Deep analytics and visualization of electric submetering for home development

Background: This work was conducted for a local home developer; the goals were I. to determine the kind of analytics and visualizations that can be applied to electric submeters to empower new homeowners with a greater understanding & control over their power usage and II. To build forecasting models for predicting power usage. To accomplish this, a data set containing 47 months of energy consumption data collected every minute, was analyzed and visualized. The data set was recorded and provided by the home developer via Amazon repository and obtained with a SQL query. First, research on our client's home development business, electric submetering, and preliminary exploration of the data was conducted. An initial project presentation and introductory meeting was then scheduled and conducted with the client. Following the introductory meeting, the data is prepared using multiple visualizations, analyzed by the creation of time series analysis for each submeter. Finally, linear, seasonal adjustment, and exponential smoothing were applied for predictive modeling of energy consumption. This report shows how data collected by electric submeters can be used and visualized to empower the home owner regarding their energy consumption.

Data set information:

1. The data was collected over 47 month period between the years of 2006 &2010 from a home in France and represent 3 submeters within the home with readings recorded every minute.
2. The data was provided by the home developer via Amazon repository, it was accessed using password protection.
3. Submeter locationwith in the home:
   - Submeter 1 – Kitchen (dishwasher, oven and microwave oven)
   - Submeter 2 – Laundry room (washing machine, dryer, refrigerator & light)
   - Submeter 3 – Utility room (Electric water heater & AC units)

Data Science: Rcode

```
#obtaining data using RMySQL
library(RMySQL)

# create a database connection
con = dbConnect(MySQL(), user='deepAnalytics', password='Sqltask1234!', dbname='dataanalytics2018',
host='data-analytics-2018.cbrosir2cswx.us-east-1.rds.amazonaws.com')

# List the data tables contained in data base
dbListTables(con)


# list attributes contained in the year 2006 data table
dbListFields(con, '2006')
```

# Down load data for project work: use dbGet Query function to download the 2006 thru 2010 data tables. Include Date, Time, sub_metering_1, sub_metering_2, and the sub_metering_3 attributes

```
dbListTables(con)

yr_2006 <- dbGetQuery(con, "SELECT Date, Time, Sub_metering_1,
            Sub_metering_2, Sub_metering_3 FROM yr_2006")
yr_2007 <- dbGetQuery(con, "SELECT Date, Time, Sub_metering_1,
            Sub_metering_2, Sub_metering_3 FROM yr_2007")
yr_2008 <- dbGetQuery(con, "SELECT Date, Time, Sub_metering_1,
            Sub_metering_2, Sub_metering_3 FROM yr_2008")
yr_2009 <- dbGetQuery(con, "SELECT Date, Time, Sub_metering_1,
            Sub_metering_2, Sub_metering_3 FROM yr_2009")
yr_2010 <- dbGetQuery(con, "SELECT Date, Time, Sub_metering_1,
            Sub_metering_2, Sub_metering_3 FROM yr_2010")
```

# investigate down loaded data use str, summary & head

```
str(yr_2006)
summary(yr_2006)
head(yr_2006)
tail(yr_2006)
```

```
str(yr_2006)
'data.frame':	21992 obs. of  5 variables:
 $ Date          : chr  "2006-12-16" "2006-12-16" "2006-12-16" "2006-12-16" ...
 $ Time          : chr  "17:24:00" "17:25:00" "17:26:00" "17:27:00" ...
 $ Sub_metering_1: num  0 0 0 0 0 0 0 0 0 0 ...
 $ Sub_metering_2: num  1 1 2 1 1 2 1 1 1 2 ...
 $ Sub_metering_3: num  17 16 17 17 17 17 17 17 17 16 ...
```

```
str(yr_2007)
summary(yr_2007)
head(yr_2007)
tail(yr_2007)
```

```
summary(yr_2007)
    Date               Time           Sub_metering_1   Sub_metering_2   Sub_metering_3
Length:521669      Length:521669      Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
Class :character   Class :character   1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000
Mode  :character   Mode  :character   Median : 0.000   Median : 0.000   Median : 0.000
                                      Mean   : 1.232   Mean   : 1.638   Mean   : 5.795
                                      3rd Qu.: 0.000   3rd Qu.: 1.000   3rd Qu.:17.000
                                      Max.   :78.000   Max.   :78.000   Max.   :20.000
```

```
str(yr_2008)
summary(yr_2008)
head(yr_2008)
tail(yr_2008)
```

```
head(yr_2008)
        Date     Time Sub_metering_1 Sub_metering_2 Sub_metering_3
1 2008-01-01 00:00:00              0              0             18
2 2008-01-01 00:01:00              0              0             18
3 2008-01-01 00:02:00              0              0             18
4 2008-01-01 00:03:00              0              0             18
5 2008-01-01 00:04:00              0              0             18
6 2008-01-01 00:05:00              0              0             17
```

```
str(yr_2009)
summary(yr_2009)
head(yr_2009)
tail(yr_2009)

str (yr_2010)
summary(yr_2010)
head(yr_2010)
tail(yr_2010)

# combine data frames that contain data for a Full year (use dplyr)
library(dplyr)
comb_data <- bind_rows(yr_2007, yr_2008, yr_2009)

#investigate new data frame
str(comb_data)
summary(comb_data)
head(comb_data)
tail(comb_data)
```

I. preprocessing the data for visualization and preliminary study

In order for R to allow complete and appropriate data analysis and visualization the date and time attributes must be converted to the right format and new attributes must be created to divide the time data into desired time periods.

Thus in in this step the :

1. Date and Time attributes were combined, in a new attribute, prior to converting them to the necessary format to allow for time series analysis
2. The new attribute was then named "DateTime" and moved to the front of the data frame
3. A timezone "Europe/Paris" was assigned to the DateTime attribute.
4. The DateTime attribute data was formatted to POSIXct for timeseries analysis.
5. New attributes were created by year, quarter, month, week, weekday, day,hour, and minute

```
# combining Data and time in a new attribute in  order to convert them
#to correct format
comb_data <-cbind(comb_data,paste(comb_data$Date,comb_data$Time), stringsAsFactors=FALSE)

# Name the new attribute (its in the 6th column)
colnames(comb_data)[6] <-"DateTime"
```

## Move the DateTime attribute within the dataset

```
yourdataframe <- yourdataframe[,c(ncol(yourdataframe), 1:(ncol(yourdataframe)-1))]
head(yourdataframe)

comb_data <- comb_data[,c(ncol(comb_data), 1:(ncol(comb_data)-1))]
head(comb_data)
```

# Convert DateTime to POSIXct time format

```
yourdataframe$DateTime <- as.POSIXct(yourdataframe$DateTime, "%Y/%m/%d %H:%M:%S")

comb_data$DateTime <- as.POSIXct(comb_data$DateTime, "%Y/%m/%d %H:%M:%S")
# add time zone
attr(comb_data$DateTime, "tzone") <- "Europe/Paris"
```

# inspect data types

```
str(comb_data)
```

```
str(comb_data)
'data.frame':	1569894 obs. of  15 variables:
 $ DateTime      : POSIXct, format: "2007-01-01 01:00:00" ...
 $ Date          : chr  "2007-01-01" "2007-01-01" "2007-01-01" "2007-01-01" ...
 $ Time          : chr  "00:00:00" "00:01:00" "00:02:00" "00:03:00" ...
 $ Sub_metering_1: num  0 0 0 0 0 0 0 0 0 0 ...
 $ Sub_metering_2: num  0 0 0 0 0 0 0 0 0 0 ...
 $ Sub_metering_3: num  0 0 0 0 0 0 0 0 0 0 ...
 $ year          : num  2007 2007 2007 2007 2007 ...
 $ quarter       : int  1 1 1 1 1 1 1 1 1 1 ...
 $ month         : num  1 1 1 1 1 1 1 1 1 1 ...
 $ week          : num  1 1 1 1 1 1 1 1 1 1 ...
 $ weekdays      : chr  "Monday" "Monday" "Monday" "Monday" ...
 $ wday          : num  2 2 2 2 2 2 2 2 2 2 ...
 $ day           : int  1 1 1 1 1 1 1 1 1 1 ...
 $ hour          : int  1 1 1 1 1 1 1 1 1 1 ...
 $ minute        : int  0 1 2 3 4 5 6 7 8 9 ...
```

# use lubrdate package to create new attributes from DateTime attribute
# these new attributes may be used to subset the data using dplyr

```
library(lubridate)
```

# create year attribute with lubridate

```
yourdataframe$year <- year(yourdataframe$DateTime)
comb_data$year <- year(comb_data$DateTime)

comb_data$quarter <- quarter(comb_data$DateTime)
comb_data$month <- month(comb_data$DateTime)
comb_data$week <- week(comb_data$DateTime)
comb_data$weekdays <- weekdays(comb_data$DateTime)
comb_data$wday <- wday(comb_data$DateTime)
comb_data$day <- day(comb_data$DateTime)
comb_data$hour <- hour(comb_data$DateTime)
comb_data$minute <- minute(comb_data$DateTime)

summary(comb_data)
```

# Preliminary statistical study
To obtain statistical information the summary() function was used to provide, the mean, mode, quartiles and characterization of the distribution. The sd() function and the sum () function were used to obtain standard deviation and the total power consumption of each submeter respectively. See chart and table below.

```
#Display summary statistics of comb_data$Sub_metering_1.
summary(comb_data$Sub_metering_1)
#Display standard deviation of comb_data$Sub_metering_1.
sd(comb_data$Sub_metering_1, na.rm = FALSE)

#Display summary statistics of comb_data$Sub_metering_2.
summary(comb_data$Sub_metering_2)
#Display standard deviation of comb_data$Sub_metering_2.
sd(comb_data$Sub_metering_2, na.rm = FALSE)

#Display summary statistics of comb_data$Sub_metering_3.
summary(comb_data$Sub_metering_3)
#Display standard deviation of comb_data$Sub_metering_2.
sd(comb_data$Sub_metering_3, na.rm = FALSE)

#determine total consumption of energy in watt-hour per meter
sum(comb_data$Sub_metering_1)
sum(comb_data$Sub_metering_2)
sum(comb_data$Sub_metering_3)
```

Desc. Stat	Kitchen watt- hour	Laundry Room watt- hour	Utility Room watt-hour
Minimum	0.0	0.0	0.0
1st quarter	0.0	0.0	0.0
Median	0.0	0.0	1.0
Mode	0.0	0.0	0.0
Mean	1.159	1.343	6.216
3rd quarter	0.0	1.0	17.0
Maximum	82.00	78.00	31.0
total	1829989	2108410	9758845
STD	6.29	5.97	8.34

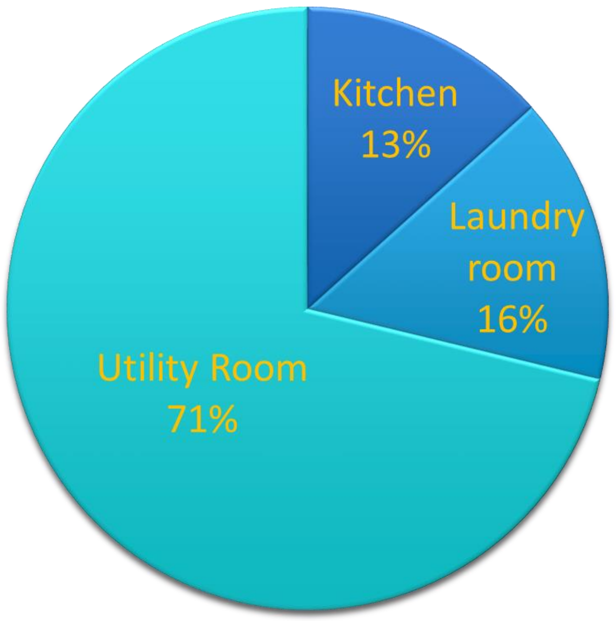


Figure1: Table of descriptive statistic of energy consumption in watt-hours for years 2007-2009. The chart describes the percentage of total energy consumption per meter location.

## Deep analysis and visualizations: subset, reduce granularity, and visualize
To study the granularity of the data a subset of the data is studied. In this case the second week of 2008 is chosen, followed by the 9the day of 2008.

# Subset the second week of 2008- All observations using dplyr
week2_2008 <- filter(comb_data, year==2008 & week ==2)
# plot week2_2008 submeter 1
plot (week2_2008$Sub_metering_1)

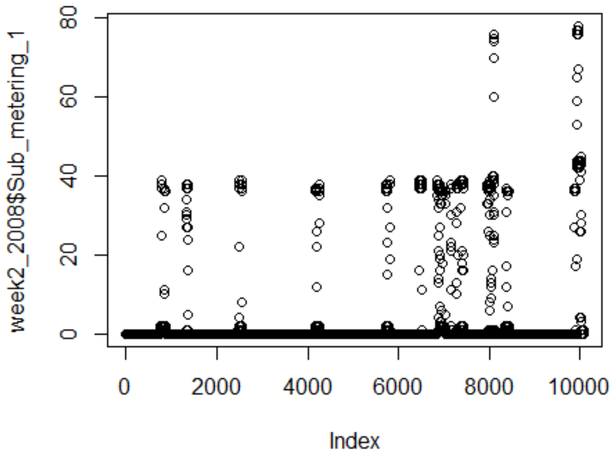


Figure 2: From looking at the data for one week in 2008 it appears that power consumption in this house peak during the end of the week.

# Subset the 9th day of 2008 all observations
#subset day 9 of the year 2008
day9_2008 <- filter(comb_data, year==2008 & month==1 & day ==9)
# plot day9_2008 submeter 1
library(plotly)
plot_ly(day9_2008, x= ~day9_2008$DateTime, y= ~day9_2008$Sub_metering_1,
    type = 'scatter', mode= 'lines')

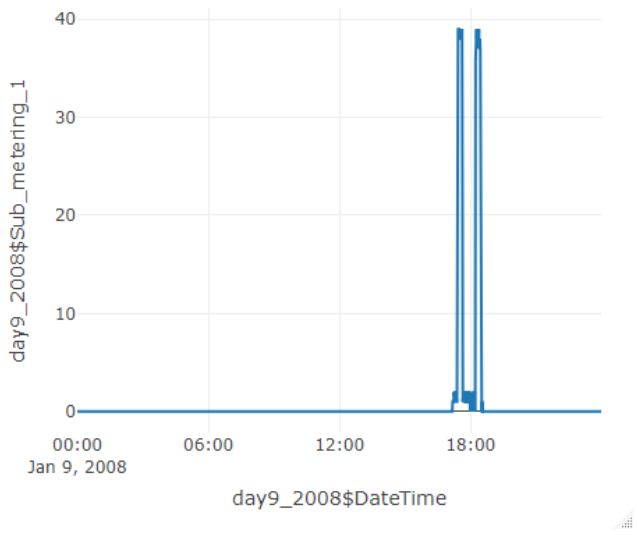


Figure 3: Day 9 of 2008 was subset and submeter 1 data plotted. It is clear that power consumption in submeter 1 peaks between 5 and 6 pm.

# plot submeter 1,2,and 3 of week 2 of 2008 using with title legends
# and labels. Use plotly

```
plot_ly(week2_2008, x = ~week2_2008$DateTime, y = ~week2_2008$Sub_metering_1, name =
'Kitchen', type = 'scatter', mode = 'lines') %>%
 add_trace(y = ~week2_2008$Sub_metering_2, name = 'Laundry Room', mode = 'lines') %>%
 add_trace(y = ~week2_2008$Sub_metering_3, name = 'Water Heater & AC', mode = 'lines') %>%
 layout(title = "Power Consumption Week 2, 2008",
     xaxis = list(title = "Date"),
     yaxis = list (title = "Power (watt-hours)"))
```

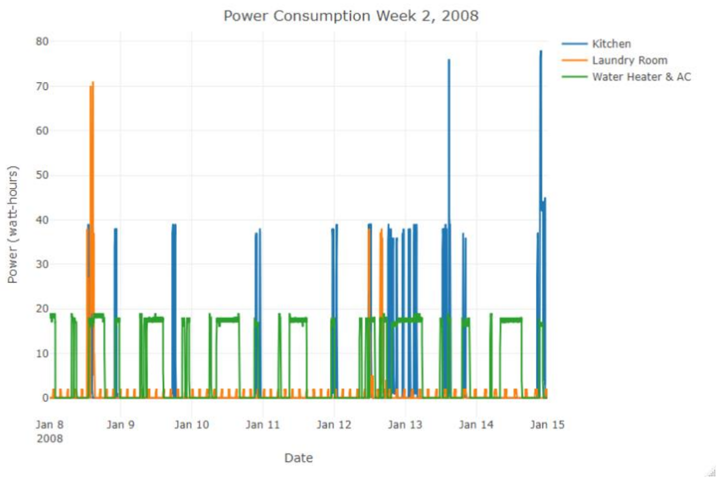


Figure 4: In this chart all three submeter data was plotted for the 2nd week of 2008. The plot gives an overview of power consumption for the week.

## Plot sub-meter 1, 2 and 3 of day 9 2008 with title, legend and labels - All observations

```
plot_ly(day9_2008, x = ~day9_2008$DateTime, y = ~day9_2008$Sub_metering_1, name = 'Kitchen', type
= 'scatter', mode = 'lines') %>%
 add_trace(y = ~day9_2008$Sub_metering_2, name = 'Laundry Room', mode = 'lines') %>%
 add_trace(y = ~day9_2008$Sub_metering_3, name = 'Water Heater & AC', mode = 'lines') %>%
 layout(title = "Power Consumption January 9th, 2008",
     xaxis = list(title = "Time"),
     yaxis = list (title = "Power (watt-hours)"))
```

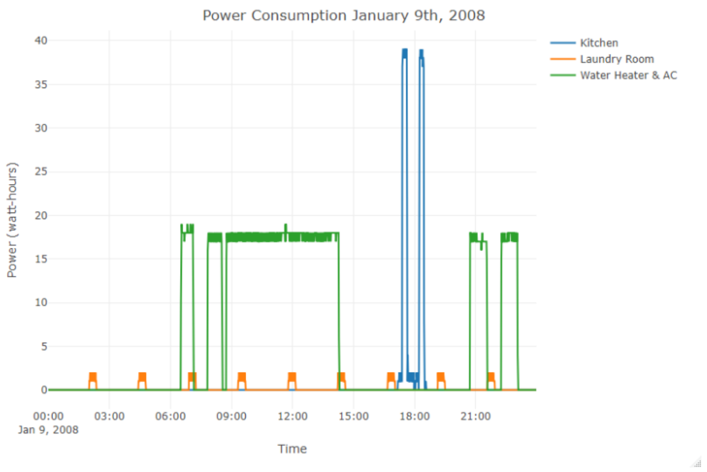


Figure 5: Recorded energy consumption for the 9th day of 2008. The plot gives an overview of power consumption for the day.

Kitchen power consumption peaked in the early evening, while the laundry room appears to be cyclic likely due to the turning on and off of the refrigerator. The water heater peaked in the early morning and around 9 pm with the AC running between the hours of 9 am to 3 pm.

# Reduce granularity in the plots by ploting in 10 minute intervals
# the data must be subset in 10 minute increments

```
day9_2008_10min <- filter(comb_data, year==2008 & month==1 &
                    day ==9 & (minute==0 | minute == 10
                    | minute == 20 | minute == 30 |
                    minute == 40 | minute == 50) )
```

## Plot sub-meter 1, 2 and 3 day9 2008 with title, legend and labels - 10 Minute frequency

```
plot_ly(day9_2008_10min, x = ~day9_2008_10min$DateTime, y = ~day9_2008_10min$Sub_metering_1,
name = 'Kitchen', type = 'scatter', mode = 'lines') %>%
  add_trace(y = ~day9_2008_10min$Sub_metering_2, name = 'Laundry Room', mode = 'lines') %>%
  add_trace(y = ~day9_2008_10min$Sub_metering_3, name = 'Water Heater & AC', mode = 'lines') %>%
  layout(title = "Power Consumption January 9th, 2008 10 Min Increments",
      xaxis = list(title = "Time"),
      yaxis = list (title = "Power (watt-hours)"))
```

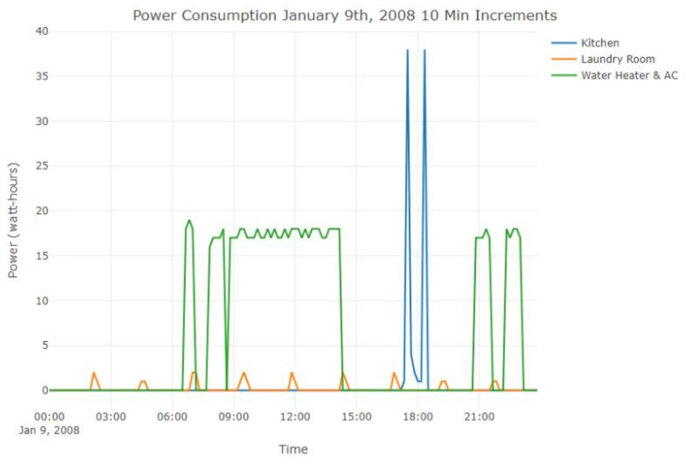


Figure 6: Recorded energy consumption for the 9th day of 2008 in **10 min intervals**. The plot gives an overview of power consumption for the day.

Kitchen power consumption peaked in the early evening, while the laundry room appears to be cyclic likely due to the turning on and off of the refrigerator. The water heater peaked in the early morning and around 9 pm with the AC running between the hours of 9 am to 3 pm.

Obtain statistical information regarding power consumption during the second week of 2008

```
#Display summary statistics of week2_2008$Sub_metering_1.
summary(week2_2008$Sub_metering_1)
#Display standard deviation of week2_2008$Sub_metering_1.
sd(week2_2008$Sub_metering_1, na.rm = FALSE)

#Display summary statistics of week2_2008$Sub_metering_2.
summary(week2_2008$Sub_metering_2)
#Display standard deviation of week2_2008$Sub_metering_2.
sd(week2_2008$Sub_metering_2, na.rm = FALSE)

#Display summary statistics of week2_2008$Sub_metering_3.
summary(week2_2008$Sub_metering_3)
#Display standard deviation of week2_2008$Sub_metering_3.
sd(week2_2008$Sub_metering_3, na.rm = FALSE)
```

#determine total consumption of energy in watt-hour per meter in week2_2008
sum(week2_2008$Sub_metering_1)
sum(week2_2008$Sub_metering_2)
sum(week2_2008$Sub_metering_3)

Desc. Stat	Kitchen watt- hour	Laundry Room watt- hour	Utility Room watt-hour
Minimum	0.0	0.0	0.0
1st quarter	0.0	0.0	0.0
Median	0.0	0.0	1.0
Mode	0.0	0.0	0.0
Mean	2.375	0.761	6.718
3rd quarter	0.0	1.0	17.0
Maximum	78.00	71	19.0
total	23942	7670	67714
STD	9.42	4.53	8.50

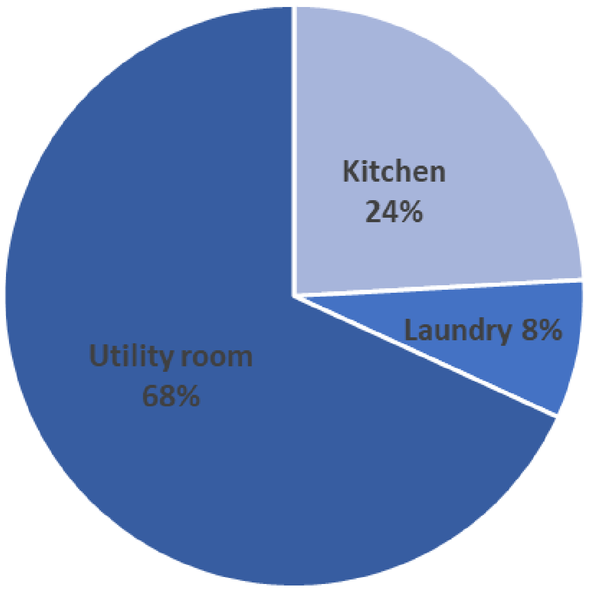


Figure 7: Table of descriptive statistic of energy consumption in watt-hours for of week 2 in 2008. The chart describes the percentage of total energy consumption per meter location.

# To continue to understand power consumption in this home analyses different week in 2008
# subset first week of May2008
may_2008_wk_1 <-filter(comb_data,year == 2008 & week == 18)

# plot submeter 1,2,and 3 of may 2008 week 1 with title legends
# and labels. Use plotly
```
plot_ly(may_2008_wk_1, x = ~may_2008_wk_1$DateTime, y = ~may_2008_wk_1$Sub_metering_1,
name = 'Kitchen', type = 'scatter', mode = 'lines') %>%
 add_trace(y = ~may_2008_wk_1$Sub_metering_2, name = 'Laundry Room', mode = 'lines') %>%
 add_trace(y = ~may_2008_wk_1$Sub_metering_3, name = 'Water Heater & AC', mode = 'lines') %>%
 layout(title = "Power Consumption May 2008 Week 1",
     xaxis = list(title = "Date"),
     yaxis = list (title = "Power (watt-hours)"))
```

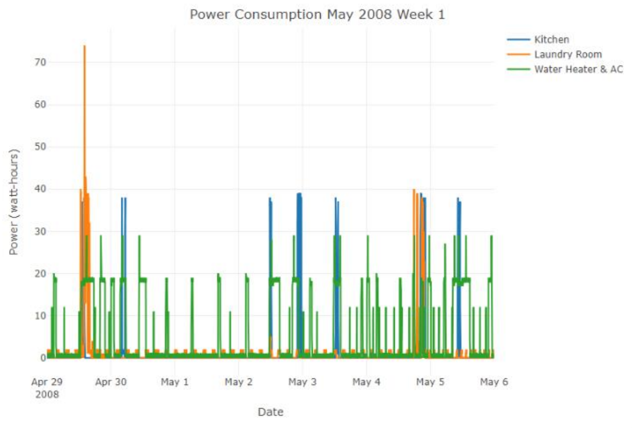


Figure 8: Data for all three submeters first week of May 2008. The plot gives an overview of power consumption for the week.

```
# analyze single day in week 1 may 2008. Reduce granularity in the plot by plotting in 10 minute
intervals. The data must be subset in 10 minute increments

May4_2008 <- filter(comb_data, year==2008 & month==5 & day ==4)

# plot submeter 1,2,and 3 of  may 4,2008  with title legends
# and labels. Use plotly
plot_ly(May4_2008, x = ~May4_2008$DateTime, y = ~May4_2008$Sub_metering_1, name = 'Kitchen',
type = 'scatter', mode = 'lines') %>%
  add_trace(y = ~May4_2008$Sub_metering_2, name = 'Laundry Room', mode = 'lines') %>%
  add_trace(y = ~May4_2008$Sub_metering_3, name = 'Water Heater & AC', mode = 'lines') %>%
  layout(title = "Power Consumption May 4,2008",
       xaxis = list(title = "Time"),
       yaxis = list (title = "Power (watt-hours)"))

# subset may 4 2008 to 10 min intervals
May4_2008_10min <- filter(comb_data, year==2008 & month==5 &
                         day ==4 & (minute==0 | minute == 10
                                | minute == 20 | minute == 30 |
                                 minute == 40 | minute == 50) )
# plot submeter 1,2,and 3 of  may 4,2008 10 min with title legends
# and labels. Use plotly
plot_ly(May4_2008_10min, x = ~May4_2008_10min$DateTime, y =
~May4_2008_10min$Sub_metering_1, name = 'Kitchen', type = 'scatter', mode = 'lines') %>%
  add_trace(y = ~May4_2008_10min$Sub_metering_2, name = 'Laundry Room', mode = 'lines') %>%
  add_trace(y = ~May4_2008_10min$Sub_metering_3, name = 'Water Heater & AC', mode = 'lines') %>%
  layout(title = "Power Consumption May 4,2008 10 min increments",
       xaxis = list(title = "Time"),
       yaxis = list (title = "Power (watt-hours)"))
```

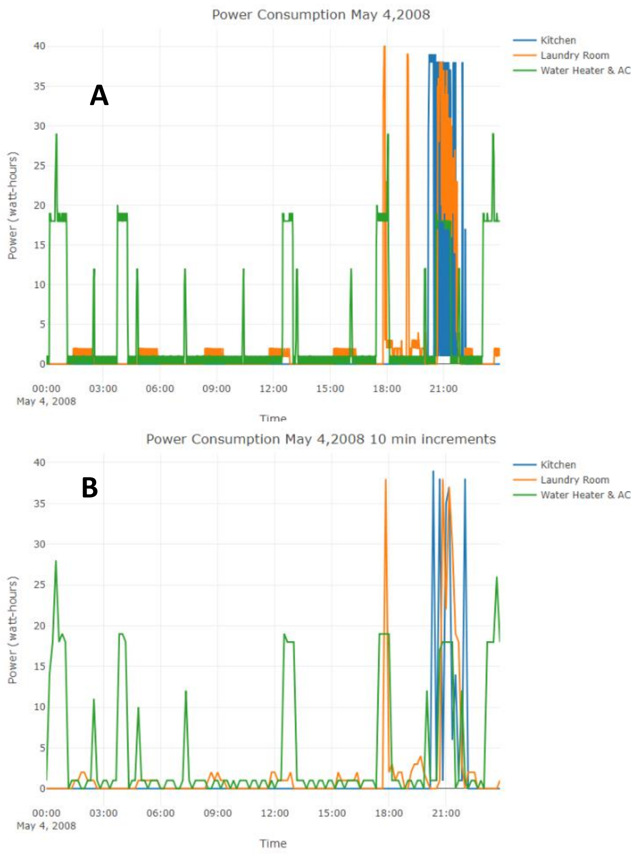


Figure 9: Recorded energy consumption for the 4th day of May 2008. **A**. is all recorded points The plot gives an overview of power consumption for the day. However, a bit busy for interpretation. **B.** is plotted with reduced granularity and in 10 minute intervals. The plot gives a much easier to understand overview of power consumption for the day. Kitchen power consumption peaked in the evening between 7 and 9 pm, while the laundry room appears to be cyclic likely due to the turning on and off of the refrigerator and peaking at 6 pm and again between 7 and 9 pm.

Obtain statistical information regarding power consumption during the first week May of 2008

```
#Display summary statistics of may 2008 week 1$Sub_metering_1.
summary(may_2008_wk_1$Sub_metering_1)
#Display standard deviation of may 2008 week 1 $Sub_metering_1.
sd(may_2008_wk_1$Sub_metering_1, na.rm = FALSE)

#Display summary statistics of may 2008 week 1$Sub_metering_2.
summary(may_2008_wk_1$Sub_metering_2)
#Display standard deviation of may 2008 week 1 $Sub_metering_2.
sd(may_2008_wk_1$Sub_metering_2, na.rm = FALSE)

#Display summary statistics of may 2008 week 1$Sub_metering_3.
summary(may_2008_wk_1$Sub_metering_3)
#Display standard deviation of may 2008 week 1 $Sub_metering_3.
sd(may_2008_wk_1$Sub_metering_3, na.rm = FALSE)

# determine total watt hours used per meter location in may 2008 week 1
sum(may_2008_wk_1$Sub_metering_1)
sum(may_2008_wk_1$Sub_metering_2)
sum(may_2008_wk_1$Sub_metering_3)

# determine total watt hours used per meter location on day 9 of 2008
# and may 4 2008
sum(day9_2008$Sub_metering_1)
sum(day9_2008$Sub_metering_2)
sum(day9_2008$Sub_metering_3)

sum(May4_2008$Sub_metering_1)
sum(May4_2008$Sub_metering_2)
sum(May4_2008$Sub_metering_3)
```

Desc. Stat	Kitchen watt- hour	Laundry Room watt- hour	Utility Room watt-hour
Minimum	0.0	0.0	0.0
1st quarter	0.0	0.0	1.0
Median	0.0	0.0	1.0
Mode	0.0	0.0	0.0
Mean	0.9695	1.087	5.02
3rd quarter	0.0	1.0	5.0
Maximum	39.00	74	29
total	9773	10962	50603
STD	5.67	4.88	7.6

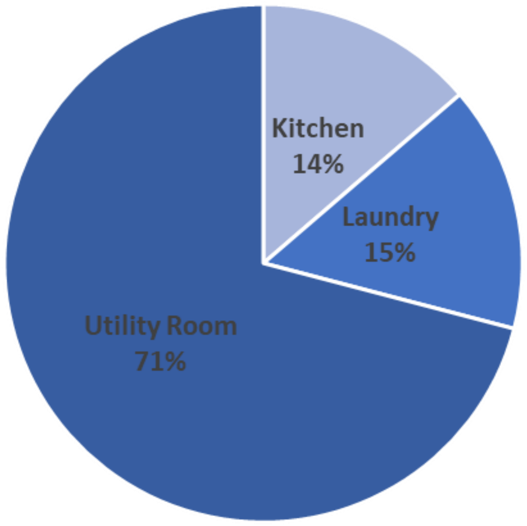


Figure10: Table of descriptive statistic of energy consumption in watt-hours for the first week of May in 2008. The chart describes the percentage of total energy consumption per meter location during this week.

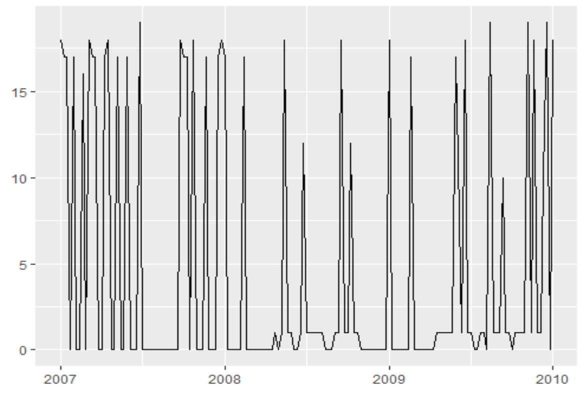
Desc. Stat	Kitchen watt- hour		Laundry Room watt- hour		Utility Room watt-hour		Total Watt-hours
Jan 9 2008	1162	11%	256	3%	8963	86%	**10381**
May 4 2008	2196	20%	2887	26%	6144	54%	**11227**

Figure 11: Table comparing energy consumption by location between January 9 and May 4 2008. It is clear that the highest energy use is in the Utility room containing the AC and water heater units. It is also evident that May 4 2008 may have been a laundry day in the household.

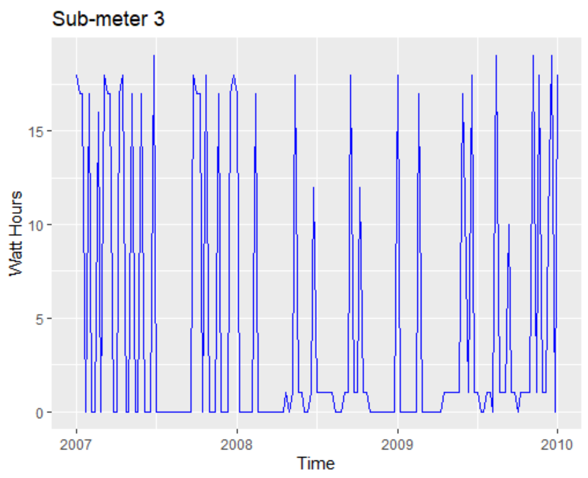
Time Series Analysis: In order to perform time series analysis a time series object must first be created. The ts() function is applied to create time series objects. In order to o study the data and gain new insights regarding energy use the data is filtered in multiple ways and multiple time series are created and plotted.

1. A three-year weekly time series with submeter 3 and one observation per week. Mondays at 8PM.
2. Additional time series were created to represent the third week in Sept. 2007 using submeters 2 &1. The week 37 time series were studied in 1, 5, 15, & 30-minute increments.
3. A one-day time series representing day 13 of the month in Septembers was also created and visualized in 1minute increments.

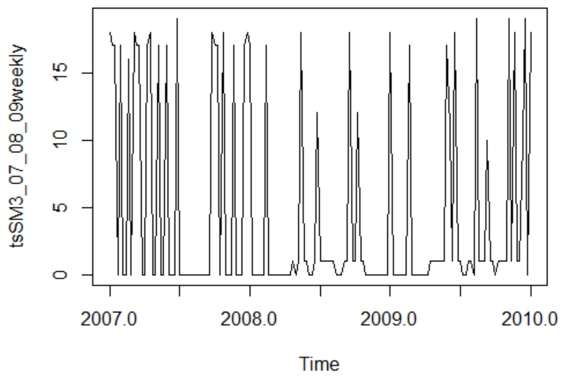
```
# prepare data time series for data analysis and predictions
# subset the data to one observation per week on Monday 8pm for 2007-2009
house07_08_09_weekly <- filter(comb_data, wday == 2 & hour == 20
                    & minute == 1)
# create and save a TS using one of the submeters. use submeter 3

tsSM3_07_08_09weekly <- ts(house07_08_09_weekly$Sub_metering_3,
                  frequency = 52, start = c(2007,1))

# plot sub-meter 3 with autoplot
library(ggplot2)
library(ggfortify)
autoplot(tsSM3_07_08_09weekly)
```



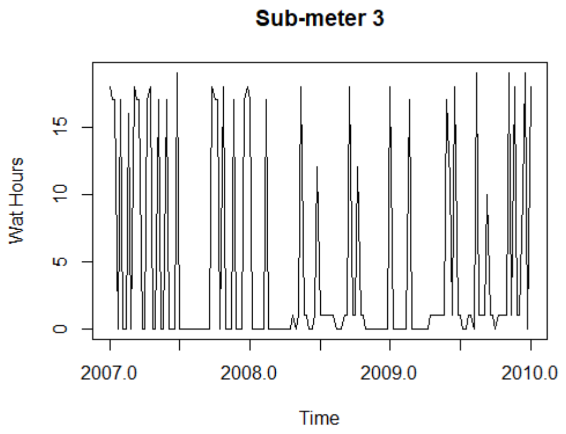
```
#plot sub-meter 3 with autoplot-use labels and color
autoplot(tsSM3_07_08_09weekly, ts.colour = 'blue', xlab = "Time",
     ylab = "Watt Hours", main = "Sub-meter 3")
```



# use plot.ts to plot sub-meter 3
plot.ts(tsSM3_07_08_09weekly)



# use plot.ts to plot sub-meter 3 add labels
plot.ts(tsSM3_07_08_09weekly, xlab = "Time",
ylab= "Wat Hours", main = "Sub-meter 3")



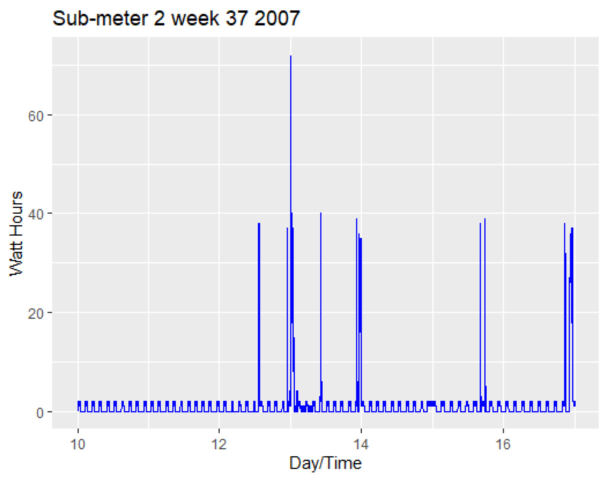
# to reduce the number of column omit the Time attribute. Also some time stamps are missing.
na.omit(comb_data$Time)

#Prepare one week time series for analysis
# subset week 37 of year 2007
house2007_3rd_weekSept <-filter(comb_data, year == 2007 & week == 37)

#create and save a time series of all data (one minute increments) using submeter 2 for one week
tsSM2_2007_week37_all <-ts(house2007_3rd_weekSept$Sub_metering_2,
frequency = 1440, start = c(10,1), end = c(16,1440))

# plot submeter2 for week37_all 2007using autoplot-with labels and color

```
autoplot(tsSM2_2007_week37_all, ts.colour = 'blue', xlab = "Day/Time",
     ylab = "Watt Hours", main = "Sub-meter 2 week 37 2007")
```

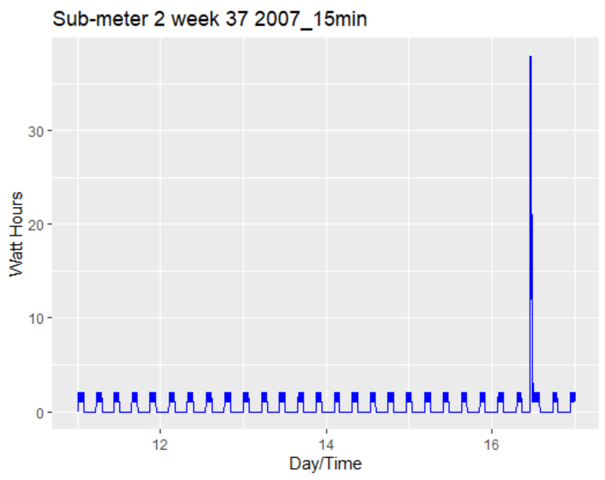


# create and save time series in 15 min increments using one of the submeters. use submeter 2

```
tsSM2_2007_week37_15min <- ts(house2007_3rd_weekSept$Sub_metering_2,
                    frequency = 672, start = c(11,0:0), end = c(16,672))
```

# plot submeter2 for week37 2007using autoplot-with labels and color

```
autoplot(tsSM2_2007_week37_15min, ts.colour = 'blue', xlab = "Day/Time",
     ylab = "Watt Hours", main = "Sub-meter 2 week 37 2007_15min")
```

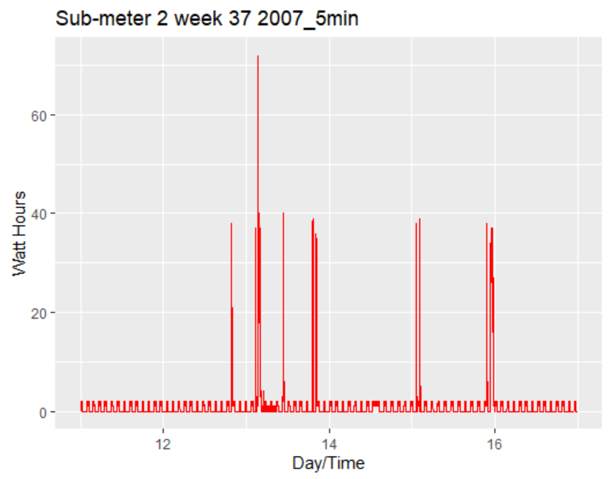


# create and save time series in 5 min increments using one of the submeters. use submeter 2

```
myts <- ts(house2007_3rd_weekSept$Sub_metering_2,
        frequency = 24*60*7/5, start = c(11,0:0), end = c(16,2016))
```

# plot submeter2 for week37 2007using autoplot-with labels and color

```
autoplot(myts, ts.colour = 'red', xlab = "Day/Time",
       ylab = "Watt Hours", main = "Sub-meter 2 week 37 2007_5min")
```



# Prepare one day time series for analysis
# subset September 13 2007

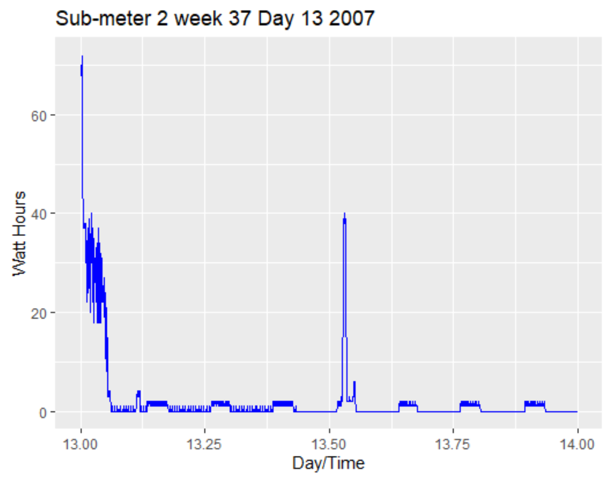
```
house2007_week37_D13 <-filter(comb_data, year==2007 & month==9 & day== 13)
```

# create and save time series using all data for week 37 day 13 using one of the submeters. use submeter 2

```
tsSM2_2007_week37_D13_all <- ts(house2007_week37_D13$Sub_metering_2,
                    frequency = 1140, start = c(13,0:0),
                    end = c(13,1140))
```

# plot submeter2 for week37 2007using autoplot-use labels and color

```
autoplot(tsSM2_2007_week37_D13_all, ts.colour = 'blue', xlab = "Day/Time",
       ylab = "Watt Hours", main = "Sub-meter 2 week 37 Day 13 2007")
```

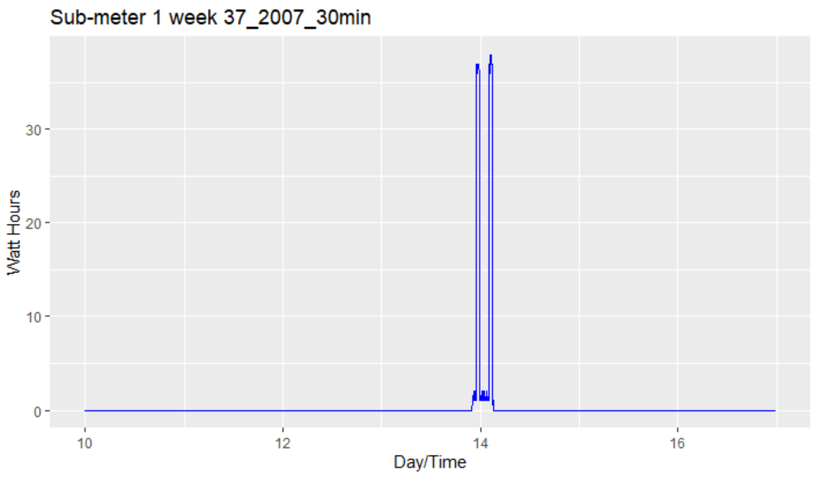


# create a time series using submeter 1 for week 37 of year 2007 in 30 min increments.

```
myts2<- ts(house2007_3rd_weekSept$Sub_metering_1,
         frequency = 24*60*7/30, start = c(10,0:0), end = c(16, 336))
```

# plot time series for submeter 1 week 37

```
autoplot(myts2, ts.colour = 'blue', xlab = "Day/Time",
       ylab = "Watt Hours", main = "Sub-meter 1 week 37_2007_30min")
```



II. Using time series data to predict and forecast the future using tslm and forecast functions

Three models were for predicting the future were created and used to forecast energy use in submeters 1, 2, and 3. The forecasts were predicted on a weekly, 15 minute, and 30 min intervals. A comparison chart showing R2, and RMSE for each model is also included.
To better understand energy consumption and forecasting trend and seasonality were also analyzed. Here the time series was decomposed into its components and seasonality was removed for a more accurate forecast.

## Apply time series linear regression to the sub-meter 3 ts object and use summary to #obtain R2 and RMSE from the model you built

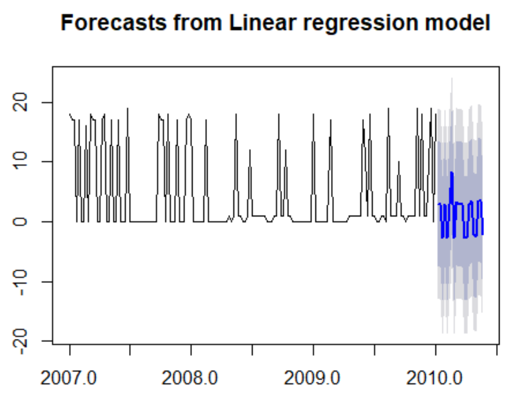
```
library(forecast)
fitSM3 <- tslm(tsSM3_07_08_09weekly ~ trend + season)
summary(fitSM3)
```

## Create the forecast for sub-meter 3. Forecast ahead 20 time periods

```
forecastfitSM3 <- forecast(fitSM3, h=20)
```

## Plot the forecast for sub-meter 3.

```
plot(forecastfitSM3)
```



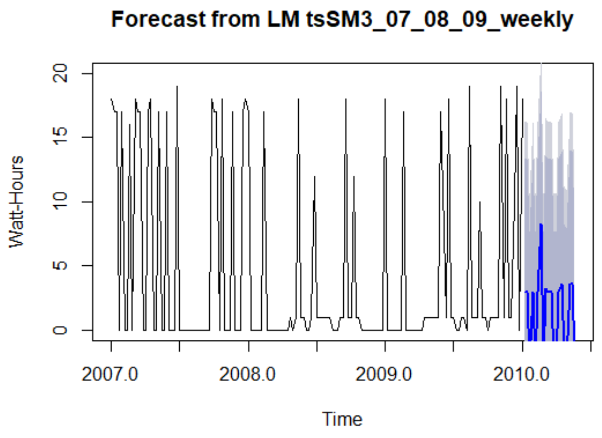
# change confidence level to forecast

## Create sub-meter 3 forecast with confidence levels 80 and 90

```
forecastfitSM3c <- forecast(fitSM3, h=20, level=c(80,90))
```

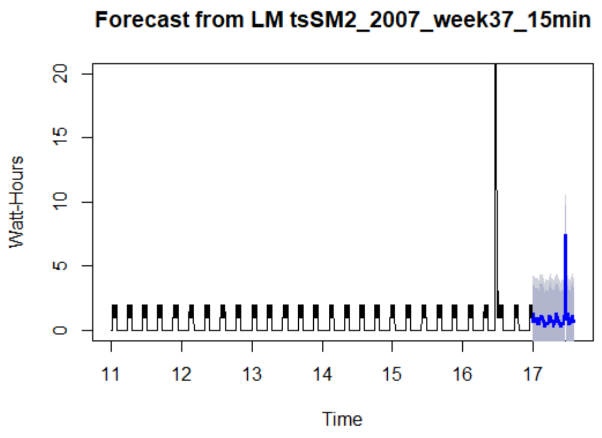
## Plot sub-meter 3 forecast, limit y and add labels

```
plot(forecastfitSM3c, ylim = c(0, 20), ylab= "Watt-Hours", xlab="Time", main = "Forecast from LM tsSM3_07_08_09_weekly")
```



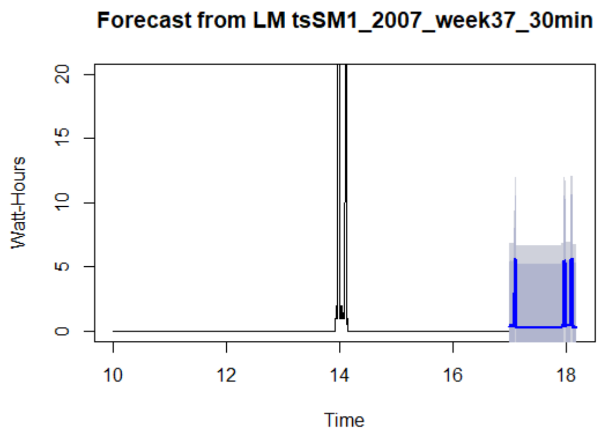
## Apply TS linear reg to submeter 2 & Create the forecast for sub-meter 2.
#Forecast ahead 400 time periods confidence level 80,90)

```
fitSM2 <- tslm(tsSM2_2007_week37_15min ~ trend + season)
summary(fitSM2)
forecastfitSM2 <- forecast(fitSM2, h=400, level = c(80,90))
plot(forecastfitSM2, ylim = c(0,20), ylab="Watt-Hours", xlab="Time", main= "Forecast from LM tsSM2_2007_week37_15min")
```



## Apply TS linear reg to submeter 1 & Create the forecast for sub-meter 1.
#Forecast ahead 400 time periods confidence level 80,90)

```
fitSM1 <- tslm(myts2 ~ trend + season)
summary(fitSM1)
forecastfitSM1 <- forecast(fitSM1, h=400, level = c(80,90))
plot(forecastfitSM1, ylim = c(0,20), ylab="Watt-Hours", xlab="Time", main= "Forecast from LM
tsSM1_2007_week37_30min")
```



Model name	R2	RMSE	Time interval
Fitsm3	0.3831	6.871	Weekly
Fitsm2	0.1788	1.898	15 minutes
Fitsm1	0.1336	3.675	30 minutes

Summary statistics for the forecast predictive models applied.

#Decomposition of a time series to see seasonality competent

To understand the trend of a timeseries independent of the seasonal component, the sesonal component was removed from the time series. To accomplish visualization of the TS component sthe decompose function from the forecast package was used.

## Decompose Sub-meter 3 into trend, seasonal and remainder

components070809SM3weekly <- decompose(tsSM3_07_08_09weekly)

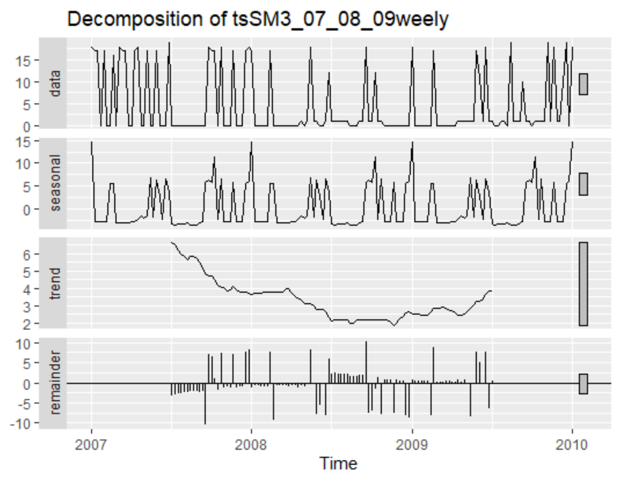
## Plot decomposed sub-meter 3

plot(components070809SM3weekly)

## Check summary statistics for decomposed sub-meter 3

summary(components070809SM3weekly)

autoplot(components070809SM3weekly, labels = NULL, main= "Decomposition of tsSM3_07_08_09weely", range.bars = NULL)



Decomposition of weekly (Mondays 8pm from 2007 – 2009) time series for submeter 3 (water heater & AC). The trend component shows a slight decrease in energy use from 2007 to the end 2008 and increase from 2008 thru 2009.

Summary of time series componets:

```
         Length Class  Mode
Data      157    ts     numeric
seasonal 157    ts     numeric
trend     157    ts     numeric
random    157    ts     numeric
figure     52    -none- numeric
type        1    -none- character
```

## Decompose Sub-meter 2 into trend, seasonal and remainder

components_tsSM2_2007_wk37_15min <- decompose(tsSM2_2007_week37_15min)

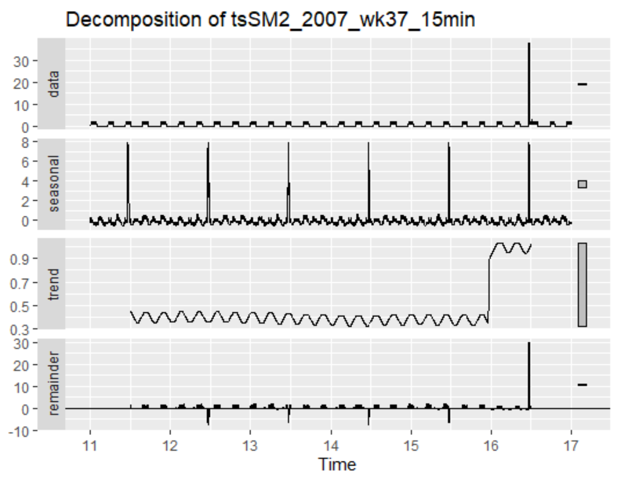
## Plot decomposed sub-meter 3

plot(components_tsSM2_2007_wk37_15min)

autoplot(components_tsSM2_2007_wk37_15min, labels = NULL, main= "Decomposition of tsSM2_2007_wk37_15min", range.bars = NULL)

## Check summary statistics for decomposed sub-meter 3

summary(components_tsSM2_2007_wk37_15min)

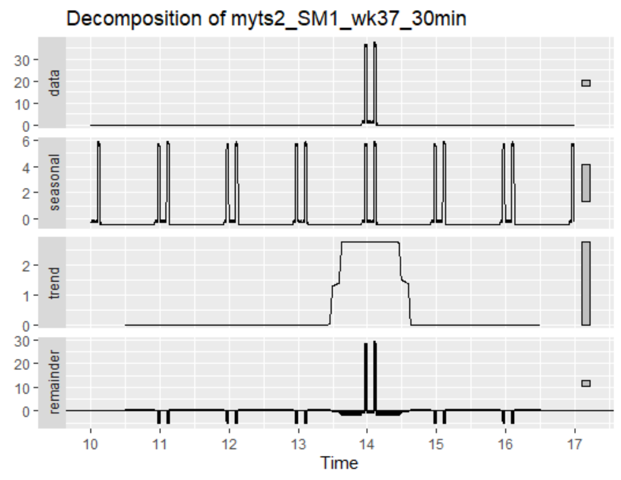


Decomposition of week 37 2007 time series in 15 min increments for submeter 2 (washing machine, dryer, refrigerator and a light). The trend component shows a constant energy use thru the week with an increase on Sept. 16th which remain thru the weekend.

Summary of time series components:

```
         Length Class  Mode

Data      4033   ts     numeric
seasonal 4033   ts     numeric
trend     4033   ts     numeric
random    4033   ts     numeric
figure     672   -none- numeric
type         1   -none- character
```

## Decompose Sub-meter 1 into trend, seasonal and remainder
components_tsSM1_myts2_30min <- decompose(myts2)
## Plot decomposed sub-meter 3
plot(components_tsSM1_myts2_30min)
autoplot(components_tsSM1_myts2_30min, labels = NULL, main= "Decomposition of myts2_SM1_wk37_30min", range.bars = NULL)
## Check summary statistics for decomposed sub-meter 3
summary(components_tsSM1_myts2_30min)



Decomposition of week 37 2007 time series in 30 min increments for submeter 1 (kitchen, dishwasher, oven, microwave oven). The trend component shows a constant energy use thru the week with an increase on Sept. 13th thru the 14th.

Summary of time series components:

```
        Length Class  Mode
Data     2353   ts     numeric
seasonal 2353   ts     numeric
trend    2353   ts     numeric
random   2353   ts     numeric
figure    336   -none- numeric
```

Simple exponential smoothing to remove seasonal components for forecasting. To do this seasonal adjustment will be done to the times series by removing the seasonal component, the adjusted timesereis will be plotted & decomposed to ensure seasonal component has been removed. Finally Holt-Winters exponential smoothing will be applied to the adusted time series and a forecast will be predicted.

# seasonal adjusting for submeter3
tsSM3_070809Adjusted <- tsSM3_07_08_09weekly - components070809SM3weekly$seasonal
autoplot(tsSM3_070809Adjusted, main = "tsSM3_070809Adjusted for seasonal adjutment")

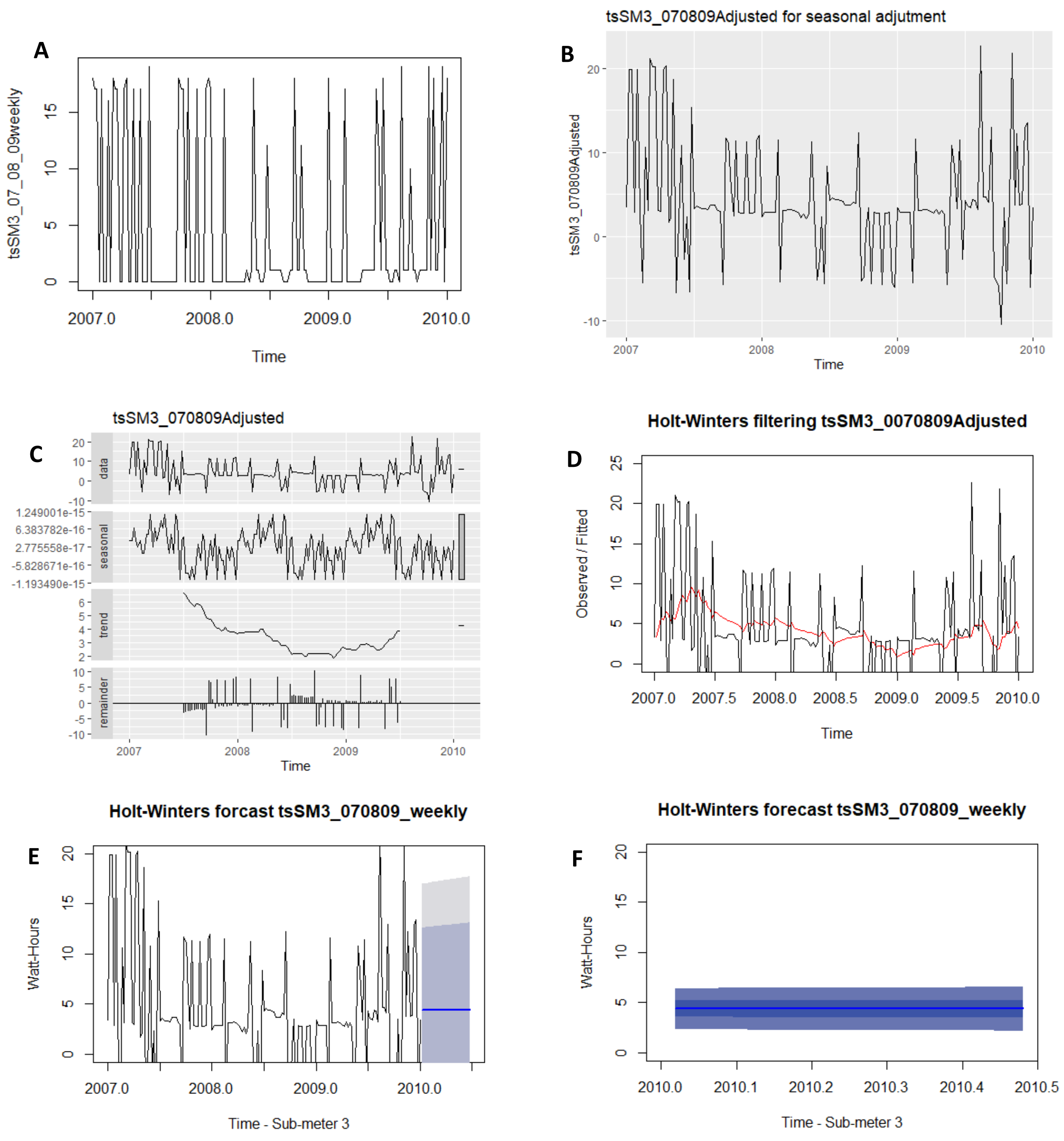
# decompose tsSM3-070809 Adjusted to confirm the removal of seasonal component

autoplot(decompose(tsSM3_070809Adjusted), main= "tsSM3_070809Adjusted")

# HoltWinters exponential smothing & plot
tsSM3_HW070809 <- HoltWinters(tsSM3_070809Adjusted, beta=FALSE, gamma=FALSE)
plot(tsSM3_HW070809, ylim = c(0, 25), main = "Holt-Winters filtering tsSM3_0070809Adjusted")

#Holtwinters Forecast
tsSM3_HW070809for <- forecast(tsSM3_HW070809, h=25)
plot(tsSM3_HW070809for, ylim = c(0, 20), ylab= "Watt-Hours", xlab="Time - Sub-meter 3", main = "Holt-Winters forcast tsSM3_070809_weekly")

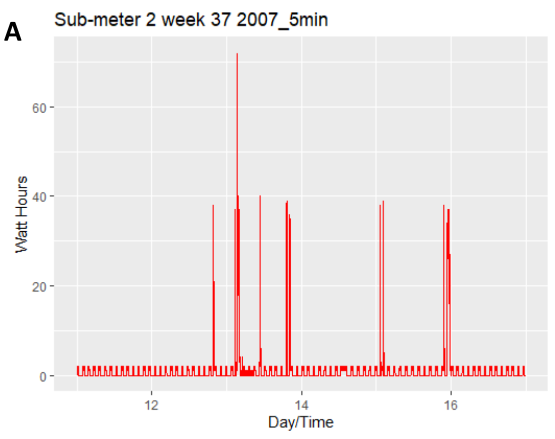
```
## Forecast HoltWinters with diminished confidence levels
tsSM3_HW070809forC <- forecast(tsSM3_HW070809, h=25, level=c(10,25))
## Plot only the forecasted area
plot(tsSM3_HW070809forC, ylim = c(0, 20), ylab= "Watt-Hours", xlab="Time - Sub-meter 3",main =
"Holt-Winters forecast tsSM3_070809_weekly", start(2010))
```



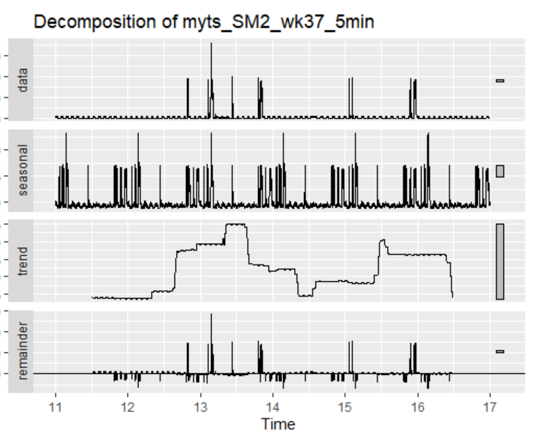
Using exponential smoothing & Holt-Winters function for forecasting of energy consumption Submeter 3. Energy consumption recorded on submeter 3 for years 2007,2008,2009 weekly time series. **A.** Raw data no adjustment or exponential smoothing was applied. **B**. Adjusted weekly time series after exponential smoothing and removal of seasonal component. **C.** Decomposition of adjusted weekly time series for confirmation seasonal component removal**. D.** Holt-Winters smoothing of the data (red) on adjusted weekly time series. **E&F**. Holt-Winters energy consumption forecast for weekly recording of year 2010 submeter 3

```
# Adjusting seasonality for submeter 1
# Adjust ts for submeter 1 week 37 30 minutes (myts2) by removing seasonal component
tsSM1_week37_30min_Adj <- myts2 - components_tsSM1_myts2_30min$seasonal
autoplot(tsSM1_week37_30min_Adj, ylab = "Watt-Hours",xlab = "Day/time", main =
"tsSM1_week37_30min_Adjusted for seasonal adjustment")

# decompose tsSM1_week37_30min_Adj to confirm the removal of sesonal component
autoplot(decompose(tsSM1_week37_30min_Adj), ylab= "Watt-Hours",main=
"tsSM1_week37_30min_Adjusted")

# Apply HoltWinters exponential smothing & plot
tsSM1_HW_week37_30min_Adj <- HoltWinters(tsSM1_week37_30min_Adj, beta=FALSE,
gamma=FALSE)
plot(tsSM1_HW_week37_30min_Adj, ylim = c(0, 30), ylab = "Watt-Hours", main = "Holt-Winters filtering
tsSM1_Week37_30min_Adj")

#Holtwinters Forecast
tsSM1_HW_week37_30min_Adj_for <- forecast(tsSM1_HW_week37_30min_Adj, h=168)
plot(tsSM1_HW_week37_30min_Adj_for, ylim = c(0, 30), ylab= "Watt-Hours", xlab="Time - Sub-meter
1", main = "Holt-Winters forcast tsSM1_2007_week37_5min")

## Forecast HoltWinters with diminished confidence levels
tsSM1_HW_week37_30min_Adj_forC <- forecast(tsSM1_HW_week37_30min_Adj, h=168,
level=c(10,25))
## Plot only the forecasted area
plot(tsSM1_HW_week37_30min_Adj_forC, ylim = c(0, 6), ylab= "Watt-Hours", xlab="Time - Sub-meter
1",main = "Holt-Winters forecast tsSM1_2007_week37_30min", start(17))
```

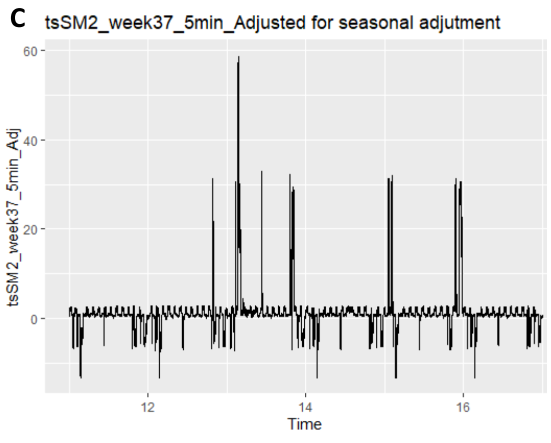
A



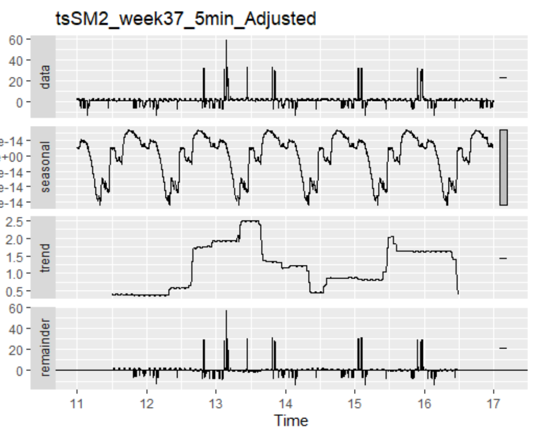
B



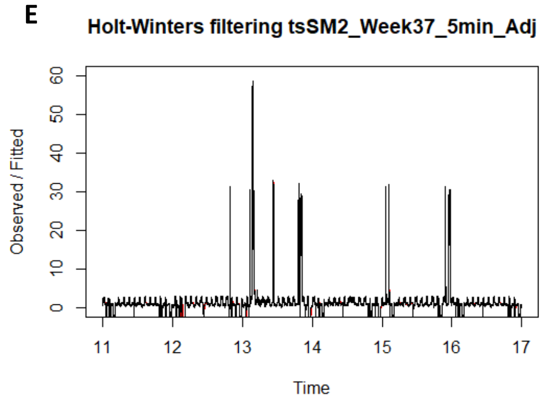
C



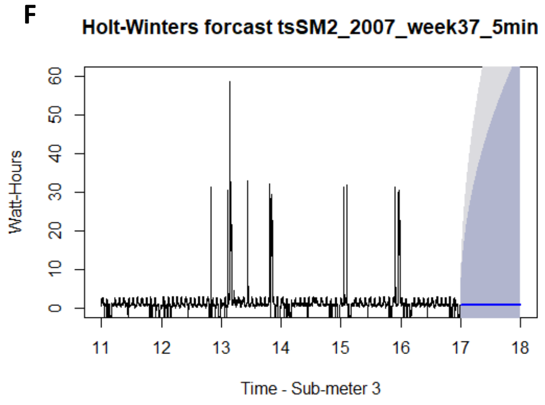
D



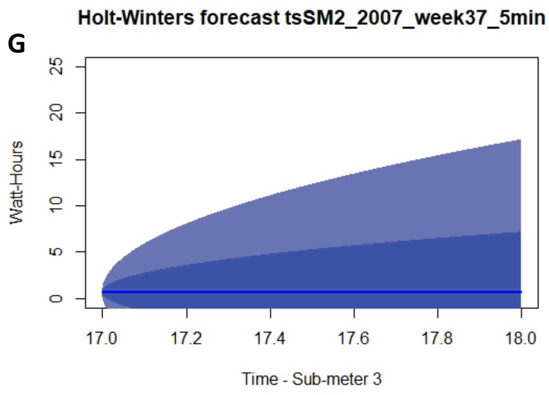
E



F



G



Using exponential smoothing & Holt-Winters function for forecasting of energy consumption. **A.** raw data No adjustment or exponential smoothing was applied. **B**. decomposition of raw data **C**. Adjusted time series after removal of seasonal component. **D**. Decomposition of adjusted time series for confirmation of seasonal component removal. **E**. Holt-Winters smoothing of the data (red) on adjusted weekly time series. **F&G**. Holt-Winters energy consumption forecast for the next 2016 time periods.

```
# seasonal adjusting for submeter2 week 37 5 min (myts)

#first it must be docomposed and create the components
# Decompose Sub-meter 2week 37 5 minutes ts into trend, seasonal and remainder
components_tsSM2_wk37_5min <- decompose(myts)
## Plot decomposed sub-meter 2 wk 37 5 min
plot(components_tsSM2_wk37_5min)
autoplot(components_tsSM2_wk37_5min, labels = NULL, main= "Decomposition of
myts_SM2_wk37_5min", range.bars = NULL)
## Check summary statistics for decomposed sub-meter 3
summary(components_tsSM2_wk37_5min)

# Adjust ts for submeter 2 week 37 5 minutes by removing seasonal component
tsSM2_week37_5min_Adj <- myts - components_tsSM2_wk37_5min$seasonal
autoplot(tsSM2_week37_5min_Adj, main = "tsSM2_week37_5min_Adjusted for seasonal adjutment")

# decompose tsSM2_week37_5min_Adj to confirm the removal of seasonal component
autoplot(decompose(tsSM2_week37_5min_Adj), main= "tsSM2_week37_5min_Adjusted")
# Apply HoltWinters exponential smothing & plot
tsSM2_HW_week37_5min_Adj <- HoltWinters(tsSM2_week37_5min_Adj, beta=FALSE, gamma=FALSE)
plot(tsSM2_HW_week37_5min_Adj, ylim = c(0, 60), main = "Holt-Winters filtering
tsSM2_Week37_5min_Adj")
```

#Holtwinters Forecast

```
tsSM2_HW_week37_5min_Adj_for <- forecast(tsSM2_HW_week37_5min_Adj, h=2016)
plot(tsSM2_HW_week37_5min_Adj_for, ylim = c(0, 60), ylab= "Watt-Hours", xlab="Time - Sub-meter 3",
main = "Holt-Winters forcast tsSM2_2007_week37_5min")
```

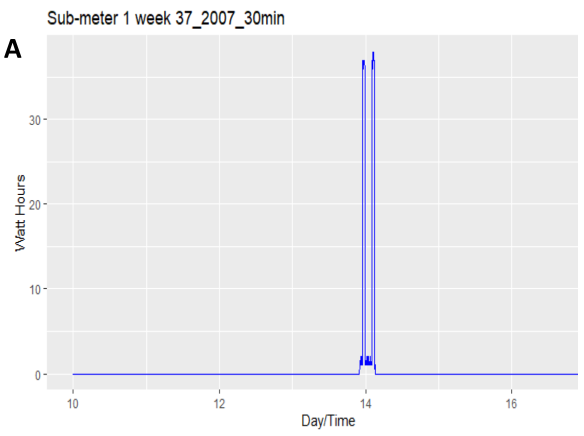
## Forecast HoltWinters with diminished confidence levels

```
tsSM2_HW_week37_5min_Adj_forC <- forecast(tsSM2_HW_week37_5min_Adj, h=2016, level=c(10,25))
```

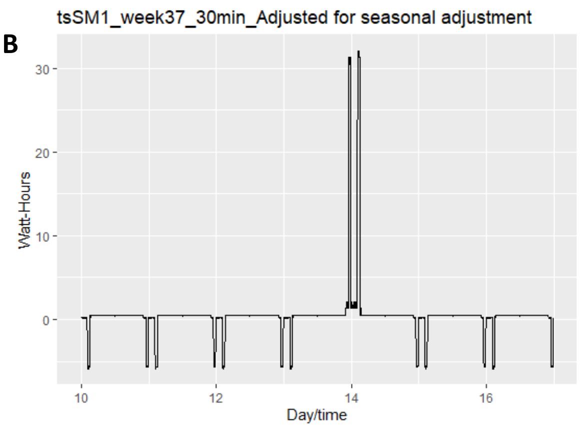
## Plot only the forecasted area

```
plot(tsSM2_HW_week37_5min_Adj_forC, ylim = c(0, 25), ylab= "Watt-Hours", xlab="Time - Sub-meter
3",main = "Holt-Winters forecast tsSM2_2007_week37_5min", start(2010))
```

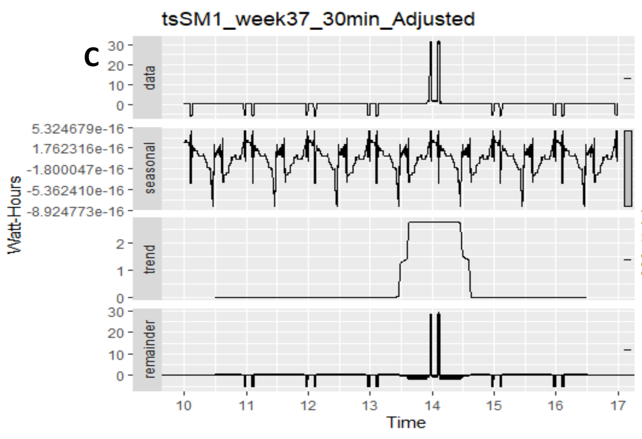
A



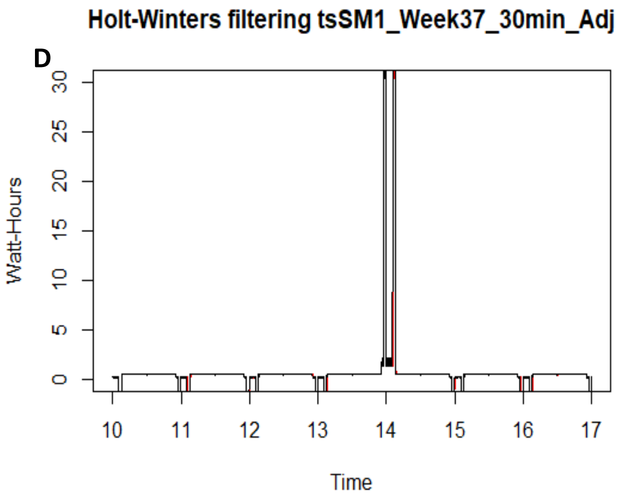
B



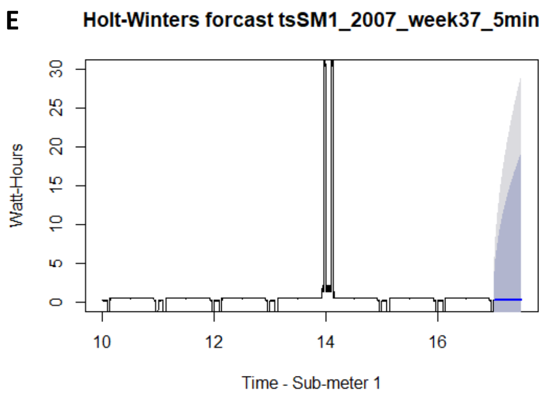
C



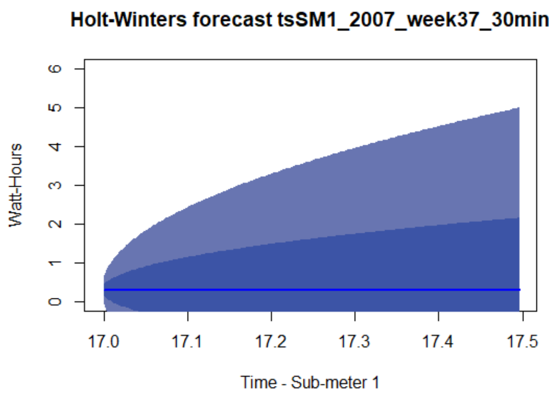
D



E



F



Using exponential smoothing & Holt-Winters function for forecasting of energy consumption. Energy consumption recorded on submeter 1 2007 week 37 in 30-minute time series **A.** Raw data no adjustment or exponential smoothing was applied. **B**. Adjusted time series after removal of seasonal component. **C.** Decomposition of adjusted weekly time series for confirmation seasonal component removal**. D.** Holt-Winters exponential smoothing of the data (red) on adjusted weekly time series. **E&F**. Holt-Winters energy consumption forecast for the next 160 time periods.

Conclusion and Remarks: This report shows how data collected from electric submeters can be used to visualize and understand power consumption in side a home. Additionally, we applied time series analysis using decomposition and exponential smoothing to develop energy use predictions for weekly, 30-minute, 15-minute and 5-minute time intervals. This type of analysis can be applied to a myriad of data collected over regular time periods.

Recommendations for this specific data set:

- Separate the AC and water heater units into individual submeters
- Monitor exterior temperature conditions to confirm AC use and assist in thermostat temp setting.
- Monitor refrigerator on individually
- Develop a notification system to inform when preset threshold energy use has been reached.