Time Series Forecasting Exam

Oriane Okou

In this Project,we will analyse the electricity consumption of a building and the outdoor air temperature over time. Our dataset contains for each timestamp every 15 minutes, the electricity consumption in kW and the outdoor air temperature in C°, from 1/1/2010 1:15 to 2/20/2010 23:45, and temperature are also available for 2/21/2010.The goal is to forecast electricity consumption for 2/21/2010.

1. Forecast based on electricity consumption only

In the first part, we will forecast the electricity consumption without using outdoor temperature:

We start by loading necessary package

```
library(readxl)
```

```
## Warning: le package 'readxl' a été compilé avec la version R 4.4.3
```

```
library(forecast)
```

```
## Warning: le package 'forecast' a été compilé avec la version R 4.4.3

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(ggplot2)
```

```
## Warning: le package 'ggplot2' a été compilé avec la version R 4.4.3
```

```
library(openxlsx)
```

```
## Warning: le package 'openxlsx' a été compilé avec la version R 4.4.3
```

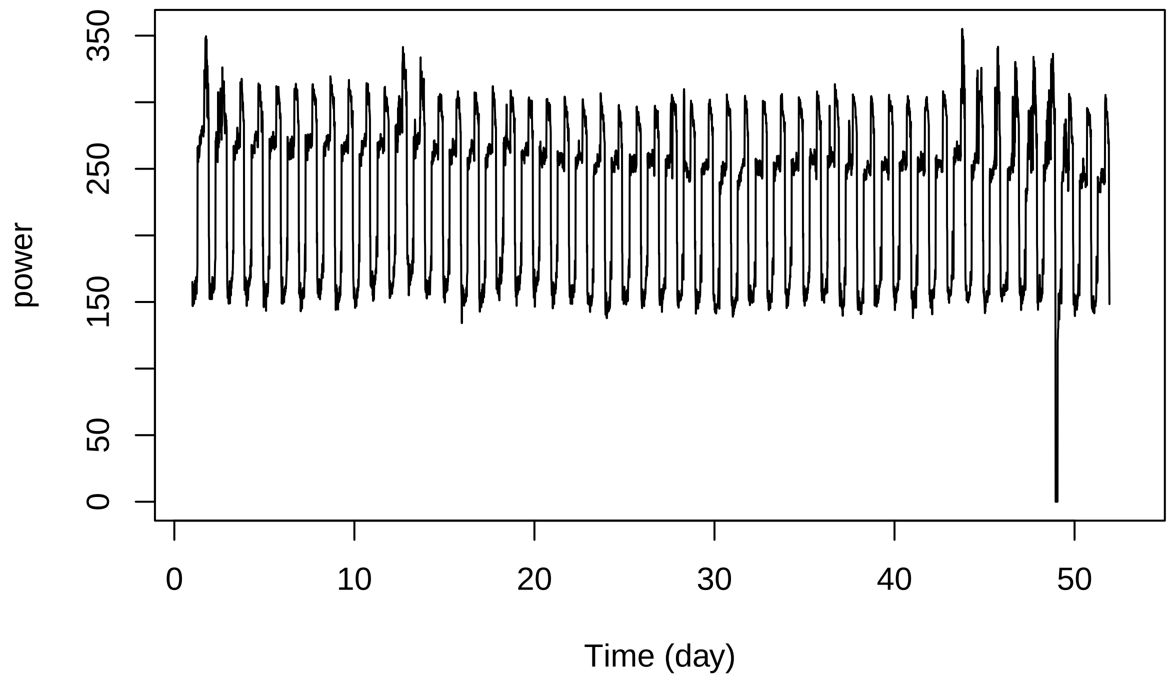
Analysis of the Time Series

We load the data and plot them.

```
data<-readxl::read_excel("2025-06-Elec-train (1).xlsx")
head(data)
```

```
## # A tibble: 6 x 3
##   Timestamp           `Power (kW)` `Temp (C°)`
##   <dttm>                     <dbl>       <dbl>
## 1 2010-01-01 01:15:00         165.        10.6
## 2 2010-01-01 01:30:00         152.        10.6
## 3 2010-01-01 01:45:00         147.        10.6
## 4 2010-01-01 02:00:00         154.        10.6
## 5 2010-01-01 02:15:00         154.        10.6
## 6 2010-01-01 02:30:00         159         10.6
```

```
power <- ts(data$`Power (kW)`,frequency = 96 ) # Frequency : 4 observations/hour * 24 hour = 96/day
plot(power, xlab = "Time (day)")
```



We clearly see a seasonal pattern, repeated each day we highs during the day and lows during the night.We don't see any particular trend on this period, even though we can notice that the envelop of the curve slightly change over the week, maybe due to a lower consumption on the weekend.

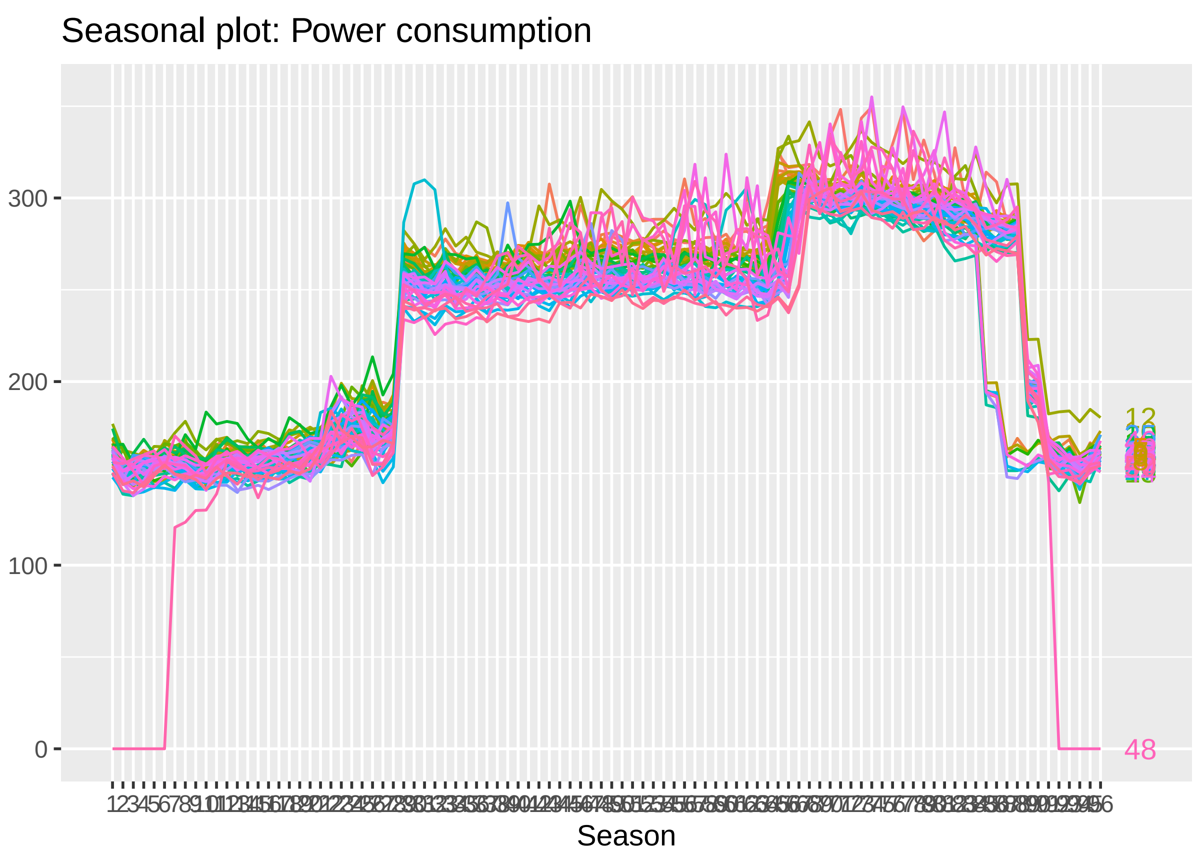
We can also notice a large drop around day ~50 (null points). These outliers will need to be dealt with. Let's further our investigation on the data to choose the best model.

```
summary(power)
```

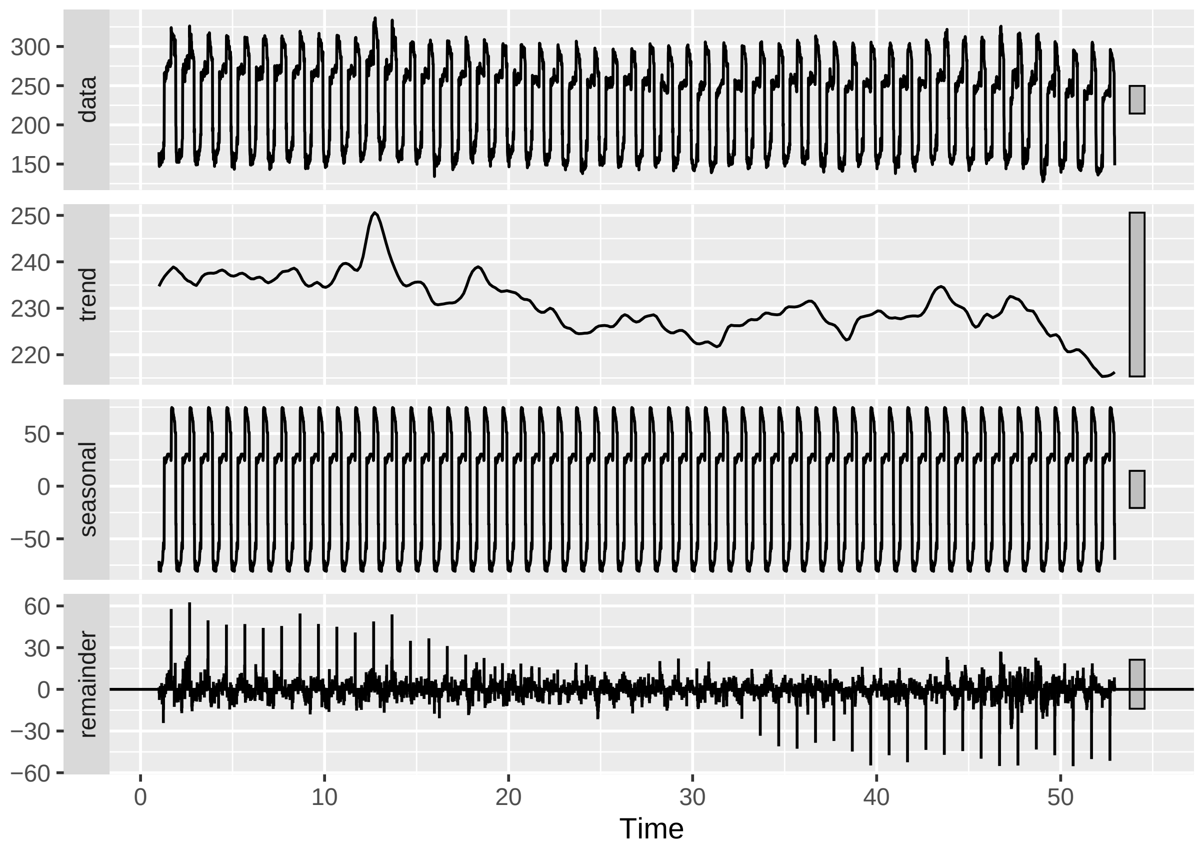
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
##     0.0   162.9   253.0   230.8   277.3   355.1      96
```

```
ggseasonplot(power, year.labels = TRUE, main = "Seasonal plot: Power consumption")
```

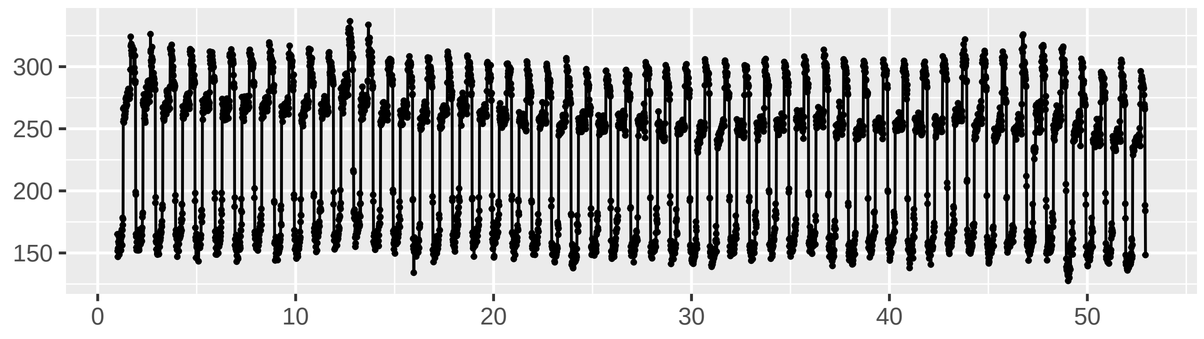
```
## Warning: Removed 96 rows containing missing values or values outside the scale range
## (`geom_line()`).

## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_text()`).
```

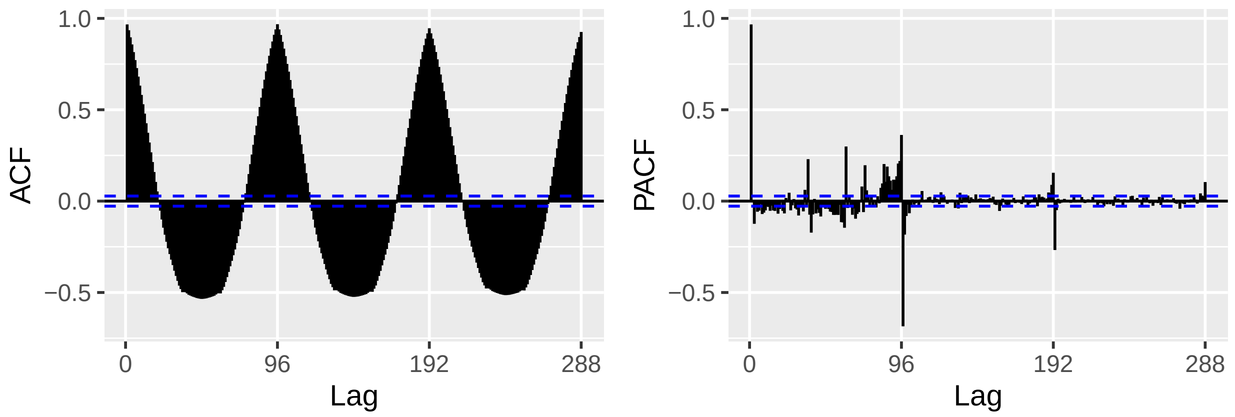


```r
power_clean <- tsclean(power)  # replace the outliers
decomp <- stl(power_clean, s.window = "periodic", robust = TRUE)
autoplot(decomp)
```



```
ggtsdisplay(power_clean)
```





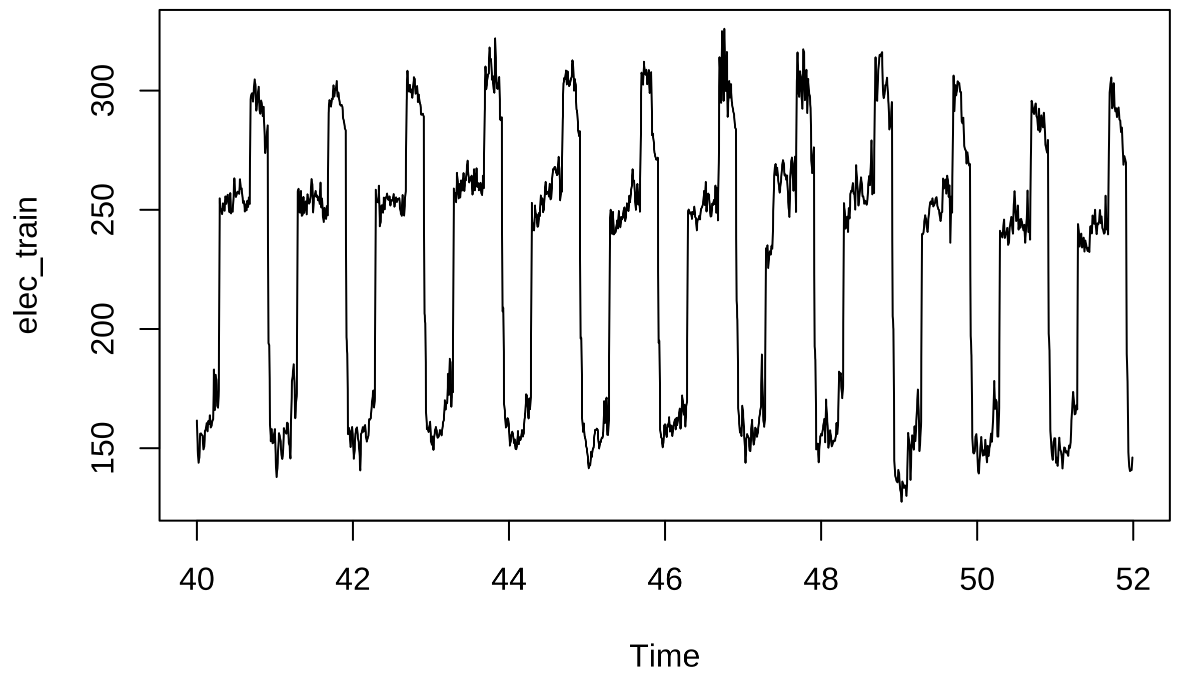
These graphs confirms our first observations: Each curve corresponds to a full day (96 points of 15 min). The daily profile is very regular: -Nighttime trough around ~150–180 kW -Rapid rise in the morning -High plateau during the day (~250–320 kW) -Drop in the evening Abnormally low or zero values are probably due to a failure or incomplete reading.

ACF/PACF: strong autocorrelation at multiples of 96 → clear seasonality.

Models and Forecasting

We start by splitting the serie into train and test.

```
elec_train=window(power_clean,start=c(40,1),end=c(51,96)) # we take fewer data to test our model more q
elec_test=window(power_clean,start=c(52,1))
plot(elec_train)
```



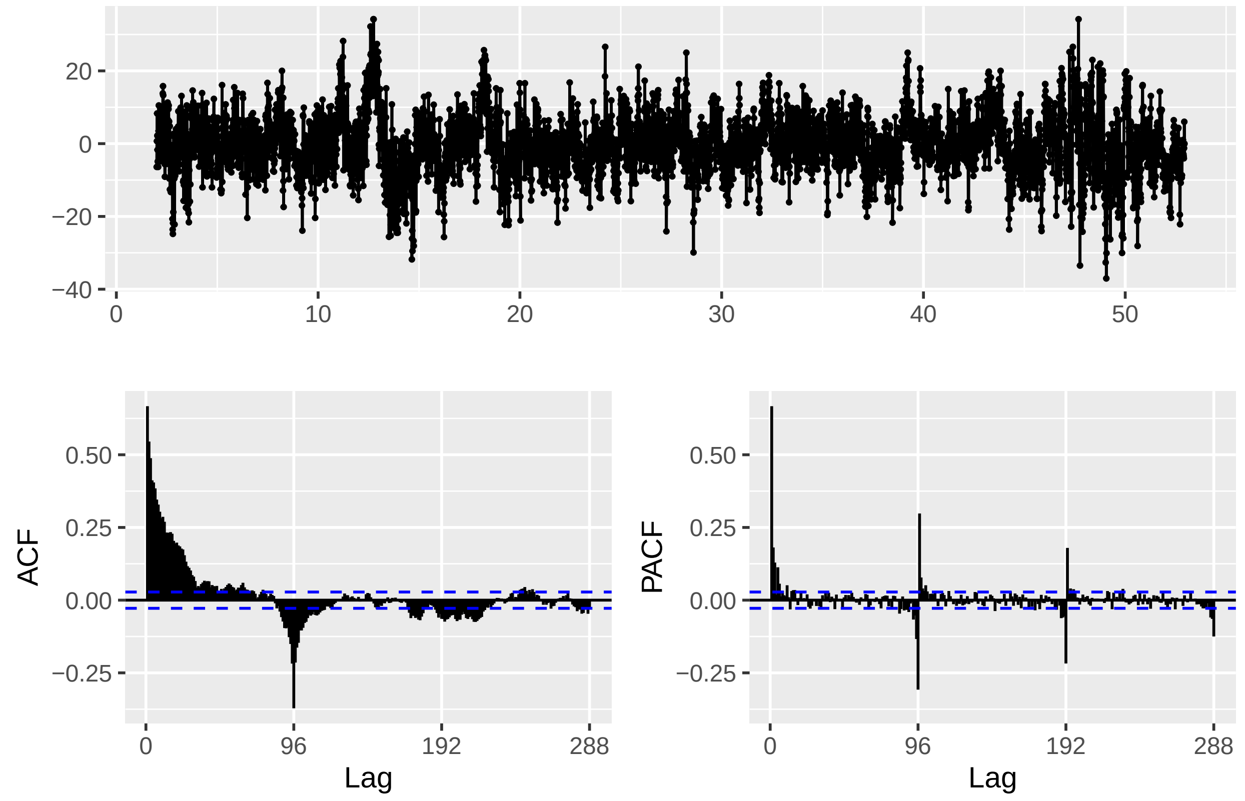
First we start with Holt Winters exponential smoothing.

```
#fit=hw(elec_train,lambda="auto")
#prevHW=forecast(fit,h=18)
#autoplot(elec_test[,"Power(kW)"],series="test data")+autolayer(prevHW$mean,series="HW")
```

96 is a too high frequency for HW model.

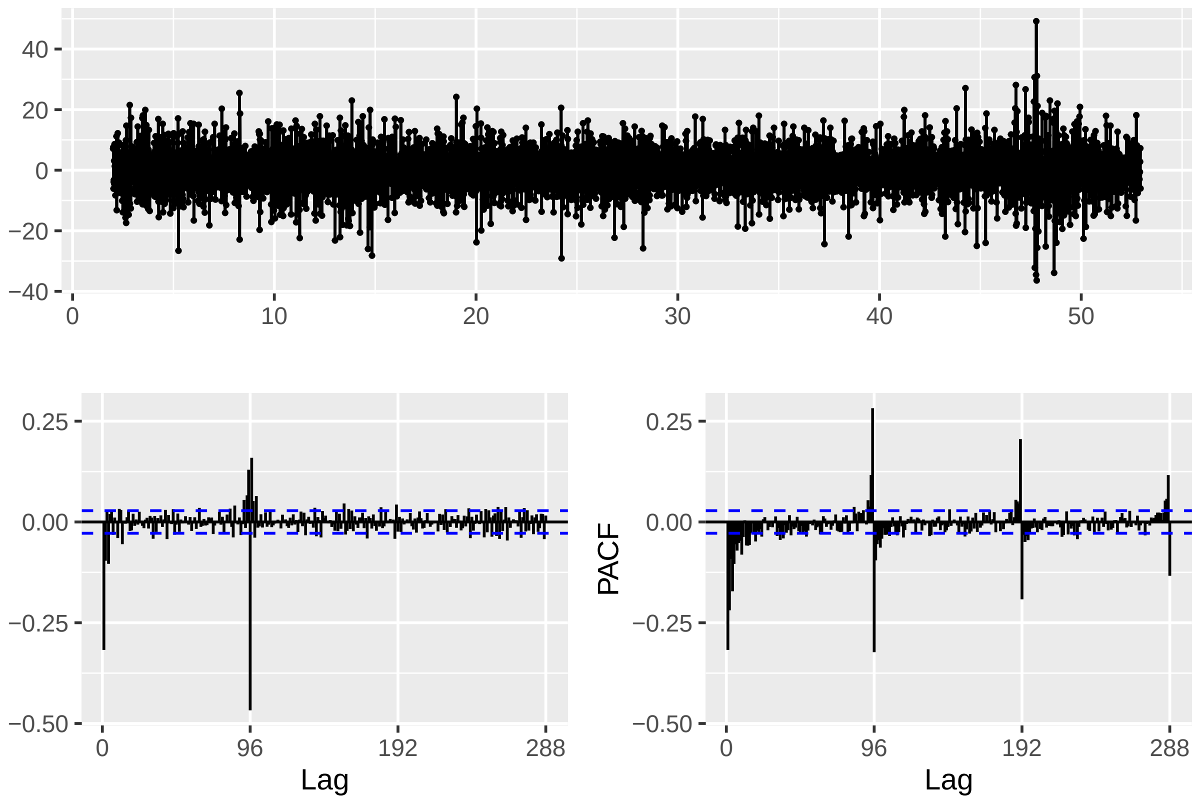
We try a SARIMA model. We start by differentiating the serie.

```
power_diff96 <-(diff(power_clean, lag = 96))
ggtsdisplay(power_diff96)
```



The seasonality has been removed.But it seems like we have a linear trend in the ACF. Let's differentiate again.

```r
power_diff96 <-diff(power_diff96)
ggtsdisplay(power_diff96)
```



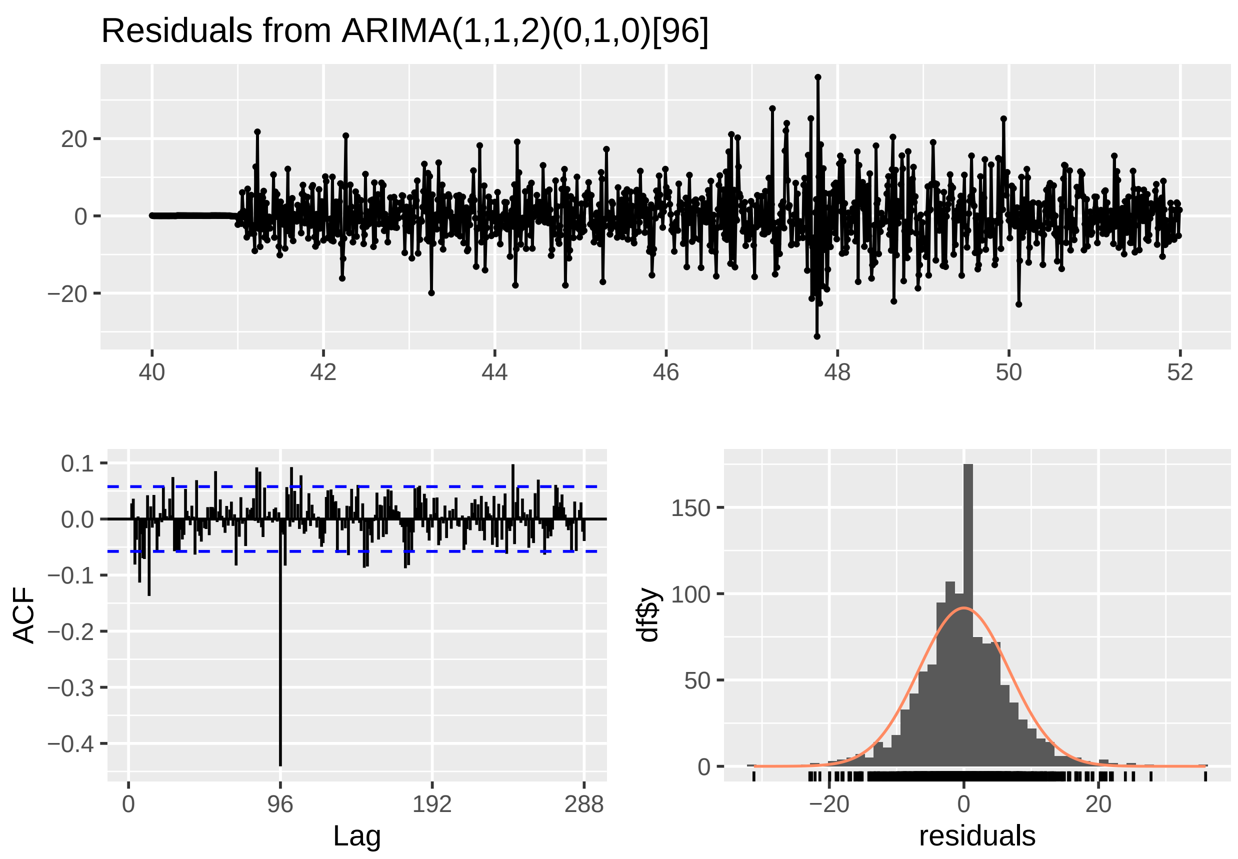
No more trend. No more seasonality. ###Let's check the auto Arima:

```
fit_auto=auto.arima( elec_train )
fit_auto
```

```
## Series: elec_train
## ARIMA(1,1,2)(0,1,0)[96]
##
## Coefficients:
##           ar1     ma1      ma2
##       -0.5160  0.0900  -0.4016
## s.e.   0.0909  0.0879   0.0395
##
## sigma^2 = 49.48:  log likelihood = -3553.7
## AIC=7115.4   AICc=7115.44   BIC=7135.24
```

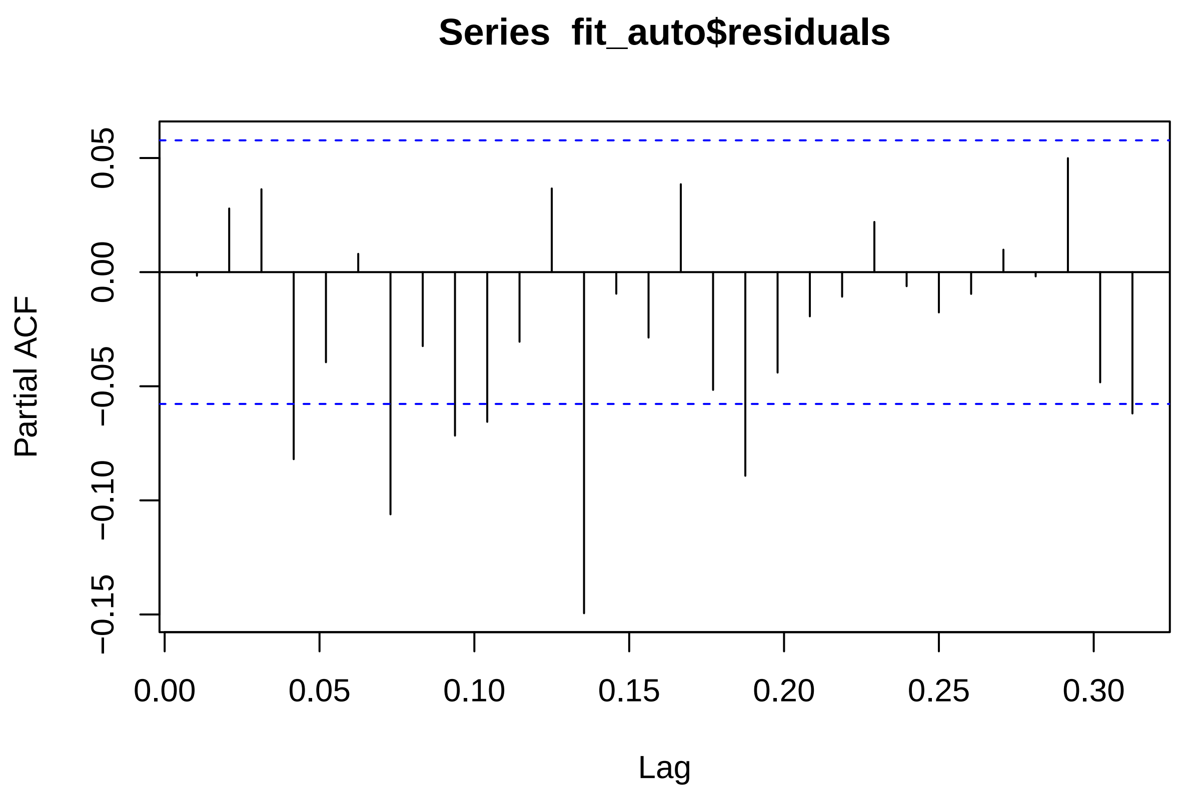
We check the residuals of this SARIMA model to see if it's white noise, and in this case the model covers all the particularity of the data. In the case we see significant spikes in the PACF, we will try to change the parameters

```
checkresiduals(fit_auto)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,2)(0,1,0)[96]
## Q* = 618.82, df = 189, p-value < 2.2e-16
##
## Model df: 3.   Total lags used: 192
```

```
pacf(fit_auto$residuals)
```

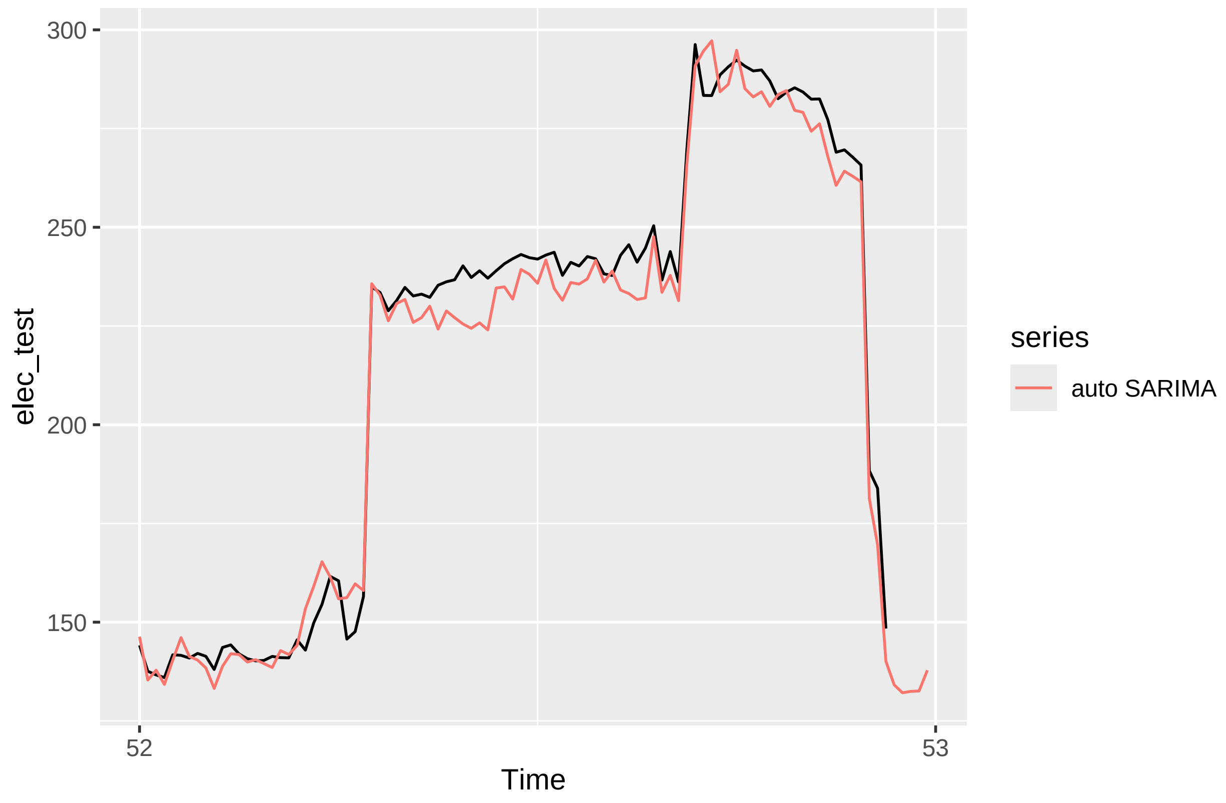


.

```
prev_auto=forecast(fit_auto,h=96)
print(sqrt(mean((prev_auto$mean-elec_test)^2)))
```

```
## [1] 6.688485
```

```
autoplot(elec_test)+autolayer(prev_auto$mean,series="auto SARIMA")
```

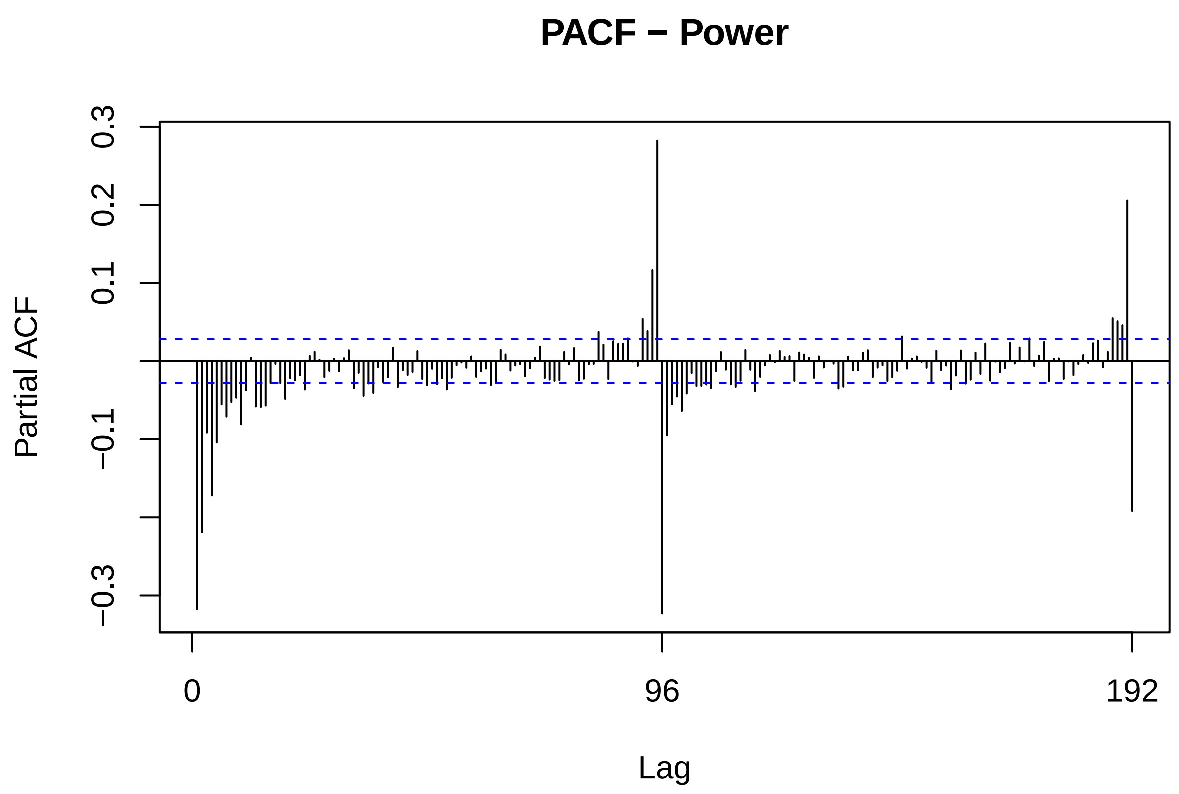


RMSE: 7.628808

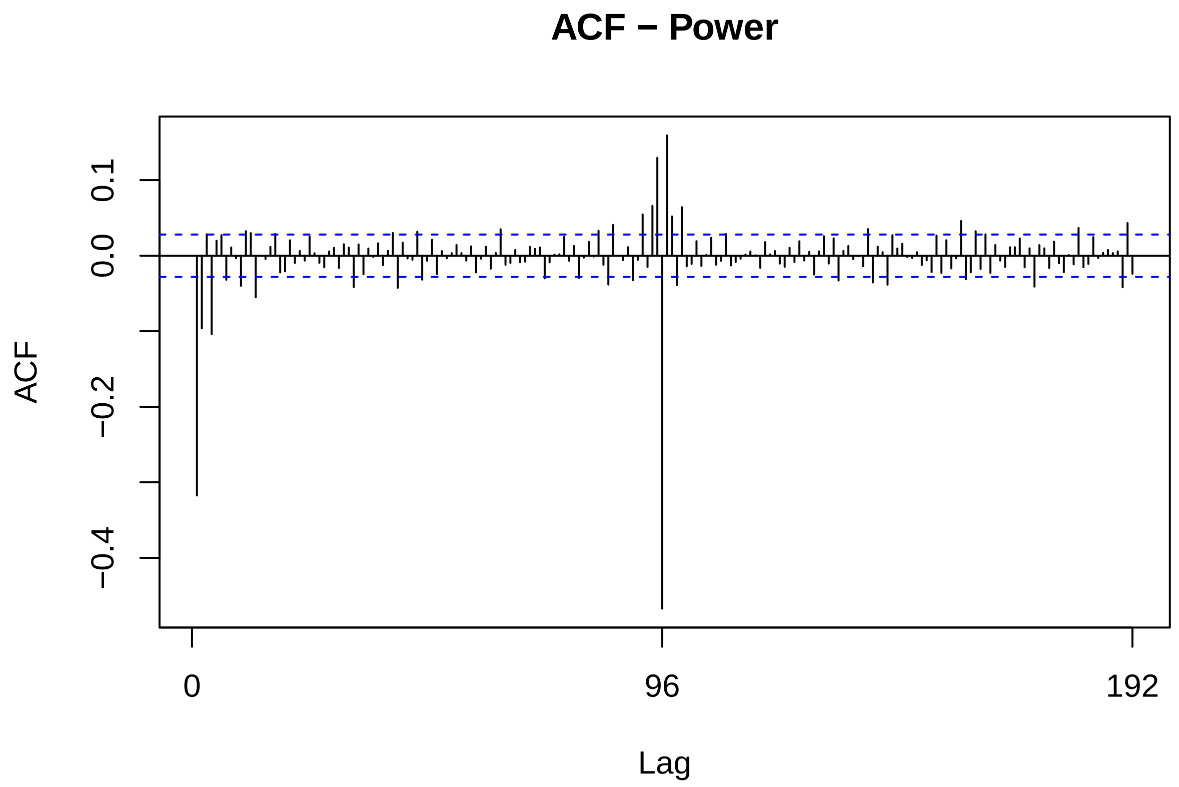
We didn't capture all correlations, we try to do a model manually

###And no let's try it manually: We plot it more clearly:

```
Pacf(power_diff96, main = "PACF - Power")
```



```r
Acf(power_diff96, main = "ACF - Power")
```



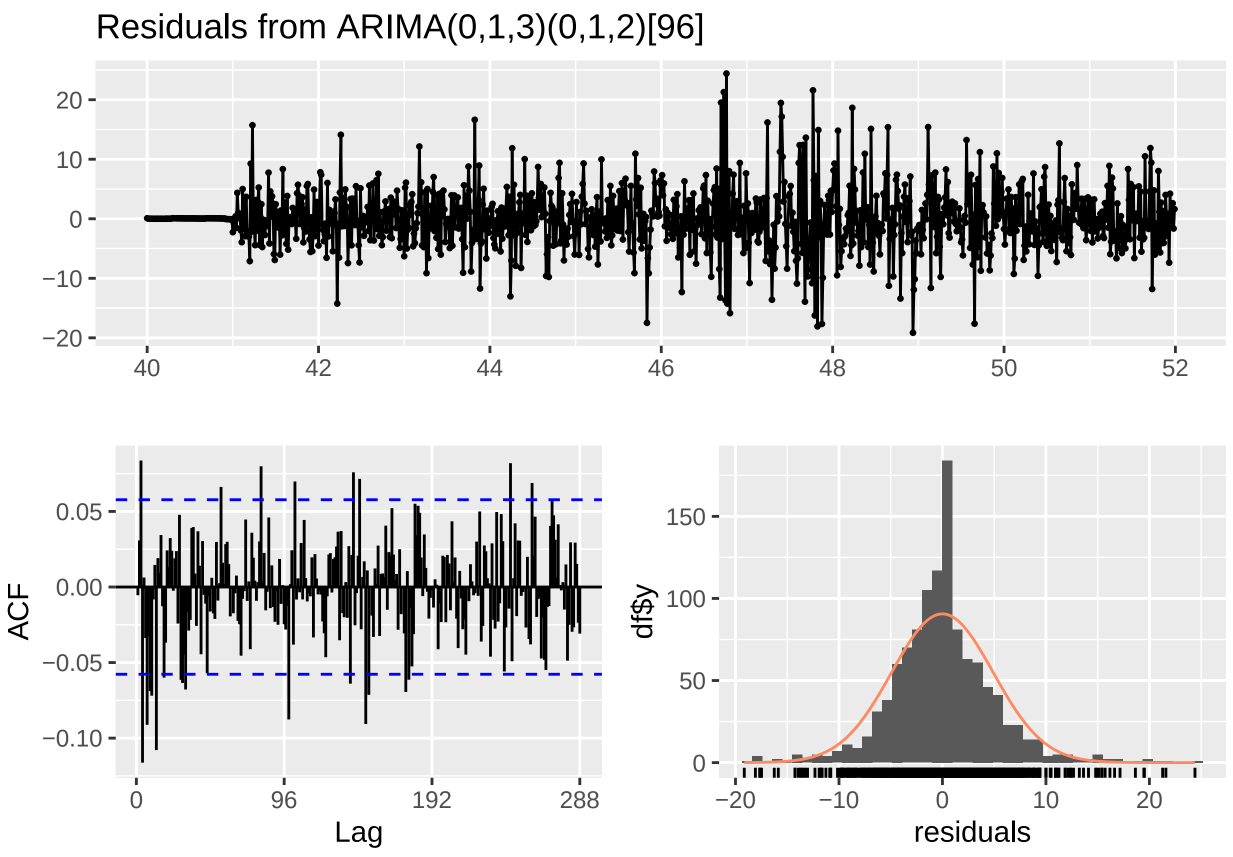
The analysis of the autocorrelation function (ACF) and partial autocorrelation function (PACF) reveals strong dependencies at small lags (up to 4) and regular peaks at multiples of 96, indicating a daily seasonal component. The non-seasonal parameters $p = 4$ and $q = 2$ were selected to model these short-term correlations, while the seasonal parameters $P = 2$ and $Q = 1$ capture the repeating structure over 96 periods. The "last significant peak" rule was not applied, as very large lags (e.g., 192) often correspond to harmonics of the seasonality and are better modeled in the seasonal part rather than by artificially increasing p or q, which would unnecessarily complicate the model and risk overfitting. Stationarity tests required both non-seasonal ($d = 1$) and seasonal ($D = 1$) differencing, leading to the final model $SARIMA(4, 1, 2)(2, 1, 1)_{96}$.

Let's test the $SARIMA(4, 1, 2)(2, 1, 1)_{96}$: After various tries of tuning a SARIMA model (changing the parameters, checking the residuals,most significant coefficient, trying to choose higher/lower parameters), we still have errors and can't find a better model due to optimization issues. We at last obtain this SARIMA:

```r
fit=arima(elec_train, order = c(0,1,3), seasonal = list(order = c(0,1,2), period = 96))
fit
```

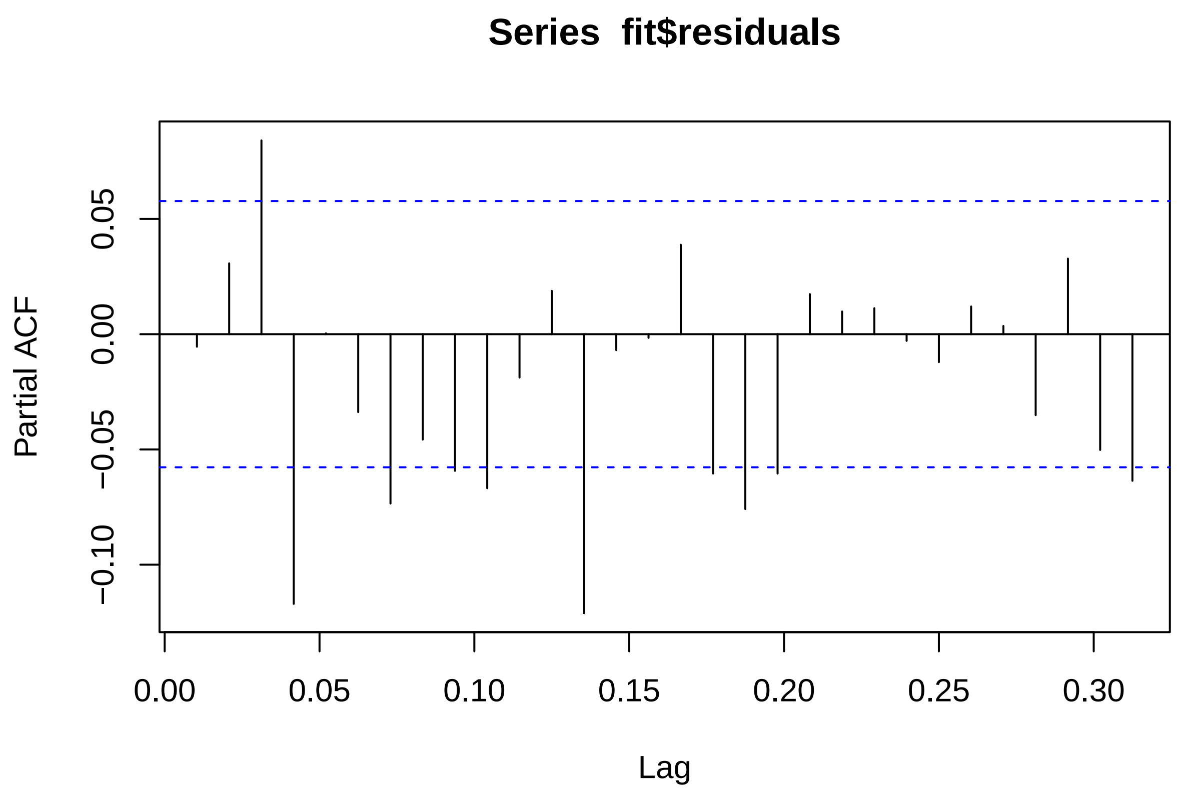
```
##
## Call:
## arima(x = elec_train, order = c(0, 1, 3), seasonal = list(order = c(0, 1, 2),
##     period = 96))
##
## Coefficients:
##           ma1      ma2     ma3     sma1     sma2
##       -0.4401  -0.1983  0.0317  -0.9208  -0.0791
## s.e.   0.0320   0.0409  0.0353   0.1415   0.0336
##
## sigma^2 estimated as 26.29:  log likelihood = -3334.45,  aic = 6680.9
```

```r
checkresiduals(fit)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,3)(0,1,2)[96]
## Q* = 282.05, df = 187, p-value = 8.573e-06
##
## Model df: 5.   Total lags used: 192
```

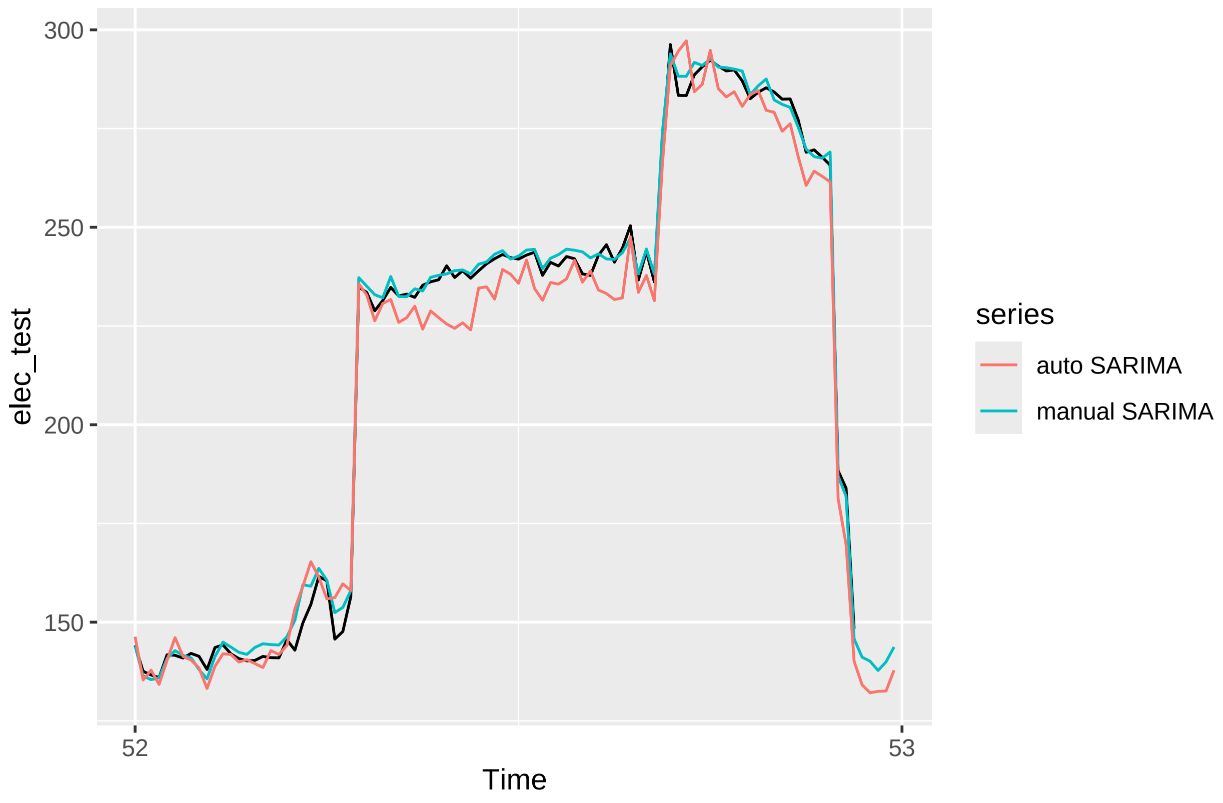
```r
pacf(fit$residuals)
```



```
prev=forecast(fit,h=96)
print(sqrt(mean((prev$mean-elec_test)^2)))
```

```
## [1] 2.655751
```

```
autoplot(elec_test)+autolayer(prev$mean,series="manual SARIMA")+autolayer(prev_auto$mean,series="auto S
```



Comparing the AIC, we stay with our manual forecast even though it's not the best one and it don't contain every auto correlations.

```
df <- data.frame(
  date = time(prev$mean),
  value = as.numeric(prev$mean)
)

# Save in CSV
write.csv(df, "elec_1.csv", row.names = FALSE)
```

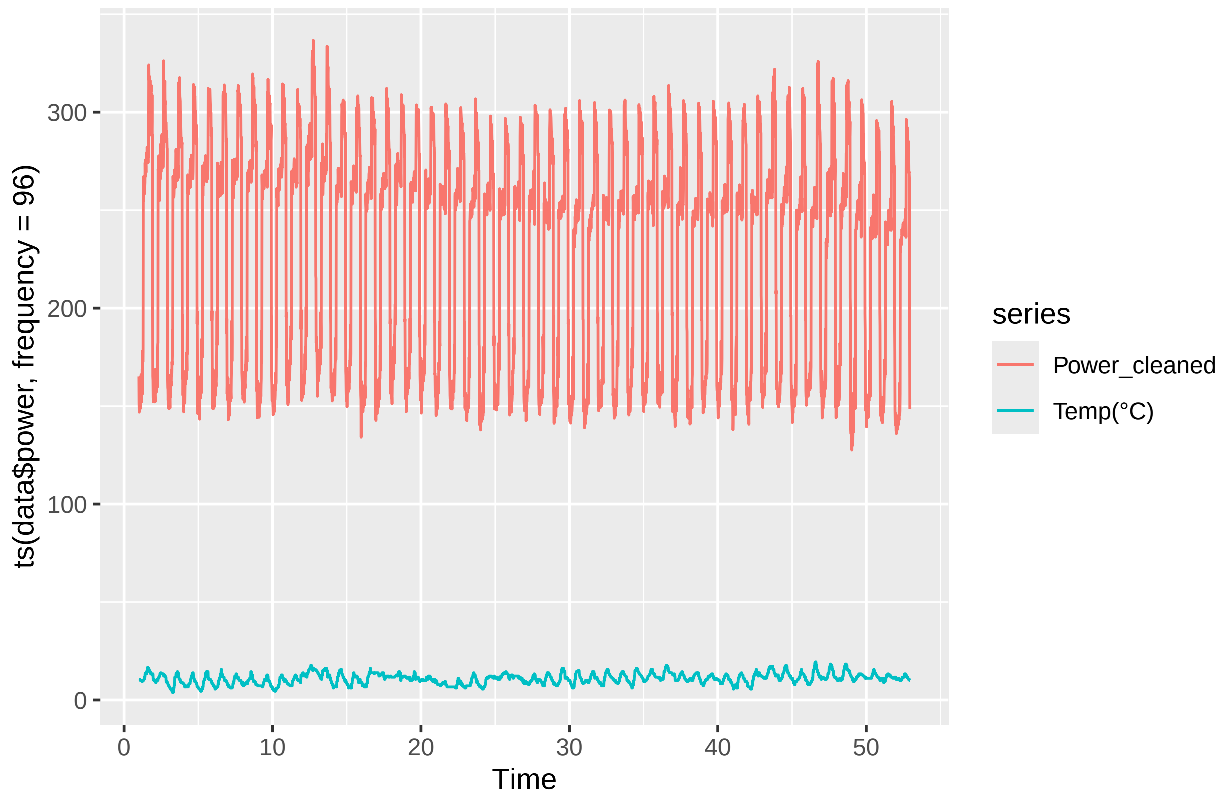
2. Forecast based on electricity consumption and outdoor temperature

We will use a dynamic regression model for forecasting electricity demand, using temperature as an external covariate. The order of the ARIMA model for the residual part is automatically selected

```
power <- ts(data$`Power (kW)`, frequency = 96)
power_clean<-tsclean(power) # to clean the outliers
data$power<-power_clean
data
```

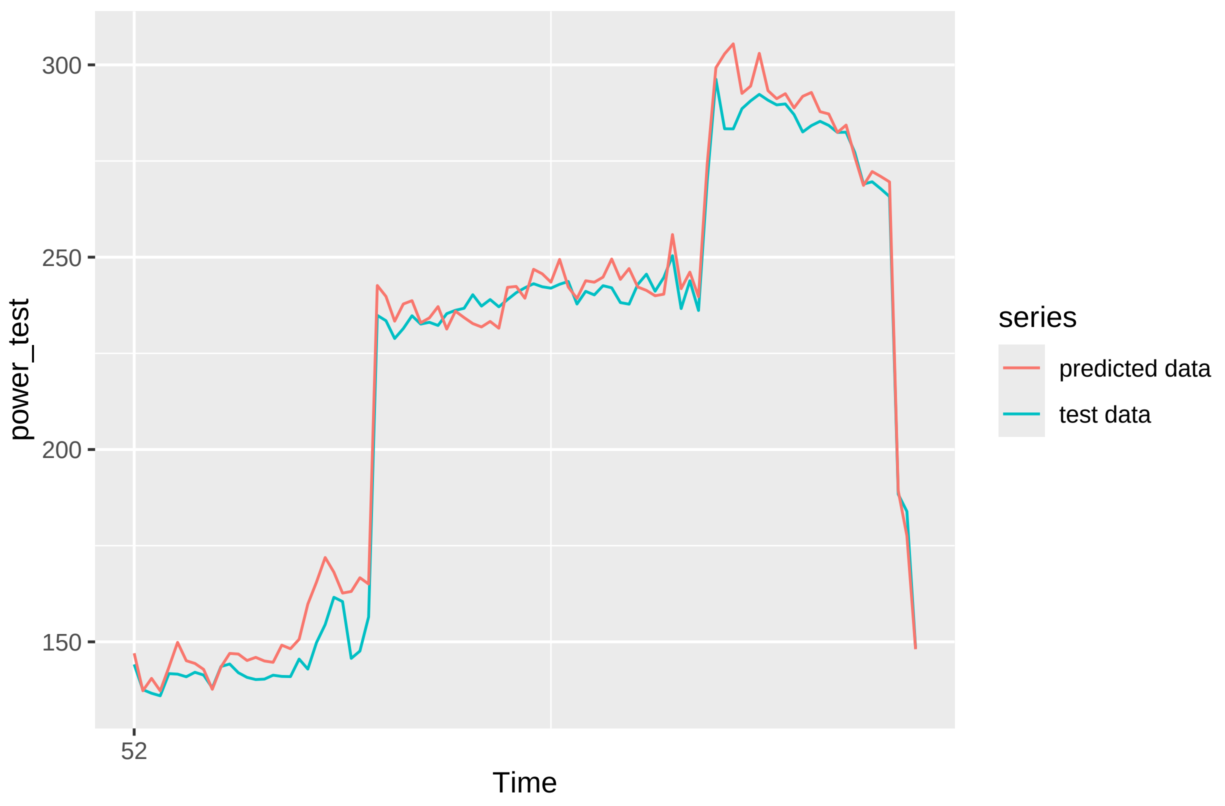
```
## # A tibble: 4,987 x 4
##    Timestamp           `Power (kW)` `Temp (C°)` power
##    <dttm>                     <dbl>       <dbl> <dbl>
##  1 2010-01-01 01:15:00         165.        10.6  165.
##  2 2010-01-01 01:30:00         152.        10.6  152.
##  3 2010-01-01 01:45:00         147.        10.6  147.
##  4 2010-01-01 02:00:00         154.        10.6  154.
##  5 2010-01-01 02:15:00         154.        10.6  154.
##  6 2010-01-01 02:30:00         159         10.6  159
##  7 2010-01-01 02:45:00         158.        10.6  158.
##  8 2010-01-01 02:59:59         163.        10.6  163.
##  9 2010-01-01 03:14:59         152.        10    152.
## 10 2010-01-01 03:29:59         149.        10    149.
## # i 4,977 more rows
```

```
autoplot(ts(data$power, frequency = 96),series="Power_cleaned")+autolayer(ts(data$`Temp (C°)`, frequency
```



```
power_train=window(ts(data$power, frequency = 96),start=c(1,1),end=c(51,96))
power_test=window(ts(data$power, frequency = 96),start=c(52,1))

# Covariates : Temp
temp_train=window(ts(data$`Temp (C°)`, frequency = 96),start=c(1,1),end=c(51,96))
temp_test=window(ts(data$`Temp (C°)`, frequency = 96),start=c(52,1))
```

```
fit2=auto.arima(power_train,xreg=temp_train,seasonal = TRUE)
prev=forecast(fit2,h=96,xreg=temp_test)
autoplot(power_test,series="test data")+autolayer(prev$mean,series="predicted data")
```



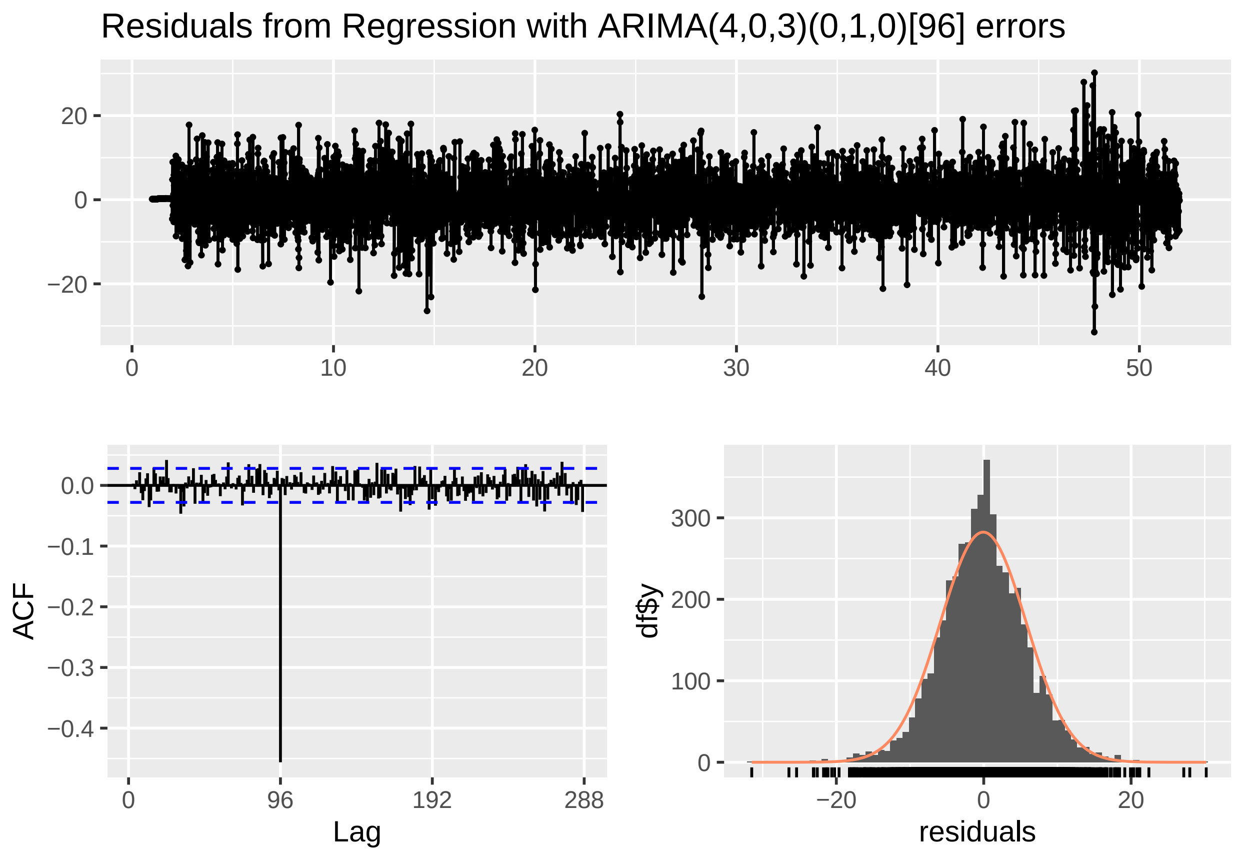
```
cat('RMSE :',sqrt(mean((prev$mean-power_test)^2)))
```

```
## RMSE : 6.711949
```

```
summary(fit2)
```

```
## Series: power_train
## Regression with ARIMA(4,0,3)(0,1,0)[96] errors
##
## Coefficients:
##          ar1     ar2     ar3      ar4     ma1      ma2      ma3    xreg
##       0.3854  0.2161  0.6326  -0.3177  0.1207  -0.0569  -0.5611  0.1669
## s.e.  0.0926  0.0751  0.0971   0.0305  0.0932   0.0761   0.0698  0.1331
##
## sigma^2 = 34.88:  log likelihood = -15331.84
## AIC=30681.69   AICc=30681.72   BIC=30739.97
##
## Training set error measures:
##                        ME     RMSE      MAE         MPE     MAPE      MASE
## Training set -0.06470033 5.842898 4.457474 -0.06620024 2.068663 0.7164286
##                      ACF1
## Training set 0.0003264559
```

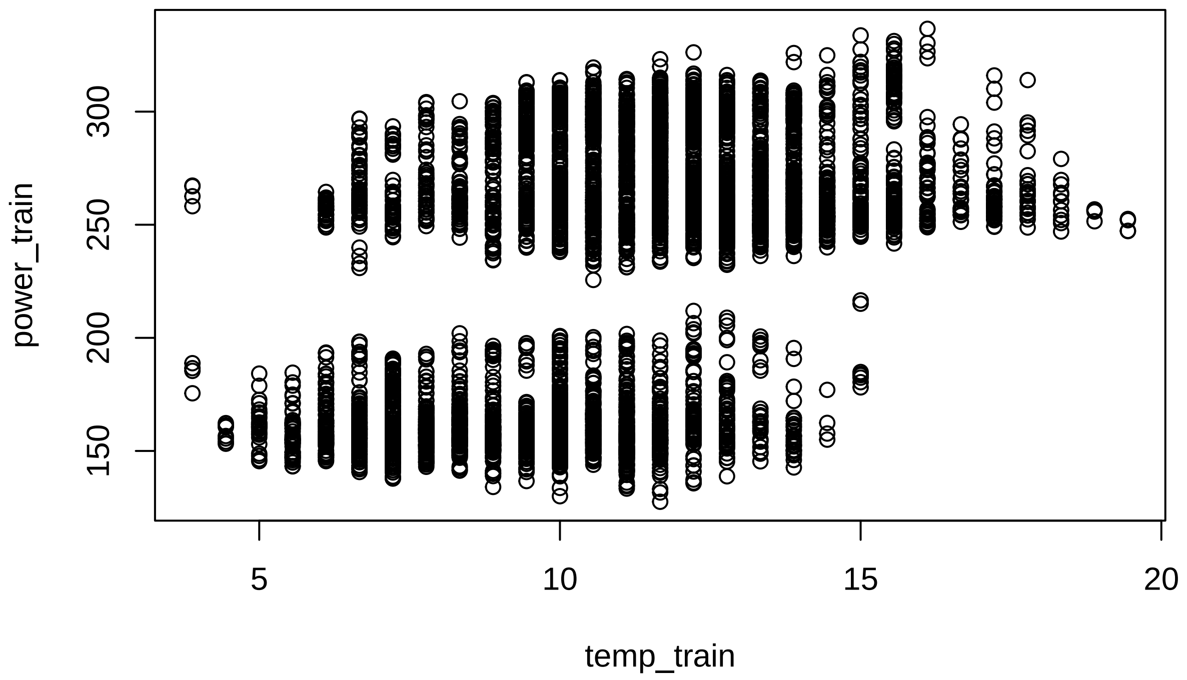
```
checkresiduals(fit2)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(4,0,3)(0,1,0)[96] errors
## Q* = 1321.7, df = 185, p-value < 2.2e-16
##
## Model df: 7.   Total lags used: 192
```

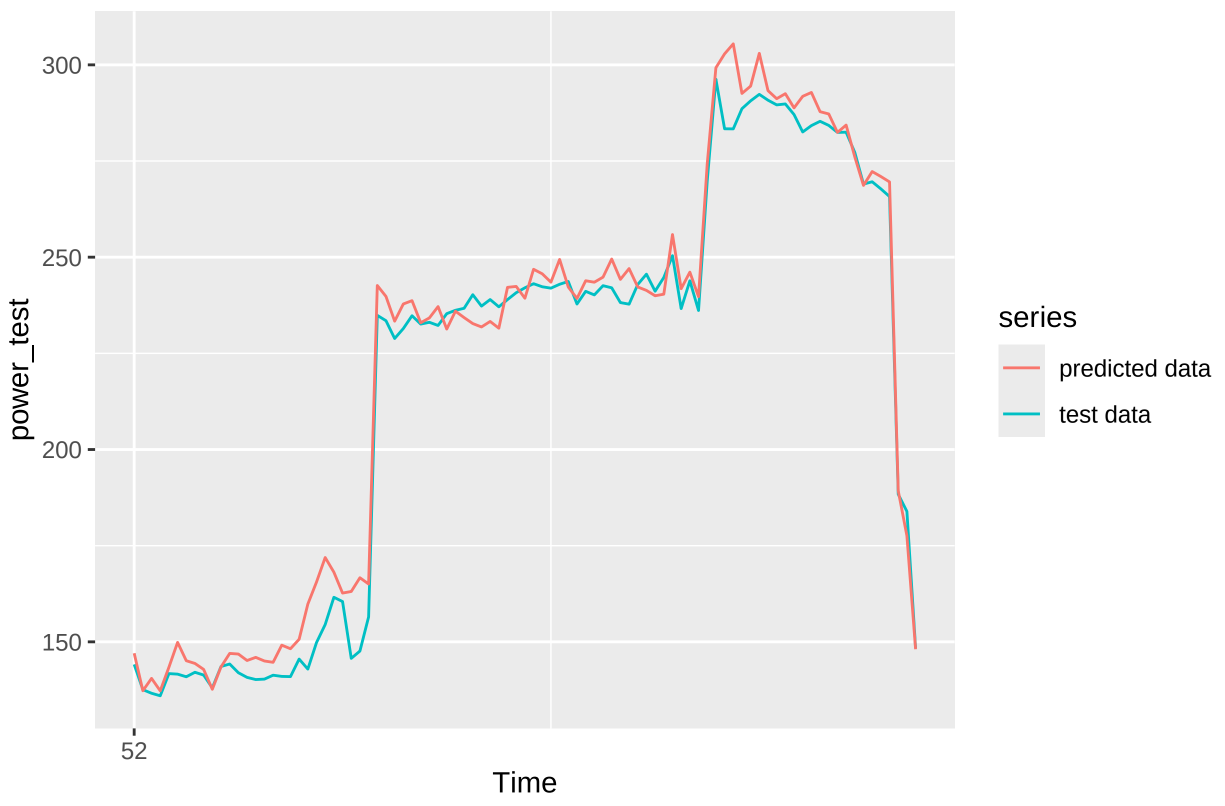
There is still some autocorrelations and the auto model only took the D as the seasonal part but doesn't have and AR or MA seasonal part. Let's have a look at the relationship between Power and Temperature

```
plot(temp_train,power_train)
```



There are 2 “bands” (day/night) + a non-linear relationship with temperature. As a result, a simple linear term Temp is unlikely to be significant.

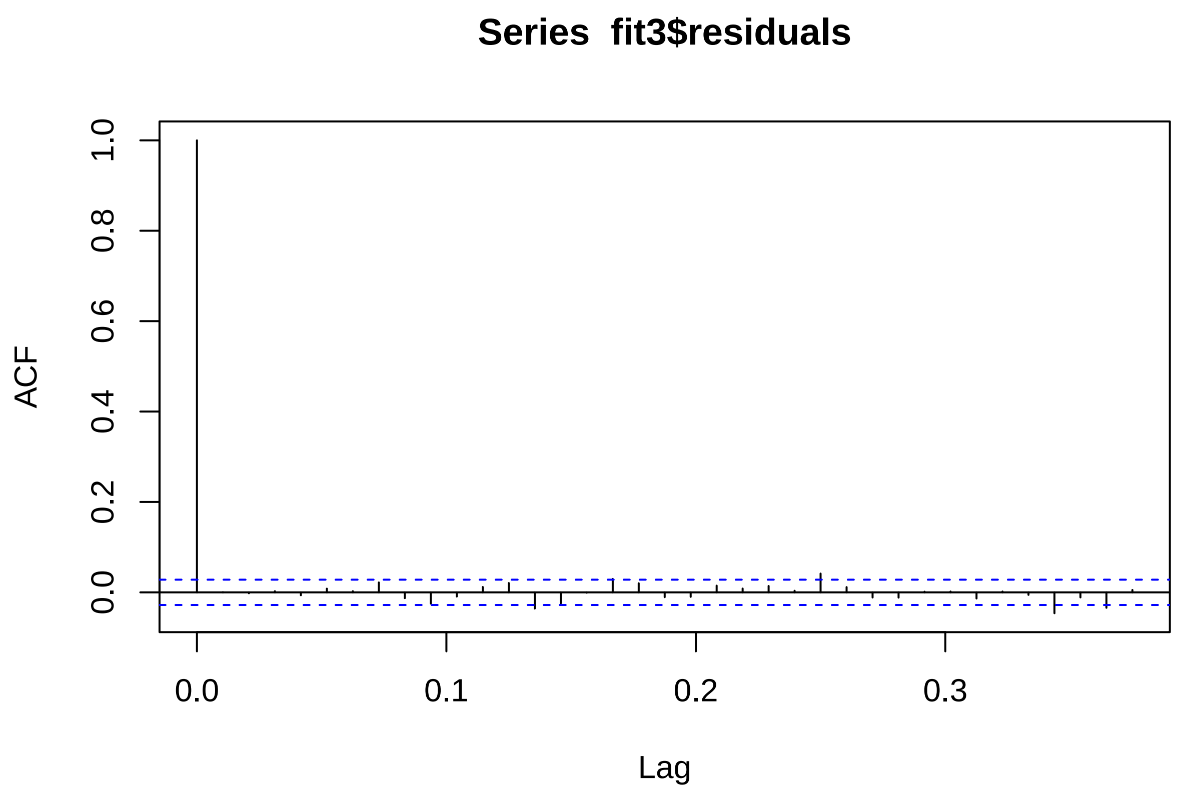
```
temp_ts  <- ts(data$`Temp (C°)`,   frequency = 96)
# Covariates : Temp + Temp^2
X_all    <- cbind(temp = temp_ts, temp2 = temp_ts^2)
temp_trainX <- window(X_all, start = c(1,1),  end = c(51,96))
temp_testX  <- window(X_all, start = c(52,1))

fit3=auto.arima(power_train,xreg=temp_trainX,seasonal = TRUE)
prev3=forecast(fit3,h=96,xreg=temp_testX)
autoplot(power_test,series="test data")+autolayer(prev$mean,series="predicted data")
```



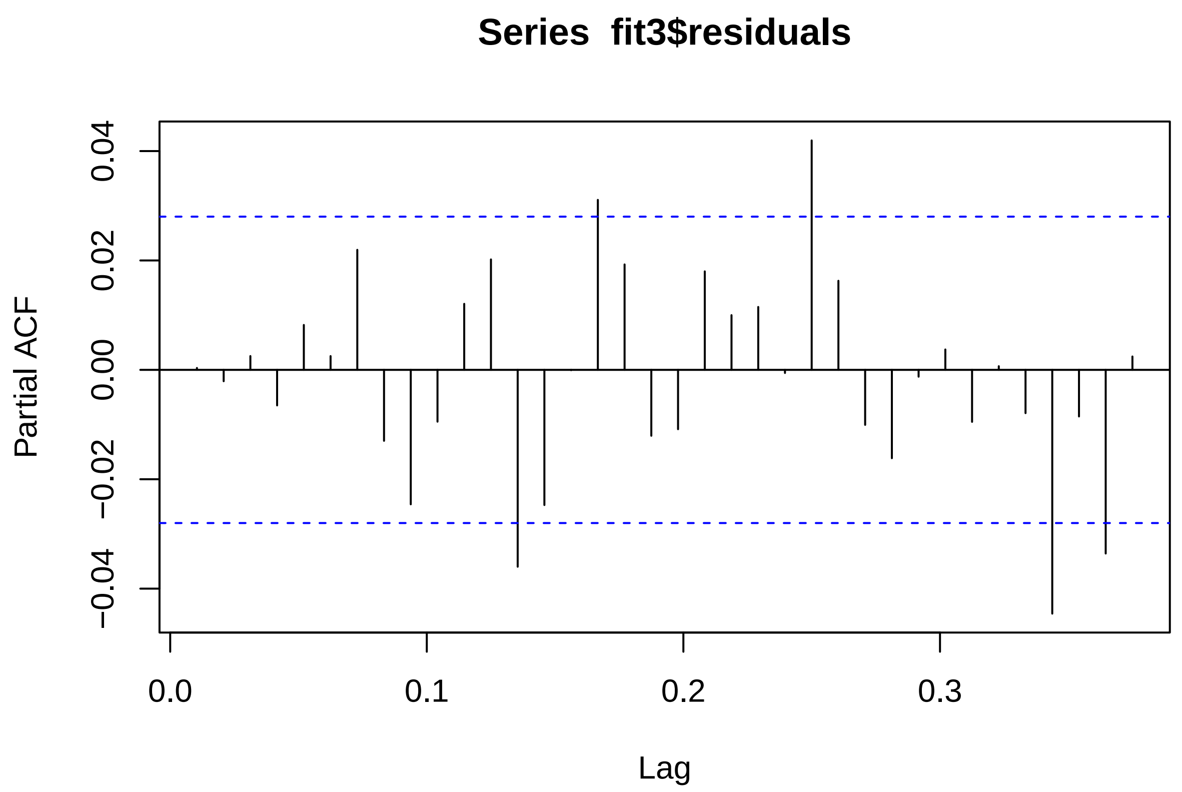
```
summary(fit3)
```

```
## Series: power_train
## Regression with ARIMA(4,0,3)(0,1,0)[96] errors
##
## Coefficients:
##          ar1     ar2     ar3      ar4     ma1      ma2      ma3     temp
##       0.3826  0.2191  0.6288  -0.3157  0.1231  -0.0590  -0.5573  -0.3253
## s.e.  0.0924  0.0759  0.0975   0.0308  0.0930   0.0769   0.0701   0.5179
##        temp2
##       0.0225
## s.e.  0.0227
##
## sigma^2 = 34.88:  log likelihood = -15331.29
## AIC=30682.57   AICc=30682.62   BIC=30747.33
##
## Training set error measures:
##                       ME     RMSE      MAE         MPE     MAPE      MASE
## Training set -0.06489539 5.842224 4.456723 -0.06636242 2.068391 0.7163078
##                     ACF1
## Training set 0.0003441555
```

```
acf(fit3$residuals)
```



```
pacf(fit3$residuals)
```



Not really satisfying, we try to do a manual model. After comparing the metrics, we keep the first forecast.

```r
df <- data.frame(
  date = time(prev$mean),
  value = as.numeric(prev$mean)
)

# Save in CSV
write.csv(df, "elec_2.csv", row.names = FALSE)
```