HuanHusted_TSA_Competition

Kelsey Husted & Yu Huan

2023-04-28

Time Series Competition

```
knitr::opts_chunk$set(echo = TRUE,tidy.opts=list(width.cutoff=80), tidy=FALSE)
```

##Load packages

```
library(readxl)
library(dplyr)
library(lubridate)
library(openxlsx)
library(ggplot2)
library(forecast)
library(Kendall)
library(tseries)
library(outliers)
library(tidyverse)
library(smooth)
library(zoo)
library(kableExtra)
```

Data wrangling and processing from 2005-2009

Hourly data was transformed into daily data with aggregate functions and pipes (i.e., tidyverse).

```
#Import data
df <- read_xlsx("./Competition/Data/load.xlsx")

#Wrangle data from hourly to daily
#Wrangling date column 2005 to 2009
df_daily <- df %>%
  mutate( Date = ymd(date)) %>%
  filter(Date < '2010-01-01')

#removing no numeric columns so rowMeans() function will work
df_dailyV2 <- df %>%
  mutate( Date = ymd(date)) %>%
  filter(Date < '2010-01-01') %>%
  select(3:26)

#Creating daily data
df_processed <- df_dailyV2 %>%
  mutate(rowMeans(df_dailyV2)) %>%
  rename(Daily_data = "rowMeans(df_dailyV2)") %>%
```

```
  select(25)

#Combining date and daily data
date <- df_daily[,2]
df_processed <- cbind(date, df_processed)

nobs = nrow(df_daily)
```

Data wrangling and processing from 2005-2010

The data needs to be formatted to include 2010 as well since the objective is to forecast for 2011. Instead of making two separate datasets, I should use the window() function for future reference.

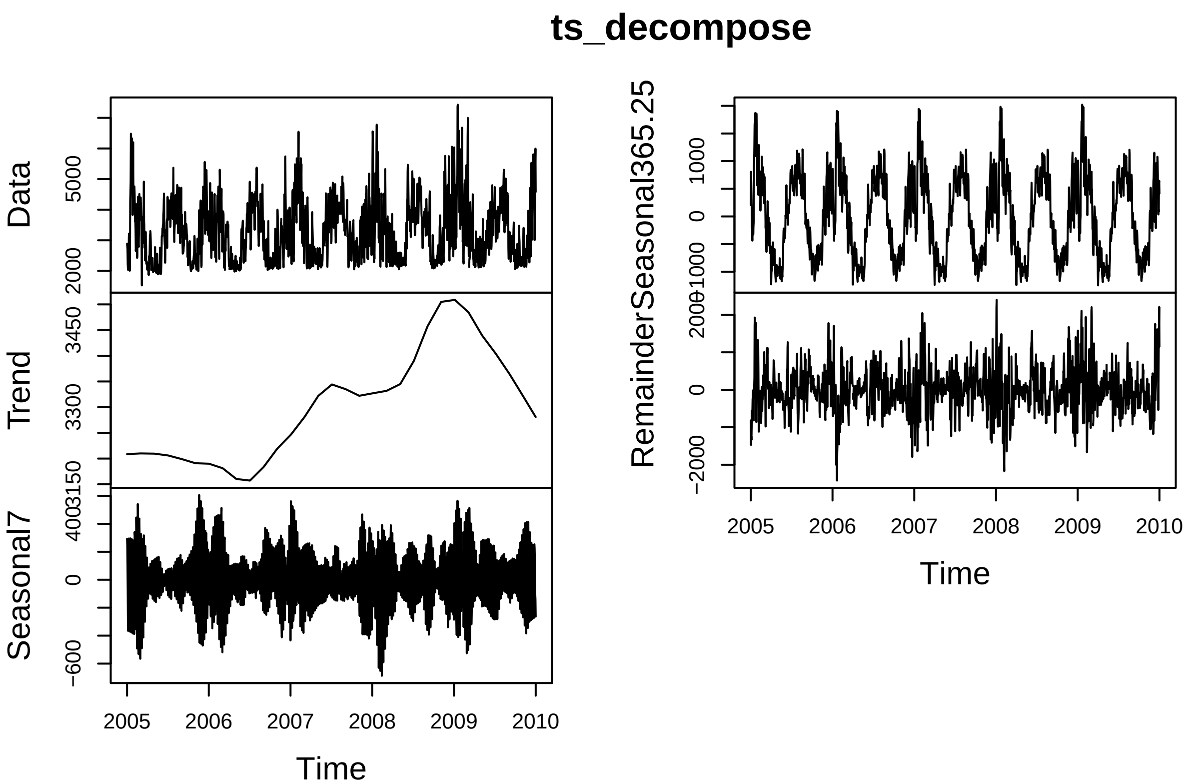
```
#Wrangle data from hourly to daily
#Wrangling date column 2005 to 2010
#removing no numeric columns so rowMeans() function will work
df_daily2010 <- df %>%
  mutate( Date = ymd(date)) %>%
  select(3:26)

#Creating daily data
df_processed2010 <- df_daily2010 %>%
  mutate(rowMeans(df_daily2010)) %>%
  rename(Daily_data = "rowMeans(df_daily2010)") %>%
  select(25)

#Combining data and daily data
date <- df[,2]
df_processed2010 <- cbind(date, df_processed2010)

nobs2010 = nrow(df_processed2010)
```

Time series object transformation

```
#ts transformation 2005 to 2009
ts_daily <- msts(df_processed$Daily_data,
                 seasonal.periods=c(7,365.25),
                 start=c(2005, 01, 01))

#ts transformation 2005 to 2010
ts_daily2010 <- msts(df_processed2010$Daily_data,
                 seasonal.periods=c(7,365.25),
                 start=c(2005, 01, 01))

#ts for accuracy test visalization

ts_daily2010_test <- msts(df_processed2010$Daily_data[1:1885],
                 seasonal.periods=c(7,365.25),
                 start=c(2005, 01, 01))
```

Time series decomposition and plot

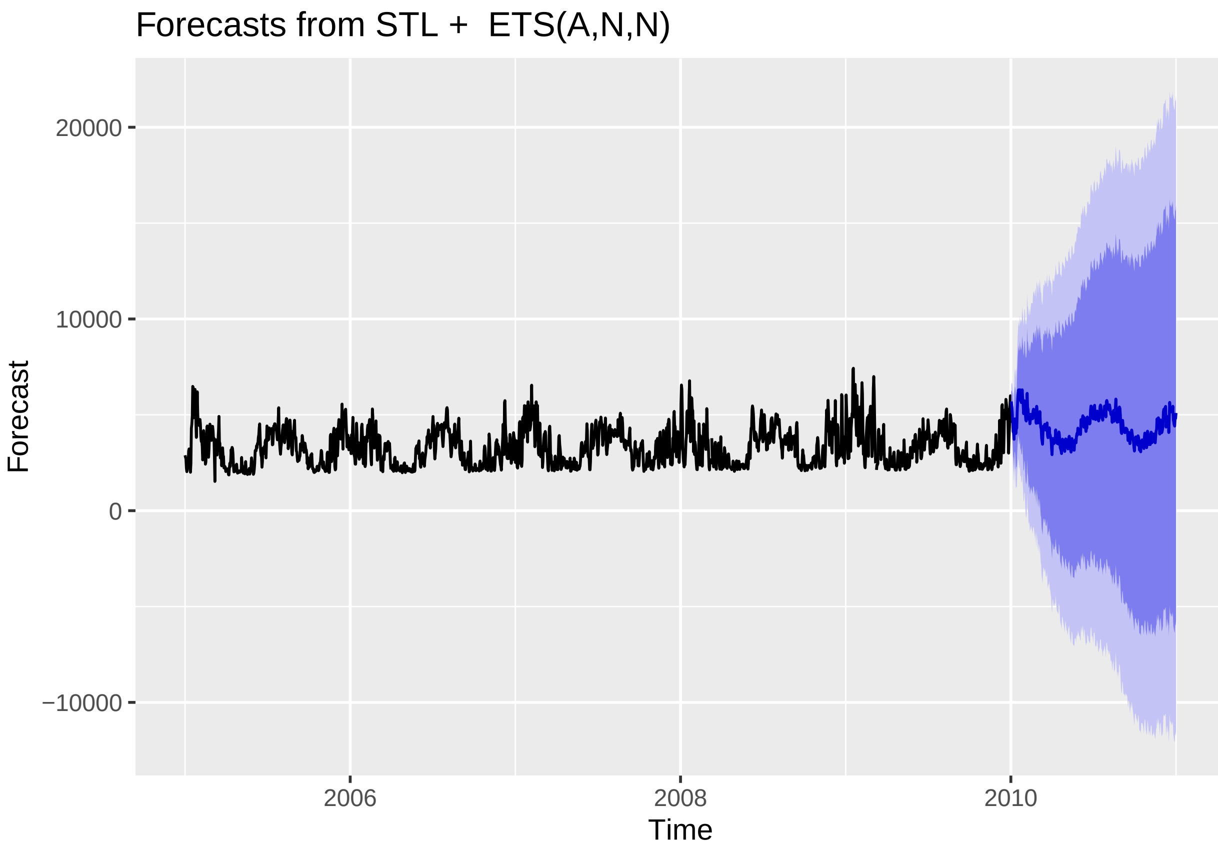
```
#Decompose time series
ts_decompose <- ts_daily %>%
  mstl()
plot(ts_decompose)
```



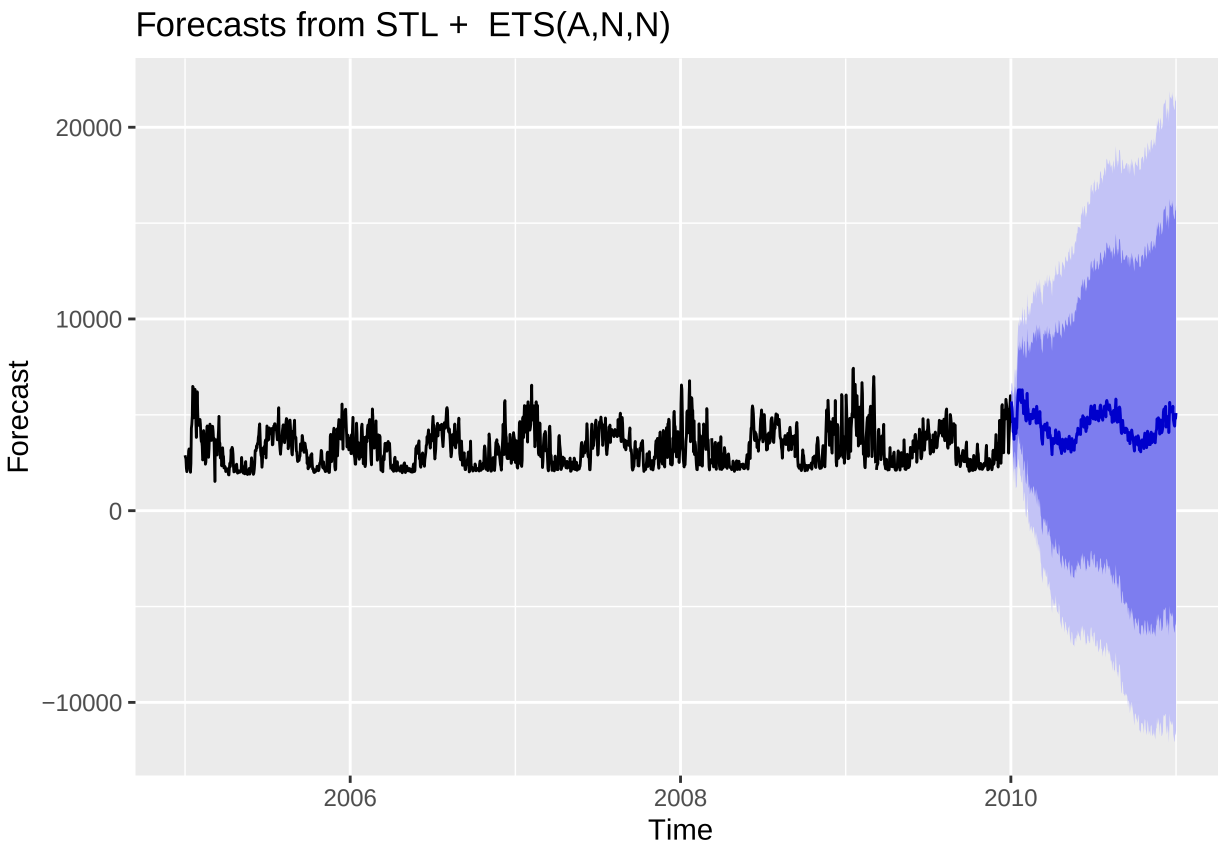
##

Model 1 STL + ETS: Forecast 2010

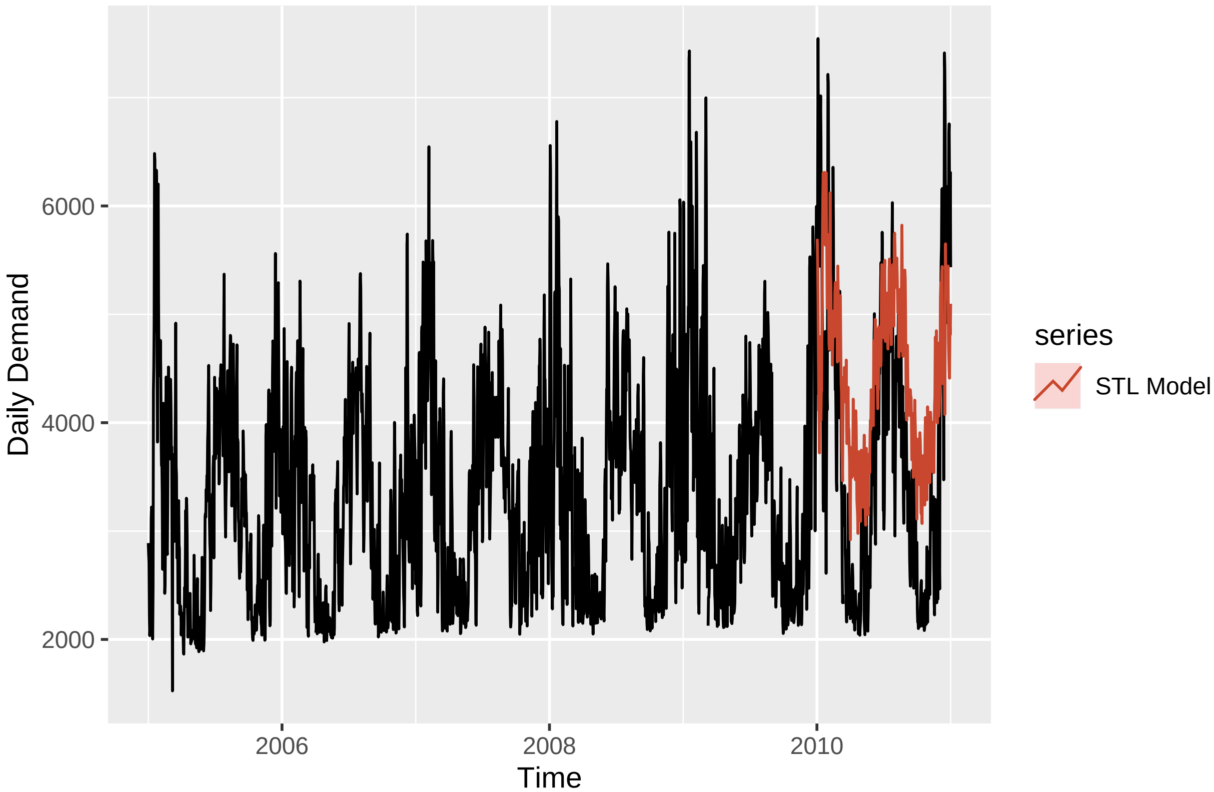
```
#Fit and forecast STL model in 2010
ETS_model <-  stlf(ts_daily,h=365)

# Forcast just first two month of 2010
ETS_model_month <-  stlf(ts_daily,h=365)

#Plot foresting
autoplot(ETS_model) + ylab("Forecast")
```



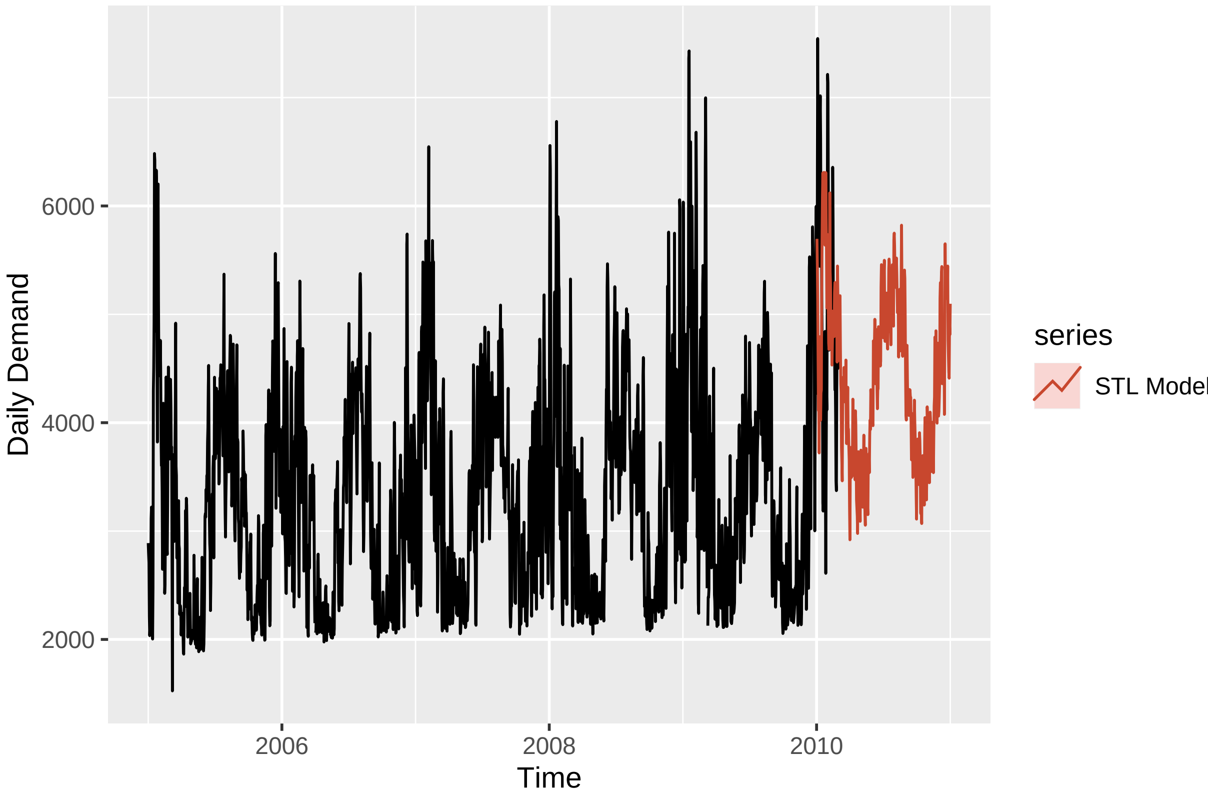

```
#Plot foresting
autoplot(ETS_model_month) + ylab("Forecast")
```



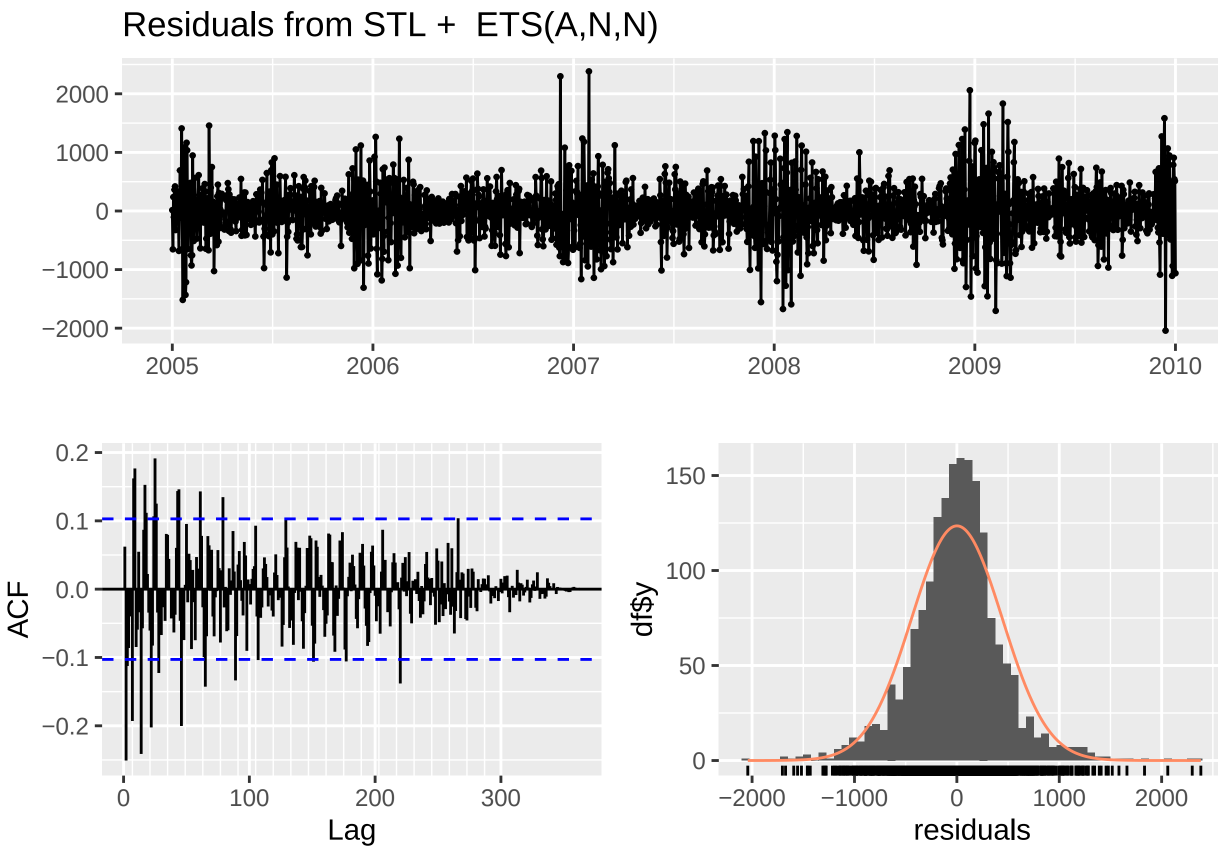

```
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(ETS_model, series="STL Model",PI=FALSE) +
  ylab("Daily Demand")
```




```
#Plot model + observed data
autoplot(ts_daily2010_test) +
  autolayer(ETS_model_month, series="STL Model",PI=FALSE) +
  ylab("Daily Demand")
```




```
#Plot the residuals

checkresiduals(ETS_model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 1309.7, df = 363, p-value < 2.2e-16
##
## Model df: 2.   Total lags used: 365
```

```
#Check accuracy of model
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
ETS_scores <- accuracy(ETS_model$mean,observed)
print(ETS_scores)
```

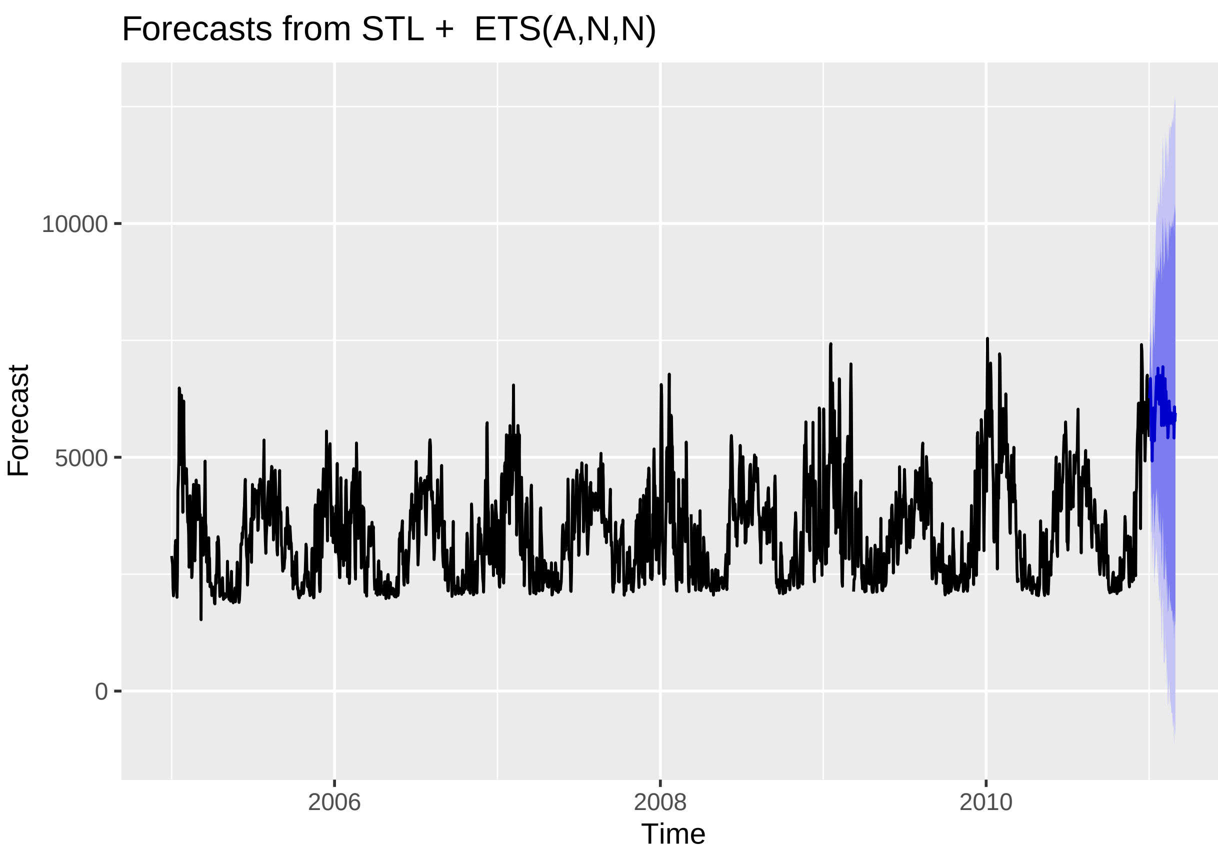
```
##                   ME     RMSE      MAE       MPE     MAPE
## Test set -984.5201 1210.625 1079.332 -35.24753 36.80666
```

```
#Check accuracy of the model using just the first two month of 2010
n_for <- 59
observed <- df_processed2010[1827:1885, "Daily_data"]
ETS_scores_for <- accuracy(ETS_model_month$mean,observed)
print(ETS_scores_for)
```

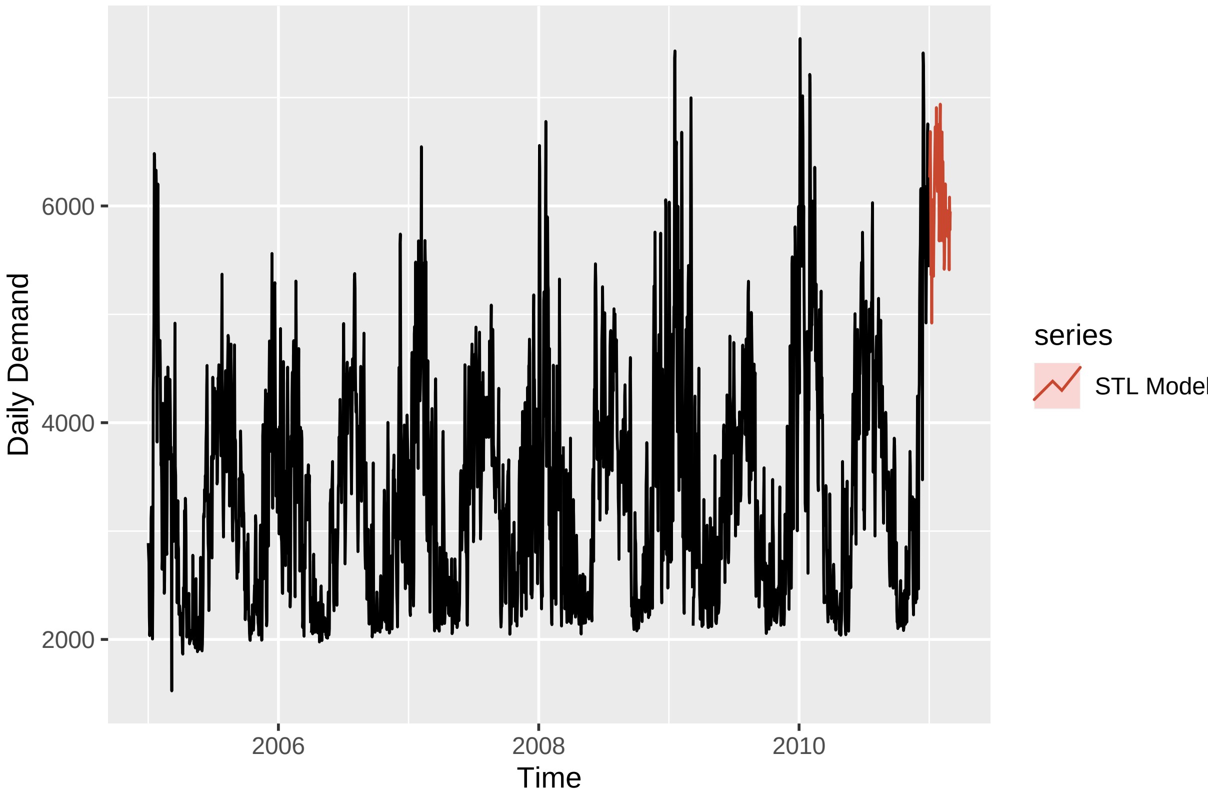
```
##                   ME     RMSE      MAE       MPE     MAPE
## Test set -103.0303 1547.348 1280.656 -9.574082 28.27524
```

Model 1 STL + ETS: Forecast 2011

```
#Fit and forecast STL model January 1st to February 28th 2011
ETS_model2011 <-  stlf(ts_daily2010,h=59)

#Plot foresting
autoplot(ETS_model2011) + ylab("Forecast")
```



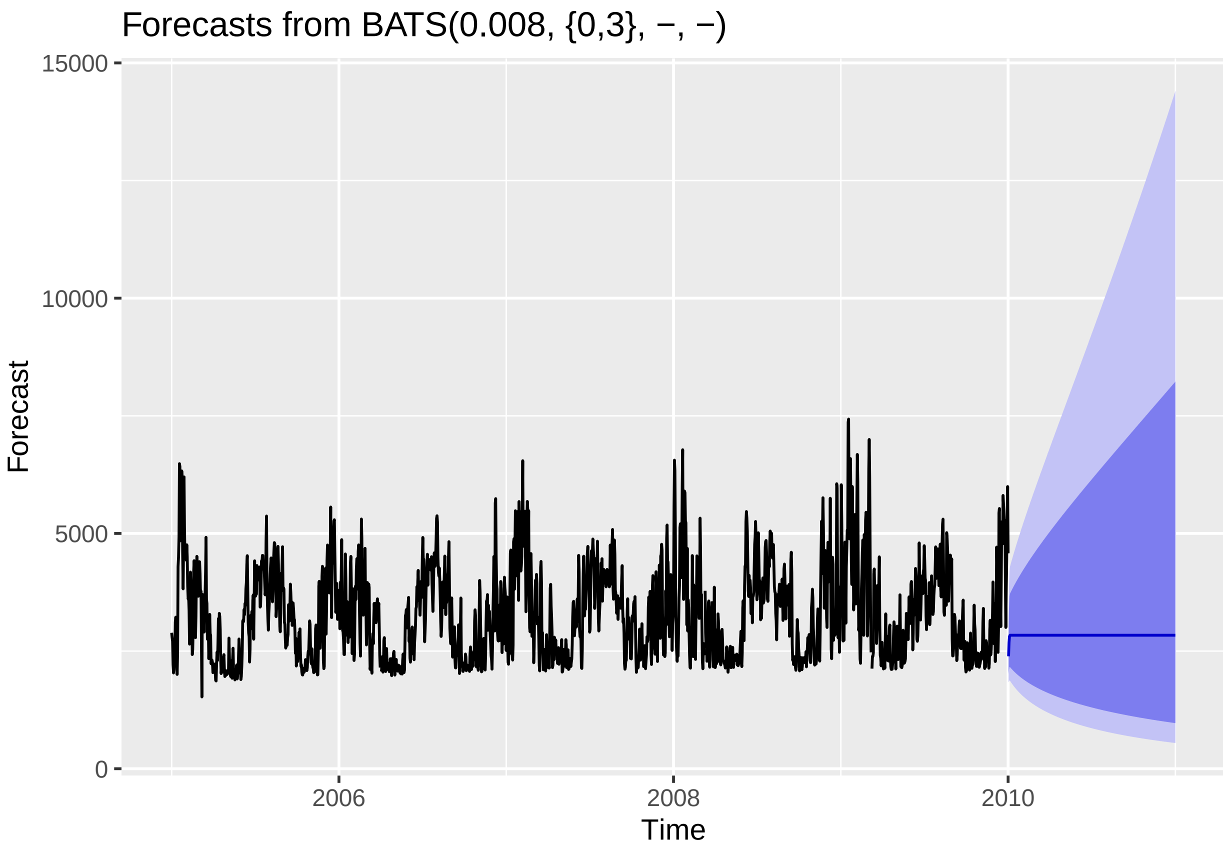

```
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(ETS_model2011, series="STL Model",PI=FALSE) +
  ylab("Daily Demand")
```



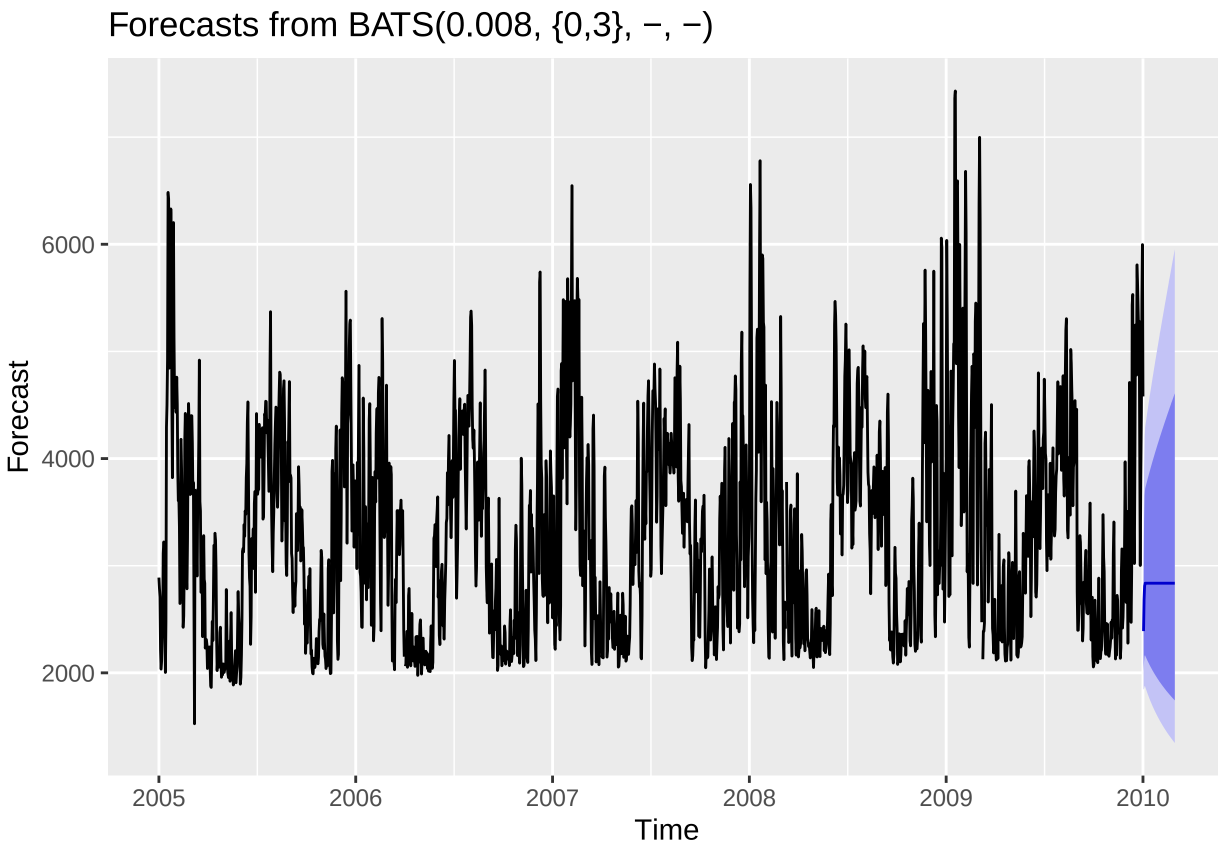
Model 2 TBATS: Forecast 2010

The model looks like a really bad fit visually and will not be used to forecast for 2011.

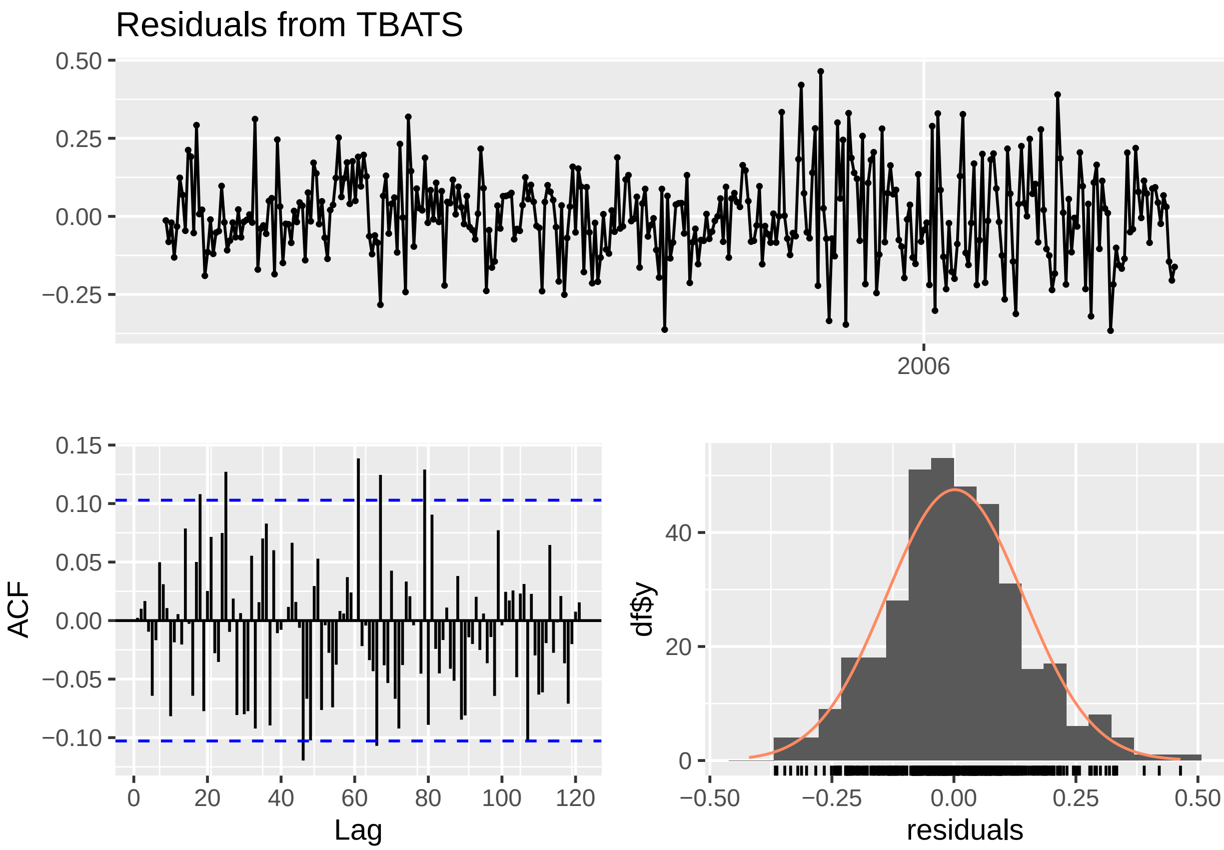
```
#Fit and forecast TBATS model
TBATS_model <-  tbats(ts_daily)

#forecast 2010
TBATS_for <- forecast(TBATS_model,h=365)

#forecast just first two month in 2010
TBATS_for_month <- forecast(TBATS_model,h=59)


#Plot foresting
autoplot(TBATS_for) + ylab("Forecast")
```




```
#Plot foresting
autoplot(TBATS_for_month) + ylab("Forecast")
```

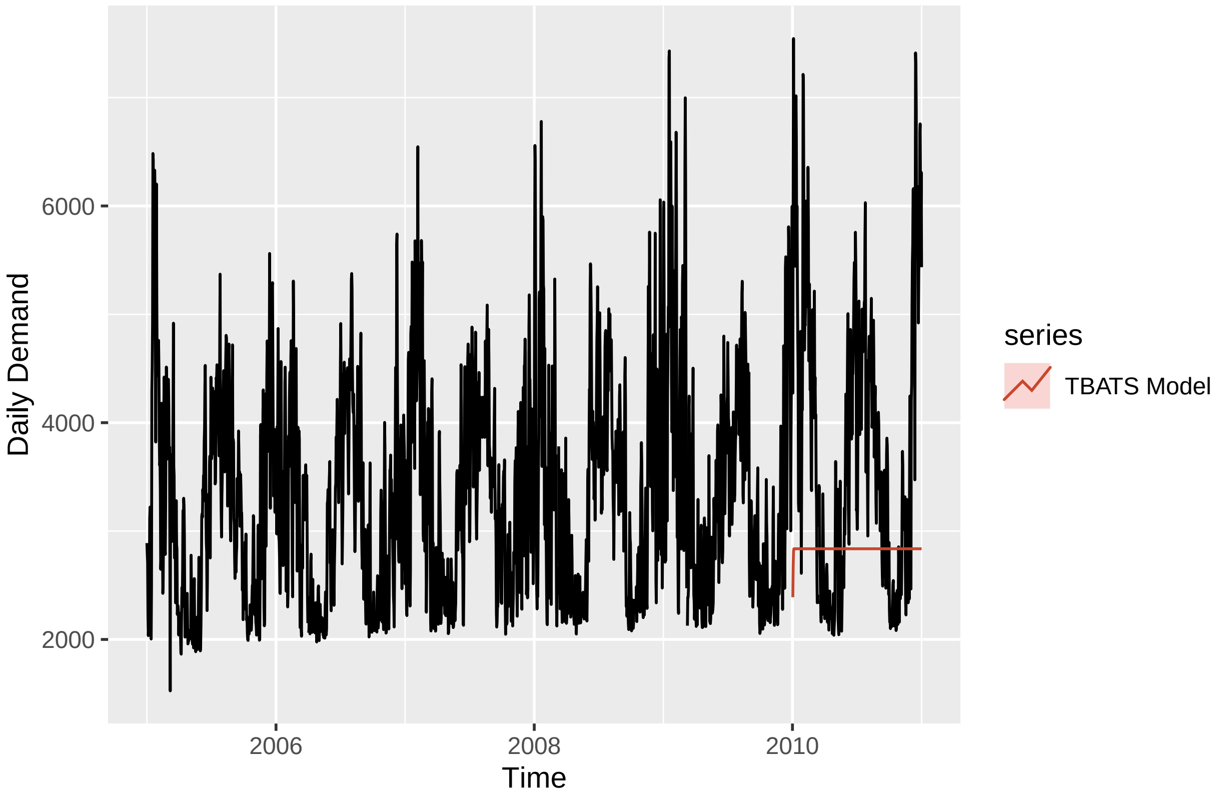



```
#Plot the residuals

checkresiduals(TBATS_model)
```

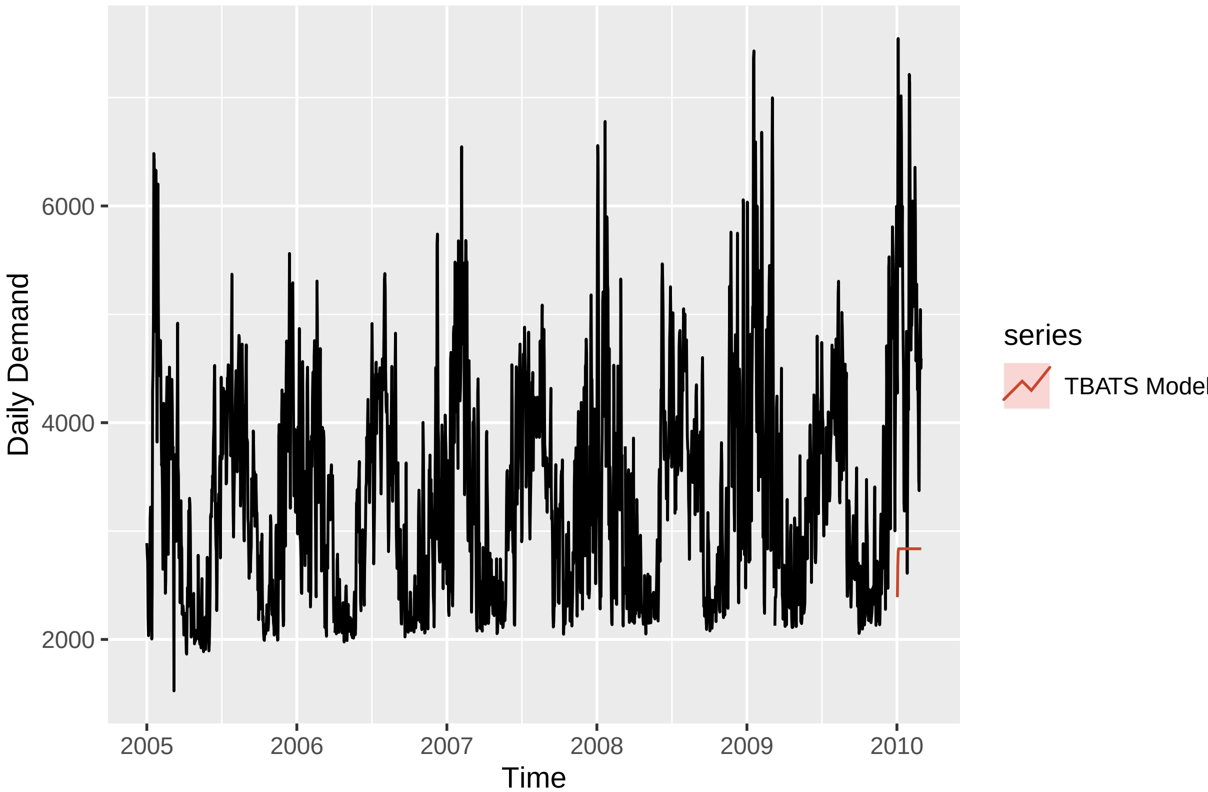



```
##
##  Ljung-Box test
##
## data:  Residuals from TBATS
## Q* = 105.04, df = 68, p-value = 0.002646
##
## Model df: 5.   Total lags used: 73
```

```
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(TBATS_for, series="TBATS Model",PI=FALSE) +
  ylab("Daily Demand")
```




```r
#Plot model + observed data
autoplot(ts_daily2010_test) +
  autolayer(TBATS_for_month, series="TBATS Model",PI=FALSE) +
  ylab("Daily Demand")
```




```
#Check accuracy of model
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
TBATS_scores <- accuracy(TBATS_for$mean,observed)
print(TBATS_scores)
```

```
##                ME     RMSE      MAE      MPE    MAPE
## Test set 613.9492 1258.164 921.0355 10.07795 23.5087
```

```
#Check accuracy of the model using just the first two month of 2010
n_for <- 59
observed <- df_processed2010[1827:1885, "Daily_data"]
TBATS_scores_for <- accuracy(TBATS_for_month$mean,observed)
print(TBATS_scores_for)
```

```
##               ME     RMSE      MAE      MPE     MAPE
## Test set 2226.34 2499.097 2234.019 40.94677 41.24095
```

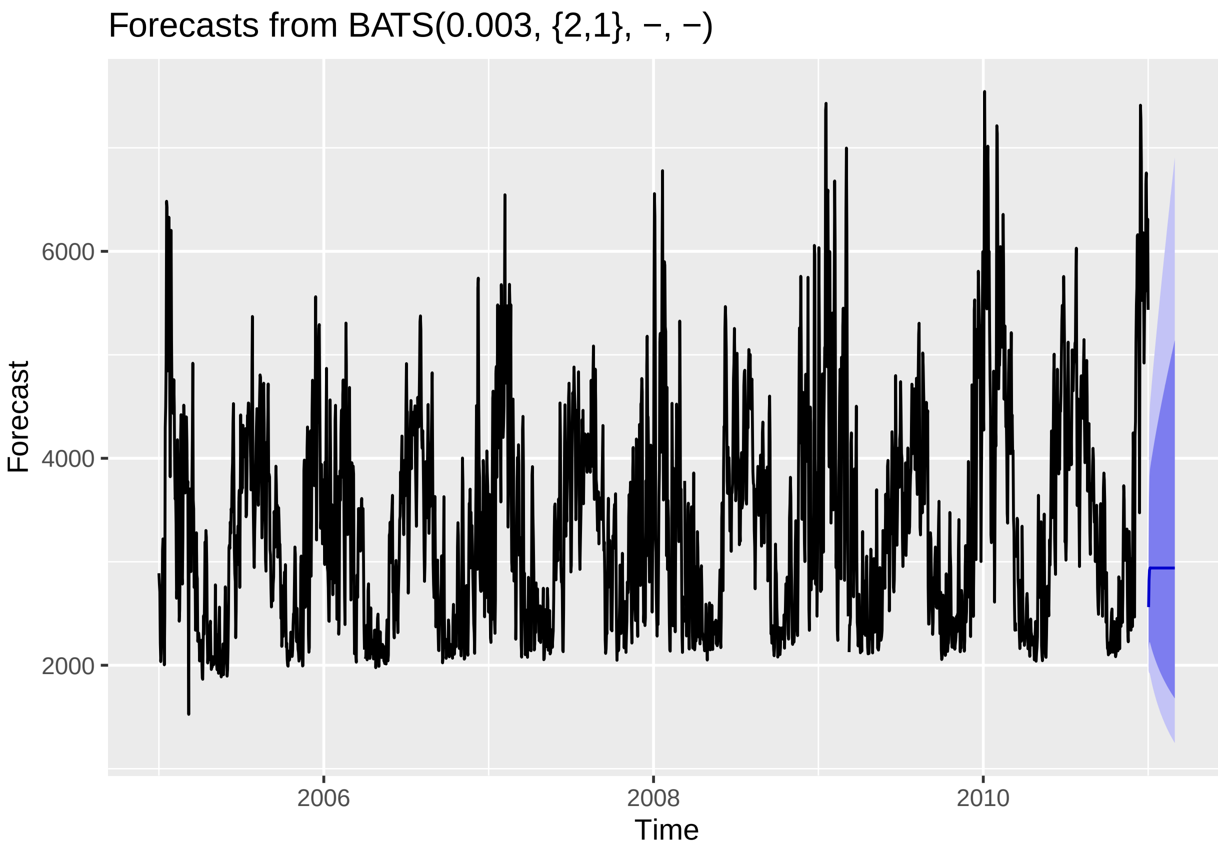
Model 2 TBATS: Forecast 2011

```
#Fit and forecast TBATS model
TBATS_model2011 <-  tbats(ts_daily2010)
```

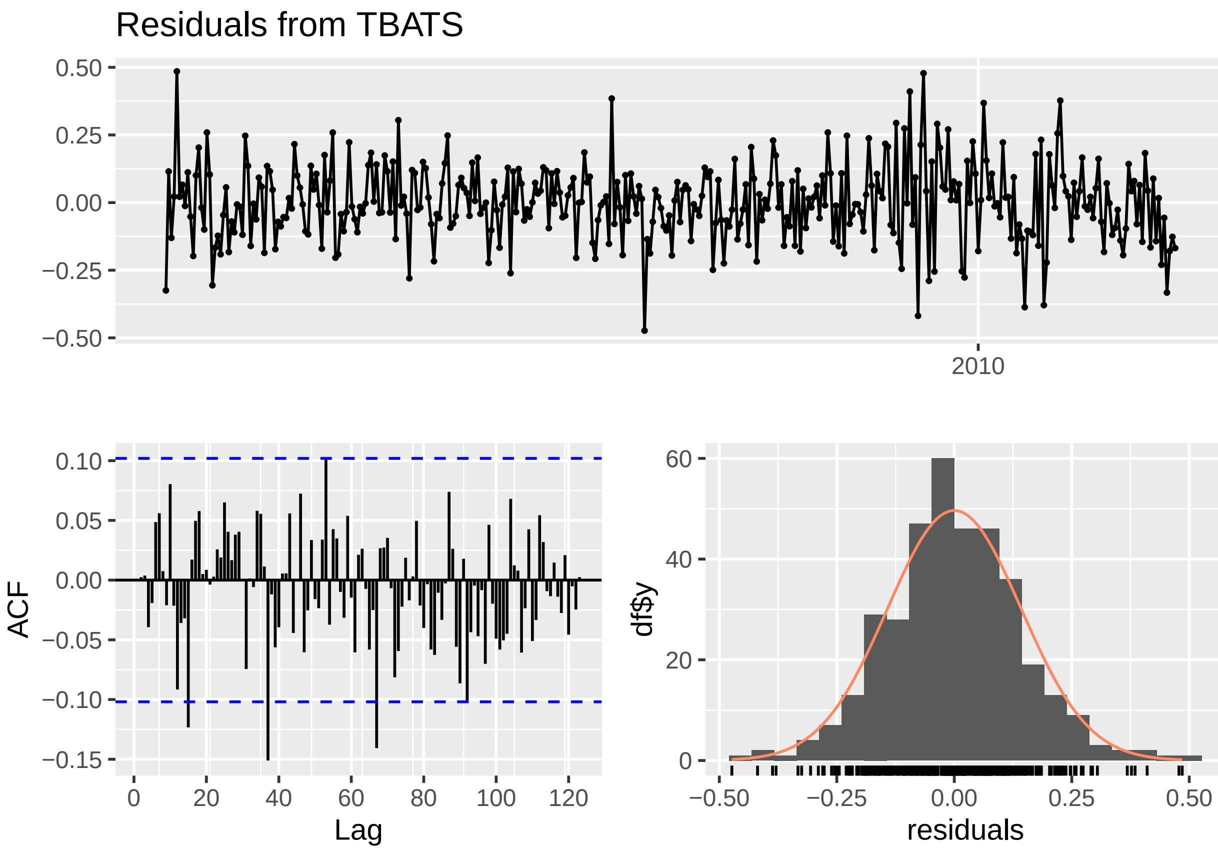
```
## Warning in tbats(ts_daily2010): Missing values encountered. Using longest
## contiguous portion of time series
```

```
#forecast
TBATS_for2011 <- forecast(TBATS_model2011,h=59)

#Plot foresting
```

```
autoplot(TBATS_for2011) + ylab("Forecast")
```

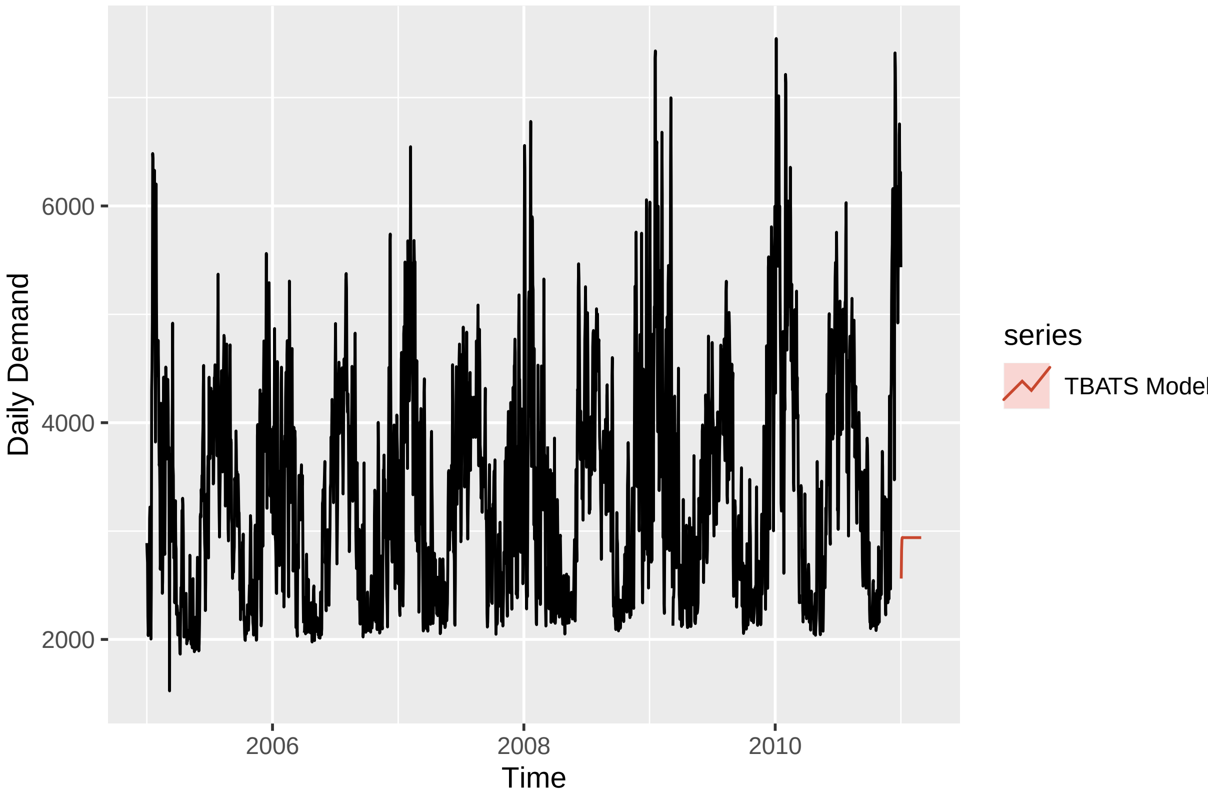



```
#Plot the residuals

checkresiduals(TBATS_model2011)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from TBATS
## Q* = 73.551, df = 69, p-value = 0.3315
##
## Model df: 5.   Total lags used: 74
```

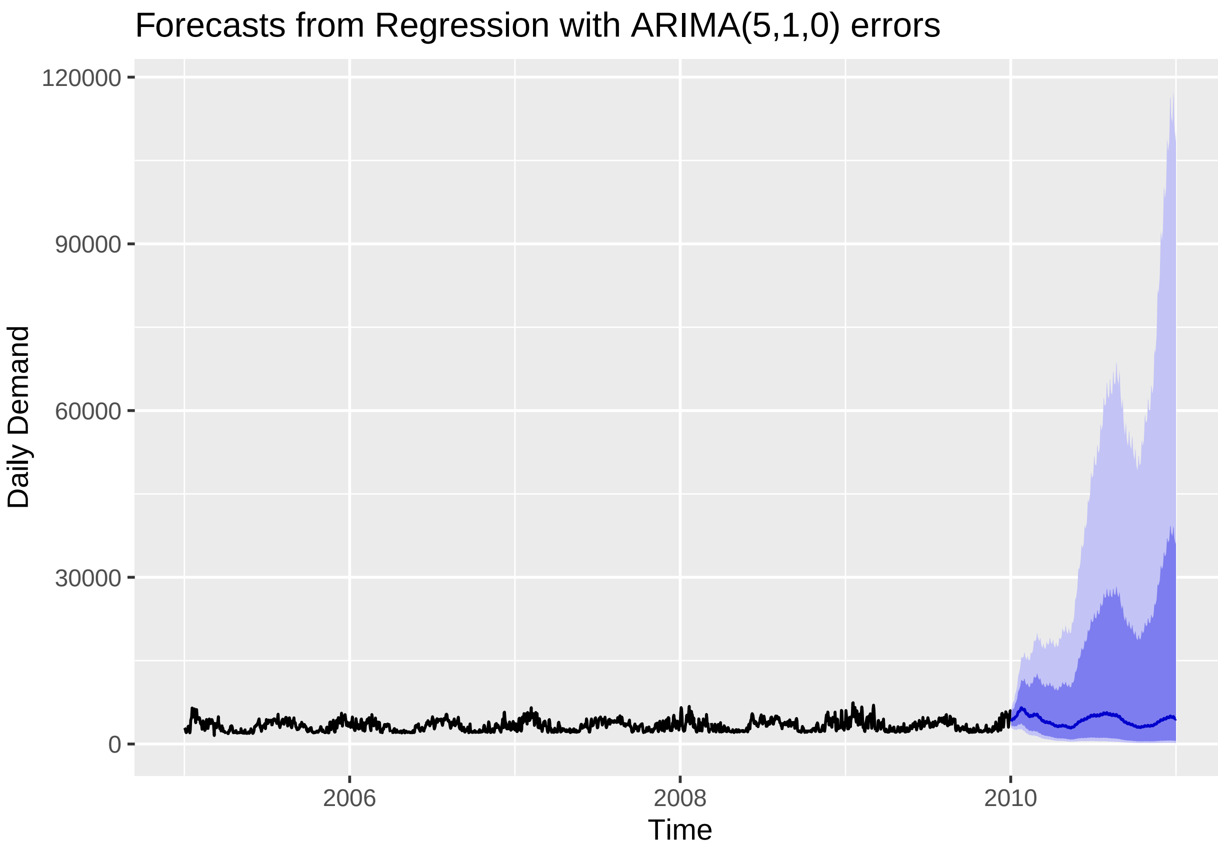
```
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(TBATS_for2011, series="TBATS Model",PI=FALSE) +
  ylab("Daily Demand")
```



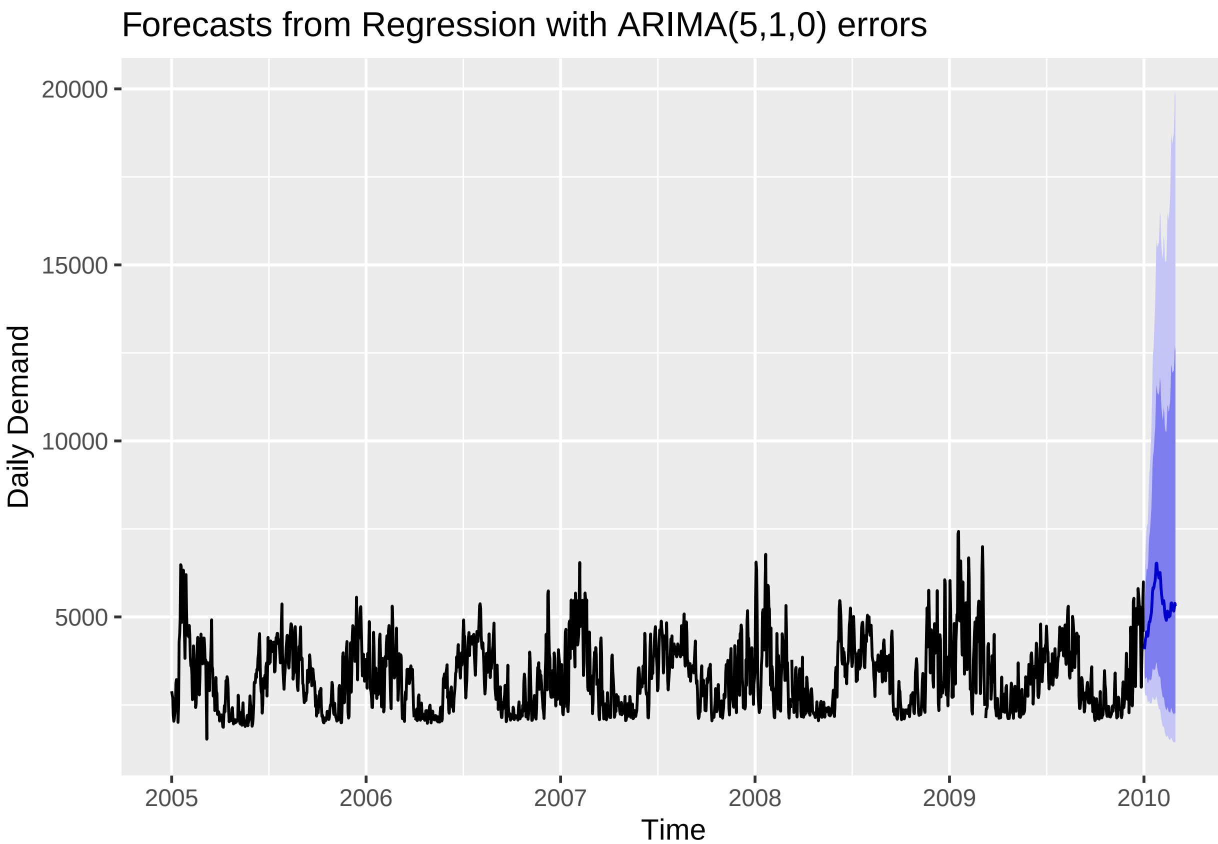
Model 3 ARIMA + FOURIER terms: Forecast 2010

```
#Fit and forecast TBATS model
ARIMA_Four_model <- auto.arima(ts_daily,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_daily,
                                          K=c(2,12))
                             )

#Forecast in 2010
ARIMA_Four_for <- forecast(ARIMA_Four_model,
                           xreg=fourier(ts_daily,
                                        K=c(2,12),
                                        h=365),
                           h=365
                           )

#Forecast just first two month in 2010
ARIMA_Four_for_month <- forecast(ARIMA_Four_model,
                           xreg=fourier(ts_daily,
                                        K=c(2,12),
                                        h=59),
                           h=59
                           )


#Plot foresting results
```

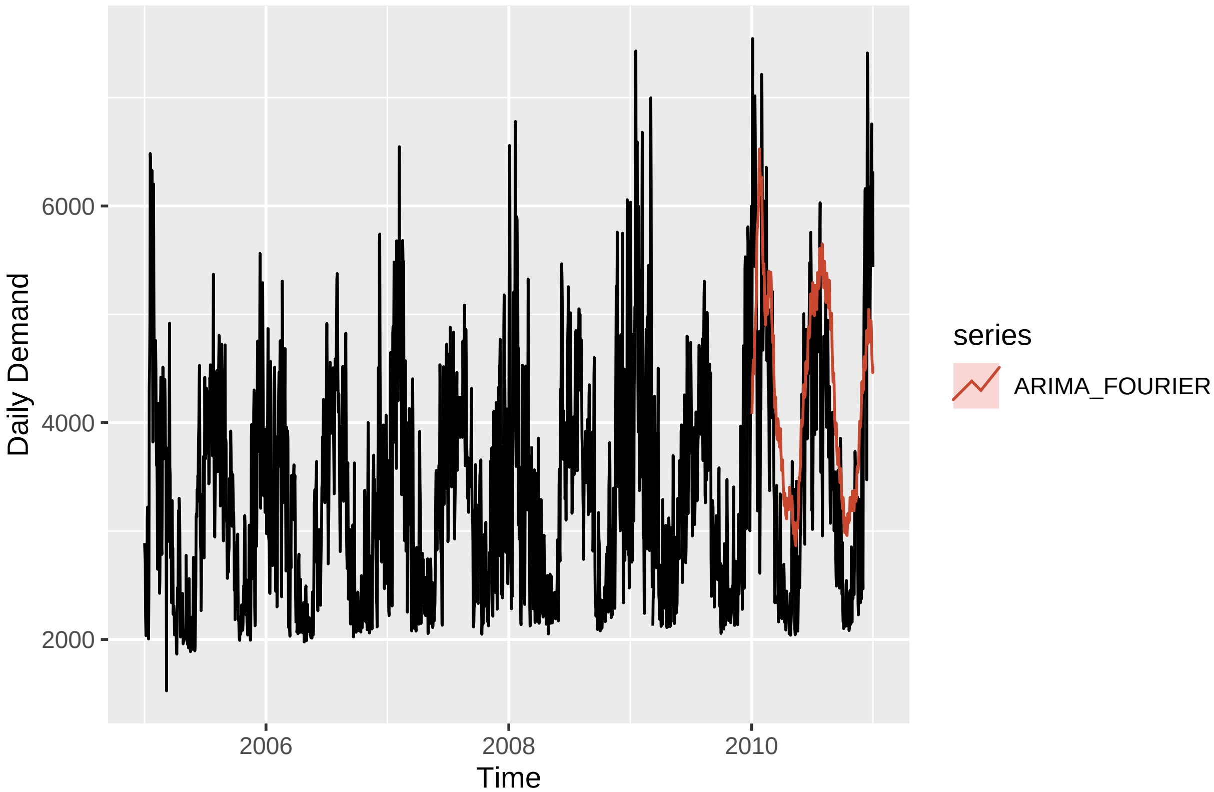
```r
autoplot(ARIMA_Four_for) + ylab("Daily Demand")
```



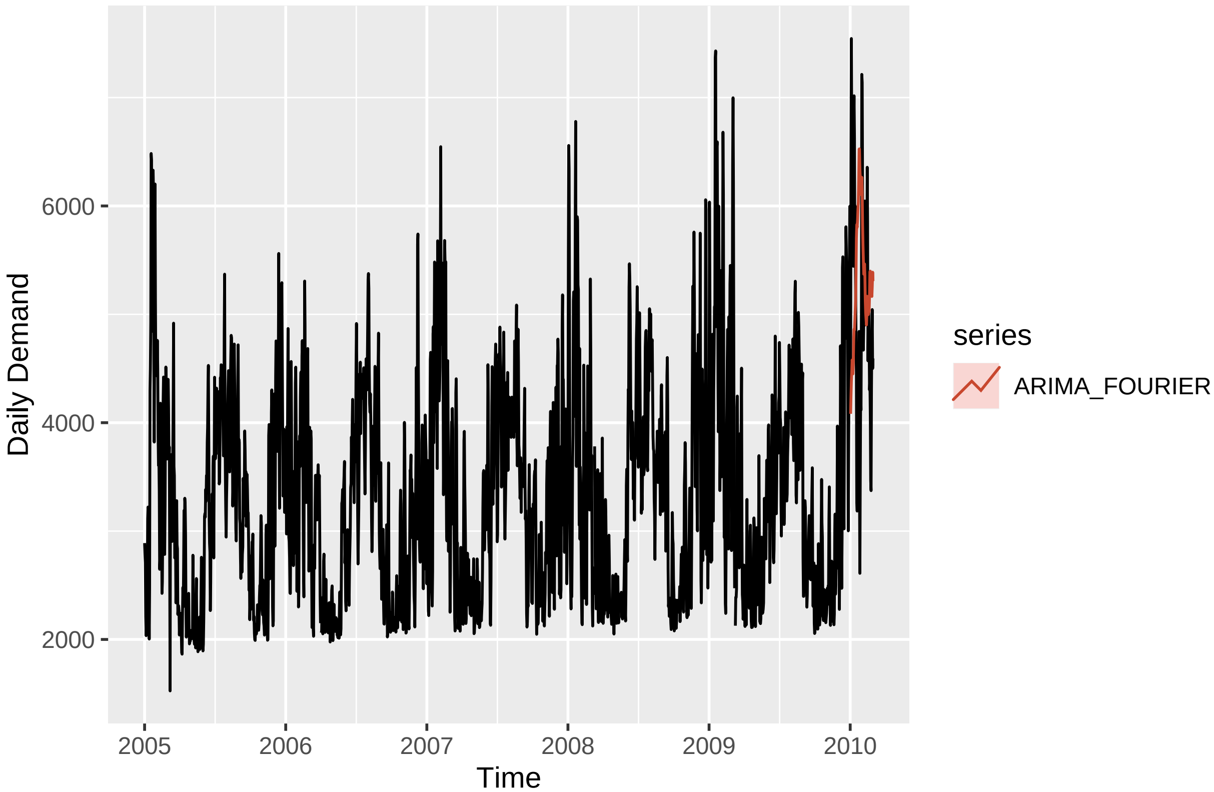

```r
#Plot foresting results
autoplot(ARIMA_Four_for_month) + ylab("Daily Demand")
```



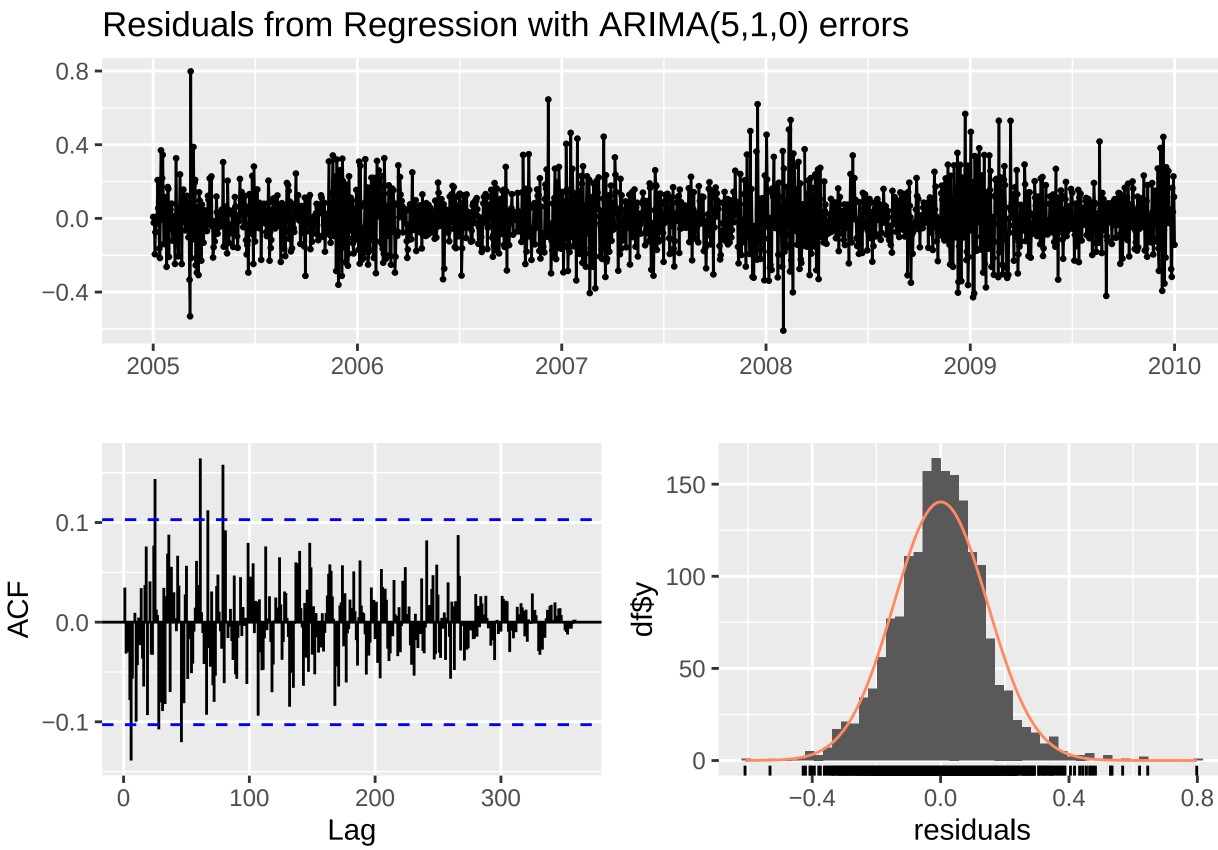

```
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(ARIMA_Four_for, series="ARIMA_FOURIER",PI=FALSE) +
  ylab("Daily Demand")
```




```
#Plot model + observed data
autoplot(ts_daily2010_test) +
  autolayer(ARIMA_Four_for_month, series="ARIMA_FOURIER",PI=FALSE) +
  ylab("Daily Demand")
```




```
# Plot the residuals

checkresiduals(ARIMA_Four_model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(5,1,0) errors
## Q* = 462.43, df = 360, p-value = 0.0002078
##
## Model df: 5.   Total lags used: 365
```

```
#Check accuracy of model
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
ARIMA_Four_scores <- accuracy(ARIMA_Four_for$mean,observed)
print(ARIMA_Four_scores)
```

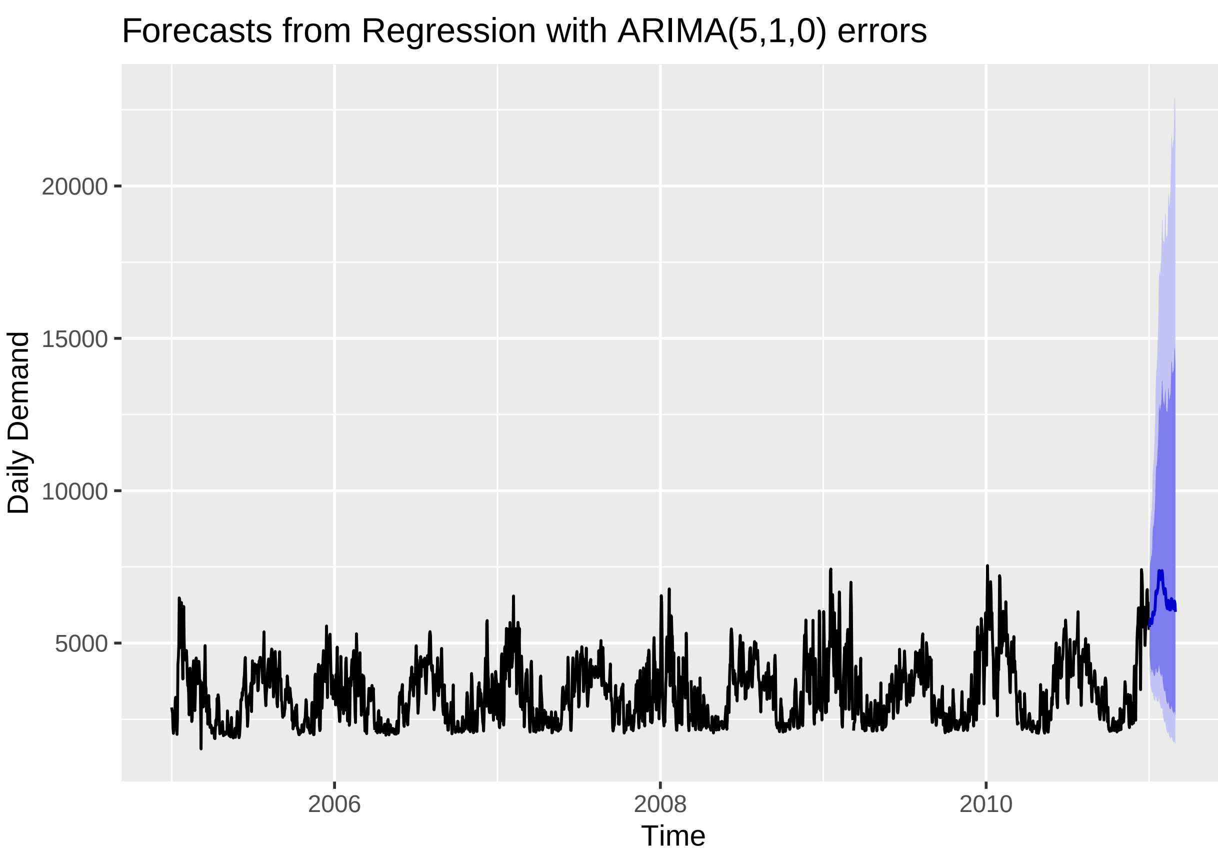
```
##                 ME     RMSE     MAE      MPE     MAPE
## Test set -853.8232 1158.956 991.685 -30.4006 32.76586
```

```
#Check accuracy of the model using just the first two month of 2010
n_for <- 59
observed <- df_processed2010[1827:1885, "Daily_data"]
ARIMA_Four_scores_for <- accuracy(ARIMA_Four_for_month$mean,observed)
print(ARIMA_Four_scores_for)
```

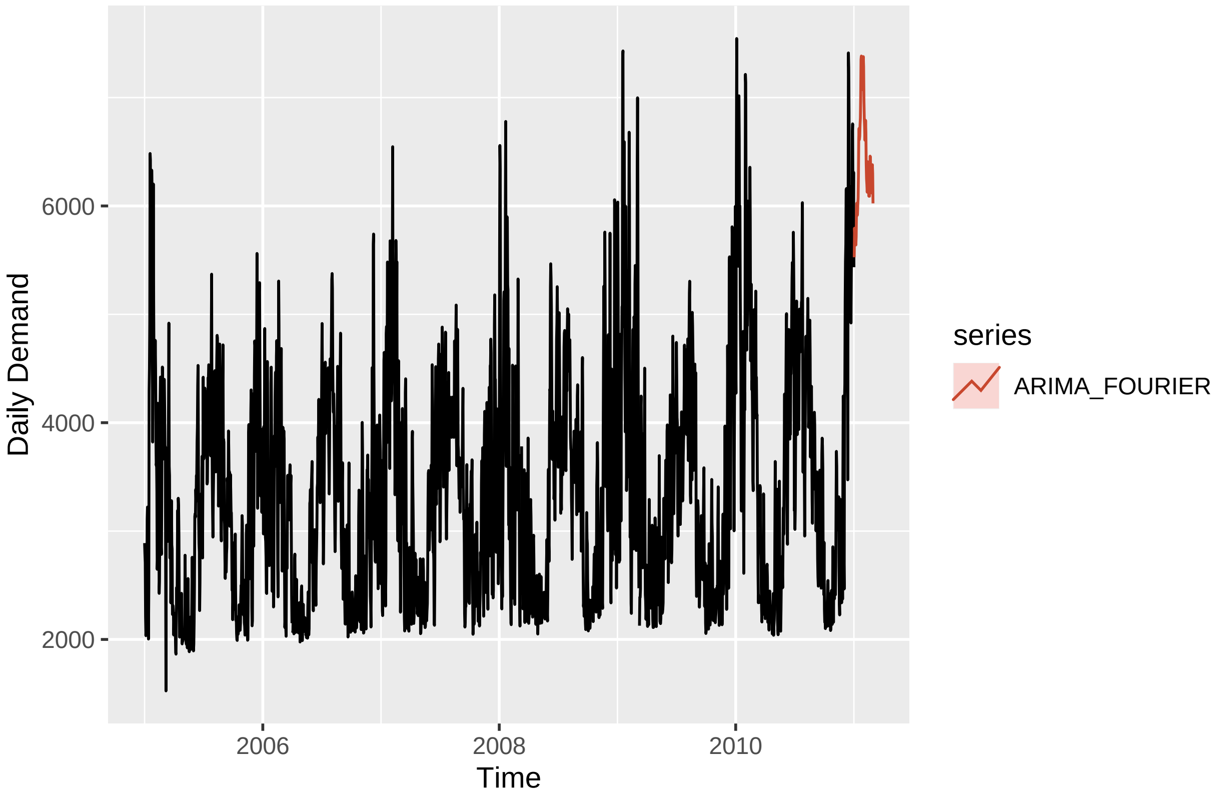
```
##                 ME     RMSE      MAE       MPE    MAPE
## Test set -284.3608 1536.398 1243.755 -13.07091 28.0451
```

Model 3 ARIMA + FOURIER terms: Forecast 2011

```
#Fit and forecast TBATS model
ARIMA_Four_model2011 <- auto.arima(ts_daily2010,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_daily2010,
                                          K=c(2,12))
                             )

#Forecast
ARIMA_Four_for2011 <- forecast(ARIMA_Four_model2011,
                           xreg=fourier(ts_daily2010,
                                        K=c(2,12),
                                        h=59),
                           h=59
                           )

#Plot foresting results
autoplot(ARIMA_Four_for2011) + ylab("Daily Demand")
```

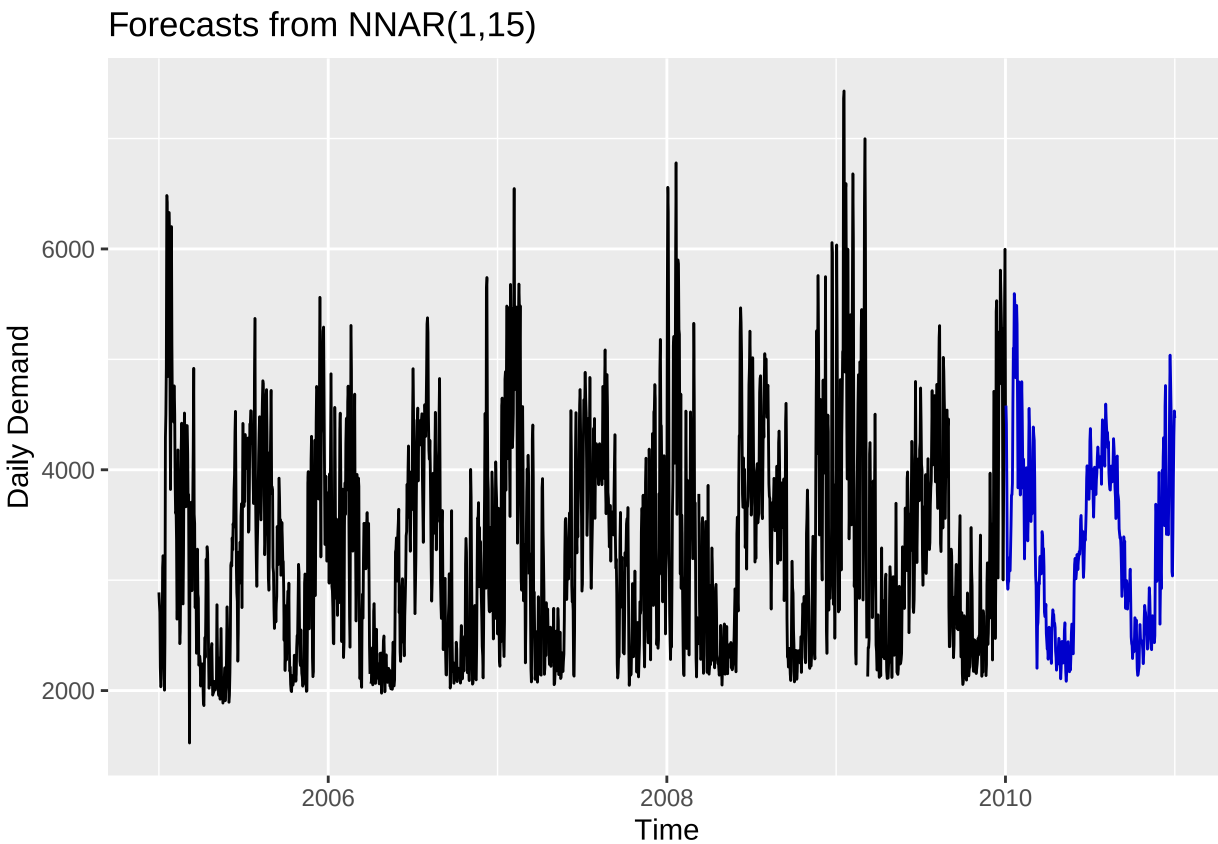



```
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(ARIMA_Four_for2011, series="ARIMA_FOURIER",PI=FALSE) +
  ylab("Daily Demand")
```

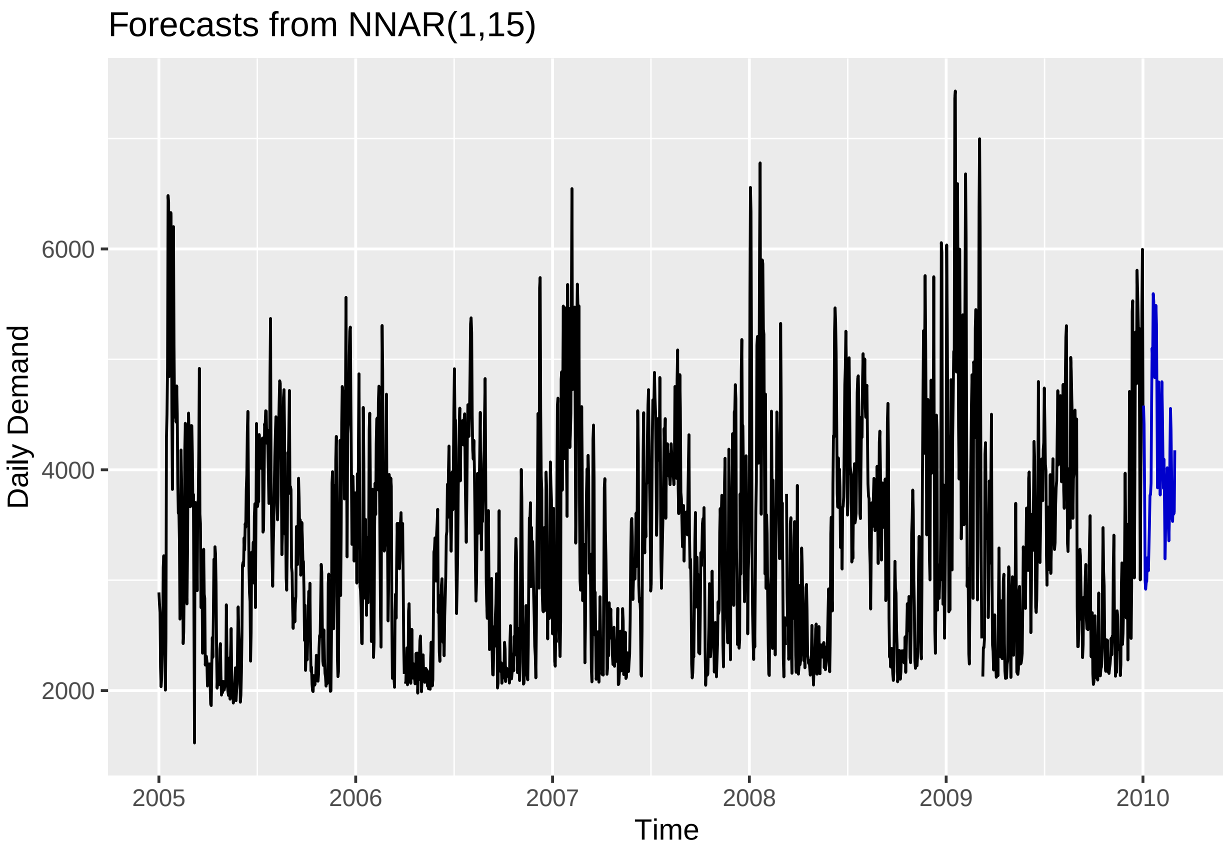


Model 4 Neural Network Time Series: Forecasts 2010

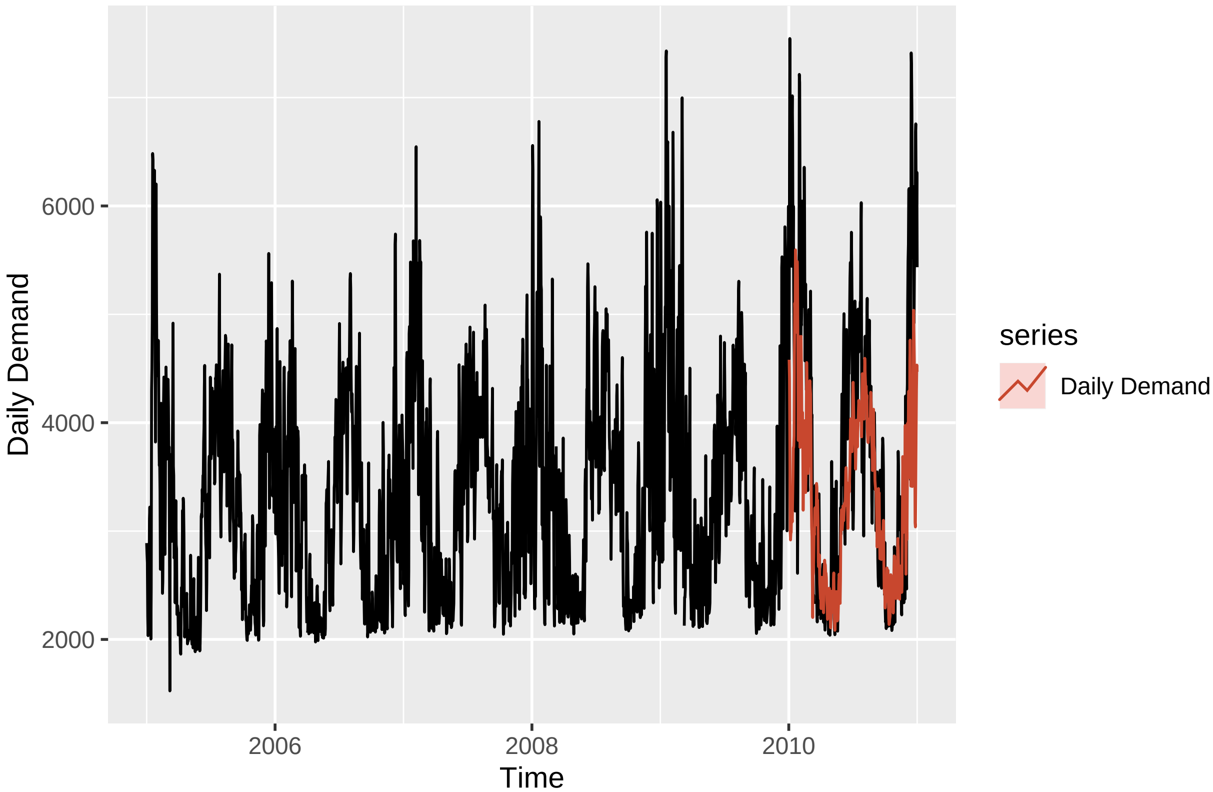
```
#NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
NN_model <- nnetar(ts_daily,decay=0.5, maxit=150, p=1,P=0,xreg=fourier(ts_daily, K=c(2,12)))

#NN_for <- forecast(NN_fit, h=365)
NN_for <- forecast(NN_model, h=365,xreg=fourier(ts_daily,
                                          K=c(2,12),h=365))
#NN_for <- forecast(NN_fit, h=365)
NN_for_month <- forecast(NN_model, h=59,xreg=fourier(ts_daily,
                                          K=c(2,12),h=59))

#Plot foresting results
autoplot(NN_for) +
  ylab("Daily Demand")
```



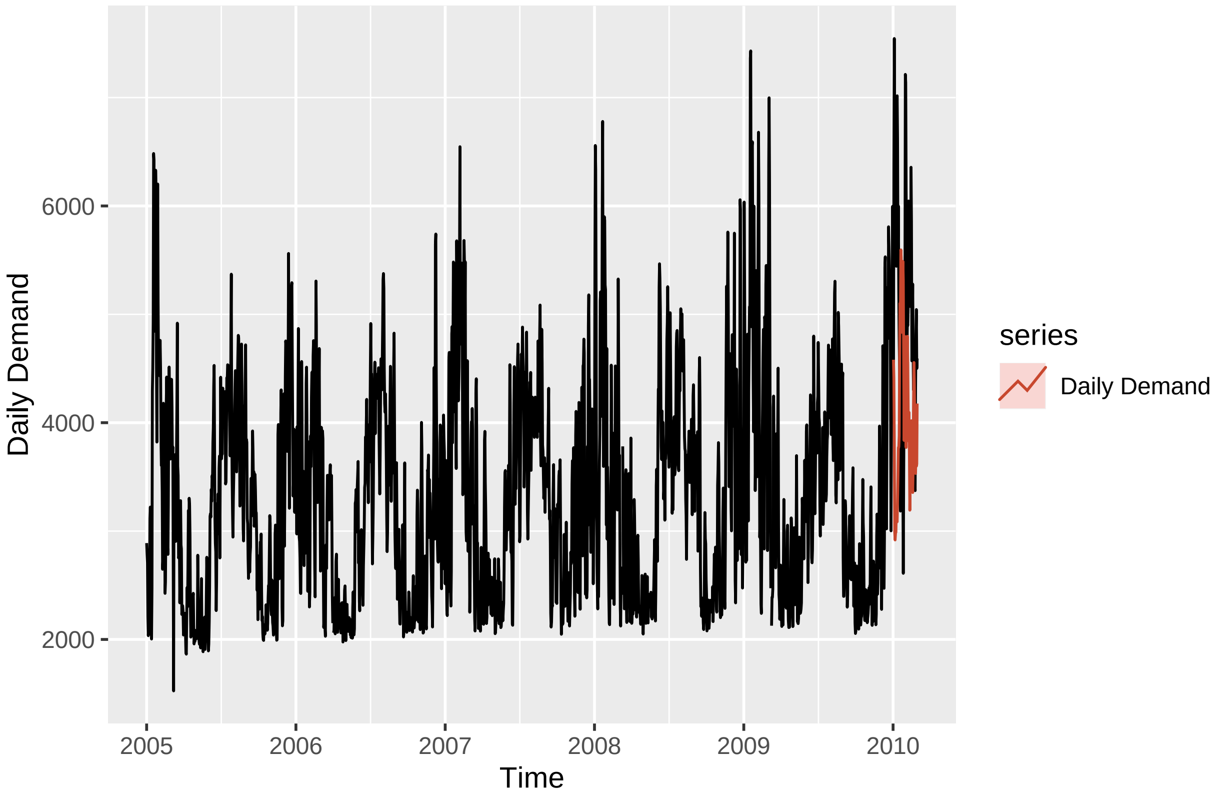

```r
#Plot foresting results
autoplot(NN_for_month) +
  ylab("Daily Demand")
```



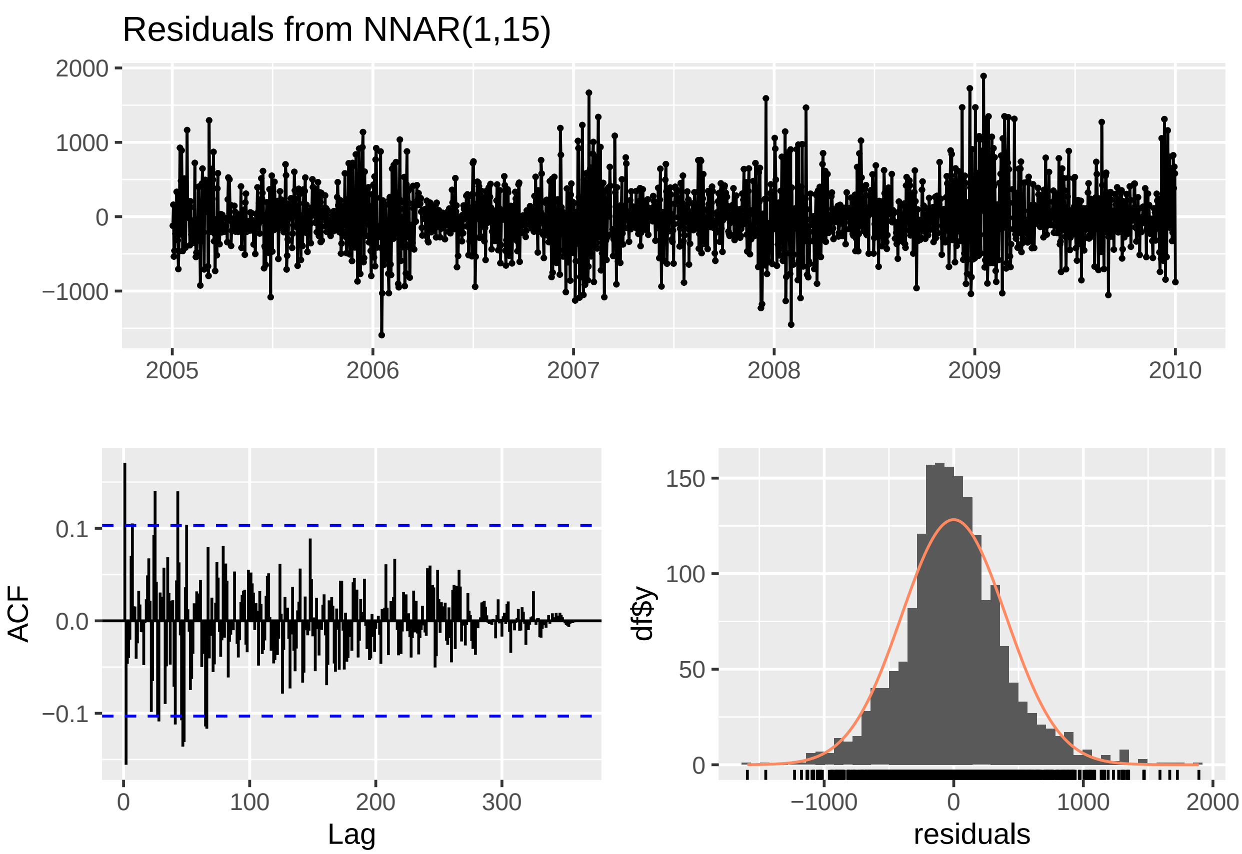

```r
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(NN_for, series="Daily Demand",PI=FALSE)+
  ylab("Daily Demand")
```




```r
#Plot model + observed data using just first two month
autoplot(ts_daily2010_test) +
  autolayer(NN_for_month, series="Daily Demand",PI=FALSE)+
  ylab("Daily Demand")
```




```r
# Check residuals
checkresiduals(NN_model)
```



```r
#Checking error variables to decide which model fits the data the best
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
NN_scores1 <- accuracy(NN_for$mean,observed)
print(NN_scores1)
```

```
##                 ME     RMSE      MAE       MPE     MAPE
## Test set 129.7843 714.3135 531.6442 0.2470648 14.81317
```

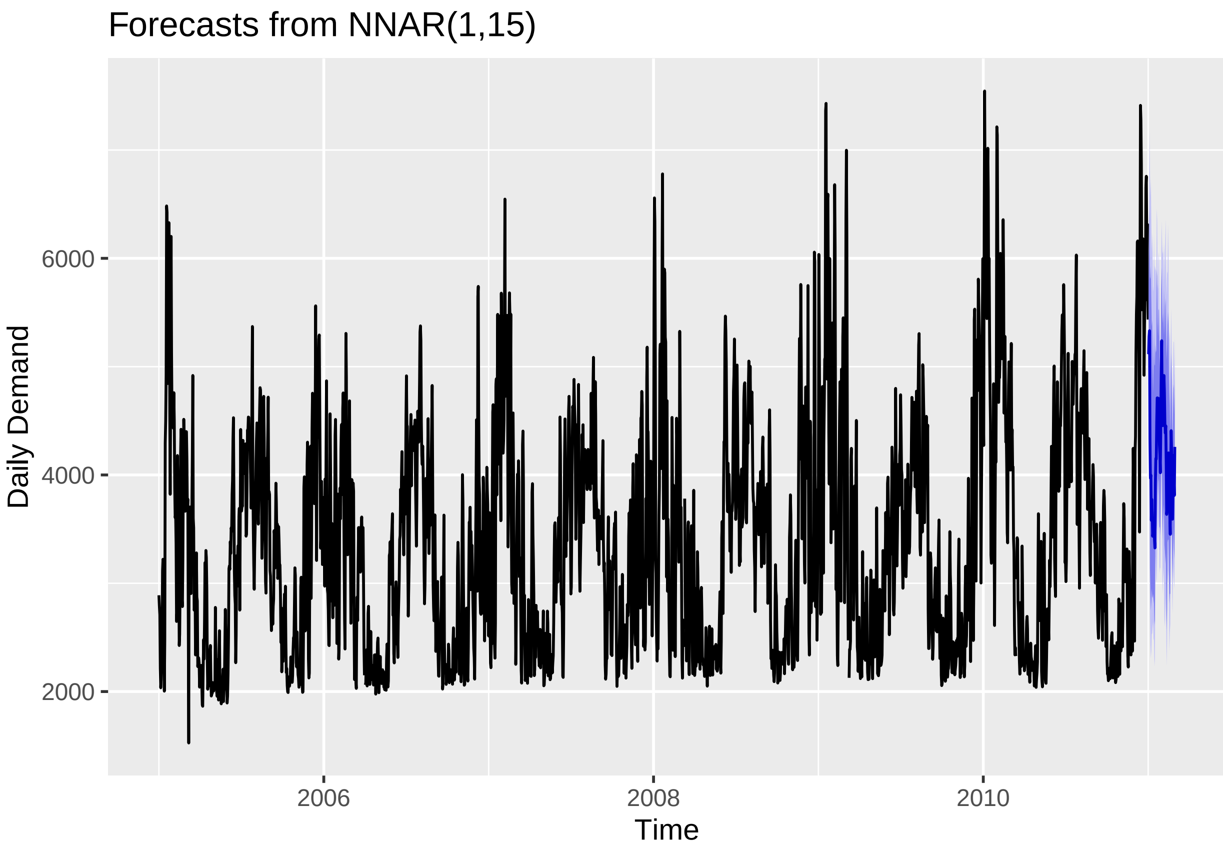
```r
#Check accuracy of the model using just the first two month of 2010
n_for <- 59
observed <- df_processed2010[1827:1885, "Daily_data"]
NN_scores1_for <- accuracy(NN_for_month$mean,observed)
print(NN_scores1_for)
```

```
##                 ME     RMSE     MAE      MPE     MAPE
## Test set 996.1749 1926.172 1580.24 12.94305 30.49053
```

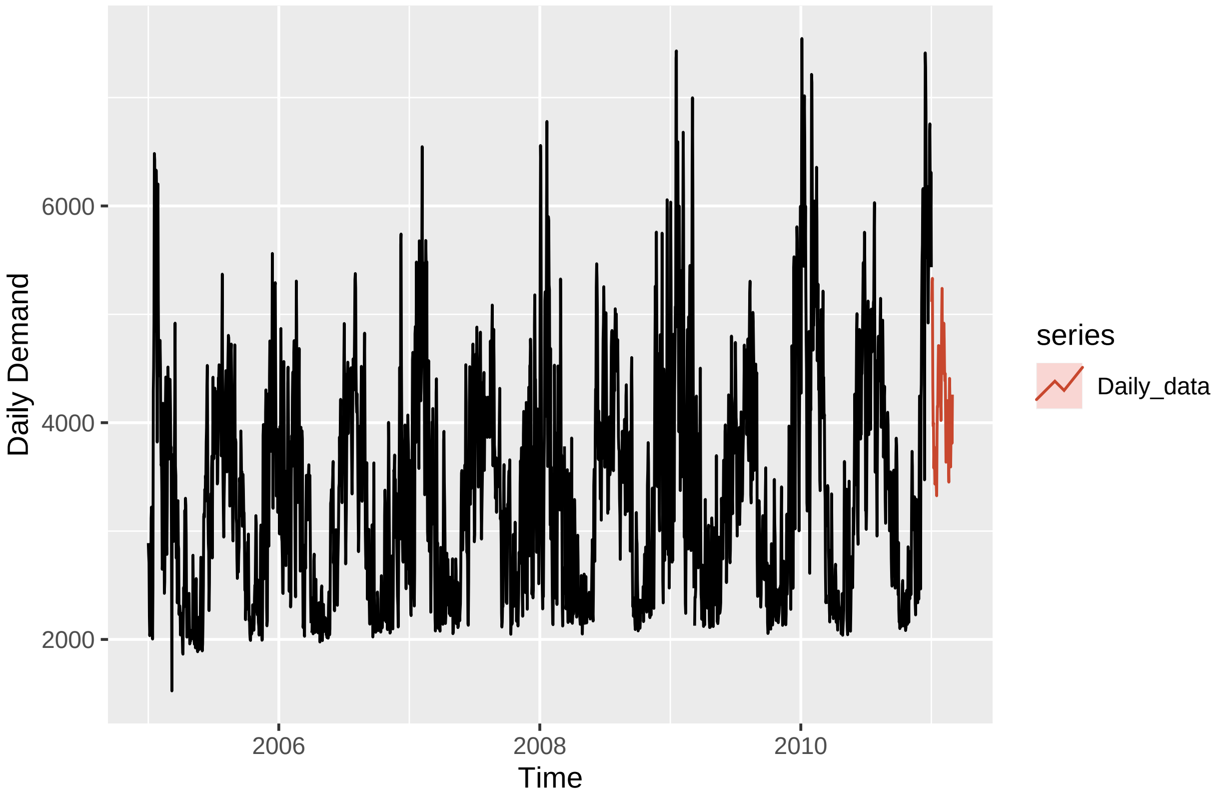
Model 4 Neural Network Time Series: Forecasts 2011

> Note: Based on the error variables calculated with the accuracy() function, the Neural Network model seems to fit the data the best.

```r
#NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
NN_model2010 <- nnetar(ts_daily2010,lambda = 0.5,p=1,P=0,xreg=fourier(ts_daily2010, K=c(2,12)))

#NN_for <- forecast(NN_fit, h=365)
NN_for2010 <- forecast(NN_model2010,PI=TRUE, h=59,xreg=fourier(ts_daily,
                                          K=c(2,12),h=59))

#Plot foresting results
autoplot(NN_for2010) +
  ylab("Daily Demand")
```




```r
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(NN_for2010, series="Daily_data",PI=FALSE)+
  ylab("Daily Demand")
```




```
n_for <- 59
observed <- df_processed2010[(nobs-n_for+1):nobs, "Daily_data"]
NN_scores1_for <- accuracy(NN_for2010$mean,observed)
print(NN_scores1_for)
```

```
##                 ME     RMSE      MAE       MPE     MAPE
## Test set -683.8418 1468.666 1283.441 -31.81401 42.89995
```

Compare performance matrix

```
#create data frame
seas_scores <- as.data.frame(rbind(ETS_scores, TBATS_scores, ARIMA_Four_scores, NN_scores1))
row.names(seas_scores) <- c("STL-ETS","TBATS", "ARIMA_FOUR","NEU-NETWORK")

#choose model with lowest RMSE
best_model_index <- which.min(seas_scores[,"RMSE"])
cat("The best model by RMSE is:", row.names(seas_scores[best_model_index,]))
```

```
## The best model by RMSE is: NEU-NETWORK
```

```
kbl(seas_scores,
      caption = "Forecast Accuracy for Seasonal Data",
      digits = array(5,ncol(seas_scores))) %>%
  kable_styling(full_width = FALSE, position = "center") %>%
  #highlight model with lowest RMSE
  kable_styling(latex_options="striped", stripe_index = which.min(seas_scores[,"RMSE"]))
```

According to the comparison matrics the best model is the neural Network model

The predicted result from Neural Network is as follows:

Table 1: Forecast Accuracy for Seasonal Data

	ME	RMSE	MAE	MPE	MAPE
STL-ETS	-984.5201	1210.6248	1079.3319	-35.24753	36.80666
TBATS	613.9492	1258.1635	921.0355	10.07795	23.50870
ARIMA_FOUR	-853.8232	1158.9565	991.6850	-30.40060	32.76586
NEU-NETWORK	129.7843	714.3135	531.6442	0.24706	14.81317

```
print(NN_for2010$mean)
```

```
## Multi-Seasonal Time Series:
## Start: 2011 2
## Seasonal Periods: 7 365.25
## Data:
##  [1] 5115.337 5276.820 5331.733 4339.016 3970.074 3997.786 3581.245 3775.986
##  [9] 3640.355 3433.763 3472.416 3458.134 3550.535 3406.581 3326.215 3925.199
## [17] 4151.027 4147.417 4490.609 4711.335 4710.861 4590.091 4706.930 4651.126
## [25] 4278.664 4182.955 4020.511 4443.150 4989.587 5239.424 4707.503 4454.901
## [33] 4691.998 4613.567 4918.071 4763.635 4484.260 4386.939 4453.338 4093.371
## [41] 3635.661 3688.656 4117.021 4142.763 4204.004 3986.060 3758.027 3625.610
## [49] 3451.885 4093.938 4409.048 4214.171 3805.231 3591.632 3868.588 4179.623
## [57] 3970.774 3811.065 4259.031
```

```
print(ARIMA_Four_for2011$mean)
```

```
## Multi-Seasonal Time Series:
## Start: 2011 2
## Seasonal Periods: 7 365.25
## Data:
##  [1] 5526.415 5794.942 5726.939 5667.654 5706.920 5639.929 5716.611 5964.147
##  [9] 6024.705 5911.567 5926.901 6015.781 6071.908 6284.935 6621.350 6715.763
## [17] 6611.811 6643.567 6751.663 6812.521 7027.192 7351.693 7383.301 7181.735
## [25] 7117.411 7124.783 7073.802 7174.451 7377.935 7285.460 6973.496 6809.690
## [33] 6728.359 6607.167 6643.298 6789.783 6681.093 6389.449 6250.021 6200.795
## [41] 6127.257 6210.527 6407.701 6371.016 6159.486 6090.784 6105.596 6089.920
## [49] 6222.002 6459.386 6448.920 6246.259 6173.030 6169.368 6120.337 6205.469
## [57] 6380.416 6298.339 6023.813
```

```
print(TBATS_for2011$mean)
```

```
## Multi-Seasonal Time Series:
## Start: 2011 2
## Seasonal Periods: 7 365.25
## Data:
##  [1] 2562.047 2821.314 2918.414 2939.593 2941.253 2940.383 2939.891 2939.747
##  [9] 2939.722 2939.722 2939.724 2939.725 2939.725 2939.725 2939.725 2939.725
## [17] 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725
## [25] 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725
## [33] 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725
## [41] 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725
## [49] 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725 2939.725
## [57] 2939.725 2939.725 2939.725
```