Simulating Water Consumption to Develop Analysis Tools

Peter Prevos | 1 February 2018
1417 words | 7 minutes

I am currently working on developing analytics for a digital water metering project. Over the next five years, we are enabling 70,000 customer water meters with digital readers and transmitters. The data i not yet available but we don't want to wait to build reporting systems until after the data is live. The R language comes to the rescue as it has magnificent capabilities to simulate data. Simulating data is a useful technique to progress a project when data is being collected. Simulated data also helps because the outcomes of the analysis are known, which helps to validate the outcomes.

The raw data that we will eventually receive from the digital customer meters has the following basic structure:

- `DevEUI`: Unique device identifier.
- `Timestamp`: Date and time in (UTC) of the transmission.
- `Count`: The number of revolutions the water meter makes. Each revolution is a pulse which equates to five litres of water in a regular customer water meter.

Every device will send an hourly data burst which contains the cumulative meter read in pulse counts. The transmitters are set at a random offset from the whole our, to minimise the risk of congestion at the receivers. The time stamp for each read is set in the Coordinated Universal Time (UTC). Using this time zone prevents issues with daylight savings. All analysis will be undertaken in the Australian Eastern (Daylight) Time zone.

This article explains how we simulated test data to assist with developing reporting and analysis. The analysis of digital metering data follows in a future post.

Simulating water consumption

For simplicity, this simulation assumes a standard domestic diurnal curve (average daily usage pattern) for indoor water use. Diurnal curves are an important piece of information in water management. The curve shows water consumption over the course of a day, averaged over a fixed period. The example below is sourced from a journal article. This generic diurnal curve consists of 24 data points based on measured indoor water consumption, shown in the graph below.

> Source: Gurung et al. (2014) Smart meters for enhanced water supply network modelling and infrastructure planning. Resources, Conservation and Recycling (90), 34-50.

This diurnal curve only includes indoor water consumption and is assumed to be independent of seasonal variation. This is not a realistic assumption, but the purpose of this simulation is not to accurately model water consumption but to provide a data set to validate the reporting and analyses.

Simulating water consumption in R

The first code snippet sets the parameters used in this simulation. The unique device identifiers (DevEUI) are simulated as six-digit random numbers. The timestamps vector consists of hourly date-time variables in UTC. For each individual transmitter, this timestamp is offset by a random time. Each transmitter is also associated with the number of people living in each house. This number is based on a Poisson distribution.

```
library(tidyverse)
## Boundary conditions
n <- 100 # Number of simulated meters
d <- 100 # Number of days to simulate
s <- as.POSIXct("2120-01-01", tz = "UTC") # Start of simulation

set.seed(1969) # Seed random number generator for reproducibility
rtu <- sample(1E6:2E6, n, replace = FALSE) # 6-digit id
offset <- sample(0:3599, n, replace = TRUE) # Unique Random offset for each RTU

## Number of occupants per connection
occupants <- rpois(n, 1.5) + 1

as_tibble(occupants) %>%
  ggplot(aes(occupants)) +
  geom_bar(fill = "dodgerblue2", alpha = 0.5) +
  xlab("Occupants") +
  ylab("Connections") +
  theme_bw(base_size = 10) +
  ggtitle("Occupants per connection")

ggsave("simulated-occupants.png", width = 6, height = 4)
```

> Simulated number of occupants per connection.

The diurnal curve is based on actual data which includes leaks as the night time use shows a consistent flow of about one litre per hour. For that reason, the figures are rounded and reduced by one litre per hour, to show a zero flow when people are usually asleep. The curve is also shifted by eleven hours because the raw data is stored in UTC.

```
diurnal <- round(c(1.36, 1.085, 0.98, 1.05, 1.58, 3.87, 9.37, 13.3, 12.1, 10.3, 8.44, 7.04, 6.11, 5.68, 5.58, 6.67, 8.32, 10.0, 9.37, 7.73, 6.59, 5.18, 3.55, 2.11)) - 1

tdiff <- 11
diurnal <- c(diurnal[(tdiff + 1): 24], diurnal[1:tdiff])
```

This simulation only aims to simulate a realistic data set and not to present an accurate depiction of reality. This simulation could be enhanced by using different diurnal curves for various customer segments and to include outdoor watering, temperature dependencies and so on.

Simulating Water Consumption

A leak is defined by a constant flow through the meter, in addition to the idealised diurnal curve. A weighted binomial distribution ($\theta = 0.1$) models approximately one in ten properties with a leak. The size of the leak is derived from a random number between 10 and 50 litres per hour.

The data is stored in a matrix through a loop that cycles through each connection. The DevEUI is repeated over the simulated time period (24 times the number of days). The second variable is the timestamp plus the predetermined offset for each RTU.

The meter count is defined by the cumulative sum of the diurnal flow, multiplied by the number of occupants. Each point in the diurnal deviates from the model curve by ±10%. Any predetermined leakage is added to each meter read over the whole period of 100 days. The hourly volumes are summed cumulatively to simulate meter reads. The flow is divided by five as each meter revolution indicate five litres.

The next code snippet simulates the digital metering data using the assumptions and parameters outlined above.

```
## Leaking properties
leaks <- rbinom(n, 1, prob = .1) * sample(10:50, n, replace = TRUE)
data.frame(DevEUI = rtu, Leak = leaks) %>%
  subset(Leak > 0)

## Digital metering data simulation
meter_reads <- matrix(ncol = 3, nrow = 24 * n * d)
colnames(meter_reads) <- c("DevEUI", "TimeStampUTC" , "Count")

for (i in 1:n) {
  r <- ((i - 1) * 24 * d + 1):(i * 24 * d)
  meter_reads[r, 1] <- rep(rtu[i], each = (24 * d))
  meter_reads[r, 2] <- seq.POSIXt(s, by = "hour", length.out = 24 * d) + offset[i]
  meter_reads[r, 3] <- round(cumsum((rep(diurnal * occupants[i], d) +
                                       leaks[i]) * runif(24 * d, 0.9, 1.1))/5)
}

meter_reads <- meter_reads %>%
  as_tibble() %>%
  mutate(TimeStampUTC = as.POSIXct(TimeStampUTC, origin = "1970-01-01", tz = "UTC"))
```

Missing Data Points

The data transmission process is not 100% reliable and the base station will not receive some reads. This simulation identifies reads to be removed from the data through the temporary variable remove. This simulation includes two types of failures:

- Faulty RTUs (2% of RTUs with missing 95% of data)
- Randomly missing data points (1% of data)

```
## Set missing indicator
meter_reads <- mutate(meter_reads, remove = 0)

## Define faulty RTUs (2% of fleet)
set.seed(1969)
faulty <- rtu[rbinom(n, 1, prob = 0.02) == 1]
meter_reads$remove[meter_reads$DevEUI %in% faulty] <- rbinom(sum(meter_reads$DevEUI %in% faulty), 1, prob = .5)

## Data loss
missing <- sample(1:(nrow(meter_reads) - 5), 0.005 * nrow(meter_reads))
for (m in missing){
  meter_reads[m:(m + sample(1:5, 1)), "remove"] <- 1
}

## Remove missing reads
meter_reads <- filter(meter_reads, remove == 0) %>%
  select(-remove)
write_csv(meter_reads, "water-quantity/simulated-consumption.csv")

##Visualise
filter(meter_reads, DevEUI %in% faulty) %>%
  mutate(TimeStampAEST = as.POSIXct(format(TimeStampUTC,
                                           tz = "Australia/Melbourne"))) %>%
  filter(TimeStampAEST >= as.POSIXct("2120-02-06") &
         TimeStampAEST <= as.POSIXct("2120-02-08")) %>%
  arrange(DevEUI, TimeStampAEST) %>%
  ggplot(aes(x = TimeStampAEST, y = Count, colour = factor(DevEUI))) +
  geom_line() +
  geom_point()
ggsave("simulated-missing.png", width = 6, height = 4)
```

The graph shows an example of the cumulative reads and some missing data points.

> Simulated water consumption with missing reads.

Analysing Digital Metering Data

Data simulation is a good way to develop your analysis algorithms before you have real data. I have also used this technique when I was waiting for survey results during my dissertation. When the data finally arrived, I simply had to plug it into the code and finetune the code. R has great capabilities to simulate reality to help you understand the data. The ggplot package provides excellent functionality to visualise water consumption.

In next week's article, I will outline how I used R and the Tidyverse package to develop libraries to analyse digital metering data.

Data Science for Water Professionals

If you like to know more about using R to analyse water data, then onsider following the course *Data Science for Water Utility Professionals*.

Data Science for Water Utility Professionals

Managing reliable water services requires not only a sufficient volume of water, but also large amounts of data. This course teaches the basics of data science using the R language and the Tidyverse libraries to analyse water management problems.

LeanPub

As seen on R Bloggers

Share this page

-
-
-
-

You might also enjoy reading these articles

Cheesecake Diagrams: Pie Charts with a Different Flavour

Factor Analysis in R: Measuring Consumer Involvement

Call Centre Workforce Planning Using Erlang C in R language