# StacomiR

*Marion Legrand, Cedric Briand*

*2018-10-11*

## stacomiR

### Introduction

Migratory fishes population are vulnerable as they are often more prone to human impact when migrating in rivers and to the ocean (McDowall, 1992). They are often counted at stations when they perform the migrations at some of their lifestages, and these counts provide valuable indices to the population size and trend. The objective of the stacomi project is to provide a common database for people monitoring fish migration, so that data from watershed are shared, and stocks exchanging between different basins are better managed. The stacomi database, is an open-source database, it managed with a JAVA interface, and results from that database are treated directly with the stacomiR project.

### Installation

The package is available from CRAN, a development version is available from R-Forge.

```
# get the package from CRAN
install.packages("stacomiR")
# get the development version
install.packages("stacomiR", repos="http://R-Forge.R-project.org")
```

### Usage

Launch the graphical interface

```
stacomi()
```

The program can be launched to use from the command line

```
stacomi(gr_interface = FALSE, login_window = TRUE, database_expected = TRUE)
```

### Data structure

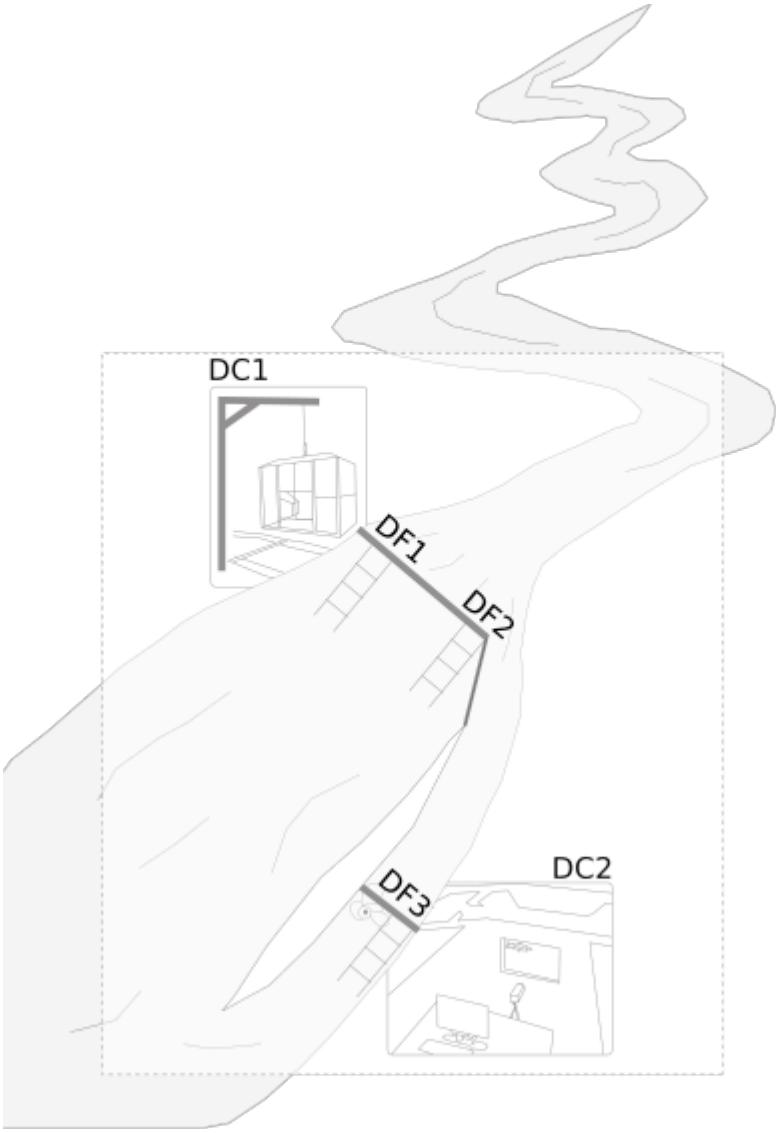
The open source postgresql database comprises a common schema with dictionaries, and different schema for different users. Each user can save its own schema and send it to others. The database comprises tables related to infrastructure, operations and fish samples. *Contact the authors to get a copy of the database.*

#### Infrastructure

##### Station

A migration report is always built on a section of a river, this is called the station. A station of fish migration monitoring is a section of a watercourse where fish upstream or downstream migration is monitored. The station covers the whole section of a single river, but can extend to several natural or artificial channels. A station consists physically of as many dams as hydrographic sections monitored (river, channels, etc.).

According to the local settings, it corresponds to one river location with a counting device, or to one or several dams. For example, in the figure below we can see a station with three crossing device (DF 1 to 3) and two counting device (DC 1 to 2), the first one beeing a trap counting device (DC1) and the other a video-counting device (DC2).




**Dams**

The concept of dam used in the context of fish migration monitoring database refers to a system blocking or guiding the migratory flow like :

- weir,
- electric guide barrier,
- netting dam,
- etc.

**Crossing device**

A crossing device (DF) is a passageway that allows and concentrates the migratory flow between upstream and downstream sections of a dam. They can be of various type :

- fishway,
- spillway,
- fish elevator,
- eel trapping ladder,
- etc.

It is possible to have more than one crossing device on the same dam.

#### Counting device

A counting device (DC) is a set of equipment installed on a crossing device used to monitor fish migration. It can be :

- a video counting device,
- a trap,
- an accoustic counting device,
- ...

#### Monitoring operation

An operation corresponds to the monitoring of a counting device during a time span.

#### Sample

A sample corresponds to a batch of fishes passing during a monitoring operation. Sample characteristics (length, weight, sex, body measurements) are attached to the sample. For each sample the species and the stage (which corresponds to a maturation stage and is related to migratory behaviour) is recorded. Samples correspond to multiple fish of the same species and stage or to individual records.

#### Other features

The database also handles, marking-recapture operations, pathologies, samples collection (scale, fin sample for genetic...), fate of fishes (released, death, farmed, etc.), etc... Some tables are also used to insert information about environmental condition such as turbidity, atmospheric pressure, temperature, flow ...

## Package structure

The package relies on S4 classes. *Referential classes* are used to access data from the database (taxa, stages, counting devices...). *Report classes* are built from referential classes and have different methods to access the database *connect methods*, generate calculations *calcule method*, or plot results. For instance, the migration report class comprises slots for :

- DC The counting device (camera, trap, acoustic device...)
- taxa The species list from the database and the taxa selected
- stage The stages list from the database and the stage selected
- starting date The date of beginning

- ending date The last date of the report

Read the help files e.g. `? report_mig` to get documentation on the following classes.

| Class | Command | description |
|---|---|---|
| report_mig | `new("report_mig")` | Migration report (single) |
| report_mig_mult | `new("report_mig_mult")` | Migr. (several DC,taxa...) |
| report_annual | `new("report_annual")` | Multi year migration counts |
| report_dc | `new("report_dc")` | Counting device operation |
| report_df | `new("report_df")` | Fishway operation |
| report_mig_env | `new("report_env")` | Migration crossed with env. factors |
| report_mig_char | `new("report_df")` | Migration with fish characteristics |
| report_mig_interannual | `new("report_mig_interannual")` | Comp. between years |
| report_sample_char | `new("report_sample_char")` | Sample characteristics |
| report_ge_weight | `new("report_ge_weight")` | Trend in glass eel weight |
| report_silver_eel | `new("report_siver_eel")` | Silver eel migration & stage |
| report_sea_age | `new("report_sea_age")` | Set sea age for Salmon |
| report_species | `new("report_species")` | Species composition |

## Working examples

### Command line

### Migration report

Examples are provided with each of the class, you can access them simply by typing `? report_mig_mult`. The program is intented to be used in conjuntion with the database, to test it without access, use the arguments `login_windows=FALSE` and `database_expected=FALSE`.

```
## launches the application in the command line without connection to the database
stacomi(gr_interface=FALSE,login_window=FALSE,database_expected=FALSE)
```

The following code is only run when there is a connection to the database. The program will create an object of the class report_mig_mult, and run it for several DC, here 5 is a vertical slot fishway, and 6 and 12 are two glass eel trapping ladder located at the Arzal dam in the Vilaine river (France). We are evaluating the migration of all stages of eel (glass eel CIV, yellow eel AGJ and silver eel AGG). Glass eel and yellow eel migrate to the watershed while silver eels are migrating back to the ocean. Data are loaded from the database with the `charge` method and the `calcule` method will interpolate daily migration from monitoring operations which do not necessarily span a day, and convert the glass eel weight in numbers.

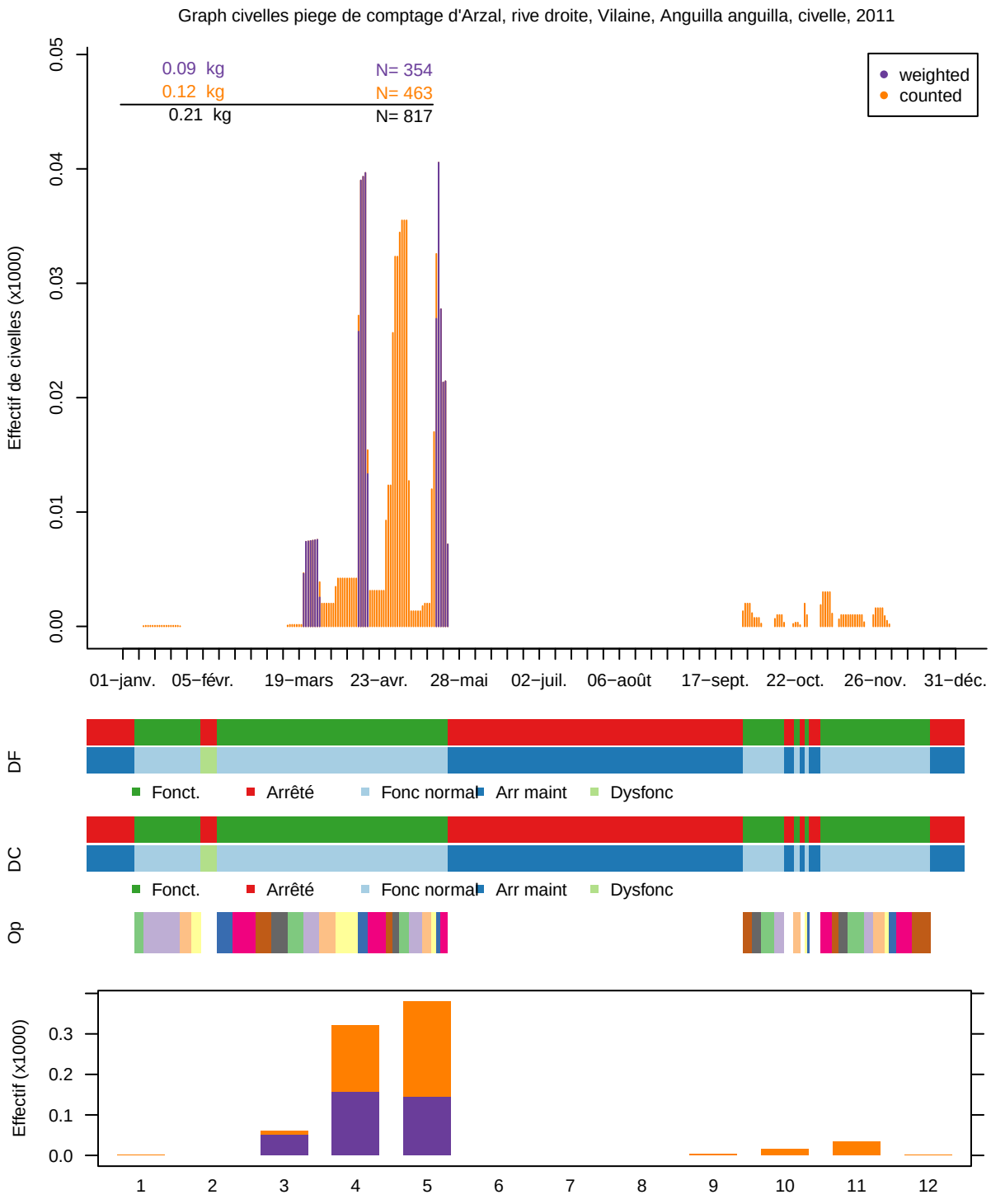
```
  stacomi(gr_interface=FALSE,
     login_window=FALSE,
     database_expected=TRUE)
r_mig_mult=new("report_mig_mult")
r_mig_mult=choice_c(r_mig_mult,
     dc=c(5,6,12),
     taxa=c("Anguilla anguilla"),
     stage=c("AGG","AGJ","CIV"),
     datedebut="2011-01-01",
     datefin="2011-12-31")
r_mig_mult<-charge(r_mig_mult)
# launching charge will also load classes associated with the report
# e.g. report_ope, report_df, report_dc
r_mig_mult<-connect(r_mig_mult)
```

```
  # calculations
  r_mig_mult<-calcule(r_mig_mult,silent=TRUE)
```

The previous line generates data not only about the report_mig_mult class, but also runs dependent classes which describe how the fishway (DF) and counting devices (DC) have been operated. Sometimes there are no data but only because the camera was not working. There are also information about the operations (e.g. periods at wich a trap content has been evaluated). Here we load what would have been generated if we had run the previous lines.

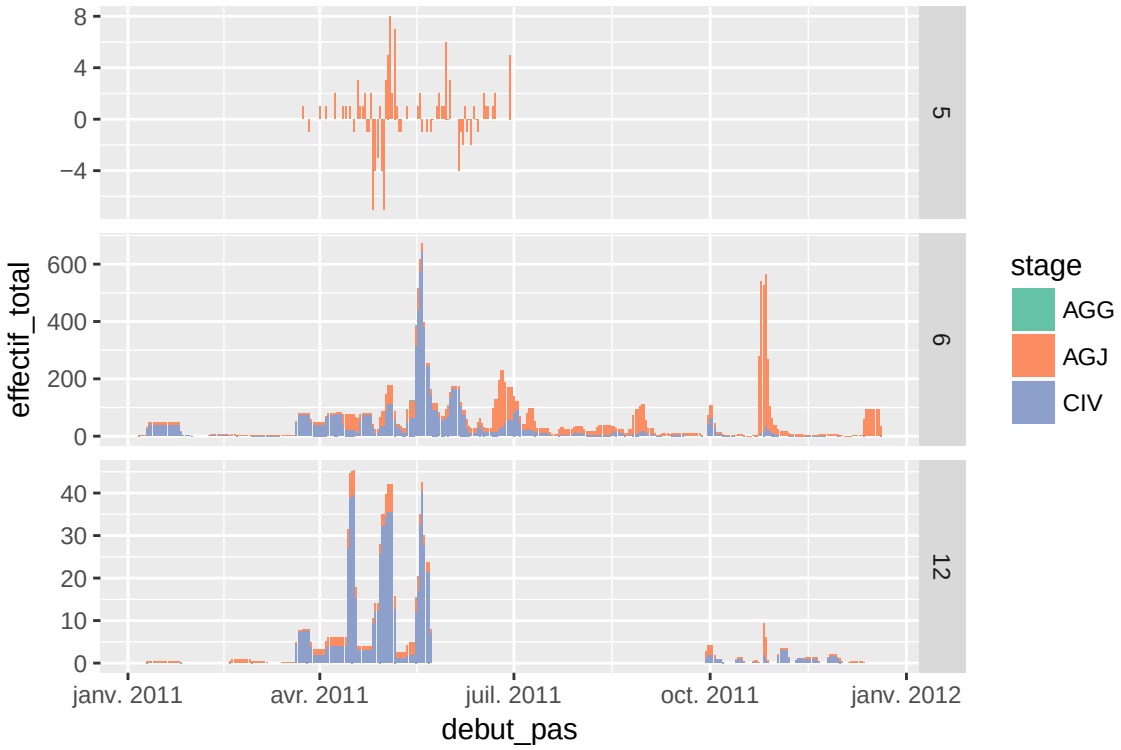
One graph per DC, taxa and stage. Below as an example, the glass eel migration in weight and number (top), the periods and type of operation for DF and DC, and the operation (trapping periods) (middle), a summary of migration per month (bottom).

```
# Without a connection at the database we can launch these lines to generate the graph
# To obtain titles in french use Sys.setenv(LANG = "fr")
stacomi(gr_interface=FALSE,
    login_window=FALSE,
    database_expected=FALSE)
data("r_mig_mult")
data("r_mig_mult_ope")
assign("report_ope",r_mig_mult_ope,envir=envir_stacomi)
data("r_mig_mult_df")
assign("report_df",r_mig_mult_df,envir=envir_stacomi)
data("r_mig_mult_dc")
assign("report_dc",r_mig_mult_dc,envir=envir_stacomi)
r_mig_mult<-calcule(r_mig_mult,silent=TRUE)

# To avoid call to dev.new() which creates a device per stage, DC, taxa, we simplify
# the object as dev.new() causes knitr to crash:
r_mig_mult@taxa@data<- r_mig_mult@taxa@data[1,]
r_mig_mult@stage@data<-r_mig_mult@stage@data[3,]
r_mig_mult@dc@dc_selectionne<-r_mig_mult@dc@dc_selectionne[3]

plot(r_mig_mult,plot.type="standard",silent=TRUE)
```




Summary of migration for different stages and counting devices

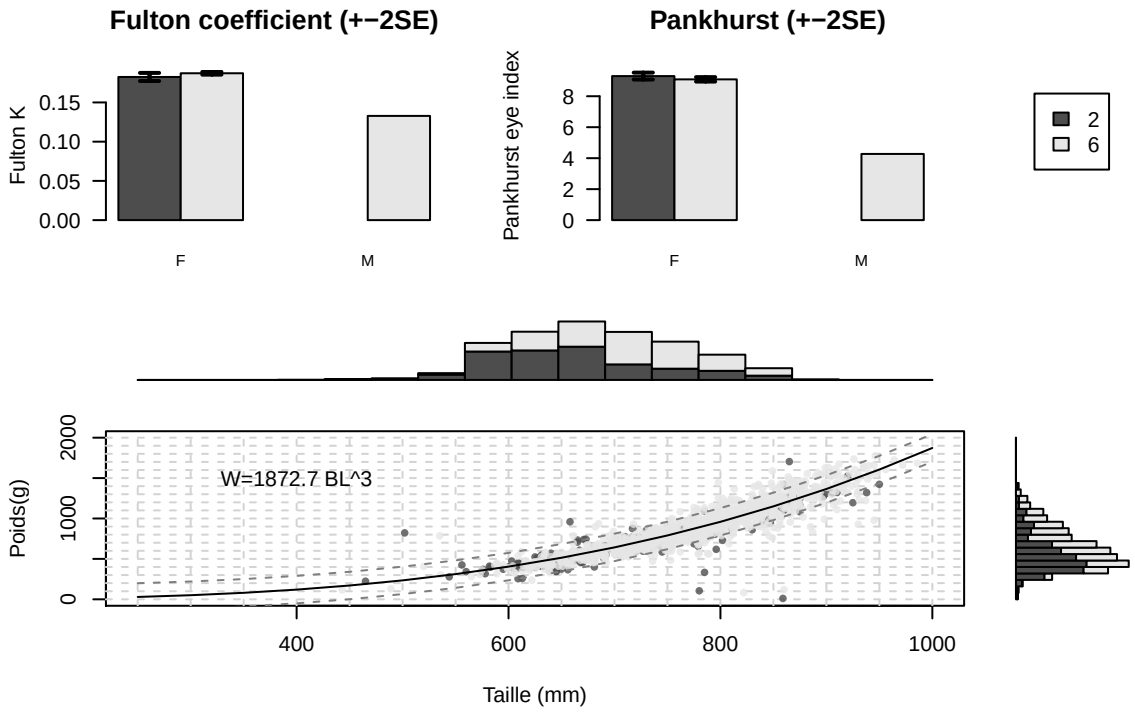
```
plot(r_mig_mult,plot.type="multiple",silent=TRUE)
```




**Silver eels**

This section provides a short example for the function calculating Durif's stages. Those maturity stages for silver eels are calculated from body characteristics. The dataset `coef_durif` corresponds to classification scores are calculated by multiplying the metrics BL = body length, W = weight, MD = mean eye diameter (Dv+Dh)/2, and FL length of the pectoral fin, with each parameter p as S=Constant+BL$p(bl)$+$W$p(W)... The function `fun_stage_durif` choose the stage achieving the highest score (Durif et al., 2009)

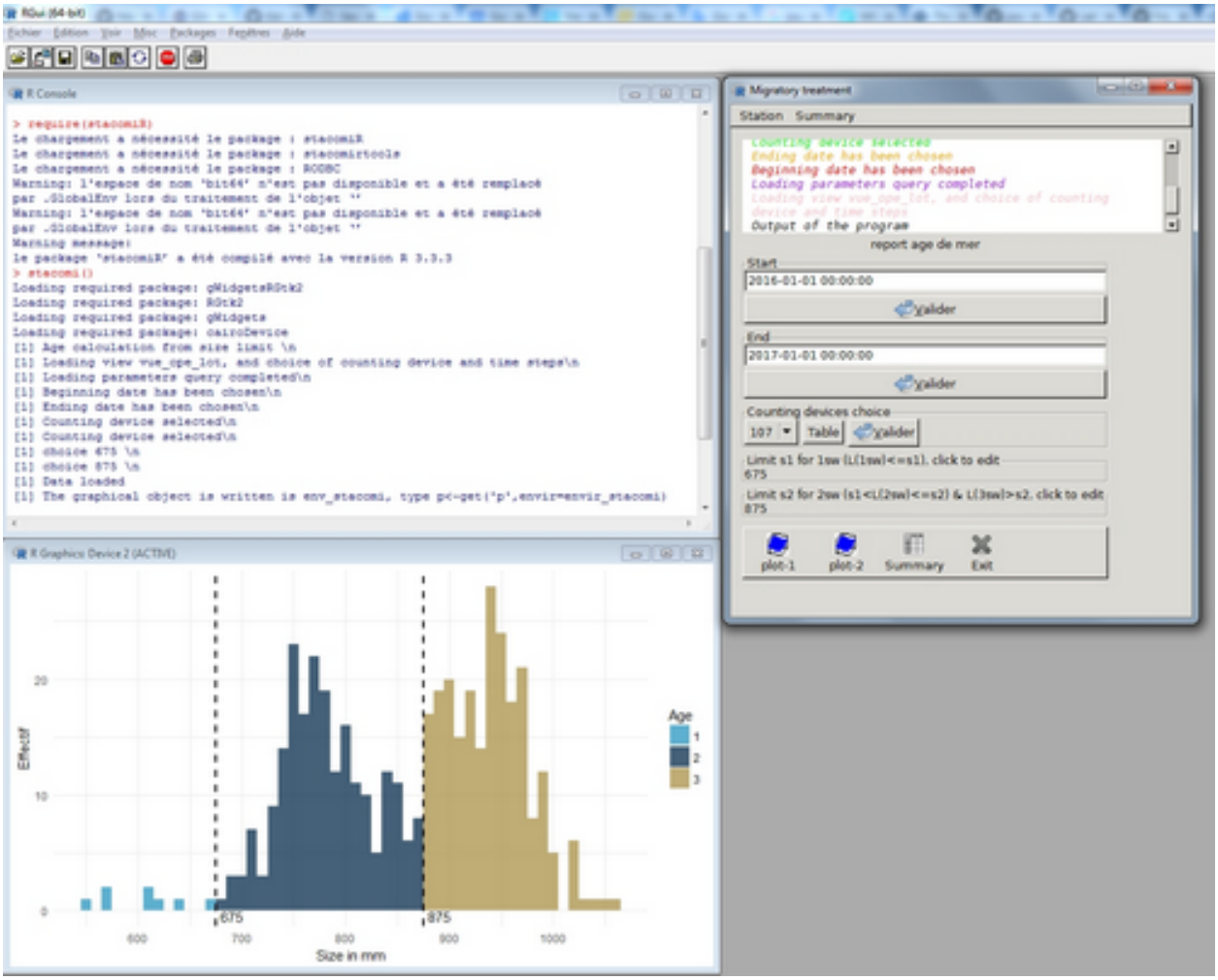
```
require(stacomiR)
data("coef_durif")
# load a dataset of class report_silver_eel with data slot already prepared
# here is an example of output
data("r_silver")
r_silver <- calcule(r_silver)
plot(r_silver, plot.type=3)
```




```
########################################
# To use the function fun_stage_durif manually
# create a matrix with columns BL","W","Dv","Dh","FL"
################################################
# here it is extracted from the data at hand
silver_eel<-as.matrix(r_silver@calcdata[[1]][,c("BL","W","Dv","Dh","FL")])
head(silver_eel) # to see the first lines
#>        BL    W    Dv    Dh    FL
#> 25710 830 1074  8.14  8.70 39.79
#> 25711 714  740  8.24  8.52 38.04
#> 25712 720  755  6.92  6.87 34.01
#> 25713 860 1101 10.53 10.43 44.47
#> 25714 716  752  7.42  8.76 33.78
#> 25715 690  622  7.83  9.25 29.58
stage <- fun_stage_durif(silver_eel) # apply the function to the matrix
stage[1:10] # look at the first 10 elements in vector silver
#>  25710  25711  25712  25713  25714  25715  25716  25717  25718  25719
#> "FIII" "FIII" "FIII"  "FIV" "FIII" "FIII"   "FV"   "FV" "FIII" "FIII"
```

### R-GTK2 graphical interface

The program is intended to be used by 'non experienced' R users. Launching `stacomi()` will create the interface. The interface looks like :




## License

The STACOMI project is released under GPL-2.