Time Series Analysis of Household Electric Consumption

Suzet Nkwaya

May 2022

```
library(TSstudio)
rm(list=ls())
```

Data Set Information:

The dataset is an archive that contains power consumption measurements gathered in a house located outside Paris, France between December 2006 and November 2010 (47 months). The dataset contains some missing values in the measurements (nearly 1,25% of the rows). All calendar time stamps are present but for some timestamps, the measurement values are missing. To fix this i imputed the missing values with the value that came right before. I believe this is an acceptable strategy because the measurements are collected every minute, and my assumption is that power consumption may vary on an monthly, hourly or daily basis and not from one minute to the next.

I am interested in seeing how power consumption changes throughout the day, and throughout the year. For the modeling part of the project I aggregated monthly data. Although it would have been interesting to look at hourly data, aggregation had the effect or reducing the size of the dataset and thus made computations much faster.

Some of the variables in the dataset are: * Date in format dd/mm/yyyy,time in format hh:mm:ss * global_active_power: household global minute-averaged active power (in kilowatt) * global_reactive_power: household global minute-averaged reactive power (in kilowatt) * voltage: minute-averaged voltage (in volt) * global_intensity: household global minute-averaged current intensity (in ampere) I the following analysis focus on **global_active_power**

Dataset Location: [Individual household electric power consumption] (https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014#)

Exploratory analysis

```
head(powerCons[,1:3])
```

```
##         date     time global_active_power
## 1 16/12/2006 17:24:00               4.216
## 2 16/12/2006 17:25:00               5.360
## 3 16/12/2006 17:26:00               5.374
## 4 16/12/2006 17:27:00               5.388
## 5 16/12/2006 17:28:00               3.666
## 6 16/12/2006 17:29:00               3.520
```

```
#find missing values
print(paste("There are ",sum(is.na(powerCons))," missing data points"))
```

```
## [1] "There are  25979  missing data points"
```

```
#add a new column to the data set which contains both the "Date" and "Time"
subpwr$datetime <- paste(subpwr$date,subpwr$time)
subpwr$datetime <-as.POSIXct(subpwr$datetime,"%Y/%m/%d %H/%M/%S")
subpwr$date <- date(subpwr$datetime)
subpwr$year <- year(subpwr$datetime)
subpwr$week <- week(subpwr$datetime)
subpwr$day <- day(subpwr$datetime)
subpwr$month <- month(subpwr$datetime)
subpwr$hour <- hour(subpwr$datetime)
subpwr$minute <- minute(subpwr$datetime)
```

Here is the data after it was aggregated by year and month

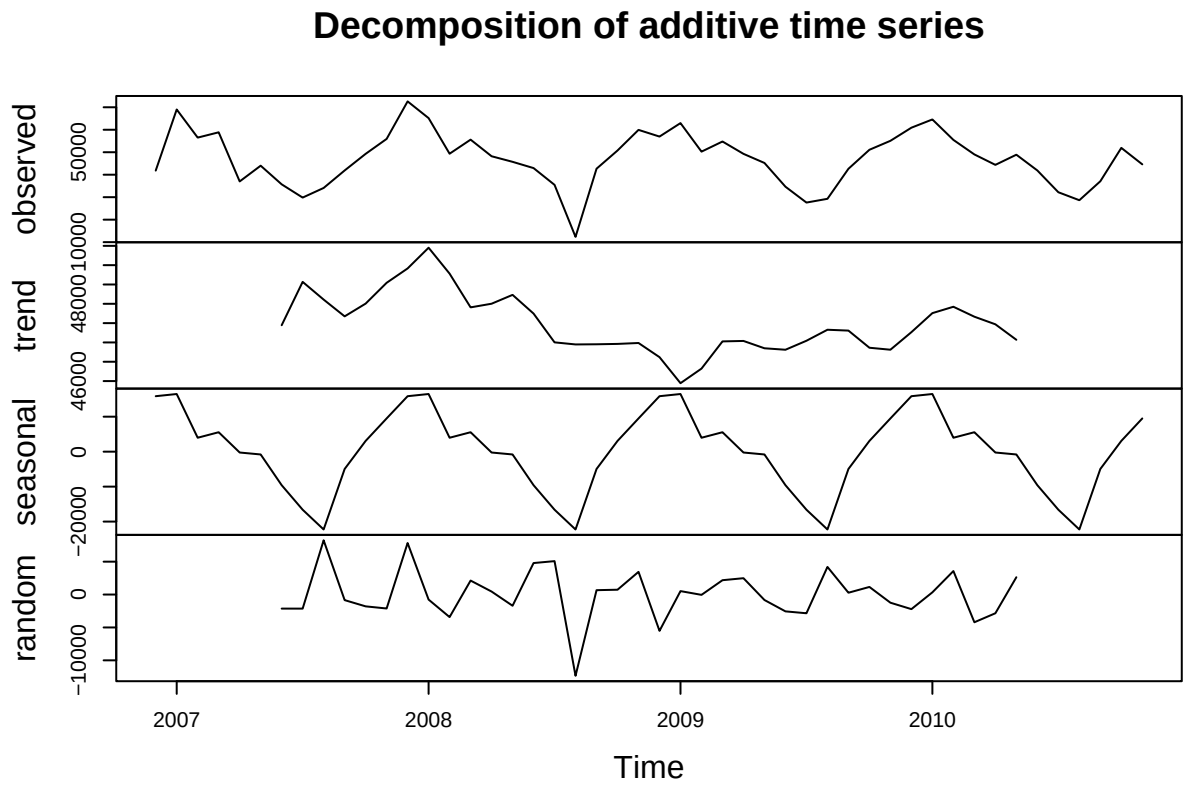
```
#the original data is collected every few minutes
#we'll transform it to be hourly

monthlydt <-aggregate(global_active_power~month+year,
                      subpwr,
                      FUN=sum)
head(monthlydt)
```

```
##   month year global_active_power
## 1    12 2006            41826.21
## 2     1 2007            69017.23
## 3     2 2007            56494.98
## 4     3 2007            58862.57
## 5     4 2007            37013.18
## 6     5 2007            44008.87
```

```
par(mfrow=c(1,1))
tsmonth <-ts(monthlydt[3],frequency=12,start=c(2006,12), end=c(2010,11))
```

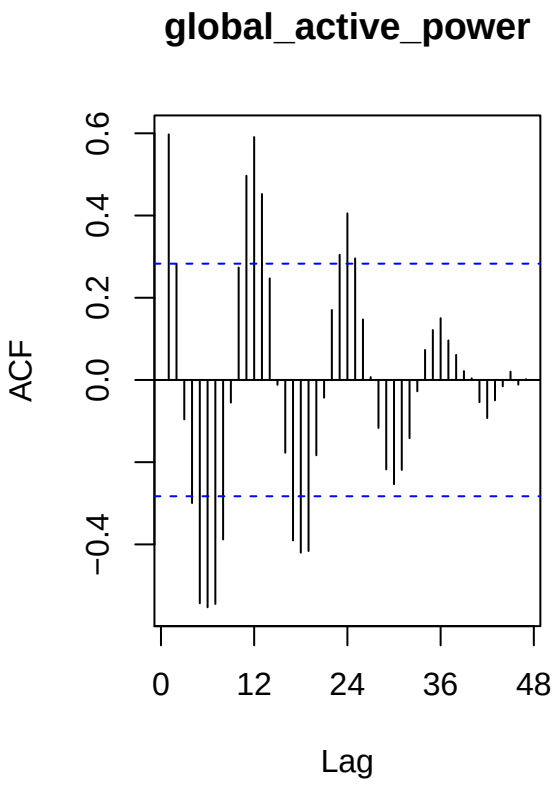
Next I decomposed the data into seasonal, trend and irregular components. And as i expected we can see a seasonal trend. This makes sense because more energy is used as the weather changes throughout the year.

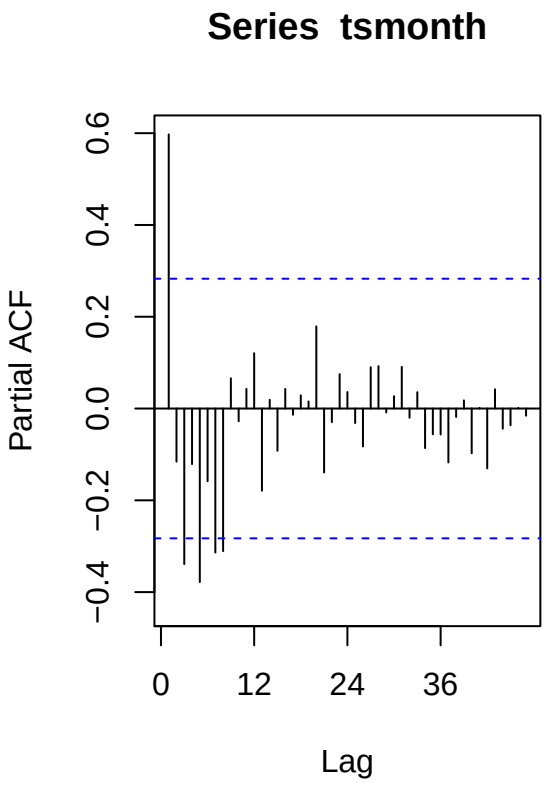
```
comp.tsmonth <-decompose(tsmonth)
plot(comp.tsmonth)
```



The data looks stationary, but we can look at the ACF and perform a Dickey-Fuller test to confirm

```
par(mfrow=c(1,2))
#acf(monthlydt$global_active_power, main="ACF Plot",length(tsmonth))
#pacf(monthlydt$global_active_power,main="PACF Plot",length(tsmonth))
Acf(tsmonth,lag.max = length(tsmonth))
Pacf(tsmonth,lag.max = length(tsmonth))
```






```
#Augmented Dickey-Fuller Test
adf.test(monthlydt$global_active_power)
```

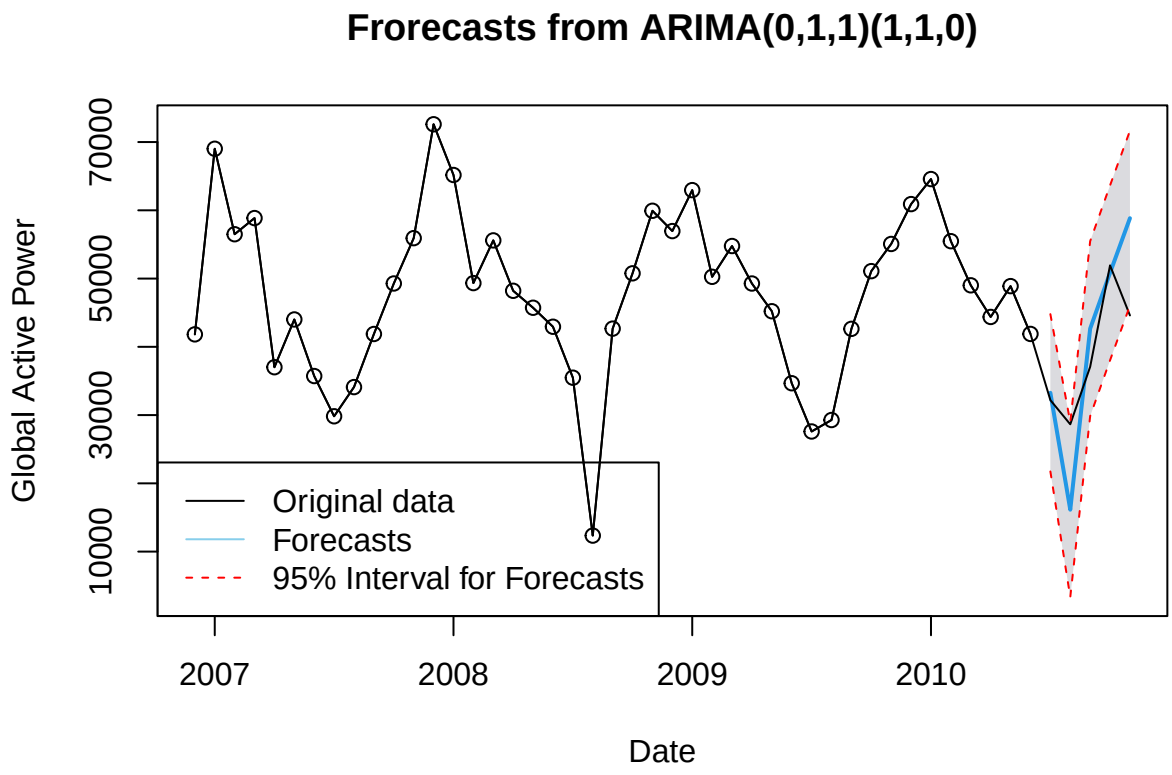
```
##
##  Augmented Dickey-Fuller Test
##
## data:  monthlydt$global_active_power
## Dickey-Fuller = -3.8333, Lag order = 3, p-value = 0.02436
## alternative hypothesis: stationary
```

looking at the ACF plot, it shows evidence of seasonality in power consumption because measurements that are 12 months apart tend to be strongly correlated. Additionally the Augmented Dickey-Fuller Test gave a pvalue that is less than 0.05, therefore we need to we reject the null hypothesis and conclude that the series is stationary

Modeling

Analysis using ARIMA

```r
set.seed(2022)
par(mfrow=c(1,1))
testsize=round(length(tsmonth)*0.1) # use 10% for testing
splitdata <- ts_split(ts.obj = tsmonth, sample.out = testsize)
training <- splitdata$train
testing <- splitdata$test
model.arima <- auto.arima(training, ic="aic",
                          seasonal=T,
                          allowmean=F,
                          allowdrift=F)
# Forecasting 5 months with test dataset.
arima.forecast <- forecast(model.arima,level = 95,
                           h=testsize)
plot(arima.forecast, type="o",ylab="Global Active Power",xlab="Date", main="Frorecasts f
lines(tsmonth)
lines(arima.forecast$lower,col="red",lty=2)
lines(arima.forecast$upper,col="red",lty=2)
legend("bottomleft",col = c("black","skyblue", "red"),
       lty = c(1, 1,2),
       legend = c("Original data","Forecasts","95% Interval for Forecasts"))
```



Using the auto arima function I got my best model to be ARMIMA(0,1,1)(1,1,1). This is a moving average model with 1 seasonal difference component, and 1 seasonal moving average

```
acc <- accuracy(arima.forecast,testing)
acc[,c(1,2,4,5,7)]
```

```
##                       ME     RMSE        MPE      MAPE       ACF1
## Training set   -79.58447 4828.878 -1.8453630  8.008056  0.1819898
## Test set     -1461.04821 8866.204 -0.9169137 19.226805 -0.2497215
```

```
#checkresiduals(arima.forecast)
```

The ARIMA model produced decent predictions judging by the accuracy metrics below and the plot above. It is important to note that the prediction interval would get wider as we try to predict values that are further in the future

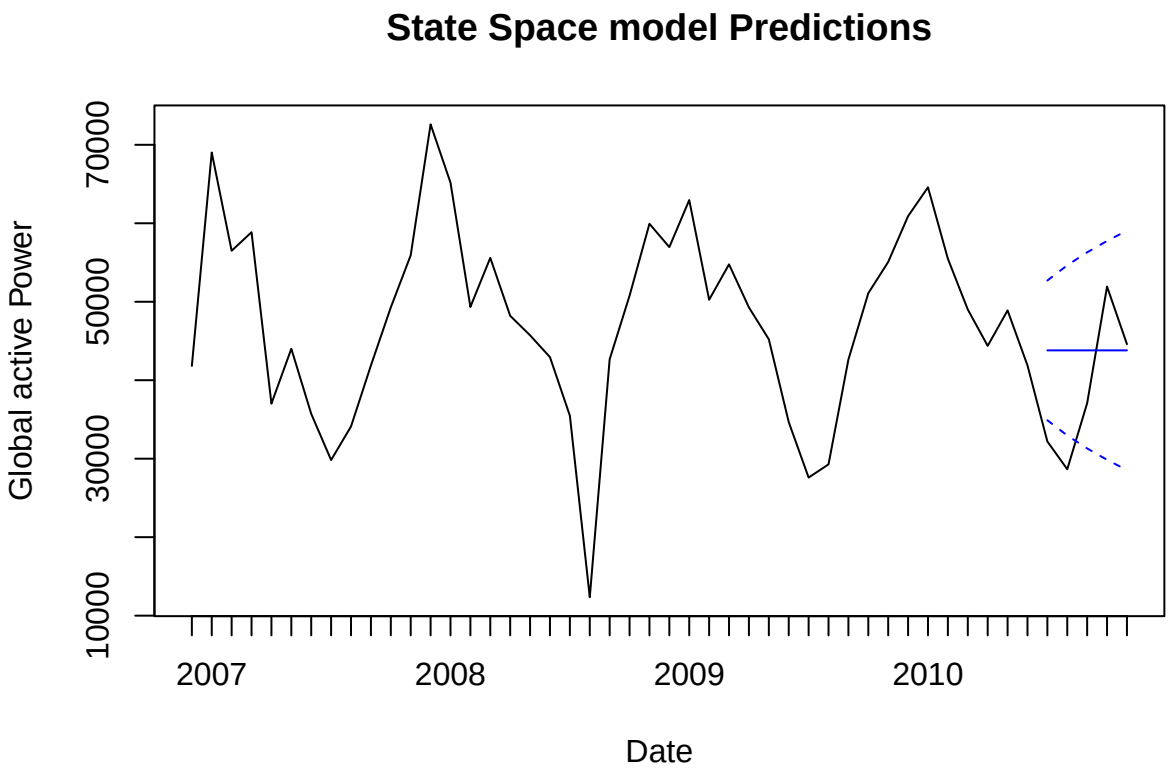
Analysis using State space model

```
model <- list(
  B=array(1, dim = c(1, 1, n)), U=matrix(0), Q=matrix("sig.sq.w"),
  Z=array(1, dim = c(1, 1, n)), A=matrix(0), R=matrix("sig.sq.v"),
  x0=matrix("mu"), tinitx=1 )

# Use EM to get starting values for direct maximization
#fit <- MARSS(c(y.marss), model=model, method = "kem")
# Direct maximization starting at EM starting values
fit <- MARSS(c(y.marss), model = model, method = "BFGS",silent=FALSE)
```

```
## Success! Converged in 444 iterations.
## Function MARSSkfas used for likelihood calculation.
##
## MARSS fit is
## Estimation method: BFGS
## Estimation converged in 444 iterations.
## Log-likelihood: -531.5882
## AIC: 1069.176   AICc: 1069.792
##
##            Estimate
## R.sig.sq.v  6264140
## Q.sig.sq.w  9999997
## x0.mu         41858
```

```
## Initial states (x0) defined at t=1
##
## Standard errors have not been calculated.
## Use MARSSparamCIs to compute CIs and bias estimates.
```

```
forc <- fit$ytT # E[y_t | y_1,...,y_m]
forc.se <- fit$ytT.se

#plot(tsmonth,col = "gray")
plot(monthlydt2$fdate,monthlydt2$global_active_power, type="l",xlab="Date",ylab="Global

lines(monthlydt2$fdate[sub.pred],forc[sub.pred], col = "blue",type="l")

lines(monthlydt2$fdate[sub.pred],forc[sub.pred] +
      forc.se[sub.pred]*qnorm(0.025),
     col = "blue",
      lty = 2,type="l")

lines(monthlydt2$fdate[sub.pred],forc[sub.pred]+
      forc.se[sub.pred]*qnorm(0.975),
      col = "blue",
      lty = 2,type="l")
```



The State space model did not do a good job a predicting future power consumption. The AIC for the ARIMA model is 641 and for the State Space model the AIC is 1069. Therefore I would chose the ARIMA model. Because the number of observations is large enough the AIC and AICc become similar because AICc converges to AIC, so used AIC