Post-Processing Telemetry Data

Download:

The sensorgnome package can be downloaded from here . Use the latest version available.

The SensorGnome records raw VHF pulses which must be post-processed in order to detect tags. Lotek receivers record tag detections directly, but it is still worth processing this output to remove likely false positives, and to distinguish among tags which might have the same Lotek ID but different burst intervals.

For the 2013 and 2014 field seasons, we have asked users to send us their raw data for post-processing, but we are (slowly) re-packaging the code to let you do this yourself. This page describes what you can do, and will be updated as the code evolves.

Note to Mac users: the filterTags() and findTags() functions don't currently work on OSX. if you need to run these and don't have easy access to a PC, let me know.

Pre-requisite: Install R and the sensorgnome package and required 3-rd party packages

- install R (version 3.0.0 or later)
- **recommended:** install rstudio, an open source GUI for R on the desktop
- install the RSQLite and other required packages available via cran by doing this within R:

```
install.packages(c("RSQLite", "lattice", "lubridate"))
```

- install the sensorgnome R package, available at http://public.sensorgnome.org/Sensorgnome_Package
  - download the file sensorgnomeXXX.zip
  - start R on your computer
  - choose this from the R menu:
    Packages | Install package(s) from local zip files...
  - browse to and select the downloaded sensorgnomeXXX.zip file

You can now do any of the tasks below.

Task: Load data from a .sqlite file

As of October 2014, the preferred format for transferring tag detection datasets from sensorgnome.org will be in '.sqlite' files. An sqlite file is a database which can contain multiple tables. One advantage over an R dataset file (".rds") is that you can load a specified subset of the data directly from the file, rather than having to read the whole file into memory and then subset it there. Another advantage is that the .sqlite file can be updated, e.g. have new data added directly to it on disk. For an .rds file, the same operation requires reading the entire file into memory, appending rows, then rewriting the full dataset. Using .sqlite files helps make sensorgnome.org data processing and distribution more efficient.

```
## use the sensorgnome package
library(sensorgnome)

## read tags with the default filtering applied (freqsd < 0.1 & runLen >
x = tags("2014_MYPROJ_MYSITE_alltags.sqlite")

## read only records for tagss from the Taylr project with IDs in the ra
x = tags("tagProj='Taylr' and id >= 400", "2014_MYPROJ_MYSITE_alltags.sq

## you can of course re-save such data as an rds file:

saveRDS(x, "2014_taylor_tags_400_up.rds")
```

See the documentation for the `tags()` function in the help for package sensorgnome.

Task: Plot data from .rds files

```
## read in the .rds file
x = readRDS("2014_MYPROJ_MYSITE_alltags.rds")

## apply our usual false-positive filtering (might be too extreme for lo
x = subset(x, freqsd < 0.1 & runLen > 2)

## use the lattice plotting library
library(lattice)
```

```
## plot detected tag label vs. time, colour code by antenna #
xyplot(label~ts, groups=ant, x, auto.key=list(corner=c(0, 1), title="Ant

## filter to show only a particular time period (Aug 1 to end of Aug 14)
## then plot signal strength vs time for each tag separately, colour cod
library(lubridate)
y = subset(x, ts >= ymd("2014-08-01") & ts < ymd("2014-08-15"))
xyplot(sig~ts|label, groups=ant, y, auto.key=list(corner=c(0, 1), title=


## print a table of time range of detections for each tag
tapply(x$ts, x$label, range)

## print all detections of a given tag
subset(x, id == 156)

## print all detections of a given tag, showing only date/time and signa
subset(x, id == 156, select=c(ts, sig))
```

Task: Filter .DTA files from Lotek Receivers

This filters tag detections from a lotek receiver, retaining only those which are "likely" to be from known tags, given the master database of registered tags. Filtered tag detections will be split into two groups: those from your project, and those from other people's projects. We encourage you to email the latter to us, or upload it to your wiki home page.

Proceed like so:

- generate a ".DTA" file from your Lotek receiver. This requires software from Lotek.
- download the **public** version of the master tag database, attached to this page; e.g. 2013_tags_public.csv
- run R
- load the sensorgnome package by doing this from inside R:

```
library(sensorgnome)
```

- filter a single file using the dta2sg() function from inside R:

```
dta2sg()
```

- you will be prompted to select the .DTA file and the public master tag database file
- you will be asked which project code corresponds to you (these are 5-letter codes abbreviating the last name of the main contact person for the project)
- you will be asked to enter a site code. This should be all caps and short, for example **RCARS.**
  Note: this prompting happens right in the Rgui console, rather than popping up a window.
- the program may take several minutes to run, during which time the Rgui interface will be unresponsive.

The function will read the .DTA file, run a filtering program, and split the results into files corresponding to your tags, and to tags from other projects. These output files are "R dataset" files (ending in **.rds**), and are saved in the same folder as the original DTA file. You can read them into R and generate two summary plots like so:

```
library(lattice)   ## load the lattice graphics package

x = readRDS("2013_Taylor_canso_lotek_filtered_my_tags.rds")

xyplot(fullID~ts, groups=ant, x, auto.key=list(corner=c(0,0), title="Ant

xyplot(sig~ts|fullID, groups=ant, x, auto.key=list(corner=c(0,0), title=
```

Visualising the filtering: annotated DTA files

To make it easier to understand the filtering (and find bugs!), you can have the package generate annotated copies of the .DTA file, which show you which tag detections were retained, and how they were grouped into runs compatible with the burst interval.

```
library(sensorgnome)
annotateDTA("c:/Users/john/Desktop/Bon_Portage_2013.DTA", "c:/Users/john/
```

This will filter tags, and use the output to annotate a copy of the .DTA file. The result is an html file which will be automatically displayed in a web browser, with a copy saved in the same folder as the original .DTA file. If you just want an annotated plain text file, then omit the **`html=TRUE`** parameter.

These functions try to remember the latest folder from which a .DTA file was read, the latest public tag database used, and the choice of project. These values are stored in variables beginning with

"sensorgnome.", and can be reset by doing one or more of the following:

```
rm(sensorgnome.project)
rm(sensorgnome.dbfile)
rm(sensorgnome.dtadir)
```

Task: Read a .DTA file into R

If you just want to read a .DTA file into R, without any filtering, use the readDTA function:

```
library(sensorgnome)
x = readDTA("c:/Users/john/Desktop/Sable_Island_2013.DTA")
```

The result will be a list with these items:

- **recv** is the receiver model and serial number
- **tags** is a data frame with these columns:
    - **ts**: timestamp
    - **id**: lotek tag ID
    - **ant**: lotek antenna code
    - **sig**: signal power, in Lotek units (0..255)
    - **lat**: latitude, or NA
    - **lon**: longitude, or NA
    - **dtaline**: line number in the dta file from which this tag record came
    - **antfreq**: nominal antenna frequency
    - **gain**: antenna gain, in Lotek units (0..99)
    - **codeset**: lotek codeset
- **pieces** is a character vector with (possibly huge) items, each corresponding to a chunk of the .DTA file
- **pieces.lines.before** is an integer vector giving the number of lines in the .DTA file before each chunk in **pieces**
  (these last two items are for use by the annotateDTA function, described above)

The DTA chunks are processed in order, so that appropriate antenna channel, frequency, and gain settings are applied to each tag table. If you think this isn't happening correctly - please let me know!

Task: Convert Lotek Signal Strength Values to dBm (EXPERIMENTAL)

We have a provisional (uncalibrated and not officially documented) function to convert Lotek signal strength values to dBm. After reading in a .DTA file, you can add a dBm column like so:

```
library(sensorgnome)
x = readDTA("c:/Users/john/Desktop/Sable_Island_2013.DTA")
x$dBm = lotekPowerTodBm(x$sig, s$gain)
```

The function lotekPowerTodBm uses the detection signal strength and the applicable receiver gain setting to estimate incoming signal power, using a formula based on sales literature from Lotek.

Task: Look for Unregistered Tags

Occasionally, tags get deployed without registering them first, because, well field seasons happen. You can look for records of consistent but unregistered tag detections in a .DTA file, then choose to add any or all of them to the master database, like so:

```
library(sensorgnome)
findUnregTags()
```

You will be prompted for a .DTA file and a tag database. The algorithm looks for runs of detections with the same ID and consistent burst interval different from those in the database. If any are found, you will be prompted to choose some to add to your copy of the master database file.
If you do this, please send us an updated copy of the master database file so that we can seek these tags in data from other people's receivers!

Task: Export Public Tag Records from DTA file

If your project has "confidential" tags (ones whose detections must be kept private), you can still share your unfiltered detections of tags from other projects. This is similar to the purpose of dta2sg() (documented above), but lets other users do their own filtering.

```
library(sensorgnome)
x = publicDTA()
```

This will prompt you for a DTA filename, a public tag database filename, the name of your project, and the name of your site. The result in x will be a list with these elements:

- x$**recv** - the receiver model and serial number
- x$**proj** - the project name
- x$**site** - the site name
- x$**tags** - the "public" tag detections from the DTA file; i.e. only those whose ID is in the public tag database. If you have private tags with the same IDs as tags in the public database, you'll need to create a copy of the public database with records of those tags removed, so that x$tags won't contain detections of your tags (this of course means that detections of other projects' tags with the same IDs will also be removed, unlike for dta2sg)

The result list for file **`WHATEVER.DTA`** will also be written to a file in the same folder called **`WHATEVER_raw_public_tags.rds`**; this file will only contain records for tags in the public database file.