---

title: “TSA Forecasting Competition” author: “Yu Hai and Jack Mitchell” date: “2022/3/31” output: pdf_document —

Link to github repository:

https://github.com/yh313/MitchellHai_ENV790_TSA_Competition

```
library(readxl)
library(lubridate)
library(ggplot2)
library(forecast)
library(Kendall)
library(tseries)
library(outliers)
library(tidyverse)
library(smooth)
library(zoo)
library(kableExtra)
#install.packages("writexl")
library(writexl)
#install.packages("smooth")
library(smooth)
```

```
load_data<-read_excel(path="./Data/load.xlsx") #import load data
head(load_data)
```

```
## # A tibble: 6 x 26
##   meter_id date                   h1    h2    h3    h4    h5    h6    h7    h8
##   <chr>    <dttm>              <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 0001     2005-01-01 00:00:00  3304  3178  2981  2944  2934  2999  3104  3296
## 2 0001     2005-01-02 00:00:00  2485  2448  2487  2553  2619  2900  3133  3399
## 3 0001     2005-01-03 00:00:00  2417  2435  2448  2537  2674  2900  3385  3472
## 4 0001     2005-01-04 00:00:00  2060  2018  2010  2094  2115  2327  2714  2758
## 5 0001     2005-01-05 00:00:00  1629  1546  1569  1566  1650  1826  2288  2411
## 6 0001     2005-01-06 00:00:00  1784  1703  1679  1729  1773  2049  2467  2629
## # ... with 16 more variables: h9 <dbl>, h10 <dbl>, h11 <dbl>, h12 <dbl>,
## #   h13 <dbl>, h14 <dbl>, h15 <dbl>, h16 <dbl>, h17 <dbl>, h18 <dbl>,
## #   h19 <dbl>, h20 <dbl>, h21 <dbl>, h22 <dbl>, h23 <dbl>, h24 <dbl>
```

```
humidity_data<-read_excel(path="./Data/relative_humidity.xlsx") #import humidity data
head(humidity_data)
```

```
## # A tibble: 6 x 30
##   date                   hr rh_ws1 rh_ws2 rh_ws3 rh_ws4 rh_ws5 rh_ws6 rh_ws7
##   <dttm>              <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 2005-01-01 00:00:00     1     99     93     93     90     87     93     93
## 2 2005-01-01 00:00:00     2     76     93     97     89     87     97     80
## 3 2005-01-01 00:00:00     3     79     93     93     90     93     97     83
## 4 2005-01-01 00:00:00     4     79     93     93     90     87     89     90
## 5 2005-01-01 00:00:00     5     79     93     96     93     87     90     93
## 6 2005-01-01 00:00:00     6     82     93     97     97     87     93     97
## # ... with 21 more variables: rh_ws8 <dbl>, rh_ws9 <dbl>, rh_ws10 <dbl>,
## #   rh_ws11 <dbl>, rh_ws12 <dbl>, rh_ws13 <dbl>, rh_ws14 <dbl>, rh_ws15 <dbl>,
## #   rh_ws16 <dbl>, rh_ws17 <dbl>, rh_ws18 <dbl>, rh_ws19 <dbl>, rh_ws20 <dbl>,
## #   rh_ws21 <dbl>, rh_ws22 <dbl>, rh_ws23 <dbl>, rh_ws24 <dbl>, rh_ws25 <dbl>,
## #   rh_ws26 <dbl>, rh_ws27 <dbl>, rh_ws28 <dbl>
```

```
temperature_data<-read_excel(path="./Data/temperature.xlsx") #import temp data
```

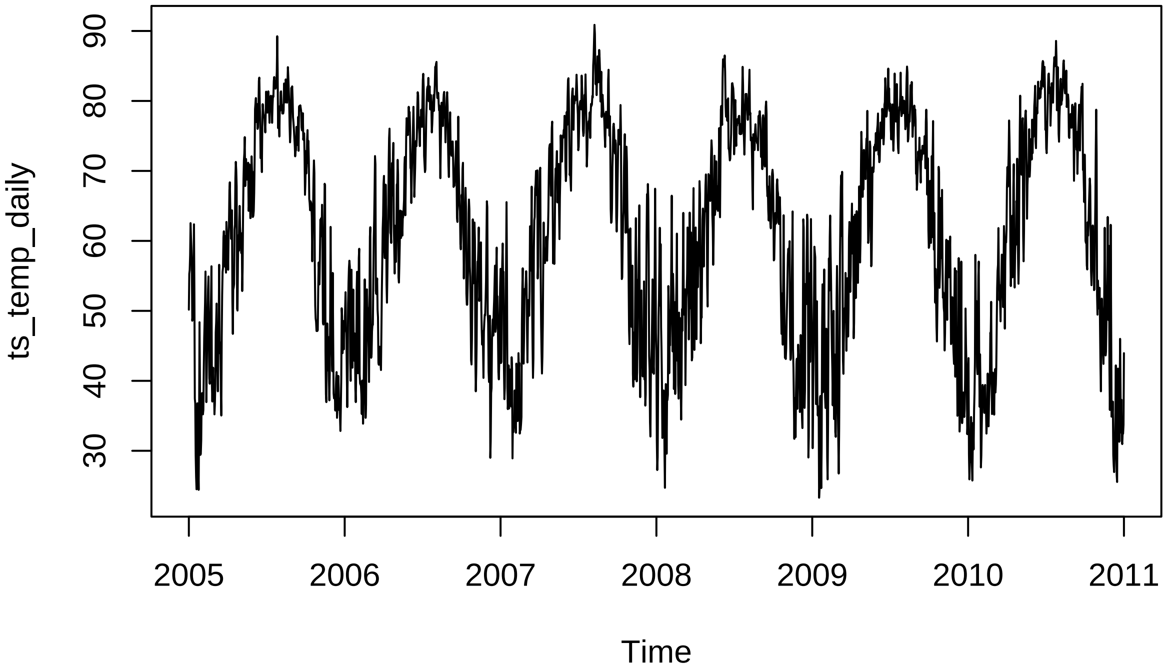
```
load_data$load_daily_avg = rowMeans(load_data[,c(3:26)]) #calculate avg daily load
load_data_rm_na <- lapply(load_data$load_daily_avg,na.aggregate)
ts_load_daily_avg<-msts(load_data$load_daily_avg,seasonal.periods =c(7,365.25),start=c(2005,01,01)) #co
ts_load_daily_avg_rm_na3<-msts(load_data_rm_na,seasonal.periods =c(7,365.25),start=c(2005,01,01)) #conv

humidity_data_daily <- humidity_data %>%  #avg daily humidity data
  mutate( Year = year(date),
          Month = month(date),
          Day = day(date)) %>%
  select( date, Year, Month, Day, hr, rh_ws1) %>%
  group_by(date,Year,Month,Day) %>%
  summarise(daily_mean_humidity = mean(rh_ws1))
```

```
## `summarise()` has grouped output by 'date', 'Year', 'Month'. You can override using the `.groups` ar
```

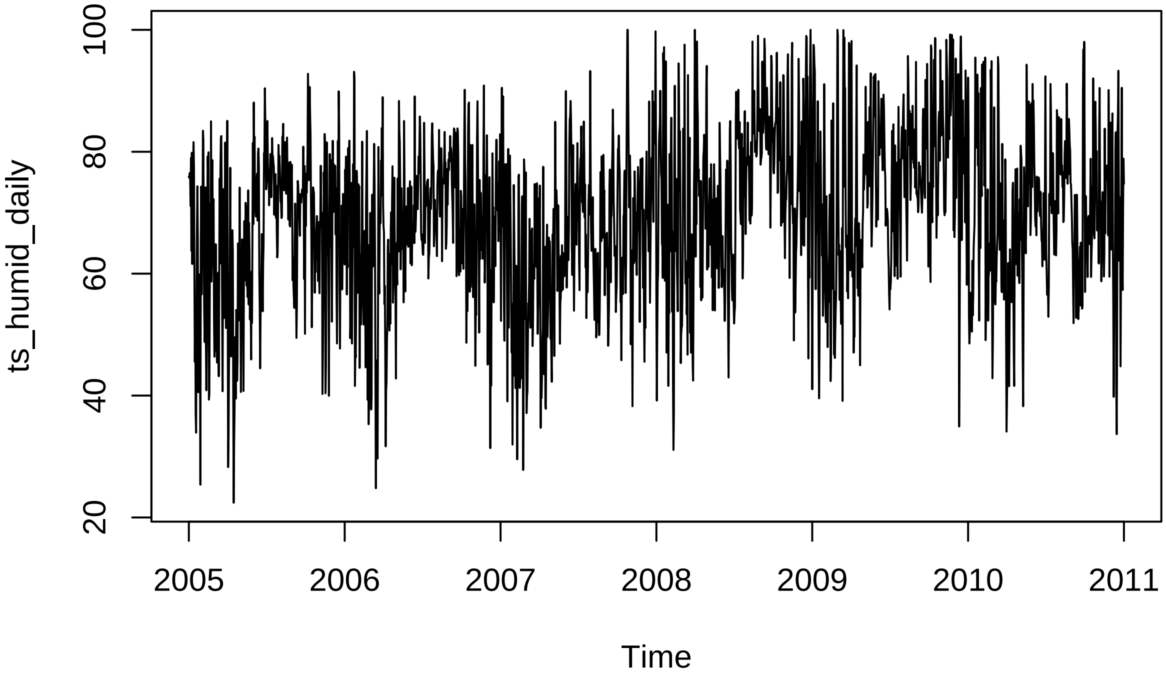
```
temperature_data_daily <- temperature_data %>% #avg daily temperature data
  mutate( Year = year(date),
          Month = month(date),
          Day = day(date)) %>%
  select( date, Year, Month, Day, hr, t_ws1) %>%
  group_by(date,Year,Month,Day) %>%
  summarise(daily_mean_temp = mean(t_ws1))
```

```
## `summarise()` has grouped output by 'date', 'Year', 'Month'. You can override using the `.groups` ar
```

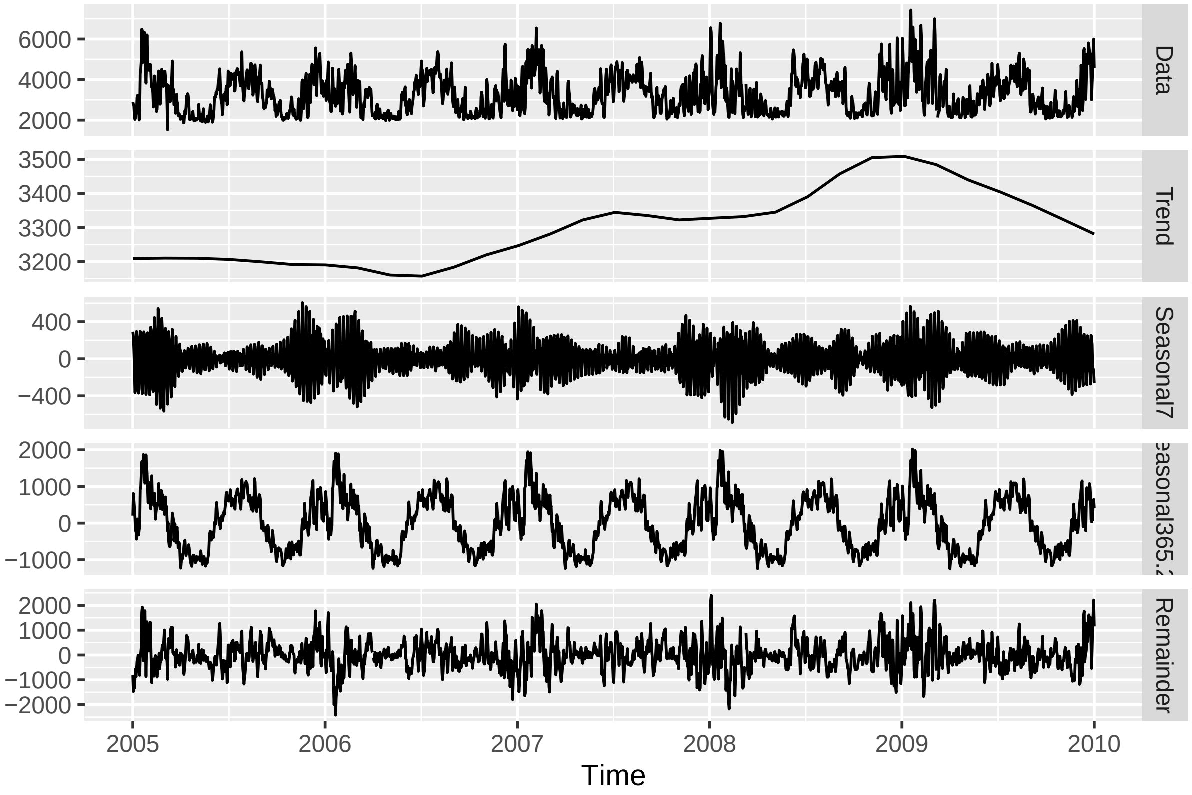
```
#convert other variables to time series
ts_temp_daily <- ts(temperature_data_daily$daily_mean_temp,frequency=365,start=c(2005,01))
ts_humid_daily <- ts(humidity_data_daily$daily_mean_humidity,frequency=365,start=c(2005,01))
plot(ts_temp_daily)
```



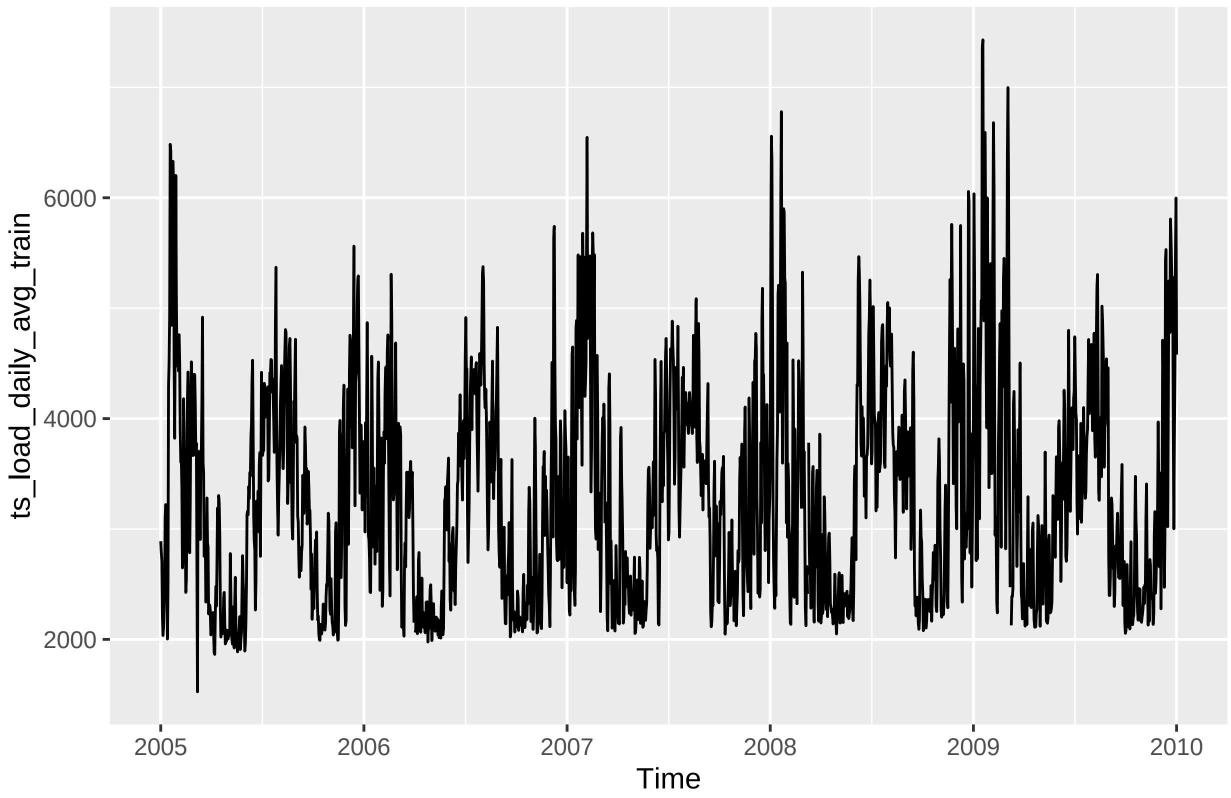
```
plot(ts_humid_daily)
```



```
n_for=365 # days in a year
ts_load_daily_avg_train<-subset(ts_load_daily_avg,end=length(ts_load_daily_avg)-n_for) #create training
ts_load_daily_avg_test<-subset(ts_load_daily_avg,start=length(ts_load_daily_avg)-n_for) #create testing
ts_load_daily_avg_train %>% mstl() %>%
autoplot() #decompose training subset
```

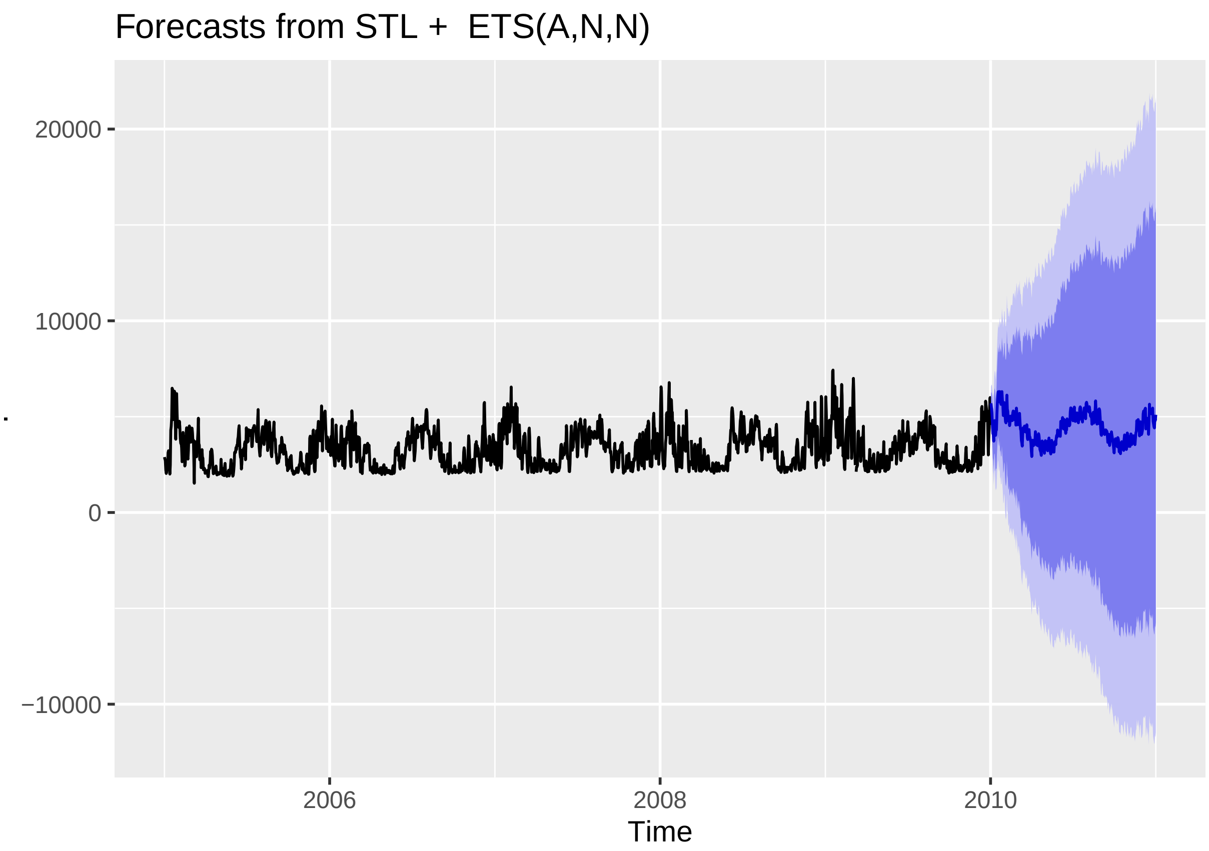


```
autoplot(ts_load_daily_avg_train) #time series plot of training subset
```

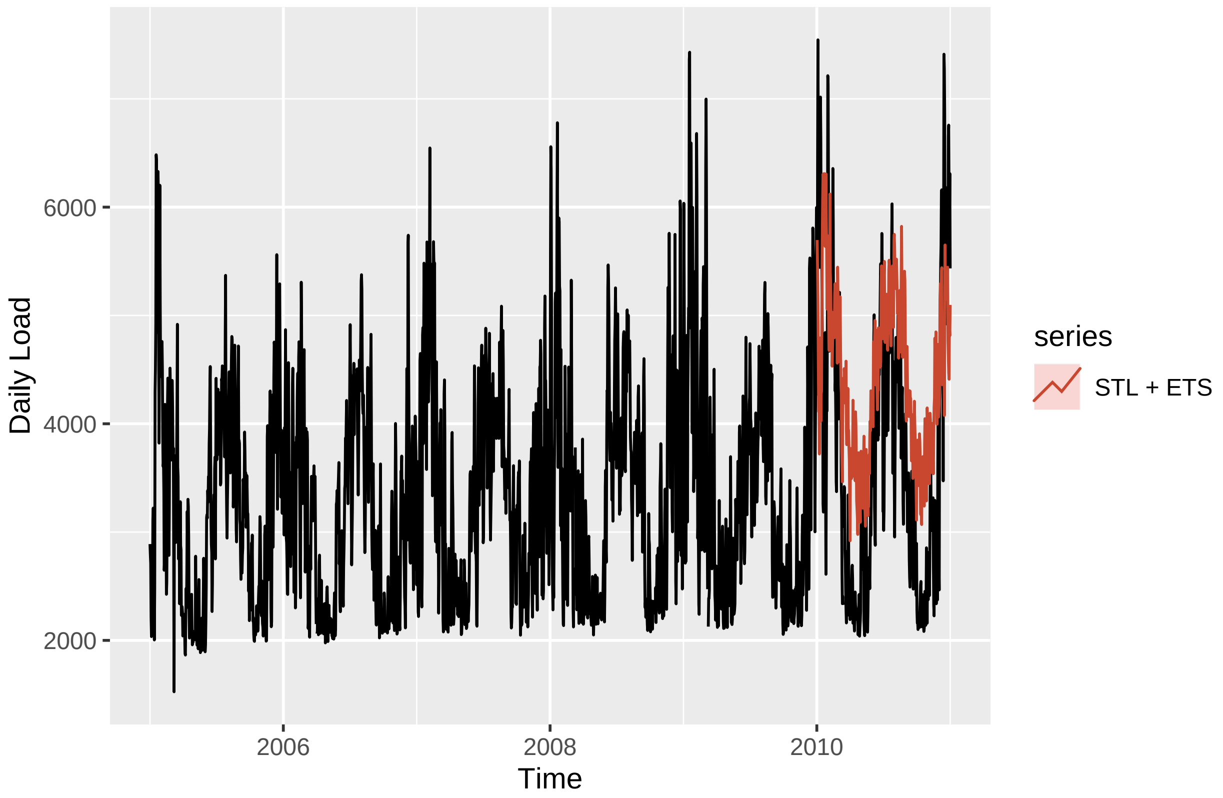


```
#splitting temperature data into train and test sets
ts_temp_daily_train<-subset(ts_temp_daily,end=length(ts_load_daily_avg)-n_for)
ts_temp_daily_test<-subset(ts_temp_daily,start=length(ts_load_daily_avg)-n_for)

#splitting humidity data into train and test sets
ts_humid_daily_train<-subset(ts_humid_daily,end=length(ts_load_daily_avg)-n_for)
ts_humid_daily_test<-subset(ts_humid_daily,start=length(ts_load_daily_avg)-n_for)
```

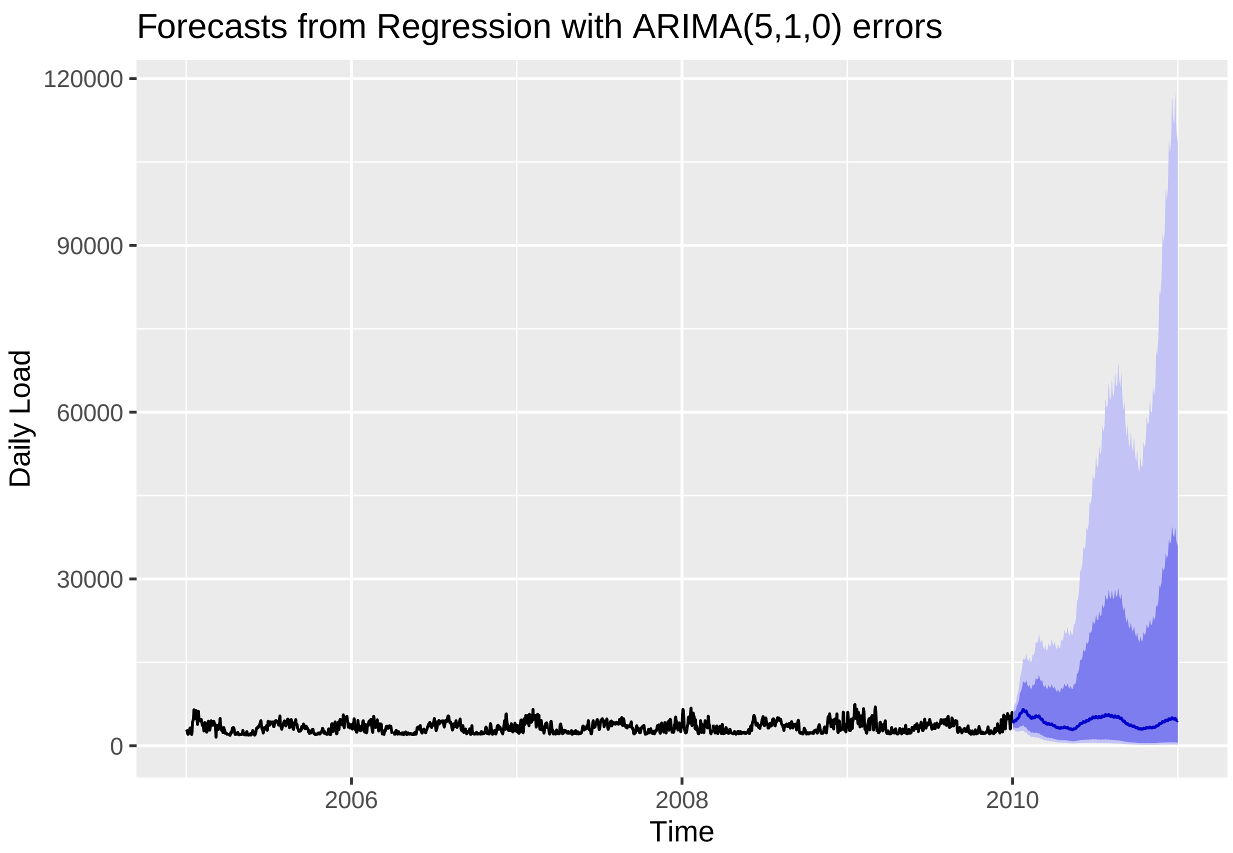
```
#STL+ETS
ts_load_daily_avg_train %>% stlf(h=365) %>% autoplot()
```



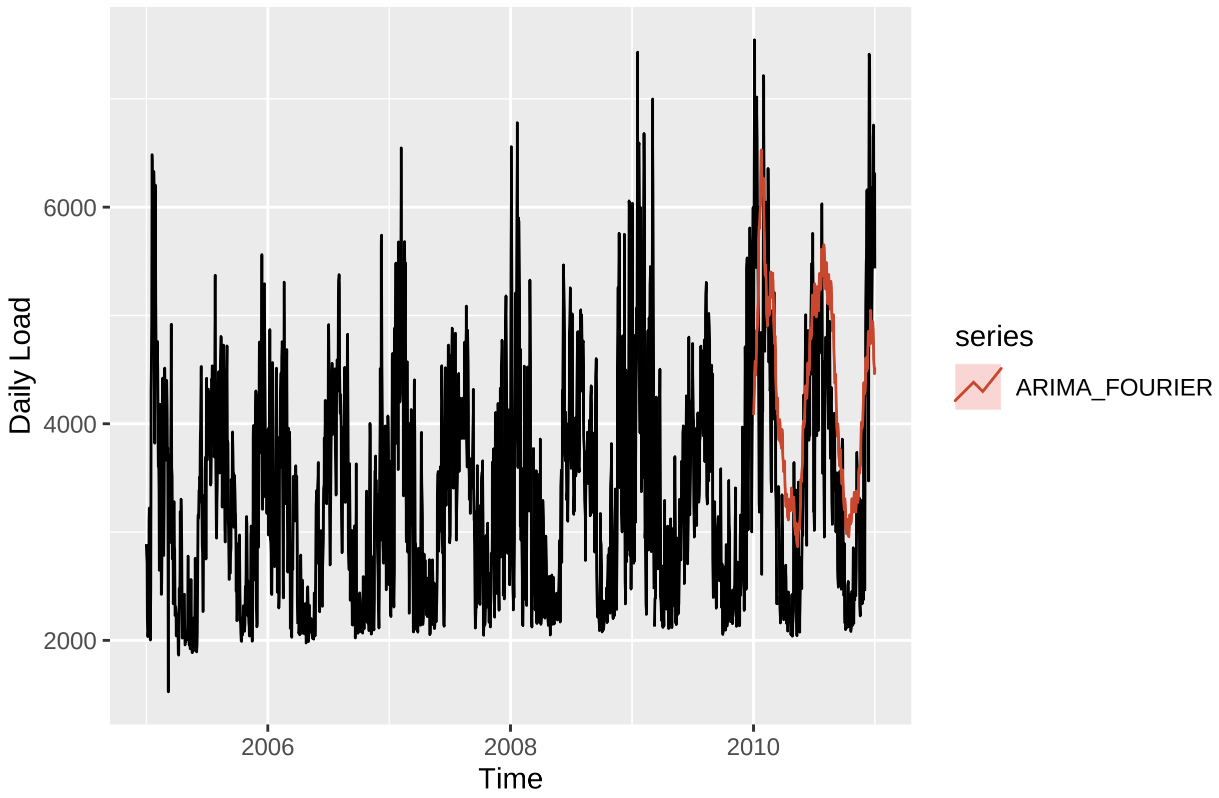
```
ETS_fit <-  stlf(ts_load_daily_avg_train,h=365)
autoplot(ts_load_daily_avg) +
  autolayer(ETS_fit, series="STL + ETS",PI=FALSE) +
  ylab("Daily Load")
```



```
#Arima model with fourier terms
ARIMA_Four_fit <- auto.arima(ts_load_daily_avg_train,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_load_daily_avg_train,
                                          K=c(2,12))
                             )
ARIMA_Four_for <- forecast::forecast(ARIMA_Four_fit,
                           xreg=fourier(ts_load_daily_avg_train,
                                        K=c(2,12),
                                        h=365),
                           h=365
                           )

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



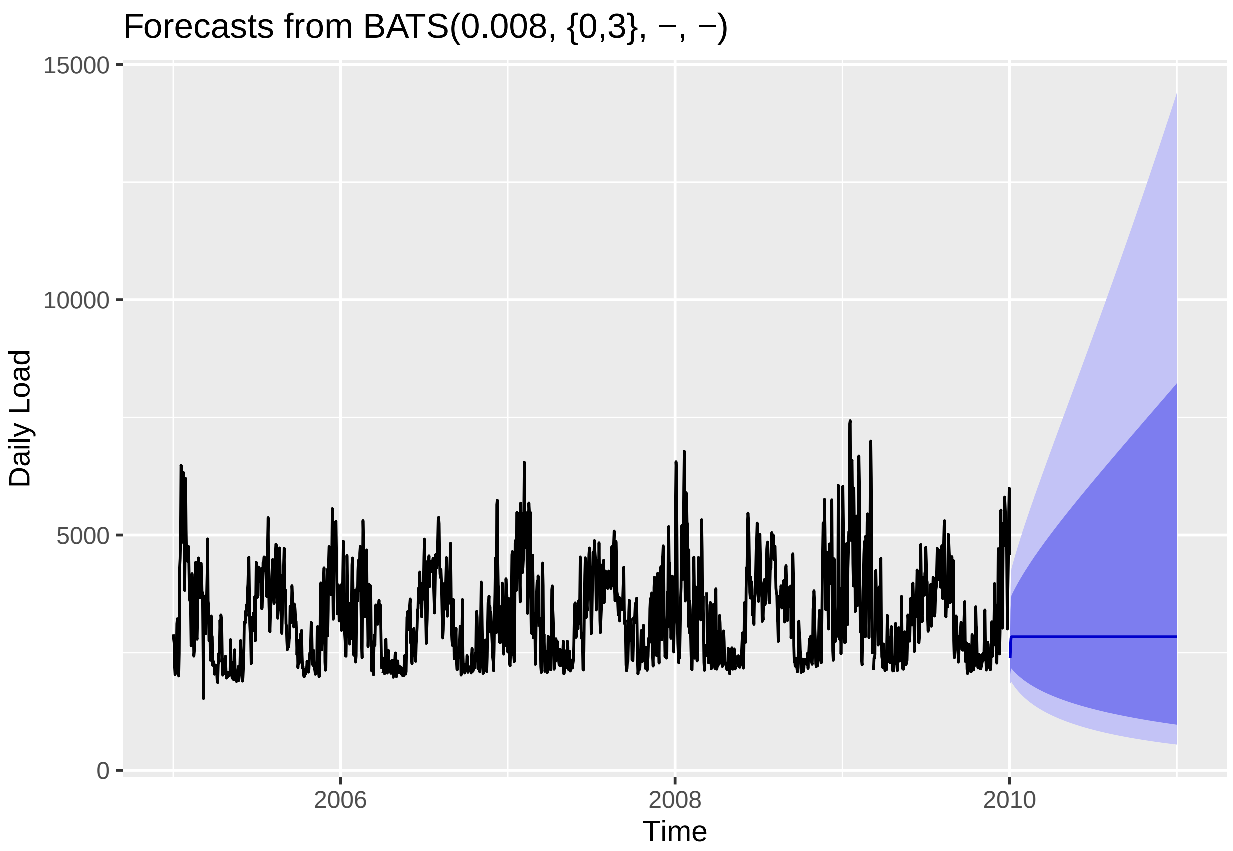
```
#Plot model + observed data
autoplot(ts_load_daily_avg) +
  autolayer(ARIMA_Four_for, series="ARIMA_FOURIER",PI=FALSE) +
  ylab("Daily Load")
```



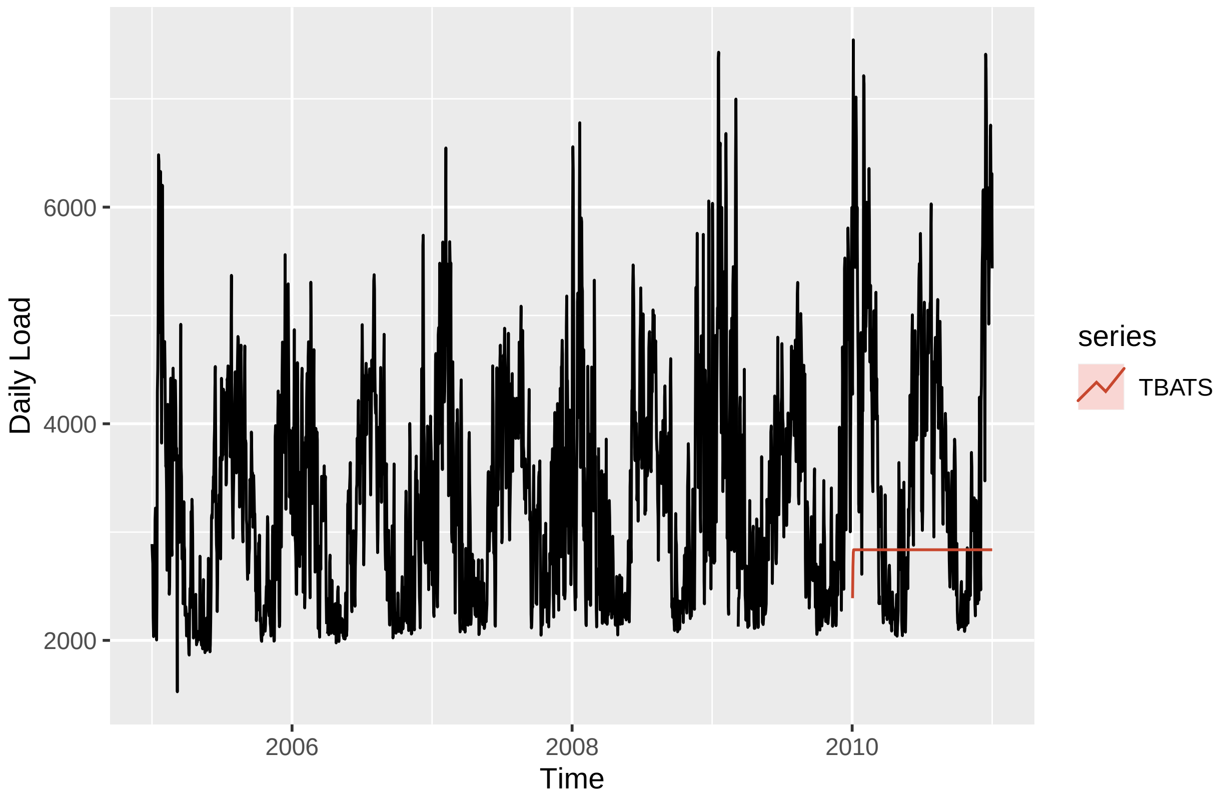
```
# TBATS can take time to fit
TBATS_fit <- tbats(ts_load_daily_avg_train)
```

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

```
TBATS_for <- forecast::forecast(TBATS_fit, h=365)

#Plot foresting results
autoplot(TBATS_for) +
  ylab("Daily Load")
```



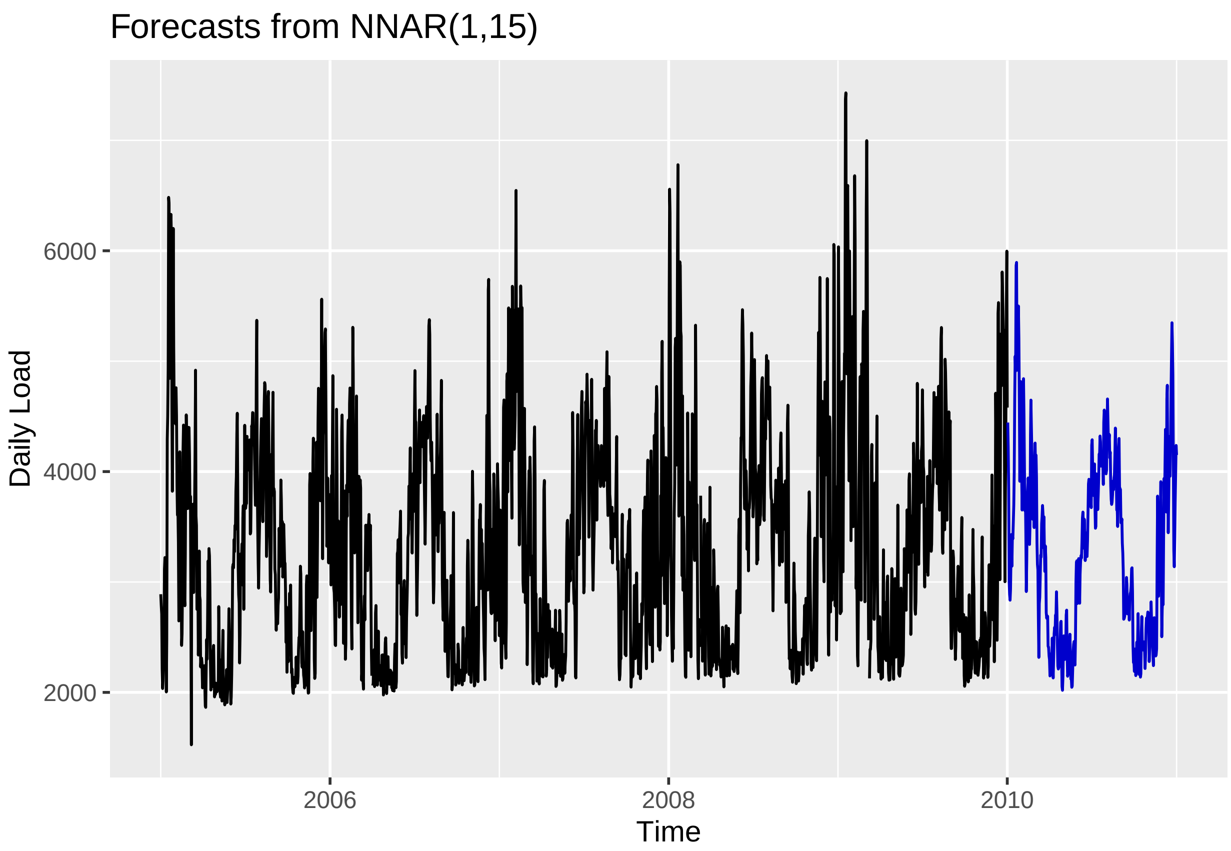
```
#Plot model + observed data
autoplot(ts_load_daily_avg) +
  autolayer(TBATS_for, series="TBATS",PI=FALSE)+
  ylab("Daily Load")
```



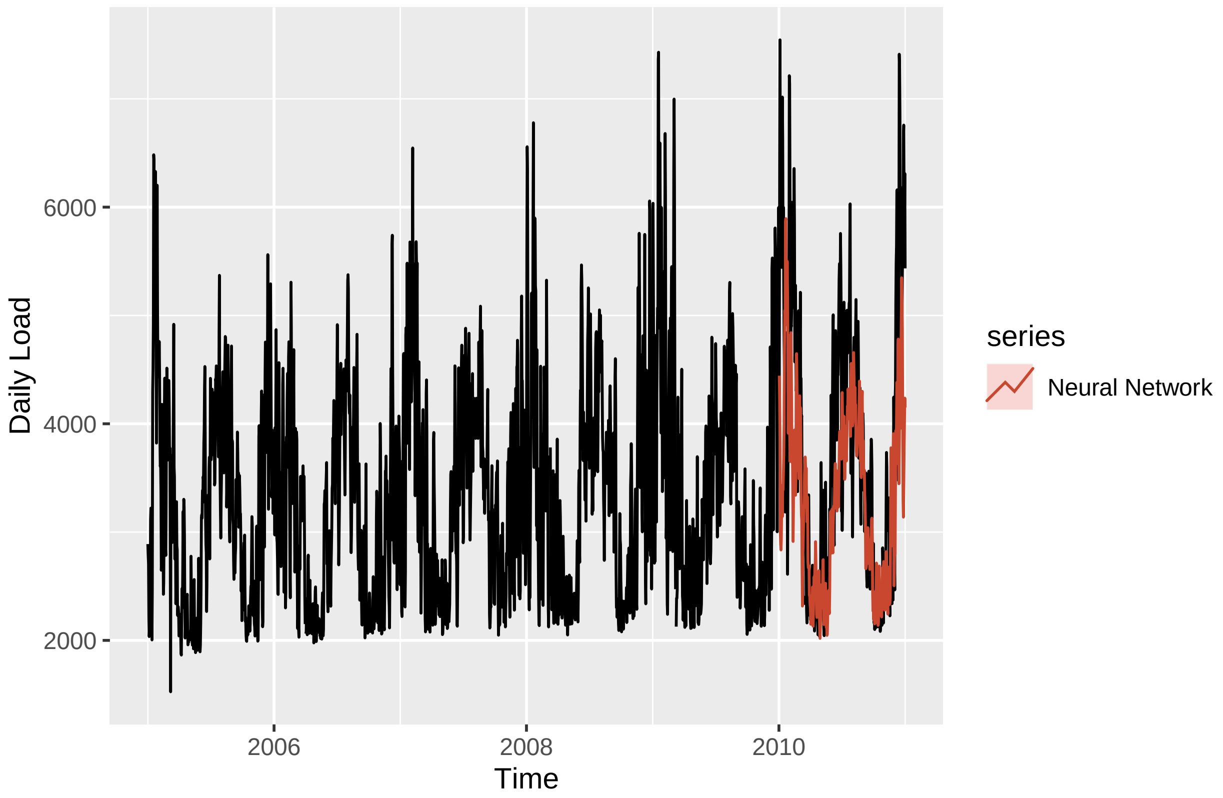
```
#NN+fourier model forcing p=1,P=0

NN_fit <- nnetar(ts_load_daily_avg_train,p=1,P=0,xreg=fourier(ts_load_daily_avg_train, K=c(2,12)))
```

```
## Warning in nnetar(ts_load_daily_avg_train, p = 1, P = 0, xreg =
## fourier(ts_load_daily_avg_train, : Missing values in x, omitting rows
```

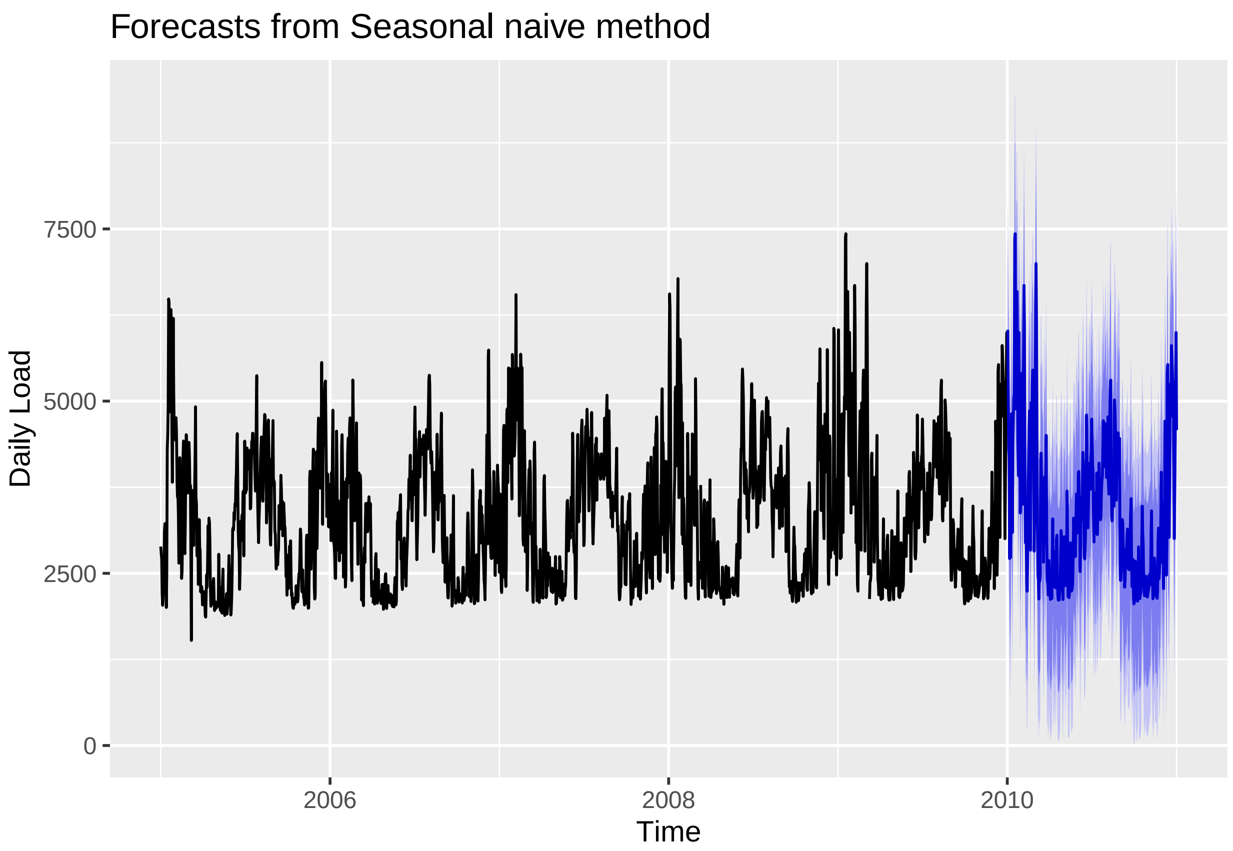
```
NN_for <- forecast::forecast(NN_fit, h=365,xreg=fourier(ts_load_daily_avg_train,
                                          K=c(2,12),h=365))

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



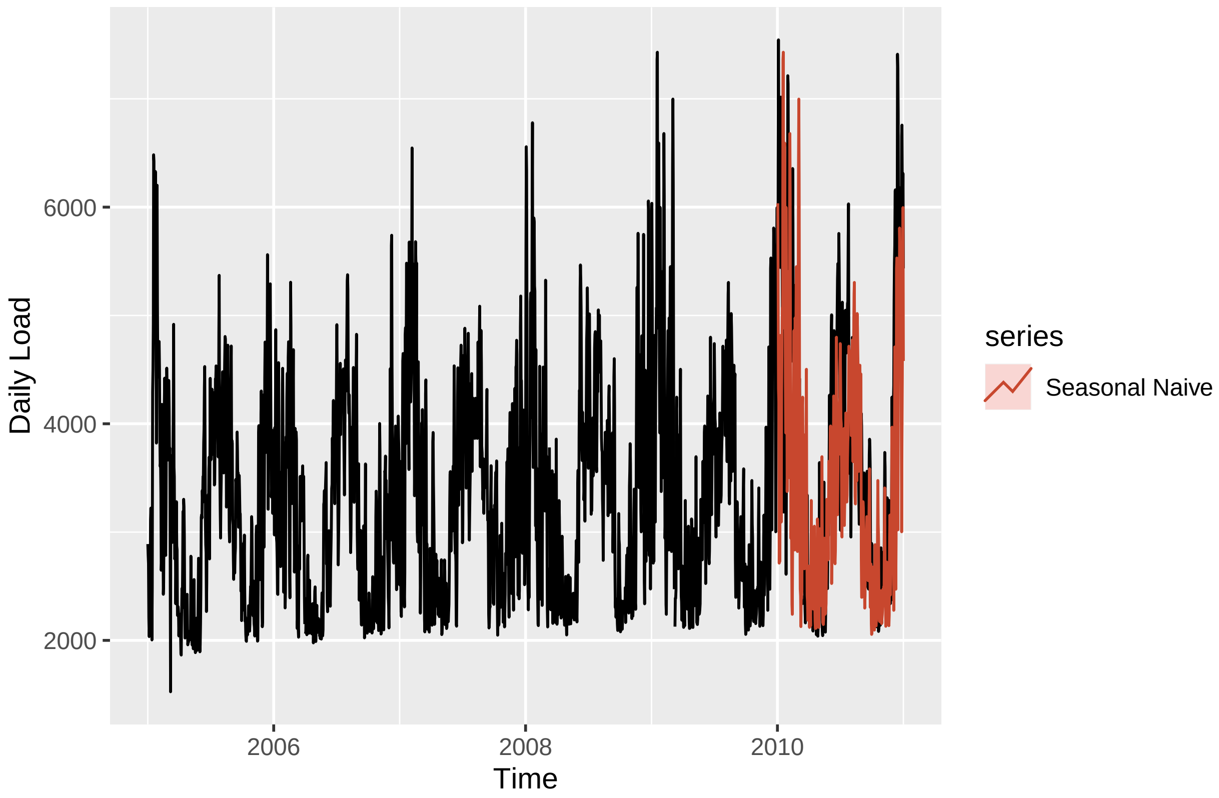
```
#Plot model + observed data
autoplot(ts_load_daily_avg) +
  autolayer(NN_for, series="Neural Network",PI=FALSE)+
  ylab("Daily Load")
```



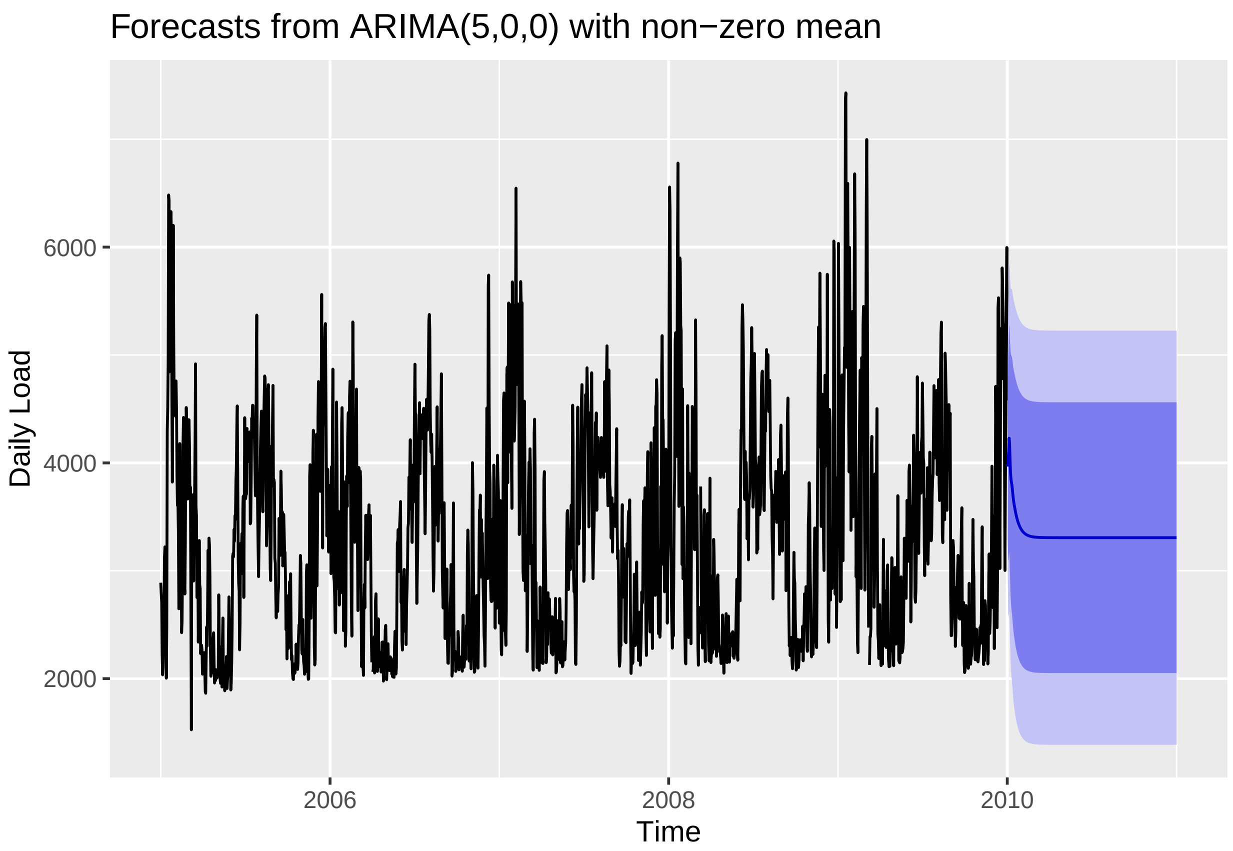
```r
#Seasonal naive model
SNAIVE <- snaive(ts_load_daily_avg_train, h=365)
autoplot(SNAIVE) +
  ylab("Daily Load")
```



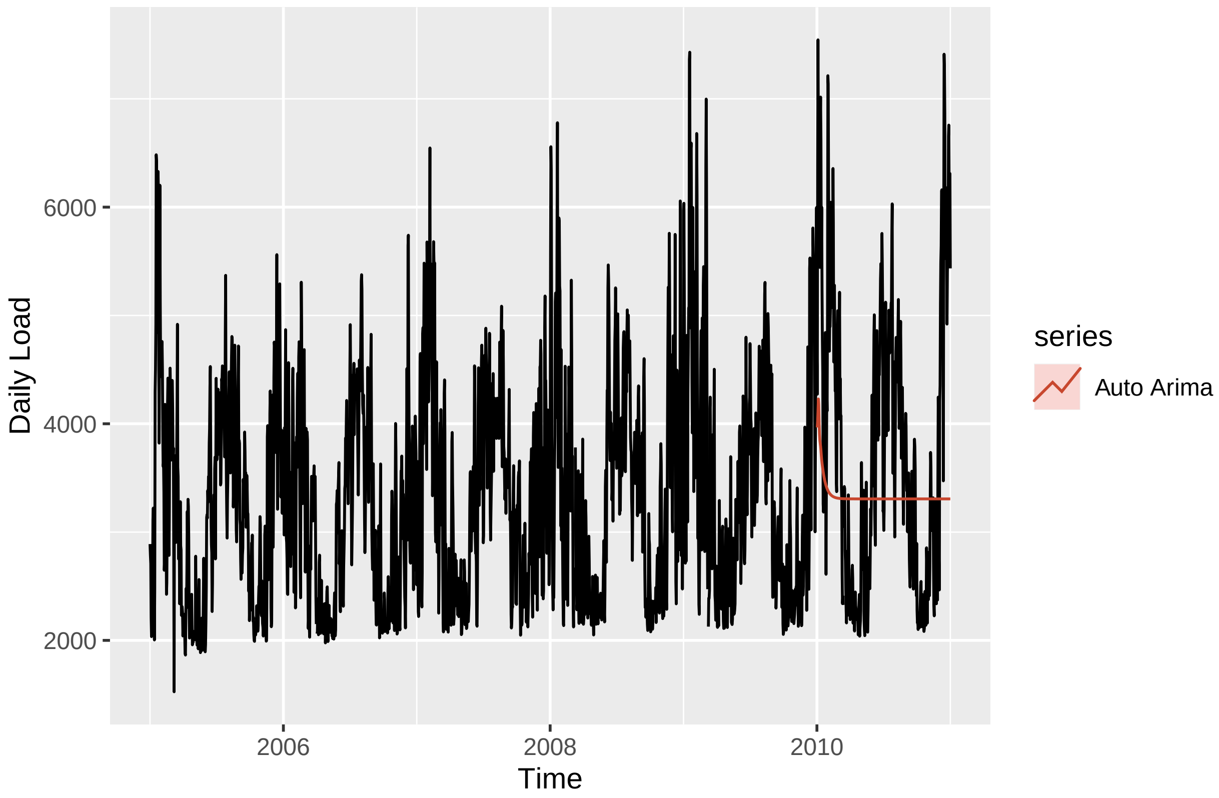
```
#Plot model + observed data
autoplot(ts_load_daily_avg) +
  autolayer(SNAIVE, series="Seasonal Naive",PI=FALSE)+
  ylab("Daily Load")
```



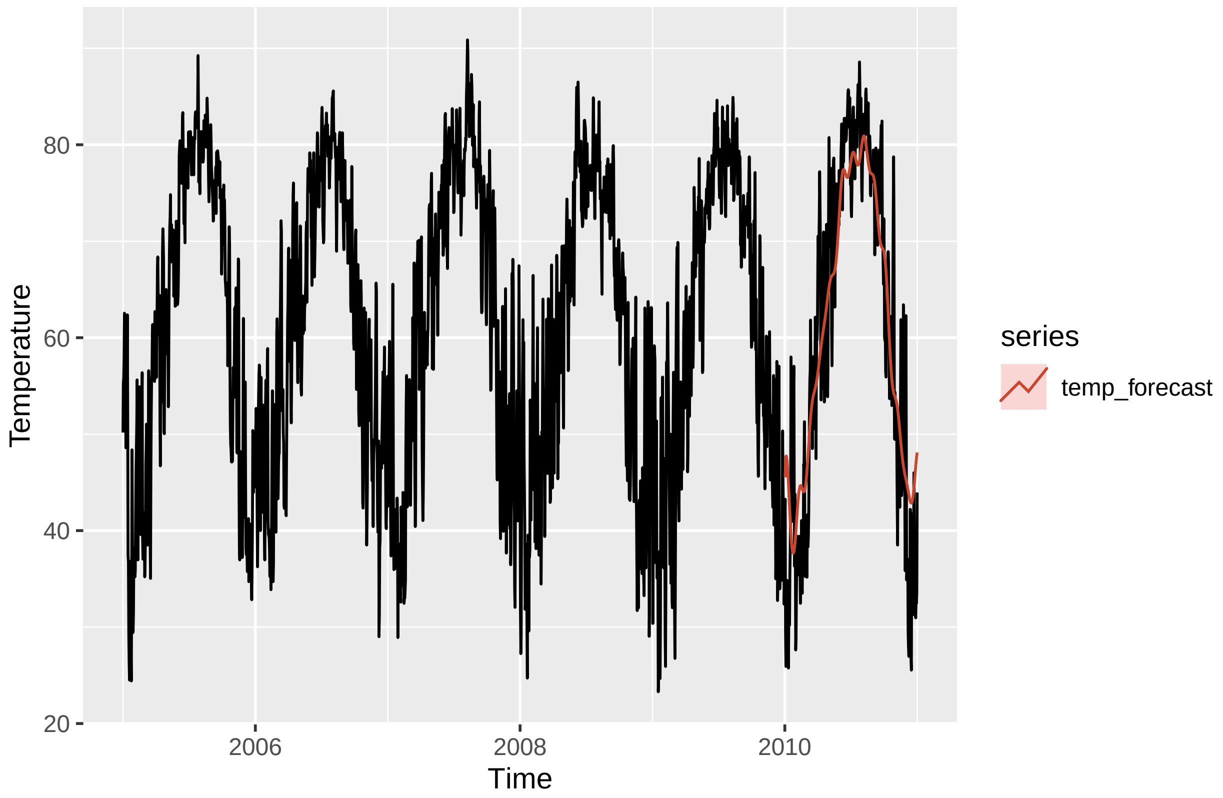
```
#Auto arima model
ARIMA_autofit <- auto.arima(ts_load_daily_avg_train, max.D = 0, max.P = 0, max.Q = 0)
ARIMA_forecast <- forecast::forecast(object = ARIMA_autofit, h = 365)
autoplot(ARIMA_forecast) +
  ylab("Daily Load")
```



```r
#Plot model + observed data
autoplot(ts_load_daily_avg) +
  autolayer(ARIMA_forecast, series="Auto Arima",PI=FALSE)+
  ylab("Daily Load")
```

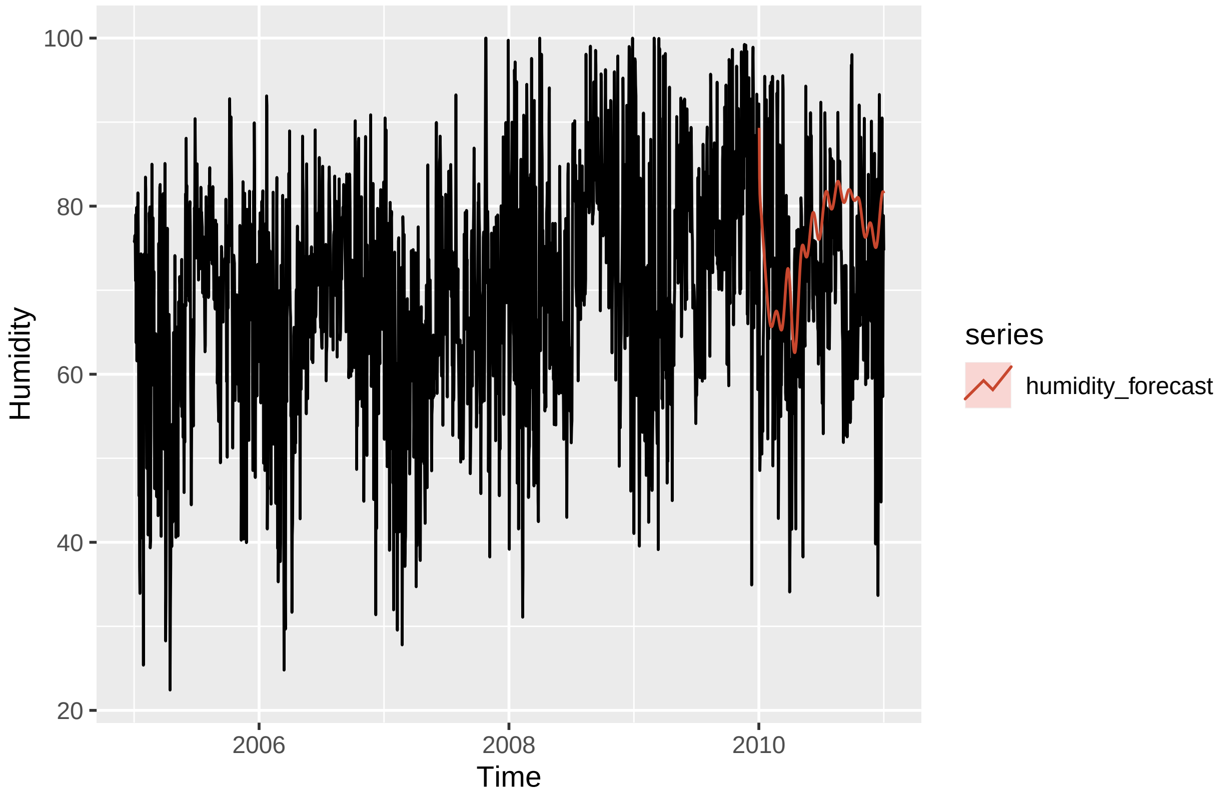


```
#create temperature forecasts
temp_fit <- auto.arima(ts_temp_daily_train,
                       seasonal=FALSE,
                       lambda=0,
                       xreg=fourier(ts_temp_daily_train,
                                    K=12)
                       )
temp_for <- forecast::forecast(temp_fit,
                     xreg=fourier(ts_temp_daily_train,
                                  K=12,
                                  h=365),
                     h=365
                     )
autoplot(ts_temp_daily) +
  autolayer(temp_for, series="temp_forecast",PI=FALSE) +
  ylab("Temperature")
```



```
temp_fit2 <- auto.arima(ts_temp_daily,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_temp_daily,
                                          K=12)
                             )
temp_for2 <- forecast::forecast(temp_fit2,
                           xreg=fourier(ts_temp_daily,
                                        K=12,
                                        h=31),
                           h=31
                           )
temp_for2_df<-data.frame(new_temp=temp_for2)
ts_temp_for2<-ts(temp_for2_df[,1])
```

```
#create humidity forecasts
humid_fit <- auto.arima(ts_humid_daily_train,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_humid_daily_train,
                                          K=12)
                             )
humid_for <- forecast::forecast(humid_fit,
                           xreg=fourier(ts_humid_daily_train,
                                        K=12,
                                        h=365),
                             h=365
                           )
autoplot(ts_humid_daily) +
  autolayer(humid_for, series="humidity_forecast",PI=FALSE) +
  ylab("Humidity")
```

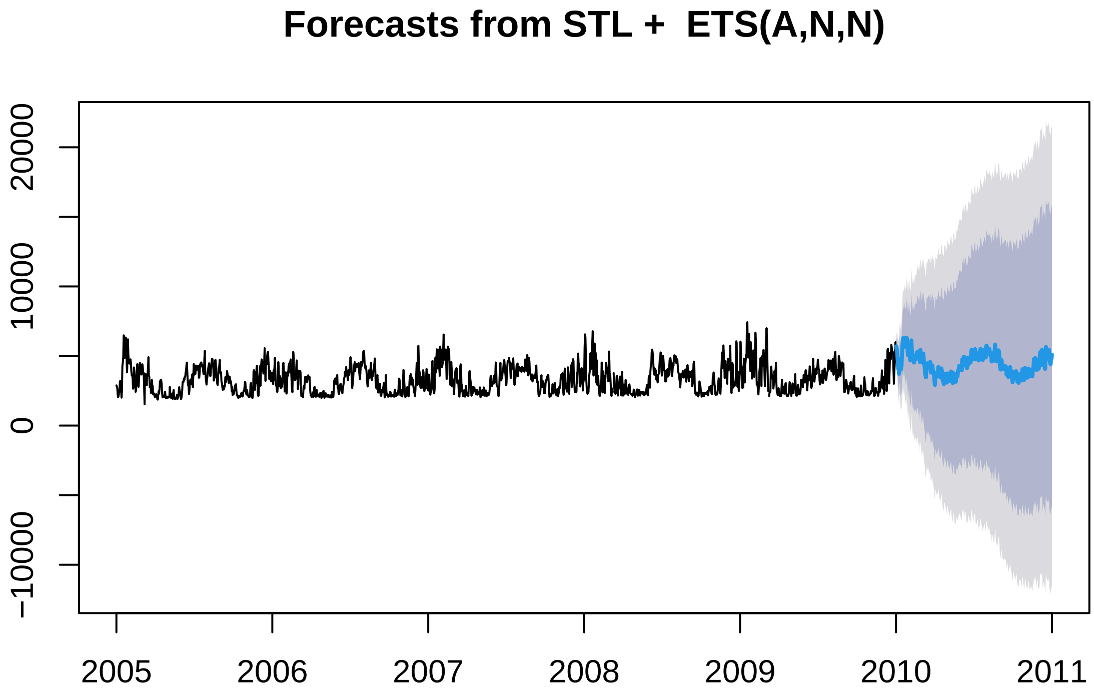


```
humid_fit2 <- auto.arima(ts_humid_daily,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_humid_daily,
                                          K=12)
                             )
humid_for2 <- forecast::forecast(humid_fit2,
                           xreg=fourier(ts_humid_daily,
                                        K=12,
                                        h=31),
                           h=31
                           )
humid_for2_df<-data.frame(new_humid=humid_for2)
ts_humid_for2<-ts(humid_for2_df[,1])
```

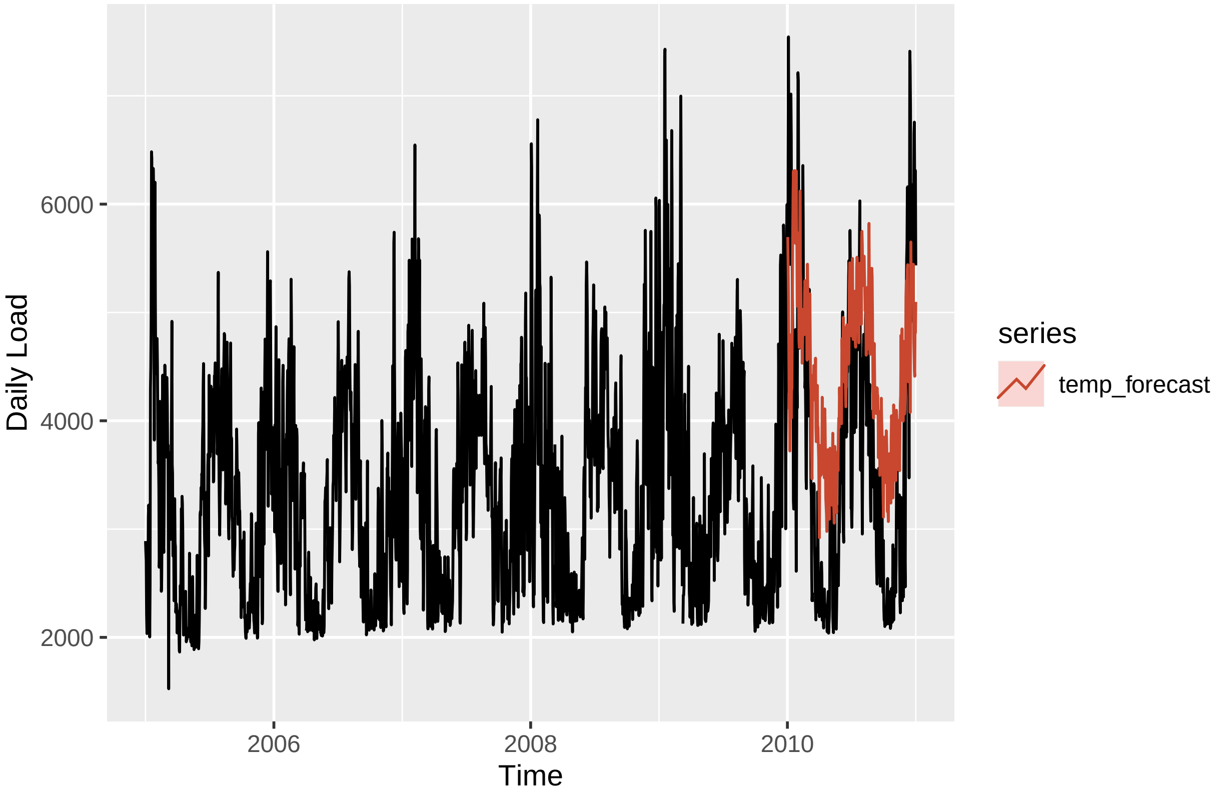
```
#SARIMA
SARIMA_autofit <- auto.arima(ts_load_daily_avg_train)
print(SARIMA_autofit)
```

```
## Series: ts_load_daily_avg_train
## ARIMA(5,0,0) with non-zero mean
##
## Coefficients:
##          ar1      ar2     ar3      ar4     ar5      mean
##       1.0045  -0.3993  0.2156  -0.0727  0.1272  3306.509
## s.e.  0.0233   0.0331  0.0340   0.0331  0.0233    95.523
##
## sigma^2 = 262752:  log likelihood = -13945.75
## AIC=27905.49   AICc=27905.55   BIC=27944.06
```

```
SARIMA_forecast <- forecast::forecast(object = ts_load_daily_avg_train, h = 365)
plot(SARIMA_forecast)
```



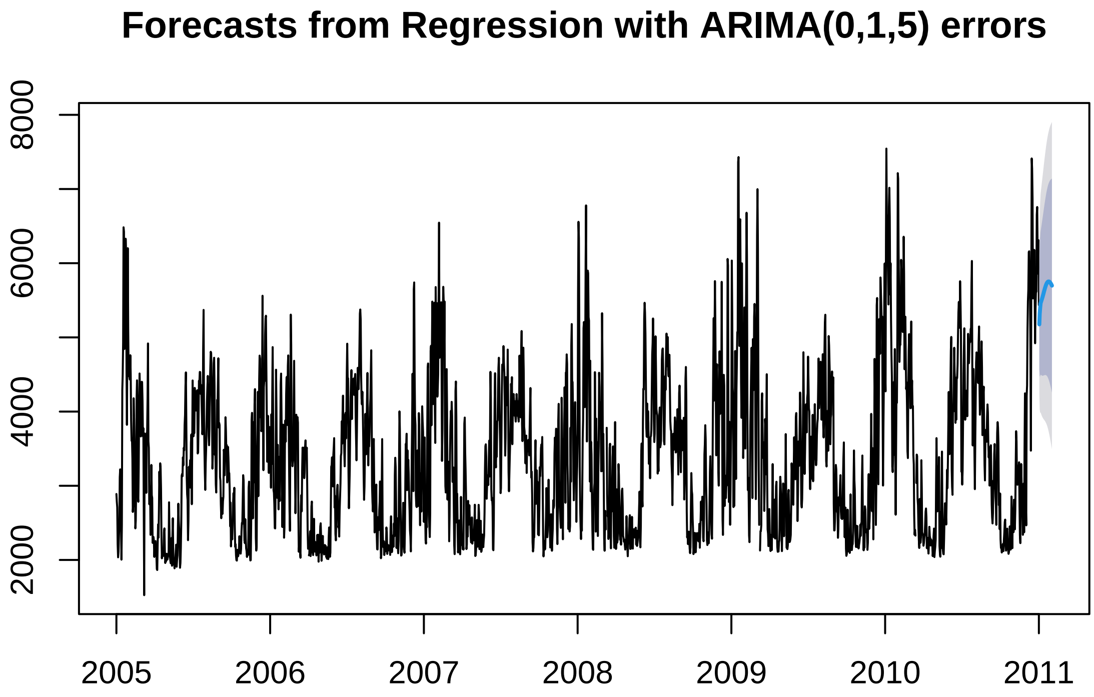
```
autoplot(ts_load_daily_avg) +
  autolayer(SARIMA_forecast, series="temp_forecast",PI=FALSE) +
  ylab("Daily Load")
```



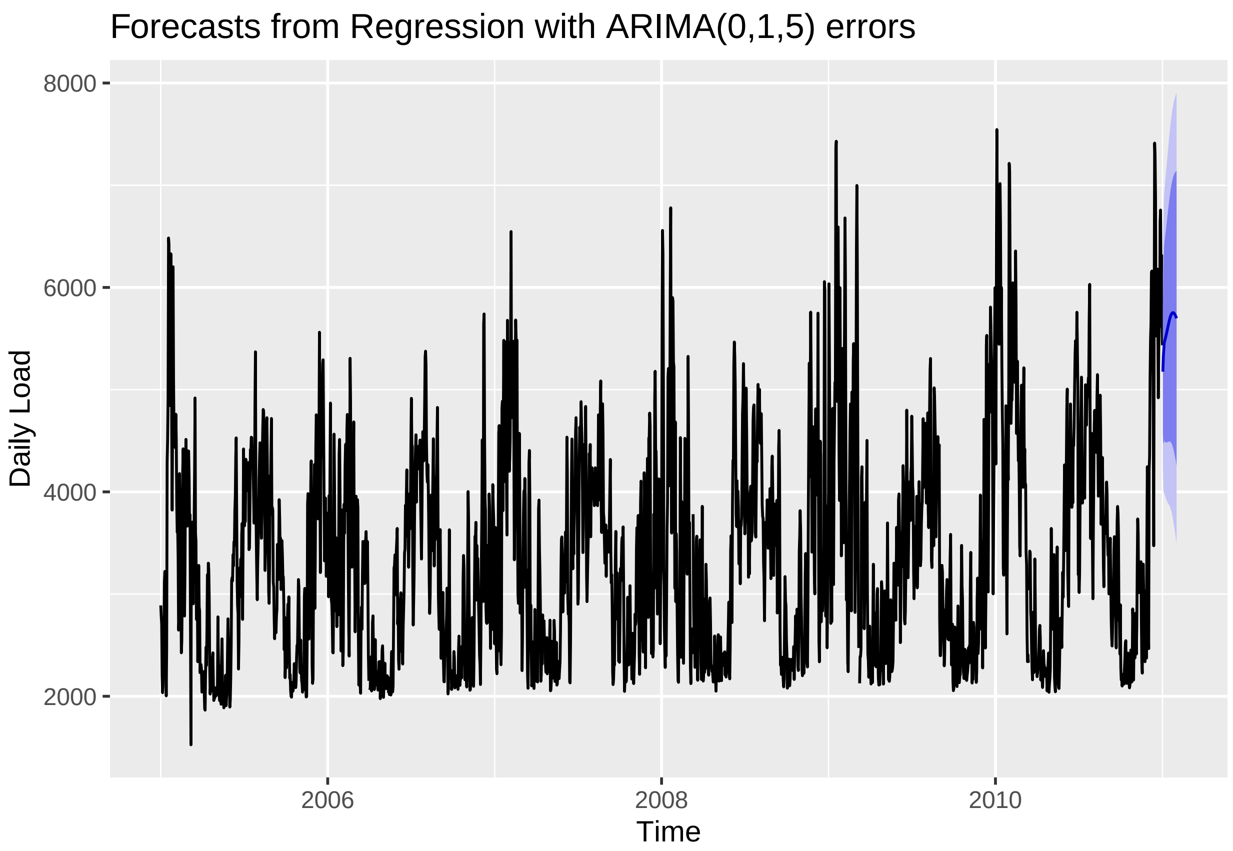
```
#SARIMA forecast with temperature for Jan 2022
SARIMA_autofit2 <- auto.arima(ts_load_daily_avg,xreg=ts_temp_daily)
print(SARIMA_autofit2)
```

```
## Series: ts_load_daily_avg
## Regression with ARIMA(0,1,5) errors
##
## Coefficients:
##           ma1      ma2      ma3      ma4      ma5   drift       xreg
##       -0.0434  -0.3345  -0.1799  -0.0940  -0.0075  1.0690   -40.3669
## s.e.   0.0216   0.0216   0.0218   0.0212   0.0214  3.4539     2.0949
##
## sigma^2 = 224819:  log likelihood = -16554.61
## AIC=33125.23   AICc=33125.29   BIC=33170.76
```

```
SARIMA_forecast2 <- forecast::forecast(object = SARIMA_autofit2, xreg=ts_temp_for2,h = 31)
plot(SARIMA_forecast2)
```



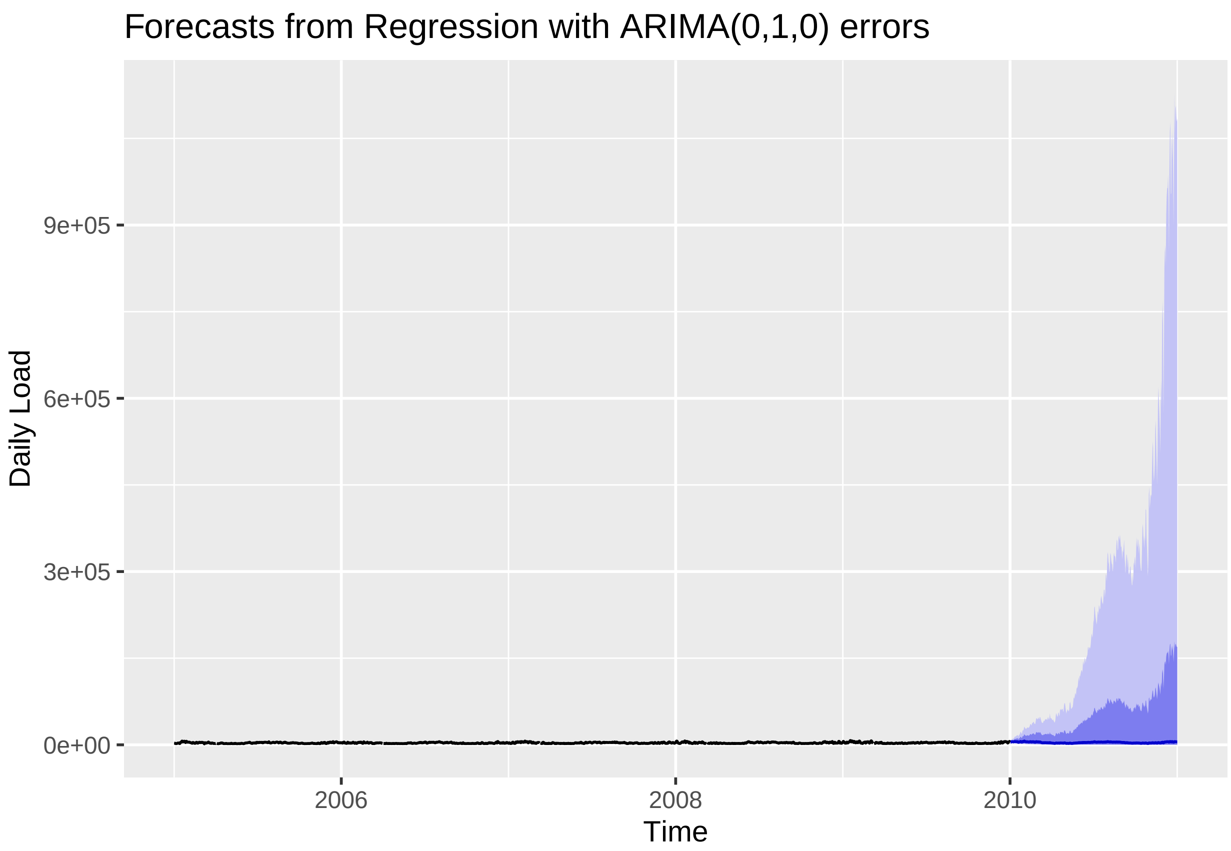
```
autoplot(SARIMA_forecast2) + ylab("Daily Load")
```



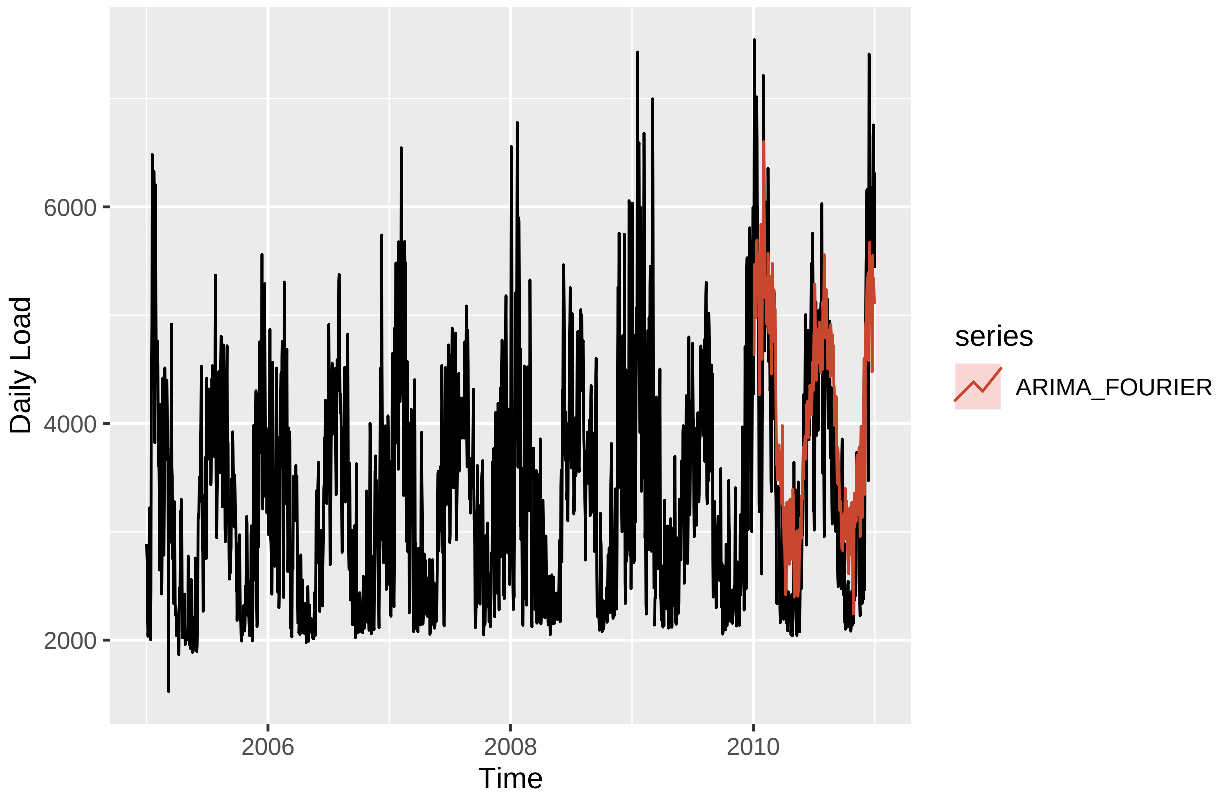
```
#Arima model with fourier terms + temperature
ARIMA_Four_temp_fit <- auto.arima(ts_load_daily_avg_train,
                              seasonal=FALSE,
                              lambda=0,
                              xreg=cbind(ts_temp_daily_train,fourier(ts_load_daily_avg_train,
                                           K=c(2,12))
                              ))


ARIMA_Four_temp_for <- forecast::forecast(ARIMA_Four_temp_fit,
                            xreg=cbind(ts_temp_daily_test[1:365],fourier(ts_load_daily_avg_train,
                                         K=c(2,12),
                                         h=365)),
                            h=365
                            )
```

```
## Warning in forecast.forecast_ARIMA(ARIMA_Four_temp_fit, xreg =
## cbind(ts_temp_daily_test[1:365], : xreg contains different column names from the
## xreg used in training. Please check that the regressors are in the same order.
```

```
#Plot foresting results
autoplot(ARIMA_Four_temp_for) + ylab("Daily Load")
```



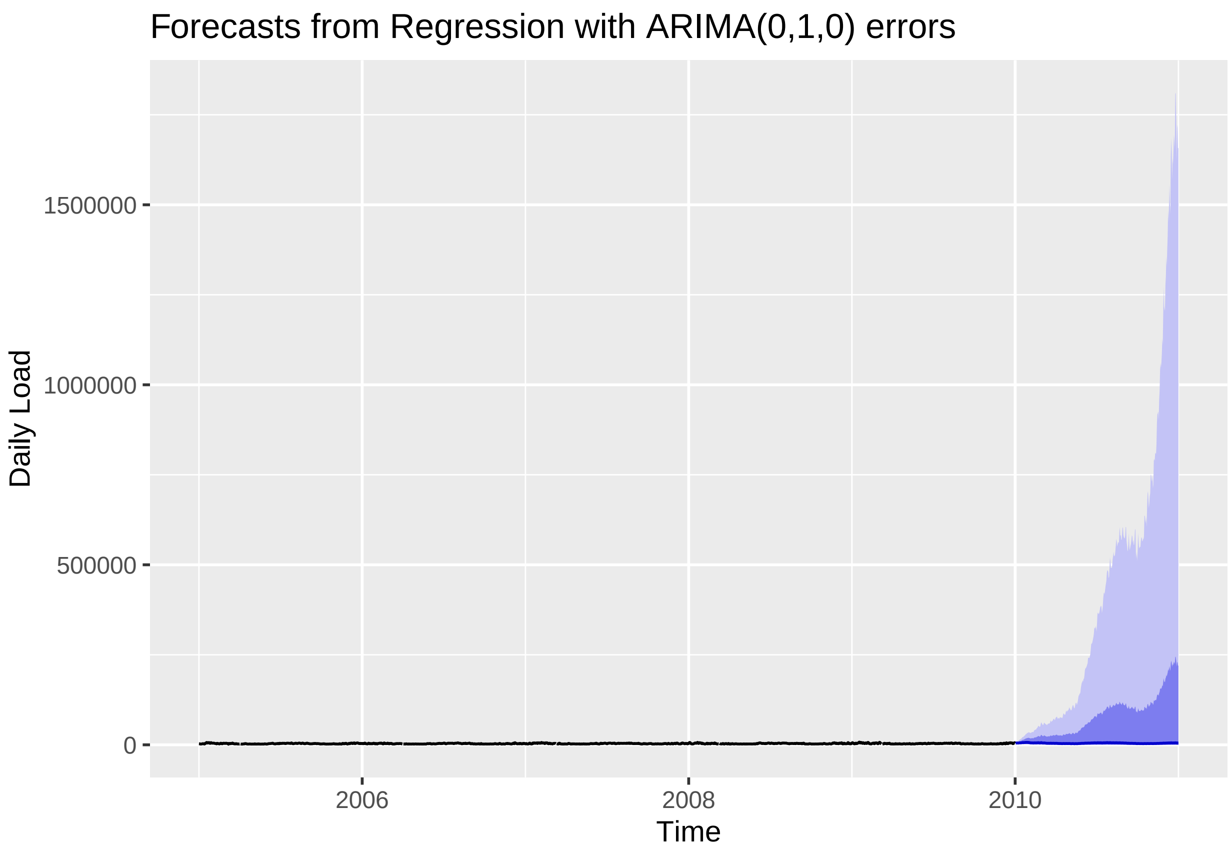
```
#Plot model + observed data
autoplot(ts_load_daily_avg) +
  autolayer(ARIMA_Four_temp_for, series="ARIMA_FOURIER",PI=FALSE) +
  ylab("Daily Load")
```



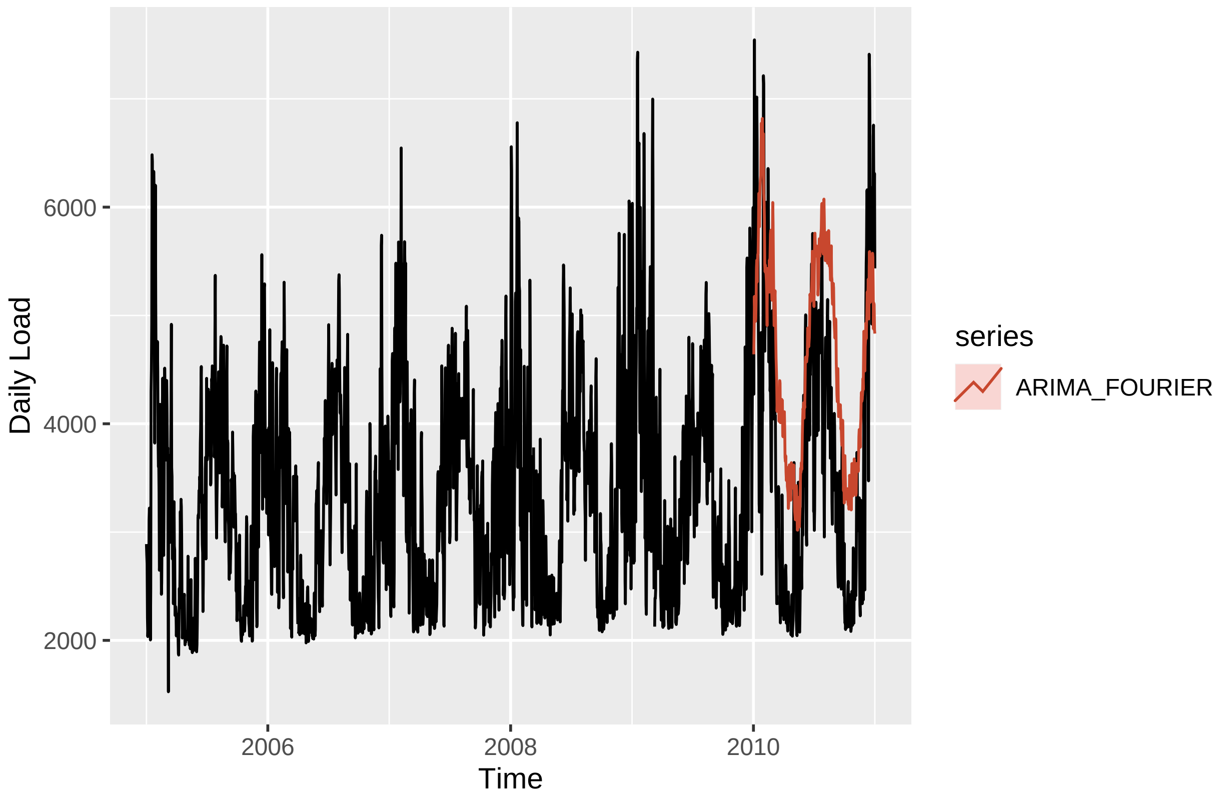
```
#Arima model with fourier terms + humidity
ARIMA_Four_humid_fit <- auto.arima(ts_load_daily_avg_train,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=cbind(ts_humid_daily_train,fourier(ts_load_daily_avg_train,
                                          K=c(2,12))
                             ))


ARIMA_Four_humid_for <- forecast::forecast(ARIMA_Four_humid_fit,
                           xreg=cbind(ts_humid_daily_test[1:365],fourier(ts_load_daily_avg_train,
                                        K=c(2,12),
                                        h=365)),
                           h=365
                           )
```

```
## Warning in forecast.forecast_ARIMA(ARIMA_Four_humid_fit, xreg =
## cbind(ts_humid_daily_test[1:365], : xreg contains different column names from
## the xreg used in training. Please check that the regressors are in the same
## order.
```

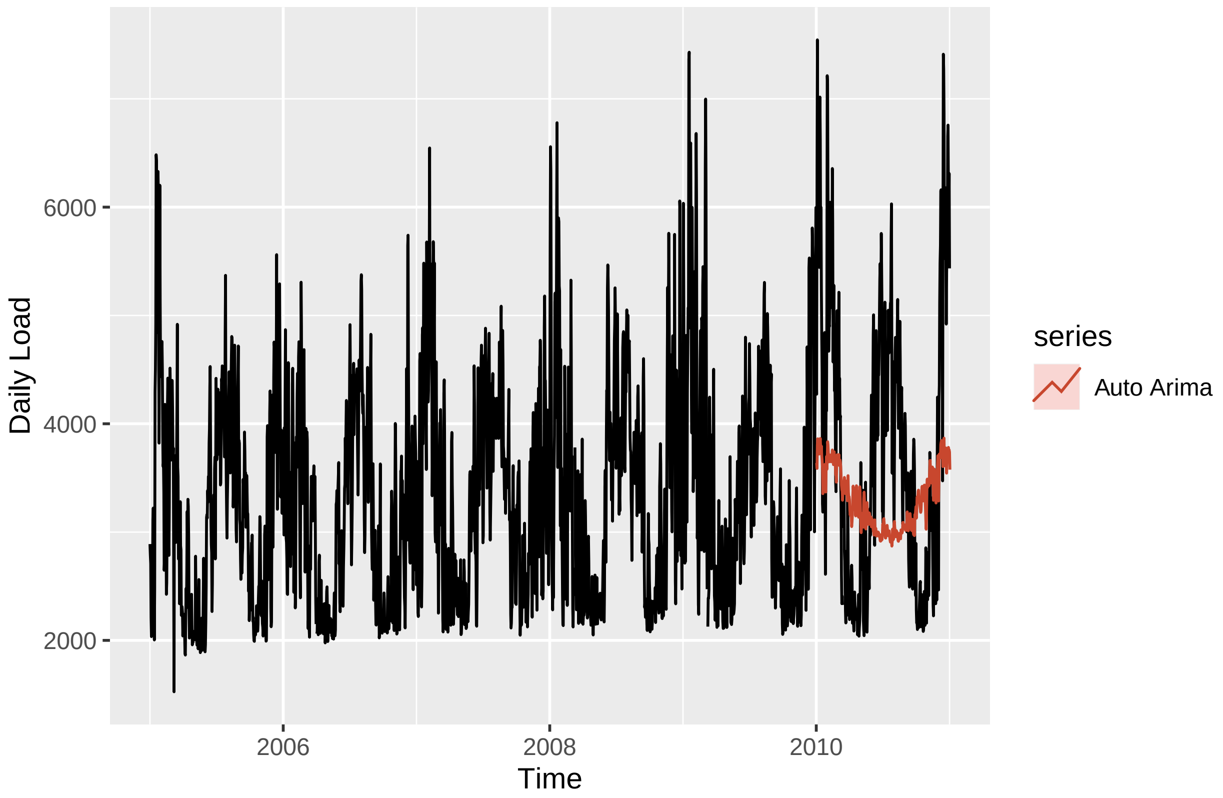
```
#Plot foresting results
autoplot(ARIMA_Four_humid_for) + ylab("Daily Load")
```



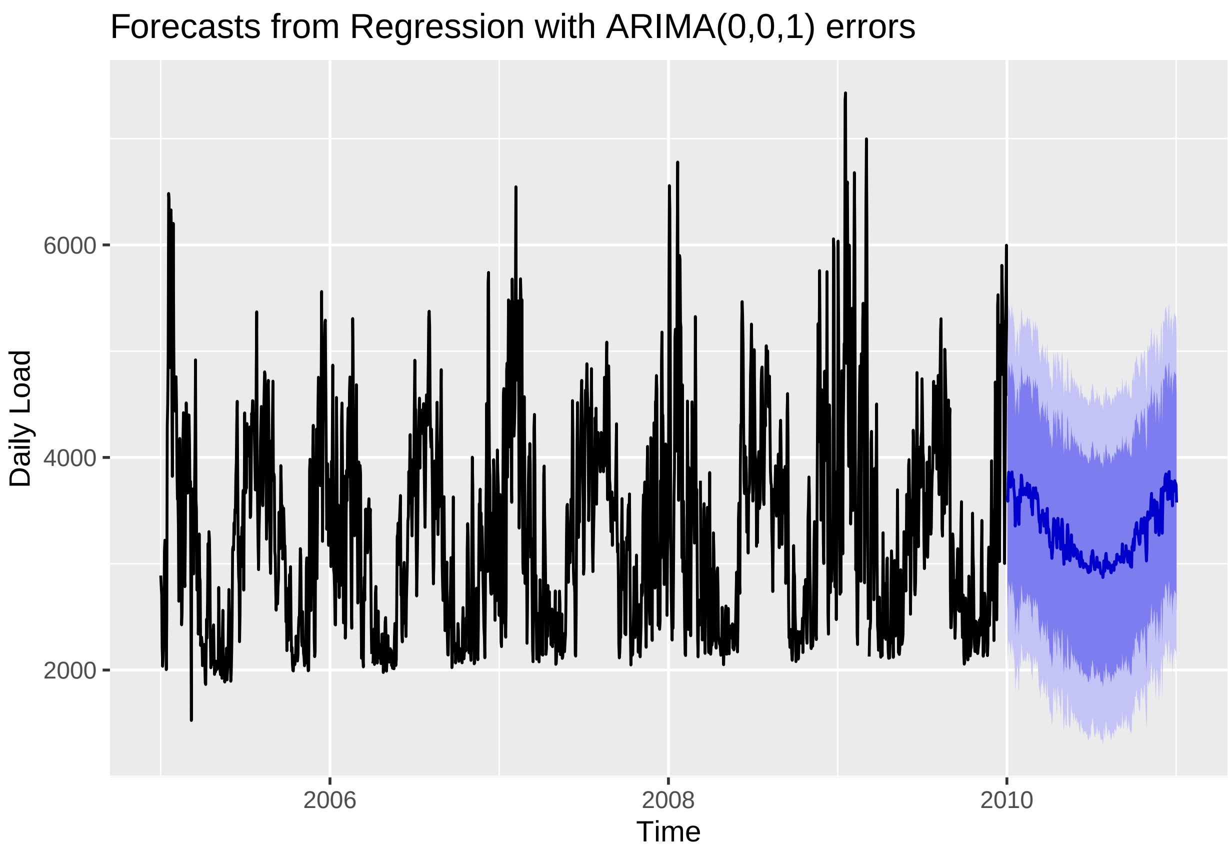
```
#Plot model + observed data
autoplot(ts_load_daily_avg) +
  autolayer(ARIMA_Four_humid_for, series="ARIMA_FOURIER",PI=FALSE) +
  ylab("Daily Load")
```



```
#Auto arima with temperature
ARIMA_with_temp_autofit <- auto.arima(ts_load_daily_avg_train, max.D = 0, max.P = 0, max.Q = 0,xreg=ts_

ARIMA_with_temp_forecast <- forecast::forecast(object = ARIMA_with_temp_autofit,xreg=ts_temp_daily_test

autoplot(ts_load_daily_avg) +
  autolayer(ARIMA_with_temp_forecast, series="Auto Arima",PI=FALSE)+
  ylab("Daily Load")
```



```
autoplot(ARIMA_with_temp_forecast) +
  ylab("Daily Load")
```



```r
#Model 1: STL + ETS
ETS_scores <- accuracy(ETS_fit$mean,ts_load_daily_avg_test)

#Model 2: ARIMA + Fourier
ARIMA_scores <- accuracy(ARIMA_Four_for$mean,ts_load_daily_avg_test)

# Model 3:  TBATS
TBATS_scores <- accuracy(TBATS_for$mean,ts_load_daily_avg_test)

# Model 4:  Neural Network
NN_scores <- accuracy(NN_for$mean,ts_load_daily_avg_test)

# Model 5: Seasonal Naive
SNAIVE_scores <- accuracy(SNAIVE$mean,ts_load_daily_avg_test)

#Model 6: Auto Arima
AutoArima_scores <- accuracy(ARIMA_forecast$mean,ts_load_daily_avg_test)
```

```r
scores <- as.data.frame(
  rbind(ETS_scores, ARIMA_scores, TBATS_scores, NN_scores, SNAIVE_scores,AutoArima_scores)
  )
row.names(scores) <- c("STL+ETS", "ARIMA+Fourier","TBATS","NN", "SNAIVE", "Auto Arima")

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

```
## The best model by RMSE is: ARIMA+Fourier
```

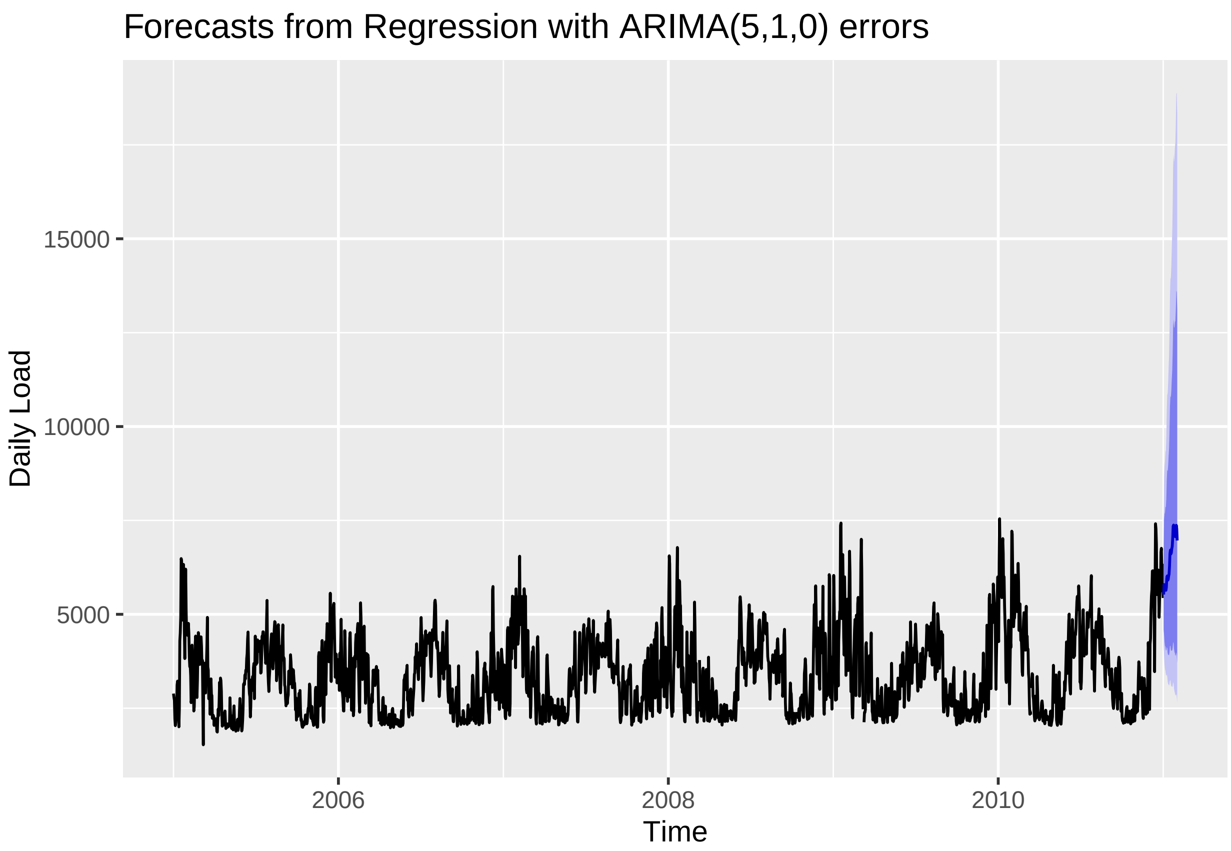
```r
kbl(scores,
      caption = "Forecast Accuracy for Load",
      digits = array(5,ncol(scores))) %>%
  kable_styling(full_width = FALSE, position = "center", latex_options = "hold_position") %>%
  #highlight model with lowest RMSE
  kable_styling(latex_options="striped", stripe_index = which.min(scores[,"RMSE"]))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

Table 1: Forecast Accuracy for Load

	ME	RMSE	MAE	MPE	MAPE	ACF1	Theil's U
STL+ETS	-662.0823	1229.251	1038.4385	-27.07241	33.15822	0.79853	2.79951
ARIMA+Fourier	-532.5219	1097.439	913.7556	-21.65772	28.02236	0.84122	2.34306
TBATS	936.1087	1619.426	1230.4608	15.34879	28.36684	0.91559	2.34101
NN	461.3108	1124.543	776.4283	6.96826	18.11293	0.82425	1.66166
SNAIVE	327.1963	1224.535	877.9736	3.52559	21.67954	0.72619	1.95747
Auto Arima	431.8726	1349.124	1087.2661	0.67161	28.46012	0.90965	2.30293

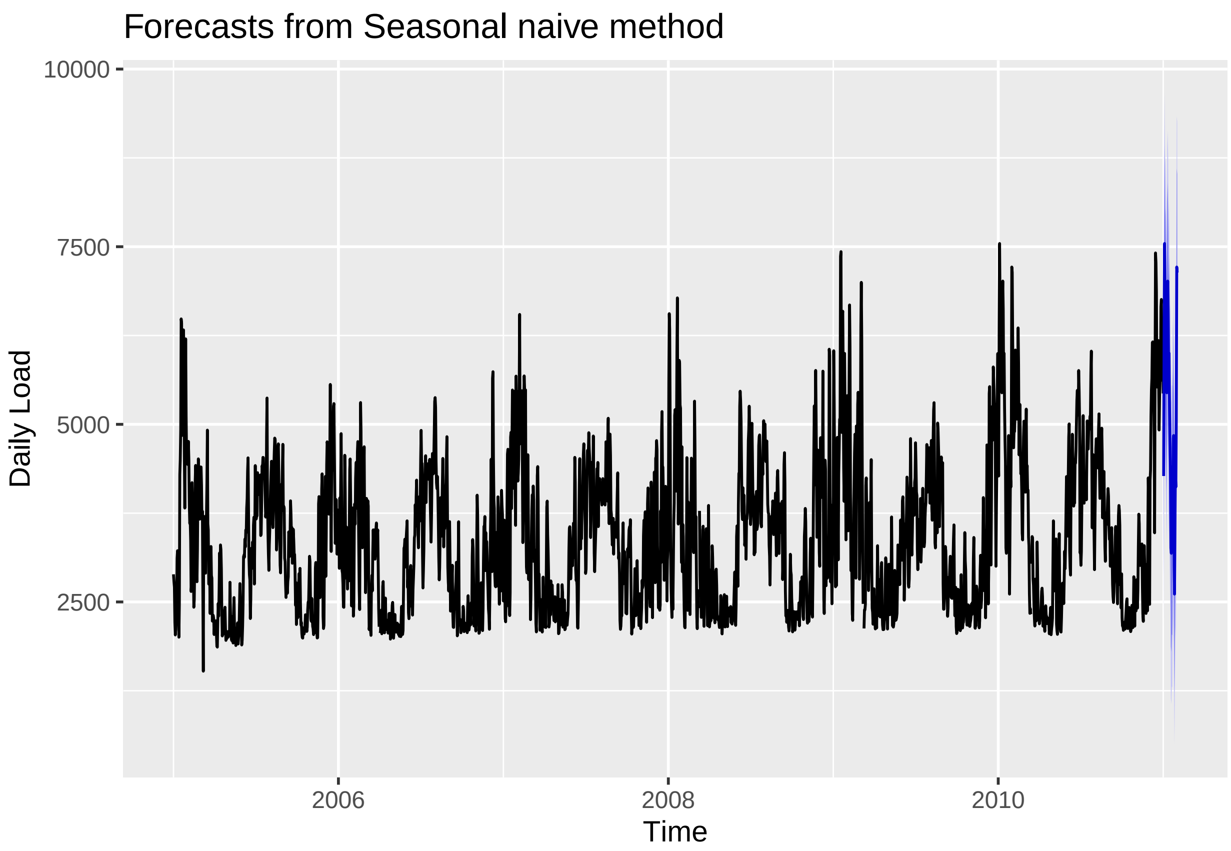
#January 2011 Forecasts

```r
#Arima model with fourier terms
ARIMA_Four_fit2 <- auto.arima(ts_load_daily_avg,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(ts_load_daily_avg,
                                          K=c(2,12))
                             )
ARIMA_Four_for2 <- forecast::forecast(ARIMA_Four_fit2,
                           xreg=fourier(ts_load_daily_avg,
                                        K=c(2,12),
                                        h=31),
                           h=31
                           )

#Plot forecasting results
autoplot(ARIMA_Four_for2) + ylab("Daily Load")
```



```r
#Convert forecasting results to dataframe
Forecast1 <- data.frame(load = ARIMA_Four_for2[["mean"]])
```

```r
#Seasonal naive model
SNAIVE_for <- snaive(ts_load_daily_avg, h=31)
autoplot(SNAIVE_for) +
  ylab("Daily Load")
```

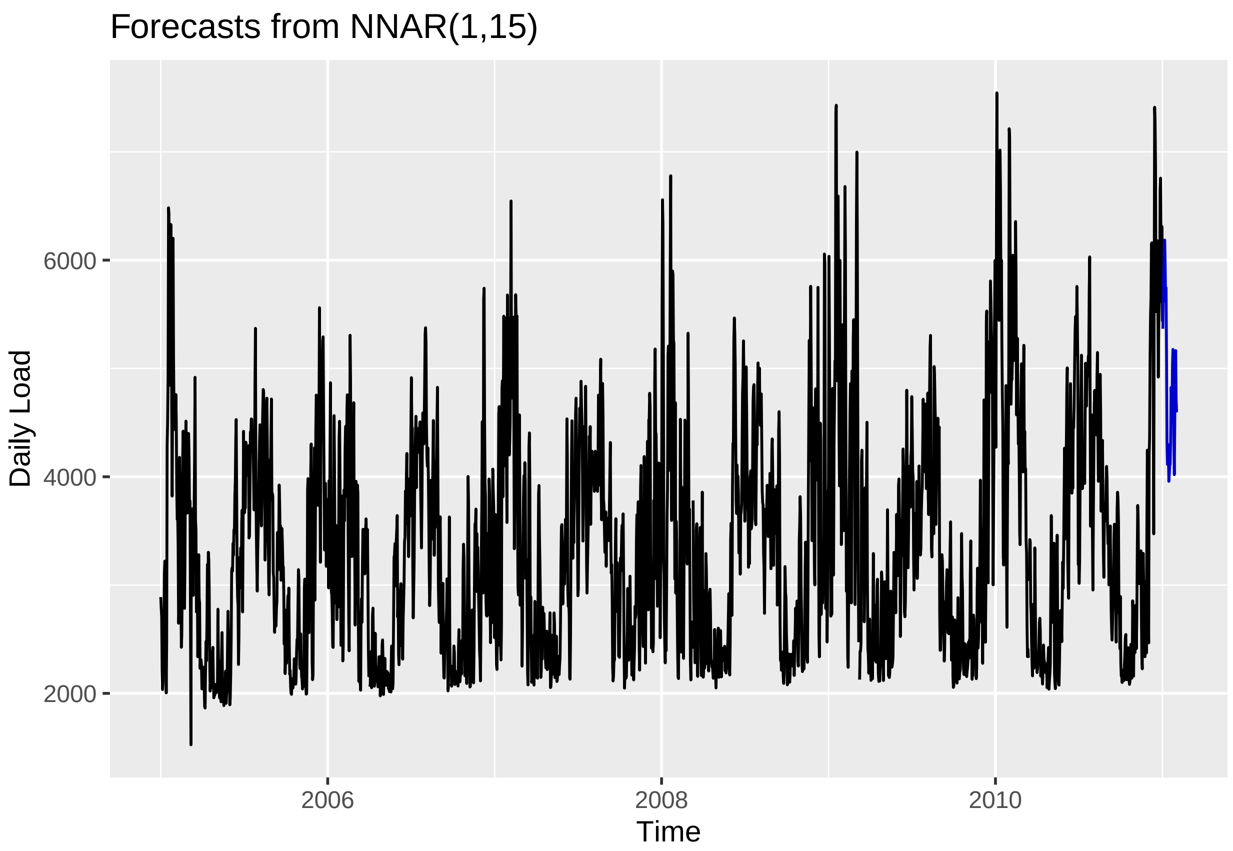


```
#Convert forecasting results to dataframe
Forecast2 <- data.frame(load = SNAIVE_for[["mean"]])
```

```
#NN+fourier
NN_fit2 <- nnetar(ts_load_daily_avg,p=1,P=0,xreg=fourier(ts_load_daily_avg, K=c(2,12)))
```

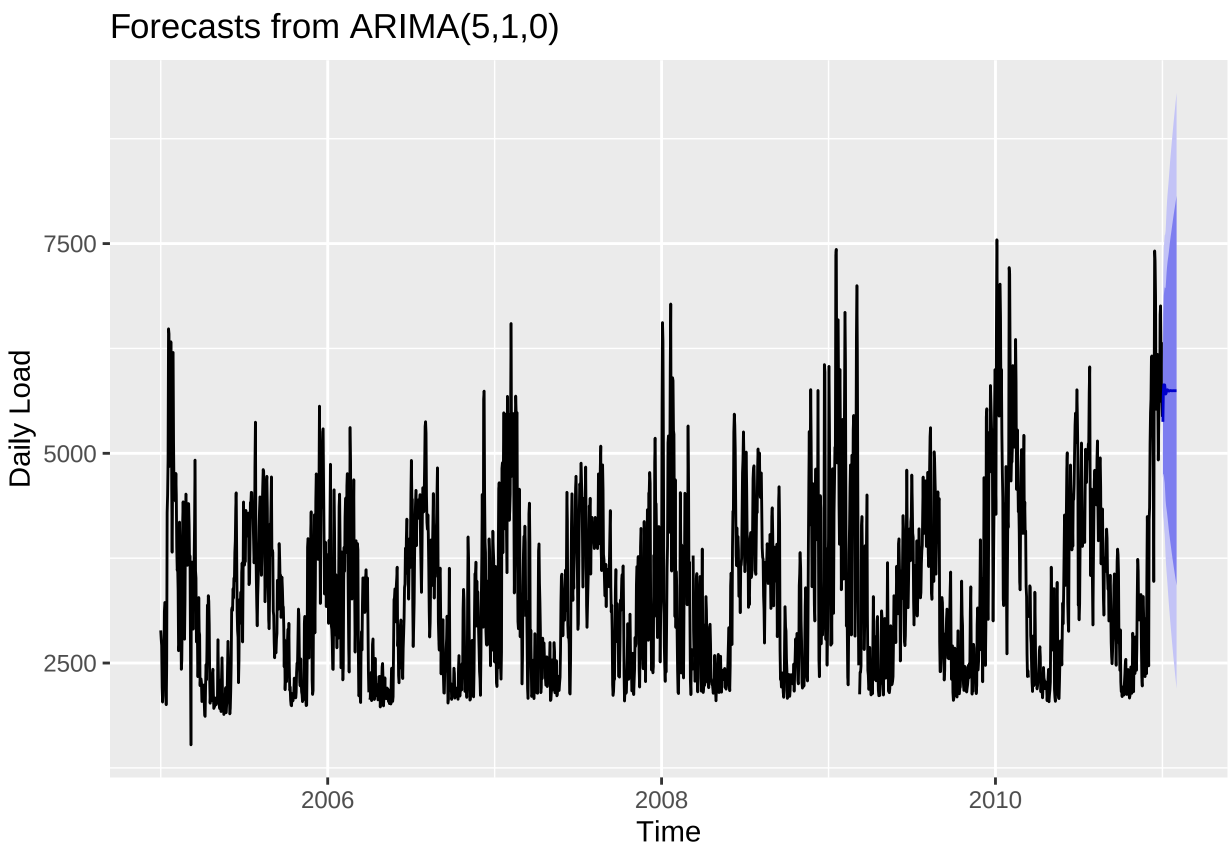
```
## Warning in nnetar(ts_load_daily_avg, p = 1, P = 0, xreg =
## fourier(ts_load_daily_avg, : Missing values in x, omitting rows
```

```
NN_for2 <- forecast::forecast(NN_fit2, h=31,xreg=fourier(ts_load_daily_avg,
                                          K=c(2,12),h=31))

#Plot foresting results
autoplot(NN_for2) +
  ylab("Daily Load")
```



```
#Convert forecasting results to dataframe
Forecast3 <- data.frame(load = NN_for2[["mean"]])
```

```
#SARIMA
SARIMA_autofit3 <- auto.arima(ts_load_daily_avg)

SARIMA_forecast3 <- forecast::forecast(object = SARIMA_autofit3,h = 31)

autoplot(SARIMA_forecast3) + ylab("Daily Load")
```

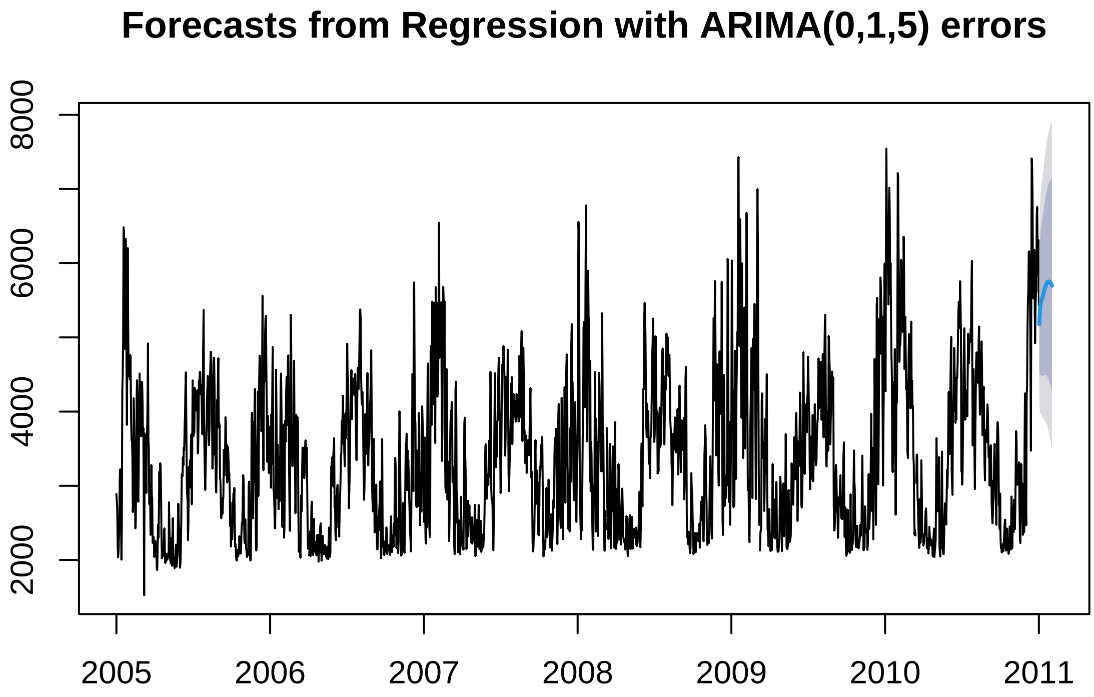


```
#Convert forecasting results to dataframe
Forecast4 <- data.frame(load = SARIMA_forecast3[["mean"]])
```

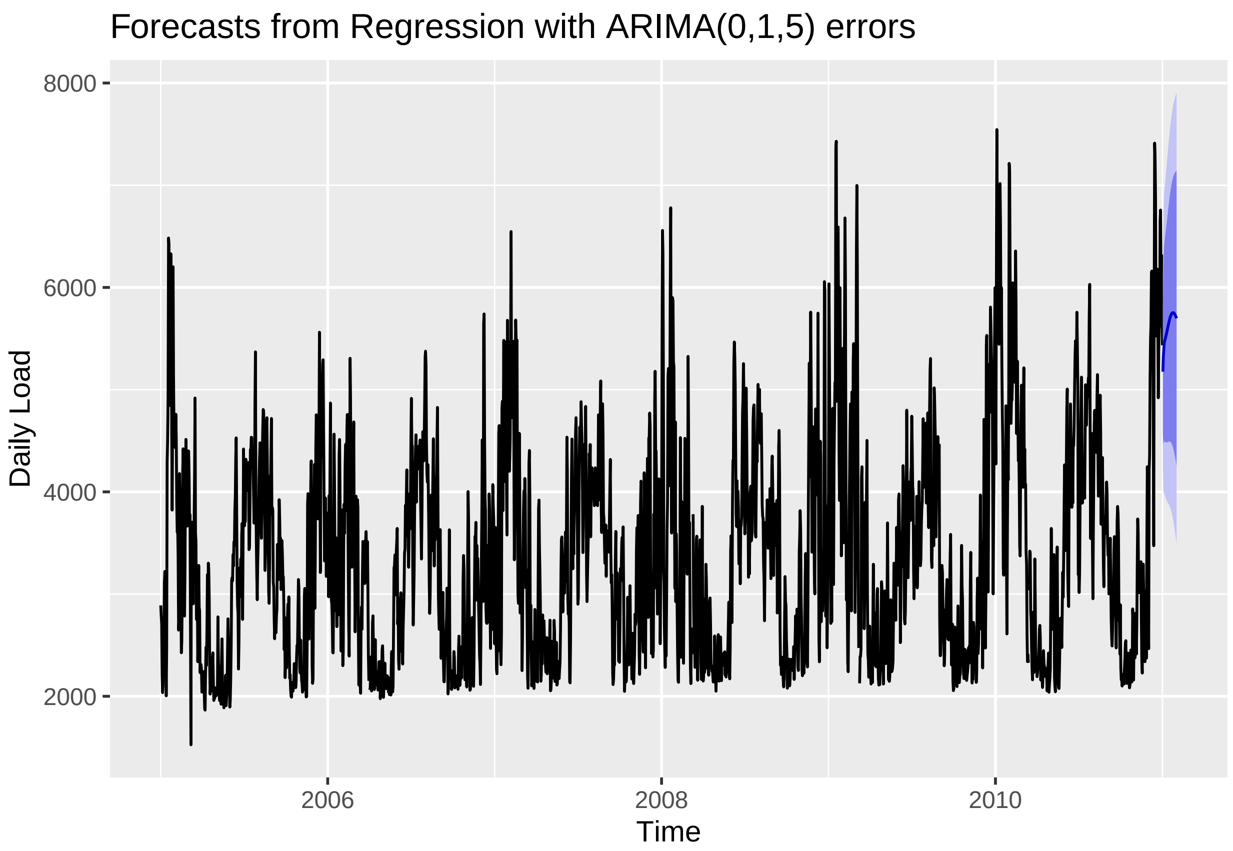
```
#SARIMA forecast with temperature
SARIMA_autofit2 <- auto.arima(ts_load_daily_avg,xreg=ts_temp_daily)
print(SARIMA_autofit2)
```

```
## Series: ts_load_daily_avg
## Regression with ARIMA(0,1,5) errors
##
## Coefficients:
##           ma1      ma2      ma3      ma4      ma5   drift      xreg
##       -0.0434  -0.3345  -0.1799  -0.0940  -0.0075  1.0690  -40.3669
## s.e.   0.0216   0.0216   0.0218   0.0212   0.0214  3.4539    2.0949
##
## sigma^2 = 224819:  log likelihood = -16554.61
## AIC=33125.23   AICc=33125.29   BIC=33170.76
```

```
SARIMA_forecast2 <- forecast::forecast(object = SARIMA_autofit2, xreg=ts_temp_for2,h = 31)
plot(SARIMA_forecast2)
```



```
autoplot(SARIMA_forecast2) + ylab("Daily Load")
```

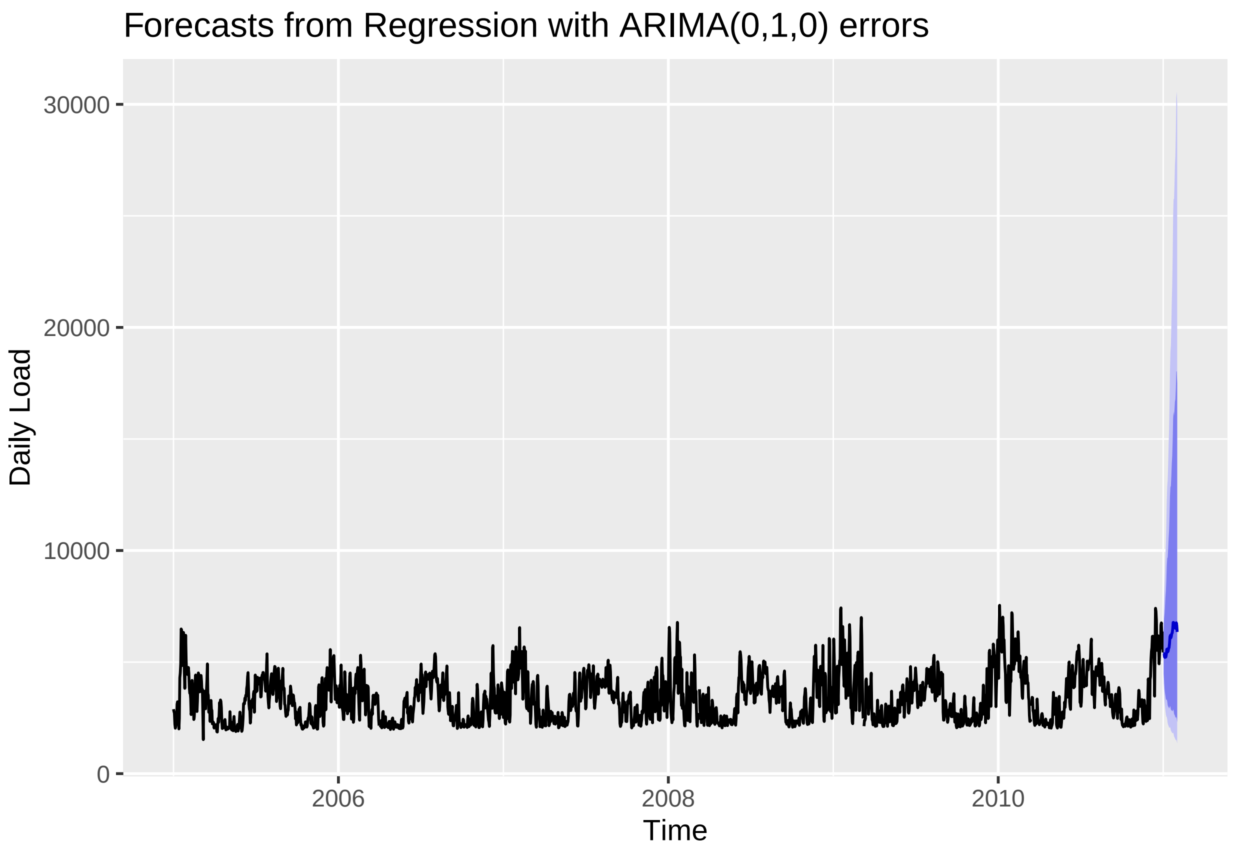


```r
#Convert forecasting results to dataframe
Forecast5 <- data.frame(load = SARIMA_forecast3[["mean"]])
```

```r
#Arima+fourier+temp
ARIMA_Four_temp_fit2 <- auto.arima(ts_load_daily_avg,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=cbind(ts_temp_daily,fourier(ts_load_daily_avg,
                                          K=c(2,12))
                             ))


ARIMA_Four_temp_for2 <- forecast::forecast(ARIMA_Four_temp_fit2,
                           xreg=cbind(ts_temp_for2,fourier(ts_load_daily_avg,
                                        K=c(2,12),
                                        h=31)),
                           h=31
                           )
```

```
## Warning in forecast.forecast_ARIMA(ARIMA_Four_temp_fit2, xreg =
## cbind(ts_temp_for2, : xreg contains different column names from the xreg used in
## training. Please check that the regressors are in the same order.
```

```
autoplot(ARIMA_Four_temp_for2) + ylab("Daily Load")
```

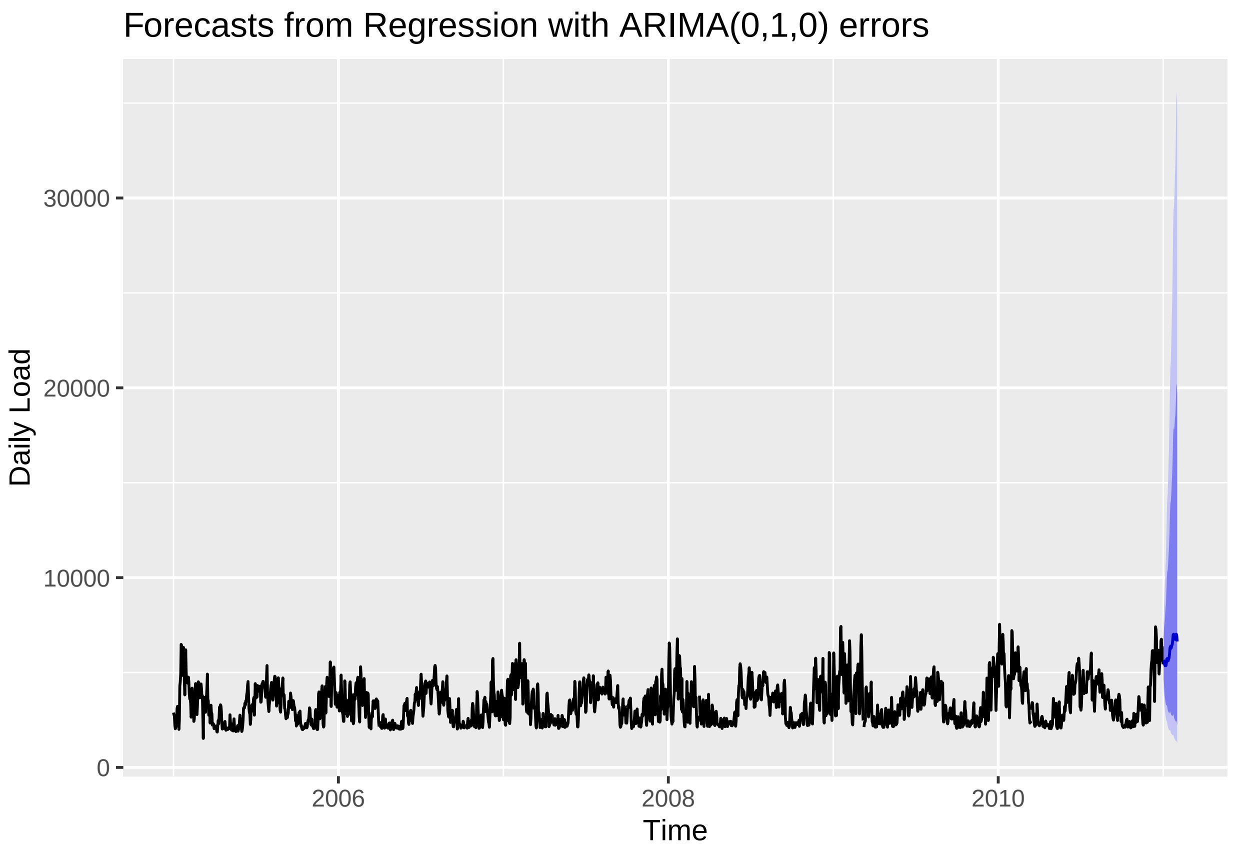


```
Forecast6 <- data.frame(load = ARIMA_Four_temp_for2[["mean"]])
```

```
#Arima+fourier+humidity
ARIMA_Four_humid_fit2 <- auto.arima(ts_load_daily_avg,
                                seasonal=FALSE,
                                lambda=0,
                                xreg=cbind(ts_humid_daily,fourier(ts_load_daily_avg,
                                             K=c(2,12))
                                ))


ARIMA_Four_humid_for2 <- forecast::forecast(ARIMA_Four_humid_fit2,
                              xreg=cbind(ts_humid_for2,fourier(ts_load_daily_avg,
                                           K=c(2,12),
                                           h=31)),
                              h=31
                              )
```

```
## Warning in forecast.forecast_ARIMA(ARIMA_Four_humid_fit2, xreg =
## cbind(ts_humid_for2, : xreg contains different column names from the xreg used
## in training. Please check that the regressors are in the same order.
```

```
autoplot(ARIMA_Four_humid_for2) + ylab("Daily Load")
```



```
Forecast7 <- data.frame(load = ARIMA_Four_humid_for2[["mean"]])
```

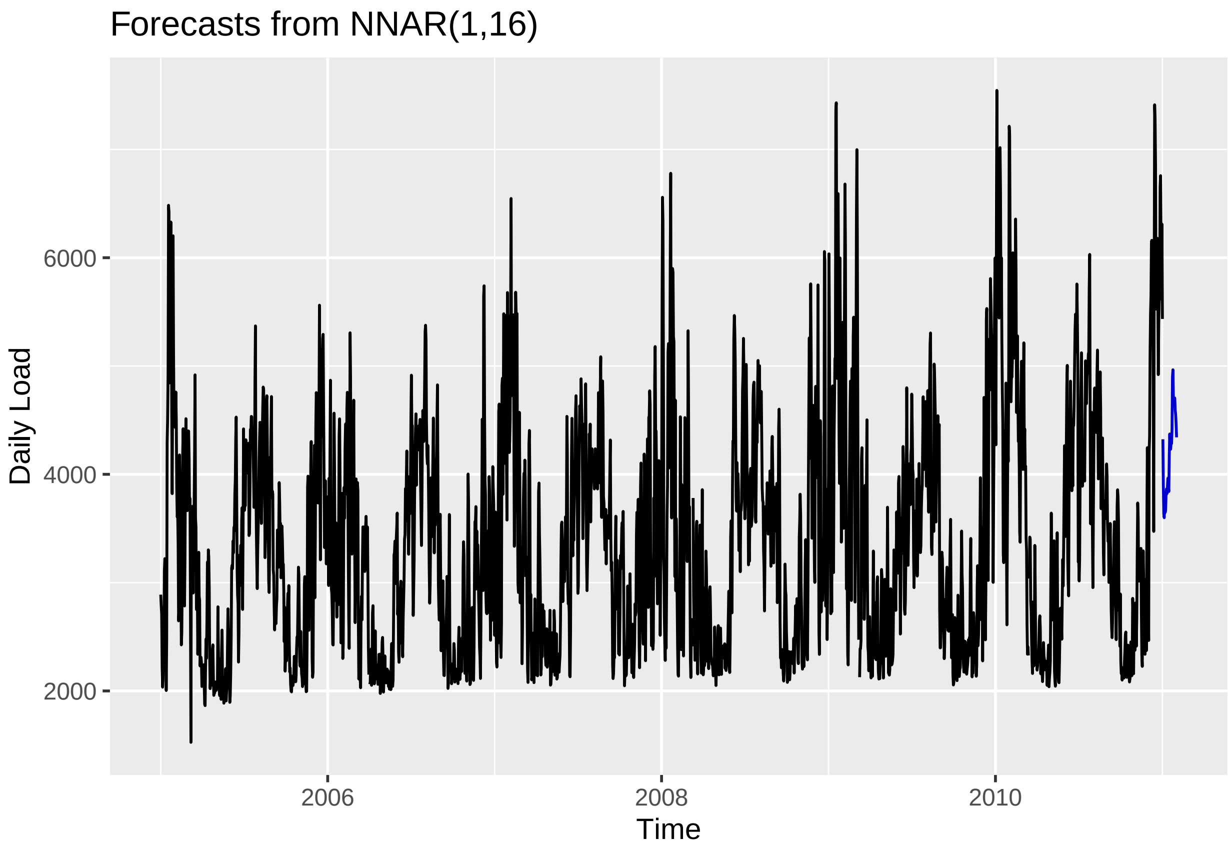
```
#NN+fourier+temperature and forcing p=1,P=0
NN_temp_fit <- nnetar(ts_load_daily_avg,p=1,P=0,xreg=cbind(ts_temp_daily,fourier(ts_load_daily_avg,
                                                           K=c(2,12))
                          ))
```

```
## Warning in nnetar(ts_load_daily_avg, p = 1, P = 0, xreg = cbind(ts_temp_daily, :
## Missing values in x, omitting rows
```

```
NN_temp_for <- forecast::forecast(NN_temp_fit,xreg=cbind(ts_temp_for2,fourier(ts_load_daily_avg,
                                         K=c(2,12),
                                         h=31)),
                          h=31
                          )
```

```
## Warning in forecast.nnetar(NN_temp_fit, xreg = cbind(ts_temp_for2,
## fourier(ts_load_daily_avg, : xreg contains different column names from the xreg
## used in training. Please check that the regressors are in the same order.
```

```
#Plot foresting results
autoplot(NN_temp_for) +
  ylab("Daily Load")
```



```
#Convert forecasting results to dataframe

Forecast8 <- data.frame(load = NN_temp_for[["mean"]])
```

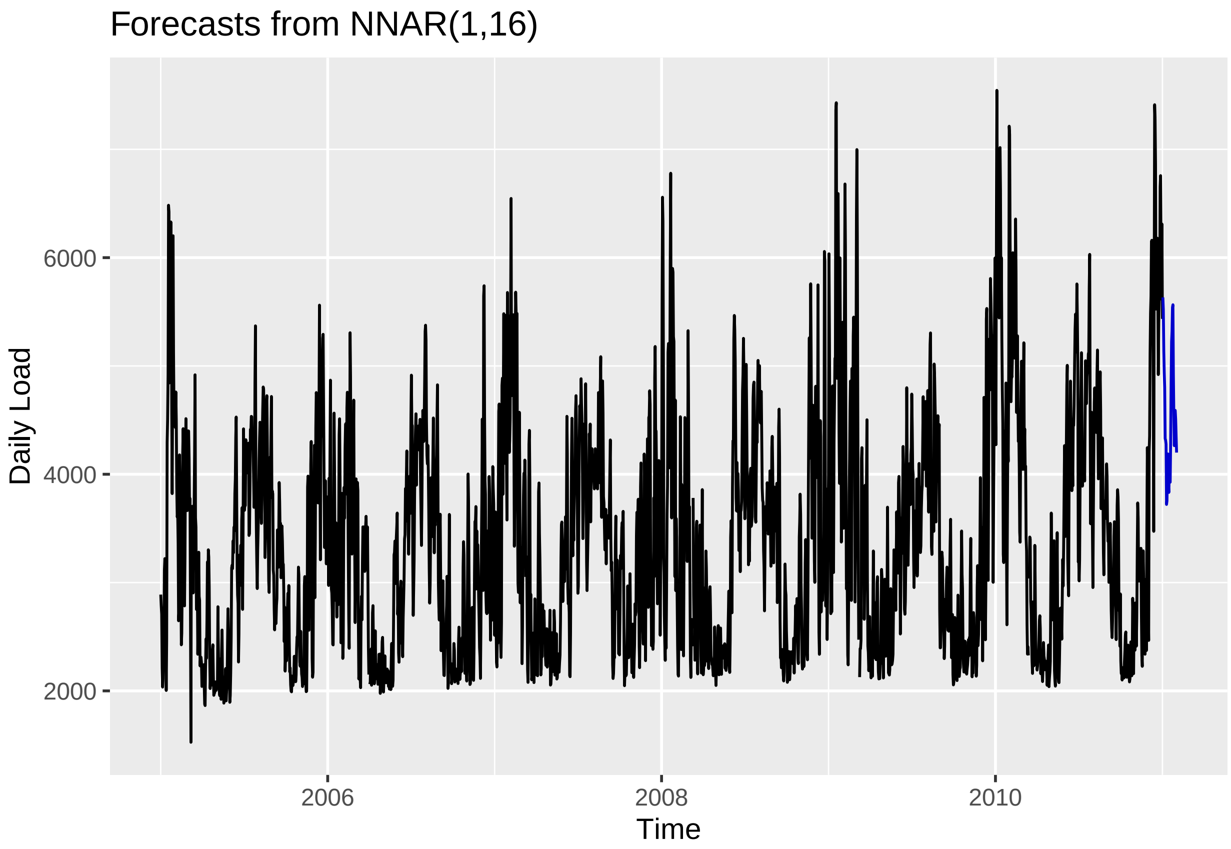
```
#NN+fourier+humidity and forcing p=1,P=0
NN_humid_fit <- nnetar(ts_load_daily_avg,p=1,P=0,xreg=cbind(ts_humid_daily,fourier(ts_load_daily_avg,
                                                     K=c(2,12))
                             ))
```

```
## Warning in nnetar(ts_load_daily_avg, p = 1, P = 0, xreg =
## cbind(ts_humid_daily, : Missing values in x, omitting rows
```

```
NN_humid_for <- forecast::forecast(NN_humid_fit,xreg=cbind(ts_humid_for2,fourier(ts_load_daily_avg,
                                       K=c(2,12),
                                       h=31)),
                           h=31
                           )
```

```
## Warning in forecast.nnetar(NN_humid_fit, xreg = cbind(ts_humid_for2,
## fourier(ts_load_daily_avg, : xreg contains different column names from the xreg
## used in training. Please check that the regressors are in the same order.
```

```
#Plot foresting results
autoplot(NN_humid_for) +
  ylab("Daily Load")
```

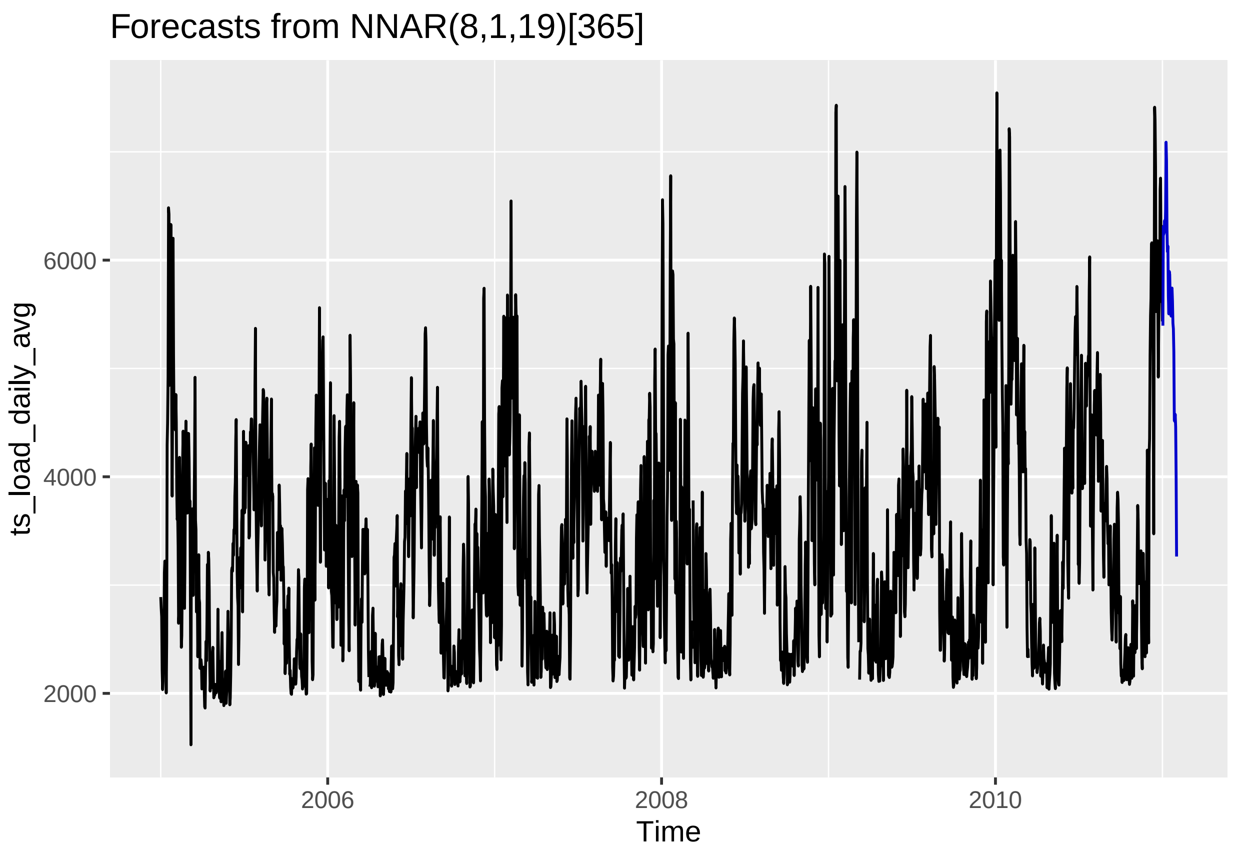


```
#Convert forecasting results to dataframe
Forecast9 <- data.frame(load = NN_humid_for[["mean"]])
```

```
#NN+fourier
NN_fit3 <- nnetar(ts_load_daily_avg,xreg=fourier(ts_load_daily_avg, K=c(2,12)))
```

```
## Warning in nnetar(ts_load_daily_avg, xreg = fourier(ts_load_daily_avg, K =
## c(2, : Missing values in x, omitting rows
```

```
#NN_for <- forecast(NN_fit, h=365)
NN_for3 <- forecast::forecast(NN_fit3,xreg=fourier(ts_load_daily_avg,
                                          K=c(2,12),h=31))
autoplot(NN_for3)
```



```
Forecast10 <- data.frame(load = NN_for3[["mean"]])
```

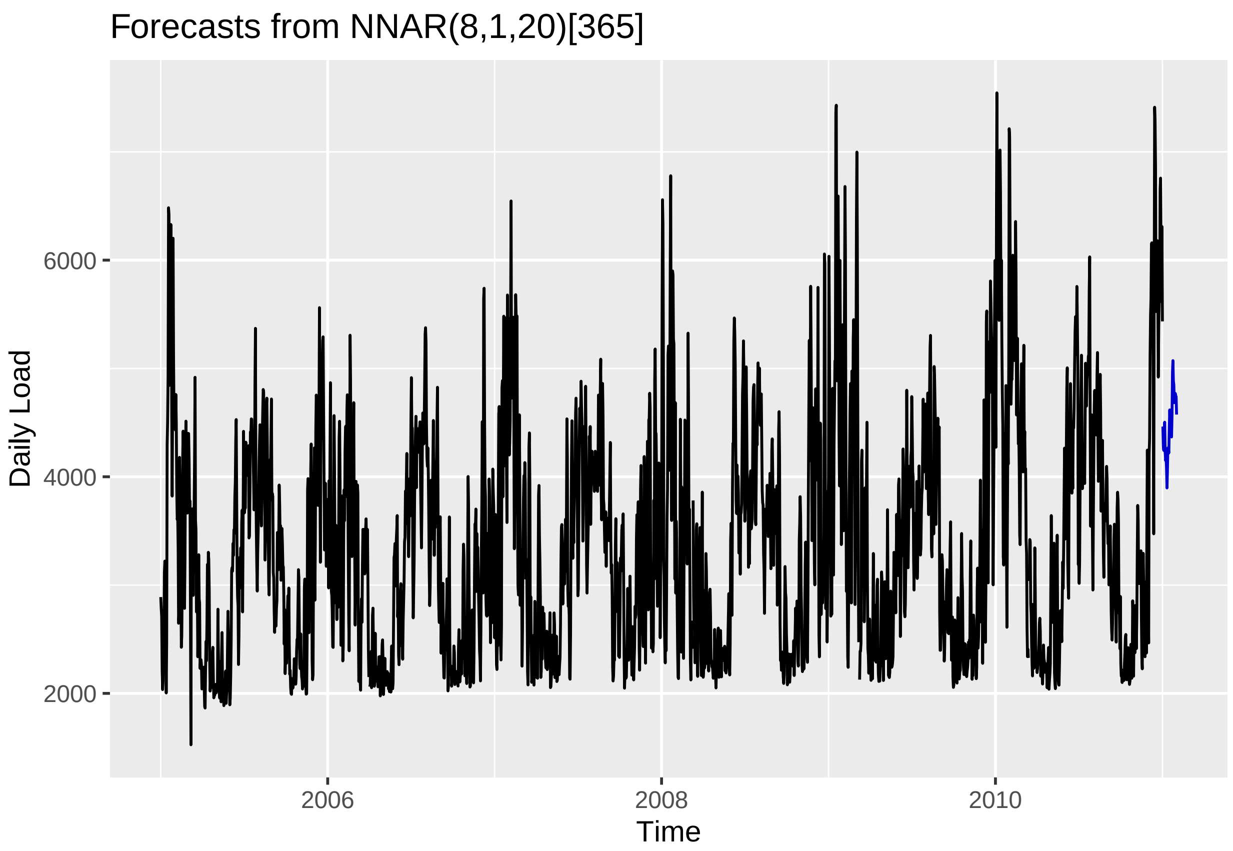
```
NN_humid_fit4 <- nnetar(ts_load_daily_avg,xreg=cbind(ts_temp_daily,fourier(ts_load_daily_avg,
                                                     K=c(2,12))
                              ))
```

```
## Warning in nnetar(ts_load_daily_avg, xreg = cbind(ts_temp_daily,
## fourier(ts_load_daily_avg, : Missing values in x, omitting rows
```

```
NN_humid_for4 <- forecast::forecast(NN_humid_fit4,xreg=cbind(ts_temp_for2,fourier(ts_load_daily_avg,
                                        K=c(2,12),
                                        h=31)),
                           h=31
                           )
```

```
## Warning in forecast.nnetar(NN_humid_fit4, xreg = cbind(ts_temp_for2,
## fourier(ts_load_daily_avg, : xreg contains different column names from the xreg
## used in training. Please check that the regressors are in the same order.
```

```
#Plot foresting results
autoplot(NN_humid_for4) +
  ylab("Daily Load")
```



```
#Convert forecasting results to dataframe
Forecast11 <- data.frame(load = NN_humid_for4[["mean"]])
```

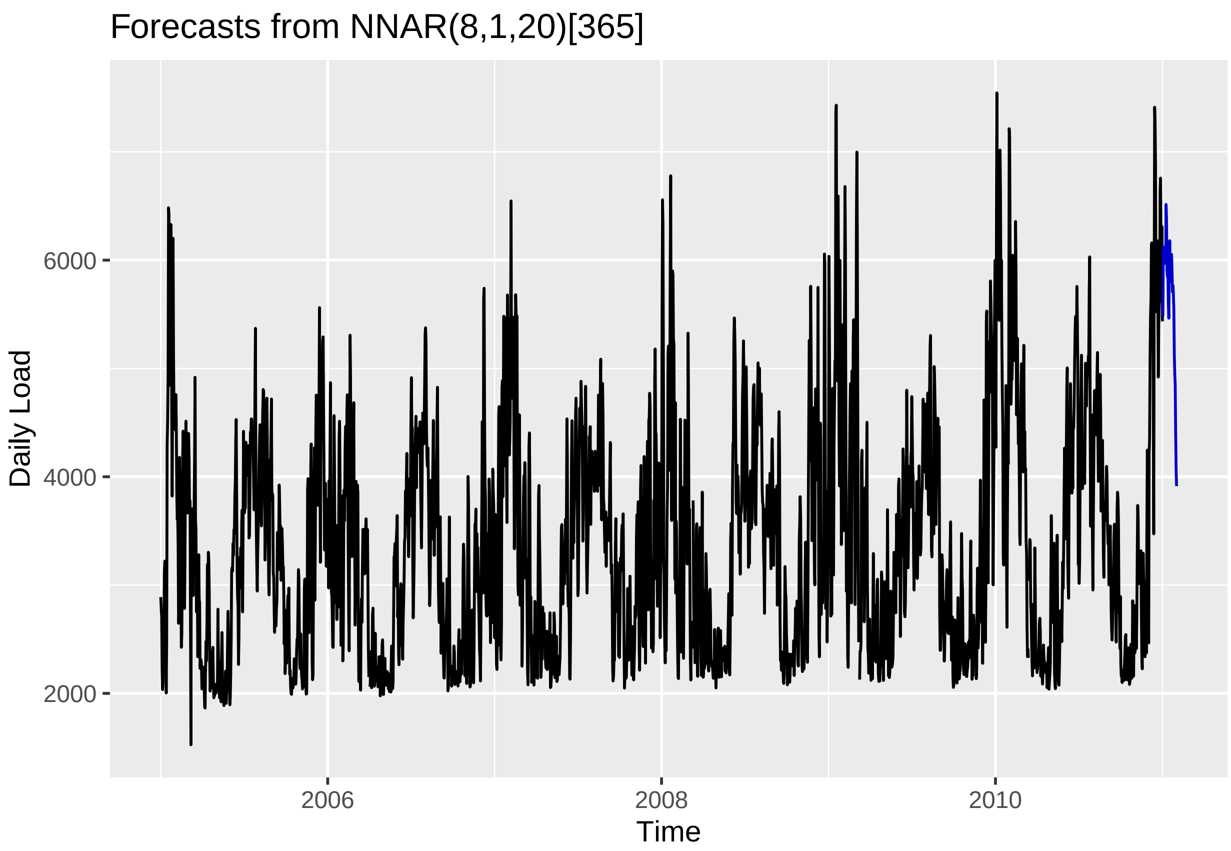
```
#NN+fourier+humidity
NN_humid_fit5 <- nnetar(ts_load_daily_avg,xreg=cbind(ts_humid_daily,fourier(ts_load_daily_avg,
                                              K=c(2,12))
                              ))
```

```
## Warning in nnetar(ts_load_daily_avg, xreg = cbind(ts_humid_daily,
## fourier(ts_load_daily_avg, : Missing values in x, omitting rows
```

```
NN_humid_for5 <- forecast::forecast(NN_humid_fit5,xreg=cbind(ts_humid_for2,fourier(ts_load_daily_avg,
                                        K=c(2,12),
                                        h=31)),
                           h=31
                           )
```

```
## Warning in forecast.nnetar(NN_humid_fit5, xreg = cbind(ts_humid_for2,
## fourier(ts_load_daily_avg, : xreg contains different column names from the xreg
## used in training. Please check that the regressors are in the same order.
```

```
#Plot foresting results
autoplot(NN_humid_for5) +
  ylab("Daily Load")
```



```
#Convert forecasting results to dataframe
Forecast12 <- data.frame(load = NN_humid_for5[["mean"]])
```

```
#write.csv(Forecast1, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast1

#write.csv(Forecast2, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast2

#write.csv(Forecast3, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast3

#write.csv(Forecast4, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast4

#write.csv(Forecast5, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast5

#write.csv(Forecast6, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast6

#write.csv(Forecast7, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast7

#write.csv(Forecast8, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast8

#write.csv(Forecast9, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast9

#write.csv(Forecast10, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast

#write.csv(Forecast11, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast

#write.csv(Forecast12, "~\\ENVIRON 790\\ENV790_TimeSeriesAnalysis_Sp2022\\Competition\\Output\\Forecast
```