Units of measurement in FLR objects

12 September, 2017

The S4 classes defined in the **FLCore** package offer certain facilities for dealing with units of measurement when managing and operating with data and results. The mechanism is relatively simple, tailored to the usual quantities used in fisheries science, and attempts to only to simplify keeping track of the units in the data. No transformation of values takes place, in contrast with the functions offered by other R packages, like units.

Required packages

To follow this tutorial you should have installed the following packages:

- FLR: FLCore

You can do so as follows:

```
install.packages(c("FLCore"), repos="http://flr-project.org/R")
```

```
library(FLCore)
```

Storing units of measurement

Units of measurements can be specified in the `units` slot of the `FLQuant` and `FLCohort` classes, as a character vector of length 1

```
fq <- FLQuant(1.287, units="kg")
print(fq)
```

```
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1
  all 1.287

units:  kg
```

and for the `FLPar` class as a vector of length equal to the number of parameters (`params`)

```
FLPar(Linf=50, K=0.5, t0=-0.2, units=c("cm", "1/y", "y"))
```

```
An object of class "FLPar"
params
Linf    K    t0
50.0  0.5 -0.2
units:  cm 1/y y
```

The content of this slot can be extracted and altered, using the usual accessor and replacement methods, in this case `units` and `units<-`.

```
units(fq)
```

```
[1] "kg"
```

```
units(fq) <- "t"
```

```
summary(fq)
```

```
An object of class "FLQuant" with:
dim  :  1 1 1 1 1 1
quant:  quant
units:  t

Min    :  1.287
1st Qu.:  1.287
Mean   :  1.287
Median :  1.287
3rd Qu.:  1.287
Max    :  1.287
NAs    :  0 %
```

Computable units of measurement

Although the content of the `units` slot is not limited in any way, and any character string is valid, the most commonly-employed units have been tabulated so that the arithmetic operators defined for the `FLQuant` and `FLPar` classes are able to operate on them and return an object with the right units.

Numbers

Numbers of fish, e.g. catch-at-age, or any other type of count data should use one of the following ways of expressing the scale at which the data is stored when not done in direct count numbers, from tens to billions:

- 10, 100, ..., 1000000000
- 10^1, 10^2, ... 10^9
- 1e1, 1e2, ..., 1e9

Any of those three notations will be understood, but the one used when returning the result of a calculation will be as follows:

- 10, 100, 1000, 1e4, 1e5, 1e6, 1e7, 1e8 and 1e9

Direct counts do not need units of measurement, so `units` should be specified as an empty character vector (`""` or `character(1)`).

Weights

Two units are currently recognized: `kg` and `t`, corresponding to kilogram and (metric) tonne.

Mortality

The distinction of the various sources of, and different ways of calculating fish mortality, are traced in **FLCore** by assigning a particular set of units of measurement to each of those quantities. This is done despite the fact that all of them are in fact unitless.

- Fishing mortality: `f`, for instantaneous fishing mortality, or `hr`, for harvest rate, the proportion of biomass caught in a given time period.
- Natural mortality: `m`, for the instantaneous mortality due to natural causes.
- Total mortality: `z`, the sum of `m` and `f`

Currency

Objects containing economic information, such as prices of fish or costs of fishing, should use the corresponding ISO 4217 currency code, of which two are already present in the table used by `uom`: `EUR` for Euro, and `USD` for US Dollar.

```
FLQuant(1.2e6, units="EUR")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1
  all 1200000

units:  EUR
```

Time

Quantities such as fishing effort are often measured in time. Two standard units are currently included in the table that `uom` queries: `d` for days and `h` for hours. These could be used for either time at sea or actual fishing time.

Missing values

If the units of measurement have not been specified, they will be `NA`, *not available*. This is what the class constructors will default to if not other value is provided.

Unitless quantities

Multiple quantities used in fisheries science are unitless, for example proportions or ratios of other quantities. An empty string of length 1, "" or `character(1)`, will be then used.

Units of measurement when computing with FLQuants

Arithmetic operations on `FLQuant` objects will invoque the `uom` function to assign a value to the `units` slot of the returned object. This function takes three character arguments: the first for the operation being carried out, one of `/`, `*`, `+` or `-`, and the content of the `units` slot of each of the objects. For example, the addition of two objects with `kg` as units would lead to another object also with `kg` as units of measurement

```
FLQuant(9230, units="kg") + FLQuant(367, units="kg")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1
  all 9597

units:  kg
```

which internally is calling `uom` with the necessary arguments

```
uom("+", "kg", "kg")
```

```
[1] "kg"
```

The addition or substraction of two objects with the same units will return a new object that shares those units of measurement.

Products of numbers

When multiplying objects containing counts of different scale, the `units` slot is computed by scaling it to the product of both units, for example

- 10 * 1000 = 1e4
- 1e4 * 100 = 1e6

```
FLQuant(c(467.34, 345.33), units="1000") * FLQuant(c(2.4, 3.5), units="100")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1      2
  all 1121.6 1208.7

units:  1e5
```

Transformation of units

A small number of operations have been identified that naturally lead to the units of the result being different to those of any of the input objects. Again, no transformation of values takes place, and only the content of `units` slot is altered, as in the previous examples. The existing rules are:

- `1000 * kg = t` for any of the alllowed notations for the first quantity, and for all subsequent calculation, such as `t / 1000 = kg`.

```
FLQuant(c(467.34, 345.33), units="1000") * FLQuant(c(2.4, 3.5), units="kg")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1      2
  all 1121.6 1208.7

units:  t
```

- `f + m = z`, and subsequent combinations, like `z - m = f` etc.

```
FLQuant(runif(8, 0.1, 0.8), units="f") %+% FLQuant(0.2, units="m")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1       2       3       4       5       6       7       8
  all 0.51475 0.35683 0.82217 0.67726 0.35724 0.45620 0.91196 0.39663

units:  z
```

Products and division of composite units

Some cases in which composite units of measurement, those formed by a string showing the operation carried out, like `100 * kg` or `EUR / d` can, for certain operations, be simplified. Two possible cases exist that `uom` is able to handle. The first one is a product of a quantity with another in which the later has as units a division with the units of the first as denominator. For example, fixed costs in a fishery, expressed as a ratio of some currency per boat, `EUR / boat`, can then be multiplied by the number of boats operating per unit of time to obtain the total amount

```
FLQuant(126, units="boat") * FLQuant(2350, units="EUR / boat")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique
```

```
     year
quant 1
  all 296100

units:  EUR
```

This works even if the units expression of the first

```
FLQuant(126, units="boat / d") * FLQuant(2350, units="EUR / boat / d")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1
  all 296100

units:  EUR
```

```
FLQuant(2350, units="EUR / boat / d") * FLQuant(2350, units="boat / d")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1
  all 5522500

units:  EUR
```

Computing ratios

The division of any object by another with the same units of measurement will lead to an object with no units in the slot, i.e. a unitless ratio.

```
FLQuant(34170, units="t") / FLQuant(2.32, units="t")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1
  all 14728

units:
```

Dealing with missing units

When the units of measurement in one of the objects are missing, they are "NA", any operation will lead to an object with "NA" in the `units` slot.

```
FLQuant(34170, units="t") / FLQuant(2.32, units="NA")
```

```
An object of class "FLQuant"
An object of class "FLQuant"
, , unit = unique, season = all, area = unique

     year
quant 1
  all 14728

units:  NA
```

References

L. T. Kell, I. Mosqueira, P. Grosjean, J-M. Fromentin, D. Garcia, R. Hillary, E. Jardim, S. Mardle, M. A. Pastoors, J. J. Poos, F. Scott, R. D. Scott; FLR: an open-source framework for the evaluation and development of management strategies. *ICES J Mar Sci* 2007; 64 (4): 640-646. doi: 10.1093/icesjms/fsm012.

Wikipedia, The Free Encyclopedia, s.v. "ISO_4217" (accessed July 13, 2017), https://en.wikipedia.org/wiki/ISO_4217

More information

- You can submit bug reports, questions or suggestions on this tutorial at https://github.com/flr/doc/issues.
- Alternatively, send a pull request to https://github.com/flr/doc/.
- For more information on the FLR Project for Quantitative Fisheries Science in R, visit the FLR webpage: http://flr-project.org.

Software Versions

- R version 3.4.1 (2017-06-30)
- FLCore: 2.6.4.9002
- **Compiled**: Tue Sep 12 10:57:11 2017

License

This document is licensed under the Creative Commons Attribution-ShareAlike 4.0 International license.

Author information

Iago MOSQUEIRA. European Commission, DG Joint Research Centre, Directorate D - Sustainable Resources, Unit D.02 Water and Marine Resources, Via E. Fermi 2749, 21027 Ispra VA, Italy.