: 24
```

ARIMA(3,1,3)(1,1,2)[12] is the best ARIMA model out of the 3.

```
autoplot(forecast(fit2,h=60))+xlab("DateTime") +
  ylab("Net Electricity Generated (Million kWh)")
```



```
#Accuracy
fit2.acc <- accuracy(forecast(fit2,h=60),coal_test)

RMSE=c(ar.mean["Test set","RMSE"],ar.naive["Test set","RMSE"],ar.snaive["Test set","RMSE"],ar.dr
ift["Test set","RMSE"],cstl.acc["Test set","RMSE"],cfitm.acc["Test set","RMSE"],ets1.acc["Test s
et","RMSE"],fit2.acc["Test set","RMSE"])

MAE = c(ar.mean["Test set","MAE"],ar.naive["Test set","MAE"],ar.snaive["Test set","MAE"],ar.drif
t["Test set","MAE"],cstl.acc["Test set","MAE"],cfitm.acc["Test set","MAE"],ets1.acc["Test set",
"MAE"],fit2.acc["Test set","MAE"])

results4 <-data.frame(RMSE,MAE, row.names = c("Mean","Naive","Seasonal Naive","Drift","STL on Ho
lt's + snaive","Holt-Winters","ETS(A,Ad,A)","ARIMA(3,1,3)(1,1,2)[12]"))
results4
```

	RMSE <dbl>	MAE <dbl>
Mean	64256.42	60203.30
Naive	40058.20	34671.19
Seasonal Naive	33647.10	29695.81
Drift	37338.28	32150.00
STL on Holt's + snaive	29480.68	24886.05
Holt-Winters	31325.66	26627.93
ETS(A,Ad,A)	31134.06	26749.64
ARIMA(3,1,3)(1,1,2)[12]	20919.52	17343.32

8 rows

From the accuracy measures, we can see that the ARIMA model performs much better than the rest.

Let’s do a cross validation on the ETS and ARIMA models and compare them again.

```
fets <- function(x, h) {
  forecast(ets(x), h=h)
}
farima <- function(x, h) {
  forecast(auto.arima(x), h=h)
}
```

Computing Errors for all the methods

```
e1<-tsCV(coal,fets,h=1)
e2<-tsCV(coal,farima,h=1)
```

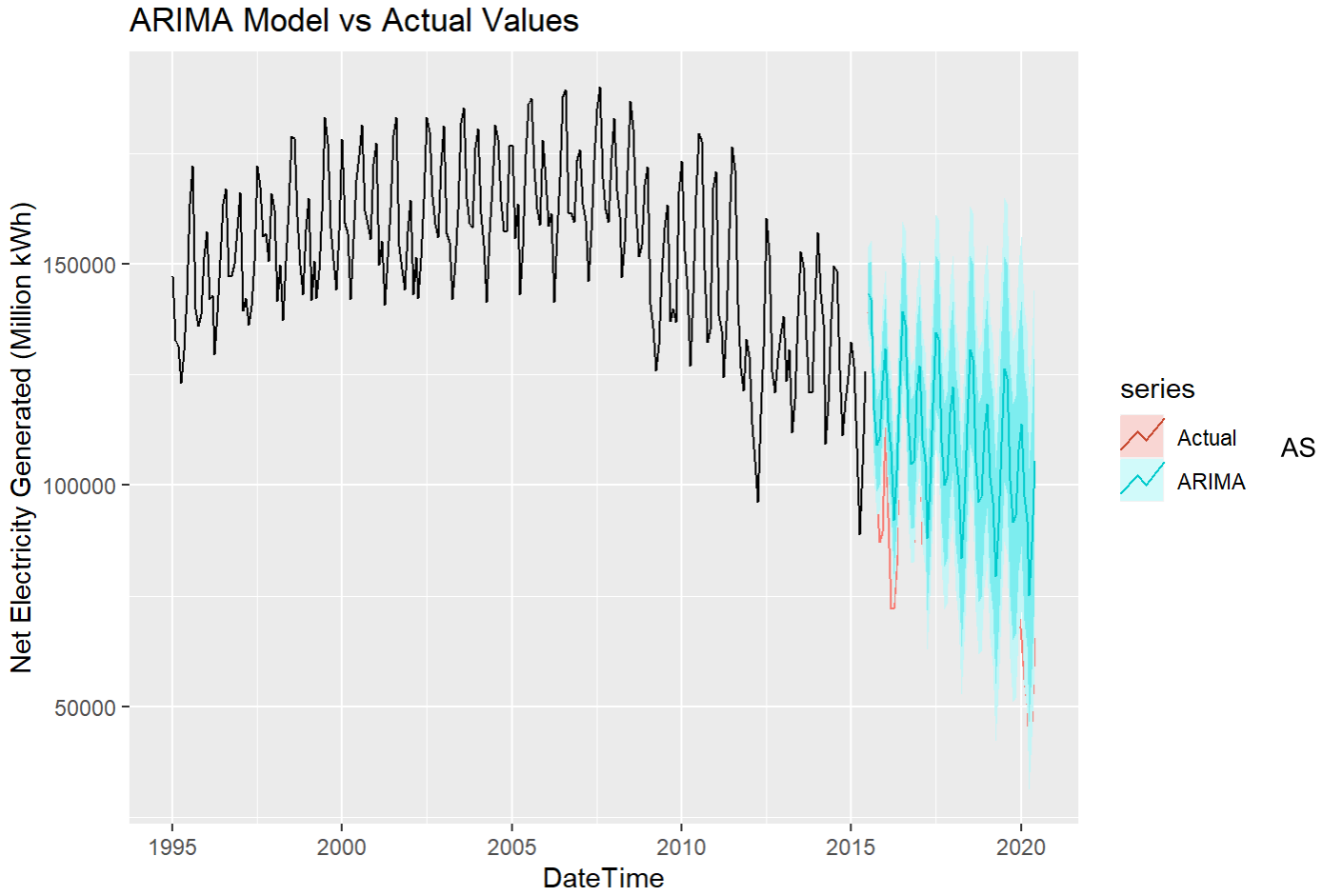
Root Mean Squared Error

```
rmse<-c(sqrt(mean(e1^2,na.rm=TRUE)),
sqrt(mean(e2^2,na.rm=TRUE)))
names(rmse)=c("ETS","ARIMA")
rmse
```

```
##      ETS    ARIMA
## 7582.407 7295.512
```

From the cross validations results we can confirm that the ARIMA model has the lowest RSME out of all the forecasting models we have tried so far.

```
autoplot(coal_train) +autolayer(coal_test, series = "Actual")+ autolayer(forecast(fit2,h=60), se
ries = "ARIMA")+xlab("DateTime") +  ylab("Net Electricity Generated (Million kWh)") + ggtitle("A
RIMA Model vs Actual Values")
```

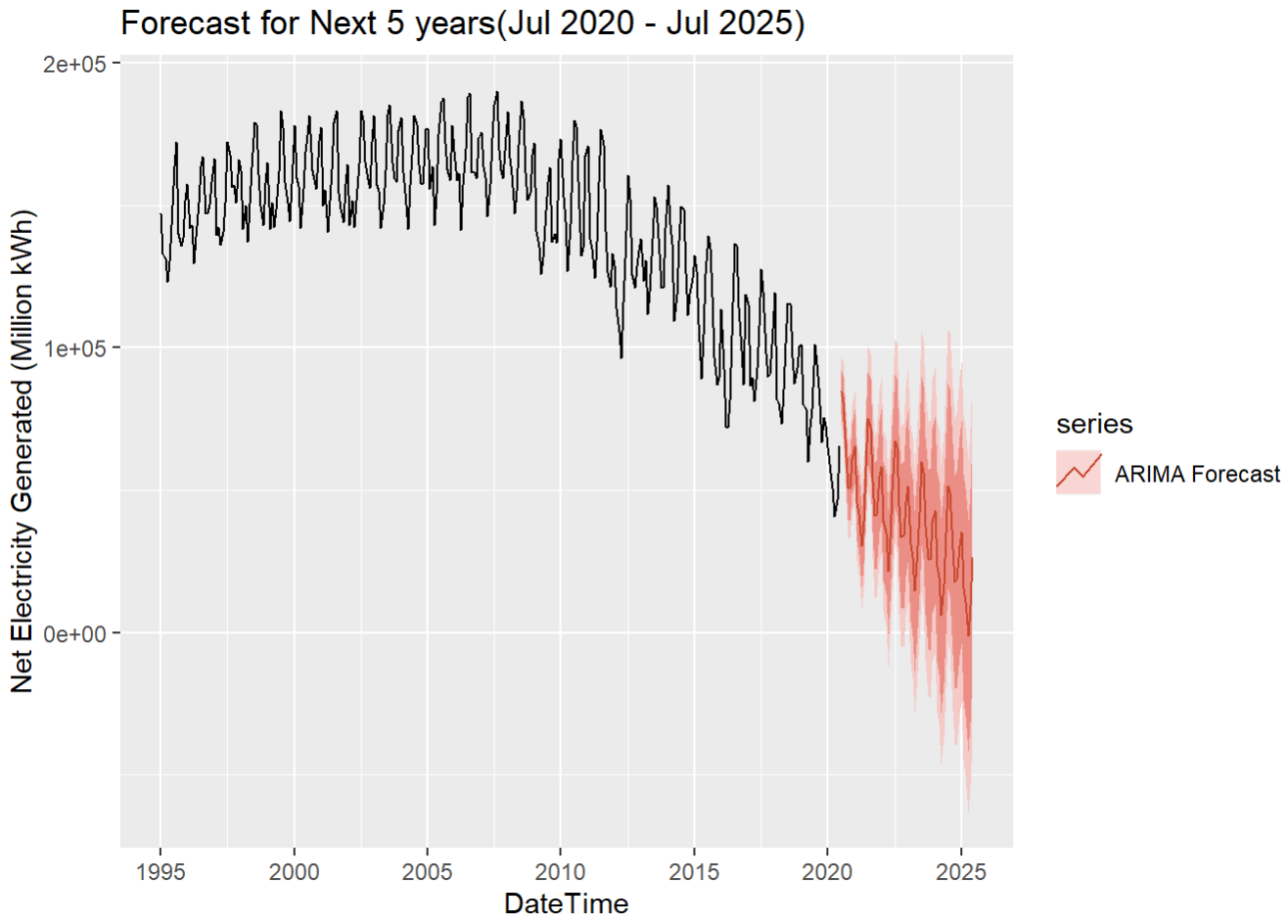


we can see from the above graph, the arima model has a good enough prediction interval that covers most of the actual values. This shows that the selected arima model is a good one.

Let's use this selected arima model on the entire dataset and forecast the Net Electricity Generation using Coal for the next 5yrs in the future.

```
best <- arima(coal,order = c(3,1,3),seasonal = c(1,1,2))
f.best <- forecast(best,h=60)

autoplot(coal) + autolayer(f.best, series = "ARIMA Forecast") +
  xlab("DateTime") +
  ylab("Net Electricity Generated (Million kWh)") +
  ggtitle("Forecast for Next 5 years(Jul 2020 - Jul 2025)")
```




```
f.best
```

```
##          Point Forecast       Lo 80    Hi 80       Lo 95     Hi 95
## Jul 2020      84499.130  76820.9189 92177.34  72756.3195  96241.94
## Aug 2020      80831.298  71378.5282 90284.07  66374.5343  95288.06
## Sep 2020      61645.591  51264.7353 72026.45  45769.4424  77521.74
## Oct 2020      50546.618  39432.2849 61660.95  33548.7124  67544.52
## Nov 2020      51073.995  39292.1881 62855.80  33055.2763  69092.71
## Dec 2020      61116.093  48795.0756 73437.11  42272.7228  79959.46
## Jan 2021      65441.423  52551.4775 78331.37  45727.9524  85154.89
## Feb 2021      45639.858  32223.9665 59055.75  25122.0224  66157.69
## Mar 2021      41729.535  27843.6997 55615.37  20492.9825  62966.09
## Apr 2021      30270.454  15870.1800 44670.73   8247.1355  52293.77
## May 2021      38938.469  24088.6180 53788.32  16227.5817  61649.36
## Jun 2021      58345.348  43053.6998 73637.00  34958.7903  81731.91
## Jul 2021      75211.438  58821.6194 91601.26  50145.3735 100277.50
## Aug 2021      71808.877  54650.5743 88967.18  45567.5177  98050.24
## Sep 2021      53136.500  35290.9976 70982.00  25844.1592  80428.84
## Oct 2021      40979.818  22493.5052 59466.13  12707.4424  69252.19
## Nov 2021      41315.993  22256.4225 60375.56  12166.8956  70465.09
## Dec 2021      54805.010  35155.7817 74454.24  24754.1086  84855.91
## Jan 2022      58400.418  38198.2945 78602.54  27503.9363  89296.90
## Feb 2022      39170.351  18442.6661 59898.04   7470.0923  70870.61
## Mar 2022      34931.193  13659.1034 56203.28   2398.3394  67464.05
## Apr 2022      21381.074   -390.3702 43152.52 -11915.4767  54677.63
## May 2022      33304.239  11033.4219 55575.06   -756.0366  67364.51
## Jun 2022      49919.540  27145.8533 72693.23  15090.1916  84748.89
## Jul 2022      67139.582  43699.4568 90579.71  31291.0039 102988.16
## Aug 2022      65114.107  41056.3337 89171.88  28320.9180 101907.30
## Sep 2022      44557.406  19936.7871 69178.03   6903.4186  82211.39
## Oct 2022      33675.813   8530.3698 58821.26  -4780.8241  72132.45
## Nov 2022      34323.382   8635.7313 60011.03  -4962.4895  73609.25
## Dec 2022      45840.538  19652.0088 72029.07   5788.6389  85892.44
## Jan 2023      51269.330  24582.6695 77955.99  10455.6050  92083.05
## Feb 2023      31072.069   3879.6703 58264.47 -10515.1159  72659.25
## Mar 2023      26379.843  -1279.2095 54038.89 -15921.0272  68680.71
## Apr 2023      14676.920 -13464.3371 42818.18 -28361.4189  57715.26
## May 2023      24529.373  -4083.0298 53141.78 -19229.5208  68288.27
## Jun 2023      42195.847  13138.2892 71253.40  -2243.8527  86635.55
## Jul 2023      60060.631  30322.2365 89799.03  14579.6811 105541.58
## Aug 2023      56255.052  25935.8792 86574.22   9885.8789 102624.22
## Sep 2023      37428.627   6573.6860 68283.57  -9759.9331  84617.19
## Oct 2023      25942.755  -5451.5228 57337.03 -22070.6494  73956.16
## Nov 2023      25680.856  -6215.0510 57576.76 -23099.7244  74461.44
## Dec 2023      39066.503   6668.4087 71464.60 -10482.1067  88615.11
## Jan 2024      42934.521  10032.9930 75836.05  -7384.0238  93253.07
## Feb 2024      23074.444 -10300.0899 56448.98 -27967.5012  74116.39
## Mar 2024      19454.622 -14406.8771 53316.12 -32332.0719  71241.32
## Apr 2024       5927.121 -28408.0973 40262.34 -46584.0641  58438.31
## May 2024      17198.565 -17592.4876 51989.62 -36009.7583  70406.89
## Jun 2024      34728.170   -534.1634 69990.50 -19200.9150  88657.25
## Jul 2024      51378.339  15482.2826 87274.39  -3519.9416 106276.62
## Aug 2024      49334.521  12867.0886 85801.95  -6437.6041 105106.65
## Sep 2024      29322.765  -7691.7662 66337.30 -27286.0756  85931.61
## Oct 2024      17711.632 -19812.7955 55236.06 -39677.0278  75100.29
```

```
## Nov 2024      18788.817 -19252.9259 56830.56 -39391.0082  76968.64
## Dec 2024      30477.692  -8067.3100 69022.69 -28471.8016  89427.19
## Jan 2025      35316.478  -3714.8311 74347.79 -24376.7582  95009.72
## Feb 2025      15801.336 -23728.3820 55331.05 -44654.1506  76256.82
## Mar 2025      10760.455 -29245.9372 50766.85 -50424.0420  71944.95
## Apr 2025      -1175.532 -41655.6971 39304.63 -63084.6019  60733.54
## May 2025       9337.205 -31622.3432 50296.75 -53305.0182  71979.43
## Jun 2025      26330.779 -15085.1139 67746.67 -37009.3632  89670.92
```

The final selected arima model has projected electricity generation using coal to decrease even more rapidly in the next 5 years.