Receiver Efficiency Index

Updated: 2019-02-19

Contents

1 Importing Libraries **1**

2 Importing Data **1**

2.1 Cleaning Data 2

3 Running REI **2**

4 Plotting with Plotly **2**

The receiver efficiency index is number between `0` and `1` indicating the amount of relative activity at each receiver compared to the entire set of receivers, regardless of positioning. The function takes a set detections and a deployment history of the receivers to create a context for the detections. Both the amount of unique tags and number of species are taken into consideration in the calculation.

The receiver efficiency index implement is implemented based on the paper Acoustic telemetry array evolution: From species- and project-specific designs to large-scale, multispecies, cooperative networks. Each receiver's index is calculated on the formula of:

$$REI = \frac{T_r}{T_a} \times \frac{S_r}{S_a} \times \frac{DD_r}{DD_a} \times \frac{D_a}{D_r}$$

REI = Receiver Efficiency Index

T_r = The number of tags detected on the receiver

T_a = The number of tags detected across all receivers

S_r = The number of species detected on the receiver

S_a = The number of species detected across all receivers

DD_a = The number of unique days with detections across all receivers

DD_r = The number of unique days with detections on the receiver

D_a = The number of days the array was active

D_r = The number of days the receiver was active

1 Importing Libraries

We will import `dplyr`,`glatos` and `plotly` to run then visualize the REI.

```
library(dplyr)
library(glatos)
library(plotly)
```

2 Importing Data

We will import the sample data below using `glatos::read_glatos_detections()` and `glatos::read_glatos_receivers()`

```
detection_file <- system.file("extdata", "walleye_detections.csv", package = "glatos")
receiver_file <- system.file("extdata", "sample_receivers.csv", package = "glatos")

receivers <- read_glatos_receivers(receiver_file)
detections <- read_glatos_detections(detection_file)
```

2.1 Cleaning Data

Below we use `dplyr::mutate()` to ensure that any recovery times that are set as `NA` are set to the current date and time. You can replace `Sys.time()` with the last known download time if you know it.

```
receivers <- receivers %>%
  mutate( recover_date_time = replace(recover_date_time,
                                      is.na(recover_date_time),
                                      Sys.time()))
```

3 Running REI

`REI()` takes two arguments. The first is a dataframe of detections the detection timstamp, the station identifier, the species, and the tag identifier. The next is a dataframe of deployments for each station. The station name should match the stations in the detections. The deployments need to include a deployment date and recovery date or last download date. Details on the columns mentioned see the preparing data section.

```
rei <- glatos::REI(detections,receivers)
```

The resulting dataframe looks like this:

```
head(rei)
  station latitude longitude         rei
1 DRF-004 42.24937 -83.11824 0.001238343
2 DRL-004 42.12746 -83.11873 0.001562130
3 DRL-010 42.07690 -83.12096 0.001413047
4 DRL-011 42.09637 -83.11681 0.001162267
5 DRU-001 42.35693 -82.93016 0.001685863
6 DRU-002 42.35278 -82.92844 0.001576594
```

4 Plotting with Plotly

Below is the code for plotting the RI using `plotly`. `plotly` allows us to interact with the map rather than haveing a static image. More about `plotly` can be found here: https://plot.ly/r/ . To export figures as image files from `plotly` you will also need the external command line program `orca`. To install `orca`, see https://github.com/plotly/orca#installation.

```
geo <- list(
  scope = 'north america',
  showland = TRUE,
  landcolor = toRGB("white"),
  showocean = TRUE,
  oceancolor = toRGB("gray"),
  showcountreies = TRUE,
  showlakes = TRUE,
  lakecolor = plotly::toRGB("gray"),
```

```
  resolution = 50,
  center = list(lat = median(rei$latitude),
                lon = median(rei$longitude)),
  lonaxis = list(range=c(min(rei$longitude)-1, max(rei$longitude)+1)),
  lataxis = list(range=c(min(rei$latitude)-1, max(rei$latitude)+1))
)


map <- rei %>%
  plot_geo(lat = ~latitude, lon = ~longitude, color = ~rei,width=900 )%>%
  add_markers(
    text = ~paste(station, ': ', rei),
    hoverinfo = "text",
    size = ~c(rei * 5 +5)
  )%>%
  layout(title = "REI", geo = geo)
```

To show the map you can just type out the variable name.

```
map
```