HuanHusted_TSA_Competition

Kelsey Husted

2023-03-31

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() functino 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.

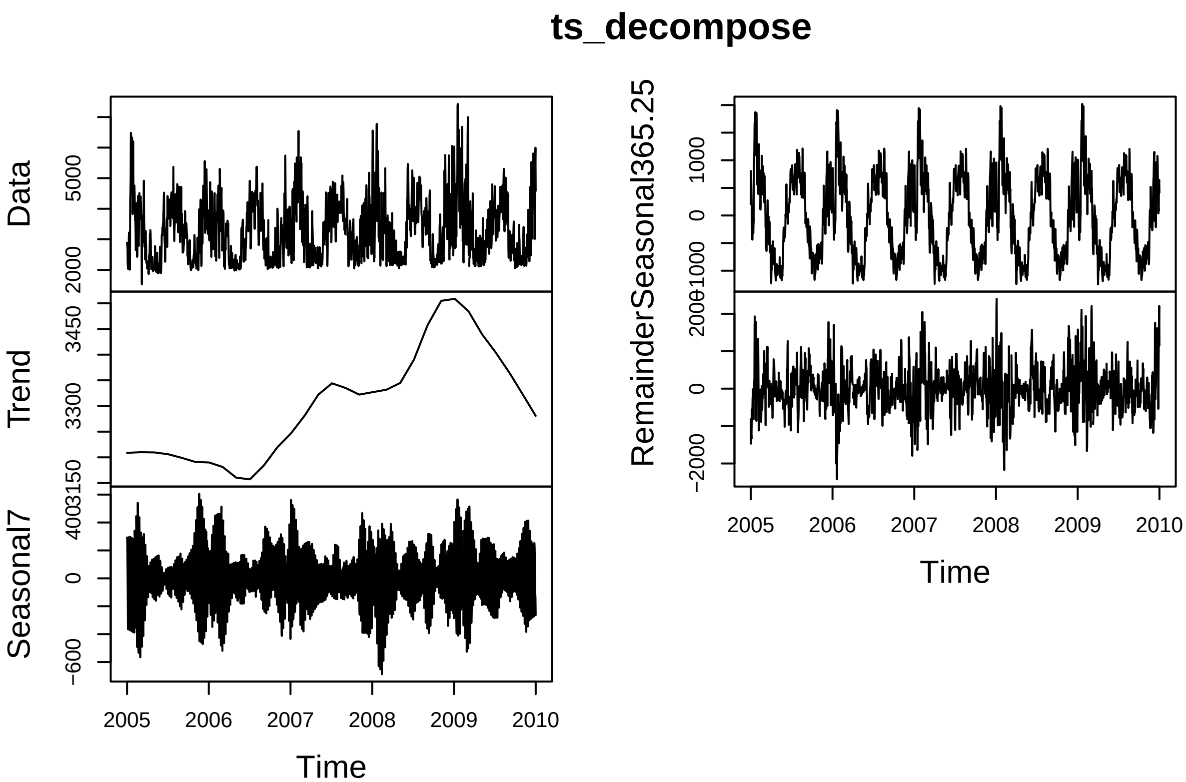
```
#Wrangle data from hourly to daily
#Wrangling date column 2005 to 2010
#removing no numeric columns so rowMeans() function will work
df_daily2010 <- df %>%
  mutate( Date = ymd(date)) %>%
  select(3:26)

#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))
```

Time series decomposition and plot

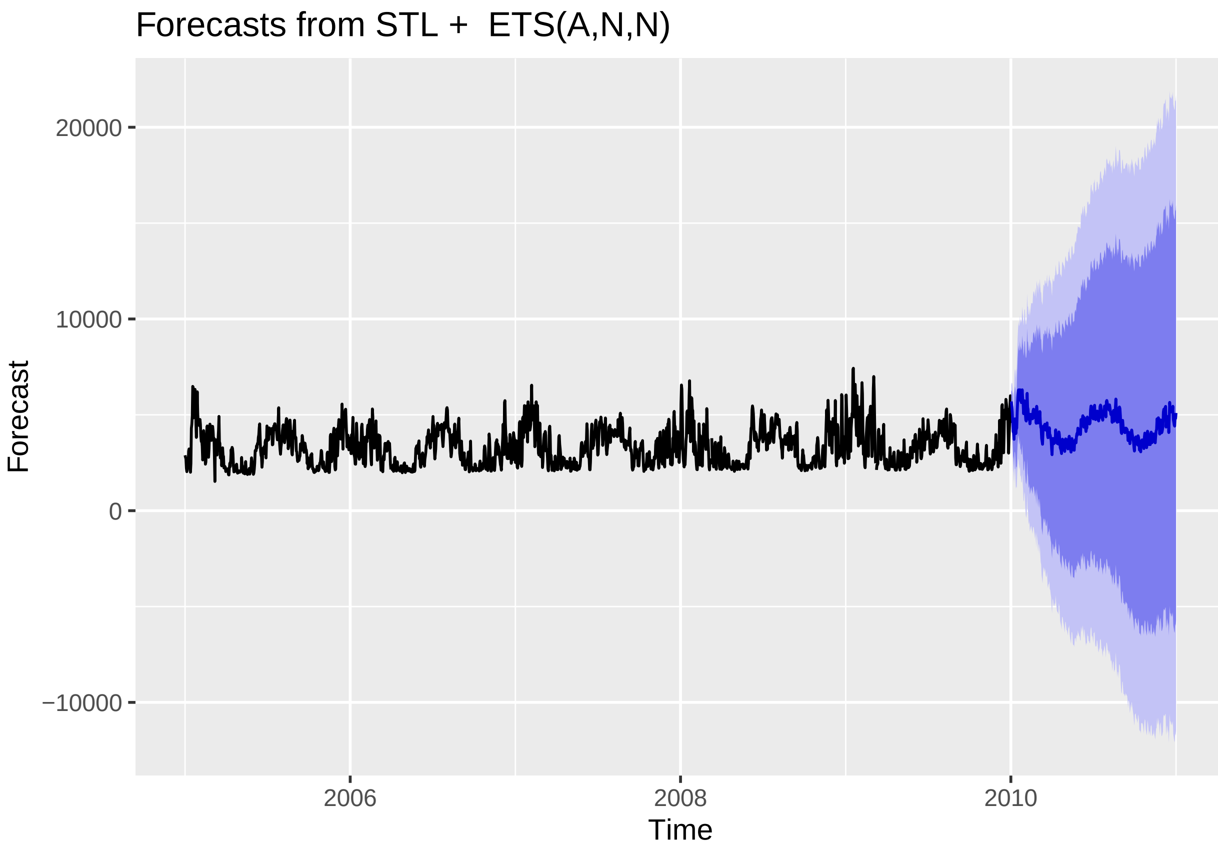
```
#Decompose time series
ts_decompose <- ts_daily %>%
  mstl()
plot(ts_decompose)
```



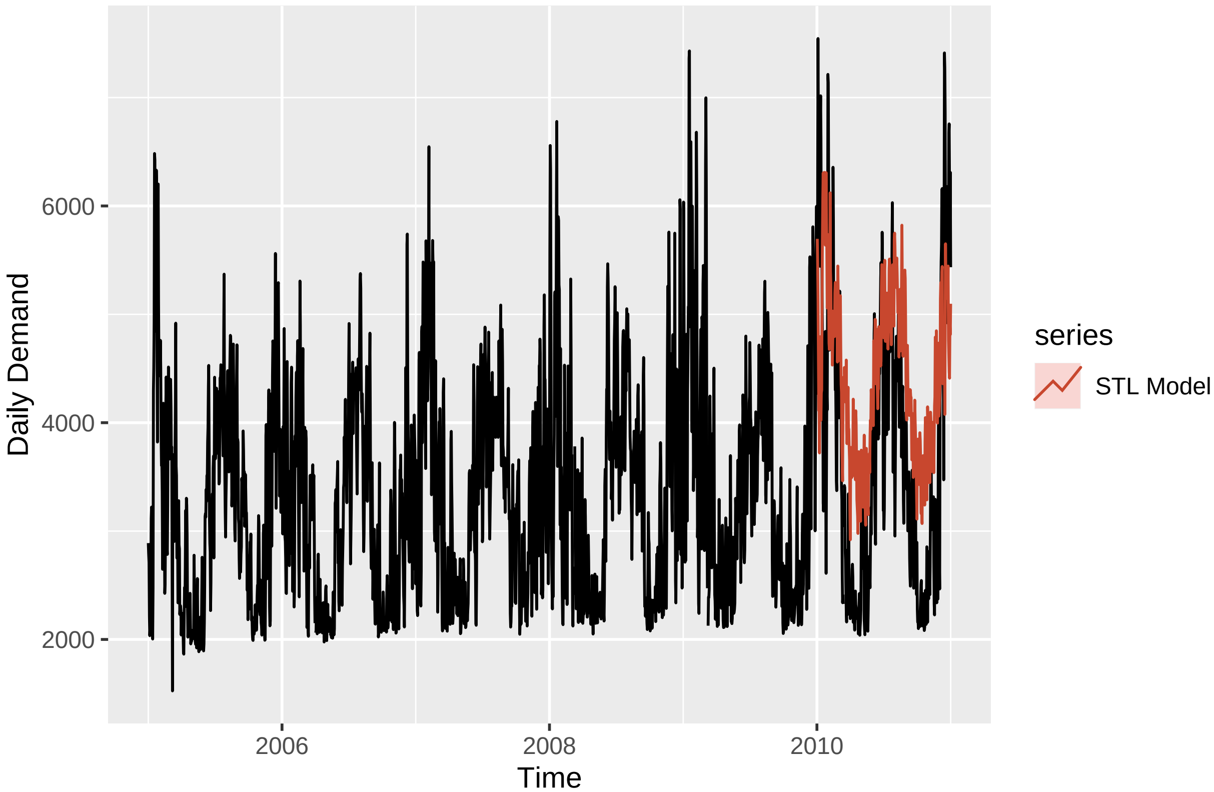
##

Model 1 STL + ETS: Forecast 2010

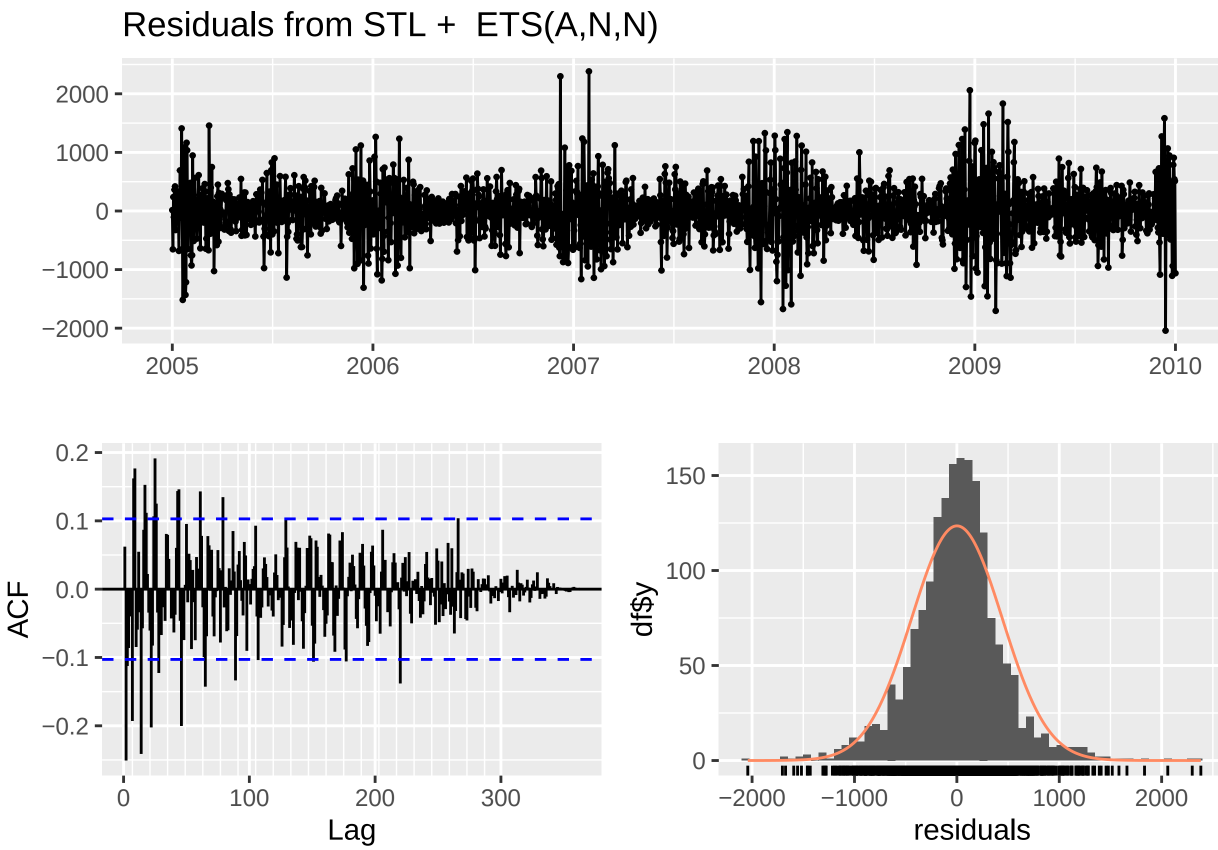
```
#Fit and forecast STL model
ETS_model <-  stlf(ts_daily,h=365)

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




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




```r
#Plot the residuals

checkresiduals(ETS_model)
```



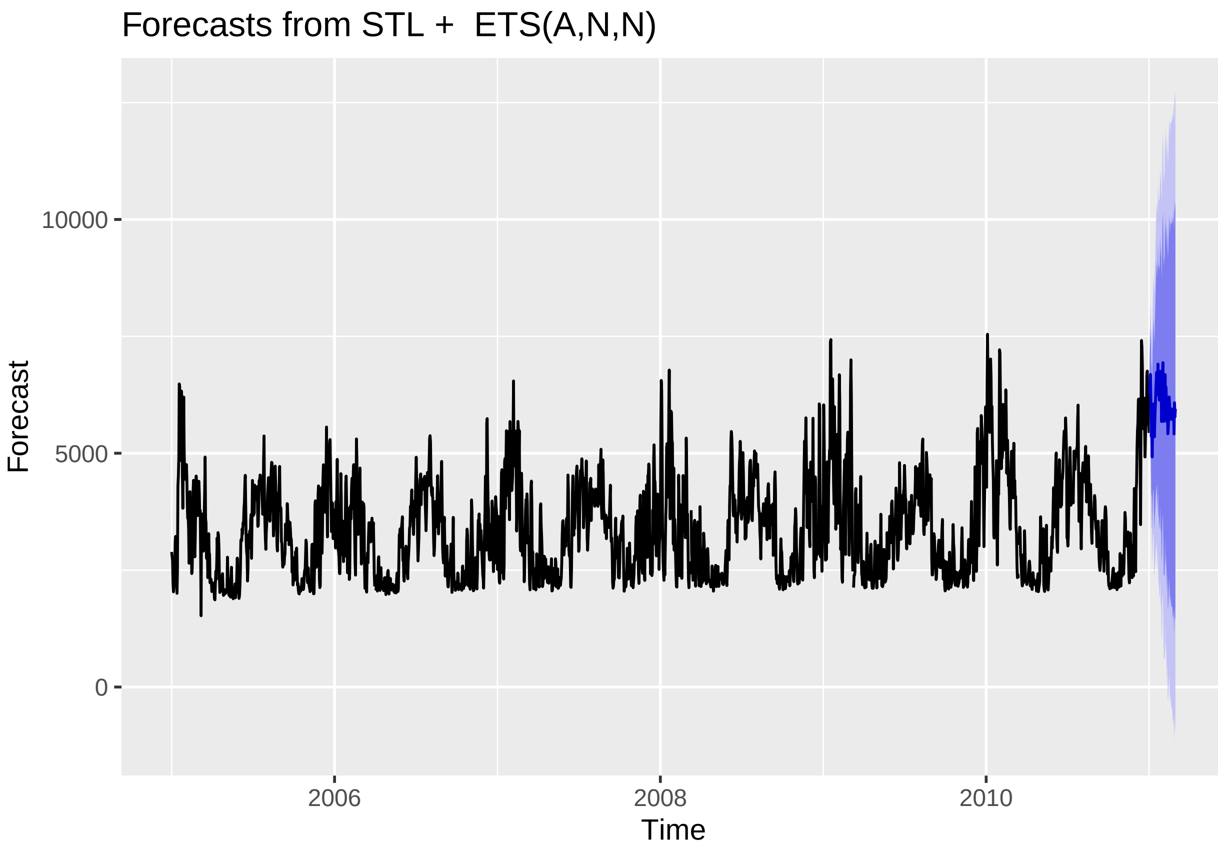

```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 1309.7, df = 363, p-value < 2.2e-16
##
## Model df: 2.   Total lags used: 365
```

```
#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)
```

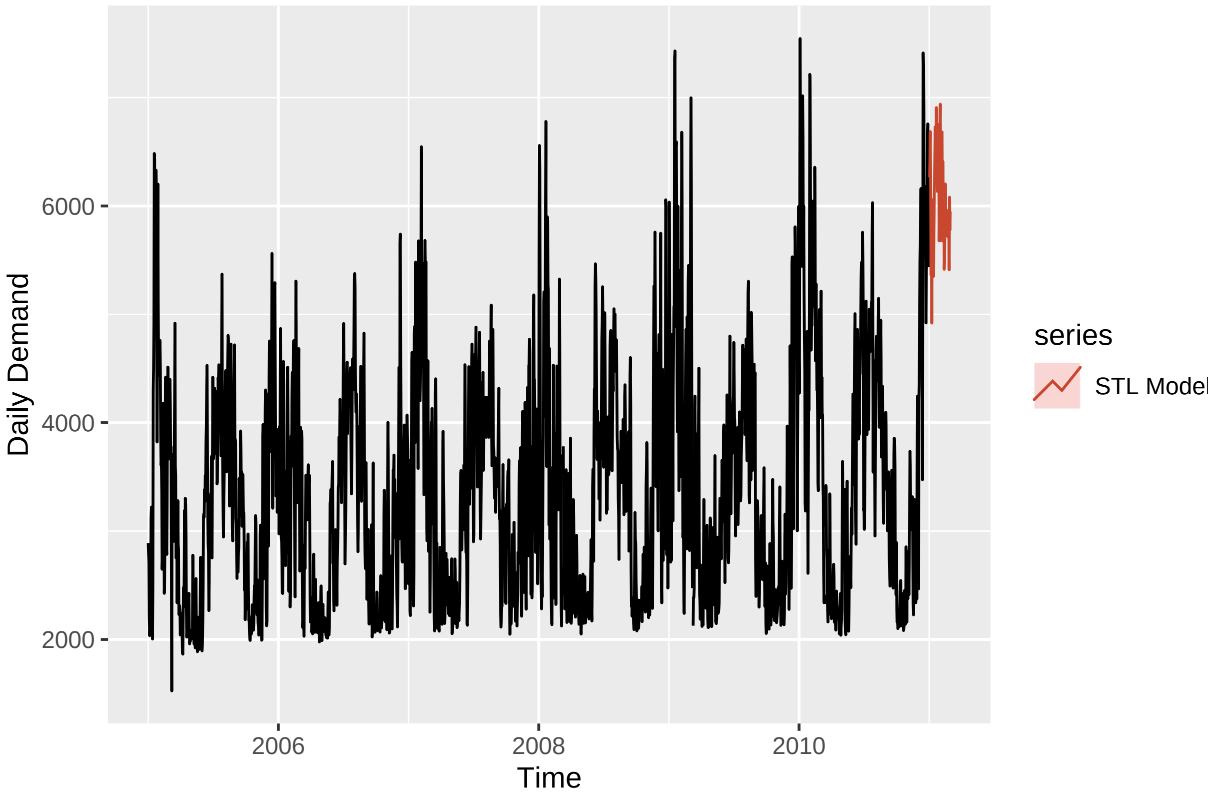
```
##                 ME     RMSE      MAE       MPE     MAPE
## Test set -984.5201 1210.625 1079.332 -35.24753 36.80666
```

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")
```




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



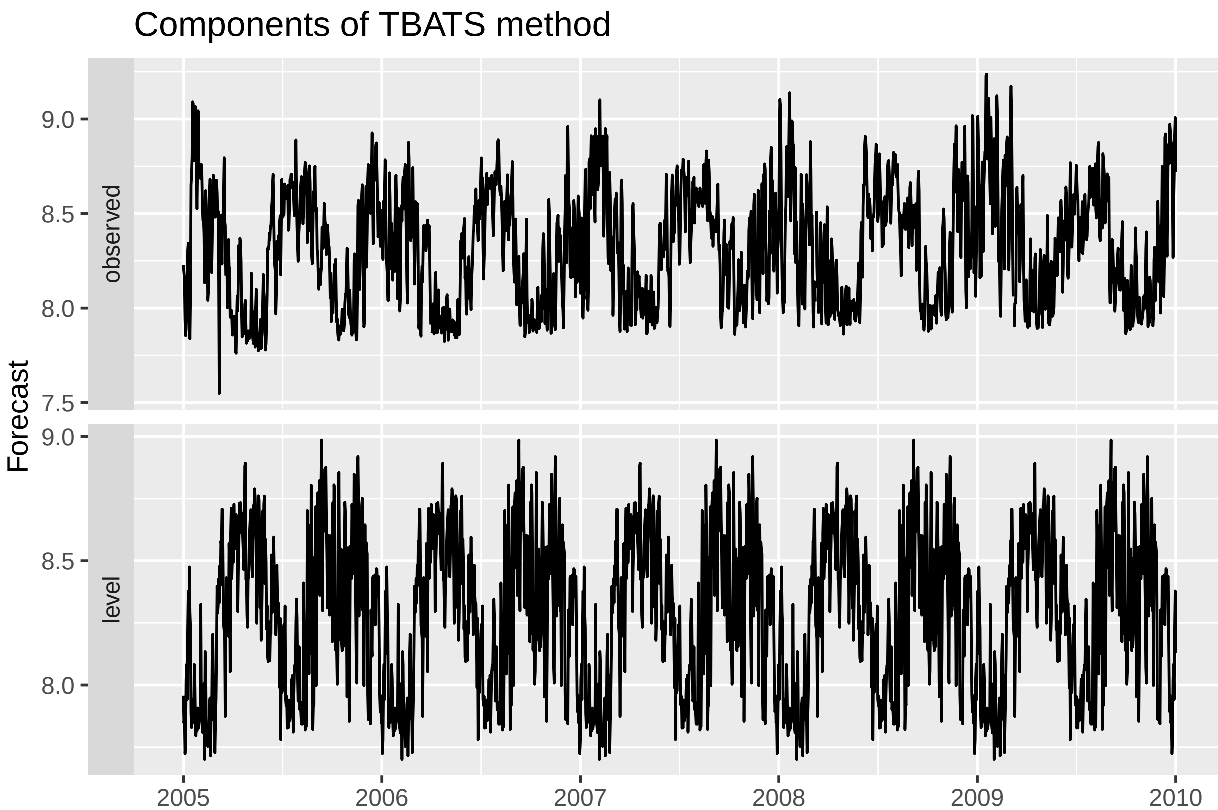

```
#Check accuracy of model
n_for <- 59
observed <- df_processed2010[(nobs-n_for+1):nobs, "Daily_data"]
ETS_scores_for <- accuracy(ETS_model2011$mean,observed)
print(ETS_scores_for)
```

```
##                  ME     RMSE      MAE       MPE     MAPE
## Test set -2528.13 2846.706 2535.511 -90.47756 90.60064
```

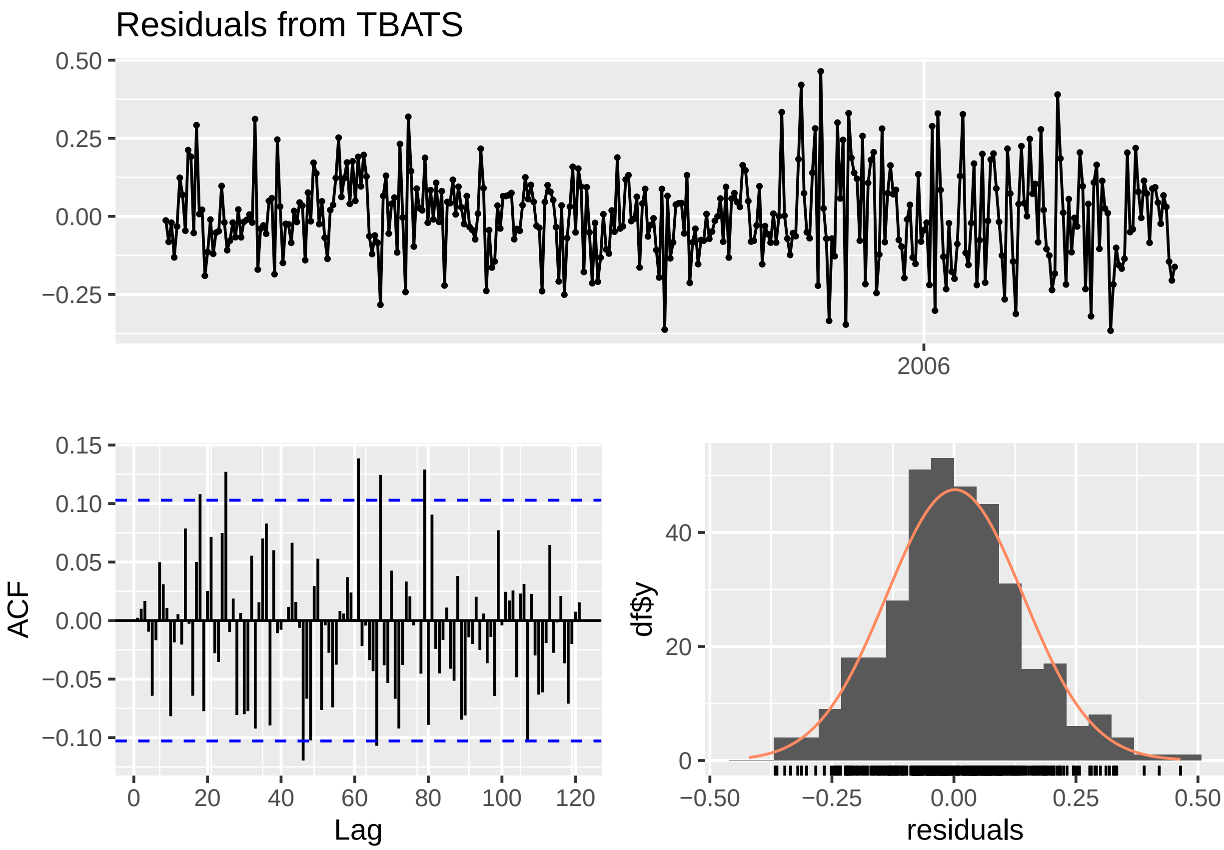
Model 2 TBATS: Forecast 2010

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
TBATS_for <- forecast(TBATS_model,h=365)

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

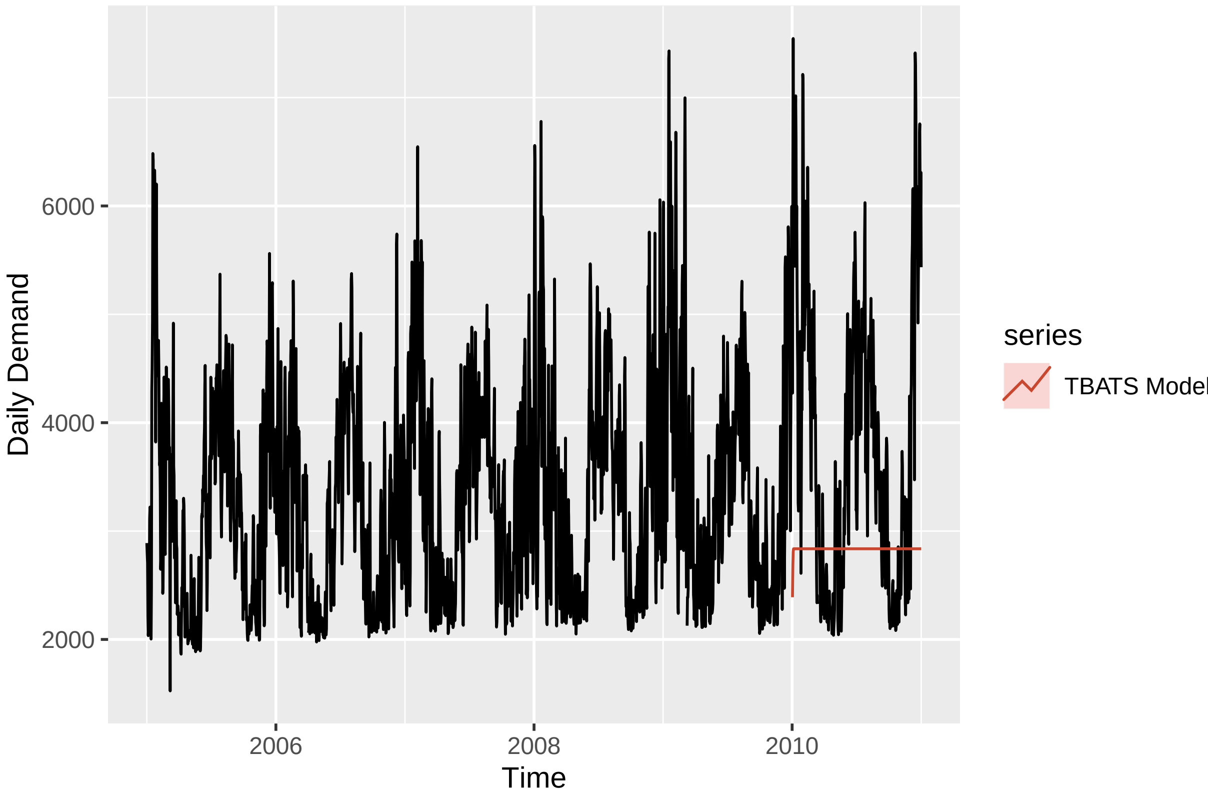



```r
#Plot the residuals

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")
```




```
#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)
```

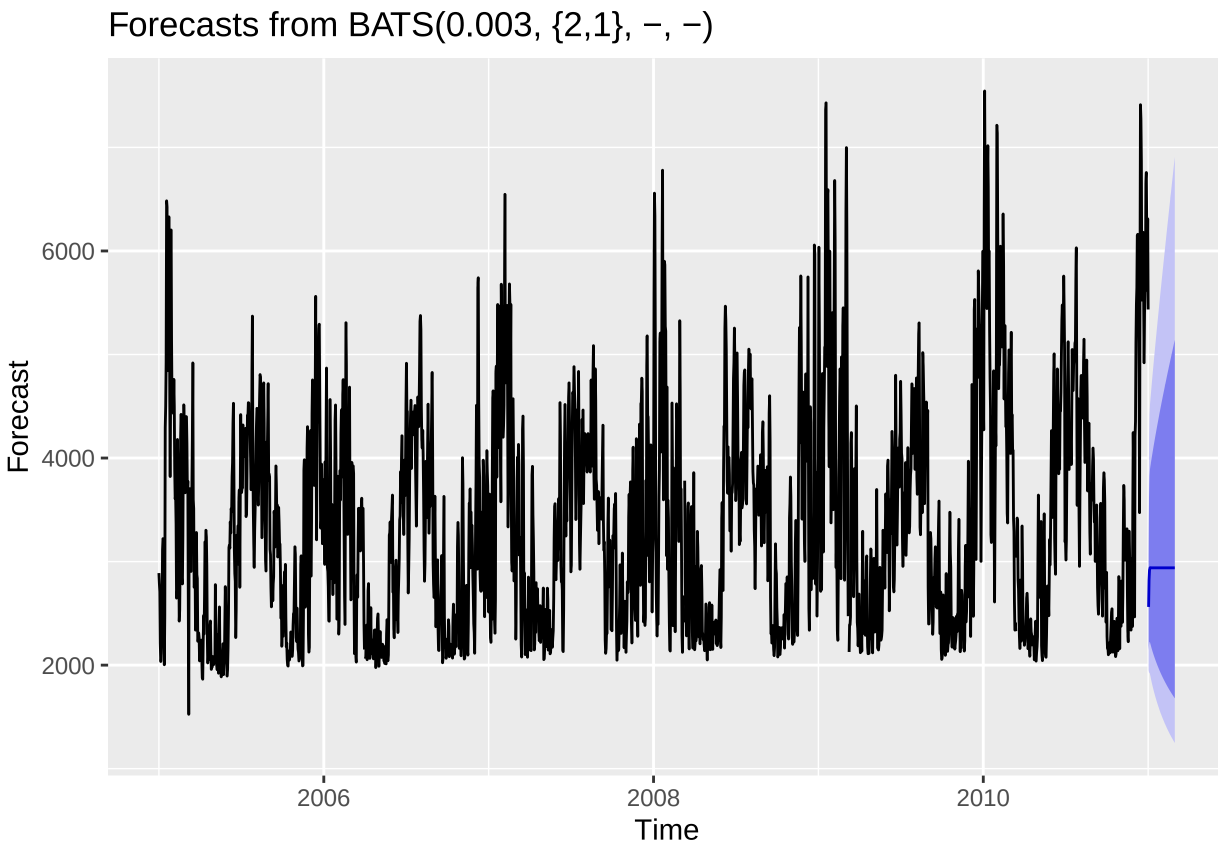
```
##                ME     RMSE      MAE      MPE    MAPE
## Test set 613.9492 1258.164 921.0355 10.07795 23.5087
```

Model 2 TBATS: Forecast 2011

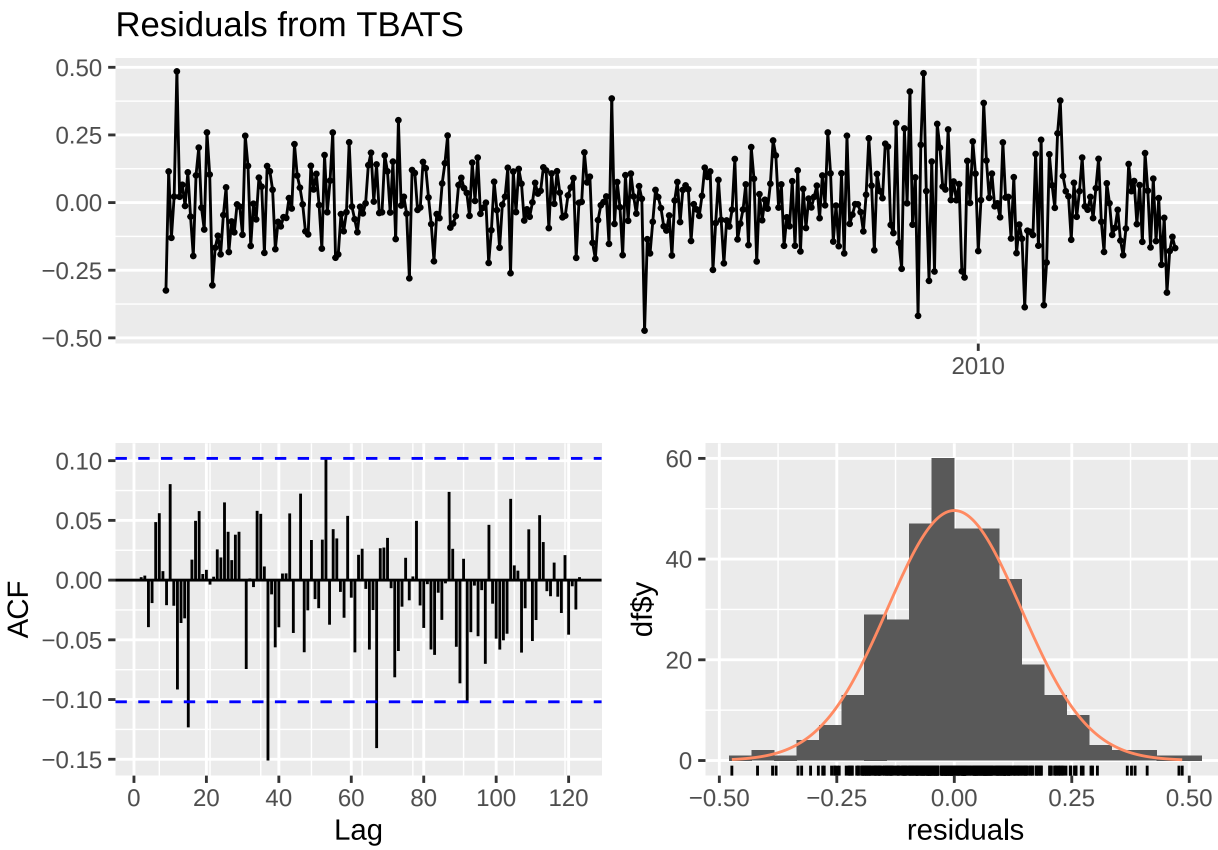
```
#Fit and forecast TBATS model
TBATS_model2011 <-  tbats(ts_daily2010)
```

```
## 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")
```

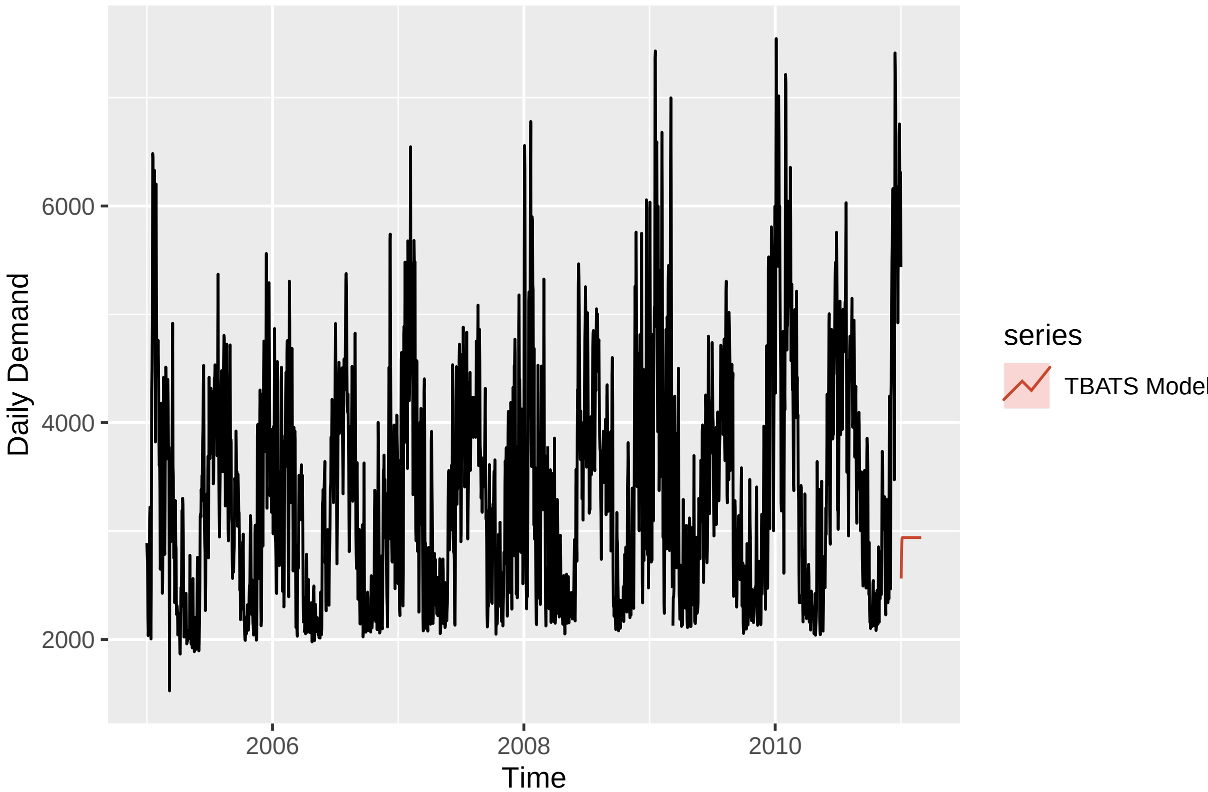



```r
#Plot the residuals

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")
```

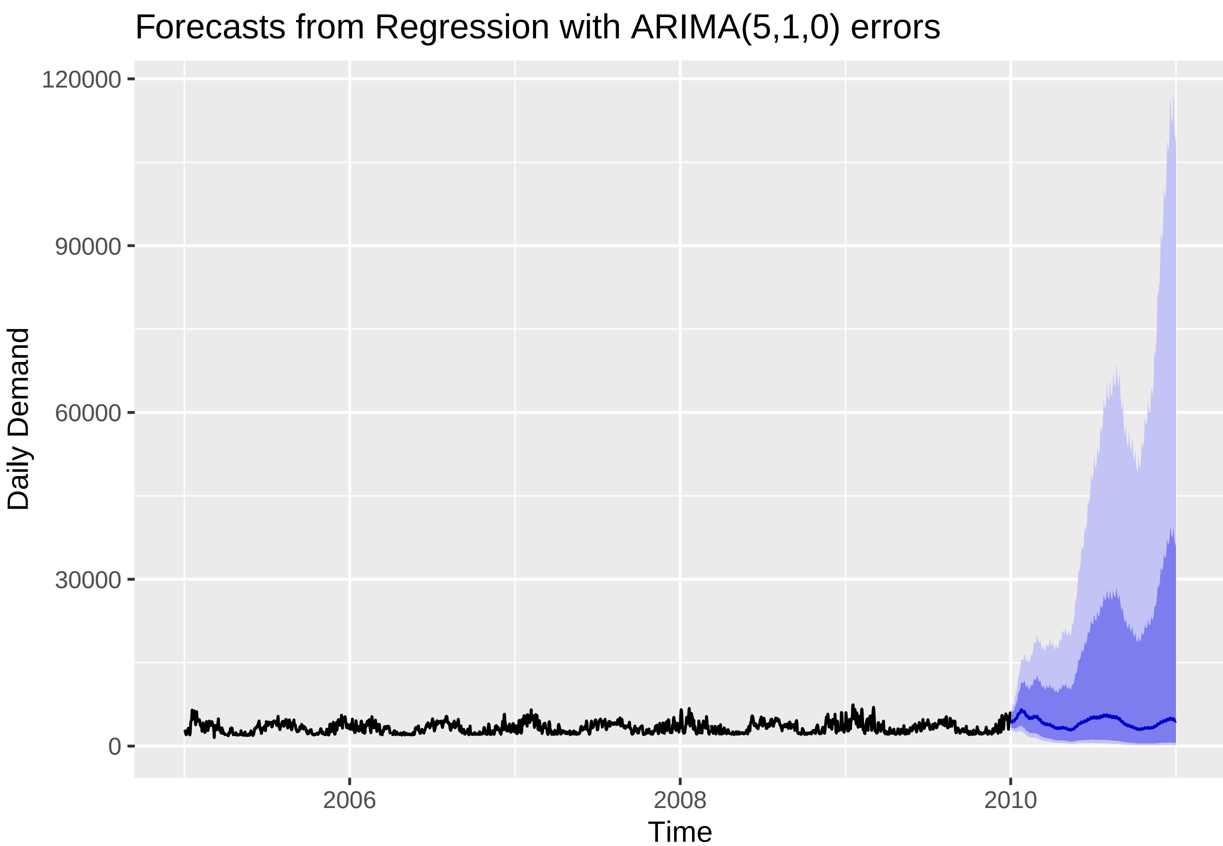



```
#Check accuracy of model
n_for <- 59
observed <- df_processed2010[(nobs-n_for+1):nobs, "Daily_data"]
TBATS_scores_for <- accuracy(TBATS_for$mean,observed)
print(TBATS_scores_for)
```

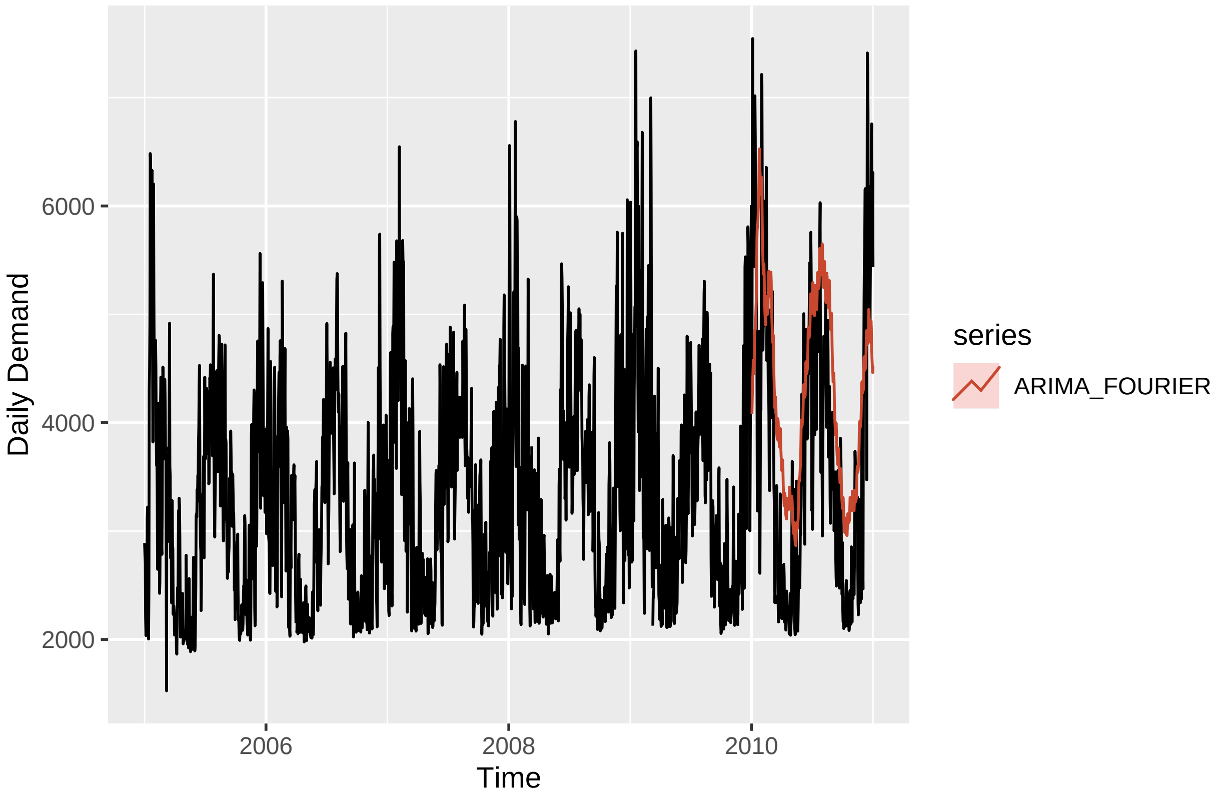
```
##                  ME     RMSE      MAE      MPE     MAPE
## Test set 681.5893 1327.303 964.6684 11.34867 23.42342
```

Model 3 ARIMA + FOURIER terms: Forecast 2010

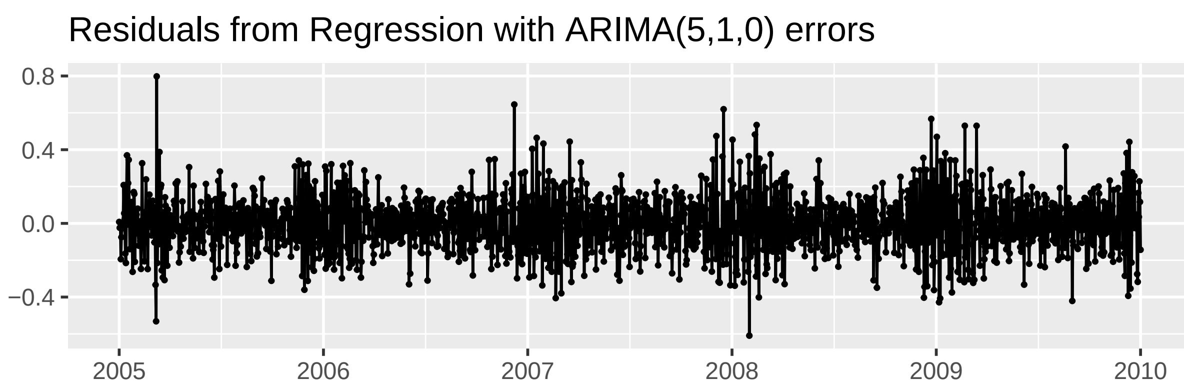
```
#Fit and forecast TBATS model
ARIMA_Four_model <- auto.arima(ts_daily,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_daily,
                                          K=c(2,12))
                             )

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

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

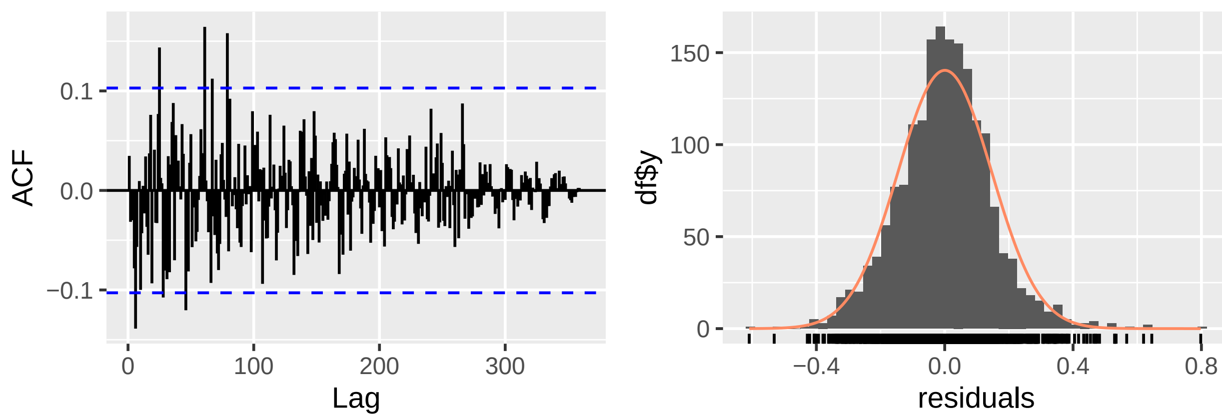



```
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(ARIMA_Four_for, series="ARIMA_FOURIER",PI=FALSE) +
  ylab("Daily Demand")
```




```
# 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
```

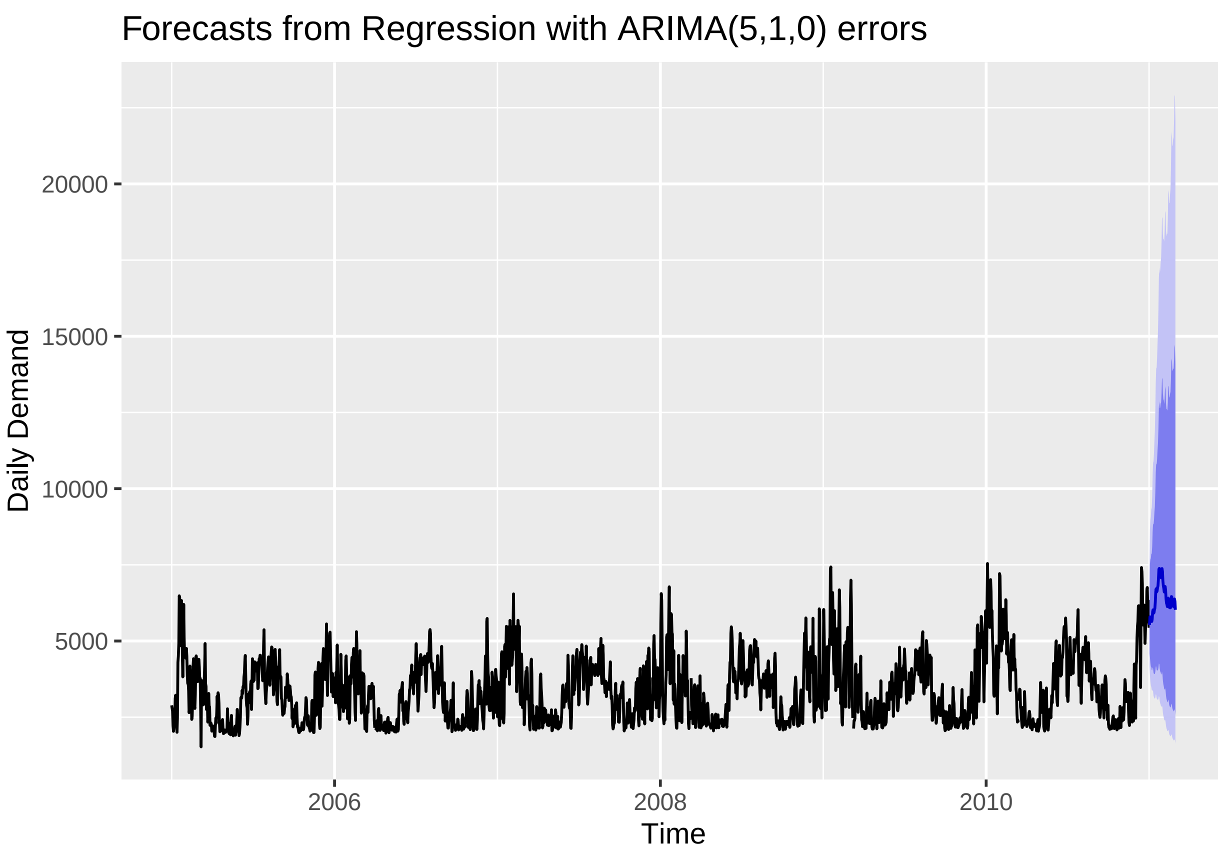
```
#Check accuracy of model
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
ARIMA_Four_scores <- accuracy(ARIMA_Four_for$mean,observed)
print(ARIMA_Four_scores)
```

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

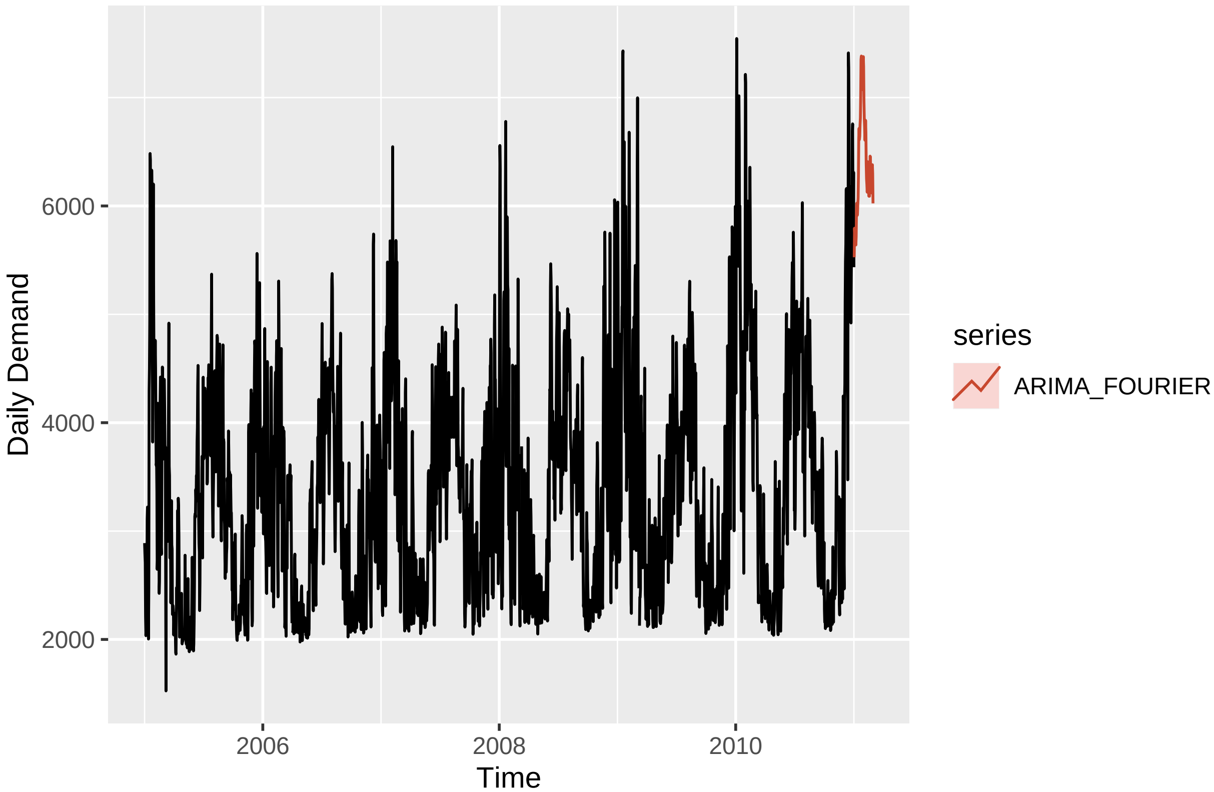
Model 3 ARIMA + FOURIER terms: Forecast 2011

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

```r
#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")
```




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



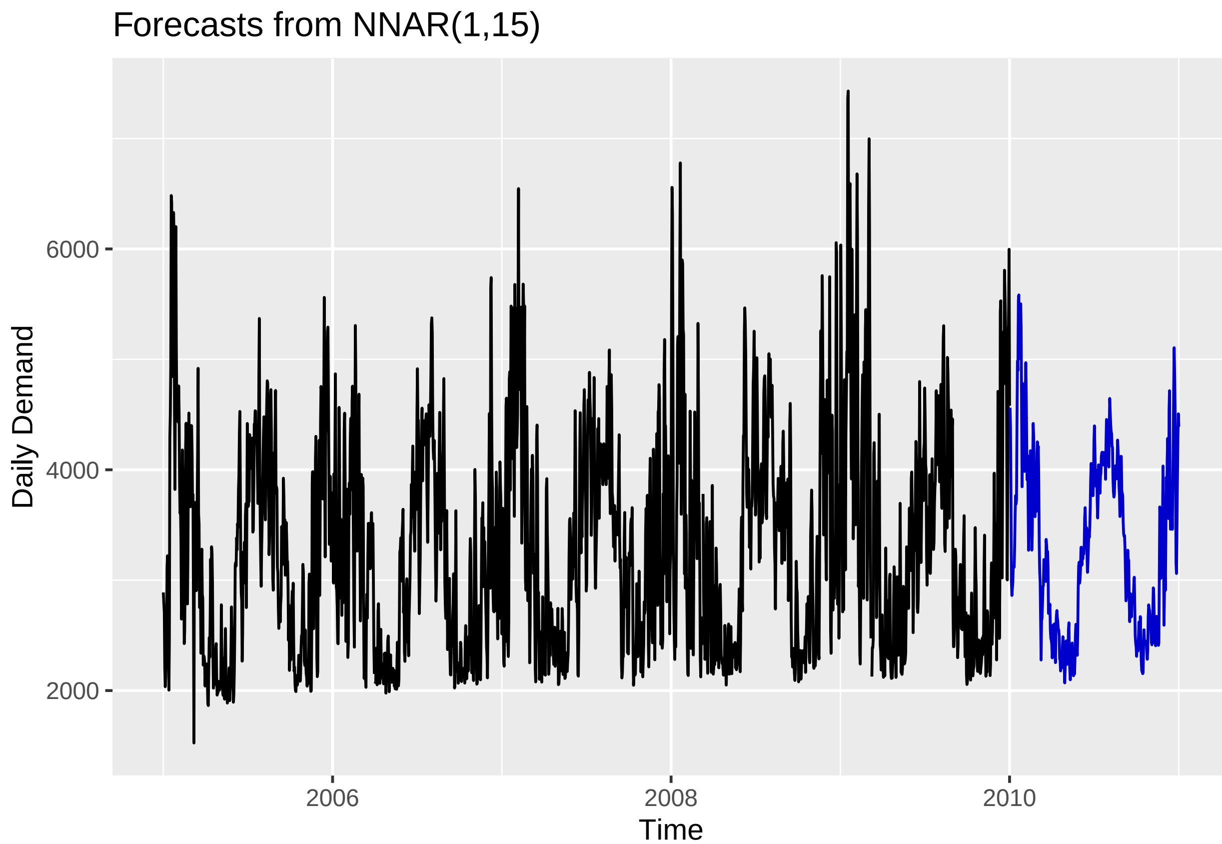

```
#Check accuracy of model
n_for <- 59
observed <- df_processed2010[(nobs-n_for+1):nobs, "Daily_data"]
ARIMA_Four_scores_for <- accuracy(ARIMA_Four_for$mean,observed)
print(ARIMA_Four_scores_for)
```

```
##                   ME    RMSE      MAE     MPE     MAPE
## Test set -1829.112 2256.61 1946.548 -67.636 69.71813
```

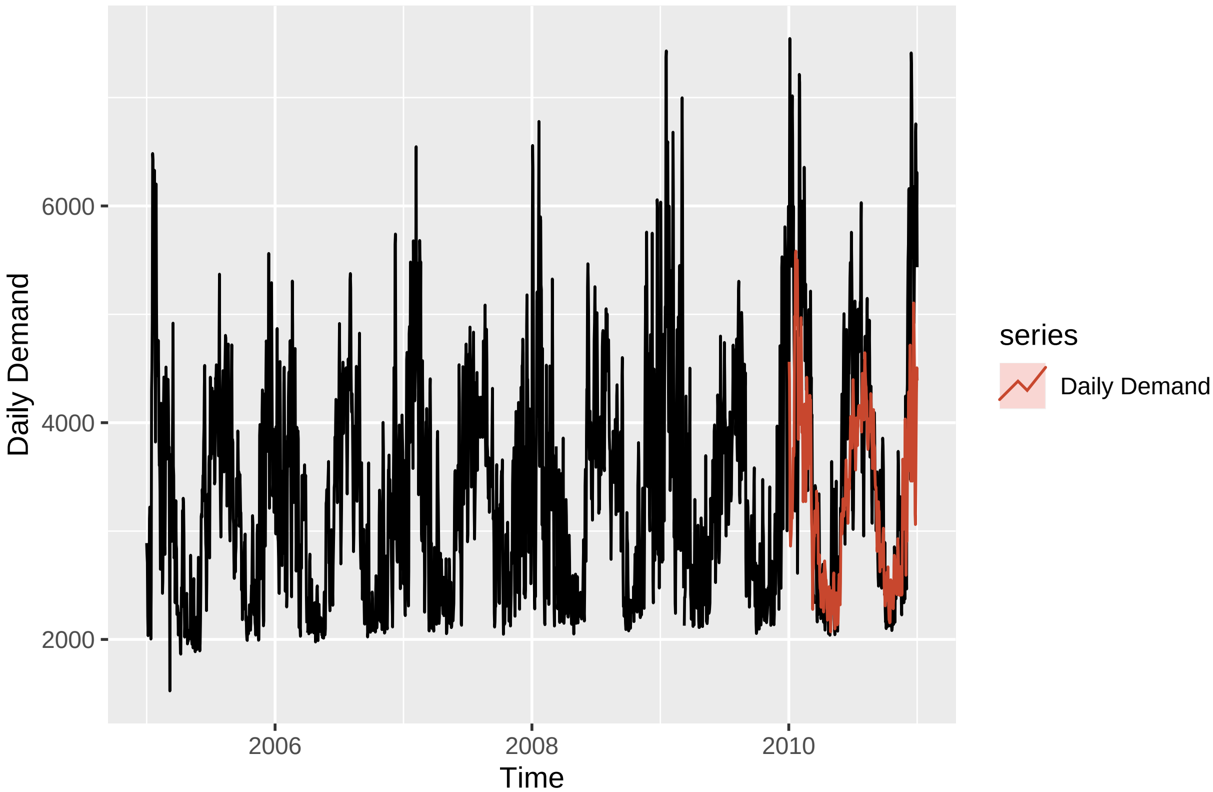
```
#print(ARIMA_Four_for2011$mean)
```

Model 4 Neural Network Time Series: Forecasts 2010

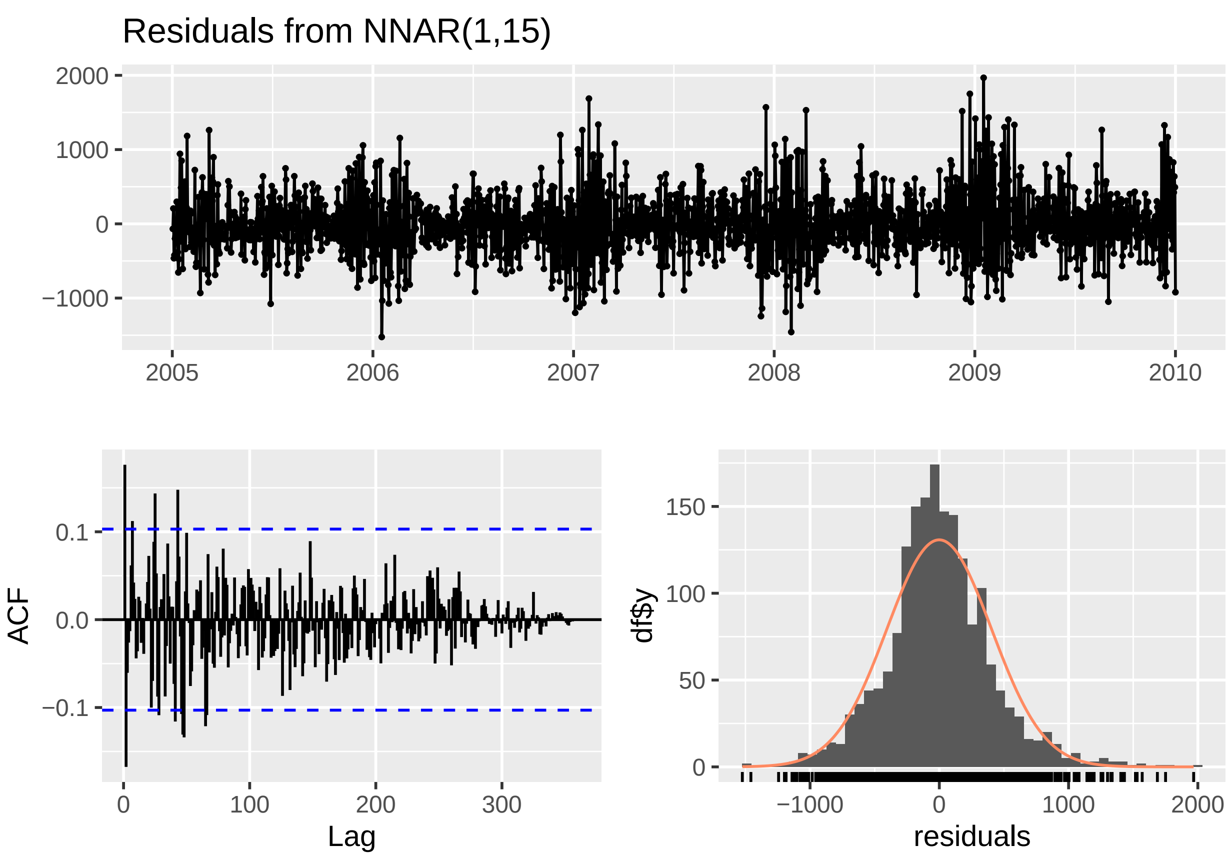
```
#NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
NN_model <- nnetar(ts_daily,decay=0.5, maxit=150, p=1,P=0,xreg=fourier(ts_daily, K=c(2,12)))

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

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




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




```
checkresiduals(NN_model)
```



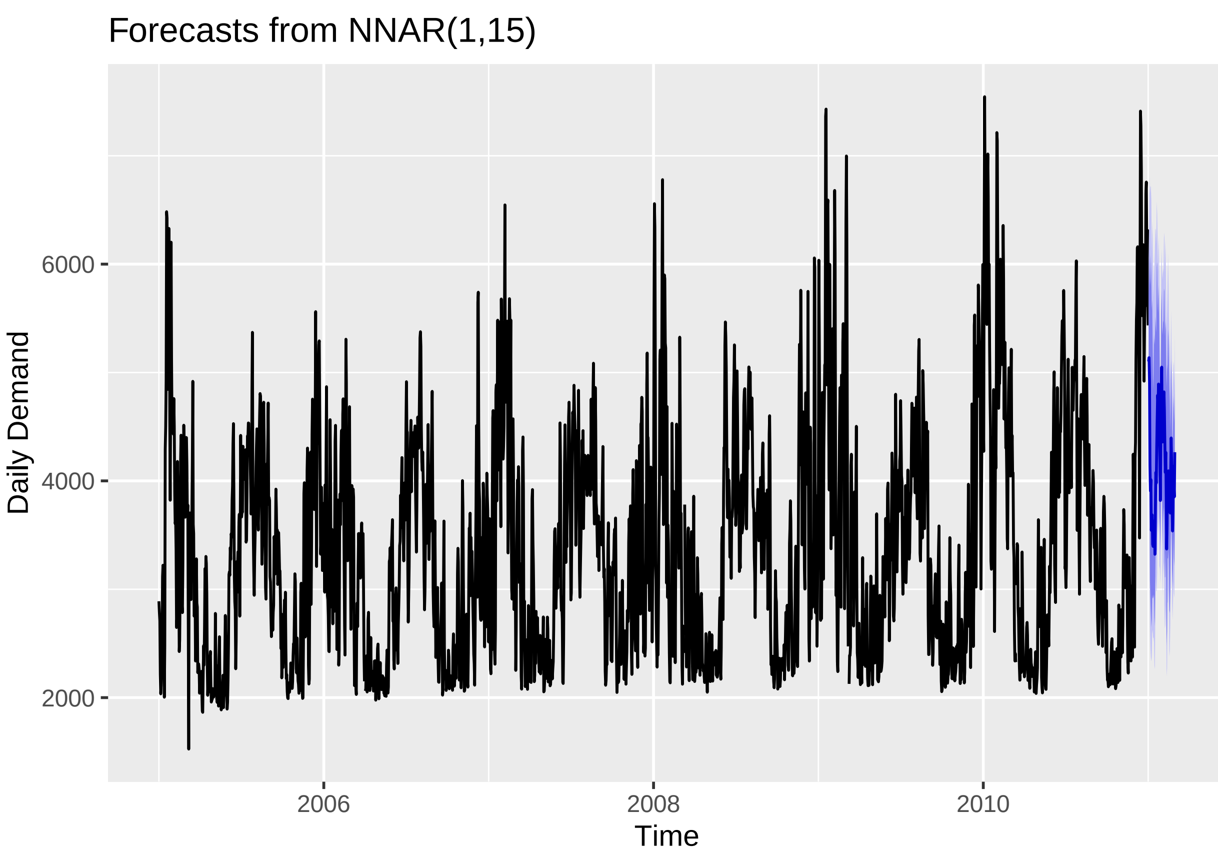
```r
#Checking error variables to decide which model fits the data the best
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
NN_scores1 <- accuracy(NN_for$mean,observed)
print(NN_scores1)
```

```
##                 ME    RMSE      MAE       MPE     MAPE
## Test set 128.3265 719.635 533.5096 0.2277706 14.88977
```

Model 4 Neural Network Time Series: Forecasts 2011

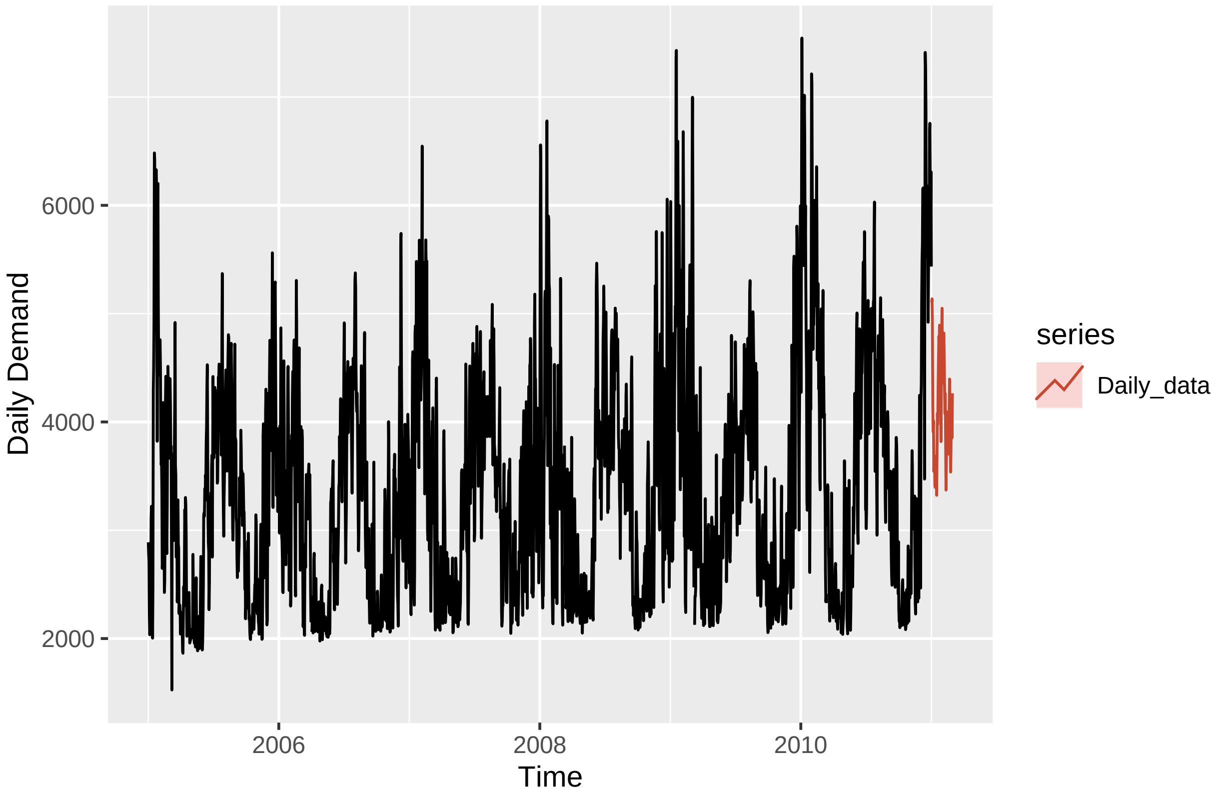
> Note: Based on the error variables calculated with the accuracy() function, the Neural Network model seems to fit the data the best.

```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 -626.8692 1416.631 1231.42 -29.89841 41.10793
```

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

```
## Multi-Seasonal Time Series:
## Start: 2011 2
## Seasonal Periods: 7 365.25
## Data:
##  [1] 5102.618 5137.065 4915.997 4143.501 3907.294 4011.922 3543.865 3655.539
##  [9] 3686.191 3414.190 3395.333 3416.974 3628.971 3626.367 3322.870 3728.883
## [17] 4078.190 3979.397 4311.477 4570.908 4790.297 4715.995 4893.897 4746.145
## [25] 4428.321 4188.823 3818.837 4311.184 4834.918 5050.009 4582.582 4358.941
## [33] 4610.946 4669.567 4821.550 4645.277 4326.864 4077.743 4262.920 3842.389
## [41] 3371.245 3586.714 4025.740 4005.608 4092.972 4026.003 3845.735 3730.121
## [49] 3702.882 4119.025 4396.147 4174.105 3799.694 3537.427 3749.126 4135.390
## [57] 3963.938 3852.745 4264.494
```

Compare performance matrix

```
#create data frame
seas_scores <- as.data.frame(rbind(ETS_scores, TBATS_scores, ARIMA_Four_scores, NN_scores1))
```

Table 1: Forecast Accuracy for Seasonal Data

	ME	RMSE	MAE	MPE	MAPE
STL-ETS	-984.5201	1210.625	1079.3319	-35.24753	36.80666
TBATS	613.9492	1258.164	921.0355	10.07795	23.50870
ARIMA_FOUR	-853.8232	1158.956	991.6850	-30.40060	32.76586
NEU-NETWORK	128.3265	719.635	533.5096	0.22777	14.88977

```
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"]))
```