TSA Forecasting Competition

Sarah Mansfield, Alex Baad

3/27/2022

```
library(tseries)
library(forecast)
library(tidyverse)
library(readxl)
library(kableExtra)
```

Goal: Forecast daily demand for January 2011 based on historical data

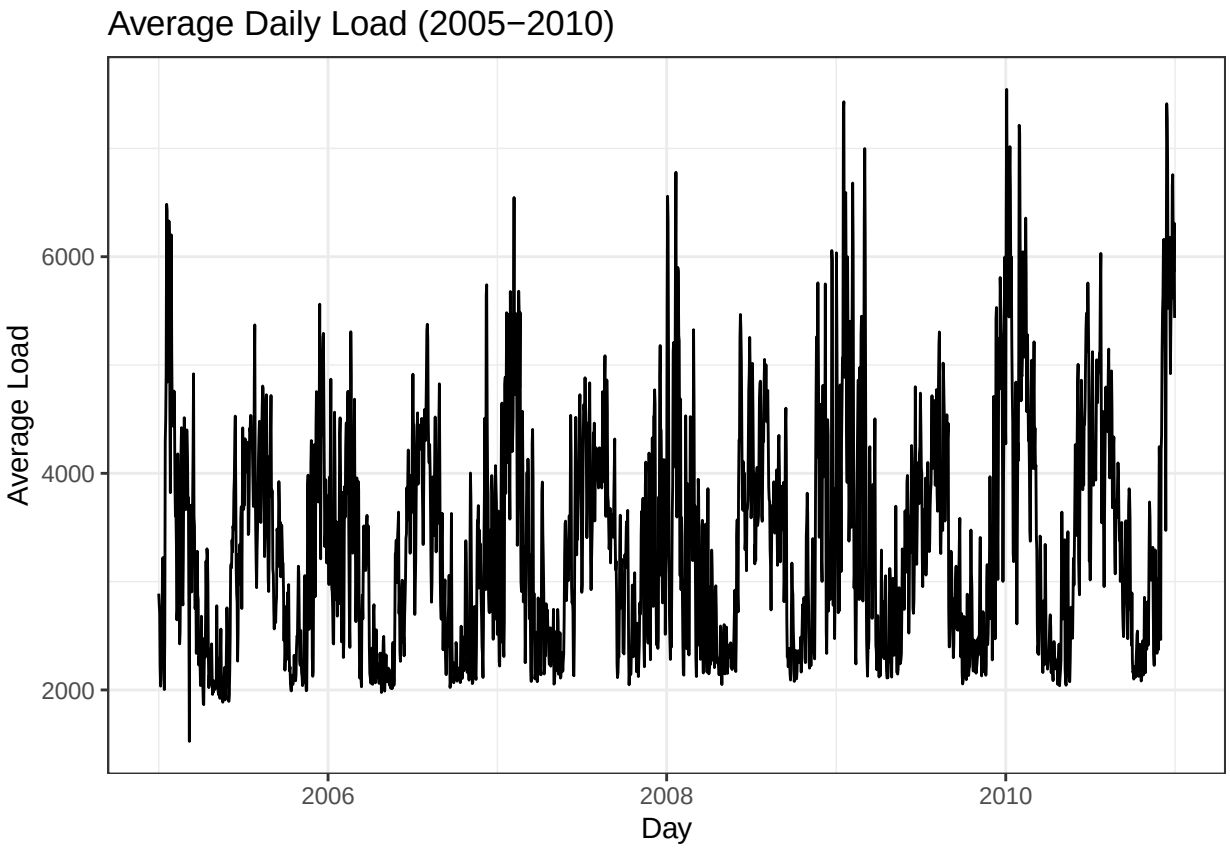
Processing the Data

```
load <- read_excel("data/load.xlsx")
relative_humidity <- read_excel("data/relative_humidity.xlsx")
temperature <- read_excel("data/temperature.xlsx")

# aggregate data
load <- load %>%
  select(-meter_id) %>%
  mutate(daily_avg_load = rowMeans(select(., starts_with("h")), na.rm = TRUE)) %>%
  select(date, daily_avg_load)

relative_humidity <- relative_humidity %>%
  mutate(avg = rowMeans(select(., starts_with("rh")), na.rm = TRUE)) %>%
  group_by(date) %>%
  summarise(daily_avg_humidity = mean(avg))

temperature <- temperature %>%
  mutate(avg = rowMeans(select(., starts_with("t")), na.rm = TRUE)) %>%
  group_by(date) %>%
  summarise(daily_avg_temp = mean(avg))

df <- inner_join(load, relative_humidity) %>%
  inner_join(temperature)
```

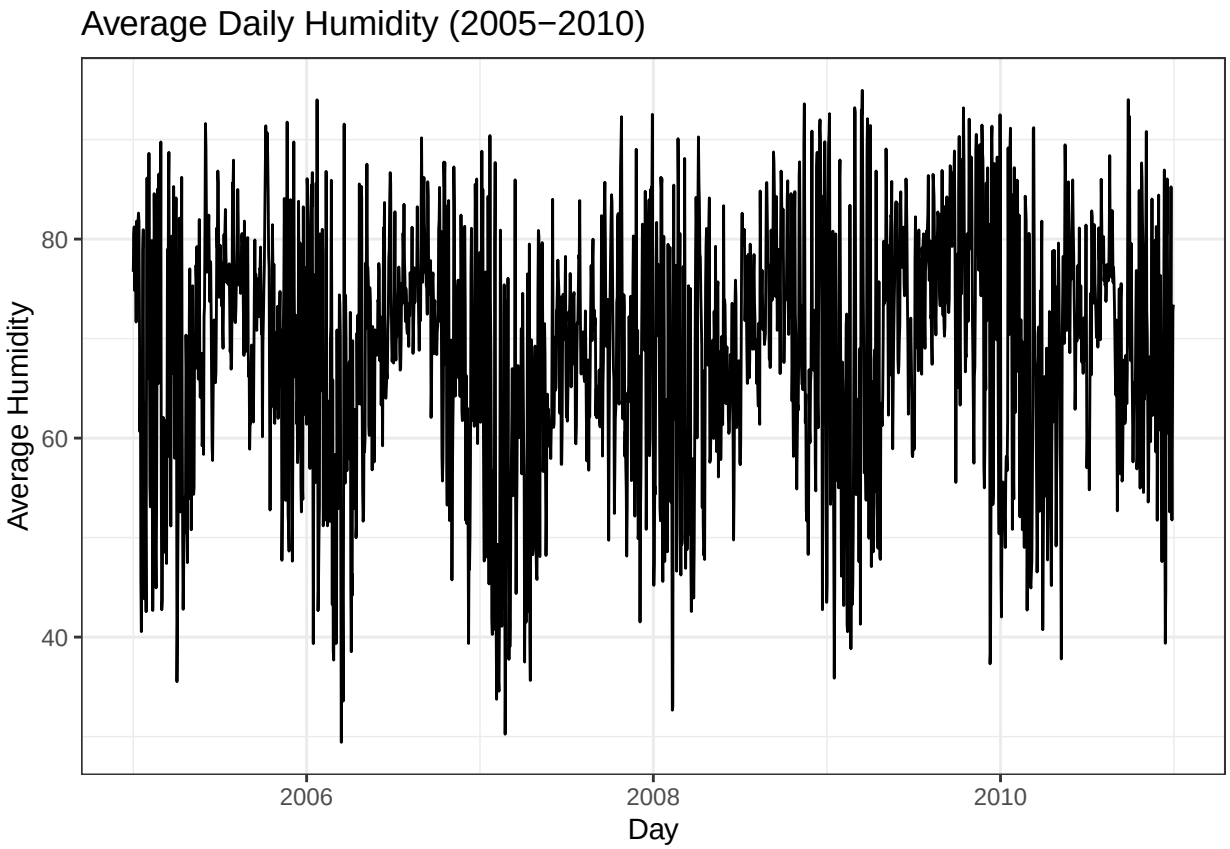
```
## Joining, by = "date"
## Joining, by = "date"
```

```
ggplot(load, aes(x = date, y = daily_avg_load)) +
  geom_line() +
```

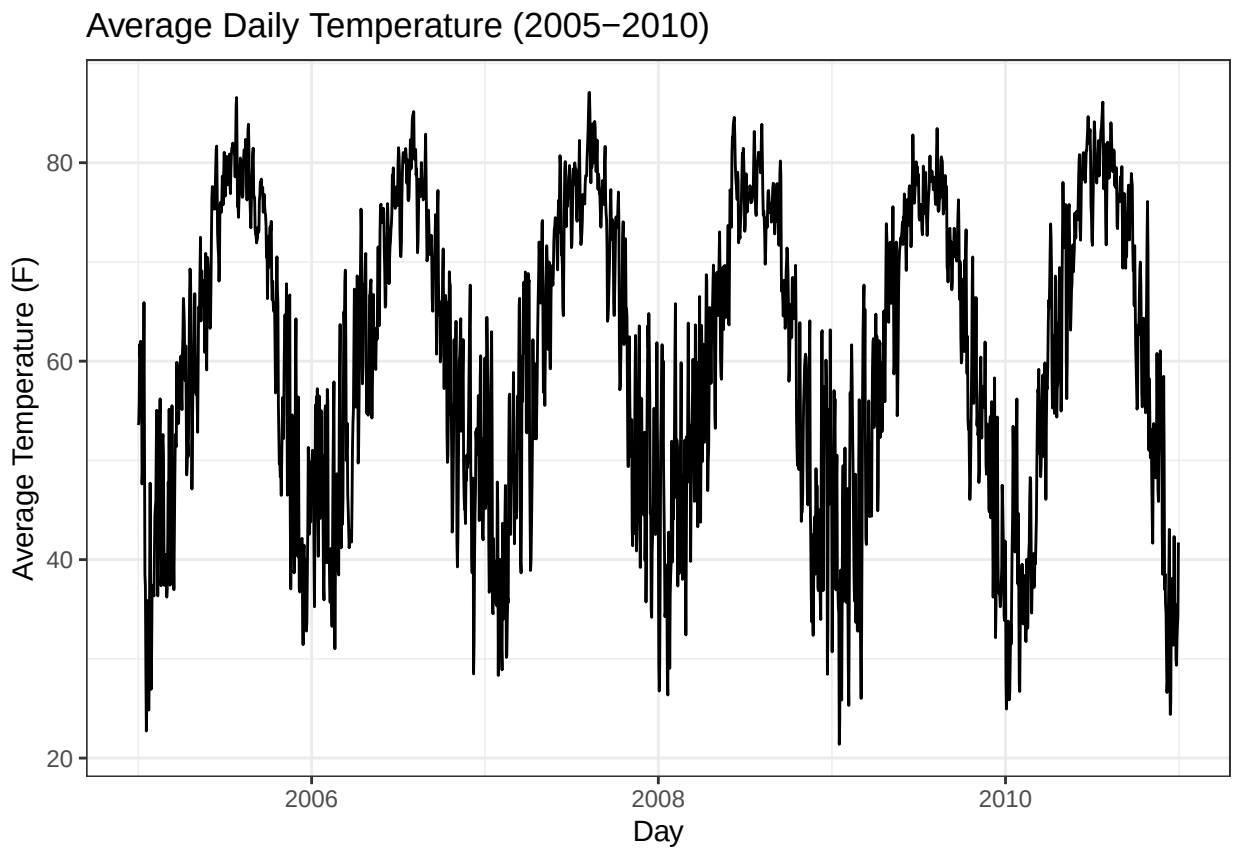
```
  labs(x = "Day", y = "Average Load",
       title = "Average Daily Load (2005-2010)") +
  theme_bw()
```




```
ggplot(relative_humidity, aes(x = date, y = daily_avg_humidity)) +
  geom_line() +
  labs(x = "Day", y = "Average Humidity",
       title = "Average Daily Humidity (2005-2010)") +
  theme_bw()
```



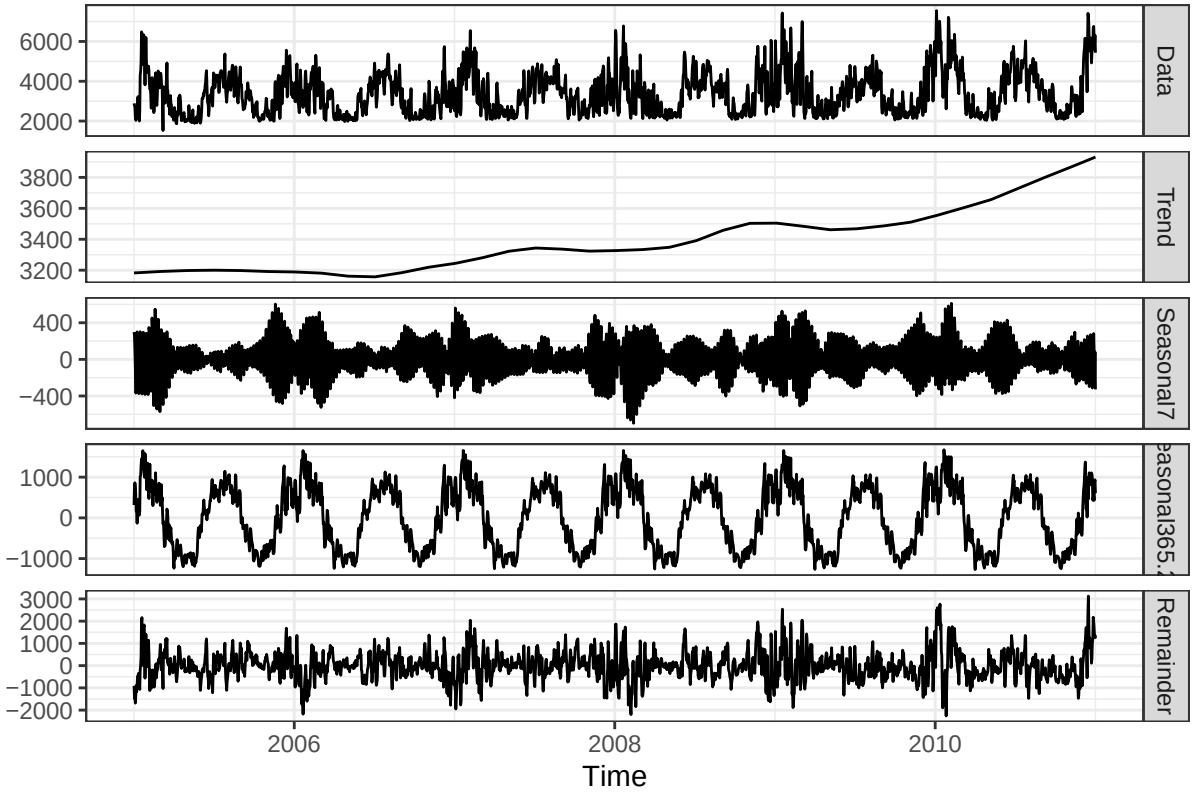

```
ggplot(temperature, aes(x = date, y = daily_avg_temp)) +
  geom_line() +
  labs(x = "Day", y = "Average Temperature (F)",
       title = "Average Daily Temperature (2005-2010)") +
  theme_bw()
```



After aggregating the data and plotting average values over time, we can see that average daily load does appear to exhibit seasonality, as well as average daily humidity and average daily temperature (although we note that temperature and humidity were not used in our model fitting process due to lack of temperature and humidity measurements for 2011).

Creating the Time-Series object

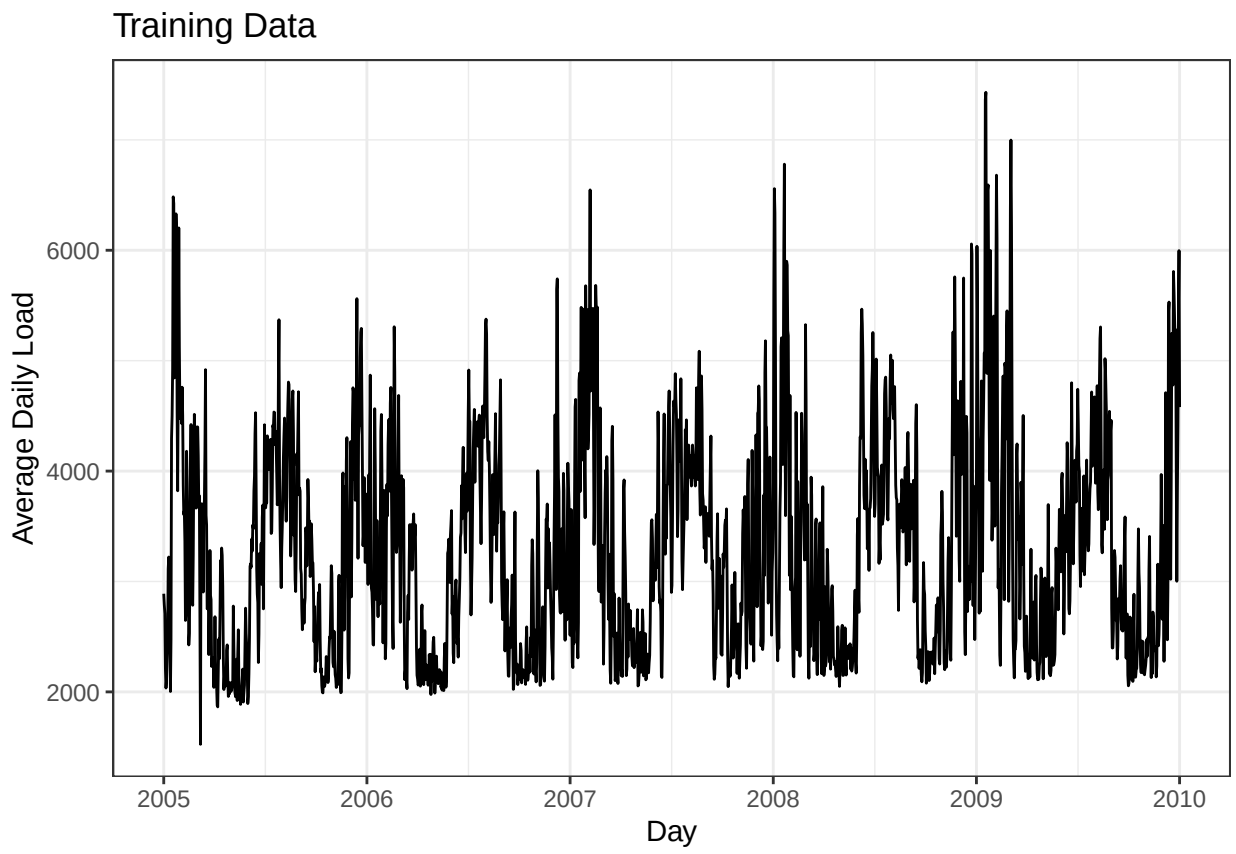
```
load_ts <- msts(df$daily_avg_load, seasonal.periods = c(7, 365.25), start = c(2005, 1, 1))

load_ts %>%
  mstl() %>%
  autoplot() +
  theme_bw()
```



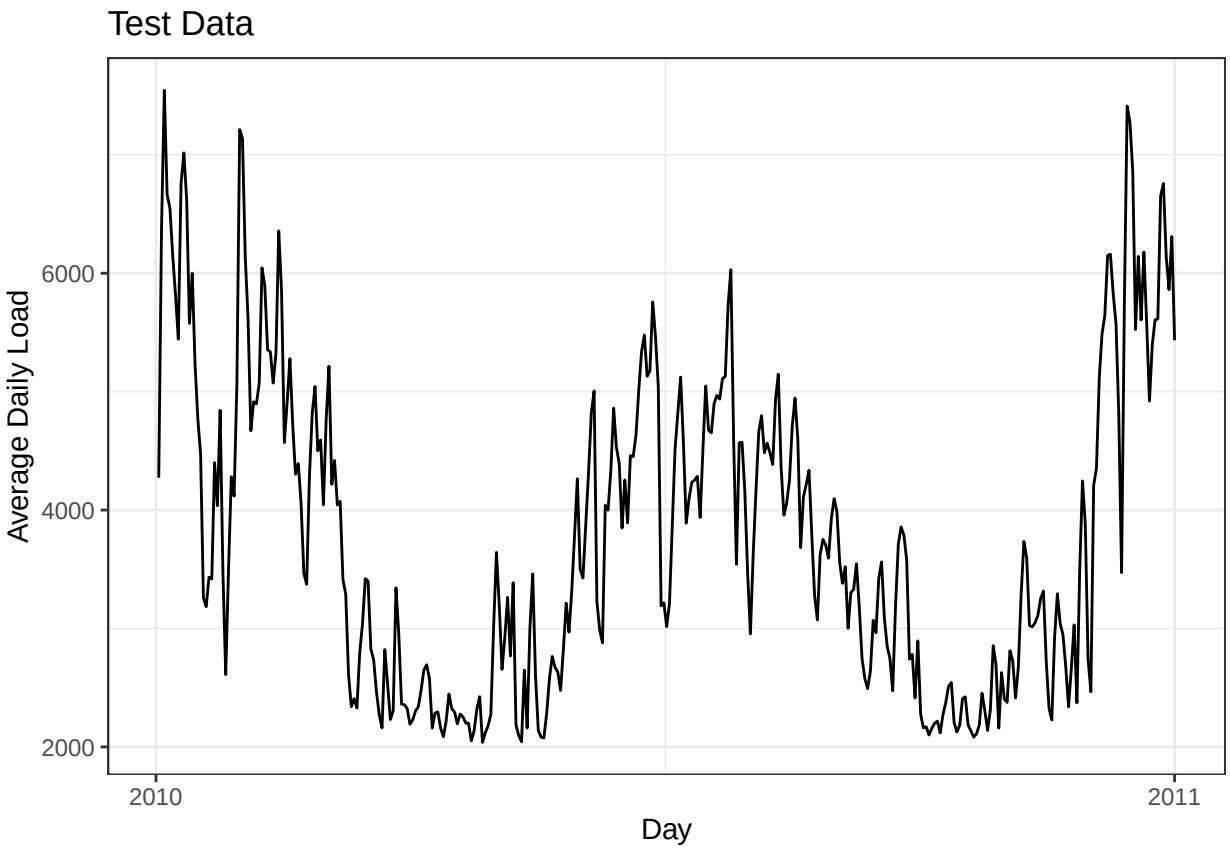
Again, after fitting a time series object to the data and decomposing, we see a strong yearly seasonal trend, as well as some evidence of a weekly trend. Our decomposition also shows that overall, there is an upward trend in daily average load from 2005-2010.

Fitting Models and Forecasting for 2010

```
n_for = 365

# training data
load_ts_train <- subset(load_ts, end = length(load_ts) - n_for)
# test data
load_ts_test <- subset(load_ts, start = length(load_ts) - n_for + 1)

autoplot(load_ts_train) +
  labs(x = "Day", y = "Average Daily Load", title = "Training Data") +
  theme_bw()
```



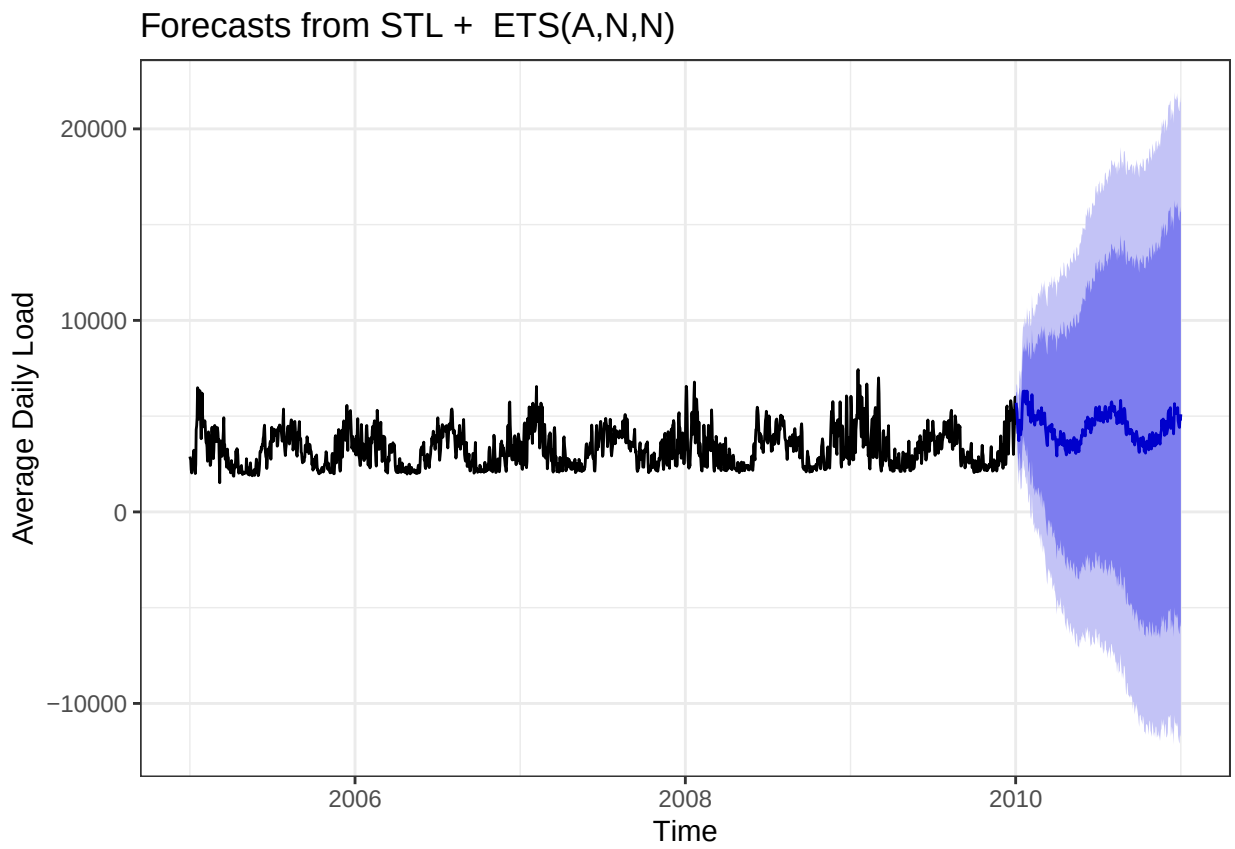

```
autoplot(load_ts_test) +
  labs(x = "Day", y = "Average Daily Load", title = "Test Data") +
  theme_bw()
```



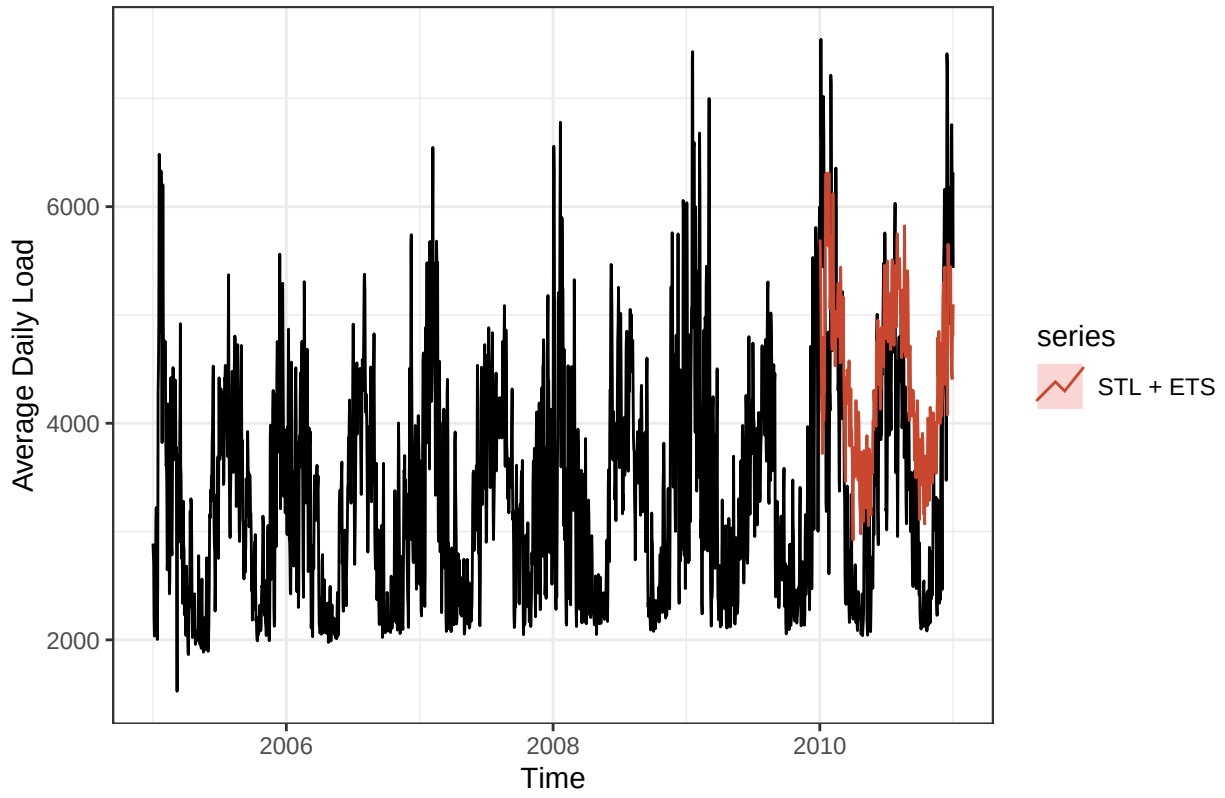
To aid in the model selection process, we'll split the data into a training and testing subset, where the data up until 2010 is used to train each model and the 2010 data is used for calculating accuracy metrics for each model.

Model 1: STL + ETS

```
ETS_fit <- stlf(load_ts_train, h=365)

# plot forecast results
autoplot(ETS_fit) +
  ylab("Average Daily Load") +
  theme_bw()
```

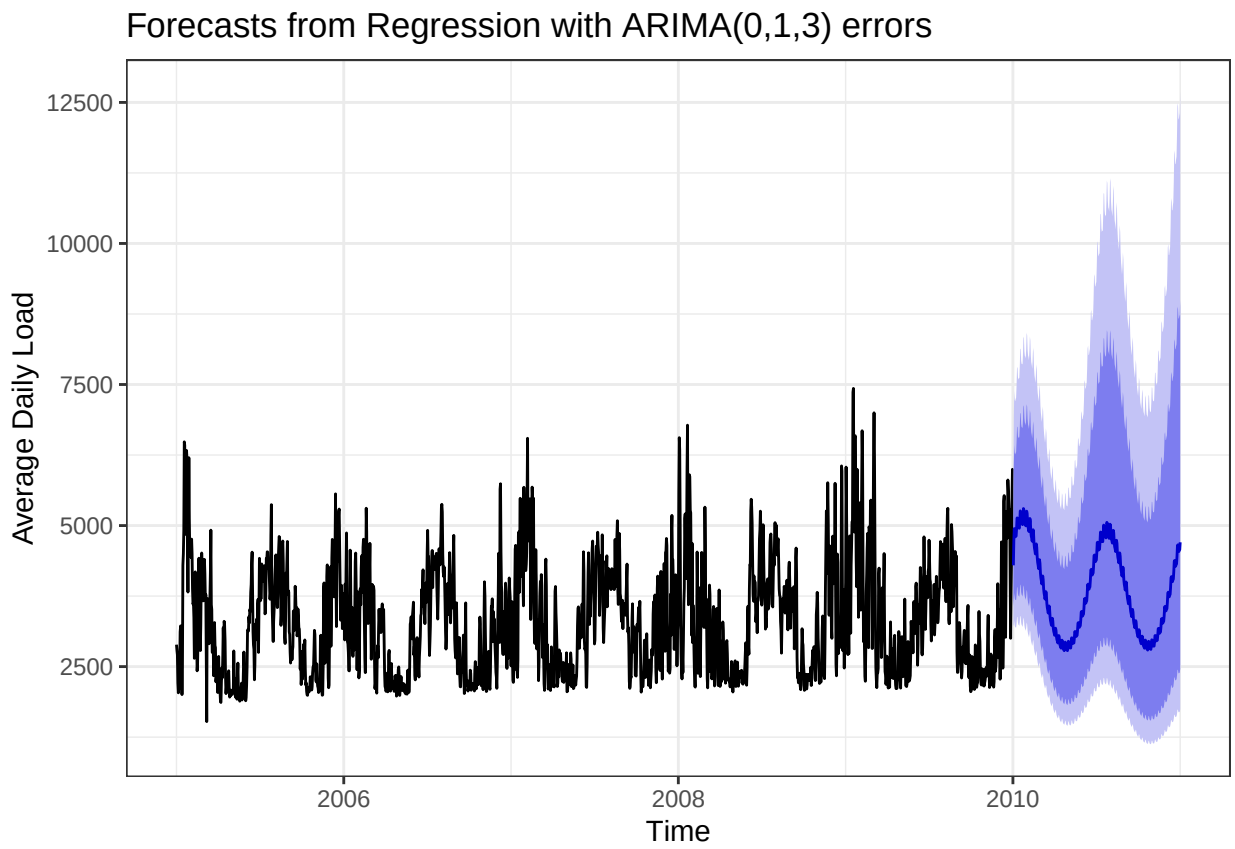



```
# plot model + observed data
autoplot(load_ts) +
  autolayer(ETS_fit, series = "STL + ETS", PI = FALSE) +
  ylab("Average Daily Load") +
  theme_bw()
```

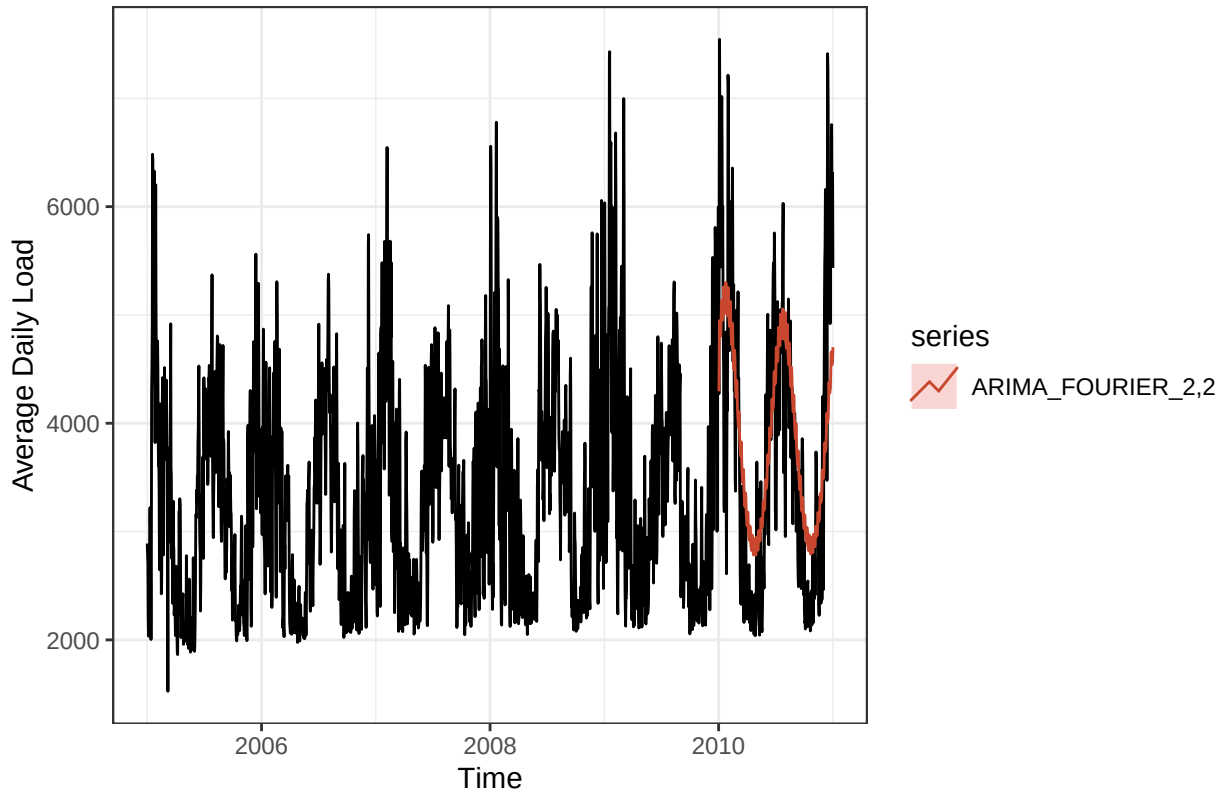


Model 2: ARIMA + Fourier Terms

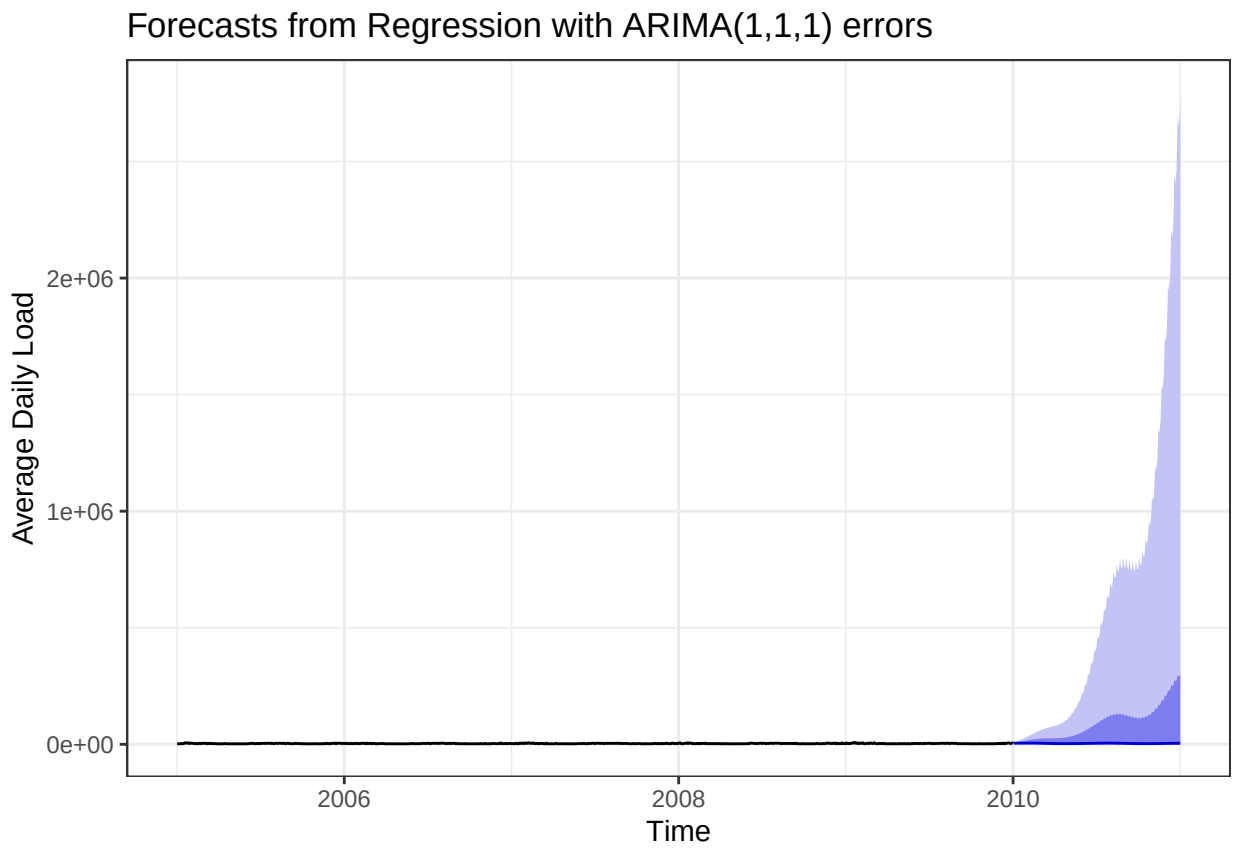
```
#Fit arima model with fourier terms as exogenous variables
#Starting with k=(2,2)
ARIMA_Four_fit_2 <- forecast::auto.arima(load_ts_train,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(load_ts_train,
                                          K=c(2,2))
                             )
#Forecast with ARIMA fit
ARIMA_Four_for_2 <- forecast::forecast(ARIMA_Four_fit_2,
                           xreg=fourier(load_ts_train,
                                        K=c(2,2),
                                        h=365),
                           h=365
                           )
#Plot foresting results
autoplot(ARIMA_Four_for_2) +
  ylab("Average Daily Load") +
  theme_bw()
```



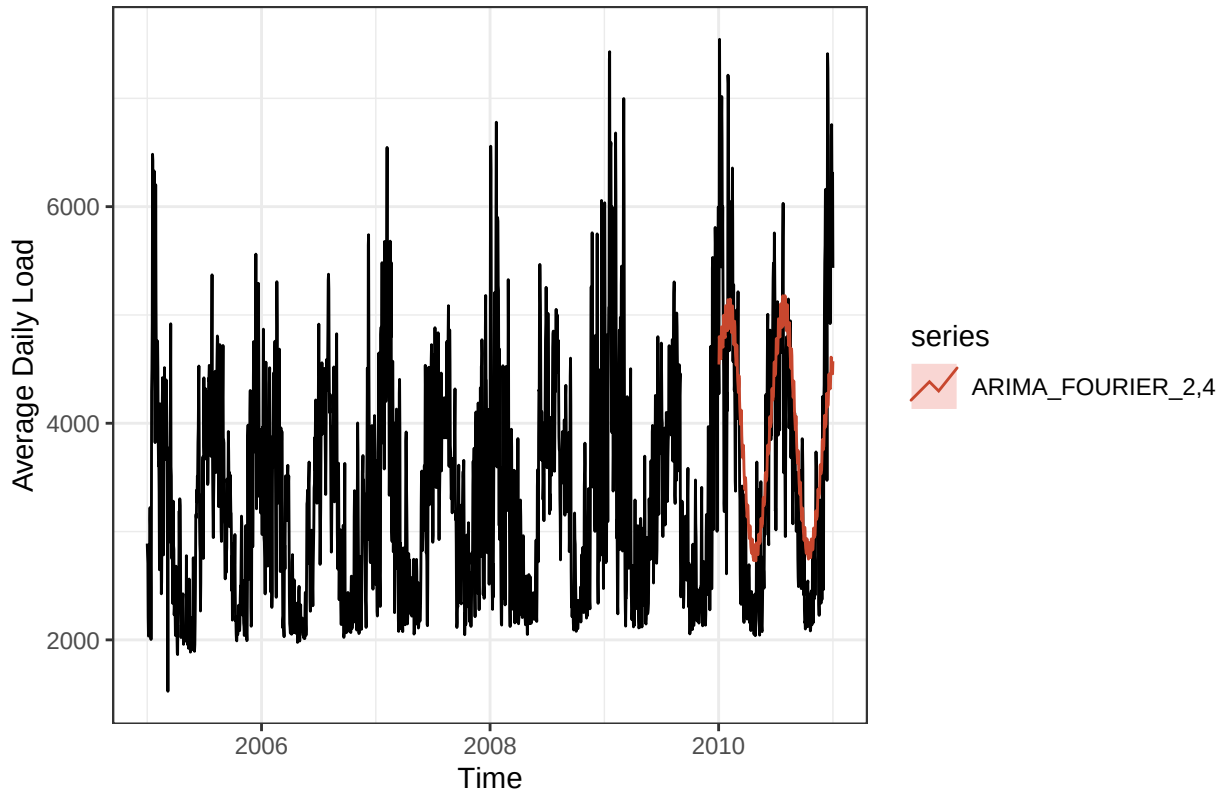

```
#Plot model + observed data
autoplot(load_ts) +
  autolayer(ARIMA_Four_for_2, series="ARIMA_FOURIER_2,2",PI=FALSE) +
  ylab("Average Daily Load") +
  theme_bw()
```



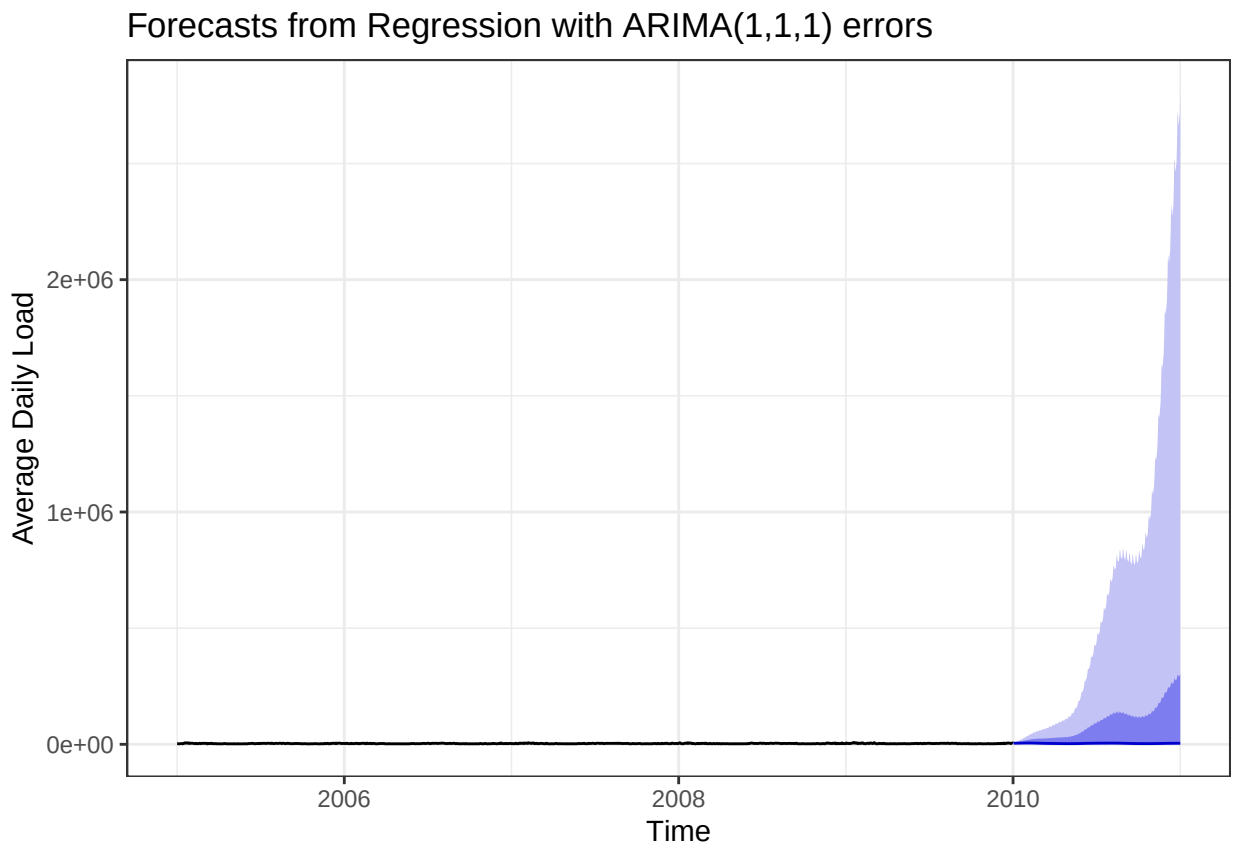

```
#Arima+Fourier(2,4)
ARIMA_Four_fit_4 <- forecast::auto.arima(load_ts_train,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(load_ts_train,
                                          K=c(2,4))
                             )
#Forecast with ARIMA fit
ARIMA_Four_for_4 <- forecast::forecast(ARIMA_Four_fit_4,
                            xreg=fourier(load_ts_train,
                                         K=c(2,4),
                                         h=365),
                            h=365
                            )
#Plot foresting results
autoplot(ARIMA_Four_for_4) +
  ylab("Average Daily Load") +
  theme_bw()
```



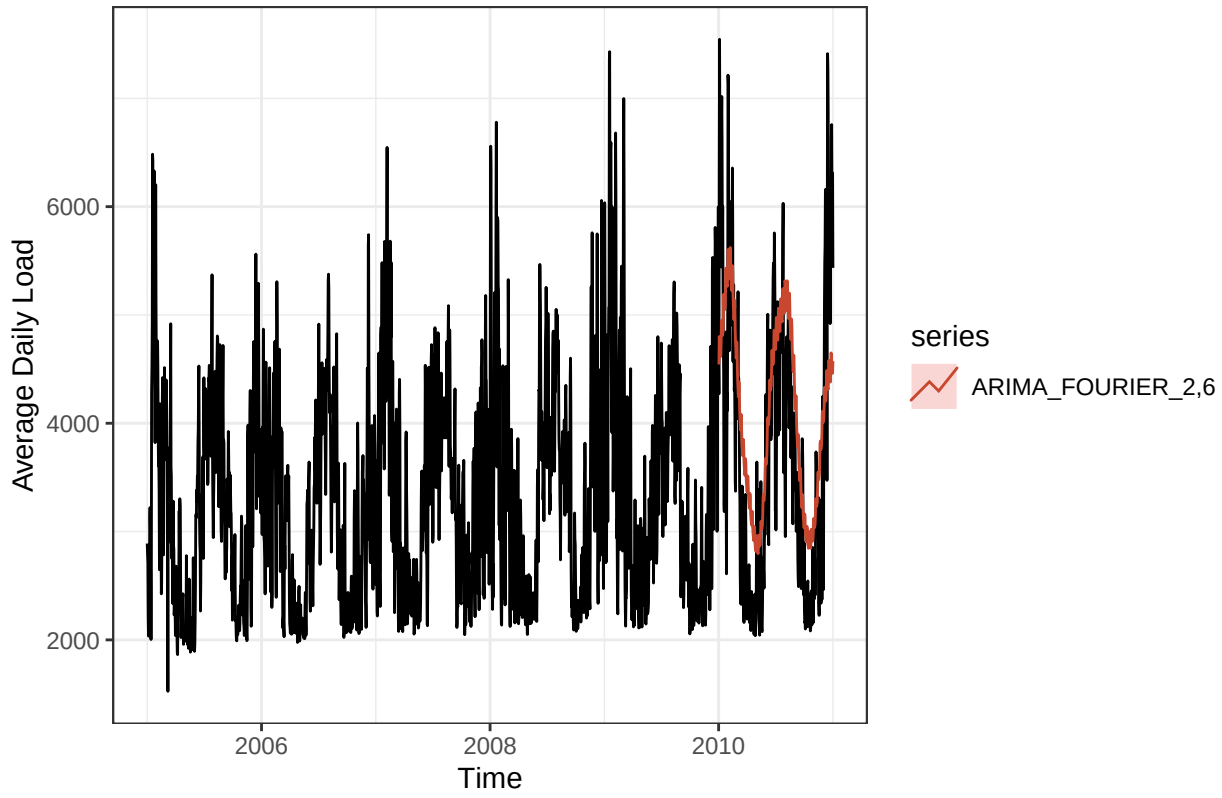

```
#Plot model + observed data
autoplot(load_ts) +
  autolayer(ARIMA_Four_for_4, series="ARIMA_FOURIER_2,4",PI=FALSE) +
  ylab("Average Daily Load") +
  theme_bw()
```



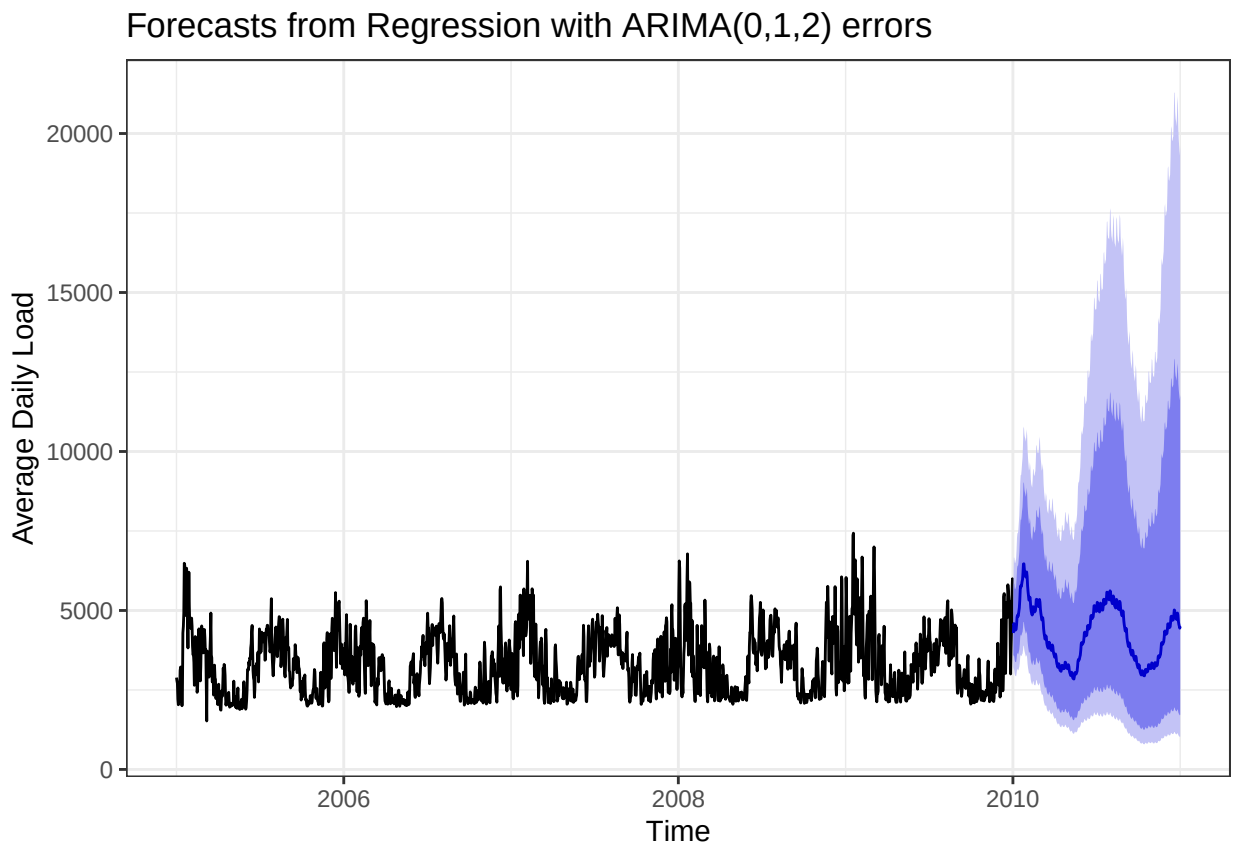

```
#Arima+Fourier(2,6)
ARIMA_Four_fit_6 <- forecast::auto.arima(load_ts_train,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(load_ts_train,
                                          K=c(2,6))
                             )
#Forecast with ARIMA fit
ARIMA_Four_for_6 <- forecast::forecast(ARIMA_Four_fit_6,
                            xreg=fourier(load_ts_train,
                                         K=c(2,6),
                                         h=365),
                            h=365
                            )
#Plot foresting results
autoplot(ARIMA_Four_for_6) +
  ylab("Average Daily Load") +
  theme_bw()
```



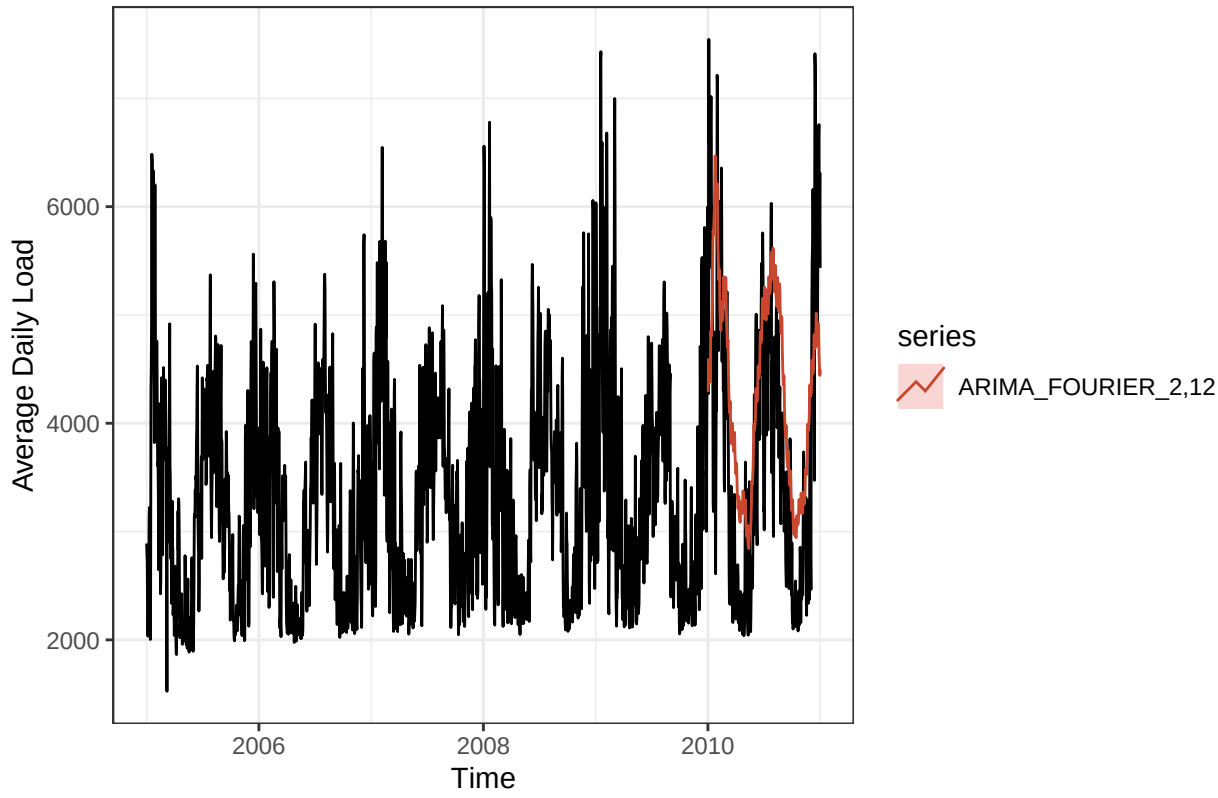

```
#Plot model + observed data
autoplot(load_ts) +
  autolayer(ARIMA_Four_for_6, series="ARIMA_FOURIER_2,6",PI=FALSE) +
  ylab("Average Daily Load") +
  theme_bw()
```




```
#Arima+Fourier(2,12)
ARIMA_Four_fit_12 <- forecast::auto.arima(load_ts_train,
                             seasonal=FALSE,
                             lambda=0,
                             xreg=fourier(load_ts_train,
                                          K=c(2,12))
                             )
#Forecast with ARIMA fit
ARIMA_Four_for_12 <- forecast::forecast(ARIMA_Four_fit_12,
                            xreg=fourier(load_ts_train,
                                         K=c(2,12),
                                         h=365),
                            h=365
                            )
#Plot foresting results
autoplot(ARIMA_Four_for_12) +
  ylab("Average Daily Load") +
  theme_bw()
```

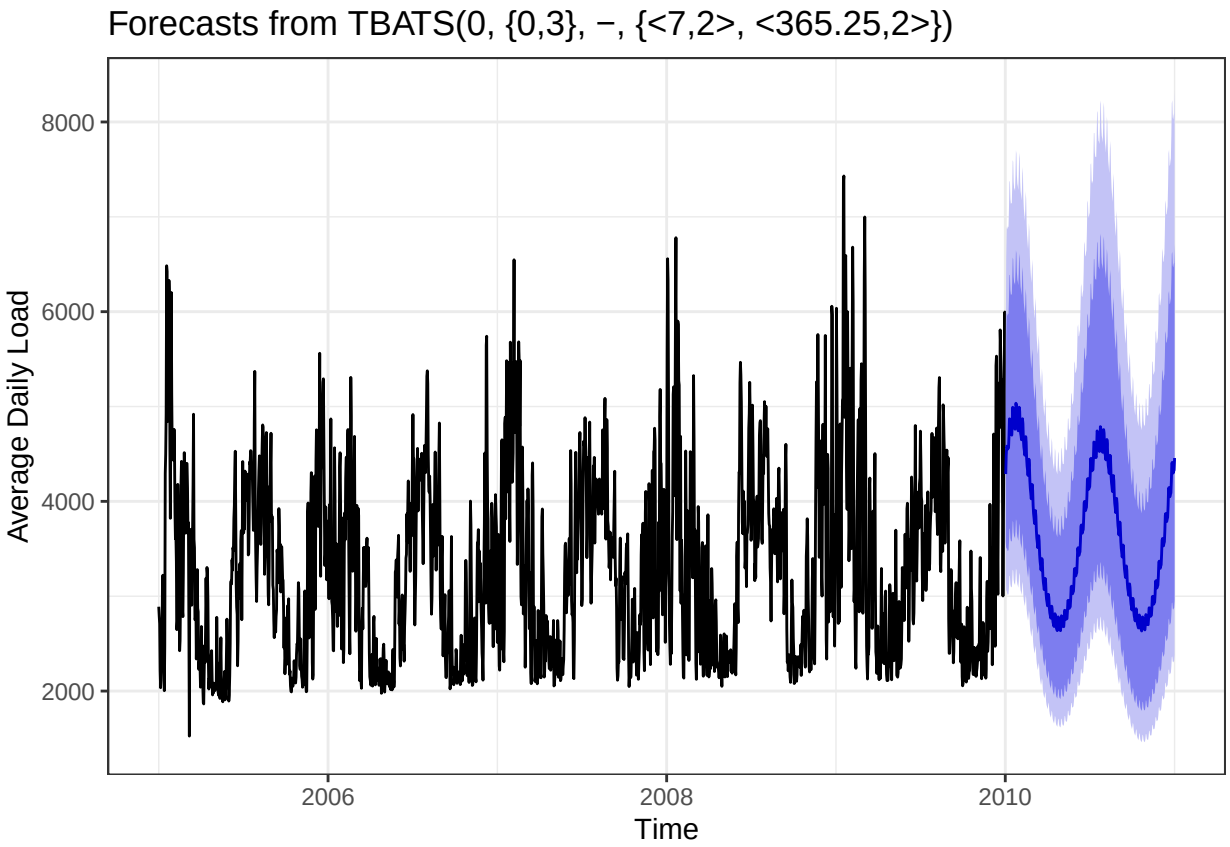



```
#Plot model + observed data
autoplot(load_ts) +
  autolayer(ARIMA_Four_for_12, series="ARIMA_FOURIER_2,12",PI=FALSE) +
  ylab("Average Daily Load") +
  theme_bw()
```

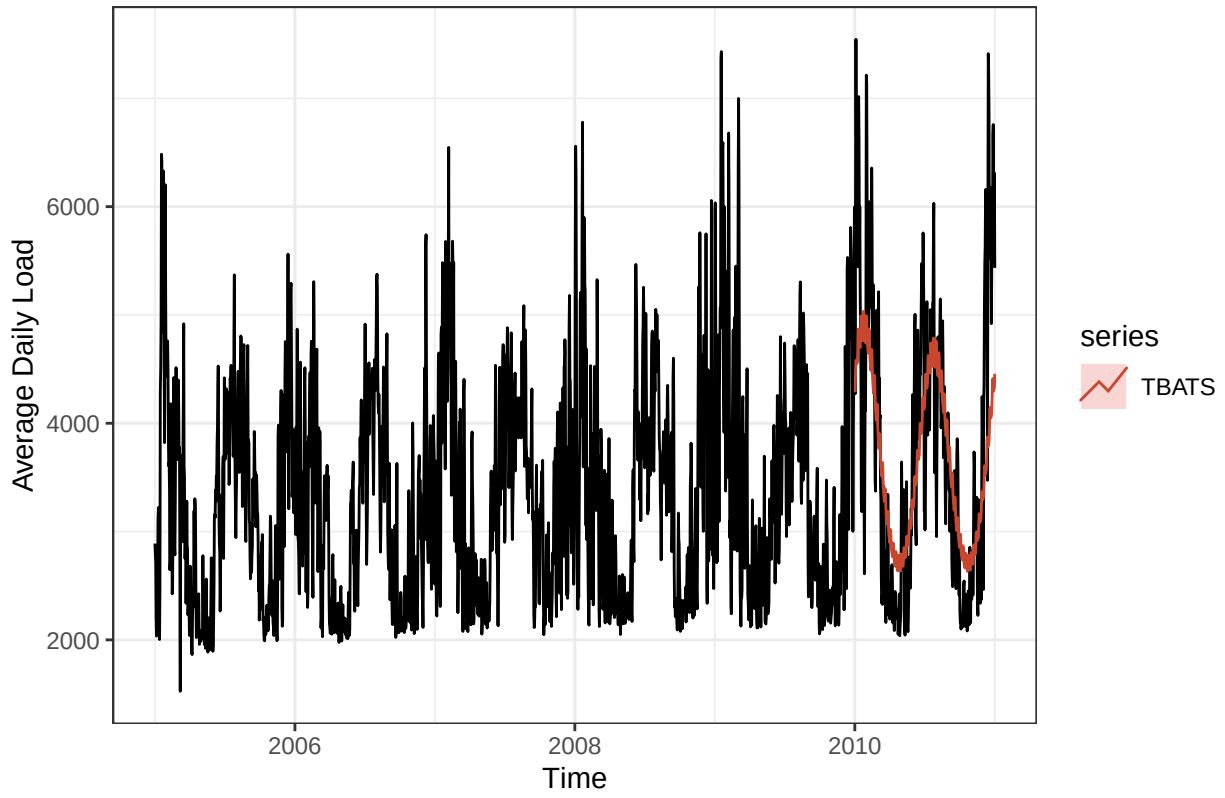


Model 3: TBATS

```
TBATS_fit <- tbats(load_ts_train)

TBATS_for <- forecast(TBATS_fit, h=365)

#Plot forecasting results
autoplot(TBATS_for) +
  ylab("Average Daily Load") +
  theme_bw()
```

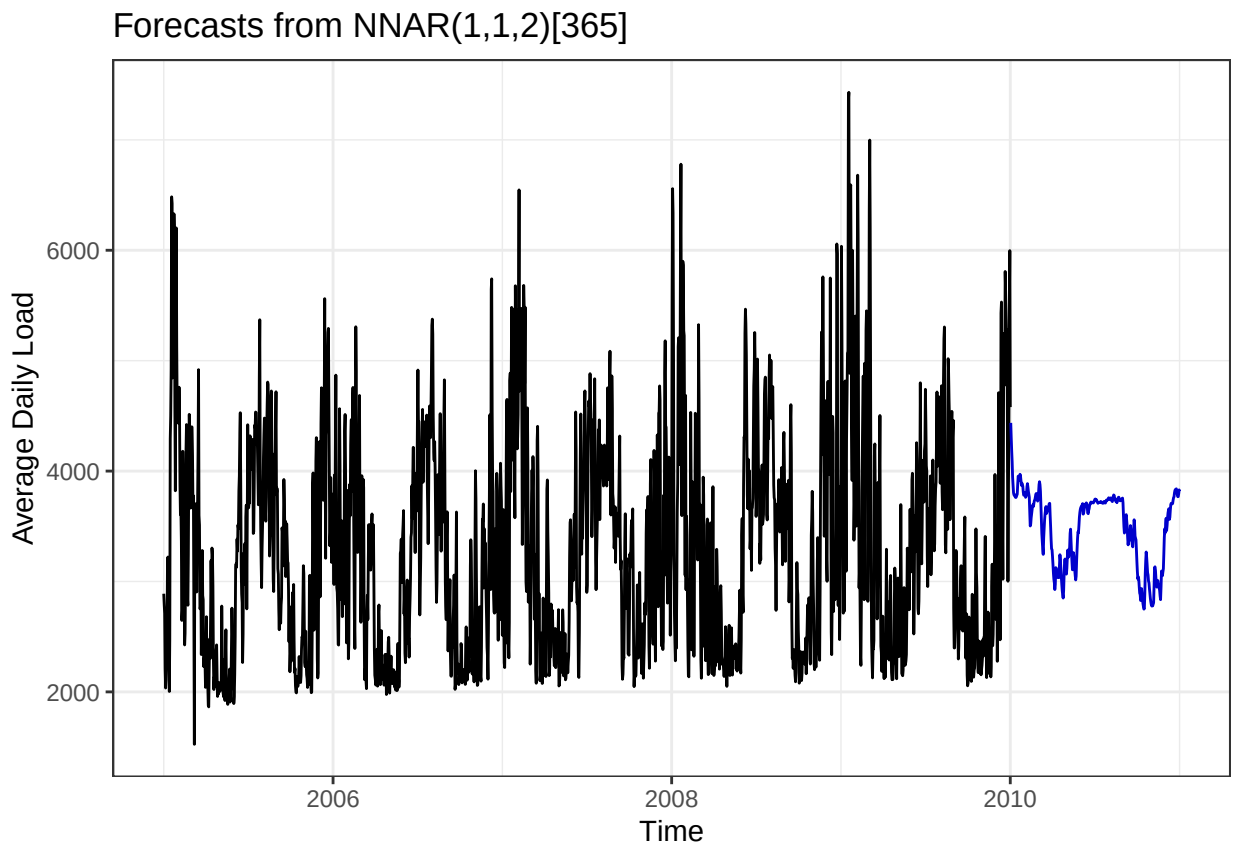



```
#Plot model + observed data
autoplot(load_ts) +
  autolayer(TBATS_for, series="TBATS",PI=FALSE)+
  ylab("Average Daily Load") +
  theme_bw()
```

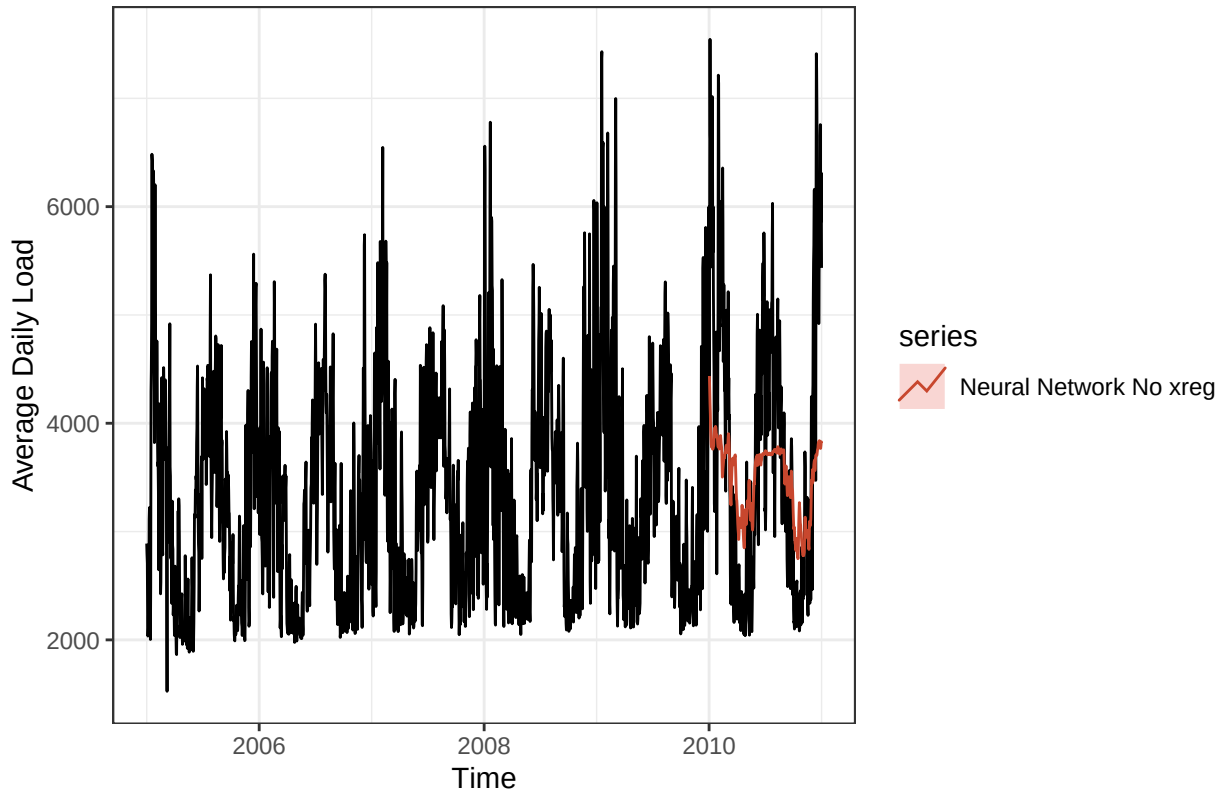


Model 4: Neural Network

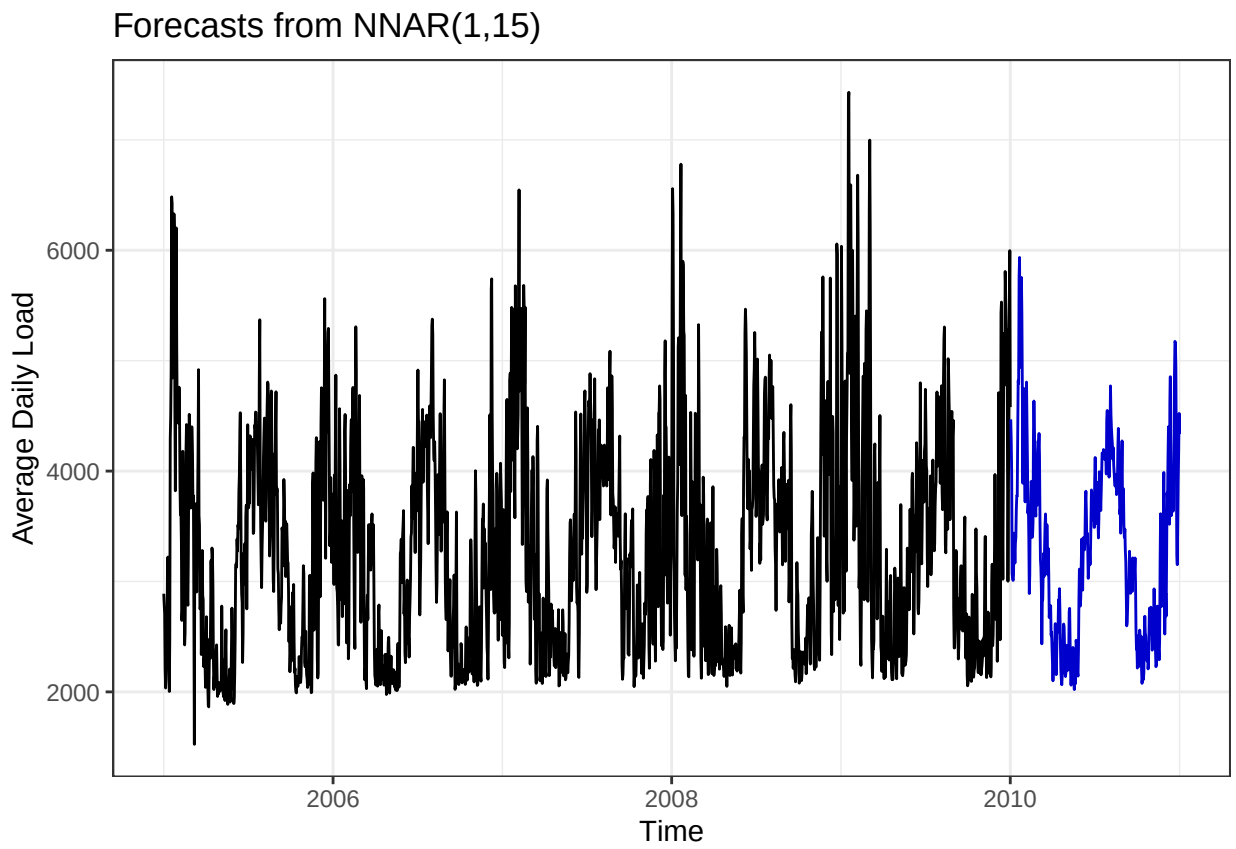
```
#p=1 P=1 no xreg
NN_fit_1 <- nnetar(load_ts_train,p=1,P=1)
NN_for_1 <- forecast(NN_fit_1, h=365)

#Plot forecasting results
autoplot(NN_for_1) +
  ylab("Average Daily Load") +
  theme_bw()
```



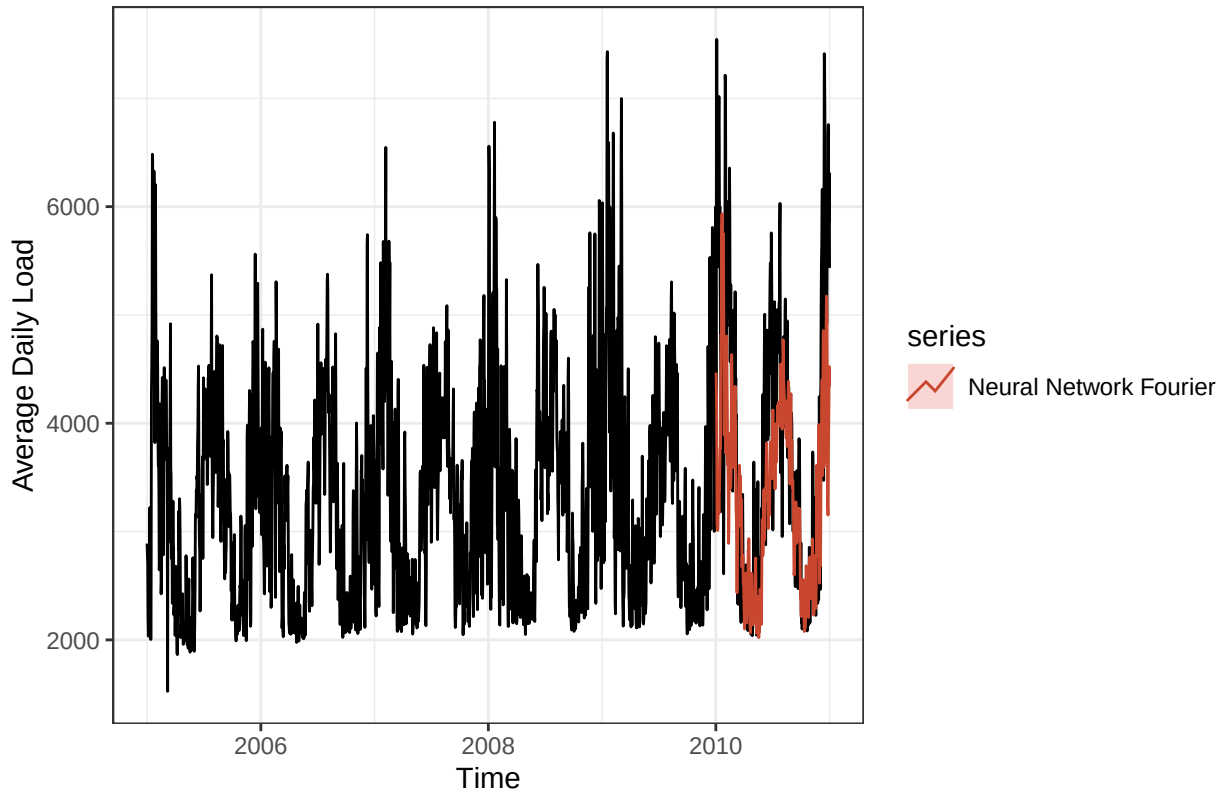

```
#Plot model + observed data
autoplot(load_ts) +
  autolayer(NN_for_1, series="Neural Network No xreg",PI=FALSE)+
  ylab("Average Daily Load") +
  theme_bw()
```




```
#p=1 P=0 xreg fourier
NN_fit_2 <- nnetar(load_ts_train,p=1,P=0,xreg=fourier(load_ts_train, K=c(2,12)))
NN_for_2 <- forecast(NN_fit_2, h=365,xreg=fourier(load_ts_train,
                                        K=c(2,12),h=365))
#Plot foresting results
autoplot(NN_for_2) +
  ylab("Average Daily Load") +
  theme_bw()
```

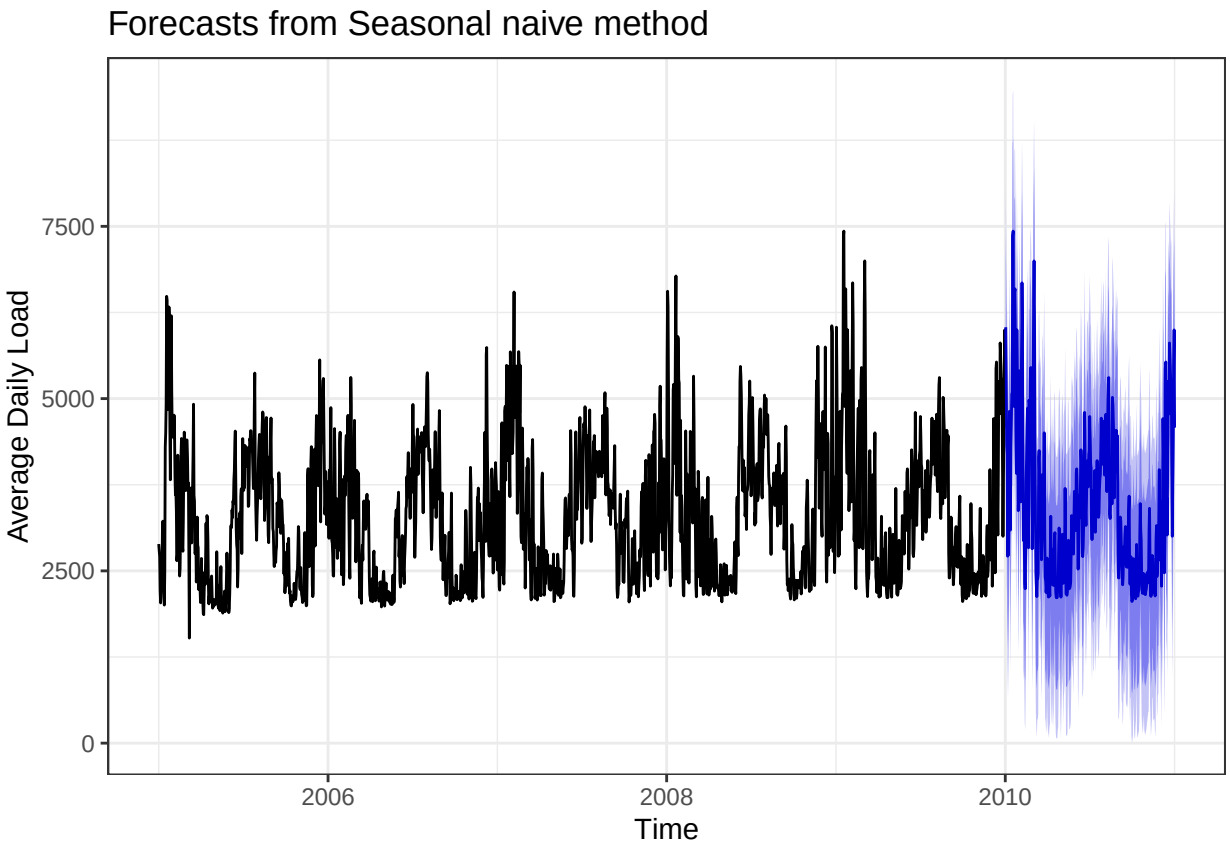



```
#Plot model + observed data
autoplot(load_ts) +
  autolayer(NN_for_2, series="Neural Network Fourier",PI=FALSE)+
  ylab("Average Daily Load") +
  theme_bw()
```

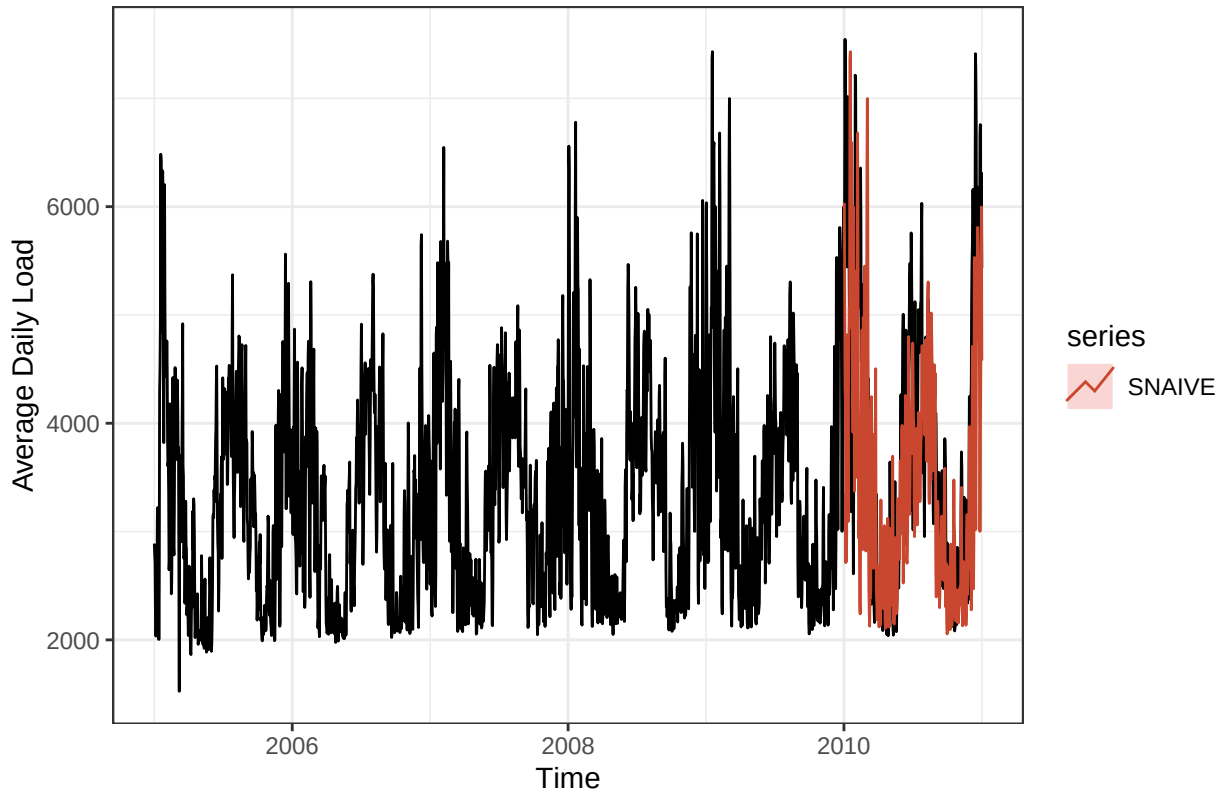


Model 5: Seasonal Naive

```
SNAIVE_seas <- snaive(load_ts_train, h=365)

autoplot(SNAIVE_seas) +
  ylab("Average Daily Load") +
  theme_bw()
```

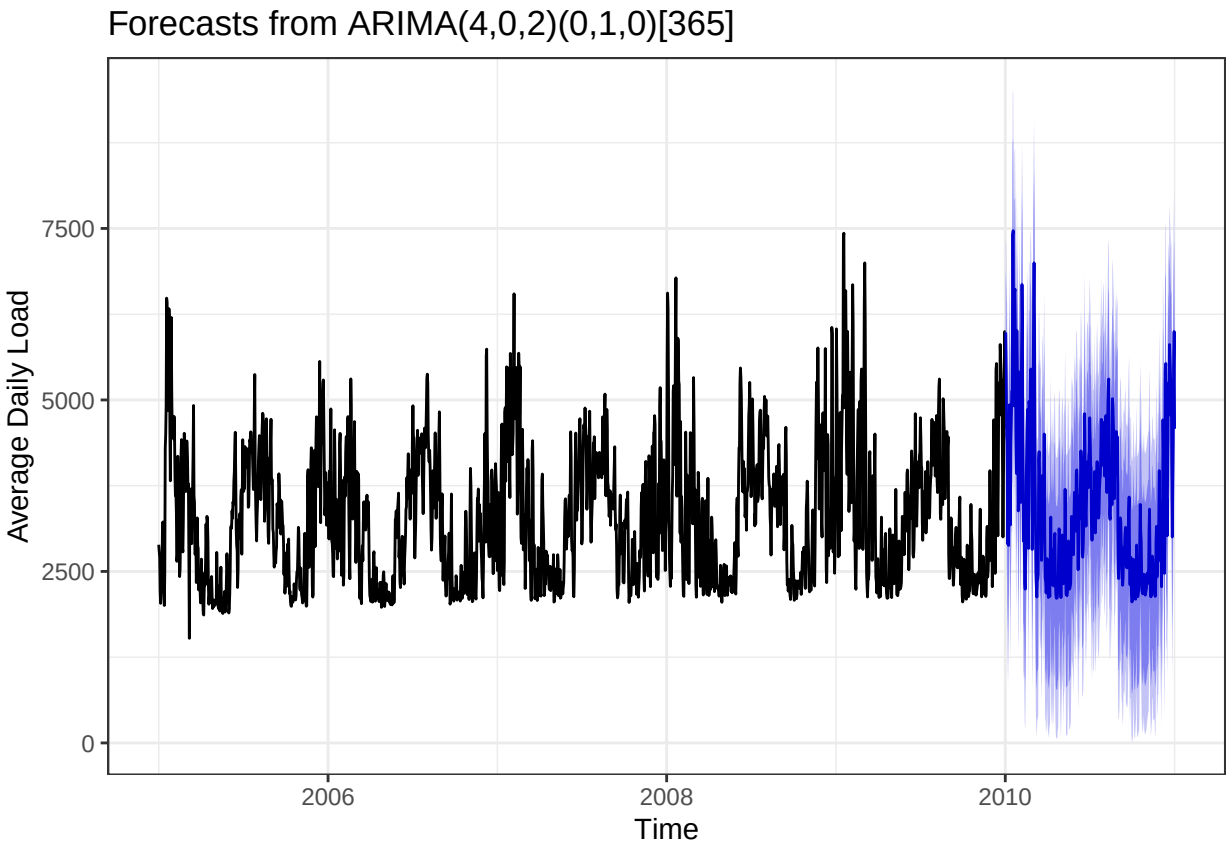



```
#Plot model + observed data
autoplot(load_ts) +
  autolayer(SNAIVE_seas, series="SNAIVE",PI=FALSE)+
  ylab("Average Daily Load") +
  theme_bw()
```

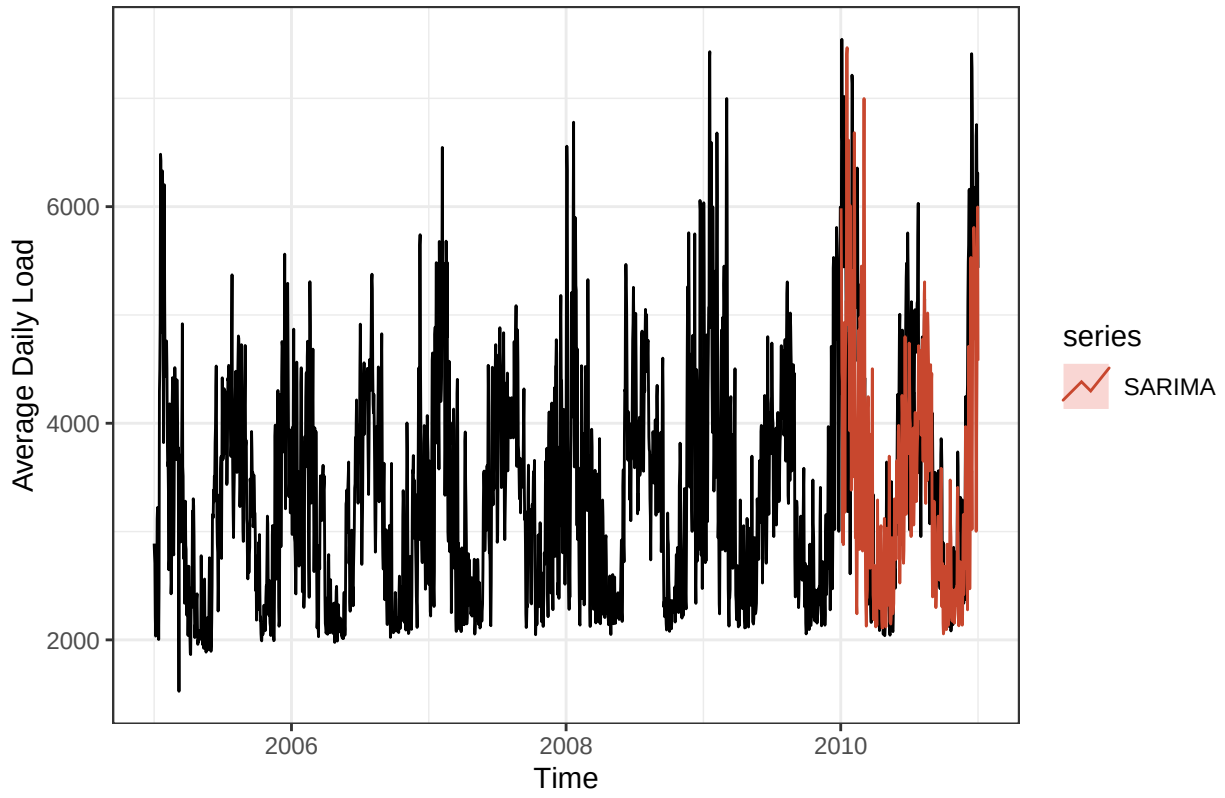


Model 6: SARIMA

```
# SARIMA_autofit <- auto.arima(load_ts_train, D=1)
# we'll save the sarima model to an rds object since it takes a long time to fit
# saveRDS(SARIMA_autofit, "sarima_mod.rds")
SARIMA_autofit <- readRDS("sarima_mod.rds")
SARIMA_for <- forecast(SARIMA_autofit, h=365)

autoplot(SARIMA_for) +
  ylab("Average Daily Load") +
  theme_bw()
```




```
#Plot model + observed data
autoplot(load_ts) +
  autolayer(SARIMA_for, series="SARIMA",PI=FALSE)+
  ylab("Average Daily Load") +
  theme_bw()
```



Comparing Models

```
# model 1: STL + ETS
ETS_scores <- accuracy(ETS_fit$mean, load_ts_test)

#model 2: Arima + Fourier
ARIMA_2_scores <- accuracy(ARIMA_Four_for_2$mean, load_ts_test)
ARIMA_4_scores <- accuracy(ARIMA_Four_for_4$mean, load_ts_test)
ARIMA_6_scores <- accuracy(ARIMA_Four_for_6$mean, load_ts_test)
ARIMA_12_scores <- accuracy(ARIMA_Four_for_12$mean, load_ts_test)

#model 3: TBATS
TBATS_scores <- accuracy(TBATS_for$mean,load_ts_test)

#model 4: NN
NN_1_scores <- accuracy(NN_for_1$mean,load_ts_test)
NN_2_scores <- accuracy(NN_for_2$mean,load_ts_test)

#model 4: seasonal naive
SNAIVE_scores <- accuracy(SNAIVE_seas$mean, load_ts_test)

#model 5: SARIMA
SARIMA_scores <- accuracy(SARIMA_for$mean, load_ts_test)
```

Table 1: Forecast Accuracy for Daily Average Load

	ME	RMSE	MAE	MPE	MAPE	ACF1	Theil's U
STL+ETS	-664.68803	1230.3995	1039.9902	-27.15911	33.22800	0.79970	2.80673
ARIMA+Fourier k(2,2)	-85.57736	896.7440	708.0656	-8.98593	20.24452	0.82508	1.70279
ARIMA+Fourier k(2,4)	-149.13266	909.8270	725.7151	-10.70178	20.93255	0.82421	1.75230
ARIMA+Fourier k(2,6)	-309.57374	951.6593	781.5702	-15.22924	23.40538	0.82465	1.95131
ARIMA+Fourier k(2,12)	-506.16490	1080.7349	898.1974	-20.91309	27.48365	0.84141	2.29946
TBATS	118.69171	912.2884	688.5789	-3.18910	18.35719	0.82733	1.54891
NN no xreg	281.80017	1138.6542	885.2196	-1.54247	23.26645	0.87801	1.91824
NN Fourier	459.59772	1123.3750	776.8677	6.87561	18.17407	0.82308	1.66116
SNAIVE	322.70434	1225.5660	879.9431	3.38456	21.77434	0.73106	1.96736
SARIMA	317.57978	1218.1438	876.2721	3.29468	21.72382	0.72943	1.96394

Comparing accuracy metrics across all of our models for a January 2010 forecast, we can see from the table that the ARIMA+Fourier k(2,2) model performed best when it came to RMSE, and the NN Fourier model performed best when it came to MAPE.

Predicting January 2011

```
submission_template <- read_excel("submission_template.xlsx",
                                  sheet = "p90", n_max = 31)
```

```
# fitting models on all the data
n_for <- 31

#ETS
ETS_fit <- stlf(load_ts, h=n_for)

#ARIMA + Fourier
ARIMA_Four_fit_2 <- forecast::auto.arima(load_ts,
                            seasonal=FALSE,
                            lambda=0,
                            xreg=fourier(load_ts,
                                         K=c(2,2))
                            )
ARIMA_Four_for_2 <- forecast::forecast(ARIMA_Four_fit_2,
                           xreg=fourier(load_ts,
                                        K=c(2,2),
                                        h=n_for),
                           h=n_for
                           )
ARIMA_Four_fit_4 <- forecast::auto.arima(load_ts,
                            seasonal=FALSE,
                            lambda=0,
                            xreg=fourier(load_ts,
                                         K=c(2,4))
                            )
ARIMA_Four_for_4 <- forecast::forecast(ARIMA_Four_fit_4,
                           xreg=fourier(load_ts,
```

```
                                        K=c(2,4),
                                        h=n_for),
                            h=n_for
                            )
ARIMA_Four_fit_6 <- forecast::auto.arima(load_ts,
                              seasonal=FALSE,
                              lambda=0,
                              xreg=fourier(load_ts,
                                           K=c(2,6))
                              )
ARIMA_Four_for_6 <- forecast::forecast(ARIMA_Four_fit_6,
                            xreg=fourier(load_ts,
                                        K=c(2,6),
                                        h=n_for),
                            h=n_for
                            )
ARIMA_Four_fit_12 <- forecast::auto.arima(load_ts,
                              seasonal=FALSE,
                              lambda=0,
                              xreg=fourier(load_ts,
                                           K=c(2,12))
                              )
ARIMA_Four_for_12 <- forecast::forecast(ARIMA_Four_fit_12,
                            xreg=fourier(load_ts,
                                        K=c(2,12),
                                        h=n_for),
                            h=n_for
                            )
#TBATS
TBATS_fit <- tbats(load_ts)
TBATS_for <- forecast(TBATS_fit, h=n_for)

#NN
NN_fit_1 <- nnetar(load_ts,p=1,P=1)
NN_for_1 <- forecast(NN_fit_1, h=n_for)

NN_fit_2 <- nnetar(load_ts,p=1,P=0,xreg=fourier(load_ts, K=c(2,12)))
NN_for_2 <- forecast(NN_fit_2, h=n_for,xreg=fourier(load_ts,
                                        K=c(2,12),h=n_for))

#SNAIVE
SNAIVE_fit <- snaive(load_ts, h=n_for)

#SARIMA
SARIMA_fit <- Arima(load_ts, order=c(4,0,2), seasonal=c(0,1,0), include.drift = TRUE)
SARIMA_for <- forecast(SARIMA_fit, h=n_for)
```

```
# build kaggle submission file
submission_template$load <- ETS_fit$mean
submission_template$load <- ARIMA_Four_for_2$mean
submission_template$load <- ARIMA_Four_for_4$mean
submission_template$load <- ARIMA_Four_for_6$mean
submission_template$load <- NN_for_1$mean
```

```
submission_template$load <- NN_for_2$mean
submission_template$load <- SNAIVE_fit$mean
submission_template$load <- SARIMA_for$mean

# change the path name for each submission file
write.csv(submission_template, "kaggle_submissions/sub4.csv", row.names = FALSE)
write.csv(submission_template, "kaggle_submissions/sub5.csv", row.names = FALSE)
write.csv(submission_template, "kaggle_submissions/sub6.csv", row.names = FALSE)
write.csv(submission_template, "kaggle_submissions/sub7.csv", row.names = FALSE)
write.csv(submission_template, "kaggle_submissions/sub8.csv", row.names = FALSE)
write.csv(submission_template, "kaggle_submissions/sub9.csv", row.names = FALSE)
write.csv(submission_template, "kaggle_submissions/sub10.csv", row.names = FALSE)
write.csv(submission_template, "kaggle_submissions/sub11.csv", row.names = FALSE)
```

We submitted every model we fit to the Kaggle competition, with the SARIMA model performing the best on the public leaderboard. However, we do note that the Kaggle results were based on only 3 days out of the 31 in January 2011, and as a result it is likely that overall there was a model that performed better across the entire month.