

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"))
```

Add inverses of columns bi and dutyCycle, for modelling

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

print(ls)
```

```
# A tibble: 44 x 6
  battery    bi dutyCycle lifespan biInv dcInv
  <fctr> <dbl>    <dbl>    <dbl> <dbl> <dbl>
1      1     10      0.5      45  0.1    2
2      1     10      1.0      33  0.1    1
3      1      2      1.0      10  0.5    1
4      1      5      1.0      21  0.2    1
5      2     10      0.5      71  0.1    2
6      2     10      1.0      52  0.1    1
7      2      2      1.0      16  0.5    1
8      2      5      1.0      33  0.2    1
9     3-1     10      0.5      87  0.1    2
10    3-1     10      1.0      64  0.1    1
# ... with 34 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. Note that
## we include a model called 'unknown' which corresponds to
## a battery called 'unknown' that is assigned a life equal
## to the average of those for NTQB2, NTQB-3-2.
## This forces a warning to be emailed to the motus admin whenever
## the metadata cache is updated, because an unmodelled tag
## is a serious metadata issue.

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",
"ACT-521" = "521",
"ACT-626" = "626",
"ACT-393" = "393",
"NTQB2-1" = "1",
"NTQB2-2" = "2",
"NTQB2-3-2" = "3-2",
"NTQB2-4-2S" = "4-2",
"NTQB2-6-1" = "6-1",
"NTQB2-6-2" = "6-2",
"unknown" = "unknown"
)

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
521	304.1	20.1	1.3
626	608.3	20.1	2.6
393	1839.8	24.3	3.3
unknown	193.1	12.0	0.3

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(log10(lifespan)~bi, groups=battery, pred,
  auto.key=list(corner=c(.6,.1), columns=3),
  main="Reported (X) and Predicted (o) Tag Lifespan by Battery Type",
  xlab="Burst Interval (seconds)",
  ylab="Lifespan (log10(days); 1->10, 2->100, 3->1000)",
  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")
  }
)
```

