

Lotek Tag Lifespan Model

John Brzustowski

4 April, 2016

We need some reasonable upper bound for the lifetimes of tags, by battery and BI.

Lotek provides a short table here:

<http://www.lotek.com/bird+bat-nano.pdf> (As of 1 April, 2016)

and there are additional models listed in “ANTC Spec Sheet.pdf” (no source), including the only numbers for battery type “3-1”. Since lifetime is listed as depending only on the battery (numeric portion of model string), we model on that.

```
lt = as.tbl(read.csv("./lotekTagLifespanByBatteryAndBI.csv"))
print(lt)
```

```
# A tibble: 7 x 5
  battery lifespan2 lifespan5 lifespan10 lifespan10onOff
  <fctr>    <int>    <int>    <int>    <int>
1      1         10         21         33         45
2      2         16         33         52         71
3    3-1         20         41         64         87
4    3-2         39         80        124        170
5    4-2         79        163        251        344
6    6-1        113        232        357        489
7    6-2        215        441        678        928
```

Rearrange so that we have columns *battery*, *bi*, *dutyCycle*, *lifespan*.

```
ls = lt %>% transmute(battery = battery, bi = 2, dutyCycle = 1, lifespan = lifespan2)
ls = ls %>%
  transmute(battery = battery, bi = 5, dutyCycle = 1, lifespan = lifespan5) %>%
  bind_rows(ls)

ls = ls %>%
  transmute(battery = battery, bi = 10, dutyCycle = 1, lifespan = lifespan10) %>%
  bind_rows(ls)

ls = ls %>%
  transmute(battery = battery, bi = 10, dutyCycle = 0.5, lifespan = lifespan10onOff) %>%
  bind_rows(ls)

ls = ls %>% mutate(biInv = 1.0 / bi, dcInv = 1.0 / dutyCycle)

print(ls)
```

```
# A tibble: 28 x 6
  battery    bi dutyCycle lifespan biInv dcInv
  <fctr> <dbl>    <dbl>    <int> <dbl> <dbl>
1      1     10      0.5         45  0.1    2
```

| | | | | | | |
|----|-----|----|-----|-----|-----|---|
| 2 | 2 | 10 | 0.5 | 71 | 0.1 | 2 |
| 3 | 3-1 | 10 | 0.5 | 87 | 0.1 | 2 |
| 4 | 3-2 | 10 | 0.5 | 170 | 0.1 | 2 |
| 5 | 4-2 | 10 | 0.5 | 344 | 0.1 | 2 |
| 6 | 6-1 | 10 | 0.5 | 489 | 0.1 | 2 |
| 7 | 6-2 | 10 | 0.5 | 928 | 0.1 | 2 |
| 8 | 1 | 10 | 1.0 | 33 | 0.1 | 1 |
| 9 | 2 | 10 | 1.0 | 52 | 0.1 | 1 |
| 10 | 3-1 | 10 | 1.0 | 64 | 0.1 | 1 |

... with 18 more rows

A simple model assumes that the battery capacity, K , depends on the battery model, and that tag power consumption r is a sum of a baseline rate r_0 plus a pulse-dependent rate r_1 . All tags transmit 4 pulses per burst, so the pulse-dependent rate r_1 depends only on duty cycle and burst interval like so: $r_1 = r_p * \frac{dutyCycle}{BI}$

The full non-linear model is :

$$lifespan = \frac{K}{r_0 + \frac{r_p * dutyCycle}{BI}}$$

This is over-parameterized (we can divide top and bottom by r_0 to get a model with two parameters), so we simplify by rewriting K/r_0 as D , the number of days of battery life at the baseline rate, and r_p/r_0 be r_t , the relative rate of power consumption during transmission, versus baseline. The new model is:

$$lifespan = \frac{D}{1 + \frac{r_t * dutyCycle}{BI}}$$

We fit the model to each type of battery:

```
par = res = pred = NULL
bnames = unique(ls$battery)
for (m in bnames) {
  res[[m]] = nls(lifespan~D / (1 + rt * dutyCycle / bi),
                 subset(ls, battery==m), list(D = 500, rt = 5))
  par = rbind(par, c(coefficients(res[[m]]), max(residuals(res[[m]]))))
  pred = rbind(pred,
               data.frame(
                 battery=m,
                 bi=1:40,
                 lifespan=predict(res[[m]], list(bi=1:40, dutyCycle=1))
               ))
}
rownames(par) = bnames
colnames(par) = c("D", "rt", "Max Residual (days)")
```

Stu Mackenzie pointed out there is a light-weight version of the *NTQB-1* which he's dubbed *NTQB-1-LW*, with “~2/3 the lifetime of the NTQB-1”. For now we'll assume that means the D parameter for that model is 2/3 that for the NTQB-1

```
par = rbind(c(par[["1","D"]] * 2 / 3, par[["1", "rt"]], NA), par)
rownames(par)[1] = "1-LW"
pred = rbind(data.frame(
  battery="1-LW",
```

```

bi=1:40,
lifespan=par[["1-LW", "D"]] / (1 + par[["1-LW", "rt"]] / (1:40)))
, pred)

```

Now map model names to batteries. We do this as a simple table because the naming scheme isn't consistent enough to bother doing it programmatically.

```
## Models and the batteries they correspond to
```

```

modelBattery = list(
"NTQB-1-LW" = "1-LW",
"NTQB-1"     = "1",
"NTQB-2"     = "2",
"NTQB-3-2"   = "3-2",
"NTQB-4-2"   = "4-2",
"NTQB-6-1"   = "6-1",
"NTQB-6-2"   = "6-2",
"NTQBW-3-2"  = "3-2",
"NTQBW-2"    = "2",
"NTQBW-4-2"  = "4-2",
"NTQBW-6-2"  = "6-2",
"ANTC-M1-1"  = "1",
"ANTC-M2-1"  = "2",
"ANTC-M3-1"  = "3-1",
"ANTC-M3-2"  = "3-2",
"ANTC-M4-2"  = "4-2",
"ANTC-M4-2S" = "4-2",
"ANTC-M4-2L" = "4-2",
"ANTC-M6-1"  = "6-1",
"ANTC-M6-2"  = "6-2",
"ANTCW-M1-1" = "1",
"ANTCW-M2-1" = "2",
"ANTCW-M3-1" = "3-1",
"ANTCW-M3-2" = "3-2",
"ANTCW-M4-2" = "4-2",
"ANTCW-M4-2S" = "4-2",
"ANTCW-M4-2L" = "4-2",
"ANTCW-M6-1" = "6-1",
"ANTCW-M6-2" = "6-2"
)

modPar = NULL
for (b in names(modelBattery))
  modPar = rbind(modPar, par[modelBattery[[b]],])
rownames(modPar) = names(modelBattery)

```

The results show good agreement with the data table from Lotek:

```
print(round(par, 1))
```

| | D | rt | Max Residual (days) |
|------|------|------|---------------------|
| 1-LW | 48.5 | 12.3 | NA |

| | | | |
|-----|--------|------|-----|
| 1 | 72.8 | 12.3 | 0.3 |
| 2 | 114.7 | 12.2 | 0.4 |
| 3-1 | 138.5 | 11.8 | 0.4 |
| 3-2 | 271.6 | 11.9 | 0.2 |
| 4-2 | 547.0 | 11.8 | 0.2 |
| 6-1 | 775.5 | 11.7 | 0.1 |
| 6-2 | 1469.0 | 11.7 | 0.2 |

The parameter r_t can be interpreted as the ratio of energy consumed during 1 second with a burst to that consumed during 1 second without a burst. Estimates of this parameter vary only by 5% across tag types, and in monotonic fashion, perhaps due to variation in battery internal resistance. The table provided by Lotek only covers $2 \leq BI \leq 20$ (the larger value from BI=10s @ 50% duty cycle); curves are extrapolated down to 1s and from 20 to 40s using the fitted model.

```
xyplot(lifespan~bi, groups=battery, pred,
  auto.key=list(corner=c(.05,.95)),
  main="Reported (X) and Predicted (o) Tag Lifespan by Battery Type",
  xlab="Burst Interval (seconds)",
  ylab="Lifespan (days)",
  type="b",
  panel = function(x, y, type, groups, ...) {
    panel.xyplot(x, y, type, groups, ...)
    meas = which(x %in% c(2, 5, 10, 20) & groups != "1-LW")
    panel.points(x[meas], y[meas], pch="X", cex=1.5)
    panel.abline(h=0, lty=2, col="gray")
  }
)
```

Reported (X) and Predicted (o) Tag Lifespan by Battery Type

