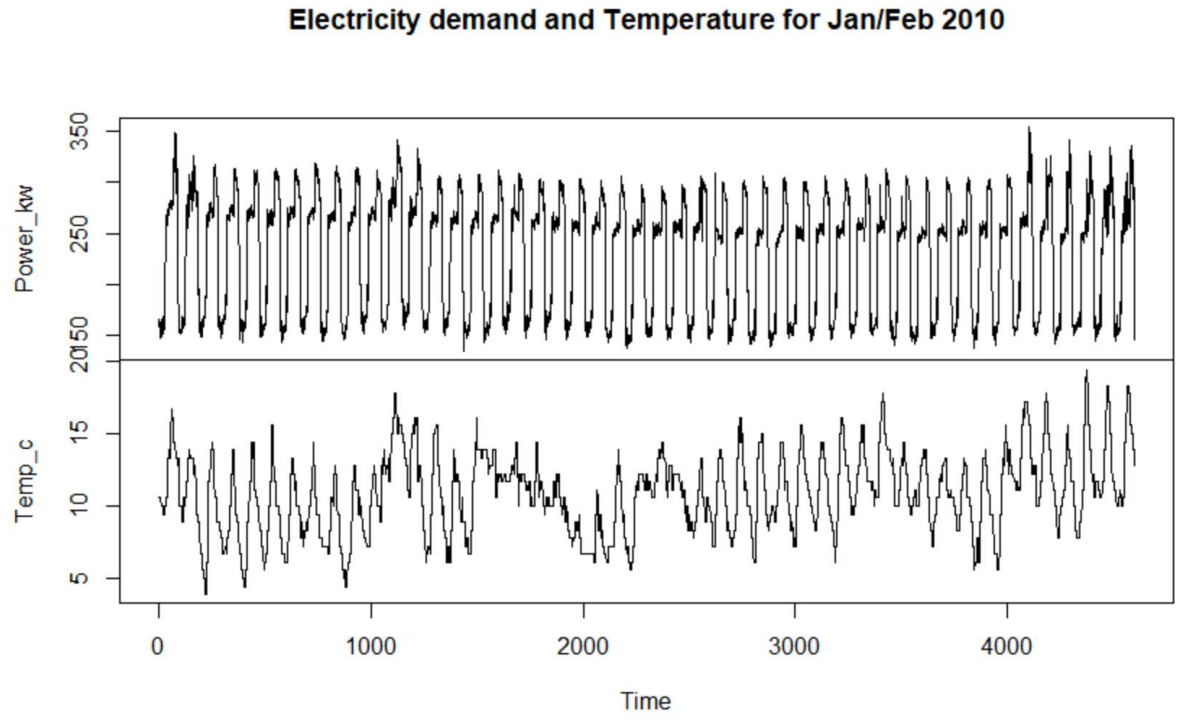
Time Series Assignment

The file Elec-train.xlsx contains electricity consumption (kW) and outdoor air temperature for one building. These quantities are measured every 15 minutes, from 1/1/2010 1:15 to 2/16/2010 23:45. In addition, outdoor air temperature are available for 2/17/2010.

The goal is to forecast electricity consumption (kW) for 2/17/2010. Two forecasts should be returned, in one Excel file entitled YourName.xlsx, with two columns (one columns per forecast) and 96 rows: 1. the first one without using outdoor temperature, 2. the second one using outdoor temperature.

The plotted data shows a daily seasonality for the electricity consumption. There are 96 observations per day (except for the first day which only appears to have 91). There is no obvious trend in the power usage although a small increase can be observed in the last few days.



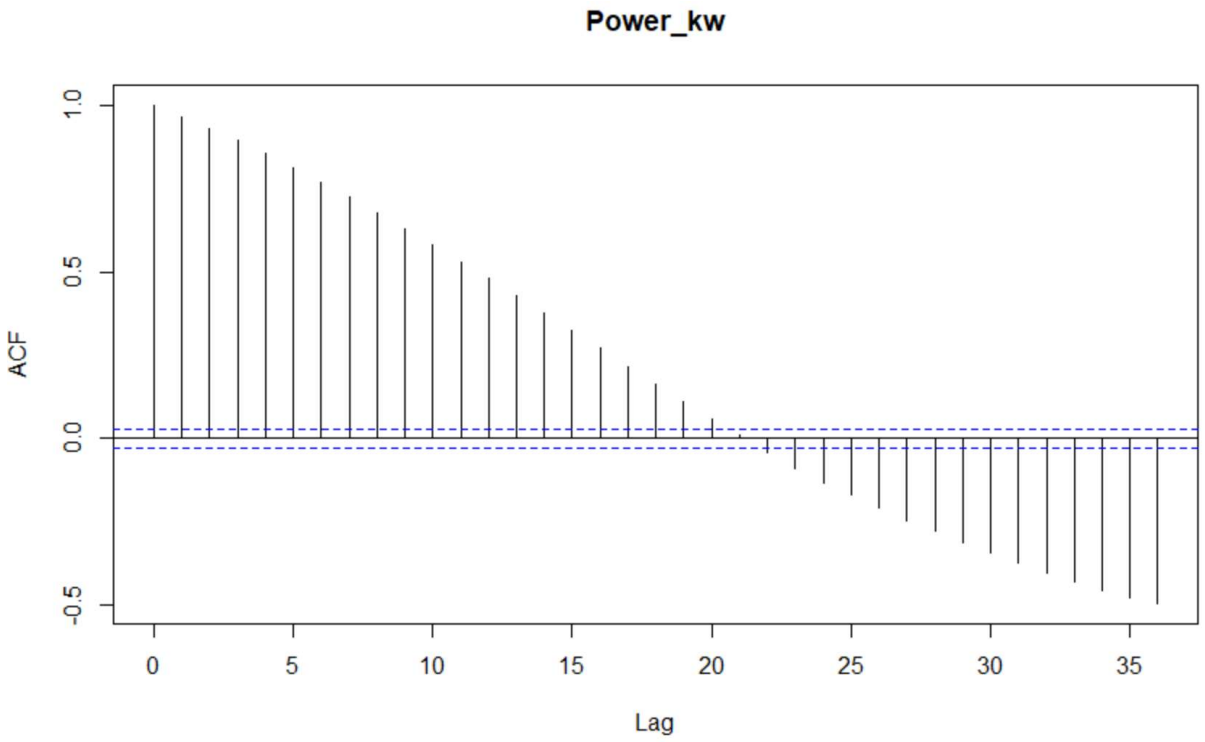
Split the dataset into train and test

```
el_train <- window(Elec_train %>%
                     select(-c(Timestamp, Temp_c)) %>%
                     as.ts,
                   start = 1,
                   end = sum(fre_elec) - 96) #Everything except the last full day (up to 15/02/2010 23:45)

el_test <- window(Elec_train %>%
                     select(-c(Timestamp, Temp_c)) %>%
                     as.ts,
                   start = sum(fre_elec) - 96 + 1,
                   end = sum(fre_elec)) #Only the last day of full data (16/02/2010)
```

Look at the auto-correlation.

```
tmp=acf(el_train,type="cor",plot = FALSE)
plot(tmp)

tmp$acf[1:3,1,1]
```

[1] 1.0000000 0.9646331 0.9321820



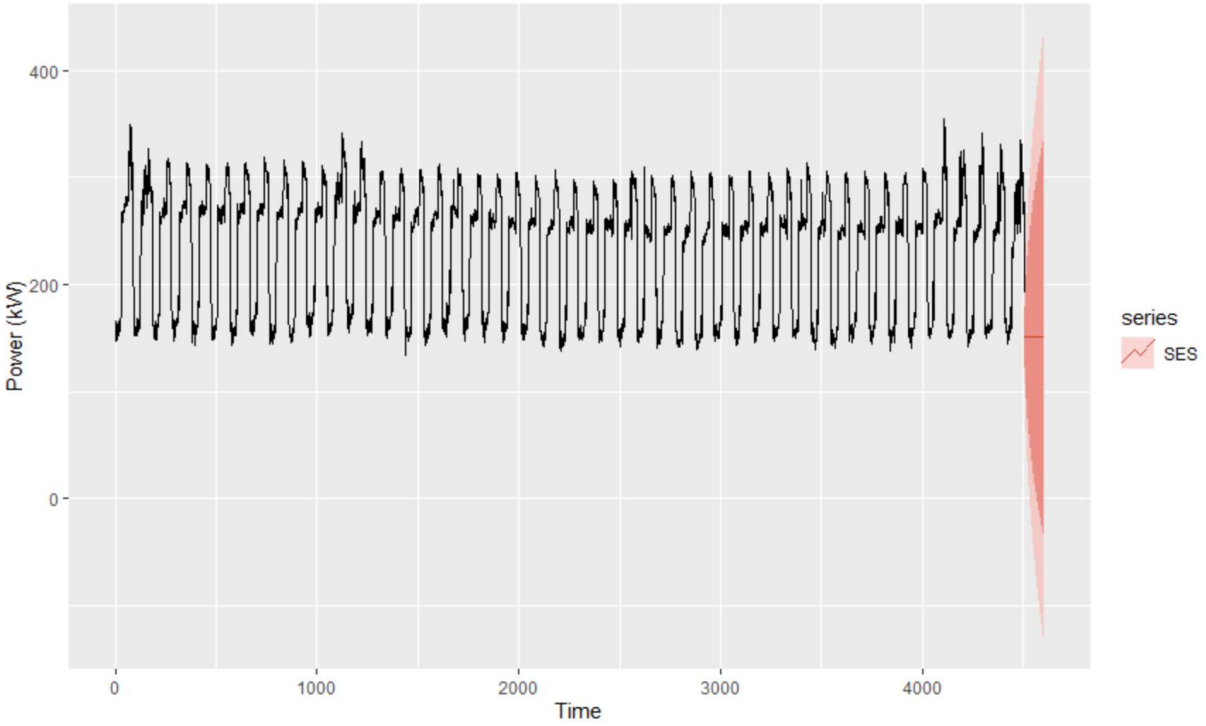
The ACF suggests a seasonal pattern.

1 Forecasting without Covariates

1.1 SES

Let's try Simple Exponential Smoothing.

```
SES = ses(el_train,h=96)
autoplot(el_train) +
  autolayer(SES,series='SES',PI=TRUE)
```



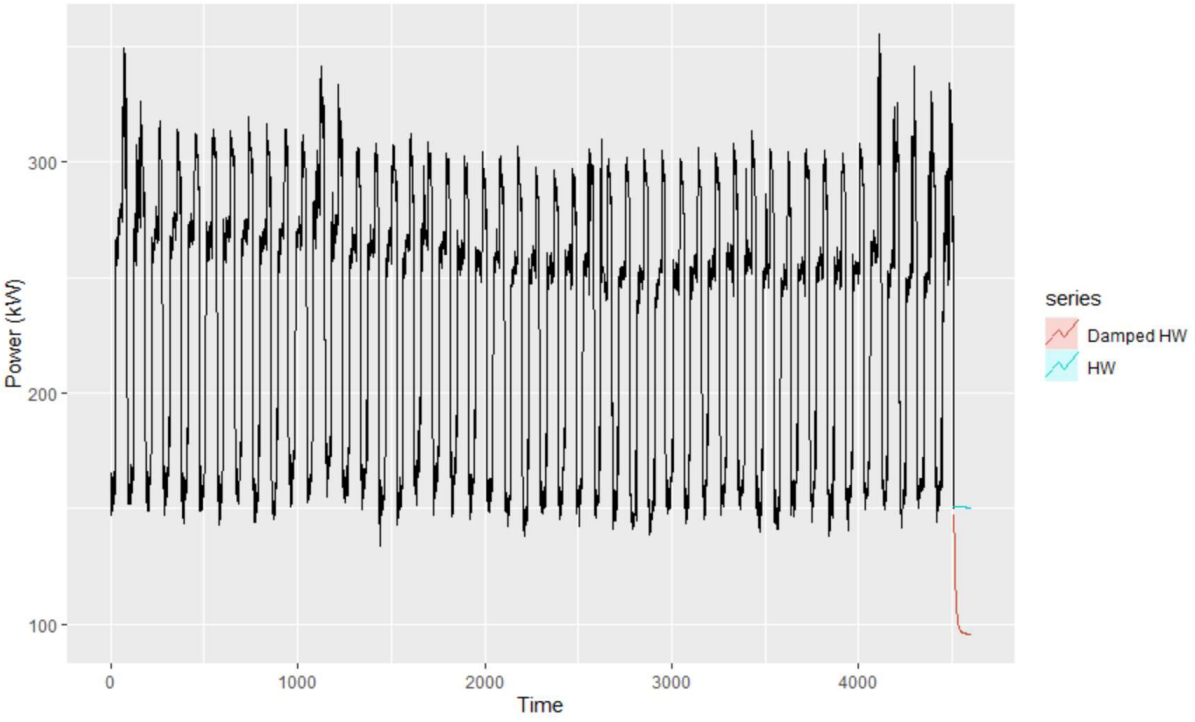

```
print(sqrt(mean((SES$mean-el_test[,"Power_kw"])^2)))
```

[1] **88.88736**

1.2 Holt-Winters

Try Holt-Winters exponential smoothing.

```
HOLT1=holt(el_train,h=96)
HOLT2=holt(el_train,damped=TRUE,phi=0.9,h=96)
autoplot(el_train) +
  autolayer(HOLT1,series='HW',PI=FALSE) +
  autolayer(HOLT2,series='Damped HW',PI=FALSE)
```




```
>print(sqrt(mean((HOLT1$mean-el_test[,"Power_kw"])^2))) #No damping effect

[1] 88.82056

>print(sqrt(mean((HOLT2$mean-el_test[,"Power_kw"])^2))) # With damping effect

[1] 129.1931
```

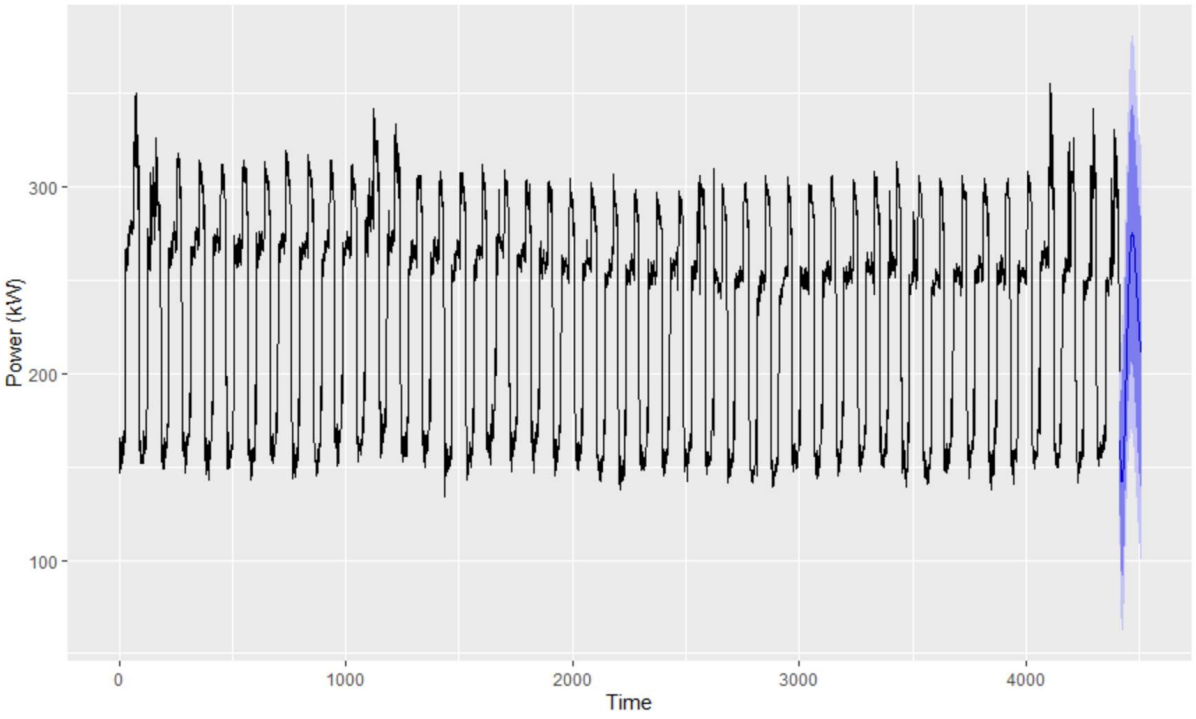
The damping effect does NOT improve things. Holt-Winters without damping provides a slightly better result than Simple Exponential Smoothing, but the results could be improved.

1.3 Auto ARIMA

Let's try auto.arima.

```
#AUTO ARIMA
armModel <- auto.arima(el_train, D=1) #Try to force the seasonality using D=1
fcstArm <- forecast(armModel,h=96)

autoplot(el_train, ylab='Power (kW)') +
  autolayer(fcstArm,series='SARIMA - no covariate',PI=TRUE, col='blue')
```

```
>summary()
ARIMA(3,0,4) with non-zero mean

Coefficients:
         ar1     ar2     ar3     ma1     ma2     ma3     ma4     mean
      0.9950  0.9351  -0.9398  -0.1017  -0.9307  0.1071  -0.0324  231.3400
s.e.  0.0078  0.0134   0.0077   0.0168   0.0172  0.0155   0.0158    0.9647

sigma^2 estimated as 213.6:  log likelihood=-18087.04
AIC=36192.08   AICc=36192.12   BIC=36249.6

Training set error measures:
                     ME    RMSE      MAE       MPE    MAPE     MASE         ACF1
Training set 9.476376e-05 14.6007 7.656776 -0.4095884 3.46415 1.073738 -0.0005560928
```




```
#Check RMSE

print(sqrt(mean((fcstArm$mean-el_test["Power_kw"])^2)))

[1] 37.71583
```

The model does not recognise the seasonality. Even trying to force the seasonality with D=1 does not add the seasonality to the ARIMA(3,0,4) model.

However, it gives a better result than exponential smoothing. Let's try manually tuning SARIMA in order to see if we can improve the result.

1.4 Manual SARIMA

Using the non-seasonal parameters from the auto-arima, attempt SARIMA adding the seasonal part manually. Difference once to remove the seasonality.

1.4.1 ARIMA(3,0,4)(0,1,0)(96)

```
fitSar1 <- Arima(el_train,order = c(3,0,4),
            seasonal = list(order=c(0,1,0),period=96),
            lambda = NULL)
```

```
ARIMA(3,0,4)(0,1,0)[96]

Coefficients:
      ar1     ar2     ar3     ma1     ma2     ma3     ma4
    0.9481  -0.0897  -0.0105  -0.2313  -0.0111  0.1103  -0.2289
s.e.  0.1414   0.0917   0.0666   0.1407   0.0949  0.0285   0.0397

sigma^2 estimated as 112.3:  log likelihood=-16306.53
AIC=32629.06   AICc=32629.1   BIC=32680.02

Training set error measures:
                    ME     RMSE      MAE       MPE     MAPE      MASE        ACF1
Training set -0.042418 10.47486 6.091744 -0.1167919 2.766339 0.8542679 0.0002020834
```

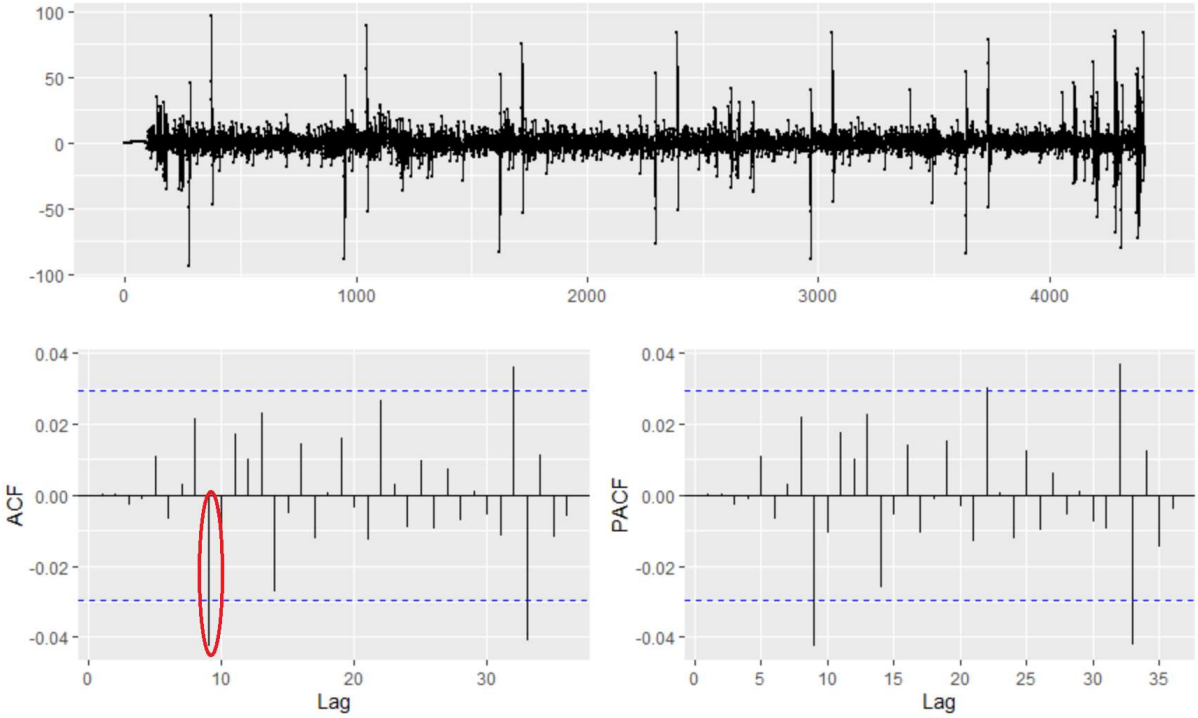
Check the residuals.

```
> checkresiduals(fitSar2)
Ljung-Box test

data:  Residuals from ARIMA(3,0,4)(0,1,0)[96]
Q* = 11.426, df = 3, p-value = 0.009632

Model df: 7.   Total lags used: 10

fitSar1 %>% residuals() %>% ggtsdisplay()
```

Low p-value means that there is still something to modalise.

Do the forecast and check the RMSE.

```
fcstSarm1 <- forecast(fitSar1,h=96)

print(sqrt(mean((fcstSarm1$mean-el_test[,"Power_kw"])^2)))
[1] 19.83111 #RMSE
```



Looks like a Moving Average (MA) of order 1 for the seasonal part, so let's change the Q value to 1. Let's also randomly change the non-seasonal q value to 1.

Try the model: ARIMA(3,0,1)(0,1,1)

1.4.2 ARIMA(3,0,1)(0,1,1)

```
fitSar2 <- Arima(el_train,order = c(3,0,1),
                 seasonal = list(order=c(0,1,1),period=96),
                 lambda = NULL)
```

```
ARIMA(3,0,1)(0,1,1)[96]

Coefficients:
         ar1     ar2     ar3     ma1     sma1
      -0.0553  0.5373  0.1669  0.7424  -0.8727
s.e.   0.0395  0.0316  0.0151  0.0378   0.0083

sigma^2 estimated as 64.83:  log likelihood=-15190.08
AIC=30392.17   AICc=30392.18   BIC=30430.38

Training set error measures:
                    ME    RMSE     MAE      MPE    MAPE     MASE      ACF1
Training set -0.3446564 7.959013 4.760137 -0.24618 2.166501 0.6675318 0.01290208
```

The AIC value is less than last time.

Check the residuals.

```
>checkresiduals(fitSar2)
Ljung-Box test

data:  Residuals from ARIMA(3,0,1)(0,1,1)[96]
Q* = 101.19, df = 5, p-value < 2.2e-16

Model df: 5.   Total lags used: 10
```

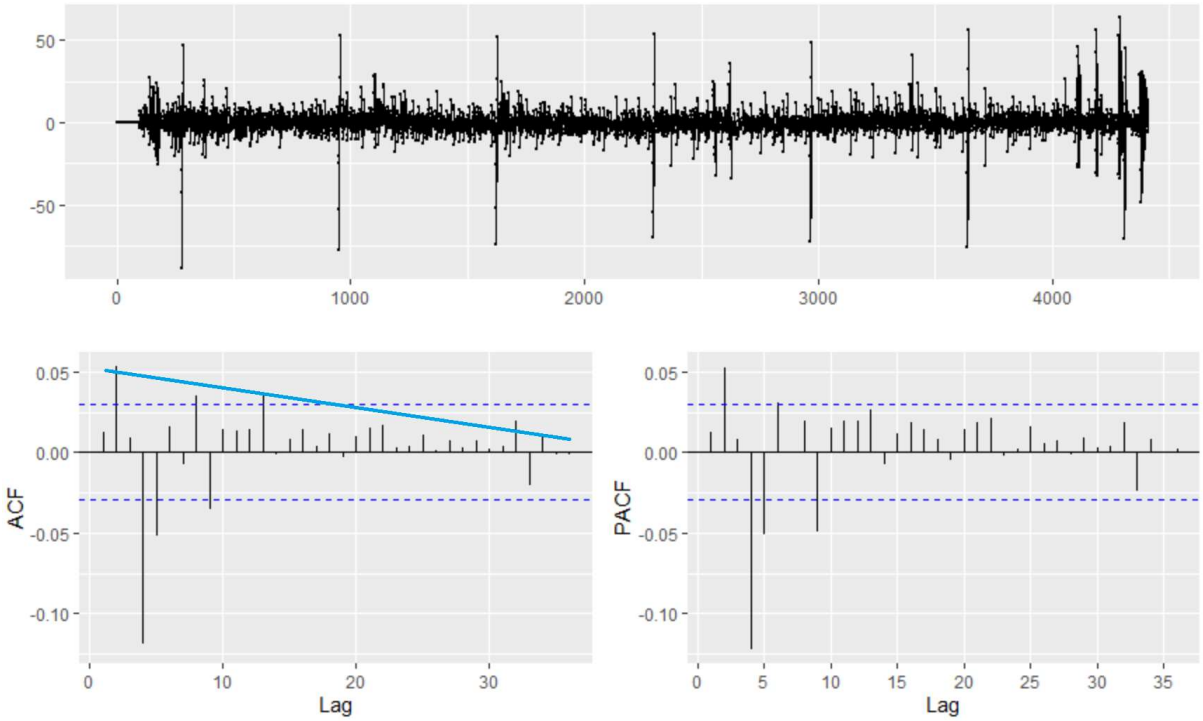
Low p-value means that there is still something to modalise.

Do the forecast and check the RMSE.

```
fcstSarm2 <- forecast(fitSar2,h=96)

print(sqrt(mean((fcstSarm2$mean-el_test[,"Power_kw"])^2)))
[1] 15.07576
```

The RMSE has improved.

Look at the ACF/PACF to see how we can further improve the model.



Although no real trend was observed when the data was plotted, the ACF and PACF appear to be showing some kind of decreasing linear trend towards zero. Let's add some differencing to the non-seasonal part: ARIMA(3,1,1,)(0,1,1).

1.4.3 ARIMA(3,1,1,)(0,1,1)

```
fitSar3 <- Arima(el_train,order = c(3,1,1),
            seasonal = list(order=c(0,1,1),period=96),
            lambda = NULL)
```

```
ARIMA(3,1,1)(0,1,1)[96]

Coefficients:
        ar1     ar2     ar3     ma1     sma1
    -0.8323  -0.2334  0.0331  0.6208  -0.8767
s.e.   0.0552   0.0238  0.0191  0.0533   0.0082

sigma^2 estimated as 70.25:  log likelihood=-15360.72
AIC=30733.44   AICc=30733.46   BIC=30771.66

Training set error measures:
                    ME     RMSE      MAE        MPE     MAPE      MASE       ACF1
Training set 0.005531446 8.283959 4.936532 -0.06447375 2.254979 0.6922683 0.002710286
```

The AIC value is a bit higher than last time.

Check the residuals.

```
>checkresiduals(fitSar3)
Ljung-Box test

data:  Residuals from ARIMA(3,1,1)(0,1,1)[96]
Q* = 216.2, df = 5, p-value < 2.2e-16

Model df: 5.   Total lags used: 10
```

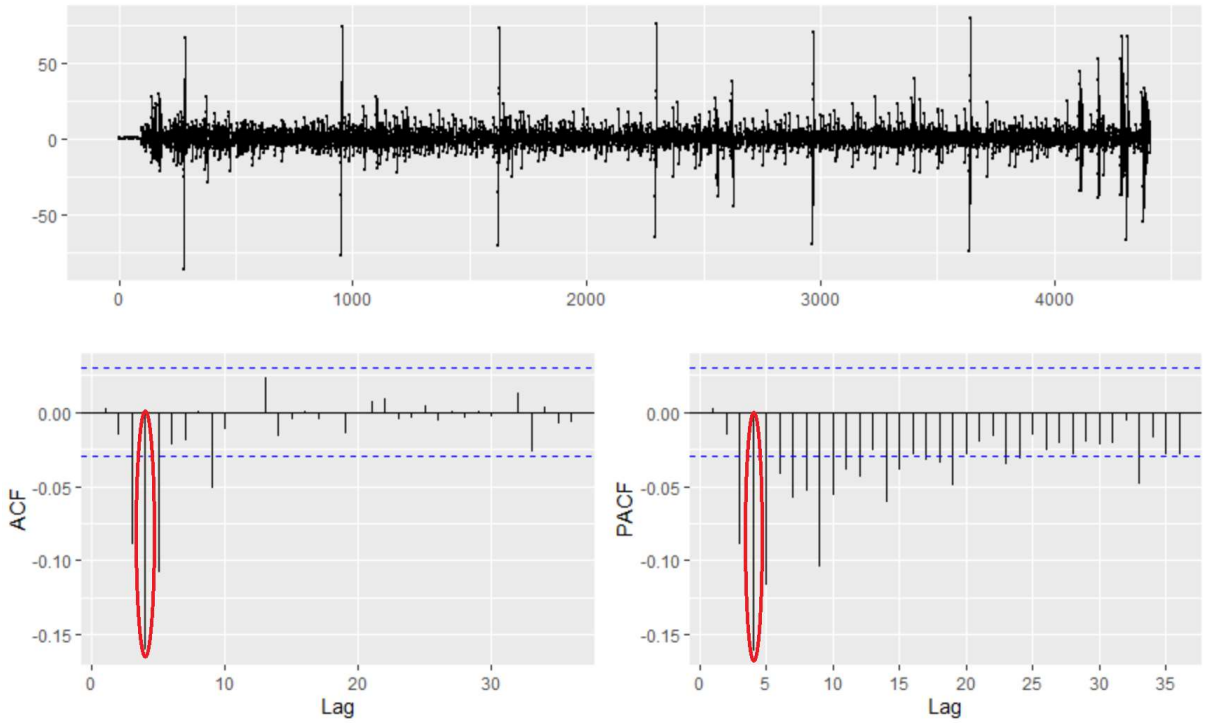
Low p-value means that there is still something to modalise.

Do the forecast and check the RMSE.

```
fcstSarm3 <- forecast(fitSar3,h=96)

print(sqrt(mean((fcstSarm3$mean-el_test[,"Power_kw"])^2)))
#[1] 14.97603
```

The RMSE has improved.

Look at the ACF/PACF to see how we can further improve the model.



Both the ACF and PACF are showing something significant at Lag 4. Let's change the AR parameter for the seasonal part: ARIMA(3,1,1,)(4,1,1)

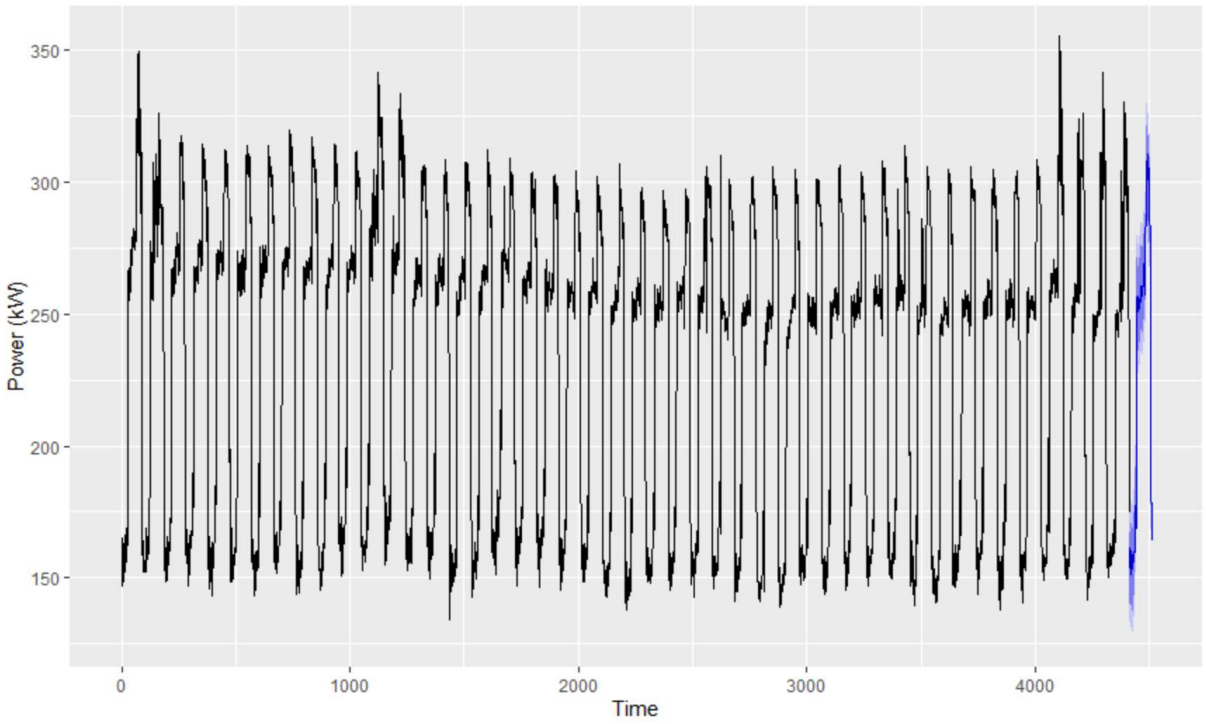
1.4.4 ARIMA(3,1,1,)(4,1,1)

```
Error in makeARIMA(trarma[[1L]], trarma[[2L]], Delta, kappa, SSinit) :
  maximum supported lag is 350
```

An error is generated. Let's try the 4 in the non-seasonal AR part: ARIMA(4,1,1)(0,1,1)

1.4.5 ARIMA(4,1,1,)(0,1,1)

```
fitSar <- Arima(el_train,order = c(4,1,1),
                seasonal = list(order=c(0,1,1),period=96),
                lambda = NULL)

f_fit<-forecast(fitSar, h=96)

autoplot(el_train, ylab='Power (kW)') +
  autolayer(f_fit,series='Manual SARIMA - no covariate',PI=TRUE, col='blue')
```



The plot looks quite good!

```
ARIMA(4,1,1)(0,1,1)[96]

Coefficients:
         ar1     ar2     ar3      ar4      ma1     sma1
      0.6623  0.0879  0.1187  -0.1643  -0.9841  -0.8740
s.e.  0.0154  0.0181  0.0180   0.0153   0.0039   0.0082

sigma^2 estimated as 64.02:  log likelihood=-15160.17
AIC=30334.33   AICc=30334.36   BIC=30378.92

Training set error measures:
                    ME    RMSE      MAE       MPE     MAPE      MASE      ACF1
Training set 0.02579198 7.90743 4.773265 -0.06366446 2.172001 0.6693727 0.01820627
```

The AIC value is the lowest so far!

```
#RMSE Manual SARIMA
print(sqrt(mean((f_fit$mean-el_test[,"Power_kw"])^2)))
[1] 14.5203
```

Best RMSE so far!

1.5 Neural Network

Let's attempt a neural network model.

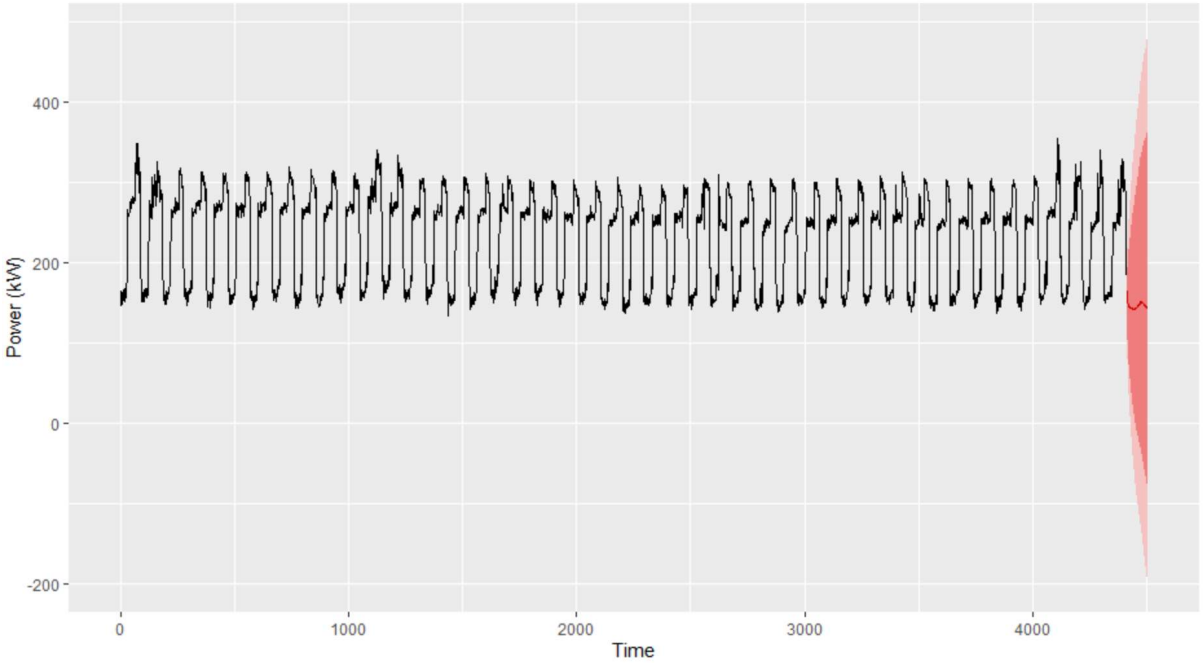
```
#Neural Network
fitNN <- nnetar(el_train)
fcstNN<-forecast(fitNN,h=96)

print(sqrt(mean((fcstNN$mean-el_test[,"Power_kw"])^2)))
1] 26.87508 #RMSE
```

The RMSE is not as good as ARIMA(4,1,1)(0,1,1).

2 Forecasting with Covariates

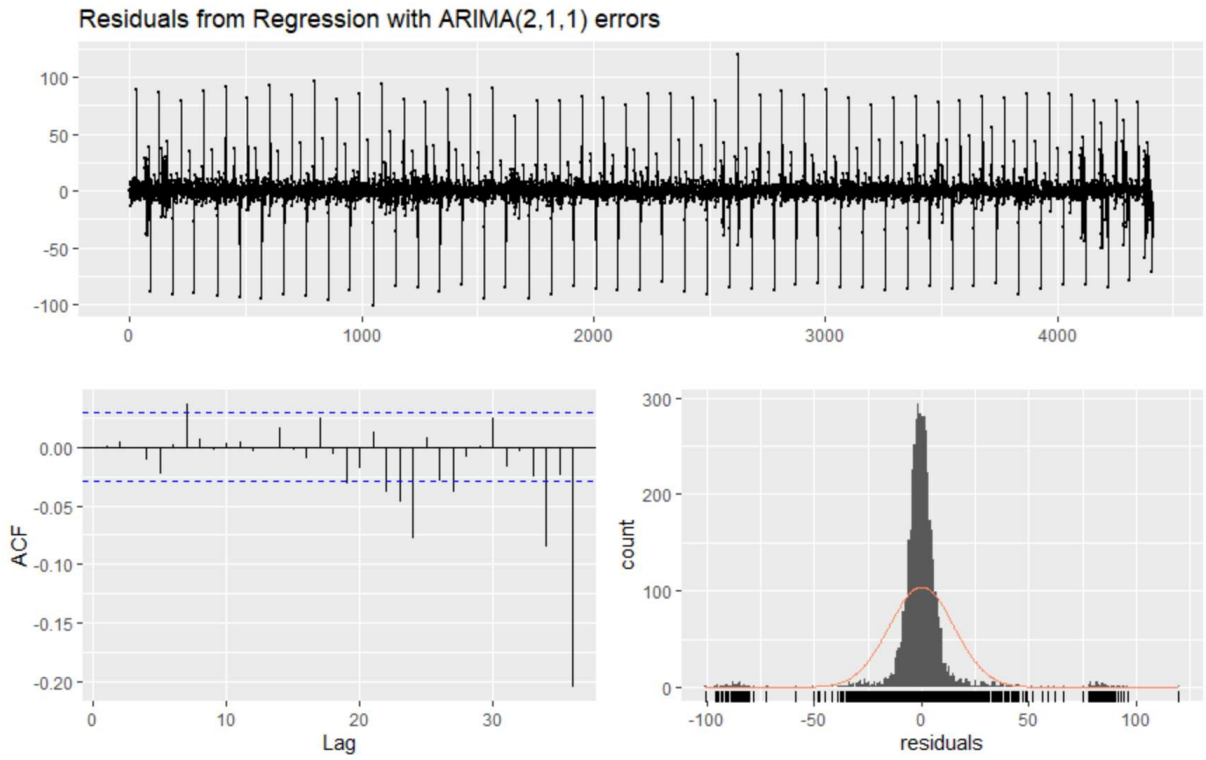
2.1 Auto ARIMA with Covariate

```
# Training 1 regressors
temp_tr1 <- Elec_train[1:(sum(fre_elec) - 96), "Temp_c"] %>% pull

# Training arima
fitArima_tr1 <- auto.arima(el_train, xreg = temp_tr1)

# Training 2 regressors
temp_tr2 <- Elec_train[(sum(fre_elec) - 96 + 1):(sum(fre_elec)), "Temp_c"] %>% pull

# Forecast arima
fitAR_test_tr1 <- forecast(fitArima_tr1, xreg = temp_tr2, h=96)

autoplot(el_train, ylab='Power (kW)') +
  autolayer(fitAR_test_tr1,series='SARIMA - with covariate',PI=TRUE, col='red')

> fitArima_tr1
Series: el_train
Regression with ARIMA(2,1,1) errors

Coefficients:
        ar1     ar2      ma1    xreg
     0.6492  0.0941  -0.6925  1.1373
s.e.  0.0927  0.0156   0.0926  0.4902

sigma^2 estimated as 229.2:  log likelihood=-18239.24
AIC=36488.47   AICc=36488.49   BIC=36520.43
```



Check the residuals.

```
checkresiduals(fitArima_tr1)

Ljung-Box test
data:  Residuals from Regression with ARIMA(2,1,1) errors
Q* = 9.341, df = 6, p-value = 0.1553
Model df: 4.   Total lags used: 10
```



The model is poor, it is not taking the seasonality into account. The RMSE value is significantly worse with the Temperature as covariate in the auto.arima model.

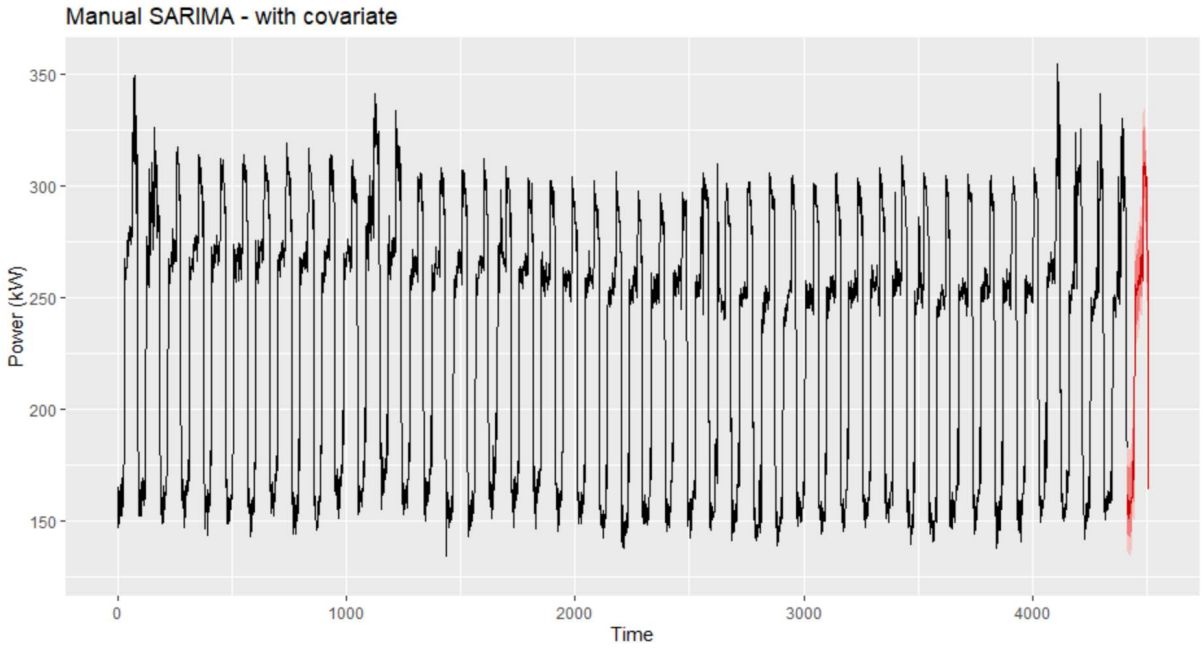
```
print(sqrt(mean((fitAR_test_tr1$mean-el_test[,"Power_kw"])^2)))
[1] 105.6828 #RMSE
```

Let's try the SARIMA model manually.

2.2 Manual SARIMA with Covariate

Let's try the best model we previously got with SARIMA: ARIMA(4,1,1)$(0,1,1)_{96}$

```
fitSar_cov <- Arima(el_train,order = c(4,1,1),
            seasonal = list(order=c(0,1,1),period=96),
            xreg = temp_tr1,
            lambda = NULL)

f_fit_cov<-forecast(fitSar_cov, xreg = temp_tr2, h=96)

autoplot(el_train, ylab='Power (kW)', main='SARIMA - with covariate') +
 autolayer(f_fit_cov, PI=TRUE, col='red')
```



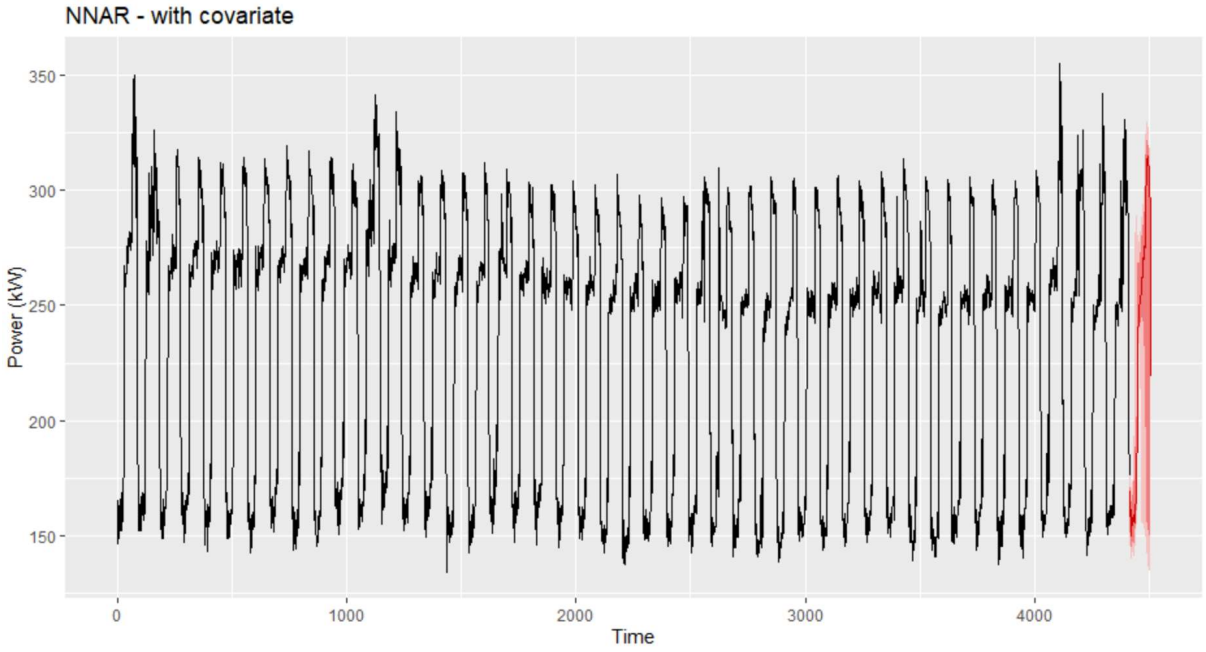
The result is comparable with Manual SARIMA without covariate (RMSE:**14.5203**). The RMSE is only fractionally higher.

```
print(sqrt(mean((f_fit_cov$mean-el_test[,"Power_kw"])^2)))
[1] 14.53807
```

2.3 Neural Network with Covariate

Let's try a neural network manually specifying a size of 20.

```
fitNN_tr1 <- nnetar(el_train,
                    xreg = Elec_train[1:(sum(fre_elec) - 96), "Temp_c"],
                    size = 20,
                    scale.inputs = T)

fitNN_test_tr1 <- forecast(fitNN_tr1,
                       xreg = Elec_train[(sum(fre_elec) - 96 + 1):(sum(fre_elec)), "Temp_c"],
                       PI=TRUE,
                       h=96)

autoplot(el_train, ylab='Power (kW)', main='NNAR - with covariate') +
 autolayer(fitNN_test_tr1,PI=TRUE, col='red')
```

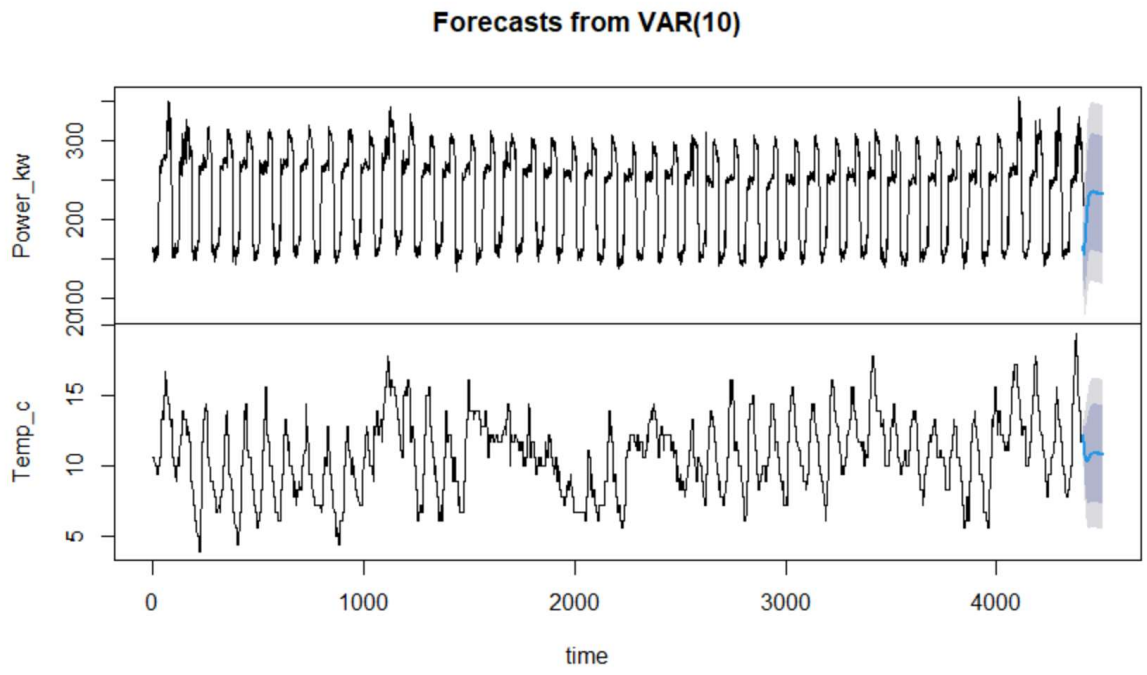



```
print(sqrt(mean((fitNN_test_tr1$mean-el_test[,"Power_kw"])^2)))
[1] 19.48748
```

2.4 Vectorial Auto-Regressive Model

Choose the order with the VARselect function

```
library(vars)
VARselect(el_train2, lag.max=10, type="const")

var <- VAR(el_train2, p=10,type = "const")

prev=forecast(var,h=96,PI=FALSE)
plot(prev)
```



Check the RMSE.

```
print(sqrt(mean((prev$forecast$Power_kw$mean-el_test[,"Power_kw"])^2)))
[1] 49.05771
```

The RMSE is not as good as the SARIMA model with Covariate: ARIMA(4,1,1)$(0,1,1)_{96}$.

3 Summary

The Neural Network and SARIMA models yielded the best results. The best model in both categories: Forecast without Covariate and Forecast with Covariate is the SARIMA(4,1,1)$(0,1,1)_{96}$ model which yielded an RMSE of **14.5203** and **14.53807** respectively.

The quadratic effect between electricity consumption and temperature that was observed in the 'elecdaily' dataset in the fpp2 package was not explored. This is because the observations here cover only a short period of time (around 6 weeks) and the quadratic effect is only observed over a longer period of time, such as a year.

The R code used to generate the final forecasts can be found in the Appendix.

4 Appendix

```
library(lubridate) # Date handling
library(dplyr) # General data handling and piping
library(readxl) # Import xlsx files
library(ggplot2) # Very nice plots
library(forecast)

# Import data
Elec_train <- read_excel("C:/Users/yalaz/Documents/DSTI/Modules/Time Series Analysis/Elec-
train.xlsx") %>%
  mutate(Timestamp = mdy_hm(Timestamp), # Convert column to correct date format
         Power_kw = as.numeric(`Power (kW)`), # Rename vars for convenience
         Temp_c = as.numeric(`Temp (C°)`),
         .keep = "none") # Deletes old copies

# Count observations by day
fre_elec <- Elec_train %>%
  mutate(day = floor_date(Timestamp, unit = "day")) %>%
  filter(day <= ymd("2010-02-17")) %>% count(day) %>% pull(n)

# All entries
el_all <- window(Elec_train %>%
                   select(-c(Timestamp, Temp_c)) %>%
                   as.ts,
                 start = 1,
                 end = sum(fre_elec))

#Final without covariates
fitSar <- Arima(el_all,order = c(4,1,1),
                seasonal = list(order=c(0,1,1),period=96),
                lambda = NULL)

f_fit<-forecast(fitSar, h=96)
f_fit$mean

#Final with covariates
temp_tr_all <- Elec_train[1:(sum(fre_elec)), "Temp_c"] %>% pull
temp_tr2 <- Elec_train[(sum(fre_elec) - 96 + 1):(sum(fre_elec)), "Temp_c"] %>% pull

fitSar_all <- Arima(el_all,order = c(4,1,1),
                    seasonal = list(order=c(0,1,1),period=96),
                    xreg = temp_tr_all,
                    lambda = NULL)

f_fit_all<-forecast(fitSar_all, xreg = temp_tr2, h=96)
f_fit_all$mean
```