HuanHusted_TSA_Competition

Kelsey Husted & Yu Huan

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)") %>%
```

```r
  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.

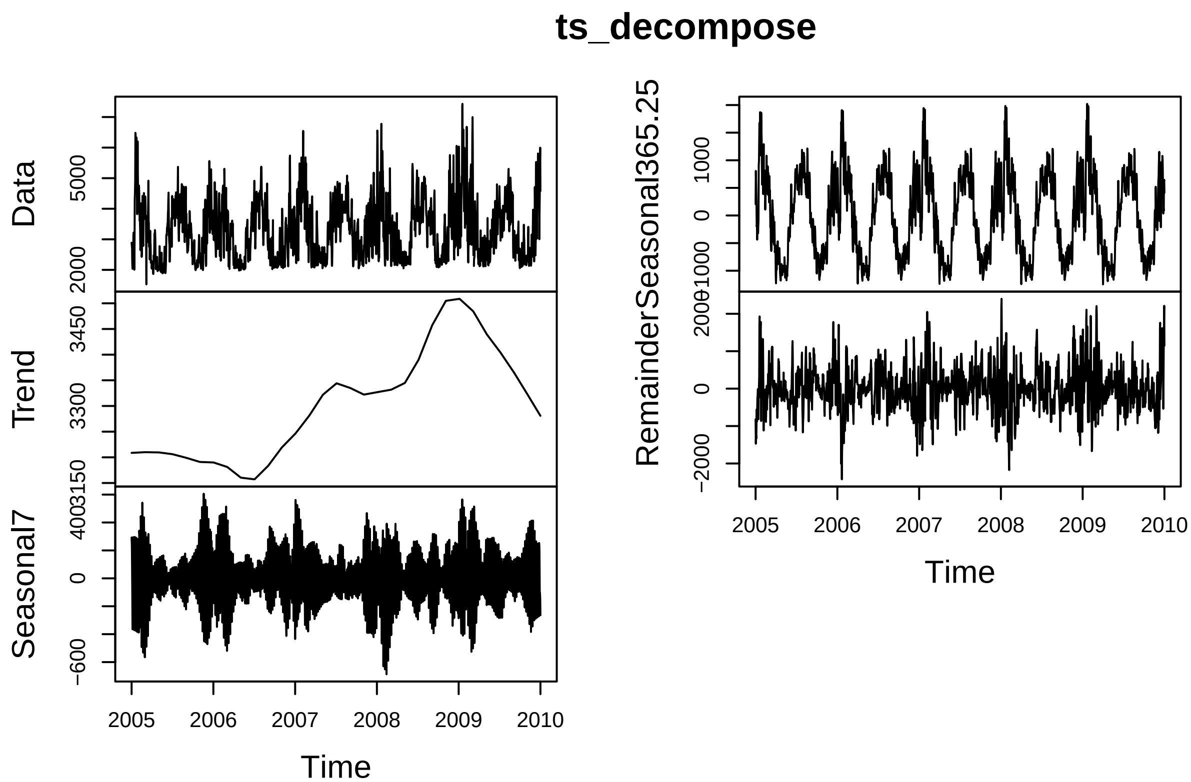
```r
#Wrangle data from hourly to daily
#Wrangling date column 2005 to 2010
#removing no numeric columns so rowMeans() function will work
df_daily2010 <- df %>%
  mutate( Date = ymd(date)) %>%
  select(3:26)

#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

```r
#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

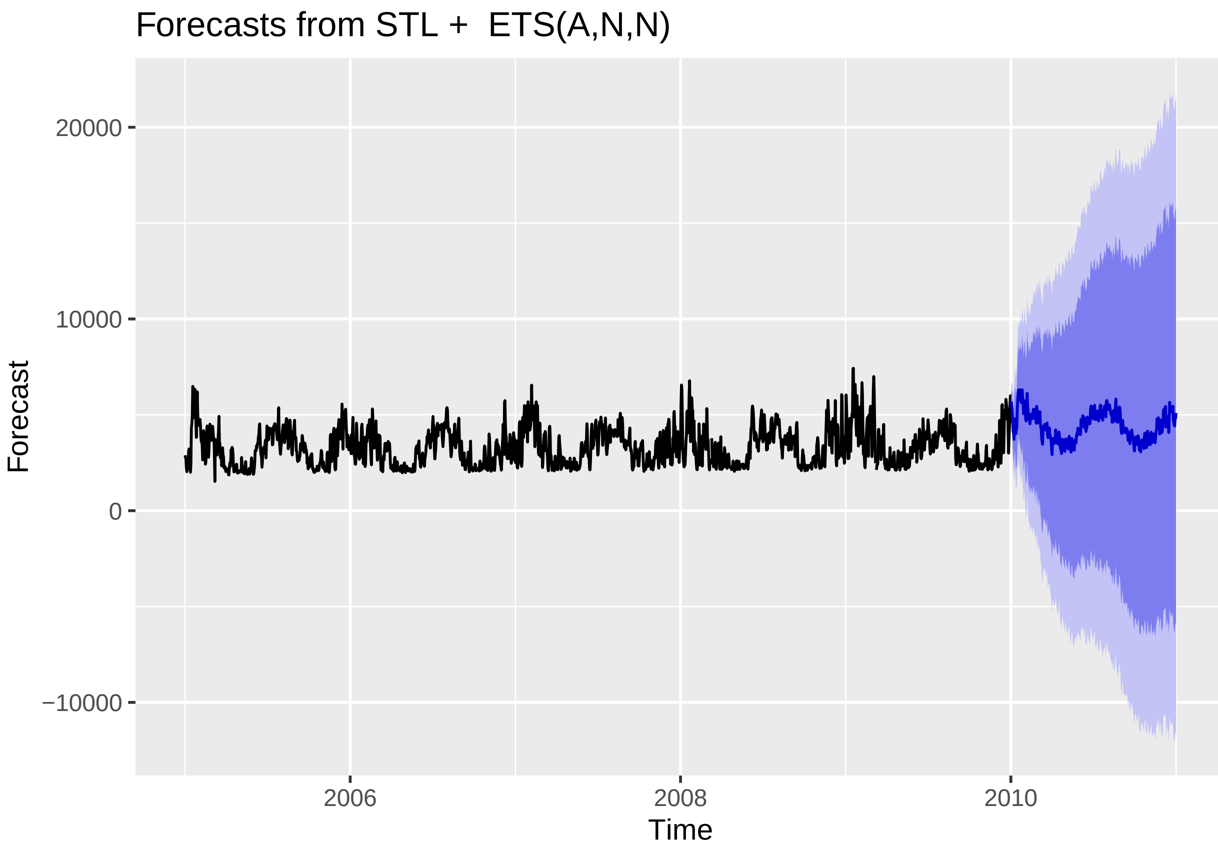
```r
#Decompose time series
ts_decompose <- ts_daily %>%
  mstl()
plot(ts_decompose)
```



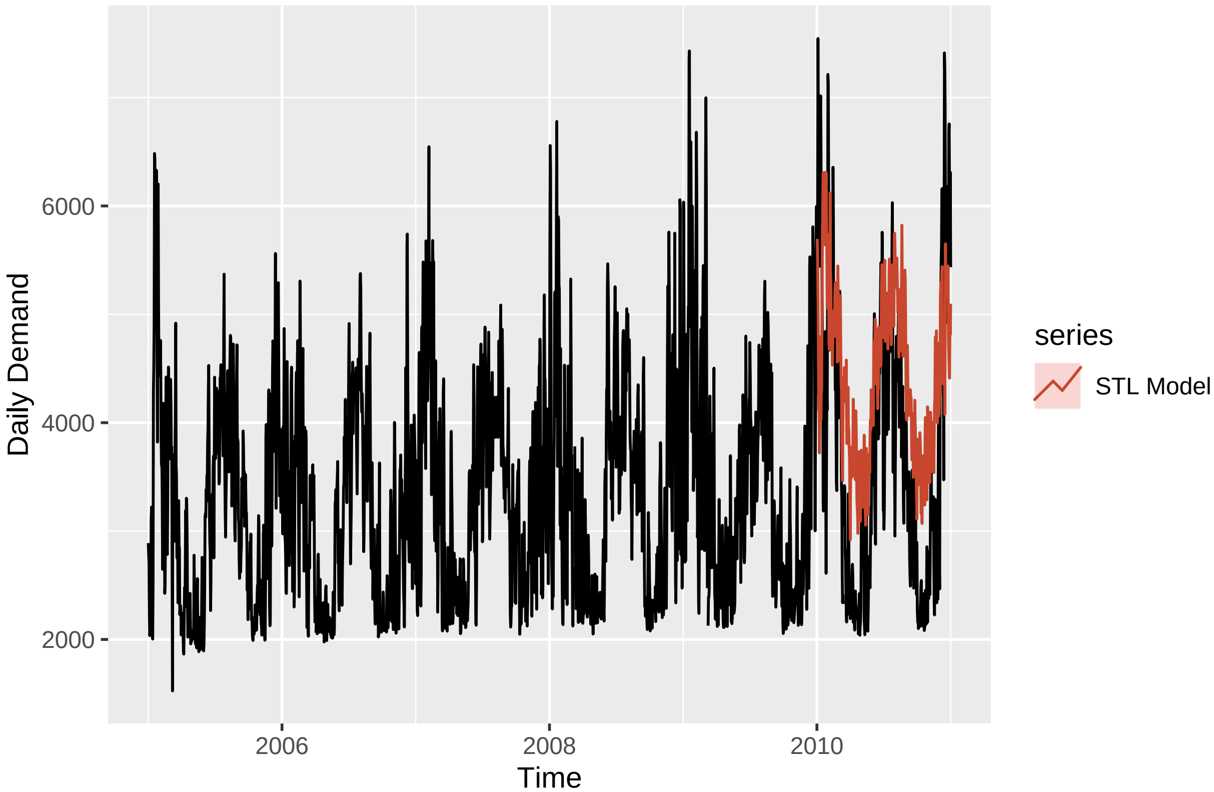
##

Model 1 STL + ETS: Forecast 2010

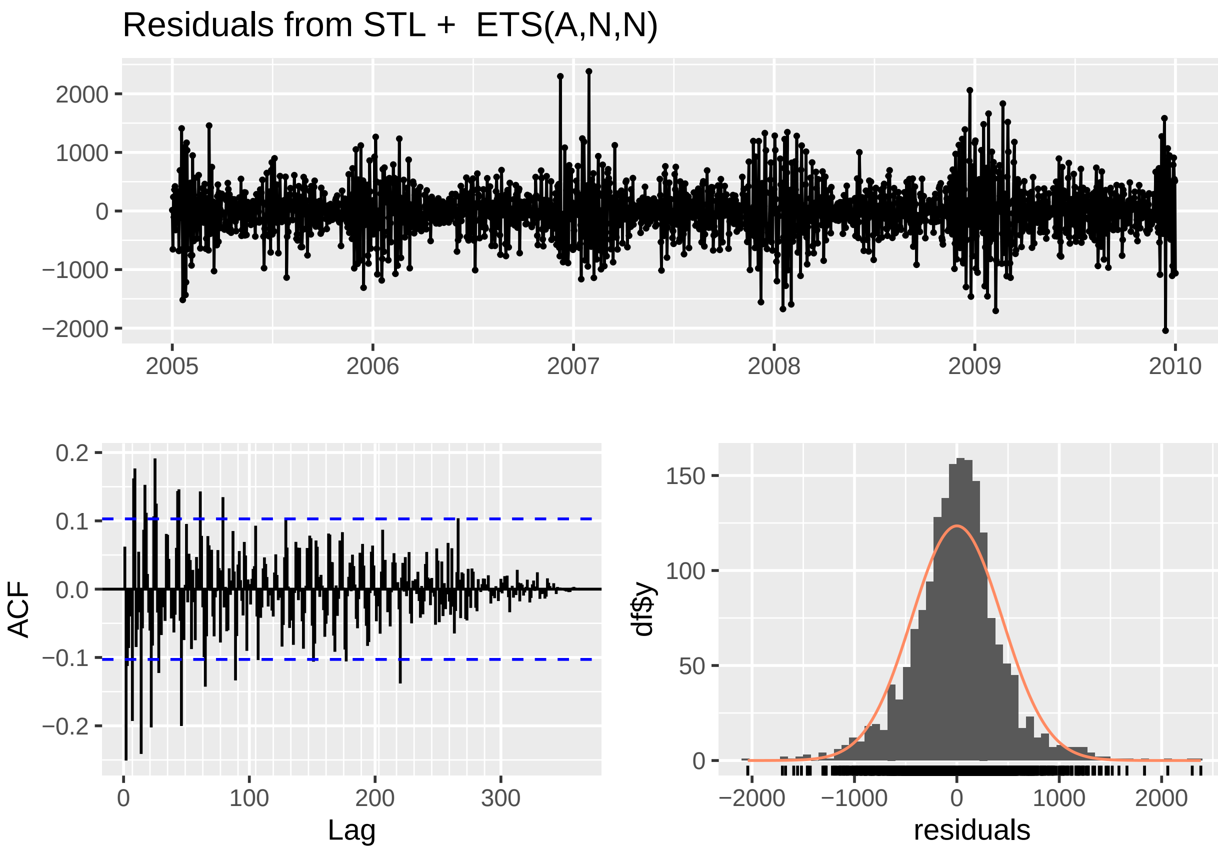
```
#Fit and forecast STL model
ETS_model <-  stlf(ts_daily,h=365)

#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")
```




```
#Plot the residuals

checkresiduals(ETS_model)
```

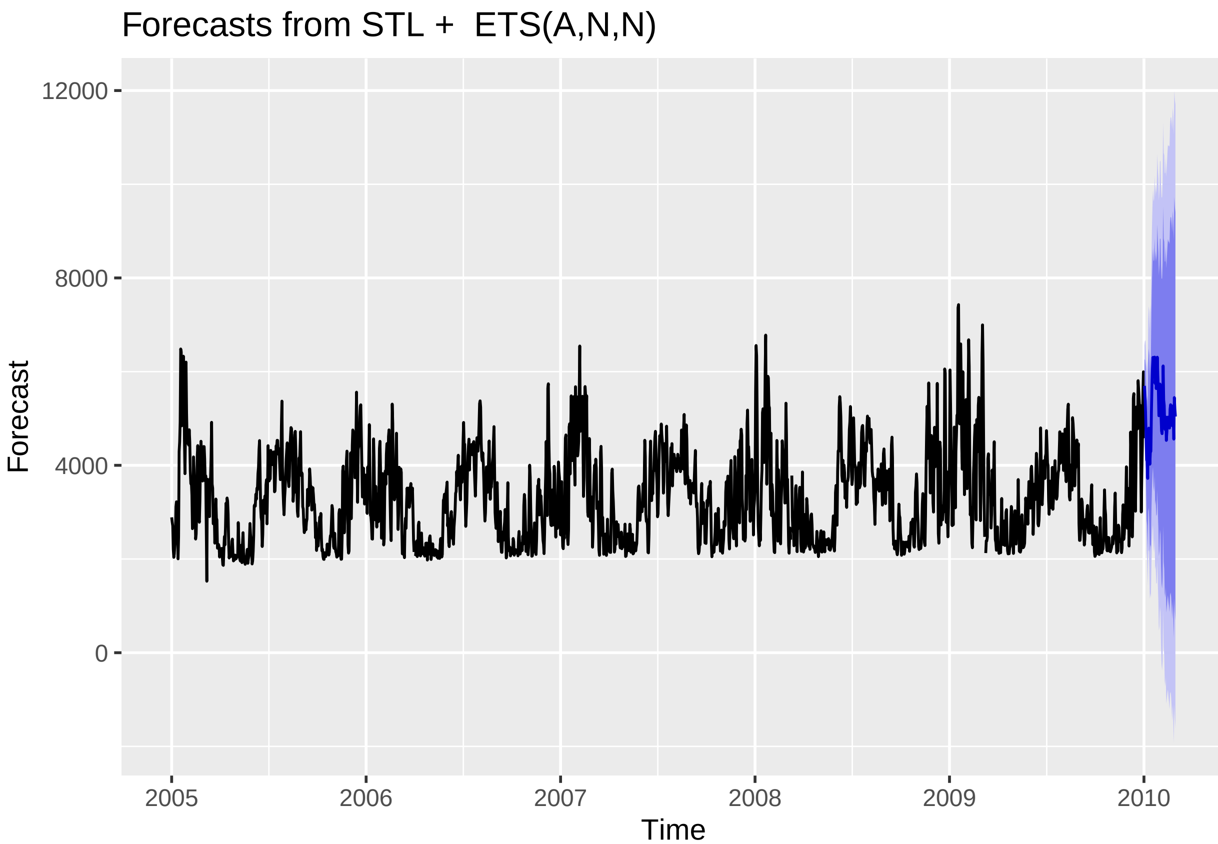



```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 1309.7, df = 363, p-value < 2.2e-16
##
## Model df: 2.   Total lags used: 365
```

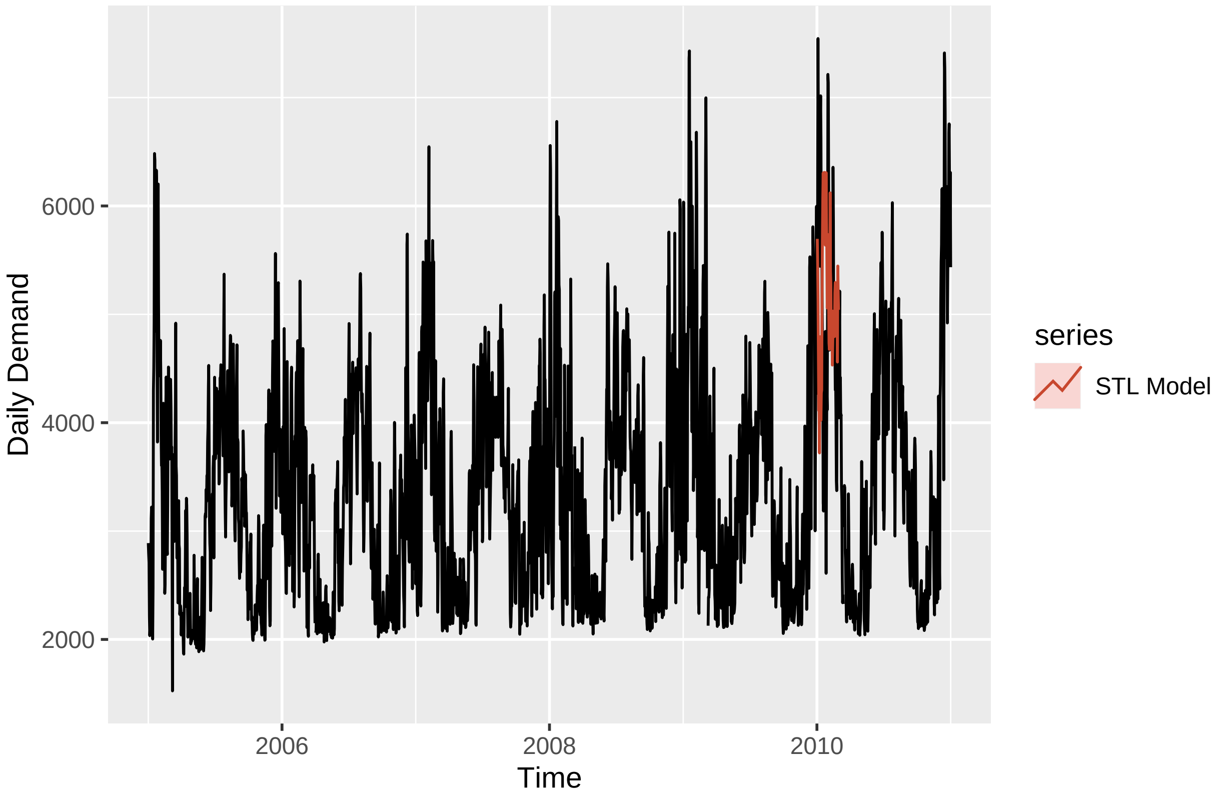
```
#Check accuracy of model
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
ETS_scores <- accuracy(ETS_model$mean,observed)
print(ETS_scores)
```

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

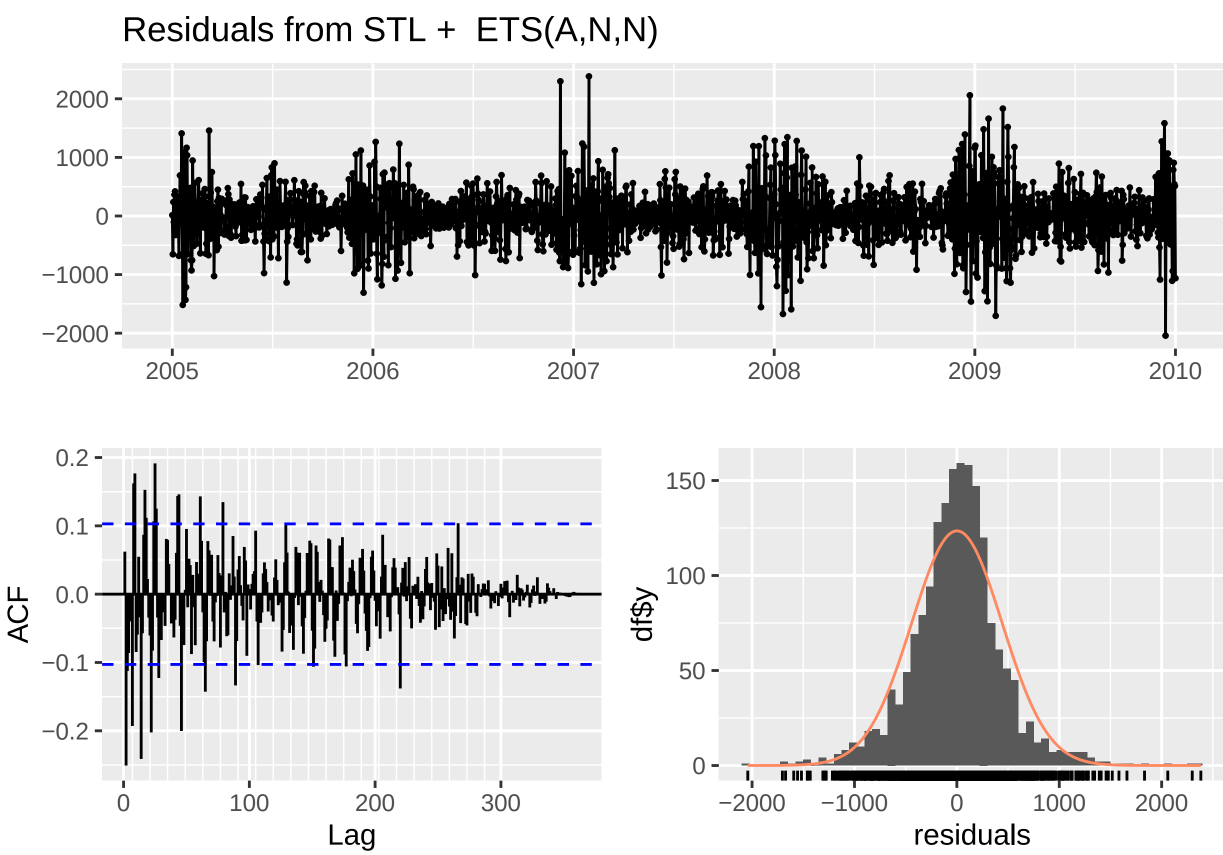
```
# Changed forecast

#Fit and forecast STL model
ETS_model <-  stlf(ts_daily,h=59)

#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)
```



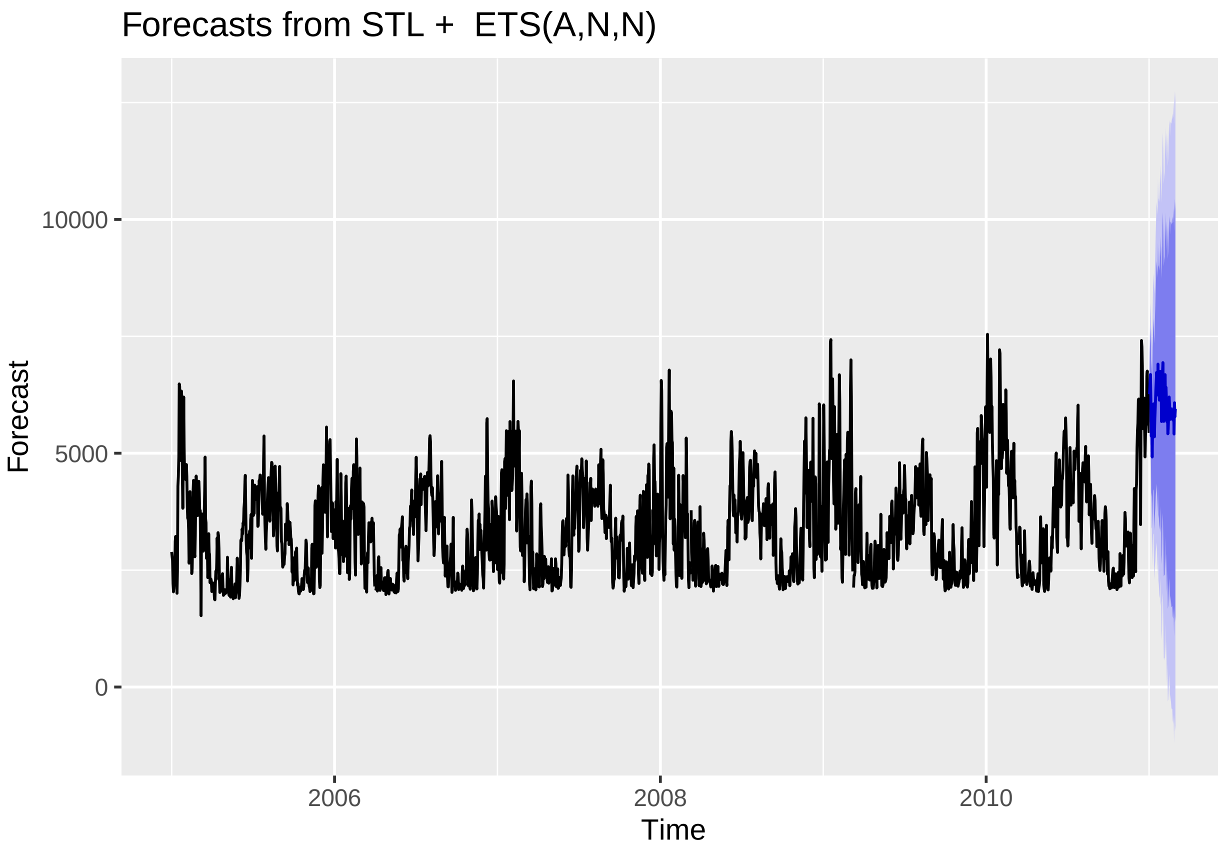

```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 1309.7, df = 363, p-value < 2.2e-16
##
## Model df: 2.   Total lags used: 365
```

```
#Check accuracy of model
n_for <- 59
observed <- df_processed2010[1827:1885, "Daily_data"]
ETS_scores <- accuracy(ETS_model$mean,observed)
print(ETS_scores)
```

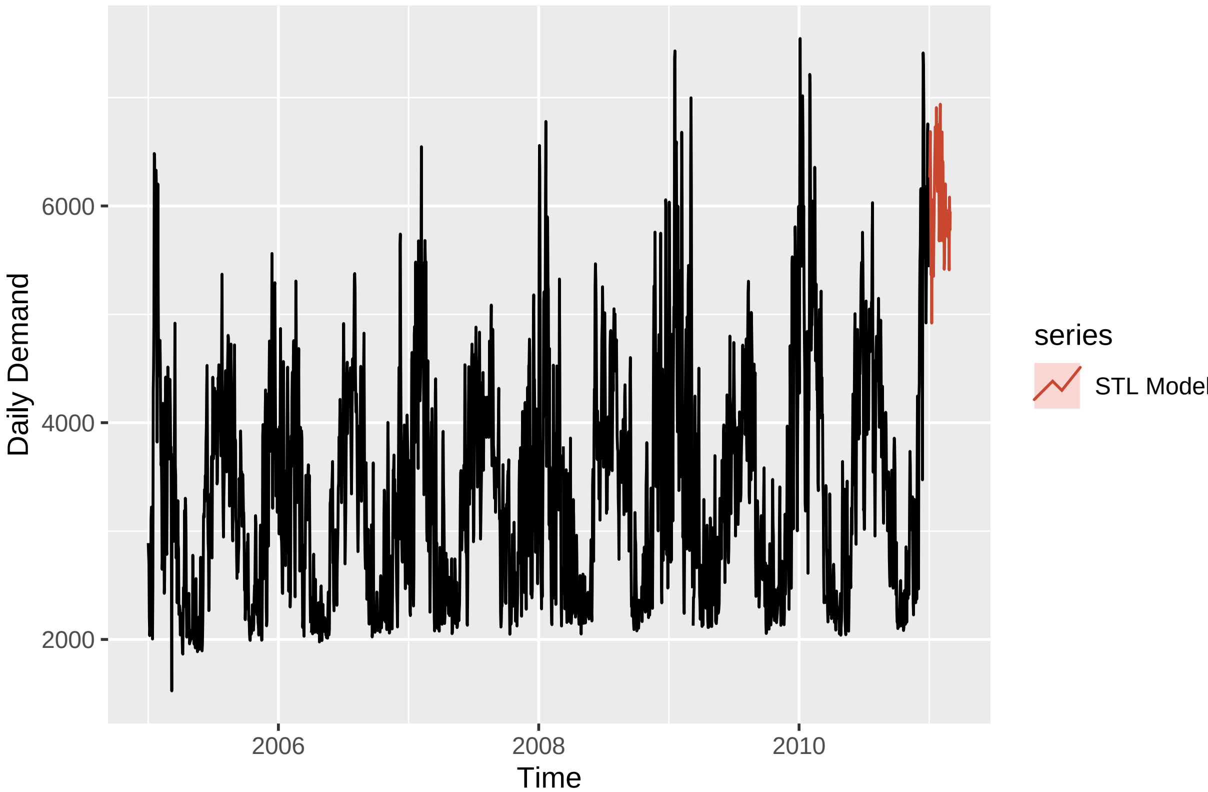
```
##                 ME     RMSE      MAE       MPE     MAPE
## Test set -103.0303 1547.348 1280.656 -9.574082 28.27524
```

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")
```




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



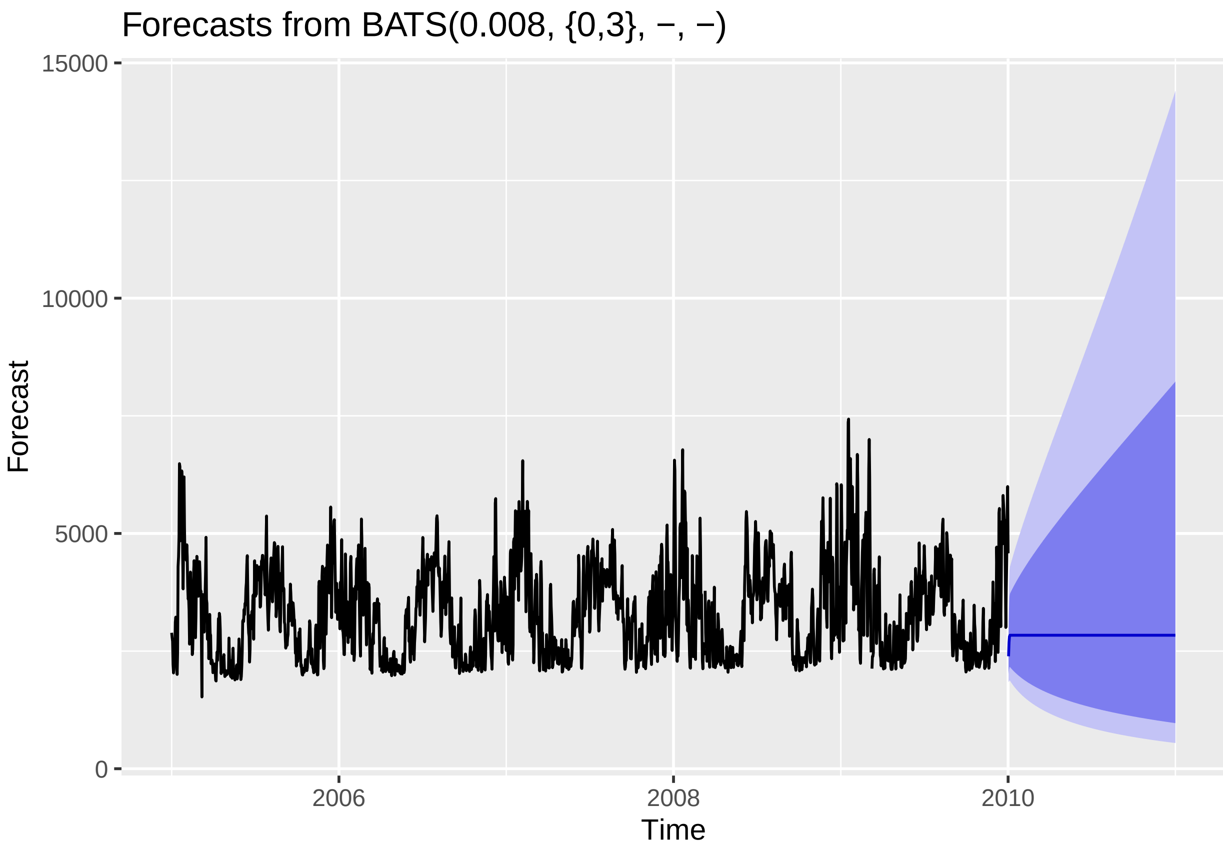

```
#Check accuracy of model
n_for <- 59
observed <- df_processed2010[(nobs-n_for+1):nobs, "Daily_data"]
ETS_scores_for <- accuracy(ETS_model2011$mean,observed)
print(ETS_scores_for)
```

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

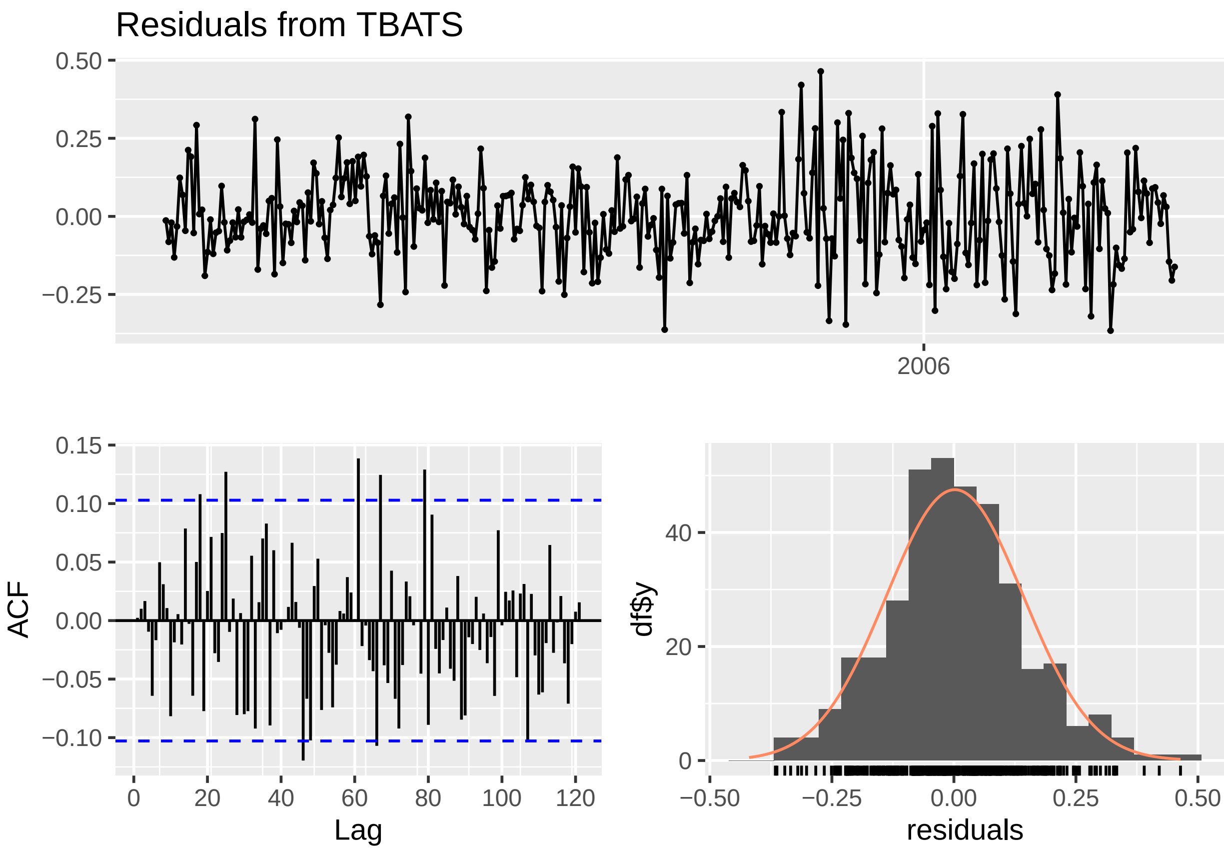
Model 2 TBATS: Forecast 2010

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_for) + ylab("Forecast")
```

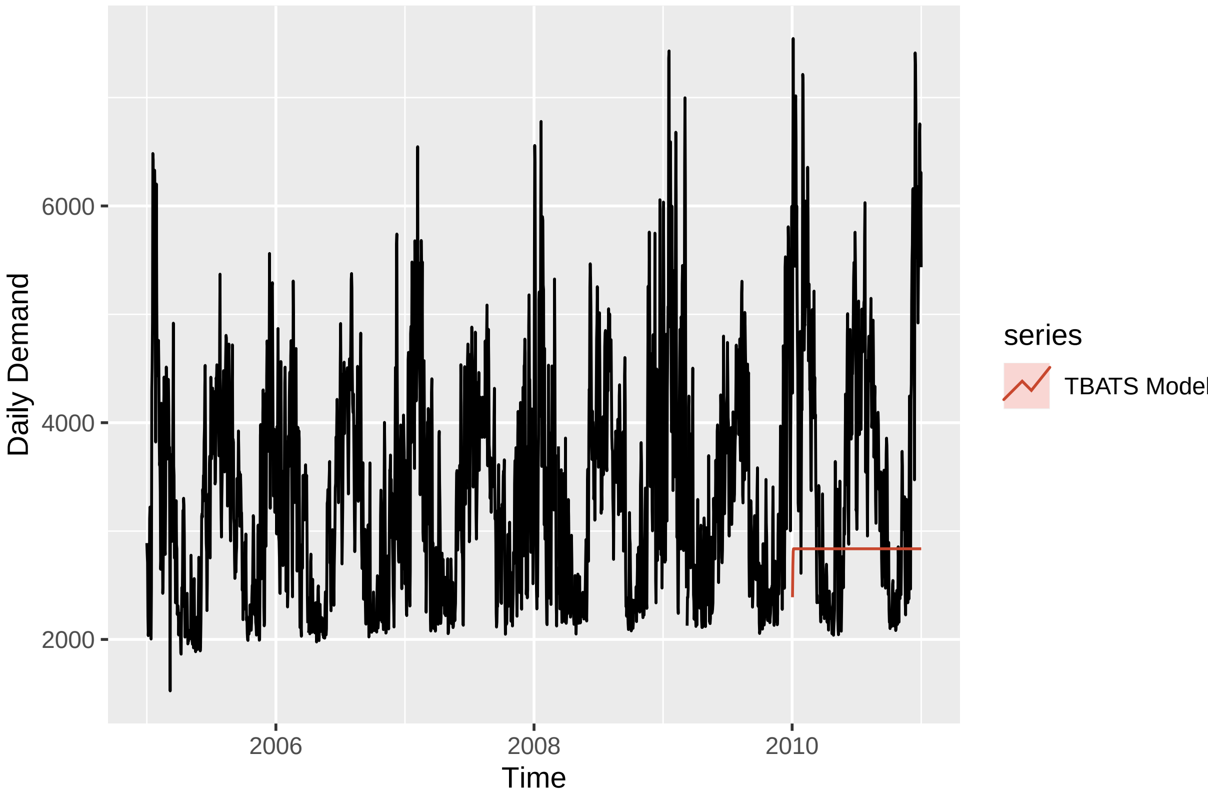



```
#Plot the residuals

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
```

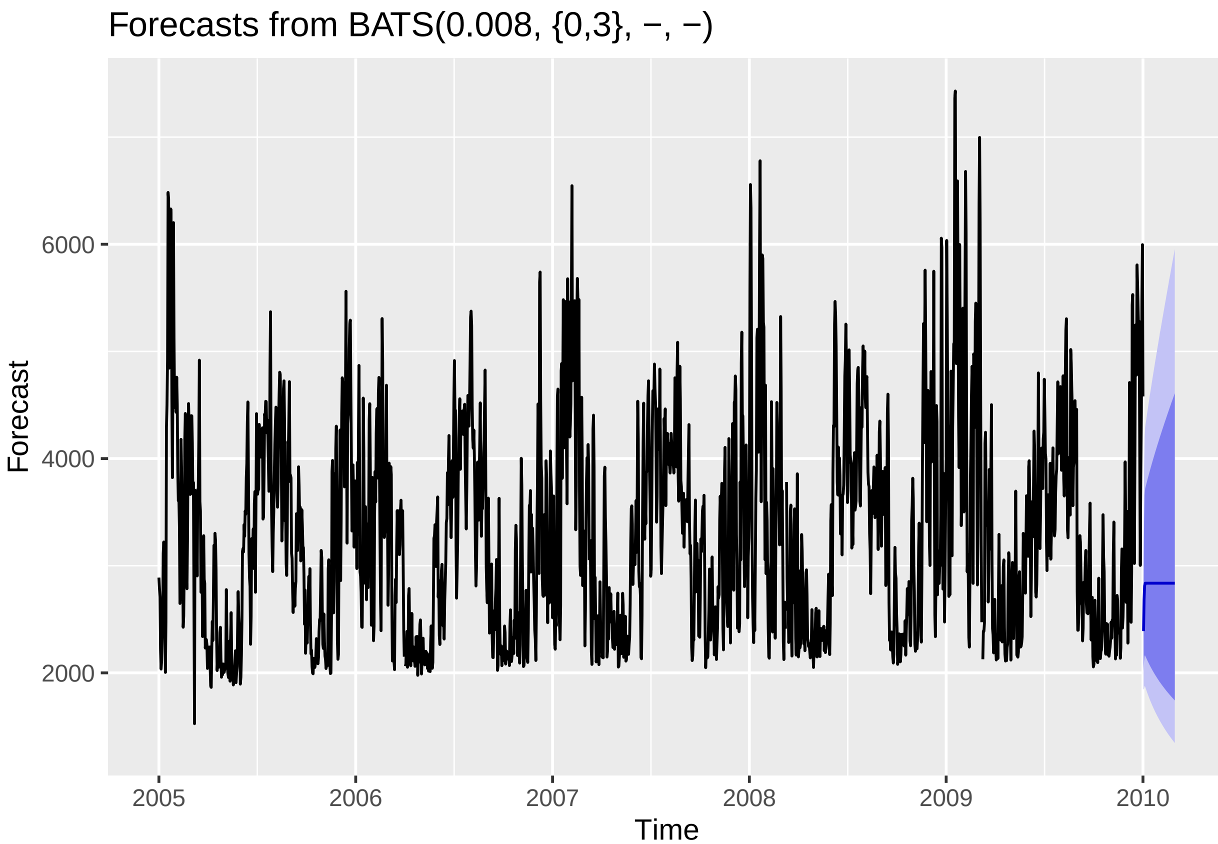
```
#Plot model + observed data
autoplot(ts_daily2010) +
  autolayer(TBATS_for, series="TBATS Model",PI=FALSE) +
  ylab("Daily Demand")
```



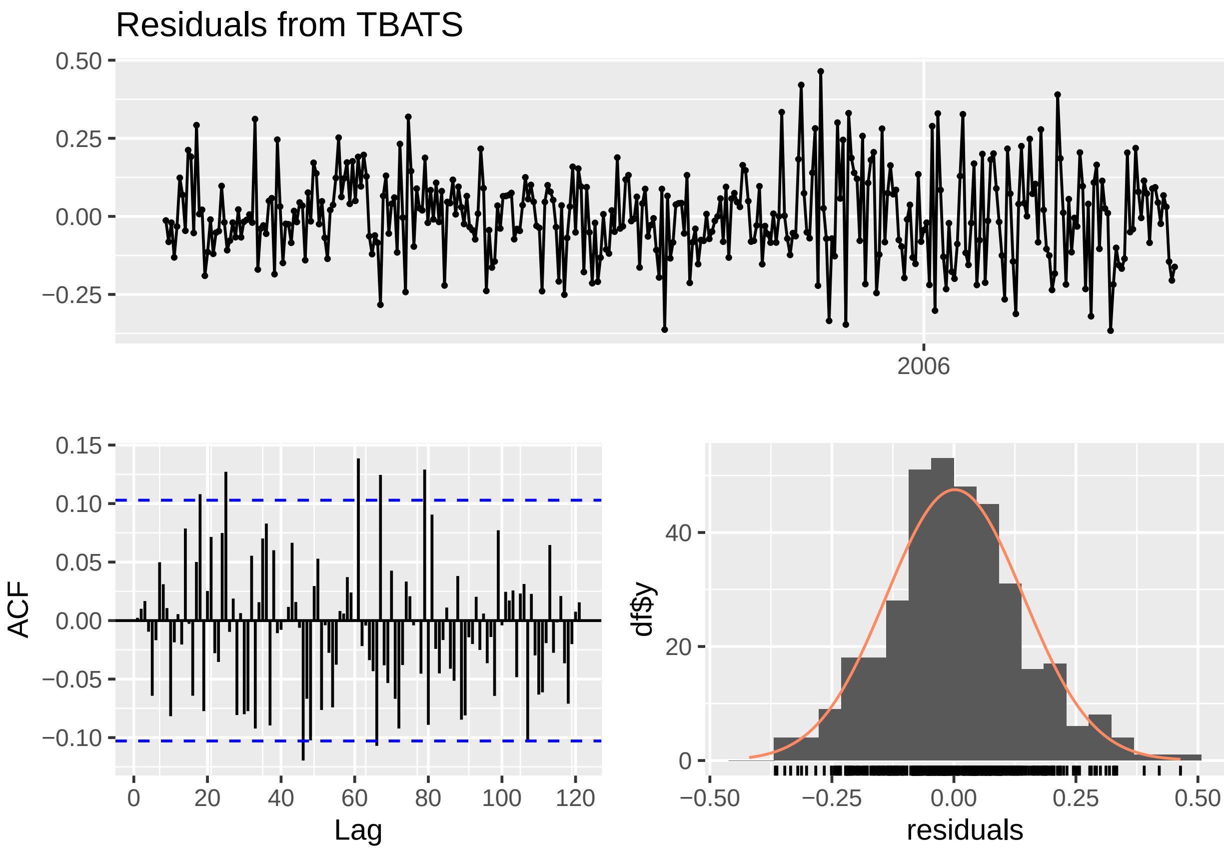

```
#Check accuracy of model
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
TBATS_scores <- accuracy(TBATS_for$mean,observed)
print(TBATS_scores)
```

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

```
# Changed version

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

#forecast
TBATS_for <- forecast(TBATS_model,h=59)

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

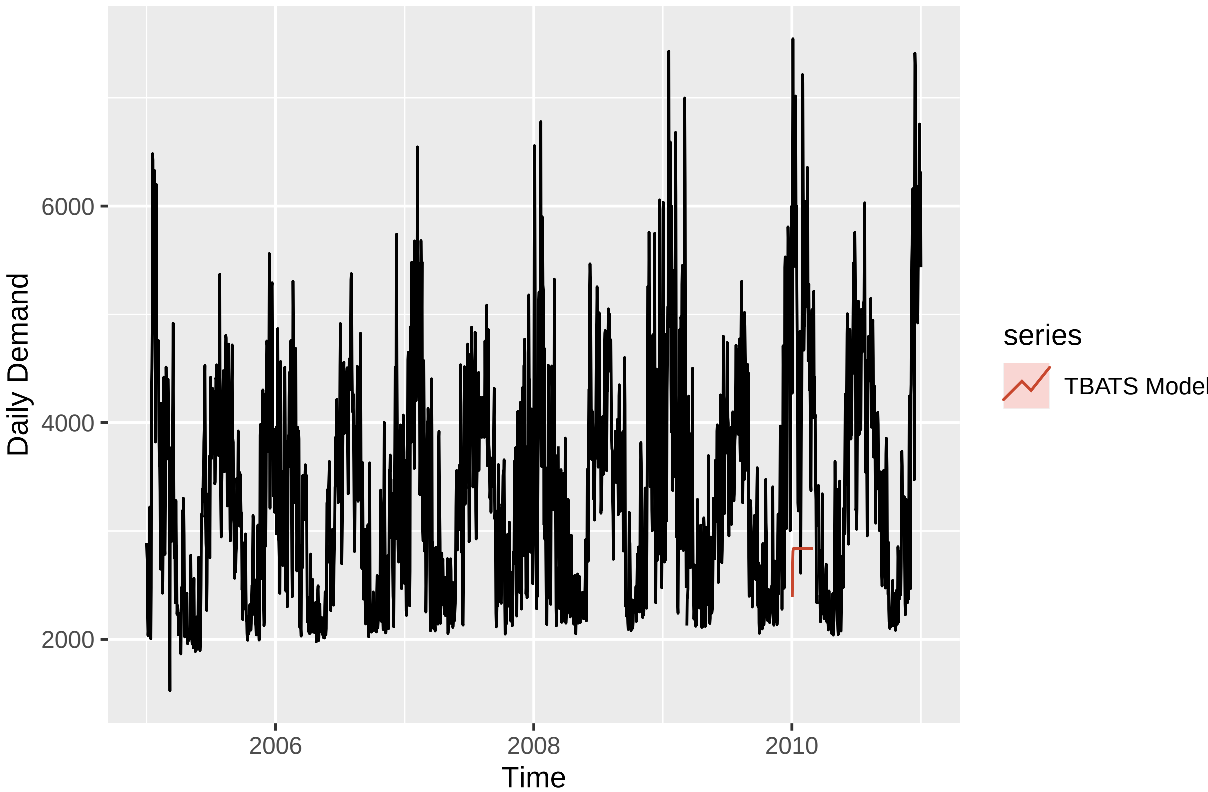



```
#Plot the residuals

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
#Check accuracy of model
n_for <- 59
observed <- df_processed2010[1827:1885, "Daily_data"]
TBATS_scores <- accuracy(TBATS_for$mean,observed)
print(TBATS_scores)
```

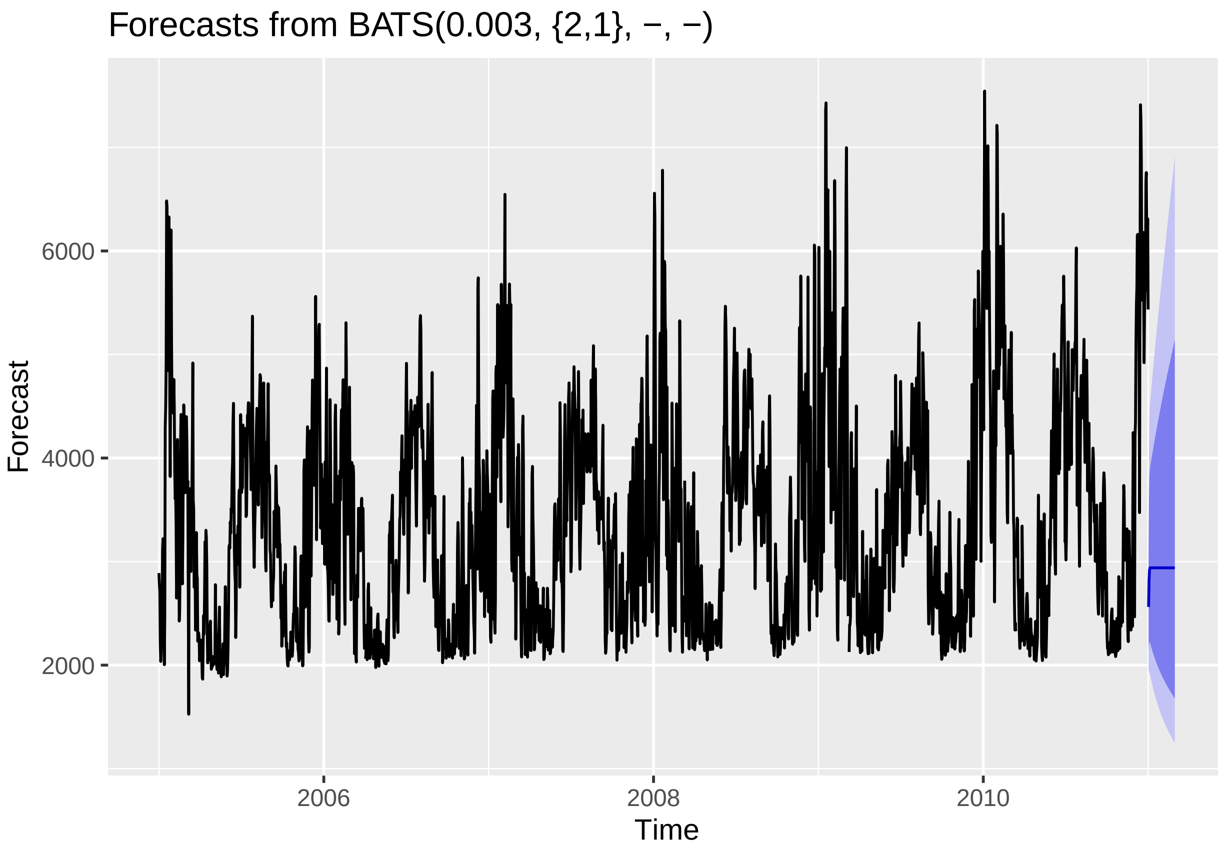
```
##                ME     RMSE      MAE      MPE     MAPE
## Test set 2226.34 2499.097 2234.019 40.94677 41.24095
```

Model 2 TBATS: Forecast 2011

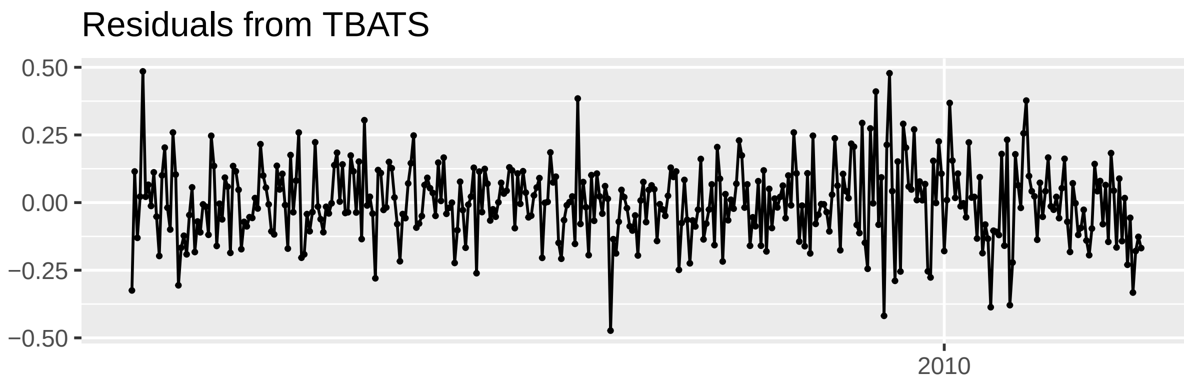
```r
#Fit and forecast TBATS model
TBATS_model2011 <-  tbats(ts_daily2010)
```

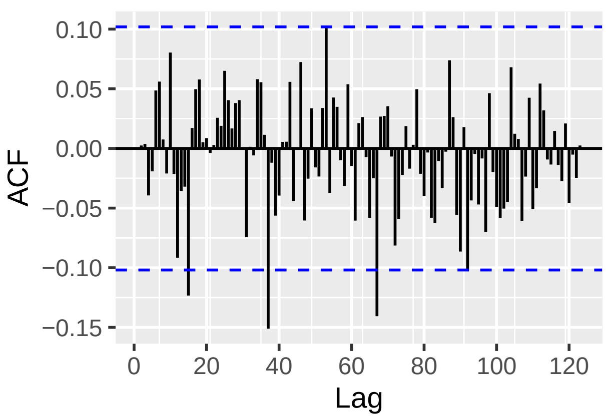
```
## Warning in tbats(ts_daily2010): Missing values encountered. Using longest
## contiguous portion of time series
```

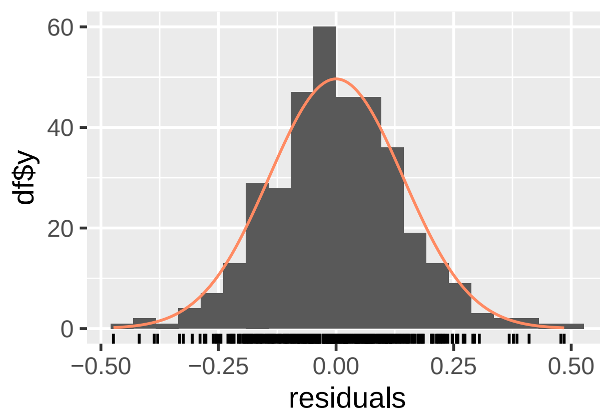
```r
#forecast
TBATS_for2011 <- forecast(TBATS_model2011,h=59)

#Plot foresting
autoplot(TBATS_for2011) + ylab("Forecast")
```




```
#Plot the residuals

checkresiduals(TBATS_model2011)
```

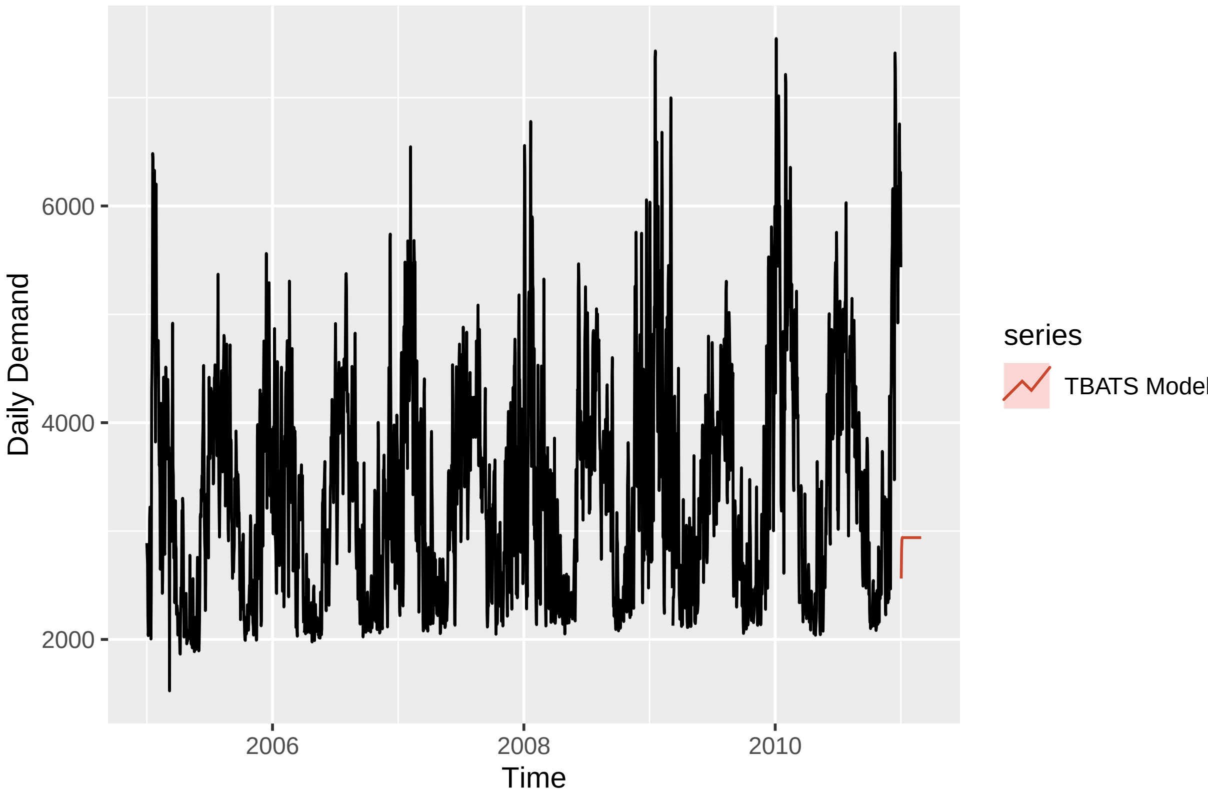







```
##
##  Ljung-Box test
##
## data:  Residuals from TBATS
## Q* = 73.551, df = 69, p-value = 0.3315
##
## Model df: 5.   Total lags used: 74
```

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

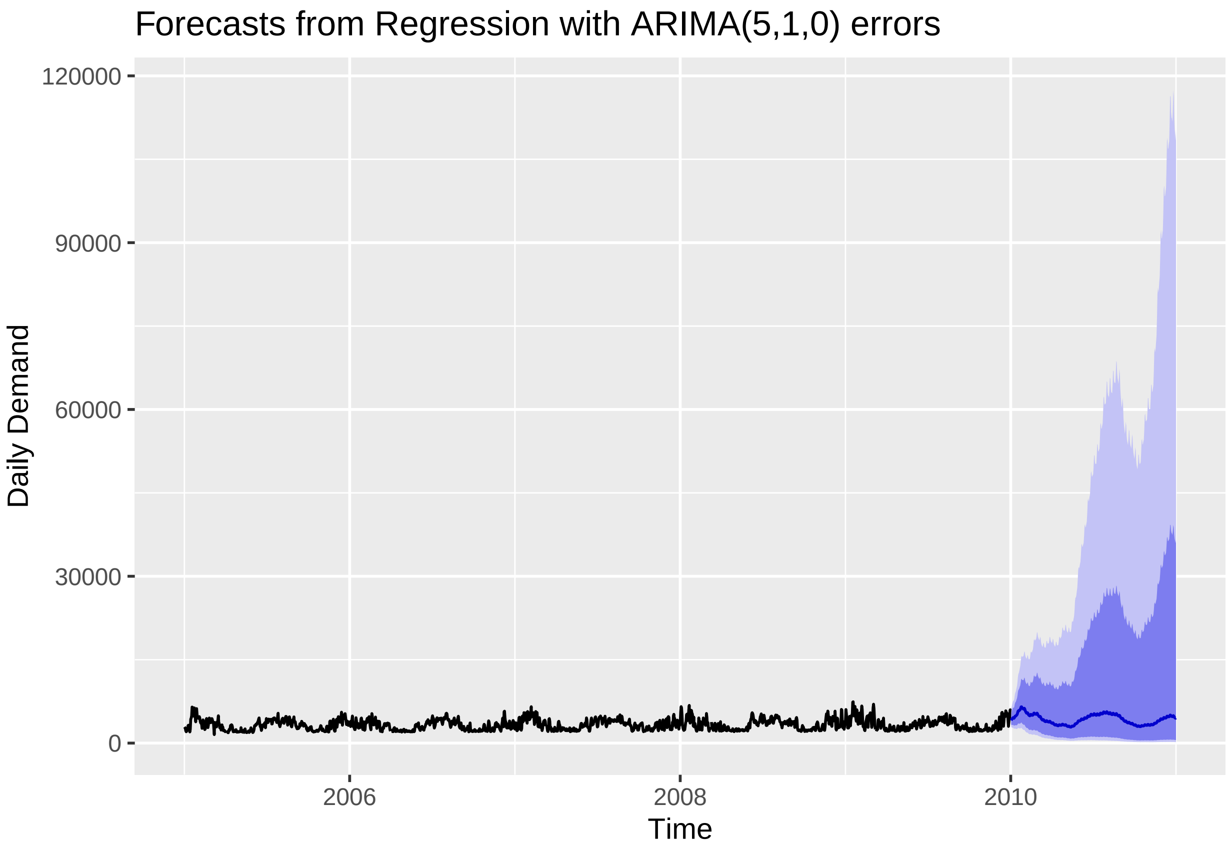



```
#Check accuracy of model
n_for <- 59
observed <- df_processed2010[(nobs-n_for+1):nobs, "Daily_data"]
TBATS_scores_for <- accuracy(TBATS_for$mean,observed)
print(TBATS_scores_for)
```

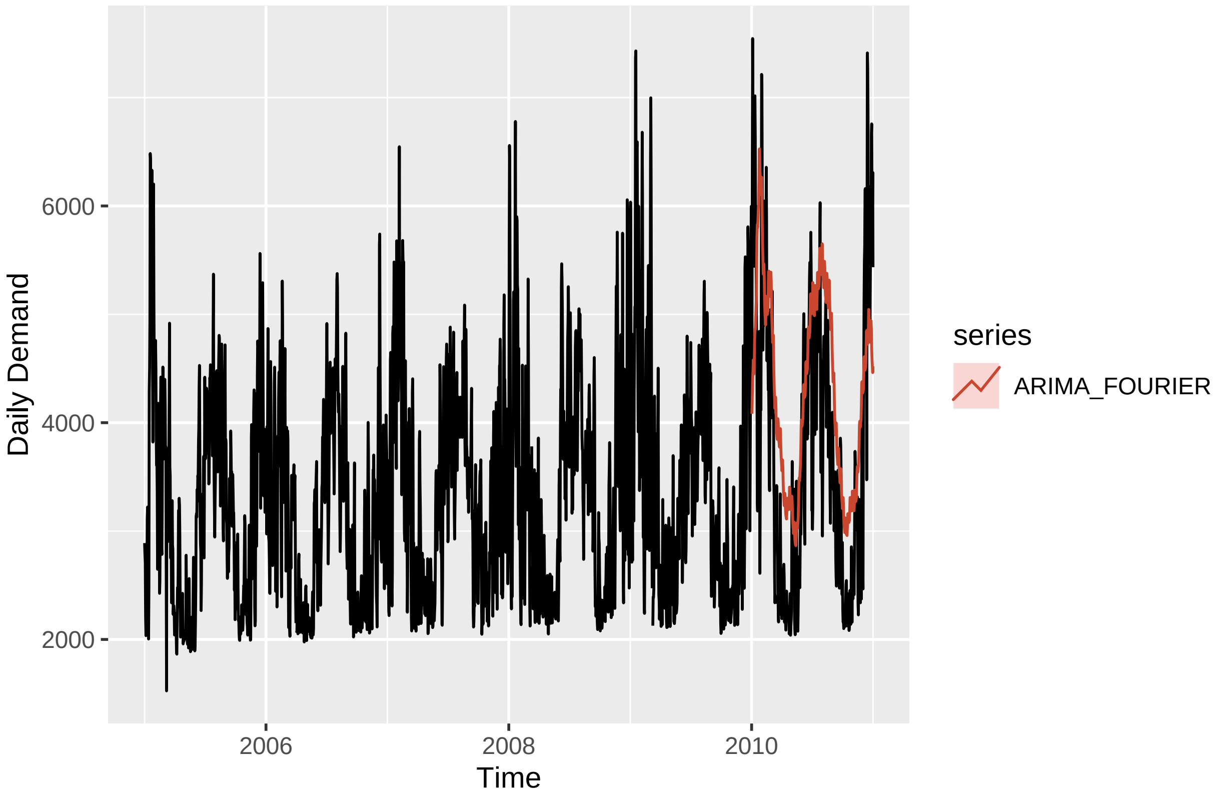
```
##                 ME     RMSE      MAE      MPE     MAPE
## Test set 681.5893 1327.303 964.6684 11.34867 23.42342
```

Model 3 ARIMA + FOURIER terms: Forecast 2010

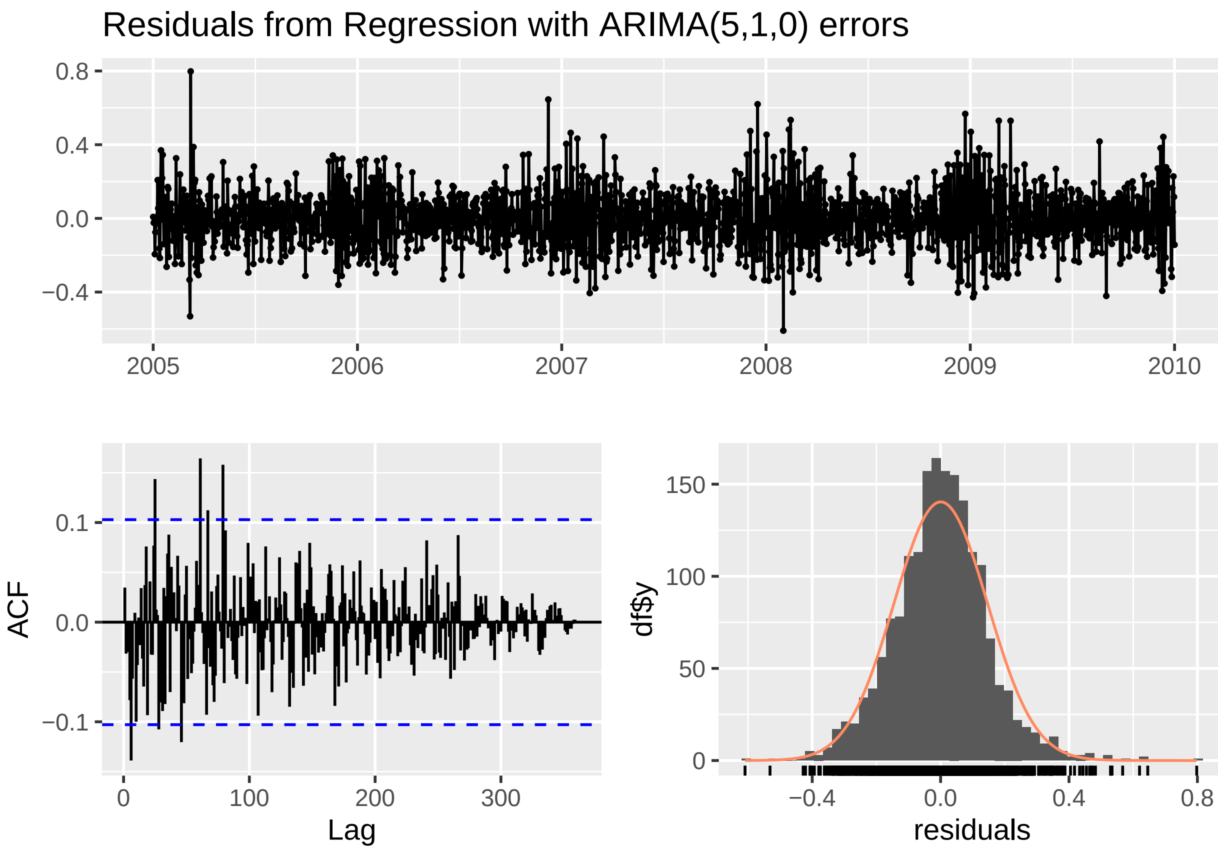
```
#Fit and forecast TBATS model
ARIMA_Four_model <- auto.arima(ts_daily,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_daily,
                                          K=c(2,12))
                             )

#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)
```



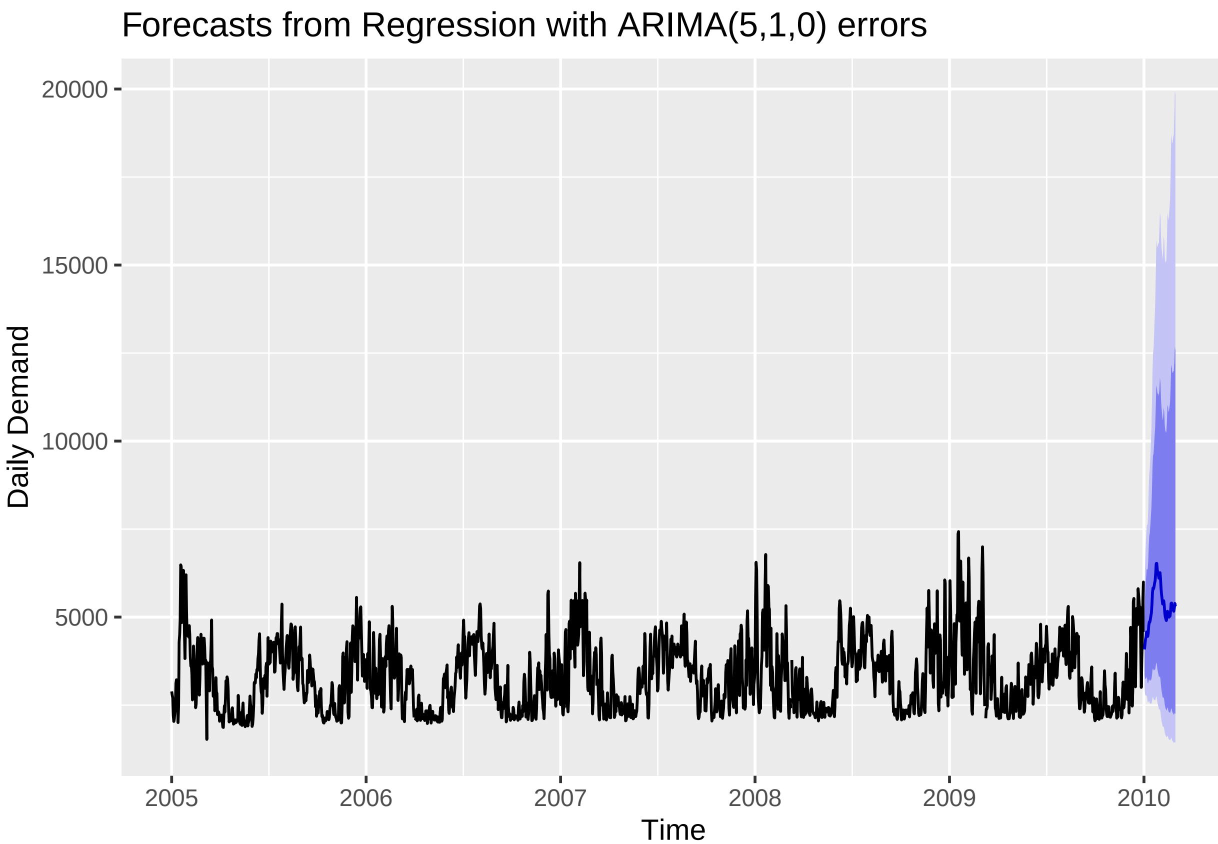

```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(5,1,0) errors
## Q* = 462.43, df = 360, p-value = 0.0002078
##
## Model df: 5.   Total lags used: 365
```

```
#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)
```

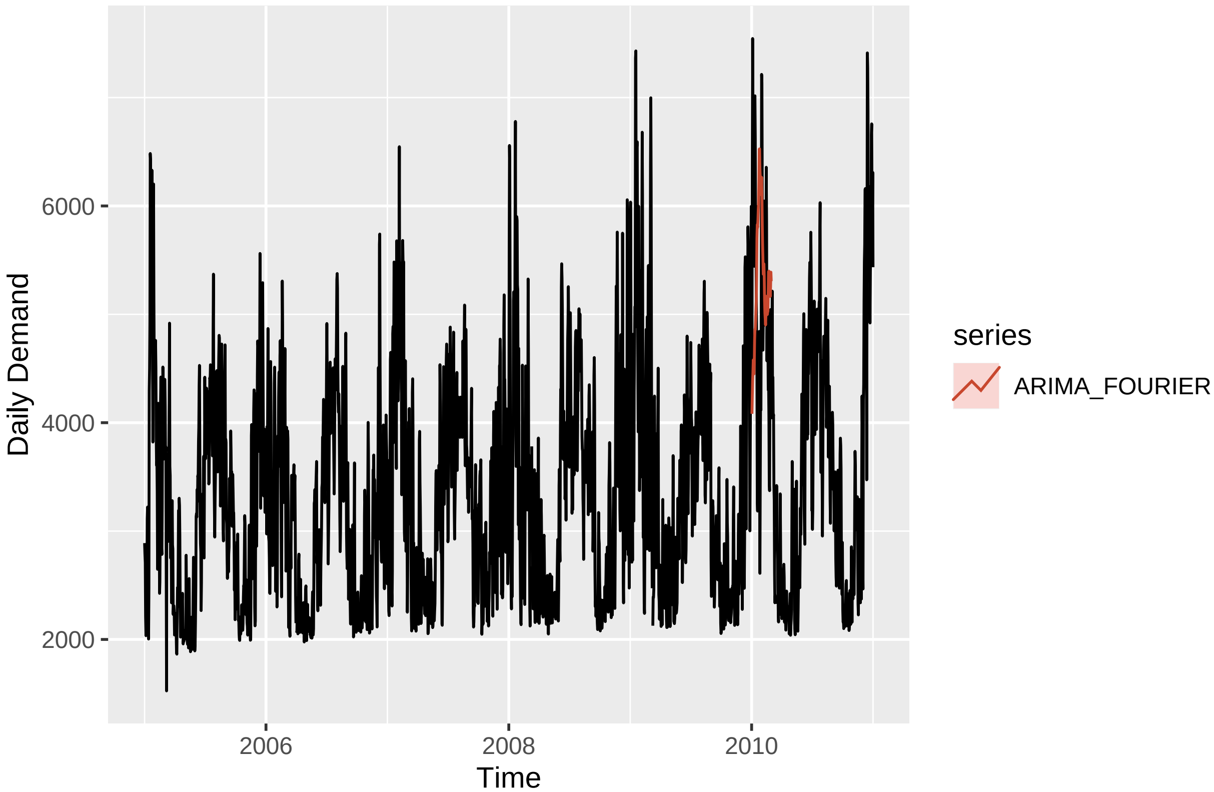
```
##                 ME     RMSE     MAE      MPE     MAPE
## Test set -853.8232 1158.956 991.685 -30.4006 32.76586
```

```
# Changed version

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,
```

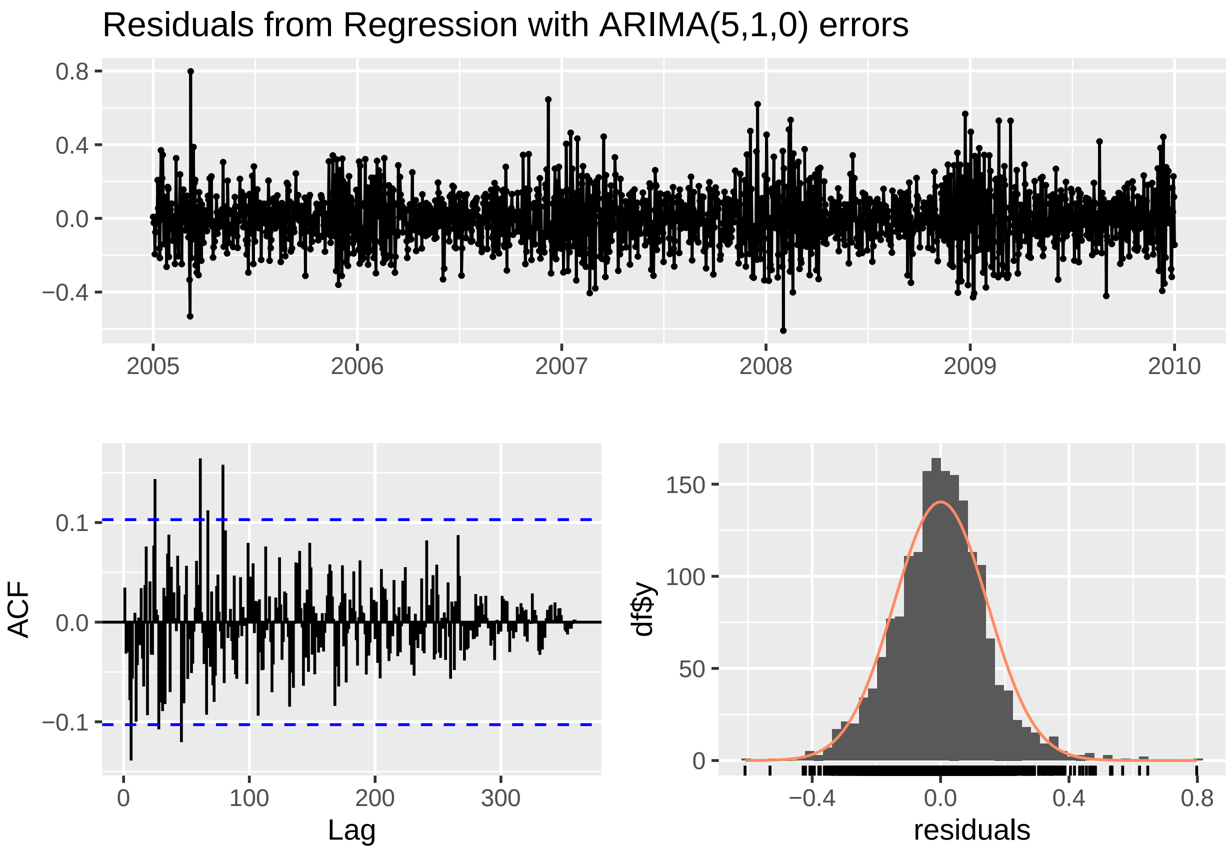
```
                                  xreg=fourier(ts_daily,
                                               K=c(2,12),
                                               h=59),
                                  h=59
                                  )

#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
```

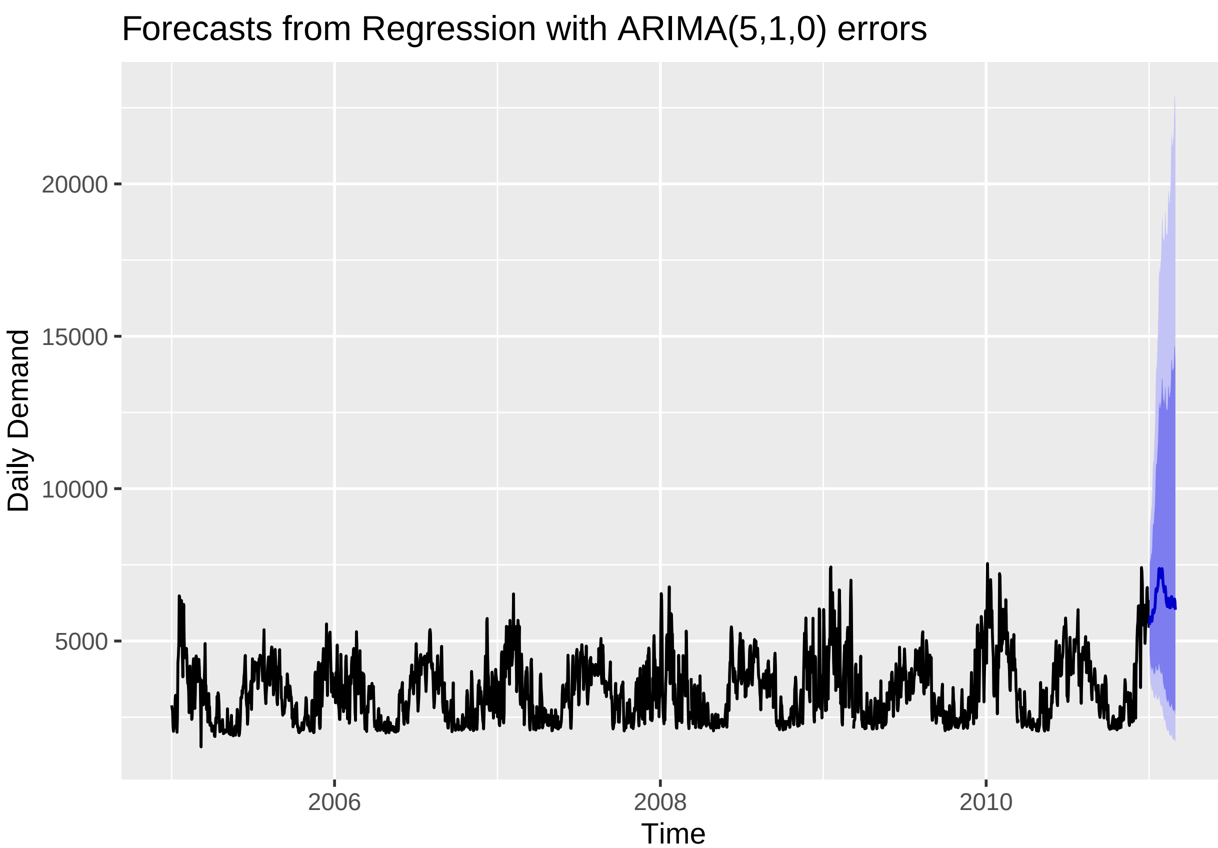
```
#Check accuracy of model
n_for <- 59
observed <- df_processed2010[1827:1885, "Daily_data"]
ARIMA_Four_scores <- accuracy(ARIMA_Four_for$mean,observed)
print(ARIMA_Four_scores)
```

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

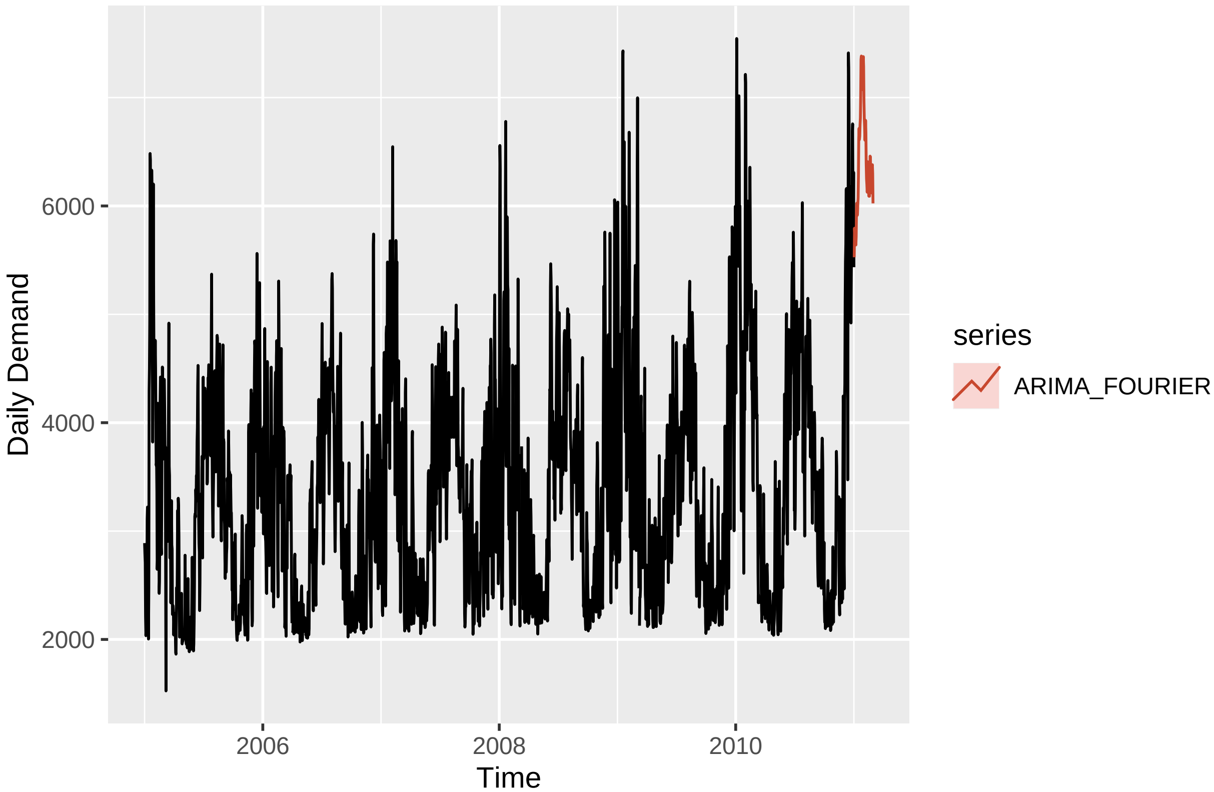
Model 3 ARIMA + FOURIER terms: Forecast 2011

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

```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")
```



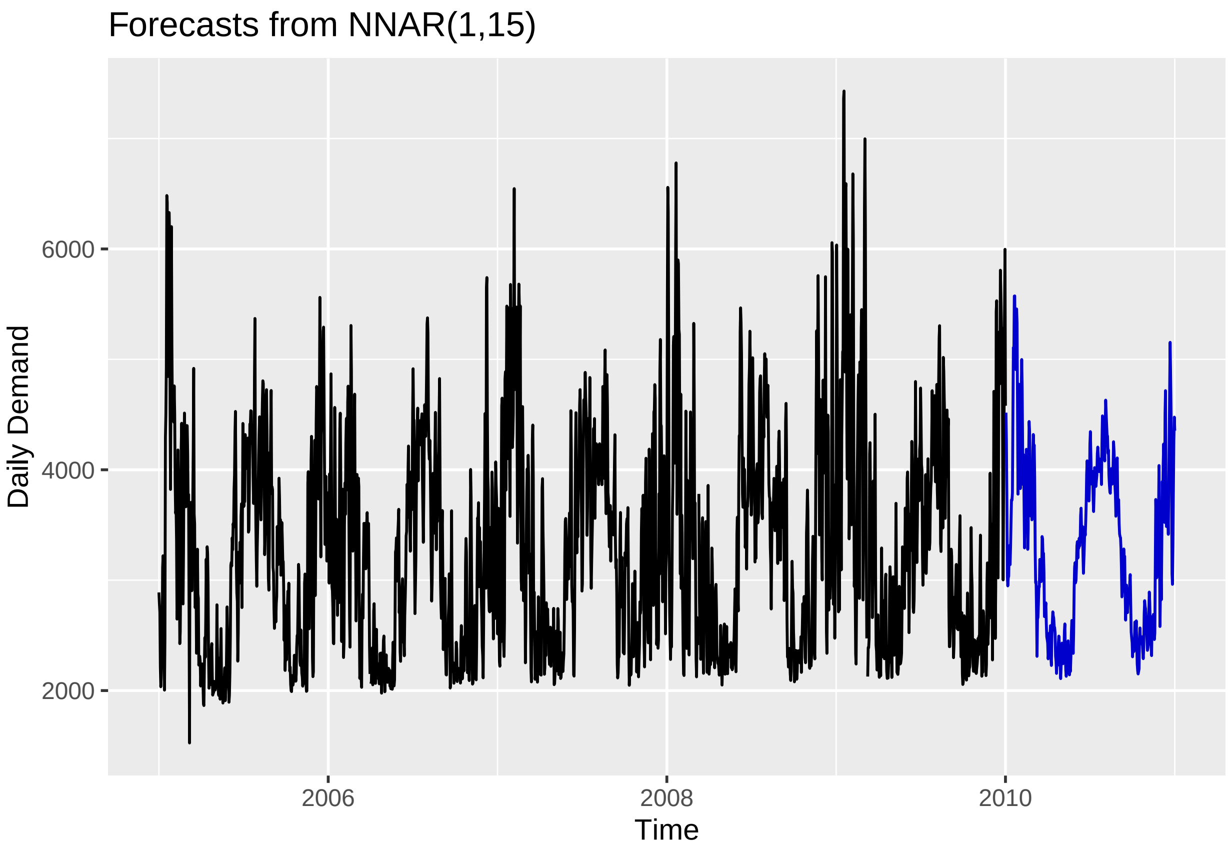

```
#Check accuracy of model
n_for <- 59
observed <- df_processed2010[(nobs-n_for+1):nobs, "Daily_data"]
ARIMA_Four_scores_for <- accuracy(ARIMA_Four_for$mean,observed)
print(ARIMA_Four_scores_for)
```

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

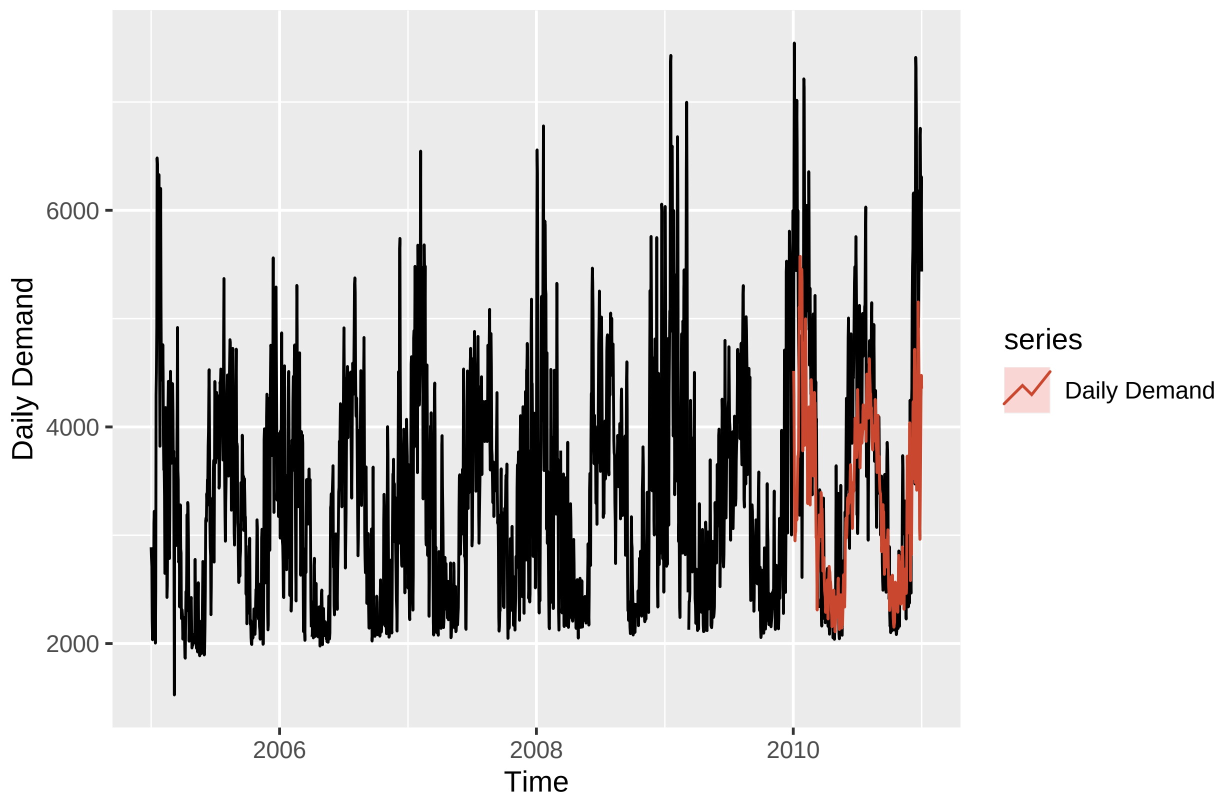
```
#print(ARIMA_Four_for2011$mean)
```

Model 4 Neural Network Time Series: Forecasts 2010

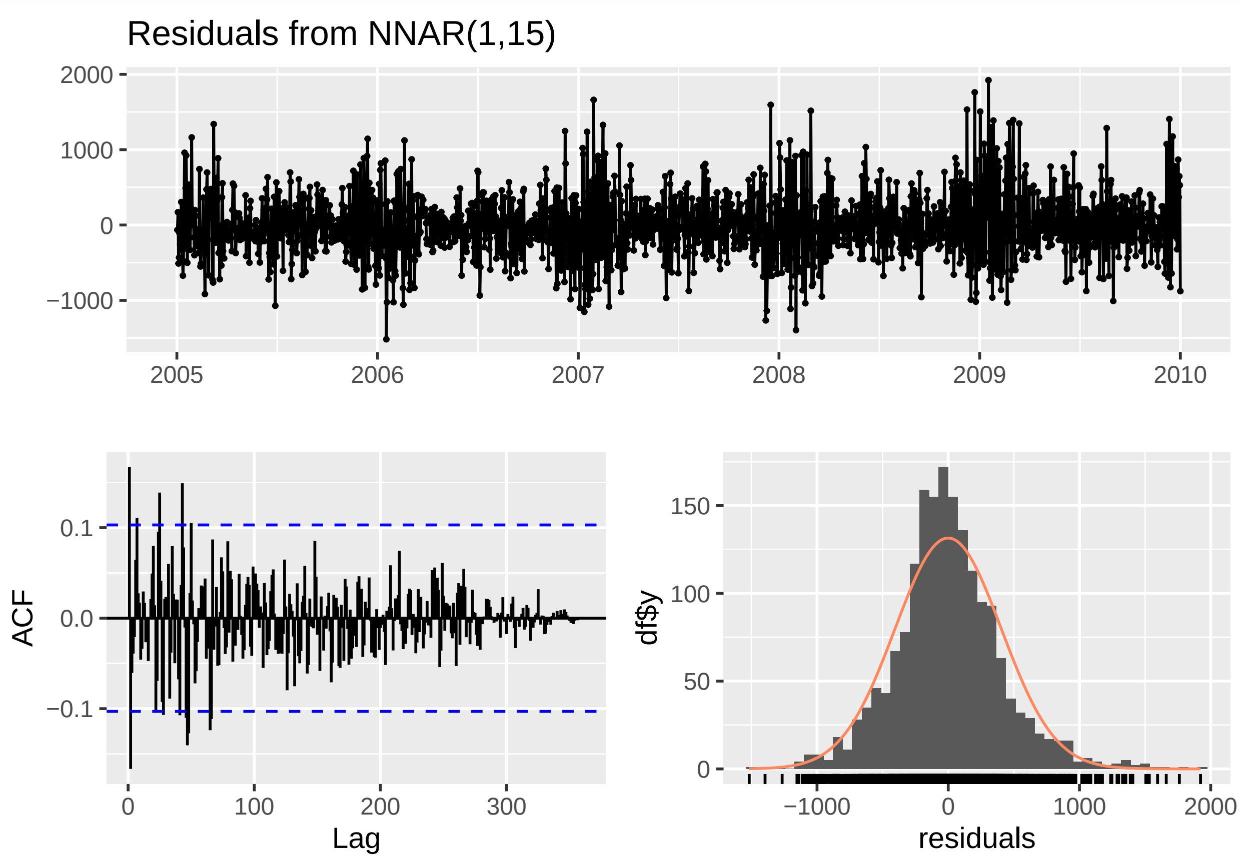
```
#NN_fit <- nnetar(ts_act_power_daily_train,p=1,P=1)
NN_model <- nnetar(ts_daily,decay=0.5, maxit=150, p=1,P=0,xreg=fourier(ts_daily, K=c(2,12)))

#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")
```




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



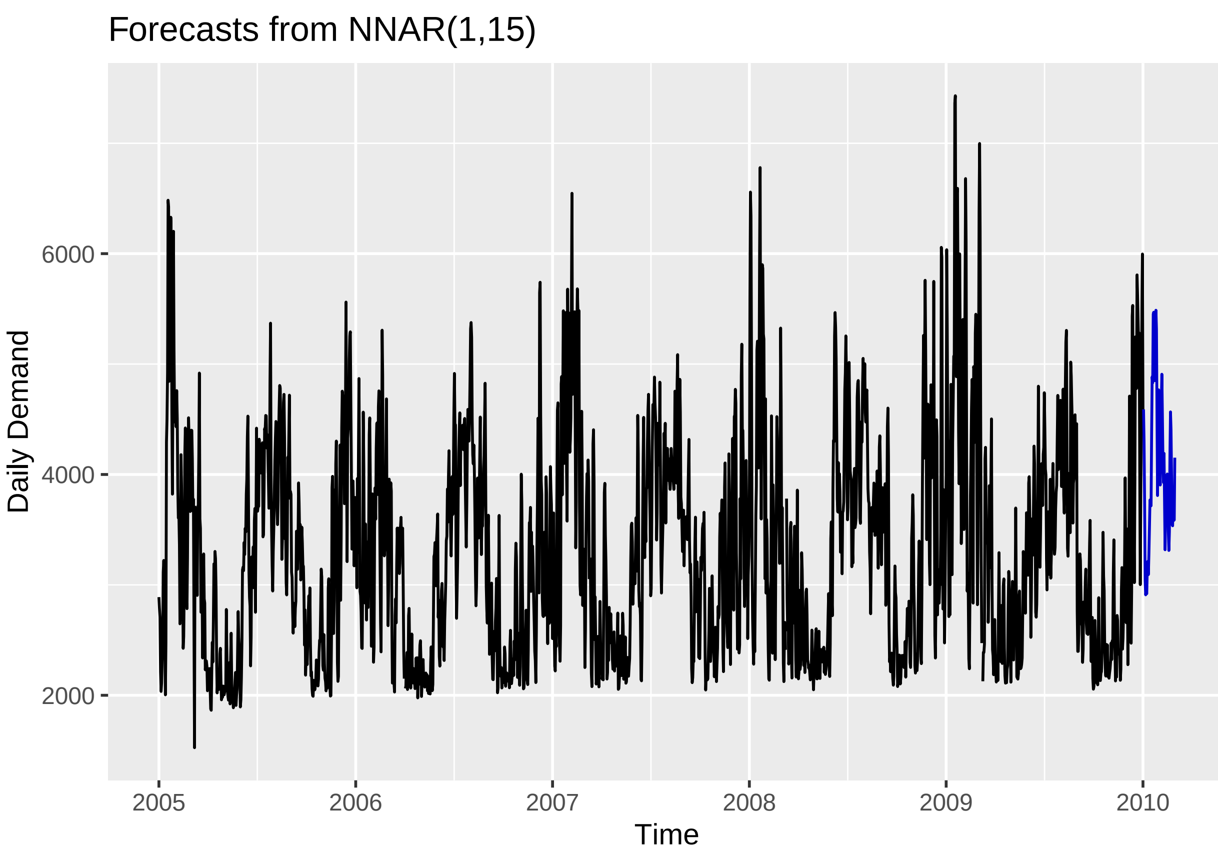

```
checkresiduals(NN_model)
```



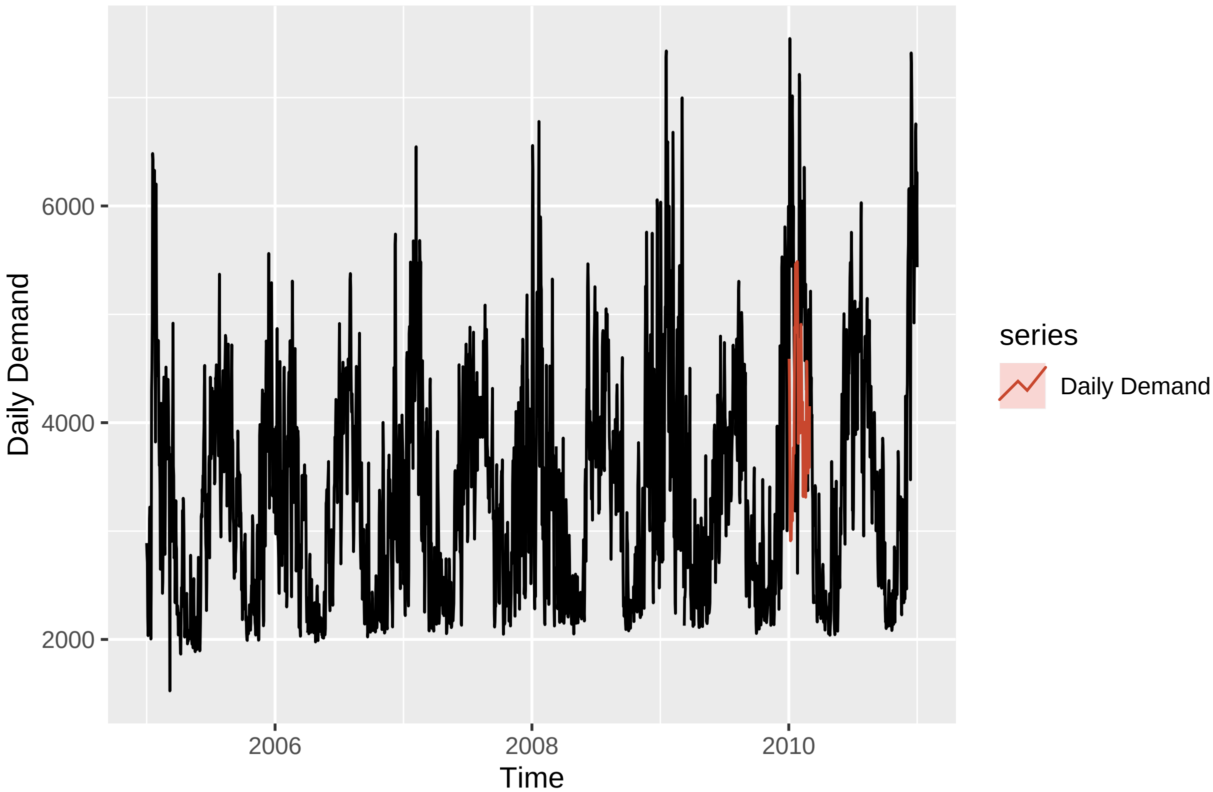
```r
#Checking error variables to decide which model fits the data the best
n_for <- 365
observed <- df_processed[(nobs-n_for+1):nobs, "Daily_data"]
NN_scores1 <- accuracy(NN_for$mean,observed)
print(NN_scores1)
```

```
##                ME     RMSE      MAE       MPE    MAPE
## Test set 127.3959 718.8671 536.2577 0.1572307 14.9895
```

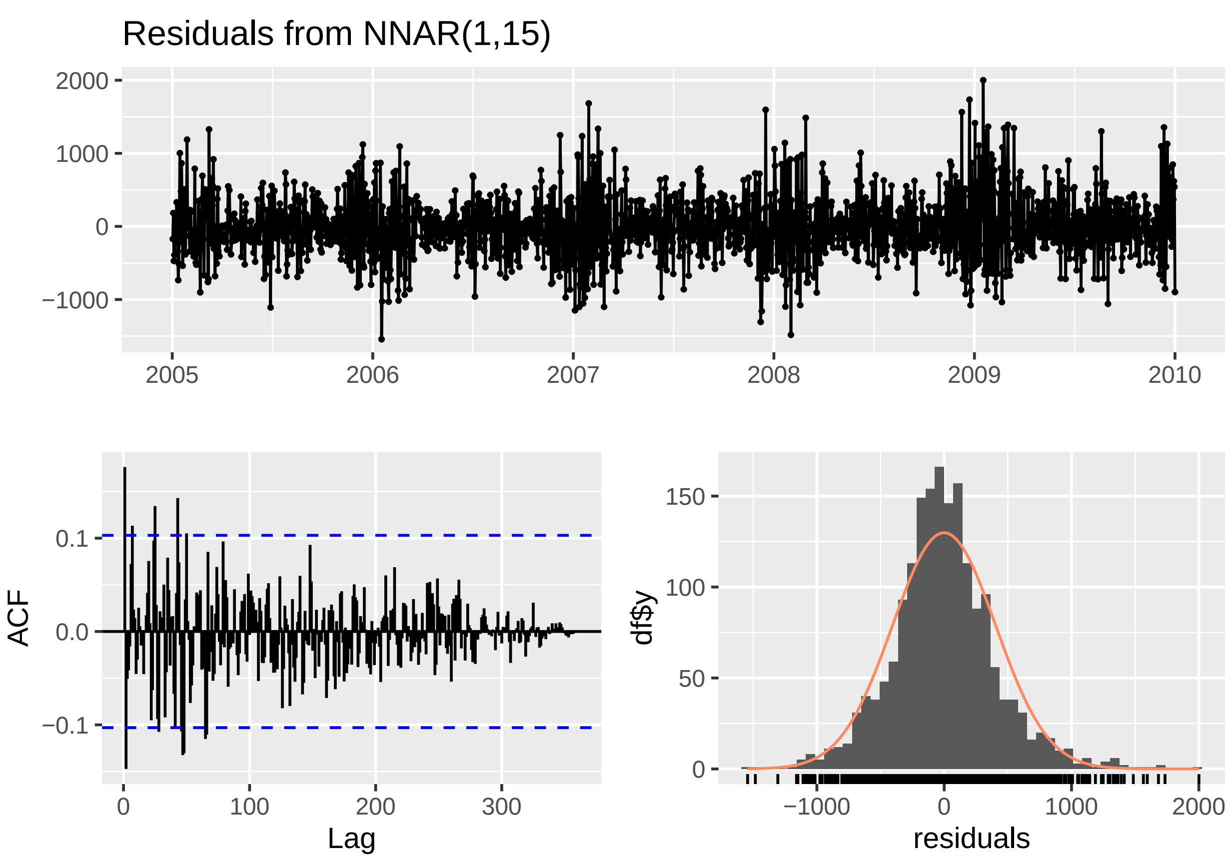
```r
# Changed version

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=59,xreg=fourier(ts_daily,
                                          K=c(2,12),h=59))

#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)
```



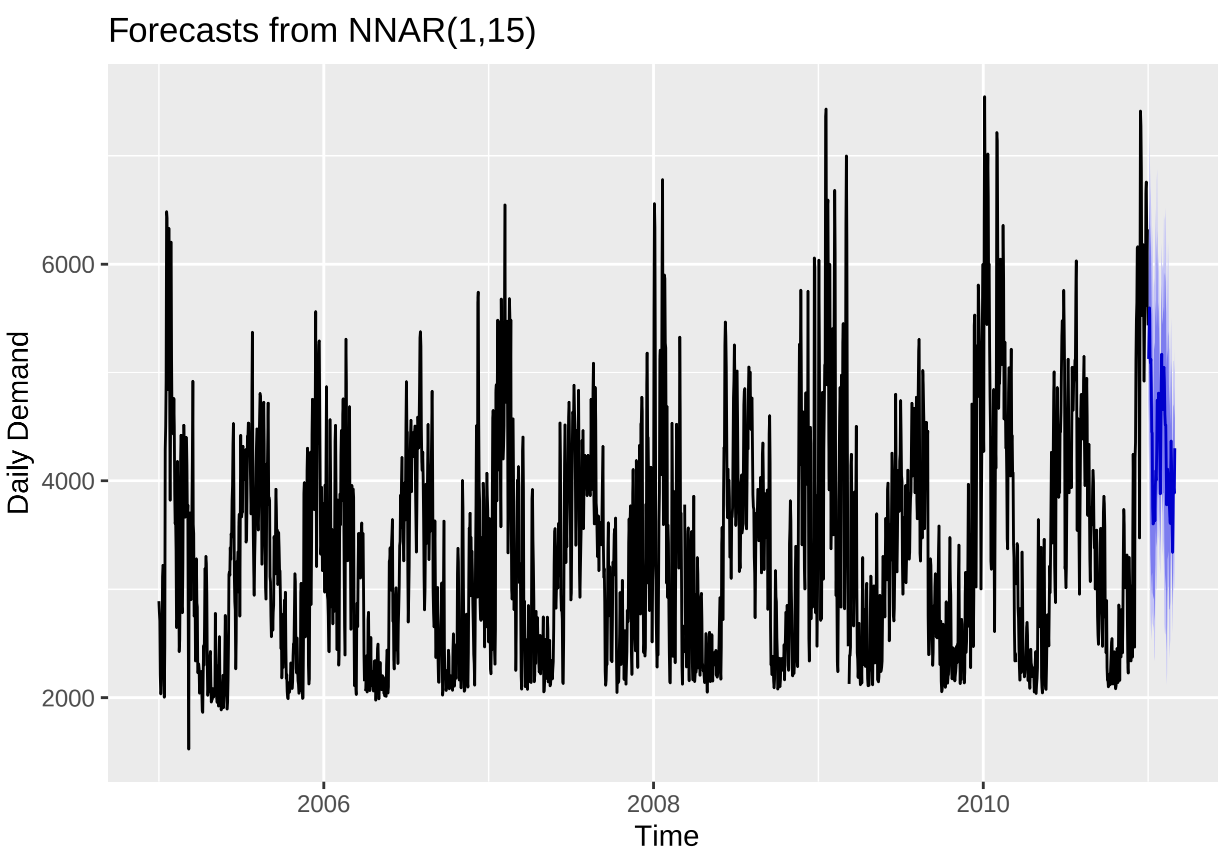
```r
#Checking error variables to decide which model fits the data the best
n_for <- 59
observed <- df_processed2010[1827:1885, "Daily_data"]
NN_scores1 <- accuracy(NN_for$mean,observed)
print(NN_scores1)
```

```
##                ME     RMSE      MAE      MPE     MAPE
## Test set 997.2496 1904.559 1558.008 13.12065 29.98057
```

Model 4 Neural Network Time Series: Forecasts 2011

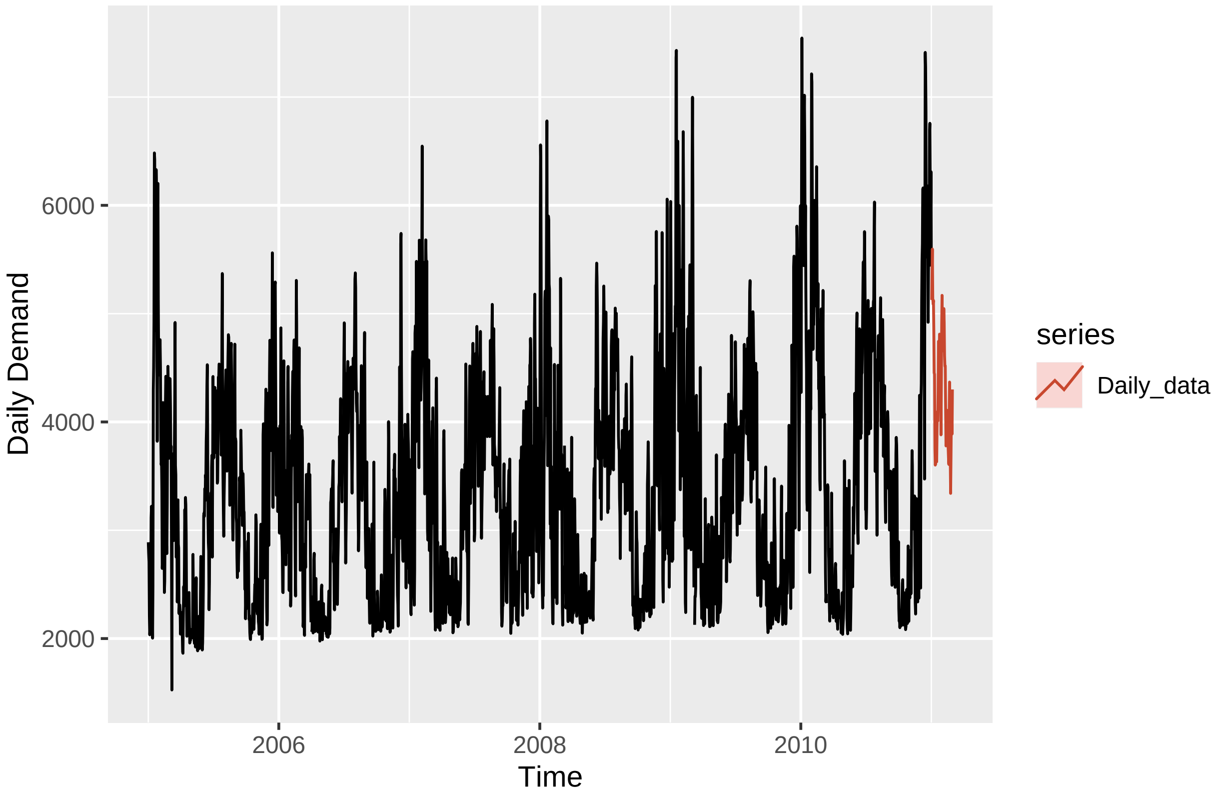
> Note: Based on the error variables calculated with the accuracy() function, the Neural Network model seems to fit the data the best.

```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 -826.2641 1617.556 1436.362 -37.20911 48.58572
```

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

```
## Multi-Seasonal Time Series:
## Start: 2011 2
## Seasonal Periods: 7 365.25
## Data:
##  [1] 5126.954 5373.144 5596.523 5140.610 5087.942 5121.933 4835.299 4453.675
##  [9] 4440.413 3758.617 3601.665 3610.857 3868.796 3802.063 3632.335 3950.547
## [17] 4093.632 4008.631 4285.970 4743.872 4734.358 4647.106 4810.932 4790.960
## [25] 4545.453 4232.877 3880.950 4359.404 4902.073 5169.138 4776.518 4658.043
## [33] 4746.998 4820.472 5047.013 4910.971 4613.401 4515.395 4516.333 4139.741
## [41] 3778.768 3994.564 4108.807 4046.701 4026.268 3902.232 3767.947 3668.704
## [49] 3607.980 4109.539 4368.426 4096.203 3634.835 3340.556 3682.422 4051.544
## [57] 3972.925 3888.103 4300.028
```

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	-103.0303	1547.348	1280.656	-9.57408	28.27524
TBATS	2226.3400	2499.097	2234.019	40.94677	41.24095
ARIMA_FOUR	-284.3608	1536.398	1243.755	-13.07091	28.04510
NEU-NETWORK	997.2496	1904.559	1558.008	13.12065	29.98057

```
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: ARIMA_FOUR
```

```
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"]))
```