Time Series Missing Values

Code

The frequency plots show that there are lots of zeros.

Zero energy power consumption does not make sense unless there is a power outage or a faulty sensor that is incorrectly recording no energy use. Electronics that are plugged in are always consuming some sort of energy, even if it is a minimal amount. Unless there are power outages on daily basis, there is a problem with the sensor or the data.

How to Handle Missing Data for Time Series?

- 1. Without Trend, without Seasonality –> Mean, Median, Mode, Random Sample Imputation
- 2. With Trend, without Seasonality –> Linear Interpolation
- 3. With Trend, with Seasonality –> Seasonal Adjustment + Interpolation

https://miro.medium.com/max/2444/1*_RA3mCS30Pr0vUxbp25Yxw.png (https://miro.medium.com/max/2444/1*_RA3mCS30Pr0vUxbp25Yxw.png)

https://towardsdatascience.com/how-to-handle-missing-data-8646b18db0d4 (https://towardsdatascience.com/how-to-handle-missing-data-8646b18db0d4)

Trend → a general systematic linear or (most often) nonlinear component that changes over time and does not repeat Seasonality → a general systematic linear or (most often) nonlinear component that changes over time and does repeat Noise → a non-systematic component that is nor Trend/Seasonality within the data

https://towardsdatascience.com/trend-seasonality-moving-average-auto-regressive-model-my-journey-to-time-series-data-with-edc4c0c8284b (https://towardsdatascience.com/trend-seasonality-moving-average-auto-regressive-model-my-journey-to-time-series-data-with-edc4c0c8284b)

#What is my data?

When modeling, summary statistics are assumed to be consistent. In time series, this is referred to as stationary. This is violated when there is a trend, seasonality, or other time-dependent structures. Option is stationary.

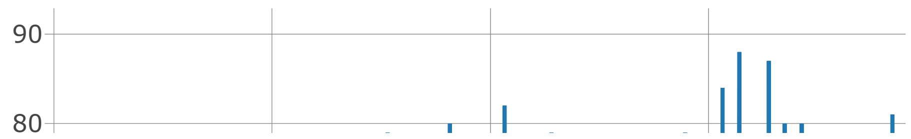
We are looking at three variables over time: Submeter1, Submeter2, Submeter3. So, I need to find out if Submeter over Time is stationary for each submeter. Based on the previous graphs over time, I can't tell whether there is a trend or not. They are too granular. By minute is just too much.

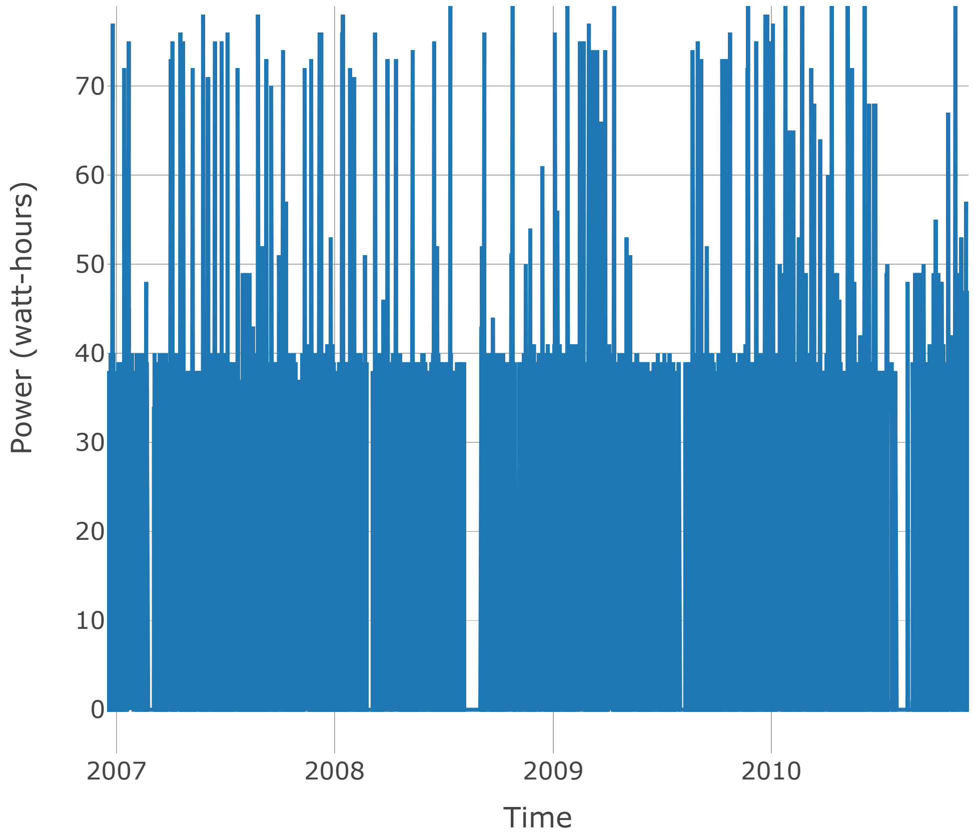
For example:

Hide

```
# Total Energy Over Time by Minutes

plot_ly(submetersByMinute, x = ~DateTime, y = ~Sub_metering_1, name = 'TotalEnergy', typ
e = 'scatter', mode = 'lines') %>%
 layout(#title = title_string,
 xaxis = list(title = "Time"),
 yaxis = list (title = "Power (watt-hours)"))
```



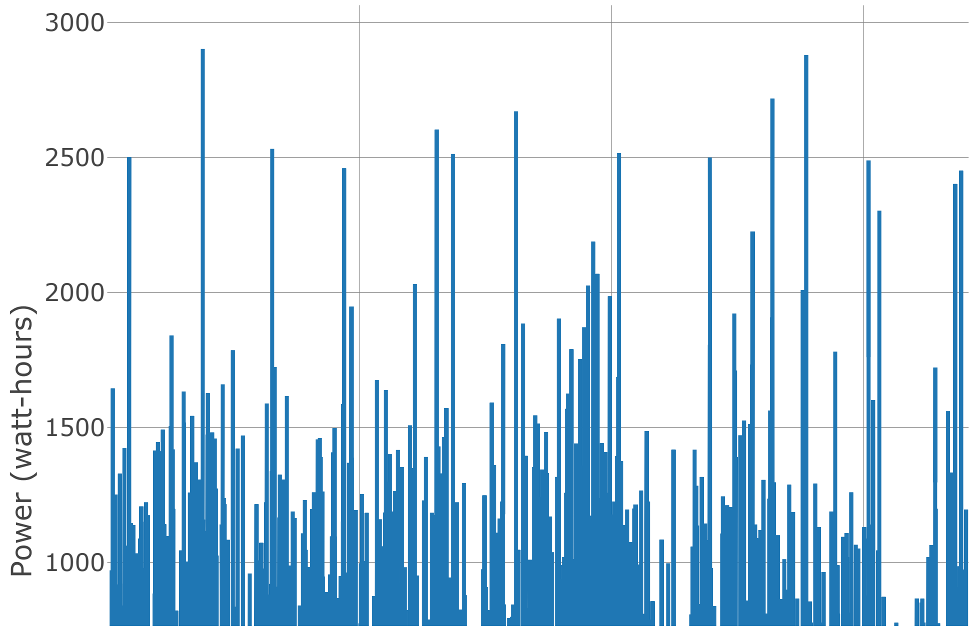


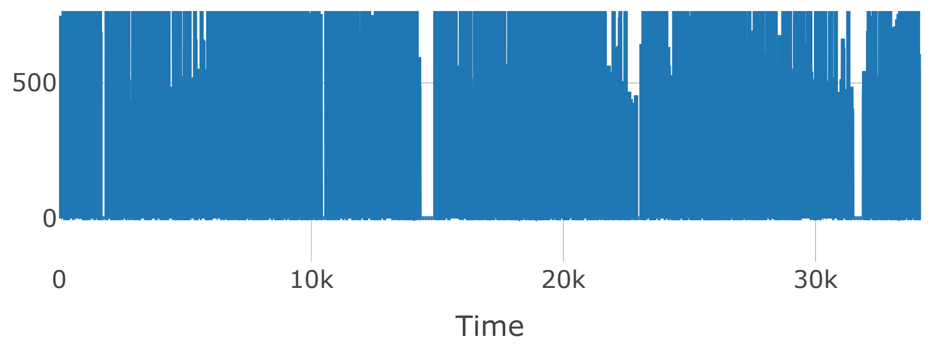
=

Hide

```
# Total Energy Over Time by Hours

plot_ly(submetersByHour, x = ~hour_index, y = ~Kitchen, name = 'Kitchen', type = 'scatte
r', mode = 'lines') %>%
 layout(#title = title_string,
 xaxis = list(title = "Time"),
 yaxis = list (title = "Power (watt-hours)"))
```

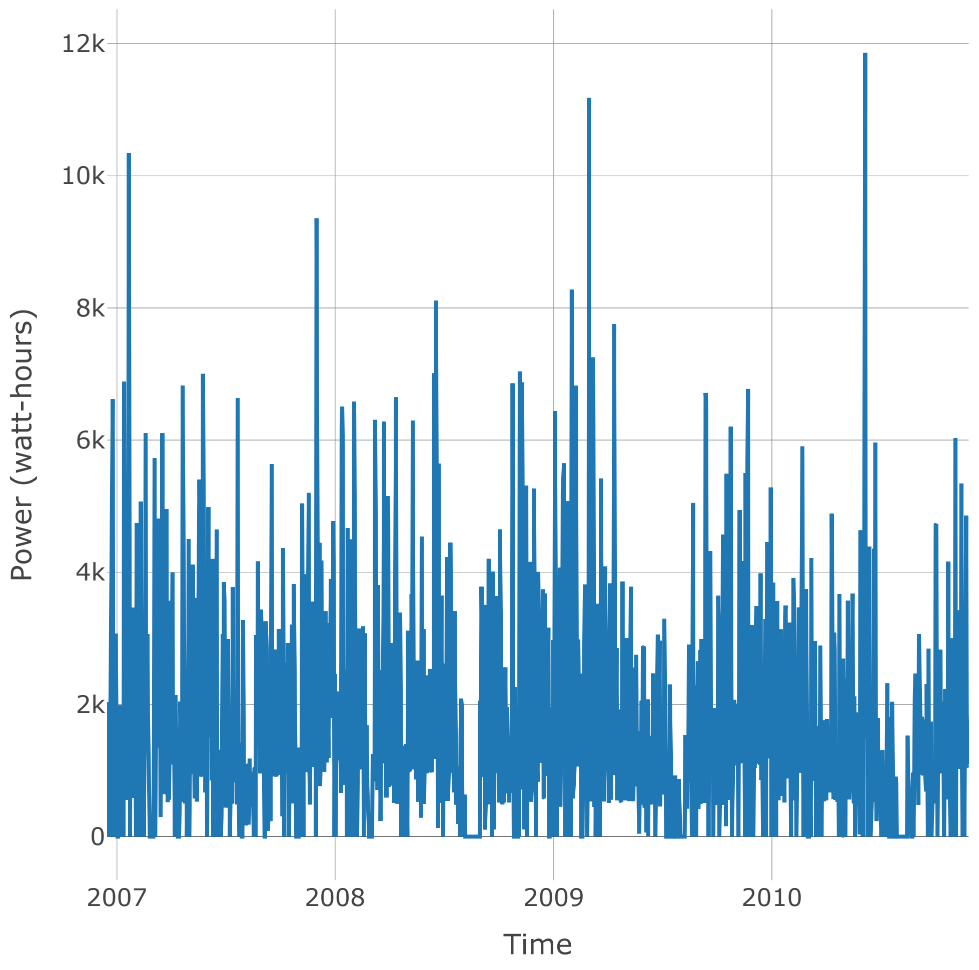




Hide

```
# Kitchen Use by Day

plot_ly(submetersByDay, x = ~date, y = ~Kitchen, name = 'Kitchen', type = 'scatter', mod
e = 'lines') %>%
 layout(#title = title_string,
 xaxis = list(title = "Time"),
 yaxis = list (title = "Power (watt-hours)"))
```



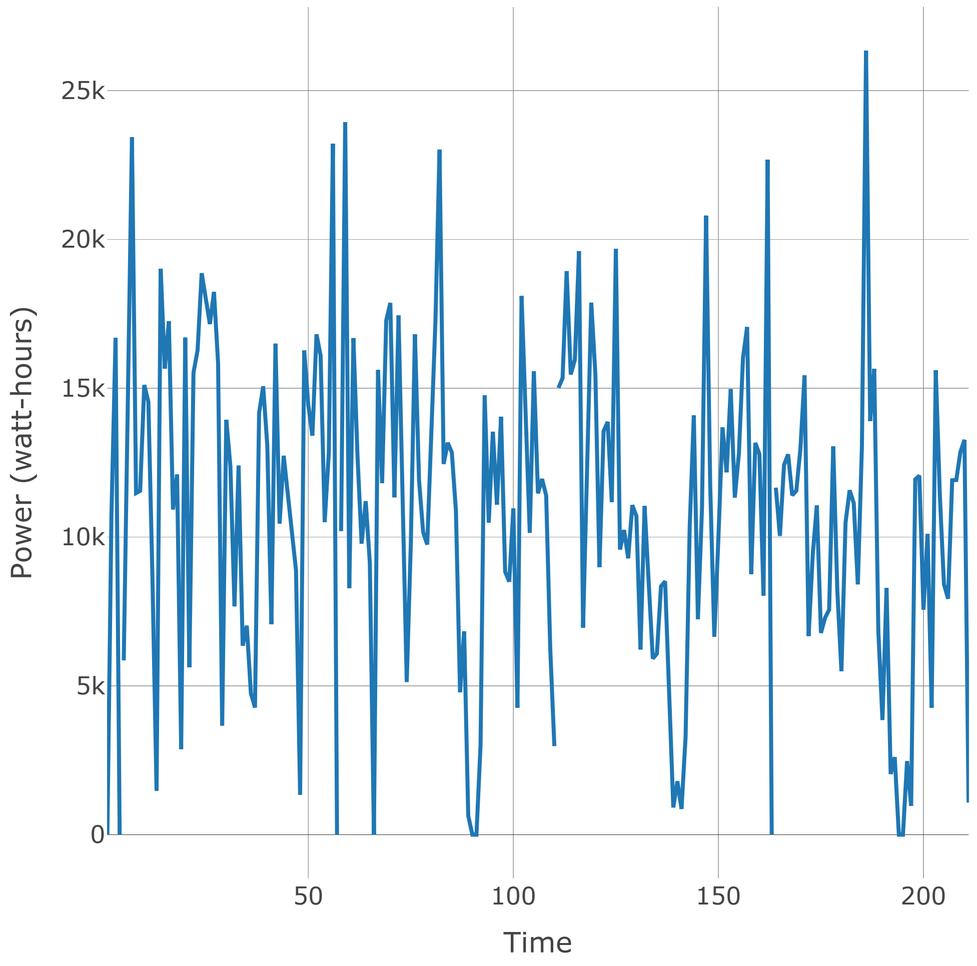
This is still pretty granular.

Let’s try by week.

Hide

```
# Kitchen Use by Week

plot_ly(submetersByWeek, x = ~week_index, y = ~Kitchen, name = 'Kitchen', type = 'scatte
r', mode = 'lines') %>%
 #add_trace(x = ~year, name = 'year', mode = 'lines') %>%
 layout(#title = title_string,
 xaxis = list(title = "Time"),
 yaxis = list (title = "Power (watt-hours)"))
```



Hide

```
NA
```

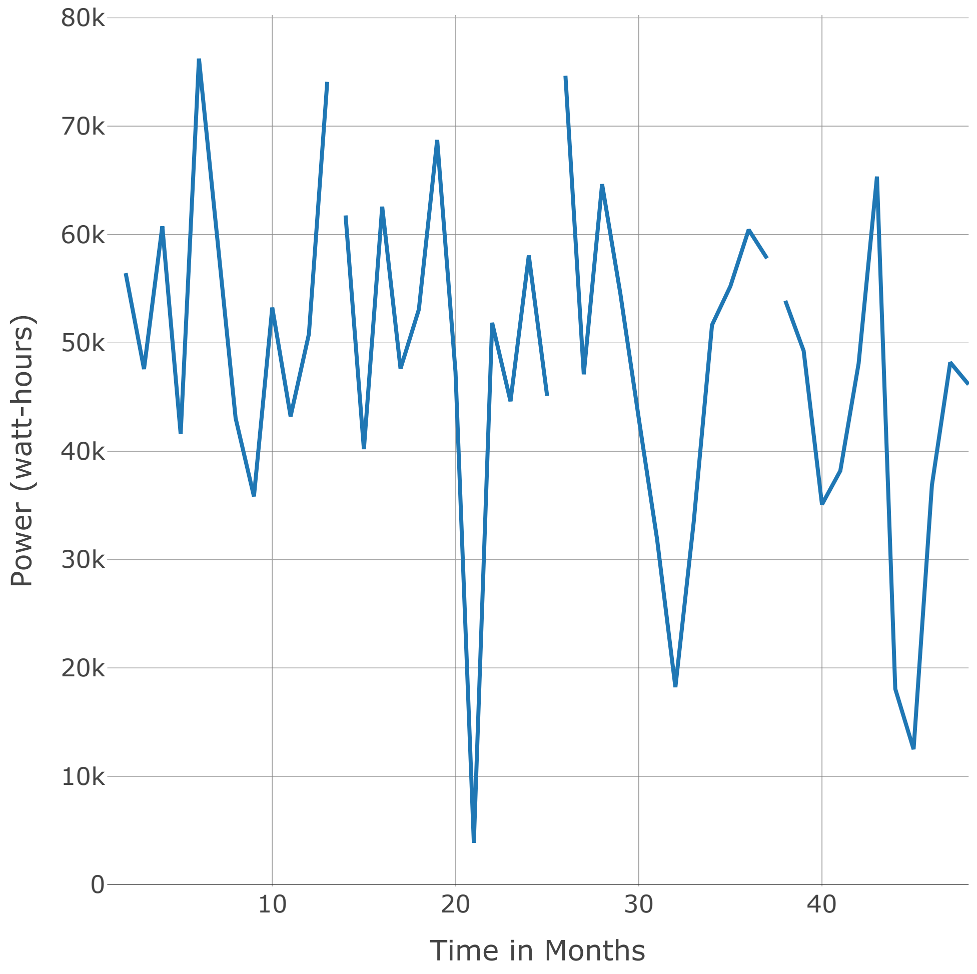
This is a bit better. You can see betweek week 100 and 120 that the data breaks. We will need to do something about the missing values.

What about by month?

Hide

```
# Kitchen use by Month

plot_ly(submetersByMonth, x = ~month_index, y = ~Kitchen, name = 'Kitchen', type = 'scat
ter', mode = 'lines') %>%
 #add_trace(x = ~year, name = 'year', mode = 'lines') %>%
 layout(#title = title_string,
 xaxis = list(title = "Time in Months"),
 yaxis = list (title = "Power (watt-hours)"))
```



Hide

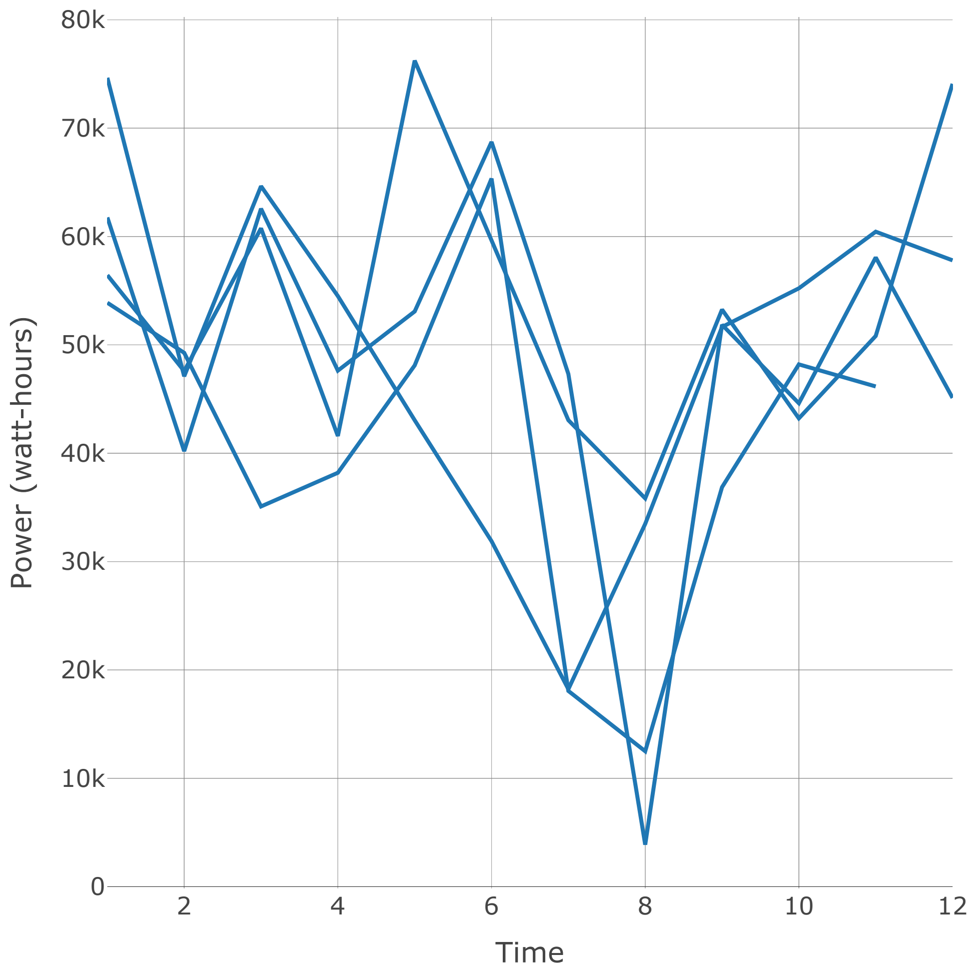
```
NA
```

The gaps are roughly at month 12 and I assume 24. Let’s look at this further by looking at monthly trends during each year.

Hide

```
# Kitchen use by Month

plot_ly(submetersByMonth, x = ~month, y = ~Kitchen, name = 'Kitchen', type = 'scatter',
 mode = 'lines') %>%
 #add_trace(x = ~month_index, name = 'year', mode = 'lines') %>%
 layout(#title = title_string,
 xaxis = list(title = "Time"),
 yaxis = list (title = "Power (watt-hours)"))
```



Hide

```
NA
```

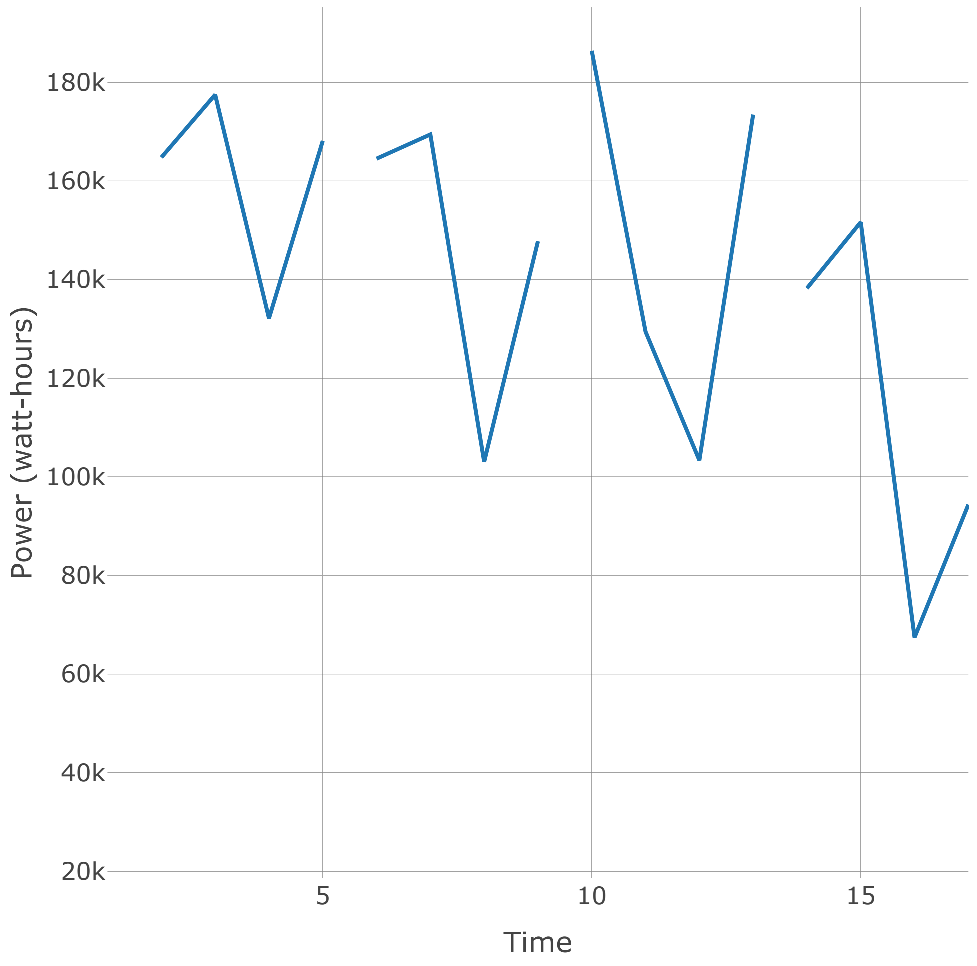
This is a seasonal pattern. Energy use changes with the months and are similar over the years.

How about by quarter?

Hide

```
# Kitchen Use by Quarter

plot_ly(submetersByQuarter, x = ~quarter_index, y = ~Kitchen, name = 'Kitchen', type =
'scatter', mode = 'lines') %>%
 #add_trace(x = ~year, name = 'year', mode = 'lines') %>%
 layout(#title = title_string,
 xaxis = list(title = "Time"),
 yaxis = list (title = "Power (watt-hours)"))
```



Hide

```
NA
```

There's definitely a repeating pattern over the quarters, (of course, imagining that the lines are connected) indicating seasonality. I'm not sure about a trend. As I decreased the granularity, I conclude that there is a trend, just a tiny bit, towards less power consumption.

The trend is not obvious, so, how do I confirm this?

Time Series Decomposition

Splits series into three components: trend, seasonality, random(noise).

Two ways: additive or multiplicative decomposition

Additive is when the seasonal variation looks constant. It doesn't change (like increase) with time. Muliplicative: The season variations change (grow) with time. This is also kind of hard to tell. I would say the size of the troughs look the same in the daily and monthly plot and somewhat in the quarterly plot.

We'll use an additive model.

What trend do we have?

We will use the decompose() from the forecast package. To use decompose(), we need to convert our dataframe into a ts (time series) object.

Decompose TS Submeter 1

Hide

```
#Create time objects of submeter_1
kitchenHourly_ts <- ts(submetersByHour[,4], start = c(2006, 12), frequency = 1248)
#submeter1Weekly_ts <- ts(submetersByWeek[,3], start = c(2007, 1), frequency = 52)
#submeter1Monthly_ts <- ts(submetersByMonth[,3], start = c(2007, 1), frequency = 12)
#submeter1Quarterly_ts <- ts(submetersByQuarter[,3], start = c(2007, 1), frequency = 4)
```

Hide

```
decomposed_submeter1Monthly_ts <- decompose(submeter1Monthly_ts, type="additive")
plot(decomposed_submeter1Monthly_ts)
```

Hide

```
decomposed_submeter1Quarterly_ts <- decompose(submeter1Quarterly_ts, type="additive")
plot(decomposed_submeter1Quarterly_ts)
```

So, Submeter 1 and Time, has a trend and is seasonal. We will use Seasonal Adjustment and Interpolation for missing values with Submeter 1. # Decompose TS Submeter 2

Hide

```
decomposed_ssubmeter2Weekly_ts <- decompose(submeter2Weekly_ts, type="additive")
plot(decomposed_ssubmeter2Weekly_ts)
```

Hide

```
decomposed_submeter2Monthly_ts <- decompose(submeter1Monthly_ts, type="additive")
plot(decomposed_submeter2Monthly_ts)
```

Hide

```
decomposed_submeter2Quarterly_ts <- decompose(submeter1Quarterly_ts, type="additive")
plot(decomposed_submeter2Quarterly_ts)
```

Same interpretation as submeter1. # Decompose TS Submeter 3

Hide

```
decomposed_ssubmeter3Weekly_ts <- decompose(submeter2Weekly_ts, type="additive")
plot(decomposed_ssubmeter3Weekly_ts)
```

Hide

```
decomposed_submeter3Monthly_ts <- decompose(submeter1Monthly_ts, type="additive")
plot(decomposed_submeter3Monthly_ts)
```

Hide

```
decomposed_submeter3Quarterly_ts <- decompose(submeter1Quarterly_ts, type="additive")
plot(decomposed_submeter3Quarterly_ts)
```

Same observation as submeter 1 and 2.

Next step: Seasonal Adjustment and Interpolation for missing values

Hide

```
statsNA(kitchenHourly_ts)
```

```
[1] "Length of time series:"
[1] 34164
[1] "-------------------------"
[1] "Number of Missing Values:"
[1] 0
[1] "-------------------------"
[1] "Percentage of Missing Values:"
[1] "0%"
[1] "-------------------------"
[1] "No NAs in the time Series."
[1] "No NAs"
```

Hide

```
statsNA(laundryRoomHourly_ts)
```

```
[1] "Length of time series:"
[1] 34164
[1] "-------------------------"
[1] "Number of Missing Values:"
[1] 0
[1] "-------------------------"
[1] "Percentage of Missing Values:"
[1] "0%"
[1] "-------------------------"
[1] "No NAs in the time Series."
[1] "No NAs"
```

Hide

```
statsNA(waterHeater_AC_ts)
```

```
[1] "Length of time series:"
[1] 34164
[1] "-------------------------"
[1] "Number of Missing Values:"
[1] 0
[1] "-------------------------"
[1] "Percentage of Missing Values:"
[1] "0%"
[1] "-------------------------"
[1] "No NAs in the time Series."
[1] "No NAs"
```

Are there really no missing values? Where are the 1.25% missing values?

Are missing values the same thing as gap in data? How can I find days or hours where there are no records at all? I can't find a function to do this. The imputeTS library was supposed to do this but it is not picking these up.

Why do my plots above (submeter over time) have gaps in there (see submeter vs quarter, submeter vs monthly). Do they correspond to the days with no power usage?

Plot days with no power

Hide

```
zeroPowerDays_Kitchen <- df2006_2010 %>%
    group_by(year, month, weekday, date) %>%
    summarize(Kitchen = sum(Sub_metering_1), LaundryRoom = sum(Sub_metering_2), WaterHea
ter_AC=sum(Sub_metering_3))  %>%
    filter(Kitchen == 0) %>%
    select(-LaundryRoom, -WaterHeater_AC) %>%
    arrange(weekday)

head(zeroPowerDays_Kitchen)
```

year <dbl>	month <dbl>	weekday <chr>	date <date>	Kitchen <dbl>
2007	1	Friday	2007-01-12	0
2007	2	Friday	2007-02-02	0
2007	2	Friday	2007-02-09	0
2007	2	Friday	2007-02-23	0
2007	3	Friday	2007-03-02	0
2007	4	Friday	2007-04-13	0

6 rows

Hide

```
View(zeroPowerDays_Kitchen)
```

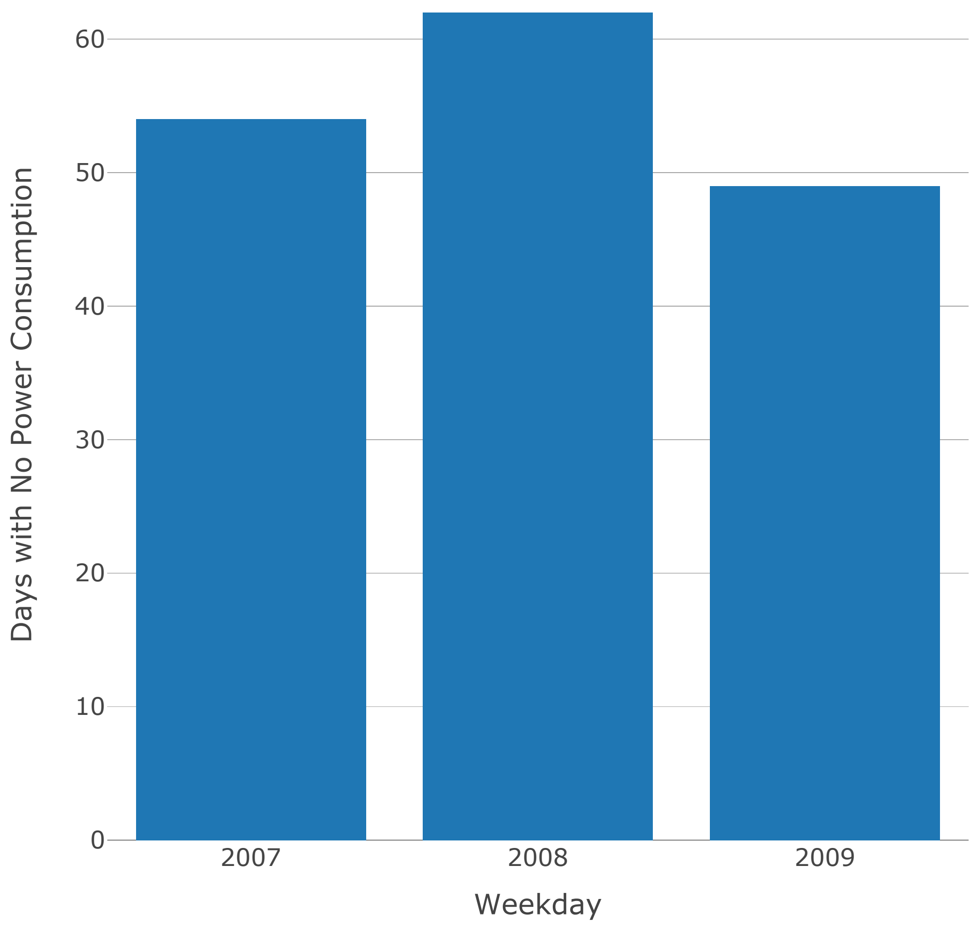
Hide

```
# Kitchen - Zero Power Days


#zeroPowerDays_Kitchen_Frequency_Weekdays <- #as.data.frame(zeroPowerDays_Kitchen_Freque
ncy_Weekdays)

xform <- list(title = "Weekday", categoryorder = "array", categoryarray = c("Monday","Tu
esday","Wednesday","Thursday","Friday","Saturday","Sunday"))


plot_ly(zeroPowerDays_Kitchen_Frequency_Weekdays, x = ~year, y = ~Total, name = 'Kitche
n', type = 'bar') %>%
 #add_trace(x = ~weekday, y = ~Total, name = 'weekday') %>%
 layout(xaxis = xform, yaxis = list(title = 'Days with No Power Consumption'))
```



Hide

```
NA
```

I’m struggling to get this plot to look nice. I want weekdays on the x-axis, broken down in to year(2007, 2008, 2009) and Count on Y axis. It’s just not working. I am moving on and will need to come back to this.

Next Step:

Look at no power kitchen usage days vs gaps in plots above. Do they match up?