EUROPEAN COMMISSION

DIRECTORATE-GENERAL FOR FISHERIES AND MARITIME AFFAIRS
Conservation Policy
Research and Scientific analysis

Studies and Pilot projects for carrying out the common fisheries policy

Call for proposal ref : **FISH/2006/15 – lot 2**

Project no : **SI2.467814**

Project acronym : **COST**

Project title : Common tool for raising and estimating properties of statistical estimates derived from the Data Collection Regulation

User Manual

Version 1.1

July 2009

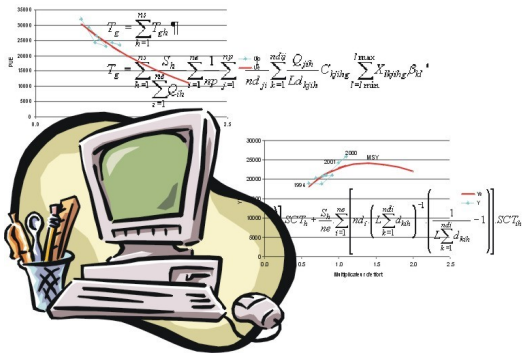
Common tool for raising and estimating properties of statistical estimates derived from the Data Collection Regulation

Studies and Pilot projects for carrying out the common fisheries policy

Call for proposal FISH/2006/15 – lot 2

Project no SI2.467814

"*This report does not necessarily reflect the views of the European Commission and in no way anticipates any future opinion of the Commission.*
The contents of this report may not be reproduced unless the source of the material is indicated.
This project has been financed by the European Commission."



COST

Supporting Web site : http://wwz.ifremer.fr/cost

Email address for assistance : not defined
Email address for general product information : jvigneau@ifremer.fr

Contents

1 Getting started **3**
1.1 Important concepts . . . 3
1.2 Installing COST packages . . . 3

2 Importing dataset **3**

3 Exploratory Data Analysis **8**
3.1 Basic methods . . . 8
3.2 Merging . . . 11
3.3 Subsetting . . . 12
3.4 Scatter plots . . . 14
3.4.1 Sampled weights and length structures . . . 14
3.4.2 Biological parameters . . . 18
3.5 Delta calculation . . . 21
3.6 Validation of the dataset . . . 23

4 Spatial Exploration **24**
4.1 Loading packages, data, and some basic manipulations . . . 24
4.2 Use of `spacePlot` . . . 25
4.3 Default plotting . . . 25
4.4 Plotting numeric variables by spatial strata . . . 27
4.5 Plotting COST objects . . . 29
4.6 Recoding variables and plotting CONSOLIDATED objects . . . 30
4.7 Plotting haul and trip positions . . . 33
4.8 Spatial data sets and other functions . . . 34

5 Stratification and recoding **37**
5.1 Determination of the stratification . . . 37
5.2 Consolidation of the dataset . . . 40

6 Reporting **42**

7 Preparing the estimations **43**

8 Raising discards samples **47**
8.1 Chosing the appropriate raising method . . . 47
8.2 Raising methods . . . 49
8.2.1 Raising by trips . . . 49
8.2.2 Raising by fishing operations . . . 52
8.2.3 Raising by an auxiliary variable . . . 55
8.2.4 Raising by fishing days . . . 57

9 Raising length structure from market sampling **60**
9.1 Analytical method . . . 60
9.2 resampling method . . . 64
9.3 Addition of dbeObjects . . . 67

10 Age structure from market sampling **68**
10.1 exploratory analysis of the Age Length Key . . . 68
10.2 Estimating age structure - Analytical . . . 71
10.3 Estimating age structure - Bootstrap . . . 73

11 Age structure without ALK **75**

12 Estimating biological parameters **78**
12.1 Individual weight . . . 78
12.2 Maturity stages . . . 81
12.3 Sex-ratio . . . 84

13 Modelling **87**
13.1 Modelling ALK . . . 87
13.2 Bayesian estimates . . . 89

14 Exportation of the estimates **95**
14.1 to InterCatch 96
14.2 to FLR 97
14.3 to CSV tables 99

15 Simulation program **99**

1 Getting started

1.1 Important concepts

COST has been developed using the most recent S4 objects, classes and methods, with the objective of developing generic and robust methods.
In this manual, you will learn how to carry out a complete analysis of your fisheries sampled data starting from a dataset already formated in the COST Data exchange Format (available on the project website http://wwz.ifremer.fr/cost). You may encounter difficulties in importing your data, although the COST team has chosen to relax the import field controls, but the main difficulty resides in the referential integrity. By referential integrity, we mean that all father tables has at least one son and all son tables have only one father. If this rule is broken, the importation to COST will fail. This is usually the main failure reason. Once the importation is succesful, the analysis proposed goes from exploratory data analysis to estimation of paramters, after validating the dataset and consolidating with sometimes the need to refine the stratification used.
The general concept is to build a R script with some editor such as Tinn-R[1]and comment as much as possible. What is stored for re-application in future years, is the script archiving all the actions you have initiated and all modifications, what so ever, you have done to the dataset. Be clear, concise and ensure that your script is re-usable by someone else. The COST team wishes you success in using our libraries.

1.2 Installing COST packages

All details of COST can be found on the project website : http://wwz.ifremer.fr/cost
The first action to do after installing R version 2.8.1 minimum is to install the COST packages. For doing so, go to 'Software/Package' web page and download all the packages on your computer
In R menu, click on 'Packages' and 'Install package(s) from zip file(s)'.
install all COST packages. The installation of packages is needed only once, and has to be repeated each time you want to install a new version of one package.
When opening a new script, the first lines of code must be

```
> library(COSTcore)

-------------------------------------------------------------------
 COSTcore - "The DCF Statistical Suite"
 (Version: 1.3-5. Built on: R 2.8.1; ; 2009-07-30 12:03:53; windows)
-------------------------------------------------------------------

> library(COSTeda)
> library(COSTdbe)
```

After loading all COST libraries, we promote the use of a dedicated folder for storing both the input and output files.

```
> path <- "." # This is 'My document' folder
> setwd(path)
```

You're ready for a COST analysis

2 Importing dataset

COST offers two possibilities for importing dataset, either from csv files, or from R dataframe objects. Both csv files or dataframes must strictly follow the FishFrame/COST format. The creation of datasets is done through the call of *csData*, *ceData* and *clData* method as shown below. The selection of the argument `check=TRUE` is strongly recommended, since it will give all details of the fields imported. There is no strong control over the naming of the variables in the first release of COST. Such a strict control, following naming convention is possible to implement in the near future to facilitate the comparison of datasets between countries, or exporting to external database.

[1]http://www.sciviews.org/Tinn-R/

Importing the three sets of data from csv files is done following the example below :

```
> data(soleData)
> sole_cs <- csData(tr = paste(path, "sole_tr.csv",
+ sep="/"),
+
+ hh = paste(path, "sole_hh.csv", sep="/"),
+
+ sl = paste(path, "sole_sl.csv", sep="/"),
+
+ hl = paste(path, "sole_hl.csv", sep="/"),
+
+ ca = paste(path, "sole_ca.csv", sep="/"),
+ check = TRUE)
```

```
        This is a cs Data object

tr_table
         $sampType is OK
         $landCtry is OK
         $vslFlgCtry is OK
         $year is OK
         $proj is OK
         $trpCode is a numeric variable and contains 47 unrecognised entries of character strings:
                 ARY178 DIL1196 DIL1197 ELR214 ELR219 FAD73 ISP2 JBO29 JEQ10 JEQ4 JEQ6 JEQ7 JEQ8 KMX5 K
         $vslLen is entirely NA
         $vslPwr is entirely NA
         $vslSize is entirely NA
         $vslType contains 28 NA values but is otherwise OK
         $harbour is entirely NA
         $foNum contains 28 NA values but is otherwise OK
         $daysAtSea contains 28 NA values but is otherwise OK
         $vslId contains 28 NA values but is otherwise OK
         $sampCtry is OK
         $sampMeth is OK
hh_table
         $sampType is OK
         $landCtry is OK
         $vslFlgCtry is OK
         $year is OK
         $proj is OK
         $trpCode is a numeric variable and contains 399 unrecognised entries of character strings:
                 ARY178 DIL1196 DIL1197 ELR214 ELR219 FAD73 ISP2 JBO29 JEQ10 JEQ4 JEQ6 JEQ7 JEQ8 KMX5 K
         $staNum is OK
         $foVal is OK
         $aggLev is OK
         $catReg is OK
         $sppReg is OK
         $date is OK
         $time is entirely NA
         $foDur is OK
         $latIni is OK
         $lonIni is OK
         $latFin is entirely NA
         $lonFin is entirely NA
         $area is OK
         $rect is OK
         $subRect is entirely NA
         $foDep is entirely NA
         $waterDep is OK
         $foCatNat is OK
         $foCatEu5 is OK
```

```
        $foCatEu6 contains 91 unrecognised entries of values:
                OTB_MOL_40_0_0 OTB_MOL_80_0_0 GNS_CRU_100_0_0 OTB_MOL_NA GTR_DEF_100_0_0 GTR_DEF_150_0
        $meshSize contains 36 NA values but is otherwise OK
        $selDev contains 107 NA values and contains values out of specified range 1 - 3
        $meshSizeSelDev is entirely NA
sl_table
        $sampType is OK
        $landCtry is OK
        $vslFlgCtry is OK
        $year is OK
        $proj is OK
        $trpCode is a numeric variable and contains 97 unrecognised entries of character strings:
                DIL1197 ELR214 ELR219 ISP2 JBO29 JEQ10 JEQ4 JEQ6 JEQ7 JEQ8 LIM1 NCA2 NIM207
        $staNum is OK
        $spp is OK
        $catchCat is OK
        $landCat is OK
        $commCatScl is OK
        $commCat is entirely NA
        $subSampCat is entirely NA
        $sex is entirely NA
        $wt is OK
        $subSampWt contains 42 NA values but is otherwise OK
        $lenCode is OK
hl_table
        $sampType is OK
        $landCtry is OK
        $vslFlgCtry is OK
        $year is OK
        $proj is OK
        $trpCode is a numeric variable and contains 200 unrecognised entries of character strings:
                DIL1197 ELR214 ELR219 ISP2 JBO29 JEQ10 JEQ4 JEQ6 JEQ7 JEQ8 LIM1 NCA2 NIM207
        $staNum is OK
        $spp is OK
        $catchCat is OK
        $landCat is OK
        $commCatScl is OK
        $commCat is entirely NA
        $subSampCat is entirely NA
        $sex is entirely NA
        $lenCls is OK
        $lenNum is OK
ca_table
        $sampType is OK
        $landCtry is OK
        $vslFlgCtry is OK
        $year is OK
        $proj is OK
        $trpCode is OK
        $staNum is OK
        $quarter is OK
        $month is OK
        $spp is OK
        $sex is OK
        $catchCat is OK
        $landCat is OK
        $commCatScl is entirely NA
        $commCat is entirely NA
        $stock is entirely NA
        $area is OK
        $rect is OK
        $subRect is entirely NA
```

```
         $lenCls is OK
         $age contains 53 NA values but is otherwise OK
         $fishId is OK
         $lenCode is OK
         $ageMeth is OK
         $plusGrp is OK
         $otoWt is entirely NA
         $otoSide is entirely NA
         $indWt is OK
         $matMeth is OK
         $matScale is OK
         $matStage contains 10 NA values but is otherwise OK


> sole_ce <- ceData(ce = paste(path, "sole_ce.csv",
+ sep="/"), check = TRUE)

        This is a ce Data object

ce_table
         $vslFlgCtry is OK
         $year is OK
         $quarter is OK
         $month contains 48 NA values but is otherwise OK
         $area is OK
         $rect is entirely NA
         $subRect is entirely NA
         $foCatNat is entirely NA
         $foCatEu5 contains 35 unrecognised entries of values:
                 DRB_DEF GN_DEF GNS_MOL LH_DEF LL_DEF PPI_MOL TBS_DEF TTB_MOL
         $foCatEu6 is entirely NA
         $harbour is entirely NA
         $vslLenCat is entirely NA
         $trpNum is OK
         $foNum is entirely NA
         $foDur is entirely NA
         $effKwDays is entirely NA
         $effGtDays is entirely NA
         $daysAtSea is entirely NA


> sole_cl <- clData(cl = paste(path, "sole_cl.csv",
+ sep="/"), check = TRUE)

        This is a cl Data object

cl_table
         $landCtry is OK
         $vslFlgCtry is OK
         $year is OK
         $quarter is OK
         $month contains 229 NA values but is otherwise OK
         $area is OK
         $rect is entirely NA
         $subRect is entirely NA
         $taxon is OK
         $landCat is OK
         $commCatScl is OK
         $commCat contains 116 NA values but is otherwise OK
         $foCatNat is entirely NA
         $foCatEu5 contains 133 unrecognised entries of values:
                 DRB_DEF GN_DEF GNS_MOL LH_DEF LL_DEF PPI_MOL TBS_DEF TTB_MOL
         $foCatEu6 is entirely NA
```

```
$harbour is entirely NA
$vslLenCat is entirely NA
$unallocCatchWt is entirely NA
$misRepCatchWt is entirely NA
$landWt is OK
$landMult is entirely NA
$landValue is entirely NA
```

Notice the messages provided by the internal control procedure, on the content of every fields of the imported datasets. COST does not consider unrecognised entries as blocking the system. This is meant to ease the use of COST methods at a national level, without being picky on the naming conventions and other reference tables used. It will be easy in a near future to consider an optional function controlling the contents of the field following an international agreed convention, in order to facilitate international use of the data.
The second method consists of importing the three sets of data from R dataframes following the example below. With the same data format as the csv files, you will notice the need to remove the first field when using the dataframe format. Here we use `check=FALSE` to avoid repeating the same display as above :

```
> sole_cs <- csData(tr = tr[,-1], hh = hh[,-1], sl = sl[,-1],
+ hl = hl[,-1], ca = ca[,-1], check = FALSE)
> sole_ce <- ceData(ce = ce[,-1], check = FALSE)
> sole_cl <- clData(cl = cl[,-1], check = FALSE)
```

Once a dataset is imported into R, there should be 3 sets of tables available, namely the commercial sampling (*csData*) dataset, the total landings dataset (*clData*) and optionnally, the total effort dataset (*ceData*).

```
> ls(pattern="sole")

[1] "sole_ce" "sole_cl" "sole_cs"
```

Let's have a look at these objects. First, the COST objects have slots containing items, that you can address by typing `costObjectslot$item`. Here are the slots of the COST object sole_cs, the items of the slot sole_cs@tr and the modalities of the field sampType:

```
> slotNames(sole_cs)

[1] "desc" "tr"   "hh"   "sl"   "hl"   "ca"

> names(sole_cs@tr)

 [1] "sampType"   "landCtry"   "vslFlgCtry" "year"       "proj"       "trpCode"
 [7] "vslLen"     "vslPwr"     "vslSize"    "vslType"    "harbour"    "foNum"
[13] "daysAtSea"  "vslId"      "sampCtry"   "sampMeth"

> unique(sole_cs@tr$sampType)

[1] "S" "M" "V"
```

The importation phase is considered as one of the most critical phase of the analysis. The need to combine data sometimes from three databases to reunify discards, market and biological sampling often leads to data integrity issues. It is therefore very important to spend as much time as necessary to prepare the datasets upstream at a national level. The benefit of COST is to enable to use of data from any sources in complement to each other, after control and validation of the appropriateness of such a combination. The most usual example is the possible use of market sampling together with the retained fraction of on-board sampling within an area, a time stratum and for a given metier, and the use of independent age information to estimate the age structure of the landings. Good luck!

3 Exploratory Data Analysis

The data exploratory analysis can be done at any phase of the analysys within COST, but the objectives of the methods developed in this section are to help the user to shift the dataset from RAW to VALIDATED status. To this aim, the methods developed are meant to detect any errors or mis-reporting of information. Once an error is detected, the best solution is to make the correction into the centralised database and re-export the dataset into R. **How to be sure the outlier spotted in a scatter plot is an error or not, and the need to delete or correct it is not in the remit of the COST project**. However, the methods are available to delete or modify specific samples in COST environment, in order to test the impact on the final results. Any modification or deletion of parts of the datasets used, in the COST environment will not affect the original datasets, and the replay of the actions in the future will only be possible if all the coding used is kept in the script with the appropriate comments.

3.1 Basic methods

The first actions to perform is to have a look at the data. The basic R methods like `head`, `tail`, `summary` have been adapted to the COST object. They are meant to display respectively the 5 first rows, the 5 last rows and the 5 first occurences and modalities of each factors for the datasets given as an argument. A new `summary` method, `cstSummary`, has been implemented specifically adapted to the COST objects. The `cstSummary` method allows some flexibility for adapting the information displayed, `str` is also a convenient way of displaying the contents of the fields and `dim` is a well-known method for counting the number of dimension of the dataframes .

```
> head(sole_cs)
> tail(sole_cl)
> summary(sole_cs@hh)

> cstSummary(sole_cs, tab="hh", sizeMax=6,
+ except="date")

 sampType landCtry  vslFlgCtry   year           proj        trpCode        staNum    foVal   aggLev  ca
 S:399    FRA:447   FRA:447    2006:447   Obsmer  :399   NIM207 : 68   999    : 48   V:447   H:399   Al
 M: 48                                   MarkSamp: 48   ELR214 : 56   1      : 19           T: 48   No
                                                        NCA2   : 42   2      : 19                   La
                                                        KMX6   : 33   3      : 18
                                                        DIL1196: 30   4      : 16
                                                        (Other):218   (Other):327

     latFin        lonFin         area          rect     subRect     foDep        waterDep
 Min.   : NA   Min.   : NA   27.7.d :177   29F0   : 55   NA's:447   NA's:447   40     : 50   7E-OTB-Div
 1st Qu.: NA   1st Qu.: NA   27.7.e : 99   28E5   : 35                         100    : 31   7G-OTB-Div
 Median : NA   Median : NA   27.7.g : 76   27E9   : 30                         30     : 19   7D-OTB-Sei
 Mean   :NaN   Mean   :NaN   27.4.c : 52   31E2   : 25                         80     : 19   7D-OTB-Mer
 3rd Qu.: NA   3rd Qu.: NA   27.7.h : 20   (Other):254                         (Other):280   (Other)
 Max.   : NA   Max.   : NA   (Other): 23   NA's   : 48                         NA's   : 48   NA's
 NA's   :447   NA's   :447
 meshSizeSelDev
 NA's:447

> str(sole_cs)

Formal class 'csData' [package "COSTcore"] with 6 slots
  ..@ desc: chr "Unknown stock"
  ..@ tr  :'data.frame':        95 obs. of  16 variables:
  .. ..$ sampType  : chr [1:95] "S" "S" "S" "S" ...
  .. ..$ landCtry  : chr [1:95] "FRA" "FRA" "FRA" "FRA" ...
  .. ..$ vslFlgCtry: chr [1:95] "FRA" "FRA" "FRA" "FRA" ...
  .. ..$ year      : num [1:95] 2006 2006 2006 2006 2006 ...
  .. ..$ proj      : chr [1:95] "Obsmer" "Obsmer" "Obsmer" "Obsmer" ...
  .. ..$ trpCode   : chr [1:95] "ARY178" "DIL1196" "DIL1197" "ELR214" ...
  .. ..$ vslLen    : num [1:95] NA NA NA NA NA NA NA NA NA NA ...
  .. ..$ vslPwr    : num [1:95] NA NA NA NA NA NA NA NA NA NA ...
  .. ..$ vslSize   : num [1:95] NA NA NA NA NA NA NA NA NA NA ...
```

```
.. ..$ vslType   : chr [1:95] "1" "1" "1" "1" ...
.. ..$ harbour   : chr [1:95] NA NA NA NA ...
.. ..$ foNum     : num [1:95] 27 30 5 56 13 3 30 23 3 11 ...
.. ..$ daysAtSea : num [1:95] 4 5 2 13 10 4 6 5 1 2 ...
.. ..$ vslId     : num [1:95] 98 85 21 42 43 41 40 22 108 48 ...
.. ..$ sampCtry  : chr [1:95] "FRA" "FRA" "FRA" "FRA" ...
.. ..$ sampMeth  : chr [1:95] "Observer" "Observer" "Observer" "Observer" ...
..@ hh  :'data.frame':        447 obs. of  29 variables:
.. ..$ sampType      : chr [1:447] "S" "S" "S" "S" ...
.. ..$ landCtry      : chr [1:447] "FRA" "FRA" "FRA" "FRA" ...
.. ..$ vslFlgCtry    : chr [1:447] "FRA" "FRA" "FRA" "FRA" ...
.. ..$ year          : num [1:447] 2006 2006 2006 2006 2006 ...
.. ..$ proj          : chr [1:447] "Obsmer" "Obsmer" "Obsmer" "Obsmer" ...
.. ..$ trpCode       : chr [1:447] "ARY178" "ARY178" "ARY178" "ARY178" ...
.. ..$ staNum        : num [1:447] 1 2 3 4 5 6 7 8 9 10 ...
.. ..$ foVal         : chr [1:447] "V" "V" "V" "V" ...
.. ..$ aggLev        : chr [1:447] "H" "H" "H" "H" ...
.. ..$ catReg        : chr [1:447] "All" "All" "Non" "All" ...
.. ..$ sppReg        : chr [1:447] "All" "All" "Non" "All" ...
.. ..$ date          : chr [1:447] "2006-04-03" "2006-04-03" "2006-04-03" "2006-04-03" ...
.. ..$ time          : chr [1:447] NA NA NA NA ...
.. ..$ foDur         : num [1:447] 150 180 165 195 190 180 195 180 165 180 ...
.. ..$ latIni        : num [1:447] 50.1 51.2 51.1 51 51 ...
.. ..$ lonIni        : num [1:447] 1.62 1.79 1.67 1.57 1.67 ...
.. ..$ latFin        : num [1:447] NA NA NA NA NA NA NA NA NA NA ...
.. ..$ lonFin        : num [1:447] NA NA NA NA NA NA NA NA NA NA ...
.. ..$ area          : chr [1:447] "27.7.d" "27.4.c" "27.4.c" "27.4.c" ...
.. ..$ rect          : chr [1:447] "29F1" "31F1" "31F1" "31F1" ...
.. ..$ subRect       : chr [1:447] NA NA NA NA ...
.. ..$ foDep         : num [1:447] NA NA NA NA NA NA NA NA NA NA ...
.. ..$ waterDep      : num [1:447] 40 40 40 40 40 40 40 40 40 40 ...
.. ..$ foCatNat      : chr [1:447] "7D-OTB-Merlan" "4C-OTB-Merlan" "4C-OTB-Merlan" "4C-OTB-Merlan" ..
.. ..$ foCatEu5      : chr [1:447] "OTB_DEF" "OTB_DEF" "OTB_DEF" "OTB_DEF" ...
.. ..$ foCatEu6      : chr [1:447] "OTB_DEF_80_0_0" "OTB_DEF_80_0_0" "OTB_DEF_80_0_0" "OTB_DEF_80_0_0
.. ..$ meshSize      : num [1:447] 80 80 80 80 80 80 80 40 80 80 ...
.. ..$ selDev        : chr [1:447] "0" "0" "0" "0" ...
.. ..$ meshSizeSelDev: num [1:447] NA NA NA NA NA NA NA NA NA NA ...
..@ sl  :'data.frame':        293 obs. of  17 variables:
.. ..$ sampType  : chr [1:293] "S" "S" "S" "S" ...
.. ..$ landCtry  : chr [1:293] "FRA" "FRA" "FRA" "FRA" ...
.. ..$ vslFlgCtry: chr [1:293] "FRA" "FRA" "FRA" "FRA" ...
.. ..$ year      : num [1:293] 2006 2006 2006 2006 2006 ...
.. ..$ proj      : chr [1:293] "Obsmer" "Obsmer" "Obsmer" "Obsmer" ...
.. ..$ trpCode   : chr [1:293] "DIL1197" "DIL1197" "DIL1197" "DIL1197" ...
.. ..$ staNum    : num [1:293] 1 1 2 2 3 3 4 4 17 19 ...
.. ..$ spp       : chr [1:293] "Solea solea" "Solea solea" "Solea solea" "Solea solea" ...
.. ..$ catchCat  : chr [1:293] "LAN" "DIS" "LAN" "DIS" ...
.. ..$ landCat   : chr [1:293] "HUC" "HUC" "HUC" "HUC" ...
.. ..$ commCatScl: chr [1:293] "EU" "EU" "EU" "EU" ...
.. ..$ commCat   : chr [1:293] NA NA NA NA ...
.. ..$ subSampCat: chr [1:293] NA NA NA NA ...
.. ..$ sex       : chr [1:293] NA NA NA NA ...
.. ..$ wt        : num [1:293] 11000 10000 26321 12000 73000 ...
.. ..$ subSampWt : num [1:293] NA 1560 NA 2570 NA ...
.. ..$ lenCode   : chr [1:293] "cm" "cm" "cm" "cm" ...
..@ hl  :'data.frame':        1409 obs. of  16 variables:
.. ..$ sampType  : chr [1:1409] "S" "S" "S" "S" ...
.. ..$ landCtry  : chr [1:1409] "FRA" "FRA" "FRA" "FRA" ...
.. ..$ vslFlgCtry: chr [1:1409] "FRA" "FRA" "FRA" "FRA" ...
.. ..$ year      : num [1:1409] 2006 2006 2006 2006 2006 ...
.. ..$ proj      : chr [1:1409] "Obsmer" "Obsmer" "Obsmer" "Obsmer" ...
.. ..$ trpCode   : chr [1:1409] "DIL1197" "DIL1197" "DIL1197" "DIL1197" ...
```

```
  .. ..$ staNum    : num [1:1409] 1 1 1 1 2 2 2 2 2 2 ...
  .. ..$ spp       : chr [1:1409] "Solea solea" "Solea solea" "Solea solea" "Solea solea" ...
  .. ..$ catchCat  : chr [1:1409] "DIS" "DIS" "DIS" "DIS" ...
  .. ..$ landCat   : chr [1:1409] "HUC" "HUC" "HUC" "HUC" ...
  .. ..$ commCatScl: chr [1:1409] "EU" "EU" "EU" "EU" ...
  .. ..$ commCat   : chr [1:1409] NA NA NA NA ...
  .. ..$ subSampCat: chr [1:1409] NA NA NA NA ...
  .. ..$ sex       : chr [1:1409] NA NA NA NA ...
  .. ..$ lenCls    : num [1:1409] 180 190 200 210 160 170 180 190 200 210 ...
  .. ..$ lenNum    : num [1:1409] 3 6 5 1 1 1 3 8 7 6 ...
  ..@ ca  :'data.frame':        1627 obs. of  31 variables:
  .. ..$ sampType  : chr [1:1627] "V" "V" "V" "V" ...
  .. ..$ landCtry  : chr [1:1627] "FRA" "FRA" "FRA" "FRA" ...
  .. ..$ vslFlgCtry: chr [1:1627] "FRA" "FRA" "FRA" "FRA" ...
  .. ..$ year      : num [1:1627] 2006 2006 2006 2006 2006 ...
  .. ..$ proj      : chr [1:1627] "BioPar" "BioPar" "BioPar" "BioPar" ...
  .. ..$ trpCode   : chr [1:1627] "12" "12" "12" "12" ...
  .. ..$ staNum    : num [1:1627] 999 999 999 999 999 999 999 999 999 999 ...
  .. ..$ quarter   : num [1:1627] 2 2 2 2 2 2 2 2 2 2 ...
  .. ..$ month     : num [1:1627] 4 4 4 4 4 4 4 4 4 4 ...
  .. ..$ spp       : chr [1:1627] "Solea solea" "Solea solea" "Solea solea" "Solea solea" ...
  .. ..$ sex       : chr [1:1627] "M" "M" "M" "F" ...
  .. ..$ catchCat  : chr [1:1627] "LAN" "LAN" "LAN" "LAN" ...
  .. ..$ landCat   : chr [1:1627] "HUC" "HUC" "HUC" "HUC" ...
  .. ..$ commCatScl: chr [1:1627] NA NA NA NA ...
  .. ..$ commCat   : chr [1:1627] NA NA NA NA ...
  .. ..$ stock     : chr [1:1627] NA NA NA NA ...
  .. ..$ area      : chr [1:1627] "27.7.d" "27.7.d" "27.7.d" "27.7.d" ...
  .. ..$ rect      : chr [1:1627] "28E9" "28E9" "28E9" "28E9" ...
  .. ..$ subRect   : chr [1:1627] NA NA NA NA ...
  .. ..$ lenCls    : num [1:1627] 330 340 320 320 350 340 330 340 320 330 ...
  .. ..$ age       : num [1:1627] 5 7 7 4 7 9 7 5 4 5 ...
  .. ..$ fishId    : num [1:1627] 1 2 3 4 5 6 7 8 9 10 ...
  .. ..$ lenCode   : chr [1:1627] "cm" "cm" "cm" "cm" ...
  .. ..$ ageMeth   : chr [1:1627] "Otoliths - slides with transmitted light" "Otoliths - slides with tra
  .. ..$ plusGrp   : chr [1:1627] "-" "-" "-" "-" ...
  .. ..$ otoWt     : num [1:1627] NA NA NA NA NA NA NA NA NA NA ...
  .. ..$ otoSide   : chr [1:1627] NA NA NA NA ...
  .. ..$ indWt     : num [1:1627] 355 360 339 416 412 411 347 411 353 365 ...
  .. ..$ matMeth   : chr [1:1627] "Visual" "Visual" "Visual" "Visual" ...
  .. ..$ matScale  : chr [1:1627] "1-7" "1-7" "1-7" "1-7" ...
  .. ..$ matStage  : chr [1:1627] "2" "2" "2" "5" ...
```

```
> dim(sole_cs)
```

```
$tr
[1] 95 16

$hh
[1] 447  29

$sl
[1] 293  17

$hl
[1] 1409   16

$ca
[1] 1627   31
```

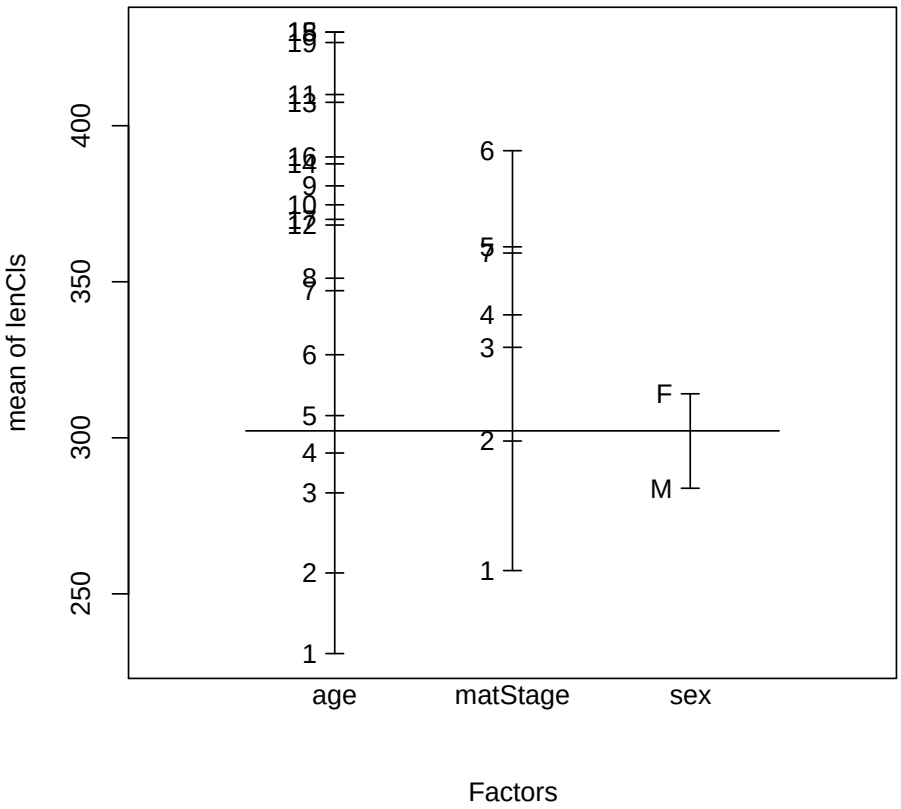
Other examples of use of `cstSummary` :

```
> cstSummary(sole_cs, tab = "tr", except
+ = c("trpCode","vslId"))
```

```
> cstSummary(sole_cs, tab = "hh", except
+ = c("trpCode","date"))
> cstSummary(sole_cs, tab = "sl", except
+ = c("trpCode"))
> cstSummary(sole_cs, tab = "hl", except
+ = c("trpCode","lenCls"))
> cstSummary(sole_cs, tab = "ca", sizeMax
+ = 15)
> cstSummary(sole_ce, sizeMax = 10)
```

One convenient check of the consistency of the biological parameters dataset is to use `csPlot.design`. this flexible method shows the median of a variable for all the modalities of factors. What is expected is to see is a coherent increase of the variable, ex: weight increases with length classes. The arguments to be passed in the function are the validated object and the name of the species ("all" by default).

```
> csPlot.design(sole_cs)
```



The outcome results in two graphics in two pages. The first page shows the mean of modalities of Age, Maturity stages and Sex for the variable individual weight, and the second page (figure above) for the variable length class (which is equal to mean length since the table ca contains one individual per record).

3.2 Merging

COSTeda contains a number of addition data manipulations functions that may be useful to the user in handling COST objects. The function `mergecsData` adds the pertinent space, time and technical fields to each data frame within the original COST objects to enable the individual data frames to be used in isolation. For example in a *csData* object the *hl* table has no information on the area, gears and time.

```
> data(cod2004cs)
> newhl <-mergecsData(cod2004cs)@hl
```

Now the data frame *newhl* has these fields replicated appropriately from the corresponding *tr* and *hh* and *sl* tables:

```
> head(newhl)

  sampType landCtry vslFlgCtry year proj   trpCode staNum          spp catchCat landCat commCatScl comm
1        M      SCO        SCO 2004 <NA> 01020104A      1 Gadus morhua      LAN     HUC   Unsorted
2        M      SCO        SCO 2004 <NA> 01020104A      1 Gadus morhua      LAN     HUC   Unsorted
3        M      SCO        SCO 2004 <NA> 01020104A      1 Gadus morhua      LAN     HUC   Unsorted
4        M      SCO        SCO 2004 <NA> 01020104A      1 Gadus morhua      LAN     HUC   Unsorted
5        M      SCO        SCO 2004 <NA> 01020104A      1 Gadus morhua      LAN     HUC   Unsorted
6        M      SCO        SCO 2004 <NA> 01020104A      1 Gadus morhua      LAN     HUC   Unsorted
  quarter month yearfromdate
1       1     1         2004
2       1     1         2004
3       1     1         2004
4       1     1         2004
5       1     1         2004
6       1     1         2004
```

So, for example, the mean length class by gear is:

```
> tapply(rep(newhl$lenCls,newhl$lenNum),
+ rep(newhl$foCatEu5,newhl$lenNum),mean)

 OTB_CRU  OTB_DEF  PTB_DEF  SSC_DEF
487.0905 546.5477 572.3444 504.7053
```

3.3 Subsetting

The dataset imported may contain much more information than needed for the analysis. Depending on the focus you want to give, you will need to subset on strata, trips, species, or any other field. The package COSTcore contains a generic function `subset` which is highly versatile and allows sub-setting by any variable and target values. The subsetting function will make a difference whether the field given for subsetting is part of the trip table (tr), fishing operation table (hh) or tables situated under these tables (ca, sl and hl). For example, subsetting on a species (in table sl) will necessitate a specific function `subsetSpp`, since in this case, we need to keep all information from sampled trips and fishing operations, to account for 0-values. Moreover, the subsetting function goes together with the table consistency check function (`tabConsist`), which enables to check the consequences of subsetting. The example below is subsetting on an area:

```
> tabConsist(list(sole_cs,sole_ce,sole_cl),"area")

       27.4.c 27.7.a 27.7.d 27.7.e 27.7.f 27.7.g 27.7.h 27.7.j
CS1_hh x      x      x      x      x      x      x      x
CS1_ca               x      x
CE1                  x      x
CL1                  x      x

> s7d_cs <- subset(sole_cs, area == "27.7.d",
+ table = "hh")
> tabConsist(list(s7d_cs,sole_ce,sole_cl),"area")

       27.7.d 27.7.e
CS1_hh x
CS1_ca x      x
CE1    x      x
CL1    x      x
```

You can notice that the table hh, i.e. sampled trips contain information from much more areas than the statistics tables. The subsetting method has truncated the dataset. One way of checking the consequence of the truncation is to call the method `dim` on both objects *`sole_cs`* and *`s7d_cs`*. By default, the selected table for subsetting is tr; if the selected field is part of another table, you need to specify it with the argument `table="hh"` for example. Subsetting one species will be done by calling the `subsetSpp` method:

```
> tabConsist(list(sole_cs,sole_ce,sole_cl),"spp")

       Solea solea
CS1_sl x
CS1_hl x
CS1_ca x

> s7d_cs <- subsetSpp(sole_cs, spp == "Solea solea")
> tabConsist(list(s7d_cs,sole_ce,sole_cl),"spp")

       Solea solea
CS1_sl x
CS1_hl x
CS1_ca x
```

The dataset contains only one species in this case, so the action does not change the original dataset. One important concept to keep in mind is that the raising and calculation method will all stard from the cs object and search for relevant information in the tables cl, ce and ca. In fact, in the raising and precision calculation methods, there are subsetting methods applied to make these information match from each of the tables. Therefore, there is no need to subset every tables, although it may be found more elegant to do so. As an example

```
> s7d_ce <- subset(sole_ce, area == "27.7.d")
> s7d_cl <- subset(sole_cl, area == "27.7.d")
> tabConsist(list(s7d_cs,s7d_ce,s7d_cl),"area")

       27.4.c 27.7.a 27.7.d 27.7.e 27.7.f 27.7.g 27.7.h 27.7.j
CS1_hh x      x      x      x      x      x      x      x
CS1_ca               x      x
CE1                  x
CL1                  x
```

Important : It is at the stage of subsetting that the user will define if the objective of the analysis is to consider only the market sampling (`samptype="M"`), only the on-board sampling (`sampType="S"`) or use both information. Concerning the age information, there may be the case where the information comes from purchasing of fish (`sampType="V"`) or from surveys (`sampType="?"`). In the example below, we will use the sampled length from market sampling and age information from purchase of fish.

```
> tabConsist(list(s7d_cs,s7d_ce,s7d_cl),"sampType")

       M S V
CS1_tr x x x
CS1_hh x x
CS1_sl x x
CS1_hl x x
CS1_ca     x

> s7d_cs <- subset(s7d_cs, sampType %in% c("M","V"))
> tabConsist(list(s7d_cs,s7d_ce,s7d_cl),"sampType")

       M V
CS1_tr x x
CS1_hh x
CS1_sl x
CS1_hl x
CS1_ca   x
```

With the same dataset, you may then write one script for the estimation of length and age structure of the catches or landings for one country, and one script for the length and age structure of the catches or landings per tuning fleet. This flexibility means that the header of each script must contain all information describing the actions that will be carried out, for archiving objectives.
There are also a number of bespoke subsetting functions for the more commonly desired actions e.g. `subSetSpp` and `subSetTarget` within COSTeda package. Though most of these operations can be performed by the generic function `subset` in some instances the bespoke functions may be quicker, especially with large data sets

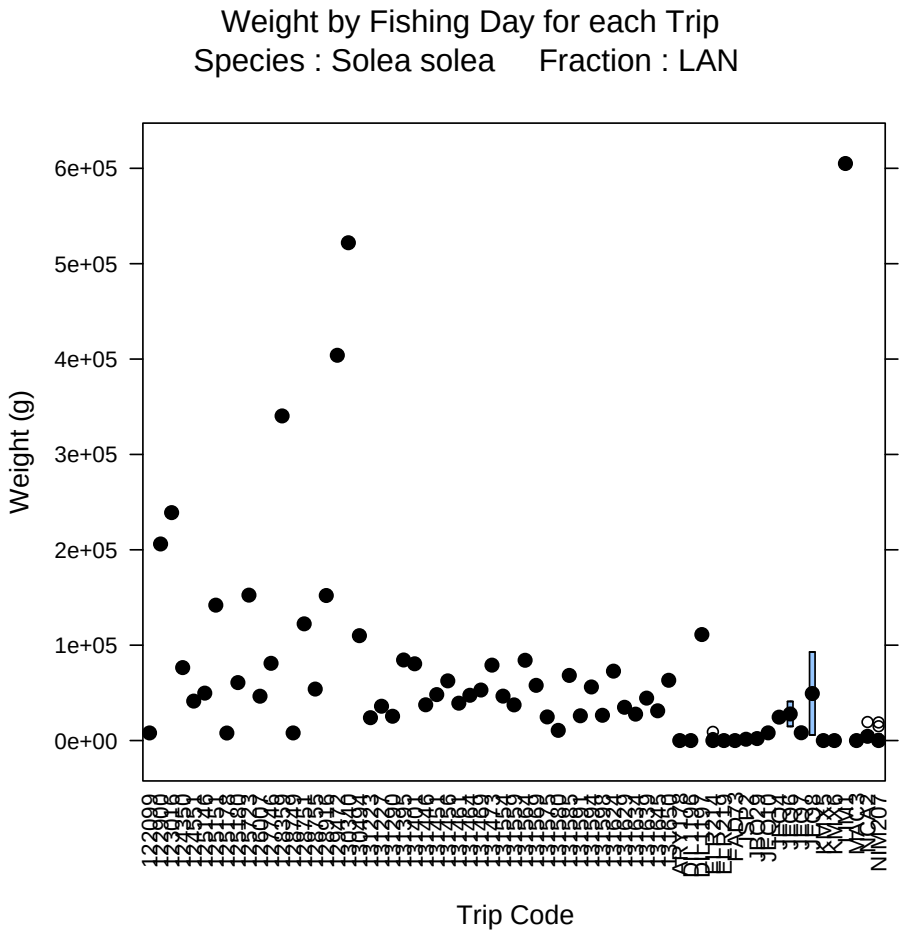
```
> codDEF <-subSetTarget(cod2004cs,"DEF")

csData subset by target assemblage DEF
New data set consists of 255 trip records
255 haul records
11371 length records
and 10244 age records

> table(codDEF@hh$foCatEu5)

OTB_DEF PTB_DEF SSC_DEF
    130      62      63
```

3.4 Scatter plots

3.4.1 Sampled weights and length structures

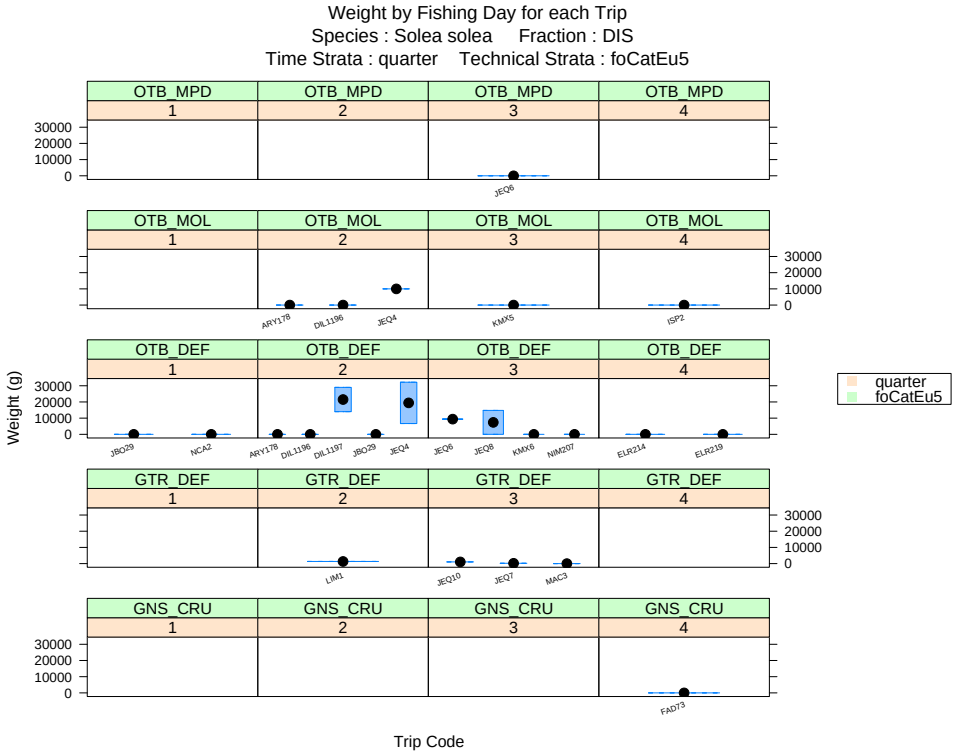
- Different methods have been implemented to produce scatter plots adapted to the fields covered by COST, i.e. discards, market or biological parameters sampling. For discards and market sampling, the main scatter plots are showing the sampled weight `landisVol` and the length structures of the samples `lenDisPlot`. The mandatory arguments are the *`csData`* object and the name of the species. One of the options available for `landisVol` allows to produce graphs of the discarded part (`fraction = "DIS"`) or the landed part (`fraction = "LAN"` by default).

```
> wTrip <- landisVol(sole_cs, species="Solea solea")
> boxplot(wTrip)
```

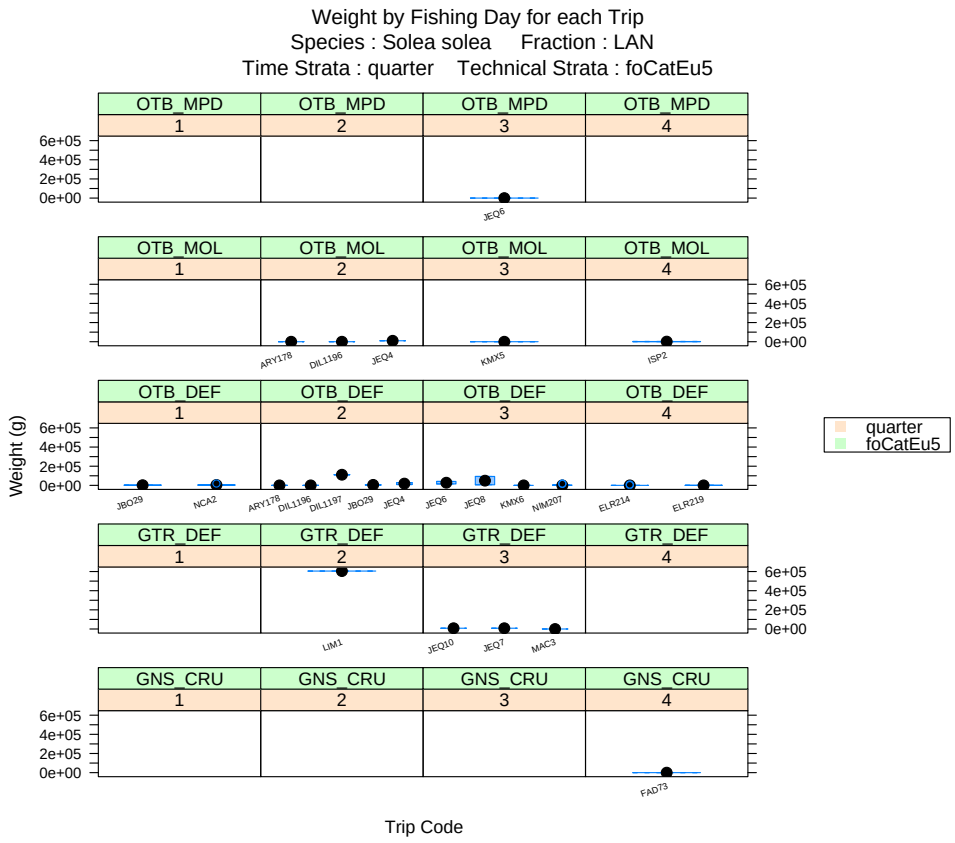


Two optional arguments are available for `landisVol`, namely `strDef` and `sampPar`. `strDef` is the object describing the stratification (see part 4) and `sampPar` is a Boolean specifying if the 0-values are accounted for or not. The result of landisVol is a list containing a lot of information that may be further used by the experienced user. The methods `plot` and `boxplot` recognise the `landisVol` object and are able to produce graphs directly. Let's see some more examples of use, with different arguments on the on-board information from the dataset:

```
> s7dsurv_cs <- subset(sole_cs, sampType %in% c("S"))
> # Taking the stratification into account:
>
> strD <- strIni(timeStrata="quarter",techStrata="foCatEu5")
> # Boxplot of the landings
>
> LDVpd <- landisVol(s7dsurv_cs,species="Solea solea",
+ strDef = strD, fraction = "DIS")
> boxplot(LDVpd, cex.lab=0.5, rot=20)
```



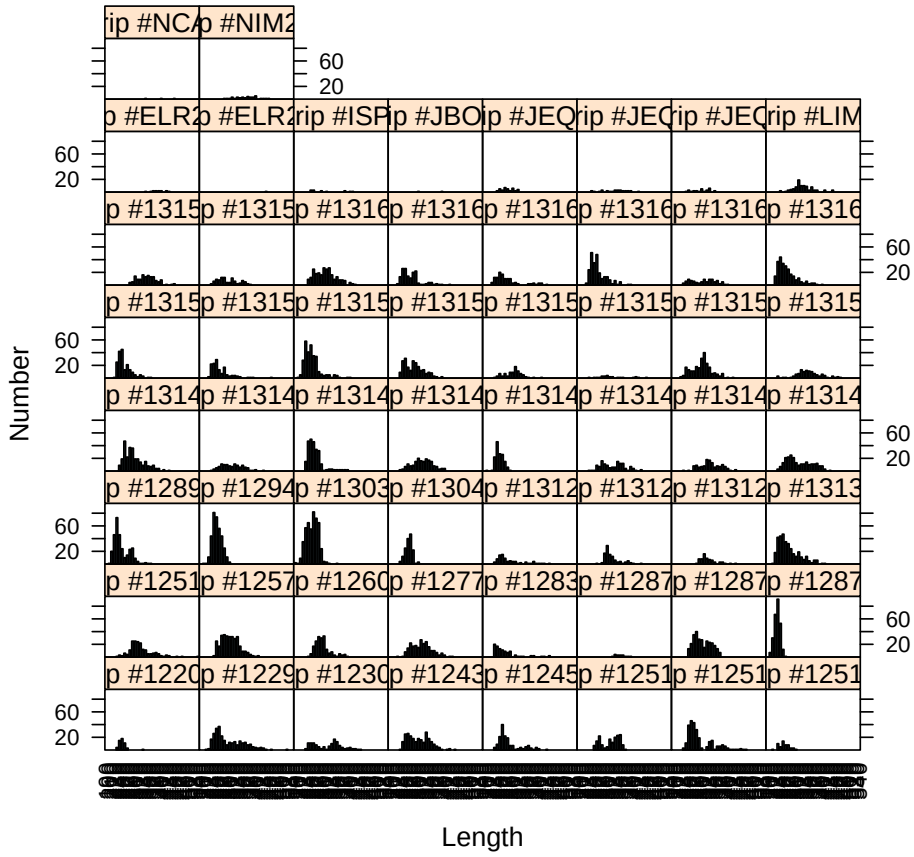
```
> # Same for the discards :
>
> LDVpl <- landisVol(s7dsurv_cs,species="Solea solea",
+ strDef = strD, fraction = "LAN")
> boxplot(LDVpl, cex.lab=0.5, rot=20)
```



- The scatter plot of the length structure of the samples is produced by `lenDisPlot`. The sampled weight may be raised to the fishing operation (`level="fo"`) or raised by trip (`level="trip"`, by default). One supplementary optional argument is available for `landisVol`, namely `tripCode` and `level`. `tripCode` is a vector allowing to focus on a restricted number of samples. this may be convenient when the datasets are large and when it becomes impossible to display all samples on a unique graphic page. The value by default is "all". Redimensioning the figures can be done by using the function `update` recalling the latest graph (`trellis.last.object()`). The main argument used here `par.strip.text=list(cex=0.5)` reduces the headers of each panel for providing more room to the histograms. With `update`, there is also the possibility (not used here) to allow a free Y-axis for each of the panels by specifying `scales=list(relation="free")`. Details of use and more arguments may be found on the help page of `xyplot`.

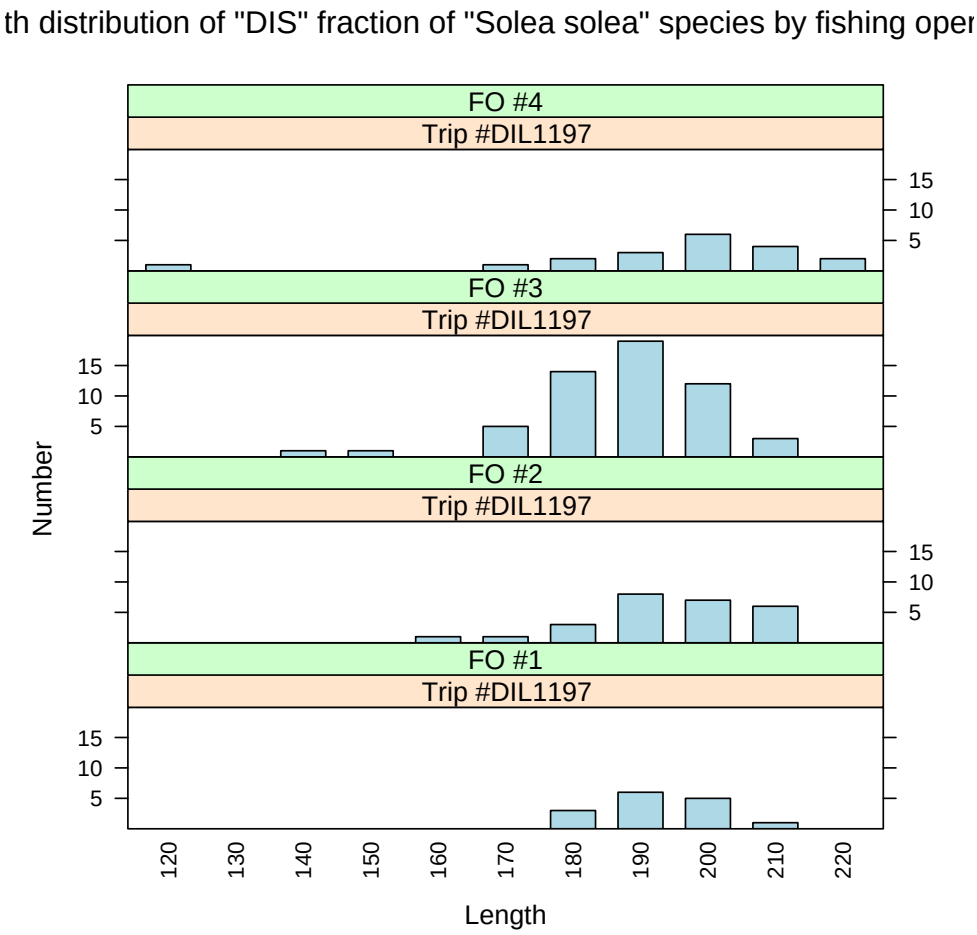
```
> lenDisPlot(sole_cs, species="Solea solea")
> update(trellis.last.object(),par.strip.text=list(cex=0.5),page=function(n) dev.copy(x11))
```

Length distribution of "LAN" fraction of "Solea solea" species by trip



You may want to look precisely at some selected trips:

```
> lenDisPlot(s7dsurv_cs,"Solea solea","DIS","DIL1197",
+ level="fo")
```



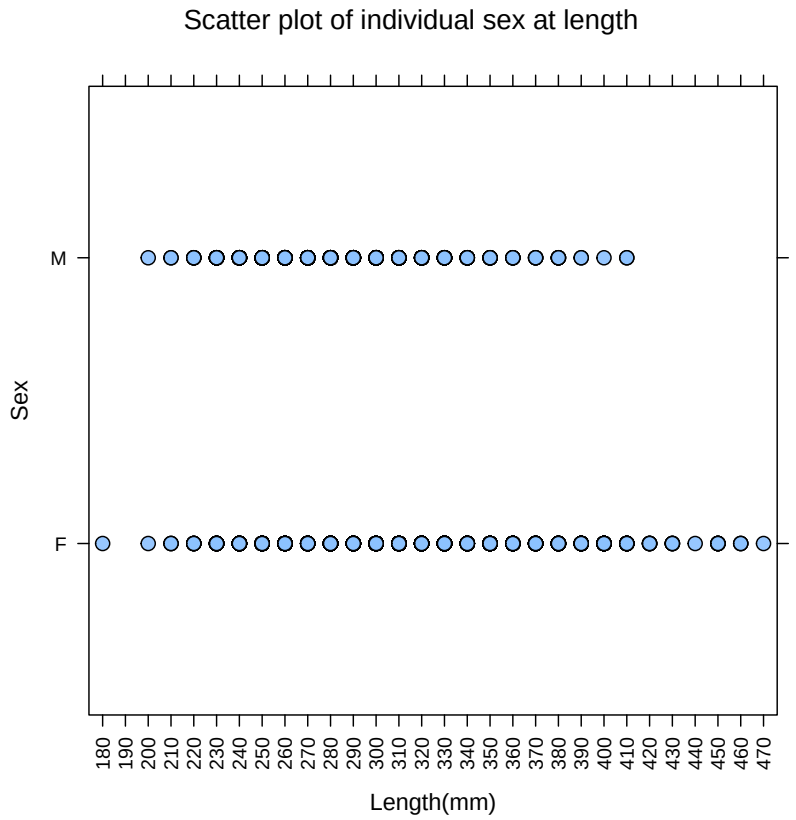
3.4.2 Biological parameters

Plotting the information concerning the biological parameters, weight at length, maturity at length, sex-ratio at length is done with the method `bioPar.plot`. In the method, you have to specify which parameters you want to display :

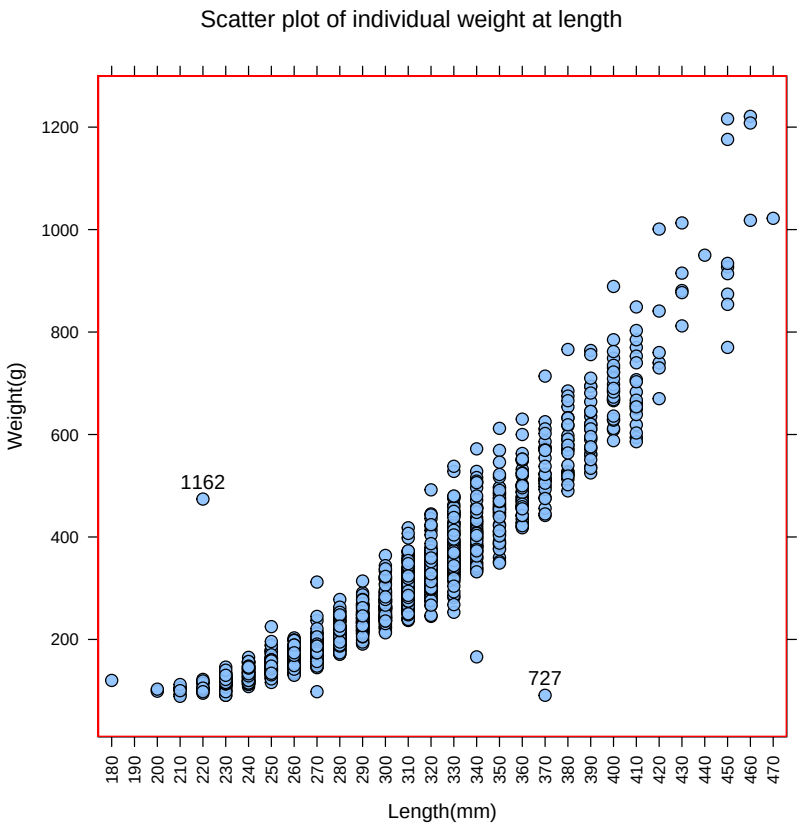
- type = ”ml” for plotting the maturity stages at length
- type =”sl” for plotting the sex-ratio at length
- type = ”wl” for plotting the individual weights at length

`bioPar.plot` offers the possibility to identify points in the figure with the option `selection = TRUE`. Using this option needs the affectation of the result into an object, in order to store the information linked to the objects indentified.

```
> bioPar.plot(s7d_cs,type="ml")
```




```
> x <- bioPar.plot(s7d_cs,type="wl", selection=TRUE)
```



Let's see what x contains :

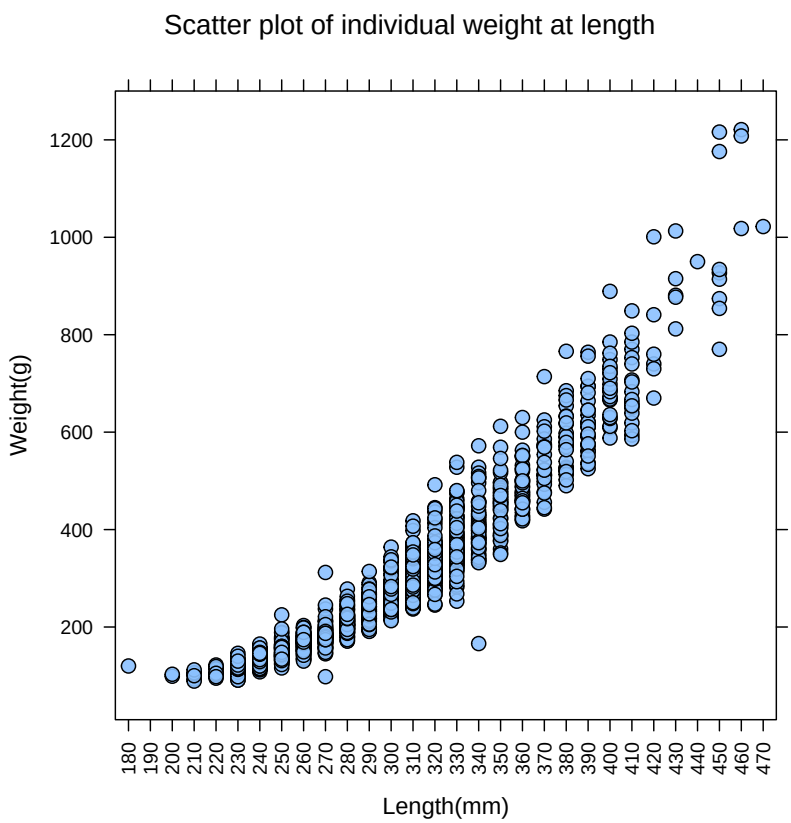
```
> x

$l
[1]  727 1162

$id.tab
     sampType landCtry vslFlgCtry year   proj trpCode staNum quarter month         spp sex catchCat lan
917         V      FRA        FRA 2006 BioPar      19    999       3     7 Solea solea   M      LAN
1466        V      FRA        FRA 2006 BioPar      23    999       4    11 Solea solea   F      LAN
                                        ageMeth plusGrp otoWt otoSide indWt matMeth matScale matStage
917  Otoliths - slides with transmitted light       -    NA    <NA>    91  Visual      1-7        2
1466 Otoliths - slides with transmitted light       -    NA    <NA>   474  Visual      1-7        2
```

As the identifiers of the samples are available, we may want to remove them for future analysis.
Important disclaimer : we, as COST team, do not promote the removal of outliers. If there are errors, they must be corrected in the central database and re-exported to COST, if there are no errors, the analysis must be carried out without modification of the dataset. COST provides all necessary methods to do all the work and it is the responsibility of the user to make good use of them. Moreover, the use of selection=TRUE, tends to make the script less robust to repeatability, since the future user will have to click on exactly the same dots to run the script withour errors and come with the same results.
If we consider, that we want to remove the samples 917 and 1466 from the previous graph, we are going to make use of the `subset` method:

```
> index <- x$id.tab$fishId
> s7dm_cs <- subset(s7d_cs, !fishId %in% index, table = "ca")
```

we look now at the new scatter plot:

```
> # the control of the removal
>
> bioPar.plot(s7dm_cs, "wl", selection = FALSE)
```



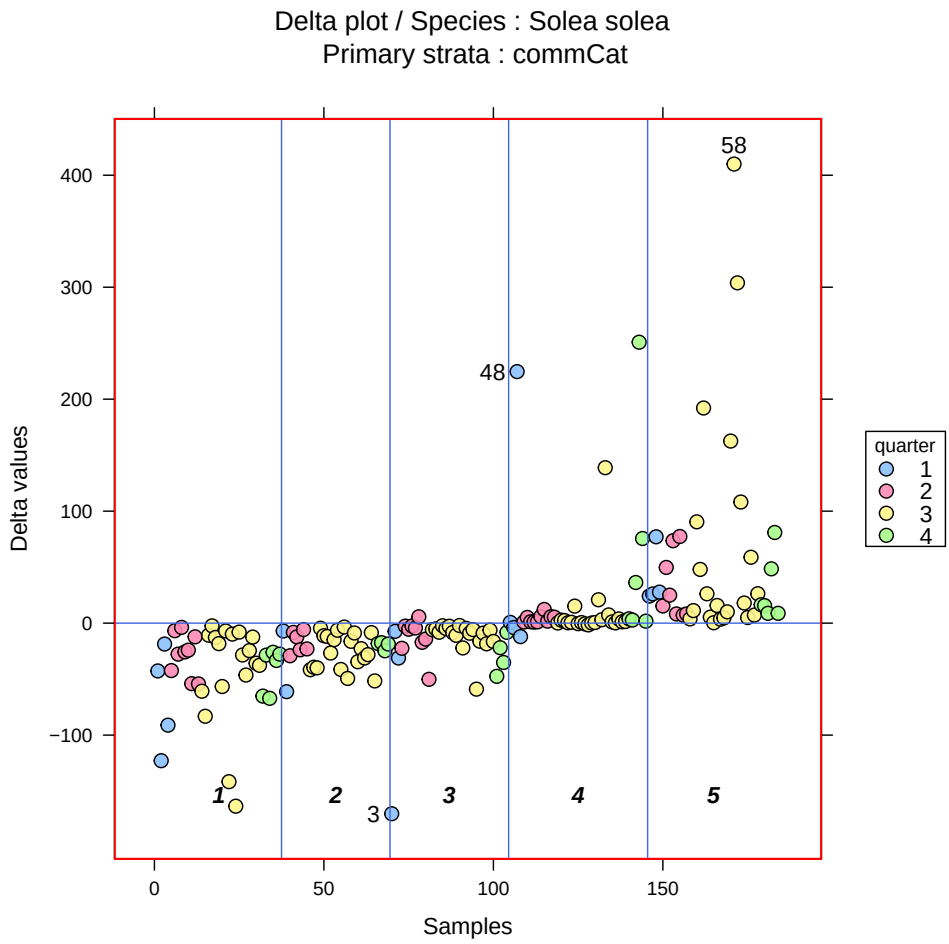
3.5 Delta calculation

The general assumption that samples represent the underlying population may be distorted by one or more samples taking most of the variance in a single stratum. Removing or reallocating such samples has to be done objectively following careful verification. Vigneau and Mahévas (2007) have presented an indicator, called Δ, derived from the formulation of the variance of a ratio estimator, which is generally the case in the fisheries sampling where a length structure is always linked to a sampled weight. Basically, theΔ index compares the number-at-length in the sample (d_{jkv}) with the number-at-length of all samples re-scaled to the sampled weight. When squared, theΔ index quantifies the exact contribution of a single sample to the overall variance. It then allows identification of the most influential sample(s) or possible outlier(s). A sample is regarded as an outlier if, after careful verification, it is considered as an accidental occurrence not representative of the population being sampled.
Let d and w be the number of fish in the sample and the sampled weight. The term j refers to the length class index ($j = 1, ..., J$), k to the stratum index ($k = 1, ..., K$), and v is the sample index ($v = 1, ..., n_k$). the variant of the Δ index implemented in COST is

$$\Delta_v = \sum_j d_{jkv} - \left(\frac{\sum_k \sum_v d_{jkv}}{\sum_k \sum_v w_{kv}} w_{kv} \right)$$

In preparation of the Δ calculation, you need to define a stratification with the method `strIni`. The three otpional arguments are `timeStrata`, `spaceStrata` and `techStrata`. These arguments are optional since they are all defined as NA by default. Here we define the time strata to be the quarter and the technical strata is the commercial category, which reflects the strategy used for collecting the samples..

```
> strD <- strIni(timeStrata="quarter",techStrata="commCat")
> deltas <- deltCalc(s7d_cs,strD,"Solea solea",strategy="cc")
> delta.out <- plot(deltas, strat1 = "techStrata",strat2 = "timeStrata", selection = TRUE)
```



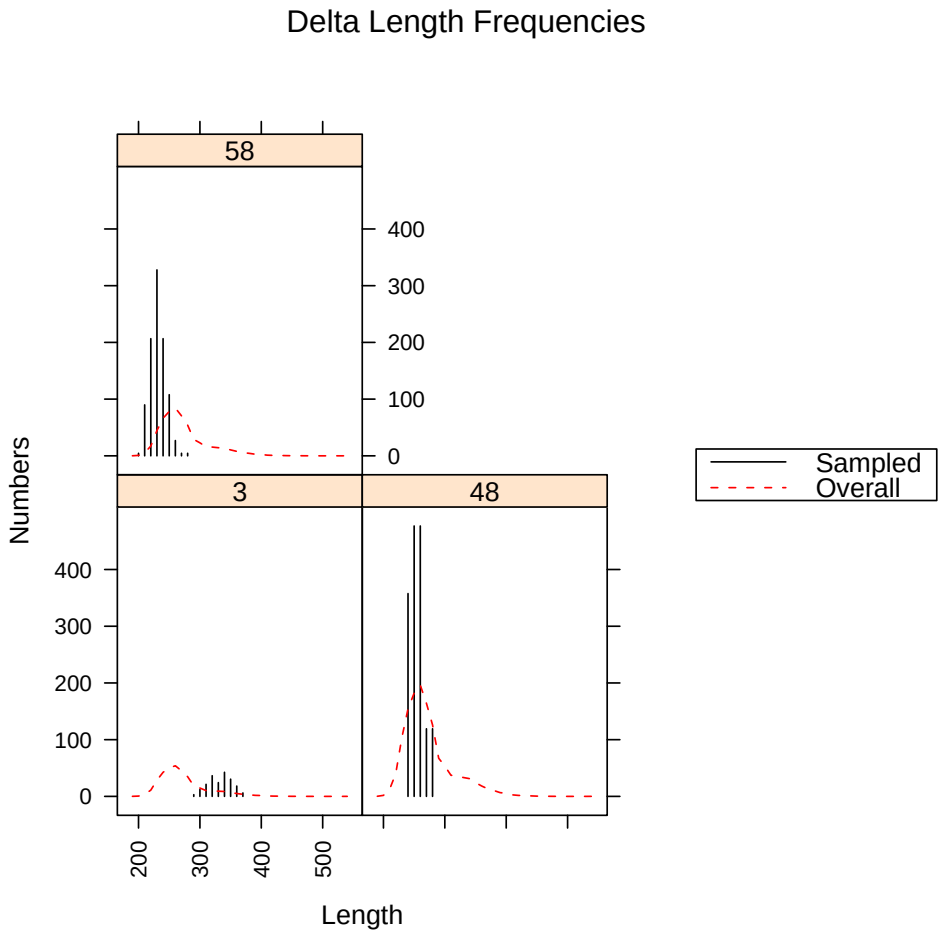
Let's describe what we see in this figure. We have called the plot with the argument `strat1 = "techStrata"`. This means that the x-axis is ordered by the technical strata information, wich is the commercial category here. Each modification of `techStrata` is visualised with a vertical line and the label is displayed in the bottom of the figure. The argument `strat2 = "timeStrata"` defines the colouring feature. Only the stratification predefined with `strIni` can be used within the plot method. The position of the dots in

the figure represent the distance between the sample and the average sample. The dots situated below the 0-line represent samples with relatively larger fish than the average and the dots situated over the 0-line represent samples with relatively smaller fish than the average. the figure presented above is typically representative of a commercial category sampling. In summary, the Δ plot method has three levels of flexibility, (i) the dataset which can be any subset of the data, (ii) the definition of the stratification which will be described in section XX and (iii) the choice of the ordering and colouring features in the plot.
The Δ plot method is the most powerful COST method to detect outliers and mis-stratification. Identifying samples may be done by selecting selection=TRUE in the argument of the plot function and affecting the result in an object, here deltas.out. The object generated by the plot function has two slots, the descriptor (desc) and the output (outPut). In the output slot, the sampled identified by clicking on the dots in the figure are given in deltas.out@outPut$sampId.

```
> names(delta.out@outPut)

[1] "species"  "fraction" "sampId"   "tabId"    "tab"

> delta.out@outPut$sampId

   SampNum trpCode staNum         spp commCat
3        3  122900    999 Solea solea       3
52      52  128751    999 Solea solea       1
58      58  128916    999 Solea solea       5
```

The object deltas.out can be used directly in the plot method. The result is as many graphs as samples identified showing the length of the actual sample (bar lines) compared to the length structure of all the samples rescaled to the weight of the actual sample (dashed line).

```
> plot(delta.out)
```



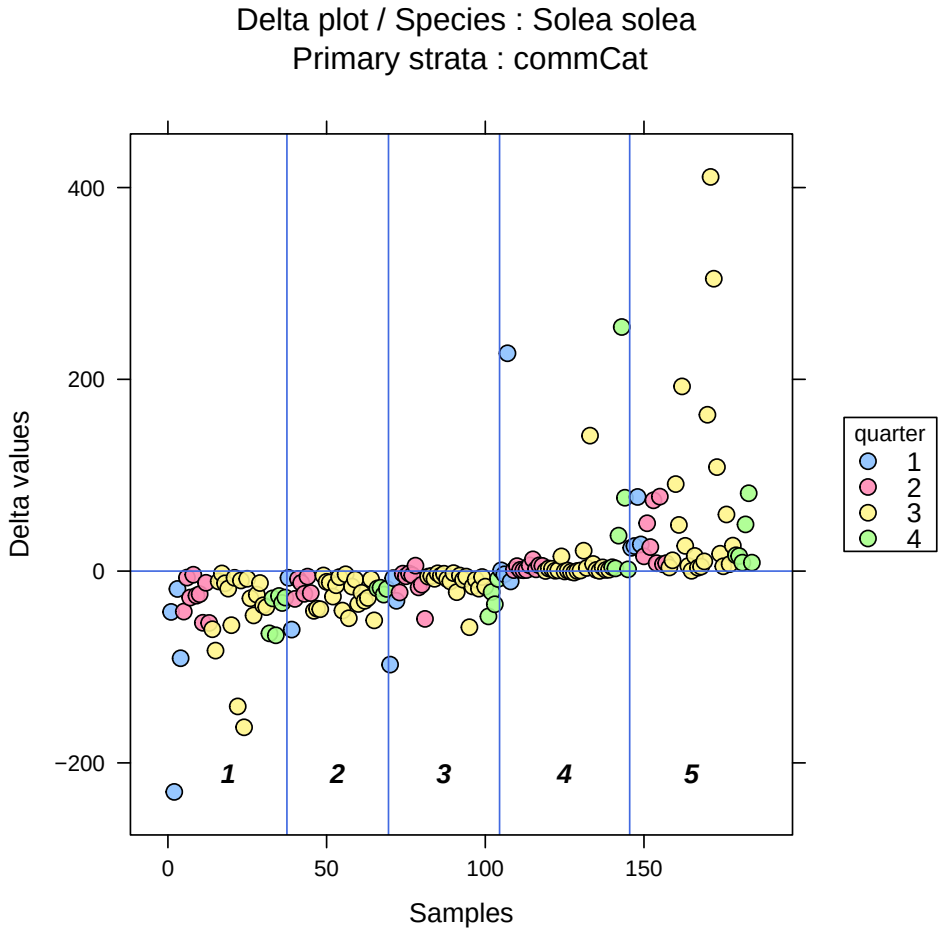
Like in section 3.3.2, the outliers and mis-stratification should be corrected in the central database and re-exported to COST. Nevertheless, it is possible to make the correction directly in the dataset used. For example, in the figure above, we think that the sample situated on the bottom on the line between

commercial categories 2 and 3 (sample no 3) is in fact, a sample of commercial category 1 and after verification this sampled trip had commercial category 1 noticed as 3 and vice versa . The code for modifying the dataset accordingly is given below:

```
> x <- delta.out@outPut$sampId
> x <- x[x$SampNum == 3,]
> s7d_cs@sl[s7d_cs@sl$trpCode == x$trpCode & s7d_cs@sl$commCat%in% c("1","3"),]$commCat <- c("3","1")
```

We check the result

```
> deltas <- deltCalc(s7d_cs,strD,"Solea solea",strategy="cc")
> deltas.out <- plot(deltas, strat1 = "techStrata",strat2 = "timeStrata", selection = FALSE)
```



The dots no 3 has moved to commercial category 1 and one of the blue dots of commercial category 1 has moved to commercial category 3. By doing so, the correction has not been made to the central database, and this may lead to confusion and errors.

3.6 Validation of the dataset

Once the dataset is controlled and validated, the dataset must be upgraded from RAW to VALIDATED status

```
> s7d_csv <- csDataVal(s7d_cs)
> s7d_cev <- ceDataVal(s7d_ce)
> s7d_clv <- clDataVal(s7d_cl)
```

At this stage, the structure of the tables has not been modified, only the class of the object has been changed

```
> class(s7d_cs)[1]

[1] "csData"

> class(s7d_csv)[1]

[1] "csDataVal"
```

4 Spatial Exploration

The spatial plotting function `spacePlot` provides a basic mapping tool with a range of spatial plotting options as well as being able to plot any numeric data within the *cs*, *cl* and *ce* data classes and their validated and consolidated forms. Accordingly the function `spacePlot` can be used in three basic forms depending on the requirements of the user.

1. Default plotting
2. Plotting of numeric variables by spatial strata
3. Plotting of COST data classes

4.1 Loading packages, data, and some basic manipulations

To load the COSTcore and COSTeda packages

```
> library(COSTcore)
> library(COSTeda)
```

Note that COSTeda requires COSTcore and will automatically load this package, thus there is no requirement to actually call the COSTcore package. The COSTeda package contains two example data sets; 2006 Ifremer data for sole in ICES areas VIId and VIIe and Marine Scotland - Science (formerly FRS) data for cod from the North Sea and Western Atlantic for 2004. These data sets consist of observer and market sampling data, landings data and effort data i.e. they contain classes csData, clData and ceData. To attach this data to the working directory:

```
> data(cod2004cs)
> data(cod2004cl)
> data(cod2004ce)
> data(sole)
```

The working directory now contains six data sets. Note that the `data(sole)` command attaches all three classes

```
> ls()

 [1] "alldepths"       "ca"              "ce"              "cl"              "CL"              "cod2004
[11] "cod2004csv"      "codDEF"          "code.list"       "codlengths"      "CS"              "delta.o
[21] "gsaareas"        "GSAareas"        "hh"              "hl"              "ICESAreaRects"   "index"
[31] "lons"            "newcoast"        "newhl"           "NHcoast"         "numeric.list"    "path"
[41] "s7d_ce"          "s7d_cec"         "s7d_cev"         "s7d_cl"          "s7d_clc"         "s7d_clv
[51] "s7dsurv_cs"      "samplingareas"   "sl"              "sole.ce"         "sole.cl"         "sole.cs
[61] "strD"            "strDef"          "strObs"          "tr"              "trips"           "uniqueM
```

We will assume this data has been checked and is OK and create validated classes for the cod sampling and landing data.

```
> cod2004csv <-csDataVal(cod2004cs)
> cod2004clv <-clDataVal(cod2004cl)
```

These data sets will be used to demonstrate the spatial plotting options within the COSTeda package.

4.2 Use of spacePlot

The spatial plotting function `spacePlot` is built around the R function `tapply` and has the general form:

```
spacePlot(COSTobj, variable, SpaceStrata, func,...)
```

where:

- `COSTobj` is an (optional) COST object of class csData, clData or ceData
- `variable` is a numeric variable to be plotted
- `SpaceStrata` is the spatial stratification over which the variable is to be plotted
- `func` is the function used to evaluate the variable (e.g. mean, length etc)

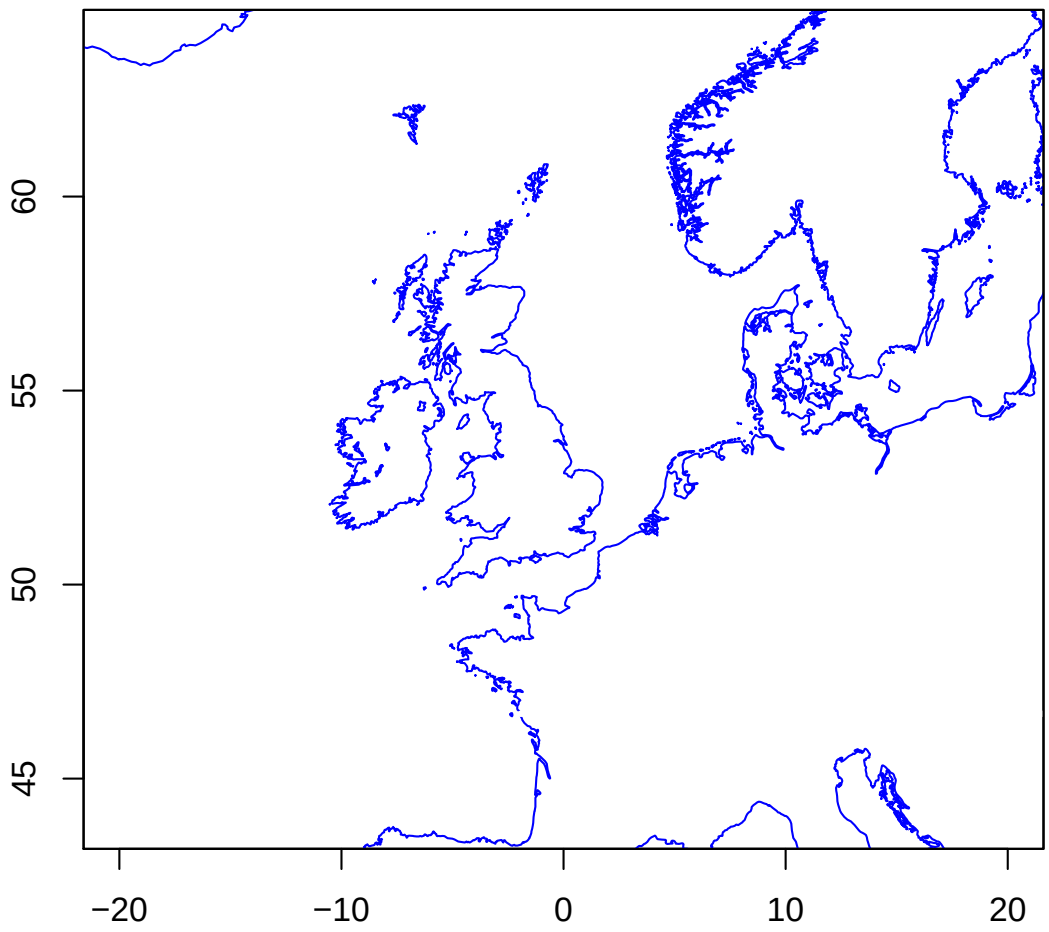
Accordingly the function `spacePlot` can be used in three basic forms depending on the requirements of the user and which arguments are passed to the function.

1. Default plotting
2. Plotting of numeric variables by spatial strata
3. Plotting of COST objects

4.3 Default plotting

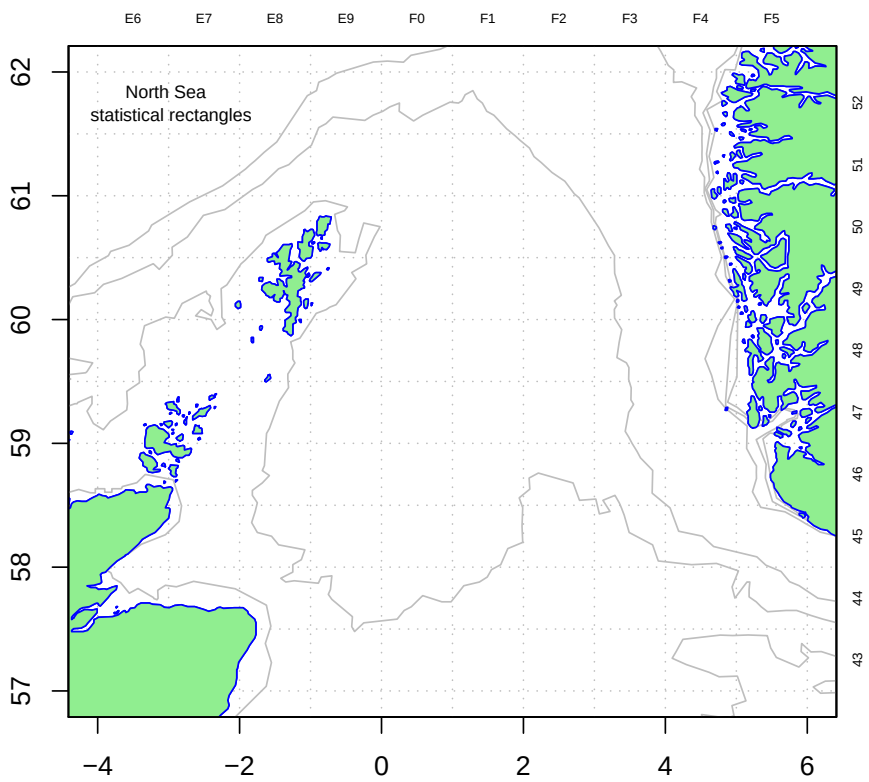
The default `spacePlot` call requires no additional arguments and will plot the North Sea and Western Atlantic.

```
> spacePlot()
```



This basic plot can be customised by specifying a number of other supported options that can be added to the plot. The extent of the area over which variables can be mapped is from -50 degrees West to 70 degrees East and 20 to 85 degrees of latitude and so, for example, to limit the plot area to the North Sea the `xlim` and `ylim` values can be set. Additionally here we plot depth contours for 50m, 100m, 200m and 500m and add a grid of the ICES statistical rectangles. Landmasses can be added by setting the argument `landmass=T` and the colour for the landmasses can be specified using the `col.land=` argument. The function `text` has also been used to add text to the plot.

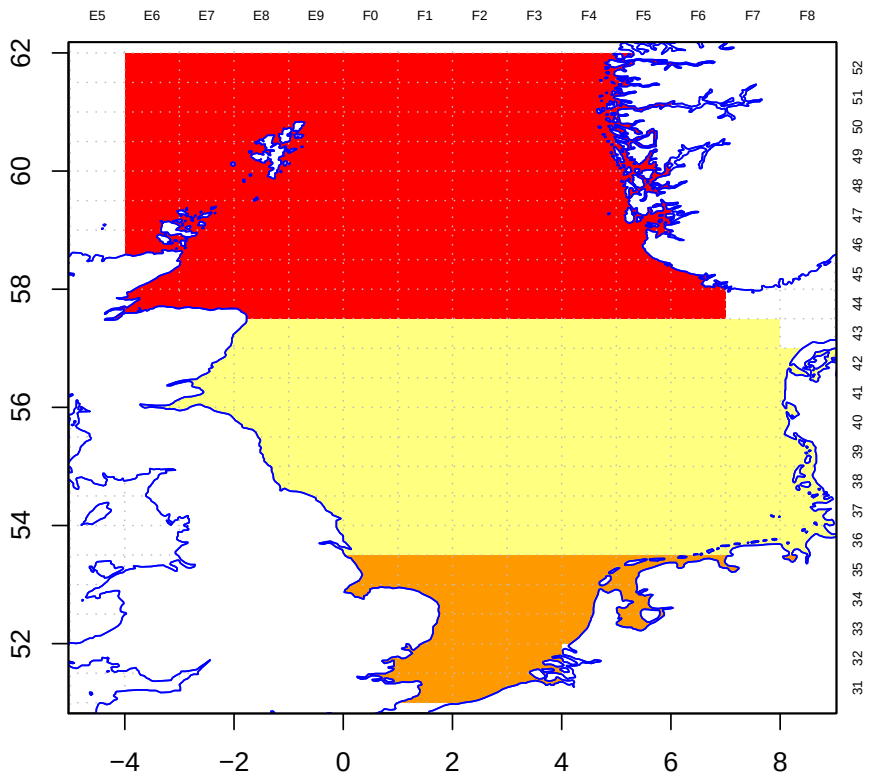
```
> spacePlot(xlim=c(-4,6),ylim=c(57,62),statrects=T,landmass=T,
+ col.land="lightgreen",depths=c(50,100,200,500))
> text(-3,61.75, "North Sea \n statistical rectangles",cex=0.7)
```



4.4 Plotting numeric variables by spatial strata

Where `spacePlot` is passed a vector of numeric values, a recognised vector of character strings specifying the spatial strata, and the name of a grouping function, it will plot these. The spatial strata can be expressed as ICES statistical rectangles, as ICES areas, as FAO areas, and as GSA areas in the Mediterranean, as well as some specific FRS sampling areas. Providing the form is correctly specified these will be automatically recognised. The named function can be any function that can be passed to `tapply`. For example to plot some (arbitrary) values by ICES areas using the Roman numeric nomenclature:

```
> spacePlot(c(56,34,45),c("IVa","IVb","IVc"),sum,statrects=T)
```

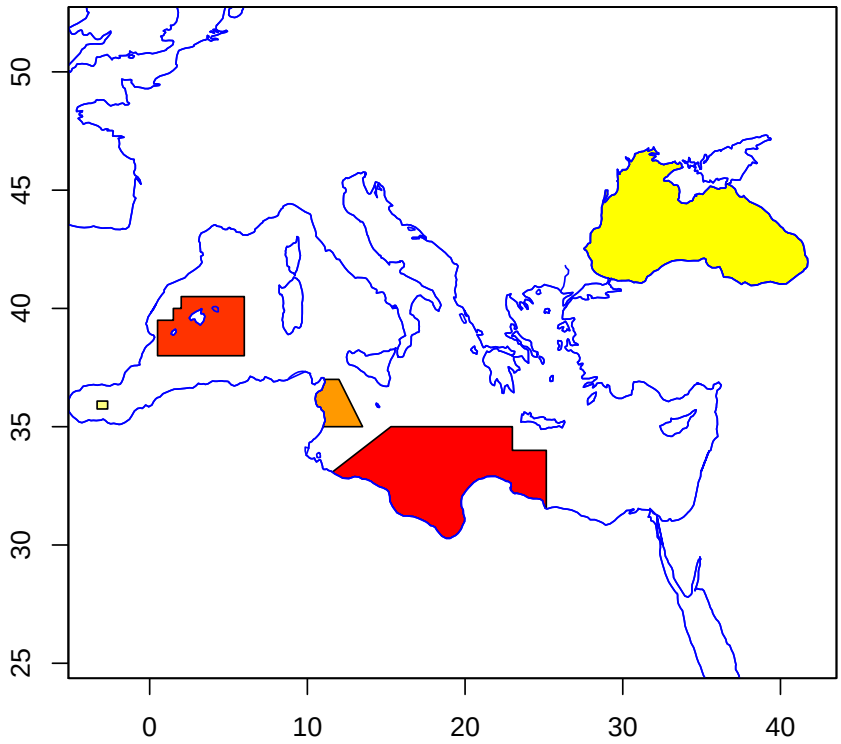


Using the FAO names would produce the same plot.

```
> spacePlot(c(56,34,45),c("27.4.a","27.4.b","27.4.c"),sum,statrects=T)
```

GSA areas in the Mediterranean are prefixed by `G37` and range from 1 to 30. The `asp=1.5` argument used here sets the aspect ratio of the plot:

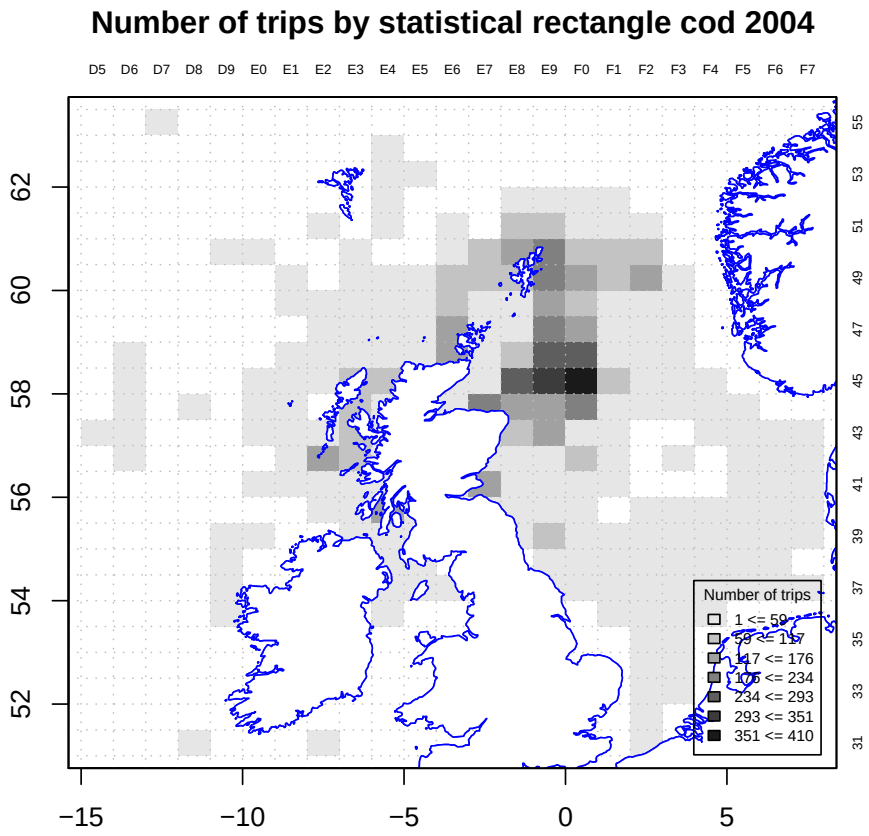
```
> values <-c(2,3,4,5,6)
> gsaareas <-c("G3702","G3729","G3713","G3705","G3721")
> spacePlot(values, gsaareas, sum,asp=1.5)
```



4.5 Plotting COST objects

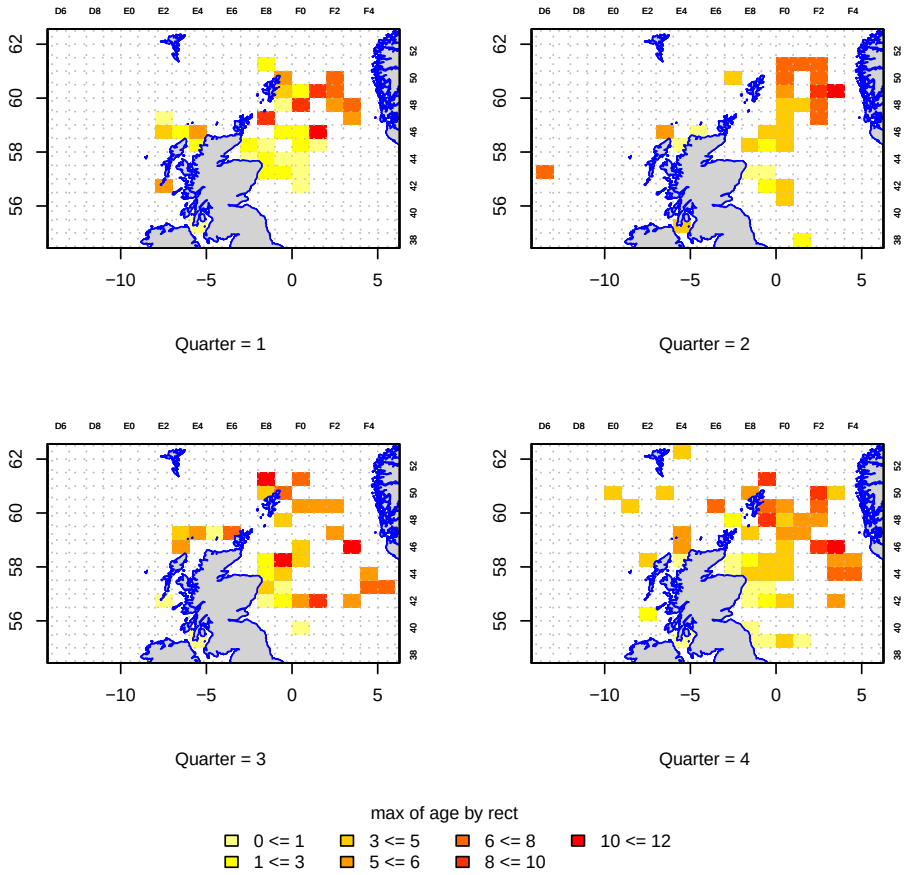
Where `spacePlot` is passed as its first argument a COST object, the `variable` argument becomes a character string specifying the name of the variable within the data set to be plotted, and the `SpaceStrata` argument becomes a character string specifying either `"area"` for areas or `"rect"` for statistical rectangles. The `maptype` argument allows the user to select between `"image"`, `"contour"`, `"bubble"` and `"value"` plots. Image plots, which are the default, are particularly suitable for plotting statistical rectangles. Here we use the cod2004ce data to plot number of trips by statistical rectangle. The numeric variable in the ceData is `"trpNum"` and we choose `"rect"` as our spatial variable. Specifying `scale=T` adds a scale to the plot and the `scale.title` argument is used to label the scale. Again we include a grid of statistical rectangles and add a landmass colour, this time white. The `colour=F` argument replaces the heat colours of the plot with grey scale colours, in many instances more suitable for non-colour printing. Note that the `xlim` and `ylim` values need not be set here, the plotting area being sized by default to the extremes of the data set.

```
> spacePlot(cod2004ce,"trpNum","rect",sum,scale=T,scale.title="Number of trips"
+ ,statrects=T,landmass=T,col.land="white",colour=F)
> title("Number of trips by statistical rectangle cod 2004",line=2)
```



Plots of stratified data are obtained by specifying the appropriate stratification to the arguments `TimeStrata` and `TechStrata` as character strings. To produce multiple plots per page the argument `nplots` can be used. For example to plot the maximum ages of cod in the 2004 market sampling data by ICES statistical rectangle and quarter:

```
> spacePlot(cod2004cs,"age","rect",max,TimeStrata="quarter",
+ statrects=TRUE,nplots=4,landmass=TRUE,
+ col.land="lightgrey")
```



4.6 Recoding variables and plotting CONSOLIDATED objects

CONSOLIDATED objects (i.e. those of class csDataCons, clDataCons and ceDataCons) can be plotted in the same way as RAW and VALIDATED objects. This allows particular time, space and technical stratifications to be plotted. For example the cod market sampling data was gathered by FRS demersal sampling areas, aggregations of statistical rectangles which differ from ICES areas. We can recode the spatial stratification in the CONSOLIDATED csData class to reflect these areas.
To recode a data set first we get a complete list of the unique statistical rectangles in a data set:

```
> rectold <- unique(cod2004cs@hh$rect)
```

These are then converted to the appropriate demersal sampling area using the `convert.statsq.samplingarea` function:

```
> rectnew <- convert.statsq.samplingarea(rectold)$parentarea
```

to which is added a character string prefix so that they are recognised in the `spacePlot` function:

```
> rectnew <-paste("FRSDMS",rectnew,sep="")
```

Comparing the old and new..

```
> head(rectold)
```

```
[1] "48F0" "44E8" "43E8" "45E9" "45F0" "44F0"
```

```
> head(rectnew)
```

```
[1] "FRSDMS1" "FRSDMS4" "FRSDMS4" "FRSDMS4" "FRSDMS5" "FRSDMS5"
```

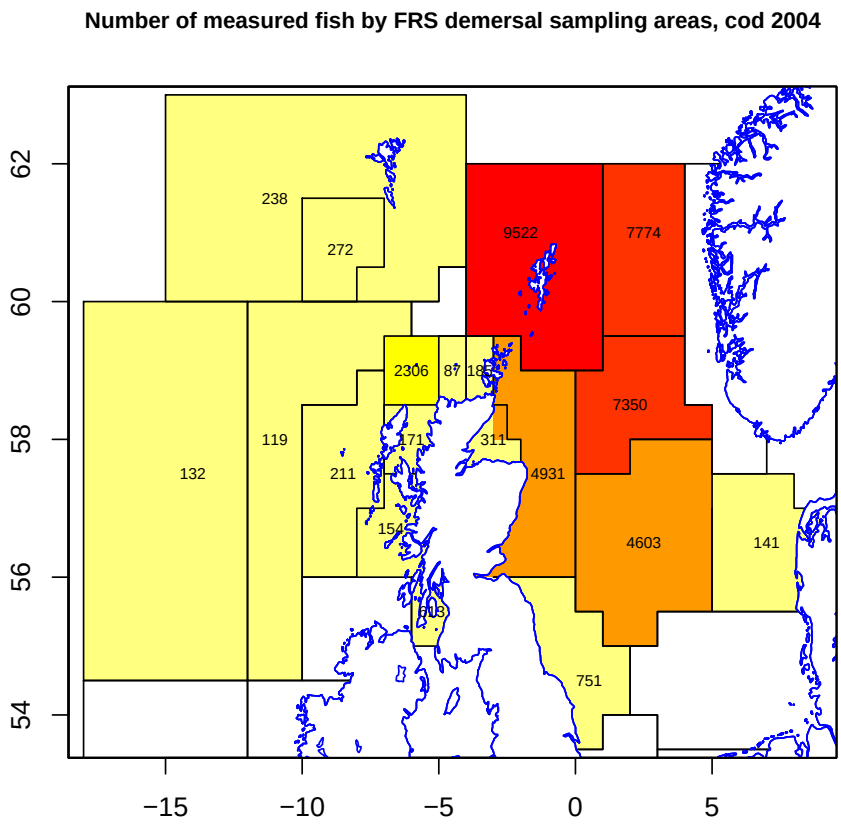
These recoded variables can be used to create a csDataCons object with the spatial stratification being that of the FRS demersal sampling areas.

```
> strDef <-strIni(timeStrata="month",spaceStrata="rect",techStrata="foCatEu5",
+ spRec=list(from=rectold,to=rectnew))
> cod2004csrecode <- csDataCons(cod2004csv,strDef)
```

Now we can plot the number of cod measured by FRS demersal sampling area. First we plot the total number of measured fish which is the sum of the `"lenNum"` field. Note that we shall specify by `"area"` though because we are plotting the `space` field of a consolidated object we would get the same result if we specified `"rect"` (or any other character string).

We can then use the `overlay=T` option to plot the actual values in each area onto this plot. The argument `area.lines=T` adds the boundaries to the sampling areas and the `cex.text` argument scales the added values

```
> spacePlot(cod2004csrecode,"lenNum","area",sum)
> spacePlot(cod2004csrecode,"lenNum","area",sum,area.lines=T,maptype="values"
+ ,cex.text=0.6,overlay=T)
> title("Number of measured fish by FRS demersal sampling areas, cod 2004",cex.main=0.8)
```



Bubble plots are an alternative way of displaying numeric values and the `overlay=T` option provides considerable scope to the user to generate single map comparison plots using combinations of sampling, effort and landings data. For example to plot a comparison of landed weight and number of fish measured by ICES rectangle using the 2004 cod data first we plot the landings to get the full spatial extent. Here arguments are also passed to set the location and the type of the scale to be added to the plot. The `cex.max.bubble` argument allows the size of the bubbles to be changed, and here the location of the scale has been set to the `topleft` of the plot. The `scale.box="n"` argument removes the box from the scale.

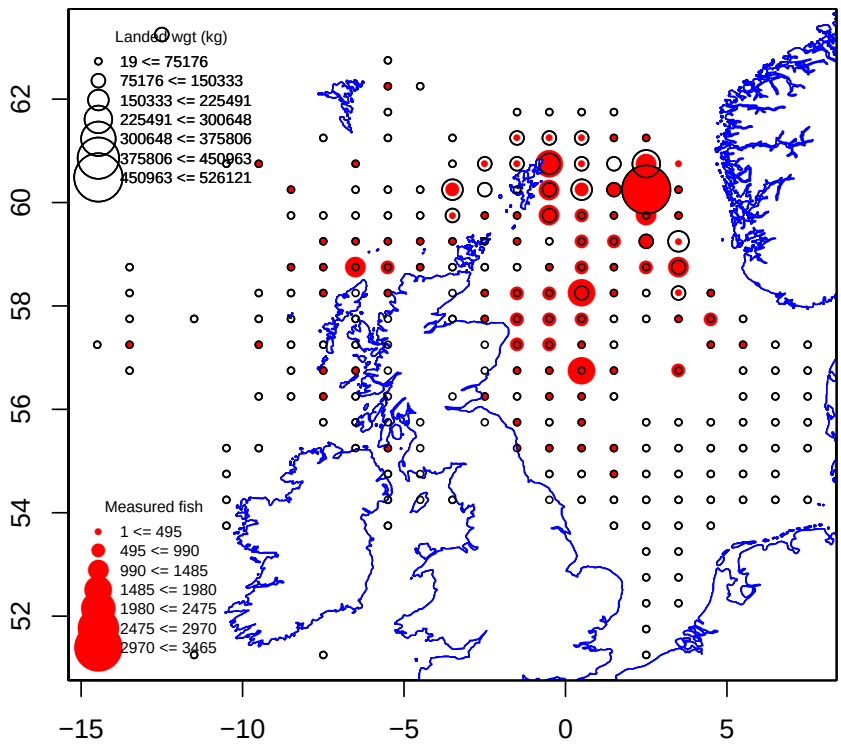
```
> spacePlot(cod2004cl,"landWt","rect",sum,maptype="bubble",col.pch=1,cex.max.bubble=4
+ ,scaleplace="topleft",scale=TRUE,scale.box="n",scale.title="Landed wgt (kg)")
```

Next we plot the csData on the same plot using the `overlay=TRUE` argument and putting the scale for this in a new location with a different title:

```
> spacePlot(cod2004cs,"lenNum","rect",sum,maptype="bubble",pch=16,
+ overlay=TRUE,cex.max.bubble=4,scaleplace="bottomleft",
+ scale=TRUE,scale.box="n",scale.title="Measured fish")
```

Finally we redo the first plot to show where landings were less than sampling:

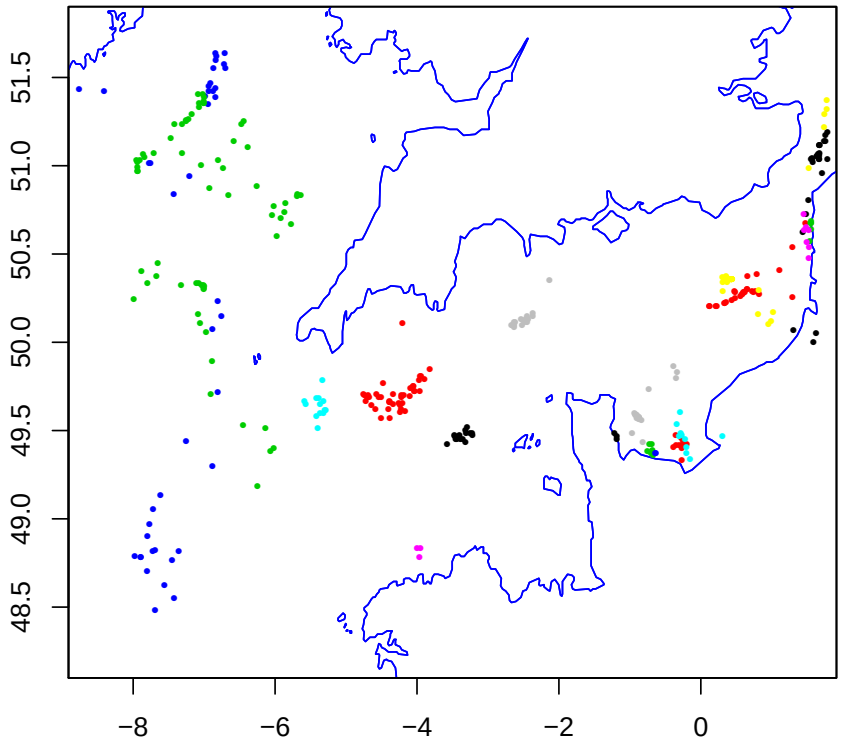
```
> spacePlot(cod2004cl,"landWt","rect",sum,maptype="bubble",
+ col.pch=1,cex.max.bubble=4,scaleplace="topleft",scale=TRUE,
+ overlay=TRUE,scale.box="n")
```



4.7 Plotting haul and trip positions

As well as the rectangles and areas specified in the COST data exchange format for all data types, observer data has fields for specifying the start and end location of hauls and trips. While not supported directly these can be plotted easily using `spacePlot`. For example, using the sole.cs data first we plot some (arbitrary) variable with the `plotmap=F` argument. This maps the relevant area without actually plotting any values. We can then plot the longitude and latitude starting points of the hauls from the hh table of the sole.cs data set. Note that we convert the trip codes to numeric values to distinguish different trips by colour.

```
> spacePlot(sole.cs,"lenNum","area",sum,plotmap=F)
> lons <-sole.cs@hh$lonIni
> lats <-sole.cs@hh$latIni
> trips <-as.numeric(factor(sole.cs@hh$trpCode))
> points(lons,lats,pch=16,cex=0.5,col=trips)
```



4.8 Spatial data sets and other functions

To support the user who wishes to pursue low level plotting options COSTeda provides a basic raw data resource which includes coastlines, depth contours, landmass polygons, grid lines, management area boundaries and lookup tables. For the manipulation of spatial data there is a range of conversion functions, for example the function `convert.icesarea.statsq` returns the constituent statistical rectangles for a given ICES area:

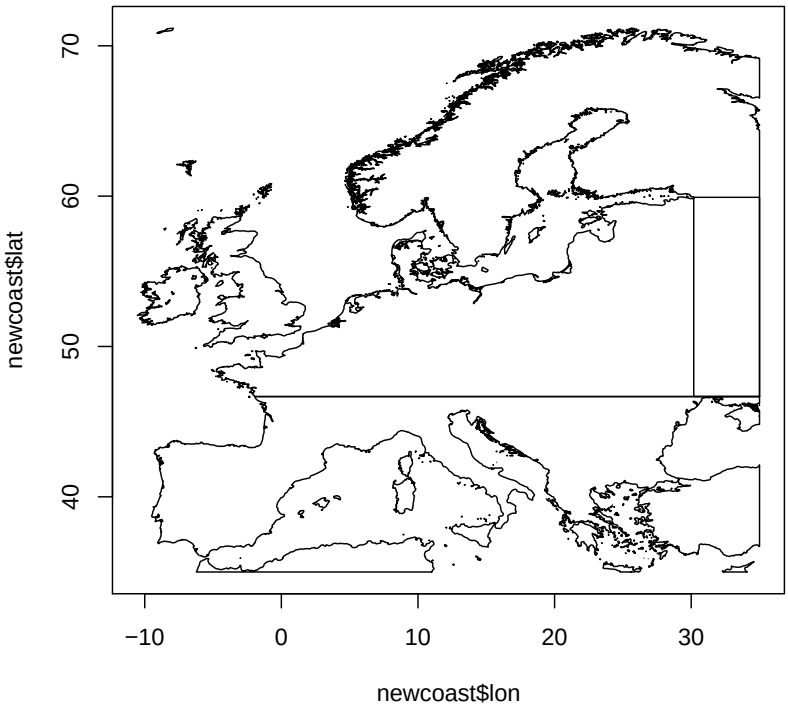
```
> statsqs <-convert.icesarea.statsq("VIa")$statsq
> head(statsqs)

[1] "38D8" "38D9" "38E0" "38E1" "38E2" "38E3"
```

and the function `convert.lon.lat.statsq` returns the statistical rectangles for a given longitude, latitude point:

```
> statsqs <-convert.lon.lat.statsq(c(-1,-1.56),c(65,65.3))
> head(statsqs)

[1] "59E9" "59E8"
```

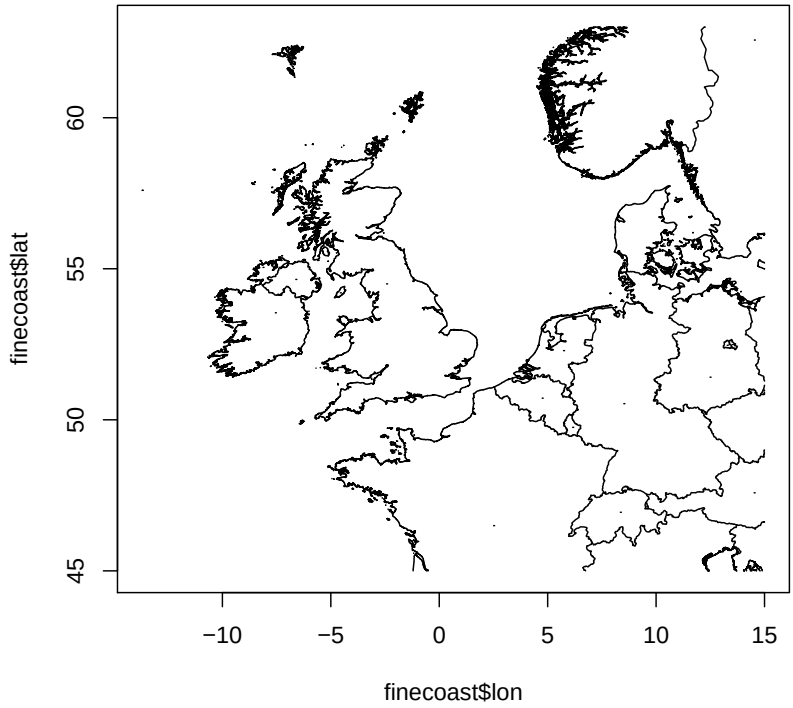
The data sets within COSTeda include 3 coastlines of different resolutions. The default used in spacePlot is called `Europe` and covers most of Western Europe

```
> data(Europe)
> plot(newcoast$lon,newcoast$lat,type="l")
```



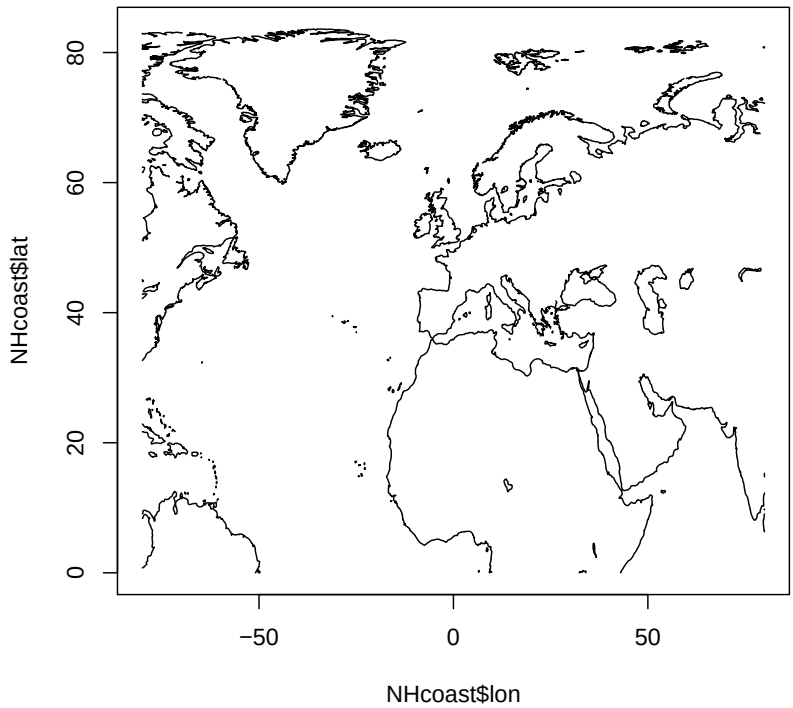
A 1:25000 resolution coastline (called finecoast) covers the North Sea:

```
> data(finecoast)
> plot(finecoast$lon,finecoast$lat,type="l")
```



and a 1:500000 resolution coastline covers most of the Northern Hemisphere

```
> data(NHcoast)
> plot(NHcoast$lon,NHcoast$lat,type="l")
```



Of the data sets within COSTeda perhaps the most useful for plotting is the data frame `ICESAreaRects` which contains all statistical rectangles their positions and ICES area designations.

```
> data(ICESAreaRects)
> head(ICESAreaRects)

  TypeName AreaCode   AreaName StatRect  lon   lat statsq division subarea subdivision
1     Ices        I Norway Sea    68J1  31.5 69.75   68J1        I       I           I
2     Ices        I Norway Sea    68J2  32.5 69.75   68J2        I       I           I
3     Ices        I Norway Sea    68J3  33.5 69.75   68J3        I       I           I
4     Ices        I Norway Sea    68J4  34.5 69.75   68J4        I       I           I
5     Ices        I Norway Sea    68J5  35.5 69.75   68J5        I       I           I
6     Ices        I Norway Sea    68J6  36.5 69.75   68J6        I       I           I
```

The list `GSAareas` gives polygons for the GSA areas used in the Mediterranean

```
> data(GSAareas)
> head(GSAareas$G3701)

        lon      lat
1 -5.600000 36.00000
2 -3.333333 36.00000
3 -3.333333 36.08333
4 -2.666667 36.08333
5 -2.666667 36.00000
6 -1.500000 36.00000
```

Details of all data sets and functions can be found in the appropriate help pages.

5 Stratification and recoding

5.1 Determination of the stratification

In COST, there are three and only three possibilities of stratification

1. time stratification : year, semester, quarter or month
2. space stratification : area, rectangle, sub-rectangle, harbour
3. technical stratification : commercial category, EU metier level 5, EU metier level 6, National metier

The dataset must reflect the sampling strategy used, i.e. you cannot use monthly stratification if the landings statistics in the cl table are given per quarter. The reverse is correct, i.e. you can choose a quarterly stratification with monthly information in the landings dataset. The detail is essential for recoding if necessary. The recoding allows all kind of flexibilities, by defining user's own groupings of time, space and/or technical units. For example, the possibility to merge metiers in the DCF is to be done by recoding. The national spatial stratification can also be done by recoding.

The first action to perform is to check the consistency of the information regarding stratification in all tables

```
> tabConsist(list(s7d_cs,s7d_ce,s7d_cl),"area")

       27.7.d 27.7.e
CS1_hh x
CS1_ca x      x
CE1    x
CL1    x

> tabConsist(list(s7d_cs,s7d_ce,s7d_cl),"quarter")

       1 2 3 4
CS1_hh x x x x
CS1_ca x x x x
CE1    x x x x
CL1    x x x x

> tabConsist(list(s7d_cs,s7d_ce,s7d_cl),"commCat")

       1 2 3 4 5
CS1_sl x x x x x
CS1_hl x x x x x
CS1_ca
CL1    x x x x x
```

Everything looks fine here and it is possible to define the corresponding stratification with the strIni method. By default, the spatial stratification is the area, so below, we do not need to specify. what we do below is notifying the stratification in an object, that will be useful in all estimation methods later.

```
> strD <- strIni(timeStrata="quarter",techStrata="commCat")
> strD

An object of class
Slot "timeStrata":
[1] "quarter"

Slot "spaceStrata":
[1] NA

Slot "techStrata":
[1] "commCat"

Slot "tpRec":
[[1]]
[1] NA
```

```
Slot "spRec":
[[1]]
[1] NA


Slot "tcRec":
[[1]]
[1] NA
```

There may be cases, such as metier strategy which will need recoding

```
> tabConsist(list(s7dsurv_cs,s7d_ce,s7d_cl),"foCatEu5")

       DRB_DEF DRB_MOL FPO_CRU GN_DEF GND_SPF GNS_CRU GNS_DEF GTR_DEF LL_DEF LLS_DEF OTB_CRU OTB_DEF OT
CS1_hh                                         x               x                              x       x
CE1    x       x       x       x      x               x       x       x      x       x       x       x
CL1    x       x       x       x      x               x       x       x      x       x       x       x
```

What do we see here? lots of metiers in the statistics tables (ce and cl) which is not surprising, and metiers in the commercial sampling tables that are absent from the statistics (GNS_CRU, OTB_MPD). After deep scrutiny (section 3.4) we want to recode GNS_DEF, OTB_MOL, GNS_CRU and OTB_MPD into respectively GTR_DEF, OTB_DEF, GTR_DEF and OTB_DEF. In the DCF world, this kind of recoding is known under the name of mergers, and COST provides all the tools necessary to assess the similarities or dissimilarities between metiers (see section 3.4 and the Delta calculation for example), and to proceed the merging.

```
> strObs <- strIni(timeStrata="quarter",techStrata="foCatEu5",
+ tcRec = list(from = c("GNS_DEF","OTB_MOL","GNS_CRU","OTB_MPD"),
+ to = rep(c("GTR_DEF","OTB_DEF"),2)))
```

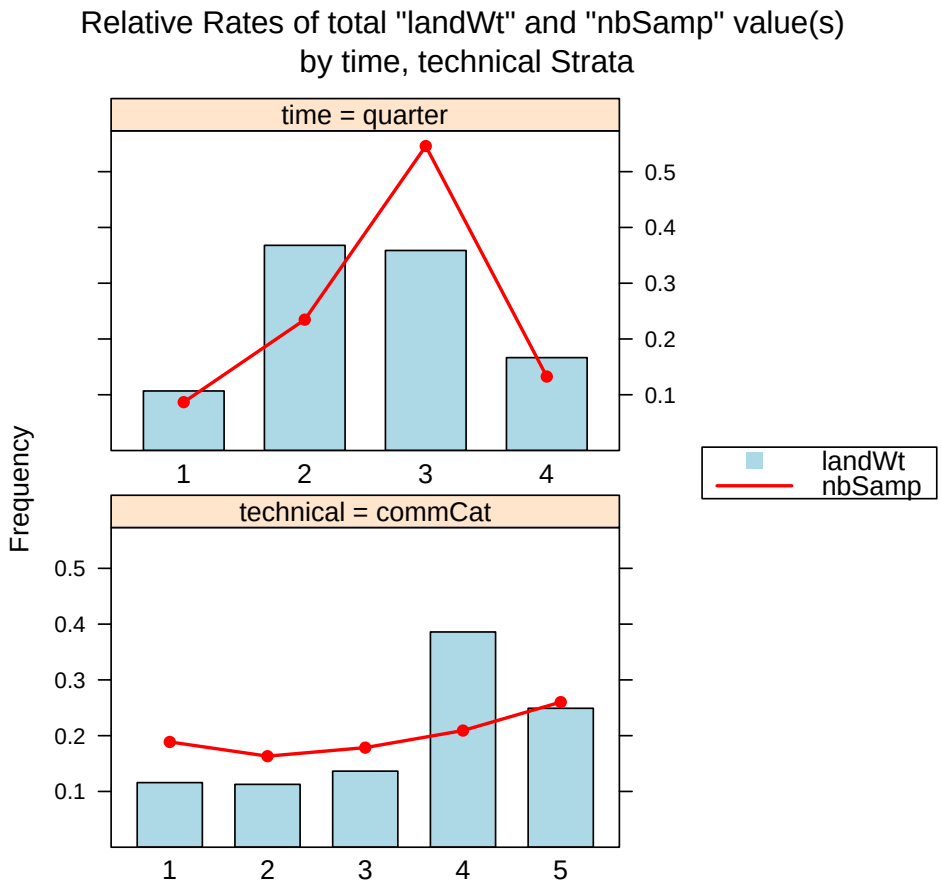
What we have done here is to recode "GNS_CRU" into "GNS_DEF" and "OTB_MPD" into "OTB_DEF" virtually. Every method that necessitates the object strIni as an argument will proceed to the change accordingly, but the datasets have not been modified. There is no constraint on recoding, it may be the case that recoding involves lots of modalities, in which case you will need to use intermediate objects, which may avoir typographic errors.
Example with the same purpose as above:

```
> uniqueMet <- unique(c(s7d_cl@cl$foCatEu5, s7dsurv_cs@hh$foCatEu5))
> uniqueMet

 [1] "DRB_DEF" "DRB_MOL" "FPO_CRU" "GN_DEF"  "GND_SPF" "GNS_DEF" "GTR_DEF" "LL_DEF"  "LLS_DEF"
     "OTB_CRU" "OTB_DEF" "OTB_MOL" "OTM_SPF" "TBB_DEF" "TBS_DEF" "GNS_CRU" "OTB_MPD"

> strObs <- strIni(timeStrata="quarter",techStrata="foCatEu5",
+ tcRec = list(from = uniqueMet[c(6,12,16,17)], to = uniqueMet[c(7,11,7,11)]))
> strObs

An object of class
Slot "timeStrata":
[1] "quarter"

Slot "spaceStrata":
[1] NA

Slot "techStrata":
[1] "foCatEu5"

Slot "tpRec":
[[1]]
[1] NA


Slot "spRec":
```

```
[[1]]
[1] NA


Slot "tcRec":
$from
[1] "GNS_DEF" "OTB_MOL" "GNS_CRU" "OTB_MPD"

$to
[1] "GTR_DEF" "OTB_DEF" "GTR_DEF" "OTB_DEF"
```

The following figure, very appreciated by the samplers, is the proportionality between the sampling intensity and the landings statstics. Following Neyman allocation optimimum, the sampling intensity must be proportional to the size and to the standard error of the strata. The size of the strata is expressed here as the total landings, and with the assumption that the standard error is proportional to the size, it is easy to point out imbalance in the sampling allocation just by looking at the proportionality between the sampling intensity and the total landings. A strong imbalance may lead to an inefficient sampling scheme, which will need to be redefined. This figure makes use of the object containing our stratification and recoding.

```
> CS <- relativeValue(s7d_csv,strD,"nbSamp")
> CL <- relativeValue(s7d_clv,strD)
> plot(CL, CS)
```



5.2 Consolidation of the dataset

Once the user has defined precisely the stratification to be used, the dataset may be upgraded from VALIDATED to CONSOLIDATED status. The call to the constructor csDataCons needs a validated object as an argument and the object containing the stratification and recoding.

```
> s7d_csc <- csDataCons(s7d_csv, strD)
> s7d_cec <- ceDataCons(s7d_cev, strD)
> s7d_clc <- clDataCons(s7d_clv, strD)
```

The warning message given here is logical since there cannot be effort data linked to commercial categories. As a result, the dataset s7d_cec is empty. At this stage, the datasets are totally redefined.and all the information needed for the raising and precision estimates are positioned in the right place in the dataset. This phase is one of the most critical in COST chain of actions, since the sampling strategy and the stratification will define which information goes in which consolidated table fields. The genericity and robustness of raising estimates and precision calculation are dependent on this critical phase. will A simple look at the new tables :

```
> head(s7d_csc)

An object of class
Slot "desc":
[1] "Unknown stock"

Slot "tr":
  PSUid     time space technical sampType landCtry vslFlgCtry     proj trpCode foNum daysAtSea vslId sa
1     1 2006 - 4   all         5        M      FRA        FRA MarkSamp  122099    NA        NA   151
2     2 2006 - 1   all         5        M      FRA        FRA MarkSamp  122900    NA        NA   167
3     3 2006 - 1   all         1        M      FRA        FRA MarkSamp  122900    NA        NA   167
4     4 2006 - 1   all         3        M      FRA        FRA MarkSamp  122900    NA        NA   167
5     5 2006 - 1   all         4        M      FRA        FRA MarkSamp  122900    NA        NA   167
6     6 2006 - 1   all         2        M      FRA        FRA MarkSamp  123016    NA        NA   171

Slot "hh":
  PSUid SSUid     time space technical sampType landCtry vslFlgCtry     proj trpCode staNum foVal aggLe
1     1     1 2006 - 4   all         5        M      FRA        FRA MarkSamp  122099    999     V
2     2     1 2006 - 1   all         5        M      FRA        FRA MarkSamp  122900    999     V
3     3     1 2006 - 1   all         1        M      FRA        FRA MarkSamp  122900    999     V
4     4     1 2006 - 1   all         3        M      FRA        FRA MarkSamp  122900    999     V
5     5     1 2006 - 1   all         4        M      FRA        FRA MarkSamp  122900    999     V
6     6     1 2006 - 1   all         2        M      FRA        FRA MarkSamp  123016    999     V

Slot "sl":
  PSUid SSUid TSUid     time space technical          sort sampType landCtry vslFlgCtry     proj trpCod
1     1     1     5 2006 - 4   all         5 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12209
2     2     1     5 2006 - 1   all         5 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12290
3     3     1     1 2006 - 1   all         1 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12290
4     4     1     3 2006 - 1   all         3 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12290
5     5     1     4 2006 - 1   all         4 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12290
6     6     1     2 2006 - 1   all         2 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12301

Slot "hl":
  PSUid SSUid TSUid     time space technical          sort sampType landCtry vslFlgCtry     proj trpCod
1     1     1     5 2006 - 4   all         5 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12209
2     1     1     5 2006 - 4   all         5 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12209
3     1     1     5 2006 - 4   all         5 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12209
4     1     1     5 2006 - 4   all         5 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12209
5     1     1     5 2006 - 4   all         5 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12209
6     1     1     5 2006 - 4   all         5 LAN-HUC-EU-NA        M      FRA        FRA MarkSamp  12209

Slot "ca":
  PSUid SSUid     time space technical       sort sampType landCtry vslFlgCtry   proj trpCode staNum
1   185     1 2006 - 2   all      <NA> LAN-HUC-NA        V      FRA        FRA BioPar      12    999 So
```

```
2   185     1 2006 - 2   all      <NA> LAN-HUC-NA        V      FRA        FRA BioPar      12    999 Sol
3   185     1 2006 - 2   all      <NA> LAN-HUC-NA        V      FRA        FRA BioPar      12    999 Sol
4   185     1 2006 - 2   all      <NA> LAN-HUC-NA        V      FRA        FRA BioPar      12    999 Sol
5   185     1 2006 - 2   all      <NA> LAN-HUC-NA        V      FRA        FRA BioPar      12    999 Sol
6   185     1 2006 - 2   all      <NA> LAN-HUC-NA        V      FRA        FRA BioPar      12    999 Sol
                                   ageMeth plusGrp otoWt otoSide indWt matMeth matScale matStage
1 Otoliths - slides with transmitted light       -    NA    <NA>   350  Visual      1-7        2
2 Otoliths - slides with transmitted light       -    NA    <NA>   479  Visual      1-7        2
3 Otoliths - slides with transmitted light       -    NA    <NA>   341  Visual      1-7        5
4 Otoliths - slides with transmitted light       -    NA    <NA>   398  Visual      1-7        3
5 Otoliths - slides with transmitted light       -    NA    <NA>   404  Visual      1-7        4
6 Otoliths - slides with transmitted light       -    NA    <NA>   489  Visual      1-7        3
```

```
> head(s7d_cec)
```

```
An object of class
Slot "desc":
[1] "Unknown stock"

Slot "ce":
  vslFlgCtry time space technical trpNum foNum foDur effKwDays effGtDays daysAtSea
1       <NA> <NA>  <NA>      <NA>     NA    NA    NA        NA        NA        NA
```

```
> head(s7d_clc)
```

```
An object of class
Slot "desc":
[1] "Unknown stock"

Slot "cl":
  landCtry vslFlgCtry     time space technical       taxon landCat commCatScl commCat unallocCatchWt mi
1      FRA        FRA 2006 - 1   all         1 Solea solea     HUC         EU       1             NA
2      FRA        FRA 2006 - 2   all         1 Solea solea     HUC         EU       1             NA
3      FRA        FRA 2006 - 3   all         1 Solea solea     HUC         EU       1             NA
4      FRA        FRA 2006 - 4   all         1 Solea solea     HUC         EU       1             NA
5      FRA        FRA 2006 - 1   all         1 Solea solea     HUC         EU       1             NA
6      FRA        FRA 2006 - 4   all         1 Solea solea     HUC         EU       1             NA
```

The new fields are (see the COST report for more explanations):

- PSUid : Primary Sampling Unit identifier : the trip
- SSUid : Secondary Sampling Unit idenitfier : the fishing operation
- TSUid : Tertiary Sampling Unit identifier : the commercial category
- sort : all information enabling to distinguish identical PSUid, SSUid and TSUid. Mainly, it corresponds to the sub-category
- time : replacing the fields date, quarter, year, and containing the information required in the stratification and recoding object.
- space : replacing the fields area, rect, and sub-rectangle and containg the information required in the stratification and recoding object.
- technical, repakcing the fields commCat, foCatEu5, foCatEu6, foCatNat and containg the information required in the stratification and recoding object.

This new dataset has a simpler structure, ready for the raisong and precision analysis. This is why it may be accurate to save the created objects for future re-use without going through the lengthy process of data exploration.

```
> save(s7d_csc, s7d_clc, s7d_cec, strD, file = "Sole_Market_7d_2006_cons.Rdata")
```

At a later stage, in a new R session it will be possible to start from this point by recalling the above mentionned objects:

```
> load("Sole_Market_7d_2006_cons.Rdata")
```

6 Reporting

A reporting method `Report` has been developed, with the aim of producing pre-formated documents summarising (i) the main elements contained in the datsets, (ii) the outcomes of the statistical estimates and (iii) the script of the analysis. The method will produce a pdf report, depending on the object given as an argument, and will write this document in a folder created under the working directory. The logo of the fishing institute providing the information can be drawn on the upper part of the pages, if a jpg file exists and passed to the function as an argument. Examples of use are as follows :

```
> library(COSTreport)
> library(COSTdata)
> data(SIBM2006cs)
> data(SIBM2006cl)
> #subset on Merluccius merluccius
> SIBM2006cs <- subsetSpp(SIBM2006cs, spp %in% "Merluccius merluccius")
> #validating
> SIBMval <- csDataVal(SIBM2006cs)
> #Defining the stratification
> strD <- strIni(timeStrata="quarter",spaceStrata="area",techStrata="foCatNat")
> #Creation of ./Report/mainVal.pdf
>
> Report(strD,SIBMval,language="EN",logo="N:/COST/sibm.jpg")
>
> #Consolidating
> SIBMconsCS <- csDataCons(SIBMval,strD)
> SIBMconsCL <- clDataCons(clDataVal(SIBM2006cl),strD)
> #Creation of ./Report/mainCons.pdf
>
> Report(strD,SIBMconsCS,SIBMconsCL,language="EN",logo="N:/COST/sibm.jpg")
>
> SIBM_NLa <-dbeObject(desc="Output object",species="Merluccius merluccius",catchCat="LAN",strataDesc=s
> SIBM_NLa <- RaiseLgth(SIBM_NLa,SIBMconsCS,SIBMconsCL)
> # looking at annual estimates per metier
> SIBM_NLa <- stratAggreg(SIBM_NLa, timeStrata=TRUE, spaceStrata=TRUE, techStrata=FALSE)
> #Creation of ./Report/maindbe.pdf
>
> Report(SIBM_NLa,language="EN",logo="N:/COST/sibm.jpg")
>
> # Creation of the ./Report/mainScript.pdf
>
> Report("./script.r",language="EN",logo="N:/COST/sibm.jpg")
>
```

The pdf documents resulting from the examples given above are shown as annexes of the User Manual.

7 Preparing the estimations

With the purpose of genericity and robustness, it has been decided to create a fixed formated object for receiving the outcomes of every methods and for any parameter estimated. This object is complex and possesses a total of no less than 20 slots. Let's create one and looks how it looks like.

```
> disEstim <- dbeObject(desc="Sole VIId Discards",species="Solea solea",catchCat="DIS",
+ strataDesc=strD, methodDesc="analytical")
```

The 6 first slots are descriptors to summarise all the information related to the object, given by the arguments passed through the dbeobject method. The 6 descritors are followed by five slots containing all estimates of the parameters and their related precision. They are :

1. sampling statistics :
   (a) number of samples related to length, related to age
   (b) number of fish measured related to length, related to age
2. Estimates related to length
   (a) length structure for each strata, bootstrap replicates
   (b) variance at length for each strata
   (c) confidence intervals and coefficient of variation for each length and each strata, and DCR indicator giving one value summarising the precision of the estimates
3. Estimates related to age
   (a) age structure for each strata, bootstrap replicates
   (b) variance at age for each strata
   (c) confidence intervals and coefficient of variation for each age and each strata, and DCR indicator giving one value summarising the precision of the estimates
4. Estimates of total number
   (a) estimates for each strata, bootstrap replicates
   (b) estimates of variance for each strata
   (c) confidence intervals and coefficient of variation for each strata, and DCR indicator giving one value summarising the precision of the estimates
5. Estimates of total weight
   (a) estimates for each strata, bootstrap replicates
   (b) estimates of variance for each strata
   (c) confidence intervals and coefficient of variation for each strata, and DCR indicator giving one value summarising the precision of the estimates

The structure of the created object is the following:

```
> str(disEstim)

Formal class 'dbeOutput' [package "COSTdbe"] with 20 slots
  ..@ desc      : chr "Sole VIId Discards"
  ..@ species   : chr "Solea solea"
  ..@ catchCat  : chr "DIS"
  ..@ param     : chr NA
  ..@ strataDesc:Formal class 'strIni' [package "COSTcore"] with 6 slots
  .. .. ..@ timeStrata : chr "quarter"
  .. .. ..@ spaceStrata: chr NA
  .. .. ..@ techStrata : chr "commCat"
  .. .. ..@ tpRec      :List of 1
  .. .. .. ..$ : logi NA
  .. .. ..@ spRec      :List of 1
  .. .. .. ..$ : logi NA
```

```
.. .. ..@ tcRec      :List of 1
.. .. .. ..$ : logi NA
..@ methodDesc: chr "analytical"
..@ nSamp     :List of 2
.. ..$ len:'data.frame':        1 obs. of  4 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. ..$ age:'data.frame':        1 obs. of  3 variables:
.. .. ..$ time : Factor w/ 0 levels: NA
.. .. ..$ space: Factor w/ 0 levels: NA
.. .. ..$ value: num NA
..@ nMeas     :List of 2
.. ..$ len:'data.frame':        1 obs. of  4 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. ..$ age:'data.frame':        1 obs. of  3 variables:
.. .. ..$ time : Factor w/ 0 levels: NA
.. .. ..$ space: Factor w/ 0 levels: NA
.. .. ..$ value: num NA
..@ lenStruc  :List of 2
.. ..$ estim:'data.frame':        1 obs. of  5 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ length   : Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. ..$ rep  :'data.frame':        1 obs. of  6 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ length   : Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. .. ..$ iter     : num NA
..@ lenVar    :'data.frame':        1 obs. of  5 variables:
.. ..$ time     : Factor w/ 0 levels: NA
.. ..$ space    : Factor w/ 0 levels: NA
.. ..$ technical: Factor w/ 0 levels: NA
.. ..$ length   : Factor w/ 0 levels: NA
.. ..$ value    : num NA
..@ lenNum    :List of 3
.. ..$ ci            :'data.frame':        1 obs. of  7 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ length   : Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. .. ..$ inf      : num NA
.. .. ..$ sup      : num NA
.. ..$ cv            :'data.frame':        1 obs. of  5 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ length   : Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. ..$ DCRcvIndicator: num NA
..@ ageStruc  :List of 2
.. ..$ estim:'data.frame':        1 obs. of  5 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
```

```
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ age      : Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. ..$ rep  :'data.frame':        1 obs. of  6 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ age      : Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. .. ..$ iter     : num NA
..@ ageVar    :'data.frame':        1 obs. of  5 variables:
.. ..$ time     : Factor w/ 0 levels: NA
.. ..$ space    : Factor w/ 0 levels: NA
.. ..$ technical: Factor w/ 0 levels: NA
.. ..$ age      : Factor w/ 0 levels: NA
.. ..$ value    : num NA
..@ ageNum    :List of 3
.. ..$ ci            :'data.frame':        1 obs. of  7 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ age      : Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. .. ..$ inf      : num NA
.. .. ..$ sup      : num NA
.. ..$ cv            :'data.frame':        1 obs. of  5 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ age      : Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. ..$ DCRcvIndicator: num NA
..@ totalN    :List of 2
.. ..$ estim:'data.frame':        1 obs. of  4 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. ..$ rep  :'data.frame':        1 obs. of  5 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. .. ..$ iter     : num NA
..@ totalNvar :'data.frame':        1 obs. of  4 variables:
.. ..$ time     : Factor w/ 0 levels: NA
.. ..$ space    : Factor w/ 0 levels: NA
.. ..$ technical: Factor w/ 0 levels: NA
.. ..$ value    : num NA
..@ totalNnum :List of 3
.. ..$ ci            :'data.frame':        1 obs. of  6 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. .. ..$ inf      : num NA
.. .. ..$ sup      : num NA
.. ..$ cv            :'data.frame':        1 obs. of  4 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
```

```
.. .. ..$ value    : num NA
.. ..$ DCRcvIndicator: num NA
..@ totalW    :List of 2
.. ..$ estim:'data.frame':        1 obs. of  4 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. ..$ rep  :'data.frame':        1 obs. of  5 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. .. ..$ iter     : num NA
..@ totalWvar :'data.frame':        1 obs. of  4 variables:
.. ..$ time     : Factor w/ 0 levels: NA
.. ..$ space    : Factor w/ 0 levels: NA
.. ..$ technical: Factor w/ 0 levels: NA
.. ..$ value    : num NA
..@ totalWnum :List of 3
.. ..$ ci            :'data.frame':        1 obs. of  6 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. .. ..$ inf      : num NA
.. .. ..$ sup      : num NA
.. ..$ cv            :'data.frame':        1 obs. of  4 variables:
.. .. ..$ time     : Factor w/ 0 levels: NA
.. .. ..$ space    : Factor w/ 0 levels: NA
.. .. ..$ technical: Factor w/ 0 levels: NA
.. .. ..$ value    : num NA
.. ..$ DCRcvIndicator: num NA
```

Given the above mentioned stucture, the use of such an object is not straightforward. You will need to create a new object when you want to estimate

- the volume of discards and the related length and age structure per strata
- the volume of catches and the related length and age structure per strata
- the volume of landings and the related length and age structure per strata
- the mean weight at length and mean weight at age per strata
- the maturity stages at length and maturity stages at age per strata
- the sex-ratio at length and sex-ratio at age per strata

You'll have to create a new object also in case you want to use different methods for the same parameter, like analytical, resampling and Bayesian, or discards estimates following different raising methods. In order to fill all the fields, you will need to update your object with several calls to different methods, such as raising for length, raising for age, calculation of confidence intervals and calculation of coefficient of variation. Moreover, when all the fields are completed, a plot method has been developed to draw all the information contained, in a homogeneaous way.

8 Raising discards samples

8.1 Chosing the appropriate raising method

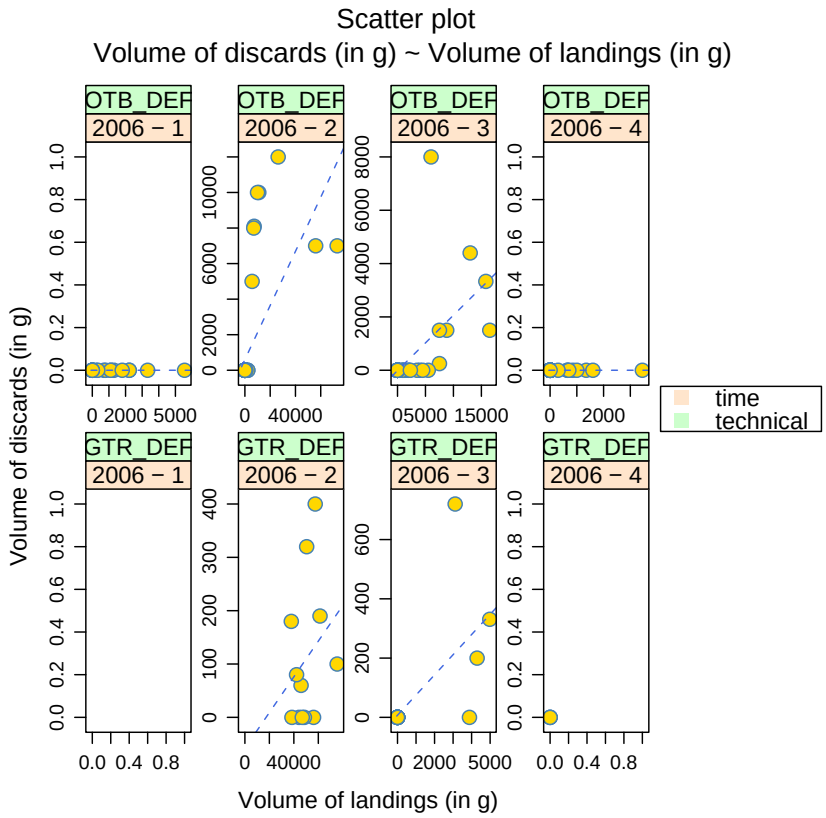
COST has implemented the outcomes of the ICES WKDRP workshop. This workshop concluded that there is a systematic method to assess, compare and finally choose a procedure to raise discards. This procedure was compiled as a key (Discard Raising Procedure Key), where two major issues/characteristics (representativeness and quality) have to be assessed, followed by a subset of decisions that lead to a final choice of procedure. It is essential to apply different raising procedures and compare the resulting discards estimates, as unforeseen problems with the data may only appear through the comparison of different procedures. Ideally the chosen raising procedure should be unbiased, precise and simple.

Raising Procedure Key

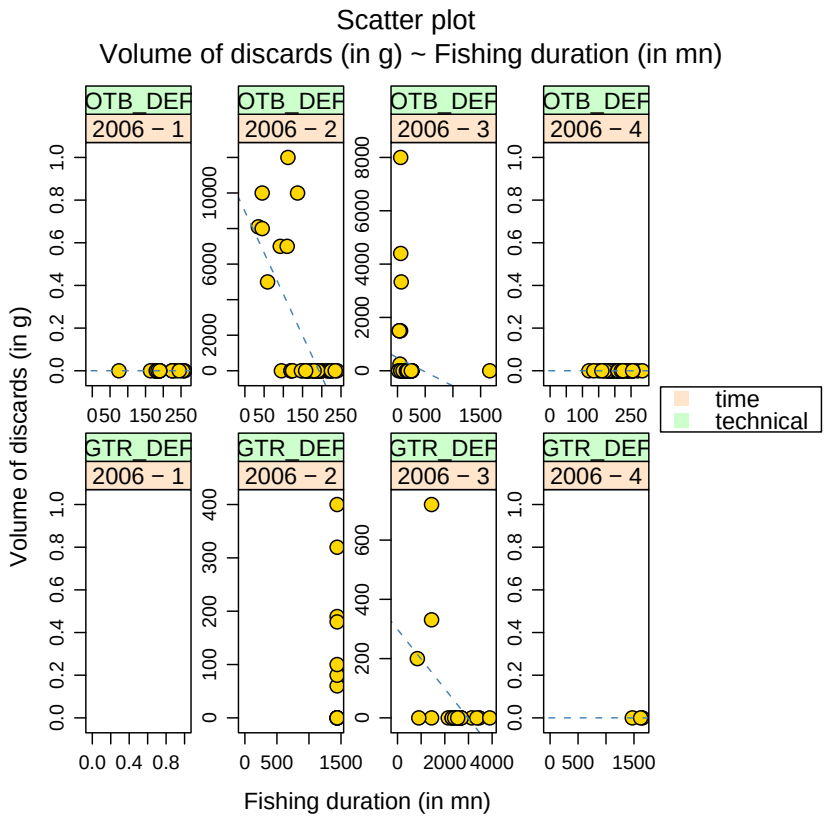
1. Is the sampling representative : mean length of vessels and mean auxiliary variable sampled compared to the population?
   (a) Yes - Raise by trips and go to 2
   (b) No - DON'T raise by trips and go to 2
2. Has the quality of the data used for raising been established (no misreporting and availability)?
   (a) Yes - Go to 3
   (b) No - Go to 6
3. Are the discard and auxiliary variable linearly related : significant relationship and positive slope?
   (a) Yes - Raise by all variables and go to 4
   (b) No - Go to 6
4. Compare the results of the different raising procedures:
   (a) Similar (<10-20% difference) Go to 5
   (b) Dissimilar (>10-20% difference) Go to 1 and find the cause of the difference!
5. Choose a method that is the least biased (trips if applicable, ratio if regression passes trough origin or variable with less concerns) and most precise (compare CV's).
6. When there is no suitable raising procedure then identify the problem, identify the population sampled (different stratification/sampling?) and start again!

In order to help the user find the most suitable raising procedure for discards, COST has developed a method for looking at the linearity between the samples (discards or landings) and and auxiliary vairable such as total landings and fishing time. The method used for this purpose is `disCorrPlot`. The principal arguments allow to test "landings" or "time" as an auxiliary variable and to respect the stratification used or not.

```
> s7dsurv_csv <- csDataVal(s7dsurv_cs)
> s7dsurv_csc <- csDataCons(s7dsurv_csv, strObs)
> Corr_L <- disCorrPlot(s7dsurv_csc,aux="landings",
+ techStrata=TRUE,timeStrata=TRUE, l.reg=TRUE, col="steelblue",bg="gold",lty=2)
```




```
> Corr_E <- disCorrPlot(s7dsurv_csc,aux="time",techStrata=TRUE,
+ timeStrata=TRUE,l.col="steelblue",bg="gold",lty=2)
```



Visually, we can have an idea of the linear relationship between the variables, but we understand that figures are preferable. the relationship wiht the fishing time shows some negative slope, and this is the worst situation for raising purpose. Any negative slope discredit totally the use of the variable as an auxiliary for raising purpose.

```
> lm_L <- lm(Corr_L$disVol~Corr_L$auxVar)
> summary(lm_L)$r.squared
```

```
[1] 0.08647183
```

```
> R2_L <- tapply(1:nrow(Corr_L),list(Corr_L$time,Corr_L$technical),function(x)
+ summary(lm(Corr_L$disVol[x]~Corr_L$auxVar[x]))$r.squared)
> R2_L
```

```
           GTR_DEF   OTB_DEF OTM_SPF
2006 - 1        NA       NaN      NA
2006 - 2 0.0697914 0.3920408      NA
2006 - 3 0.4307892 0.3896124      NA
2006 - 4 0.0000000       NaN      NA
```

```
> lm_E <- lm(Corr_E$disVol~Corr_E$auxVar)
> summary(lm_E)$r.squared
```

```
[1] 0.01207936
```

```
> R2_E <- tapply(1:nrow(Corr_E),list(Corr_E$time,Corr_E$technical),function(x)
+ summary(lm(Corr_E$disVol[x]~Corr_E$auxVar[x]))$r.squared)
> R2_E
```

```
           GTR_DEF    OTB_DEF OTM_SPF
2006 - 1        NA        NaN      NA
2006 - 2 0.0000000 0.51683709      NA
2006 - 3 0.2150383 0.02933148      NA
2006 - 4       NaN        NaN      NA
```

The landings seem to be a good candidate here for raising OTB_DEF, but no raising variables seem to fit correctly for GTR_DEF. If the number of trips is known without bias, raising discards for GTR_DEF should be done by trips. COST has also implemented a new method developed by Lejamtel and Vigneau (2008) based on days at sea, which are values usually known accurately.

8.2 Raising methods

Three of the four raising procedures are described in Vigneau (2007) as a specific interpretation of Cochran (1977), namely raising by trips, raising by hauls and raising by an auxiliary variable. The fourth raising procedure, raising by fishing days, is a variant of raising by hauls introducing an intermediate sampling level (days) between the trips and the fishing operations. This method has been the object of a specific development for the COST project. The generic method developed is `totVolume`, and the difference is made by defining appropriate arguments.

8.2.1 Raising by trips

The call to the `totVolume` method is done with argument `type="trip"` and successively with val="weight" for estimating the total weight per strata and val="nAtLength" for estimating the numberlength per strata. Furthermore, a call to the `dbeCalc` method with successively `type="CV"` and `type="CI"` completes the `disEstim` object with the precision estimates.

```
> s7dsurv_cec <- ceDataCons(s7d_cev, strObs)
> s7dsurv_clc <- clDataCons(s7d_clv, strObs)
> disEstim <- totVolume(disEstim,s7dsurv_csc,s7dsurv_cec, type="trip",val="weight")
> disEstim@totalW$estim
```

```
      time space technical    value
1 2006 - 1   all   OTB_DEF    0.000
2 2006 - 2   all   GTR_DEF  387.030
3 2006 - 2   all   OTB_DEF 6383.257
```

```
4 2006 - 3   all   GTR_DEF  135.525
5 2006 - 3   all   OTB_DEF 2739.746
6 2006 - 4   all   GTR_DEF    0.000
7 2006 - 4   all   OTB_DEF    0.000
```

```
> disEstim <- dbeCalc(disEstim,type="CV",vrbl="w",update=TRUE)
> disEstim@totalWnum$cv
```

```
      time space technical     value
1 2006 - 1   all   OTB_DEF       NaN
2 2006 - 2   all   GTR_DEF 0.0000000
3 2006 - 2   all   OTB_DEF 0.6134160
4 2006 - 3   all   GTR_DEF 0.7726972
5 2006 - 3   all   OTB_DEF 0.6524121
6 2006 - 4   all   GTR_DEF       NaN
7 2006 - 4   all   OTB_DEF       NaN
```

```
> disEstim <- dbeCalc(disEstim,type="CI",vrbl="w",update=TRUE)
> disEstim <- totVolume(disEstim,s7dsurv_csc,s7dsurv_cec, type="trip",val="nAtLength")
> # We look only at a part of the object (length structure quarter 3)
> disEstim@lenStruc$estim[disEstim@lenStruc$estim$time == "2006 - 3",]
```

```
       time space technical length     value
37 2006 - 3   all   GTR_DEF    120    0.0000
38 2006 - 3   all   GTR_DEF    140    0.0000
39 2006 - 3   all   GTR_DEF    150    0.0000
40 2006 - 3   all   GTR_DEF    160    0.0000
41 2006 - 3   all   GTR_DEF    170    0.0000
42 2006 - 3   all   GTR_DEF    180    0.0000
43 2006 - 3   all   GTR_DEF    190  216.6667
44 2006 - 3   all   GTR_DEF    200    0.0000
45 2006 - 3   all   GTR_DEF    210  216.6667
46 2006 - 3   all   GTR_DEF    220    0.0000
47 2006 - 3   all   GTR_DEF    230  650.0000
48 2006 - 3   all   GTR_DEF    240  108.3333
49 2006 - 3   all   OTB_DEF    120    0.0000
50 2006 - 3   all   OTB_DEF    140    0.0000
51 2006 - 3   all   OTB_DEF    150    0.0000
52 2006 - 3   all   OTB_DEF    160    0.0000
53 2006 - 3   all   OTB_DEF    170    0.0000
54 2006 - 3   all   OTB_DEF    180    0.0000
55 2006 - 3   all   OTB_DEF    190  841.5289
56 2006 - 3   all   OTB_DEF    200  496.6400
57 2006 - 3   all   OTB_DEF    210 9208.0160
58 2006 - 3   all   OTB_DEF    220 6518.4000
59 2006 - 3   all   OTB_DEF    230 8877.4400
60 2006 - 3   all   OTB_DEF    240    0.0000
```

```
> disEstim <- dbeCalc(disEstim,type="CV",vrbl="l",update=TRUE)
> # We look only at a part of the object (CV of length structure quarter 3)
> disEstim@lenNum$cv[disEstim@lenNum$cv$time == "2006 - 3",]
```

```
       time space technical length     value
37 2006 - 3   all   GTR_DEF    120       NaN
38 2006 - 3   all   GTR_DEF    140       NaN
39 2006 - 3   all   GTR_DEF    150       NaN
40 2006 - 3   all   GTR_DEF    160       NaN
41 2006 - 3   all   GTR_DEF    170       NaN
42 2006 - 3   all   GTR_DEF    180       NaN
43 2006 - 3   all   GTR_DEF    190 1.0000000
44 2006 - 3   all   GTR_DEF    200       NaN
45 2006 - 3   all   GTR_DEF    210 0.5000000
46 2006 - 3   all   GTR_DEF    220       NaN
```

```
47 2006 - 3   all   GTR_DEF    230 0.7637626
48 2006 - 3   all   GTR_DEF    240 1.0000000
49 2006 - 3   all   OTB_DEF    120       NaN
50 2006 - 3   all   OTB_DEF    140       NaN
51 2006 - 3   all   OTB_DEF    150       NaN
52 2006 - 3   all   OTB_DEF    160       NaN
53 2006 - 3   all   OTB_DEF    170       NaN
54 2006 - 3   all   OTB_DEF    180       NaN
55 2006 - 3   all   OTB_DEF    190 1.0009547
56 2006 - 3   all   OTB_DEF    200 1.0024133
57 2006 - 3   all   OTB_DEF    210 0.7139135
58 2006 - 3   all   OTB_DEF    220 0.7402396
59 2006 - 3   all   OTB_DEF    230 0.8000875
60 2006 - 3   all   OTB_DEF    240       NaN
```

```
> disEstim <- dbeCalc(disEstim,type="CI",vrbl="l",update=TRUE)
> # We look only at a part of the object (CI of length structure quarter 3)
> disEstim@lenNum$ci[disEstim@lenNum$ci$time == "2006 - 3",]
```

```
       time space technical length     value      inf        sup
37 2006 - 3   all   GTR_DEF    120    0.0000 0.000000     0.0000
38 2006 - 3   all   GTR_DEF    140    0.0000 0.000000     0.0000
39 2006 - 3   all   GTR_DEF    150    0.0000 0.000000     0.0000
40 2006 - 3   all   GTR_DEF    160    0.0000 0.000000     0.0000
41 2006 - 3   all   GTR_DEF    170    0.0000 0.000000     0.0000
42 2006 - 3   all   GTR_DEF    180    0.0000 0.000000     0.0000
43 2006 - 3   all   GTR_DEF    190  216.6667 0.000000   641.3255
44 2006 - 3   all   GTR_DEF    200    0.0000 0.000000     0.0000
45 2006 - 3   all   GTR_DEF    210  216.6667 4.337235   428.9961
46 2006 - 3   all   GTR_DEF    220    0.0000 0.000000     0.0000
47 2006 - 3   all   GTR_DEF    230  650.0000 0.000000  1623.0157
48 2006 - 3   all   GTR_DEF    240  108.3333 0.000000   320.6628
49 2006 - 3   all   OTB_DEF    120    0.0000 0.000000     0.0000
50 2006 - 3   all   OTB_DEF    140    0.0000 0.000000     0.0000
51 2006 - 3   all   OTB_DEF    150    0.0000 0.000000     0.0000
52 2006 - 3   all   OTB_DEF    160    0.0000 0.000000     0.0000
53 2006 - 3   all   OTB_DEF    170    0.0000 0.000000     0.0000
54 2006 - 3   all   OTB_DEF    180    0.0000 0.000000     0.0000
55 2006 - 3   all   OTB_DEF    190  841.5289 0.000000  2492.4699
56 2006 - 3   all   OTB_DEF    200  496.6400 0.000000  1472.3856
57 2006 - 3   all   OTB_DEF    210 9208.0160 0.000000 22092.2841
58 2006 - 3   all   OTB_DEF    220 6518.4000 0.000000 15975.5753
59 2006 - 3   all   OTB_DEF    230 8877.4400 0.000000 22798.5332
60 2006 - 3   all   OTB_DEF    240    0.0000 0.000000     0.0000
```

The DCR indicator for precision are, respectively for the weight estimates and length structure estimates

```
> disEstim@totalWnum$DCRcvIndicator
```

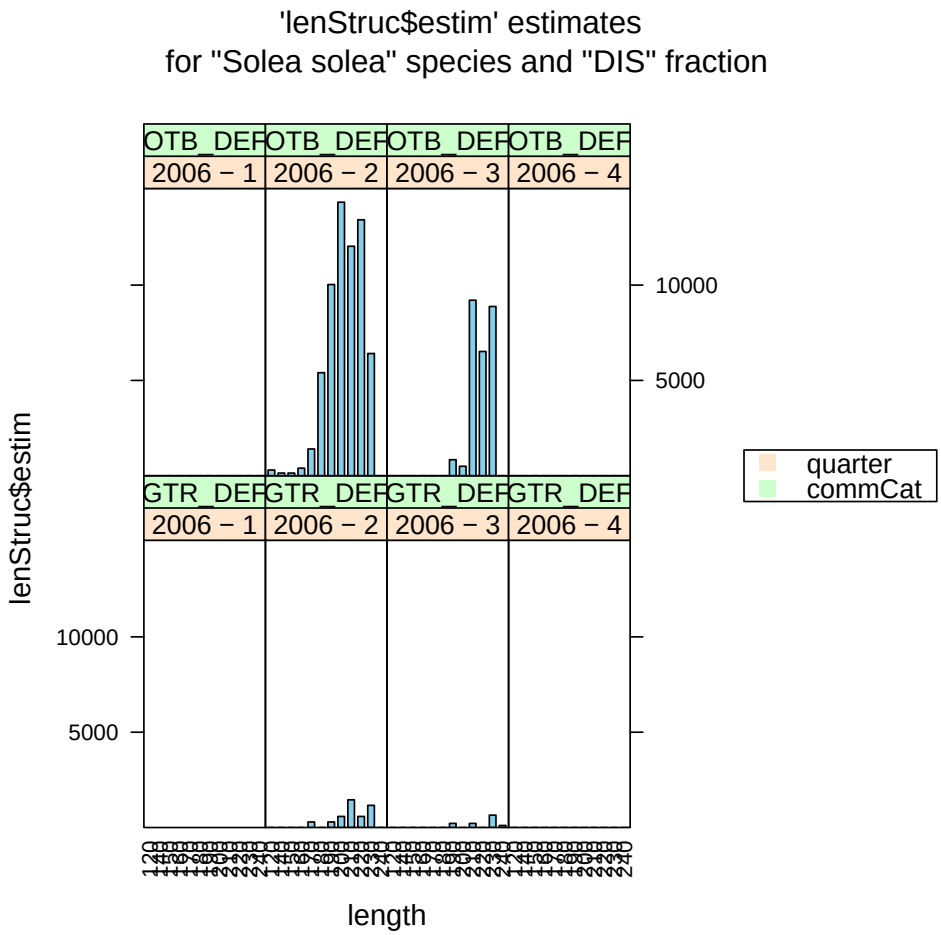
```
[1] 0.6021171
```

```
> disEstim@lenNum$DCRcvIndicator
```

```
[1] 0.7820517
```

Plots of the length structure of discards and related CVs is done through the call of the generic plot method

```
> dbePlot(disEstim, elmt="lenStruc$estim")
> dbePlot(disEstim, elmt="lenVar")
```



8.2.2 Raising by fishing operations

The call to the `totVolume` method is done with argument `type="fo"` and successively with val="weight" for estimating the total weight per strata and val="nAtLength" for estimating the numberlength per strata. Furthermore, a call to the `dbeCalc` method with successively `type="CV"` and `type="CI"` completes the `disEstim` object with the precision estimates.

```
> disEstim <- totVolume(disEstim,s7dsurv_csc,s7dsurv_cec, type="fo",val="weight")
> disEstim@totalW$estim

      time space technical     value
1 2006 - 1   all   OTB_DEF    0.0000
2 2006 - 2   all   GTR_DEF  535.3250
3 2006 - 2   all   OTB_DEF 6726.0461
4 2006 - 3   all   GTR_DEF  264.0801
5 2006 - 3   all   OTB_DEF 1478.7670
6 2006 - 4   all   GTR_DEF    0.0000
7 2006 - 4   all   OTB_DEF    0.0000

> disEstim <- dbeCalc(disEstim,type="CV",vrbl="w",update=TRUE)
> disEstim@totalWnum$cv

      time space technical    value
1 2006 - 1   all   OTB_DEF      NaN
2 2006 - 2   all   GTR_DEF 0.000000
3 2006 - 2   all   OTB_DEF 1.458473
```

```
4 2006 - 3   all   GTR_DEF 2.425776
5 2006 - 3   all   OTB_DEF 1.988564
6 2006 - 4   all   GTR_DEF      NaN
7 2006 - 4   all   OTB_DEF      NaN
```

```
> disEstim <- dbeCalc(disEstim,type="CI",vrbl="w",update=TRUE)
> disEstim <- totVolume(disEstim,s7dsurv_csc,s7dsurv_cec, type="fo",val="nAtLength")
> # We look only at a part of the object (length structure quarter 3)
> disEstim@lenStruc$estim[disEstim@lenStruc$estim$time == "2006 - 3",]
```

```
       time space technical length     value
37 2006 - 3   all   GTR_DEF    120    0.0000
38 2006 - 3   all   GTR_DEF    140    0.0000
39 2006 - 3   all   GTR_DEF    150    0.0000
40 2006 - 3   all   GTR_DEF    160    0.0000
41 2006 - 3   all   GTR_DEF    170    0.0000
42 2006 - 3   all   GTR_DEF    180    0.0000
43 2006 - 3   all   GTR_DEF    190  422.1905
44 2006 - 3   all   GTR_DEF    200    0.0000
45 2006 - 3   all   GTR_DEF    210  422.1905
46 2006 - 3   all   GTR_DEF    220    0.0000
47 2006 - 3   all   GTR_DEF    230 1266.5714
48 2006 - 3   all   GTR_DEF    240  211.0952
49 2006 - 3   all   OTB_DEF    120    0.0000
50 2006 - 3   all   OTB_DEF    140    0.0000
51 2006 - 3   all   OTB_DEF    150    0.0000
52 2006 - 3   all   OTB_DEF    160    0.0000
53 2006 - 3   all   OTB_DEF    170    0.0000
54 2006 - 3   all   OTB_DEF    180    0.0000
55 2006 - 3   all   OTB_DEF    190  454.2119
56 2006 - 3   all   OTB_DEF    200  268.0595
57 2006 - 3   all   OTB_DEF    210 4969.9907
58 2006 - 3   all   OTB_DEF    220 3518.2810
59 2006 - 3   all   OTB_DEF    230 4791.5636
60 2006 - 3   all   OTB_DEF    240    0.0000
```

```
> disEstim <- dbeCalc(disEstim,type="CV",vrbl="l",update=TRUE)
> # We look only at a part of the object (CV of length structure quarter 3)
> disEstim@lenNum$cv[disEstim@lenNum$cv$time == "2006 - 3",]
```

```
       time space technical length    value
37 2006 - 3   all   GTR_DEF    120      NaN
38 2006 - 3   all   GTR_DEF    140      NaN
39 2006 - 3   all   GTR_DEF    150      NaN
40 2006 - 3   all   GTR_DEF    160      NaN
41 2006 - 3   all   GTR_DEF    170      NaN
42 2006 - 3   all   GTR_DEF    180      NaN
43 2006 - 3   all   GTR_DEF    190 2.341256
44 2006 - 3   all   GTR_DEF    200      NaN
45 2006 - 3   all   GTR_DEF    210 2.737133
46 2006 - 3   all   GTR_DEF    220      NaN
47 2006 - 3   all   GTR_DEF    230 2.430331
48 2006 - 3   all   GTR_DEF    240 2.341256
49 2006 - 3   all   OTB_DEF    120      NaN
50 2006 - 3   all   OTB_DEF    140      NaN
51 2006 - 3   all   OTB_DEF    150      NaN
52 2006 - 3   all   OTB_DEF    160      NaN
53 2006 - 3   all   OTB_DEF    170      NaN
54 2006 - 3   all   OTB_DEF    180      NaN
55 2006 - 3   all   OTB_DEF    190 2.268247
56 2006 - 3   all   OTB_DEF    200 2.269440
57 2006 - 3   all   OTB_DEF    210 1.859768
58 2006 - 3   all   OTB_DEF    220 2.068286
```

```
59 2006 - 3   all   OTB_DEF    230 2.115867
60 2006 - 3   all   OTB_DEF    240      NaN
```

```
> disEstim <- dbeCalc(disEstim,type="CI",vrbl="l",update=TRUE)
> # We look only at a part of the object (CI of length structure quarter 3)
> disEstim@lenNum$ci[disEstim@lenNum$ci$time == "2006 - 3",]
```

```
       time space technical length     value inf       sup
37 2006 - 3   all   GTR_DEF    120    0.0000   0     0.000
38 2006 - 3   all   GTR_DEF    140    0.0000   0     0.000
39 2006 - 3   all   GTR_DEF    150    0.0000   0     0.000
40 2006 - 3   all   GTR_DEF    160    0.0000   0     0.000
41 2006 - 3   all   GTR_DEF    170    0.0000   0     0.000
42 2006 - 3   all   GTR_DEF    180    0.0000   0     0.000
43 2006 - 3   all   GTR_DEF    190  422.1905   0  2359.529
44 2006 - 3   all   GTR_DEF    200    0.0000   0     0.000
45 2006 - 3   all   GTR_DEF    210  422.1905   0  2687.108
46 2006 - 3   all   GTR_DEF    220    0.0000   0     0.000
47 2006 - 3   all   GTR_DEF    230 1266.5714   0  7299.708
48 2006 - 3   all   GTR_DEF    240  211.0952   0  1179.764
49 2006 - 3   all   OTB_DEF    120    0.0000   0     0.000
50 2006 - 3   all   OTB_DEF    140    0.0000   0     0.000
51 2006 - 3   all   OTB_DEF    150    0.0000   0     0.000
52 2006 - 3   all   OTB_DEF    160    0.0000   0     0.000
53 2006 - 3   all   OTB_DEF    170    0.0000   0     0.000
54 2006 - 3   all   OTB_DEF    180    0.0000   0     0.000
55 2006 - 3   all   OTB_DEF    190  454.2119   0  2473.494
56 2006 - 3   all   OTB_DEF    200  268.0595   0  1460.394
57 2006 - 3   all   OTB_DEF    210 4969.9907   0 23086.000
58 2006 - 3   all   OTB_DEF    220 3518.2810   0 17780.570
59 2006 - 3   all   OTB_DEF    230 4791.5636   0 24662.290
60 2006 - 3   all   OTB_DEF    240    0.0000   0     0.000
```

The DCR indicator for precision are, respectively for the weight estimates and length structure estimates

```
> disEstim@totalWnum$DCRcvIndicator
```

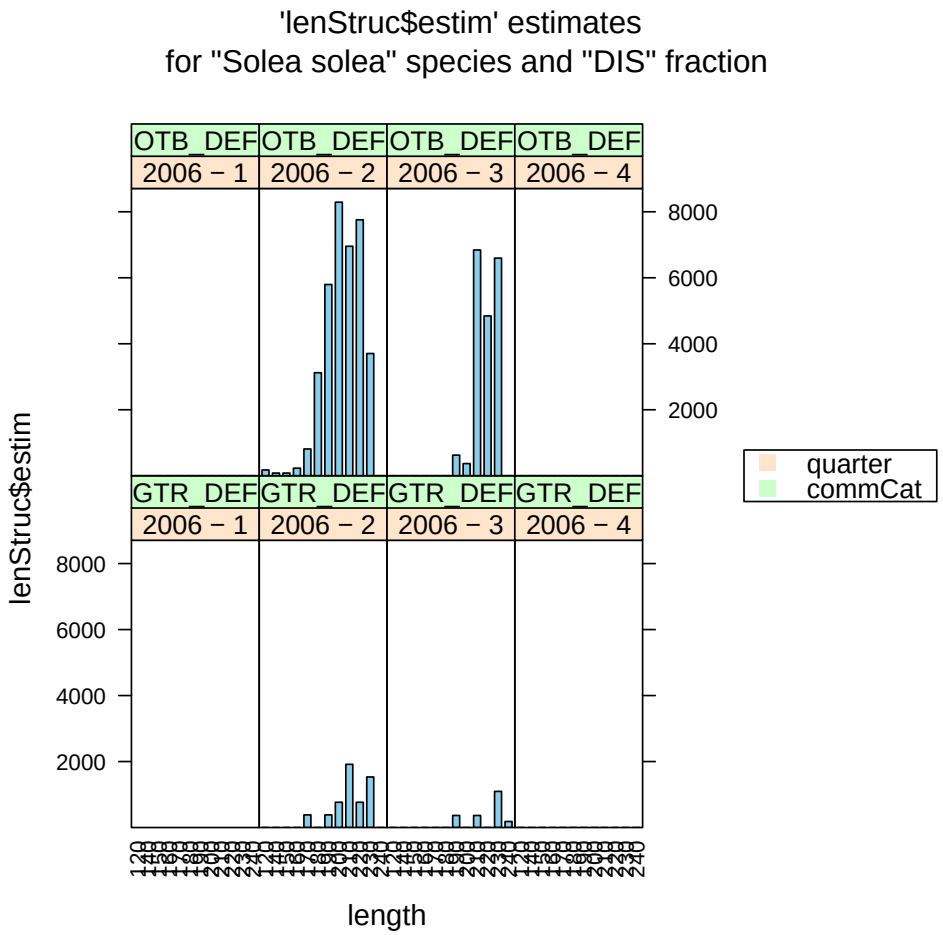
```
[1] 1.487189
```

```
> disEstim@lenNum$DCRcvIndicator
```

```
[1] 1.596369
```

Plots of the length structure of discards and related CVs is done through the call of the generic plot method

```
> dbePlot(disEstim, elmt="lenStruc$estim")
> dbePlot(disEstim, elmt="lenVar")
```



8.2.3 Raising by an auxiliary variable

The call to the `totVolume` method is done with argument `type="time"` or `"landings"`, and successively with val="weight" for estimating the total weight per strata and val="nAtLength" for estimating the numberlength per strata. Furthermore, a call to the `dbeCalc` method with successively `type="CV"` and `type="CI"` completes the `disEstim` object with the precision estimates.

```
> disEstim <- totVolume(disEstim,s7dsurv_csc,s7dsurv_cec, s7dsurv_clc, type="landings",val="weight")
> disEstim@totalW$estim

      time space technical    value
1 2006 - 1   all   OTB_DEF     0.00
2 2006 - 2   all   GTR_DEF  1351.35
3 2006 - 2   all   OTB_DEF 53598.84
4 2006 - 3   all   GTR_DEF 33301.58
5 2006 - 3   all   OTB_DEF 51183.31
6 2006 - 4   all   GTR_DEF     0.00
7 2006 - 4   all   OTB_DEF     0.00

> disEstim <- dbeCalc(disEstim,type="CV",vrbl="w",update=TRUE)
> disEstim@totalWnum$cv

      time space technical       value
1 2006 - 1   all   OTB_DEF         NaN
2 2006 - 2   all   GTR_DEF 0.000000000
3 2006 - 2   all   OTB_DEF 0.060387556
```

```
4 2006 - 3   all   GTR_DEF 0.002416767
5 2006 - 3   all   OTB_DEF 0.026163070
6 2006 - 4   all   GTR_DEF         NaN
7 2006 - 4   all   OTB_DEF         NaN
```

```
> disEstim <- dbeCalc(disEstim,type="CI",vrbl="w",update=TRUE)
> disEstim <- totVolume(disEstim,s7dsurv_csc,s7dsurv_cec, s7dsurv_clc, type="landings",val="nAtLength")
> # We look only at a part of the object (length structure quarter 3)
> disEstim@lenStruc$estim[disEstim@lenStruc$estim$time == "2006 - 3",]
```

```
       time space technical length     value
19 2006 - 3   all   GTR_DEF    190  6422.557
20 2006 - 3   all   GTR_DEF    200     0.000
21 2006 - 3   all   GTR_DEF    210  6422.557
22 2006 - 3   all   GTR_DEF    220     0.000
23 2006 - 3   all   GTR_DEF    230 19267.671
24 2006 - 3   all   GTR_DEF    240  3211.278
25 2006 - 3   all   OTB_DEF    190     0.000
26 2006 - 3   all   OTB_DEF    200     0.000
27 2006 - 3   all   OTB_DEF    210     0.000
28 2006 - 3   all   OTB_DEF    220     0.000
29 2006 - 3   all   OTB_DEF    230     0.000
30 2006 - 3   all   OTB_DEF    240     0.000
```

```
> disEstim <- dbeCalc(disEstim,type="CV",vrbl="l",update=TRUE)
> # We look only at a part of the object (CV of length structure quarter 3)
> disEstim@lenNum$cv[disEstim@lenNum$cv$time == "2006 - 3",]
```

```
       time space technical length       value
19 2006 - 3   all   GTR_DEF    190 0.030390181
20 2006 - 3   all   GTR_DEF    200         NaN
21 2006 - 3   all   GTR_DEF    210 0.001174583
22 2006 - 3   all   GTR_DEF    220         NaN
23 2006 - 3   all   GTR_DEF    230 0.020651648
24 2006 - 3   all   GTR_DEF    240 0.030390181
25 2006 - 3   all   OTB_DEF    190         NaN
26 2006 - 3   all   OTB_DEF    200         NaN
27 2006 - 3   all   OTB_DEF    210         NaN
28 2006 - 3   all   OTB_DEF    220         NaN
29 2006 - 3   all   OTB_DEF    230         NaN
30 2006 - 3   all   OTB_DEF    240         NaN
```

```
> disEstim <- dbeCalc(disEstim,type="CI",vrbl="l",update=TRUE)
> # We look only at a part of the object (CI of length structure quarter 3)
> disEstim@lenNum$ci[disEstim@lenNum$ci$time == "2006 - 3",]
```

```
       time space technical length     value       inf       sup
19 2006 - 3   all   GTR_DEF    190  6422.557  6040.006  6805.108
20 2006 - 3   all   GTR_DEF    200     0.000     0.000     0.000
21 2006 - 3   all   GTR_DEF    210  6422.557  6407.771  6437.343
22 2006 - 3   all   GTR_DEF    220     0.000     0.000     0.000
23 2006 - 3   all   GTR_DEF    230 19267.671 18487.783 20047.558
24 2006 - 3   all   GTR_DEF    240  3211.278  3020.003  3402.554
25 2006 - 3   all   OTB_DEF    190     0.000     0.000     0.000
26 2006 - 3   all   OTB_DEF    200     0.000     0.000     0.000
27 2006 - 3   all   OTB_DEF    210     0.000     0.000     0.000
28 2006 - 3   all   OTB_DEF    220     0.000     0.000     0.000
29 2006 - 3   all   OTB_DEF    230     0.000     0.000     0.000
30 2006 - 3   all   OTB_DEF    240     0.000     0.000     0.000
```

The DCR indicator for precision are, respectively for the weight estimates and length structure estimates

```
> disEstim@totalWnum$DCRcvIndicator
```

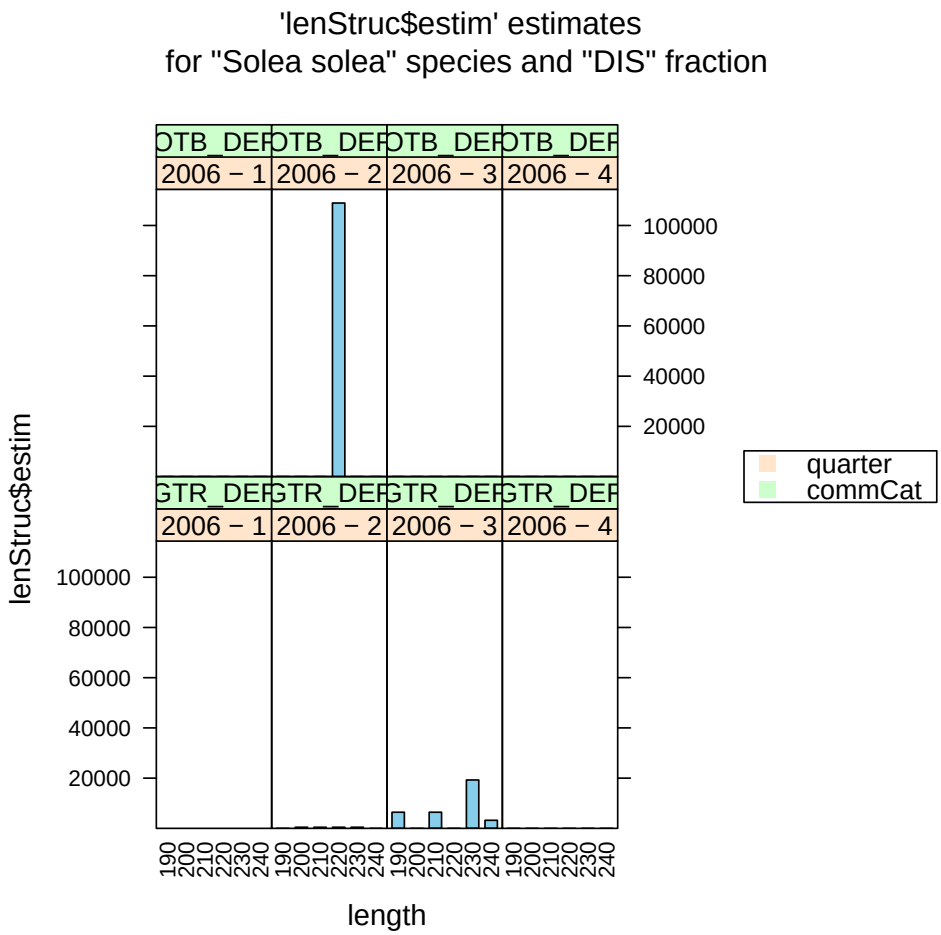
```
[1] 0.3339402
```

```
> disEstim@lenNum$DCRcvIndicator
```

```
[1] 0.7451818
```

Plots of the length structure of discards and related CVs is done through the call of the generic plot method

```
> dbePlot(disEstim, elmt="lenStruc$estim")
> dbePlot(disEstim, elmt="lenVar")
```



8.2.4 Raising by fishing days

The call to the `totVolume` method is done with argument `type="trip"` and successively with val="weight" for estimating the total weight per strata and val="nAtLength" for estimating the numberlength per strata. Furthermore, a call to the `dbeCalc` method with successively `type="CV"` and `type="CI"` completes the `disEstim` object with the precision estimates.

```
> disEstim <- totVolume(disEstim,s7dsurv_csc,s7dsurv_cec, type="fd",val="weight")
> disEstim@totalW$estim
```

```
      time space technical     value
1 2006 - 1   all   OTB_DEF    0.0000
2 2006 - 2   all   GTR_DEF 1537.4800
3 2006 - 2   all   OTB_DEF 3879.5824
4 2006 - 3   all   GTR_DEF  152.4656
5 2006 - 3   all   OTB_DEF 1768.8974
6 2006 - 4   all   GTR_DEF    0.0000
7 2006 - 4   all   OTB_DEF    0.0000
```

```
> disEstim <- dbeCalc(disEstim,type="CV",vrbl="w",update=TRUE)
> disEstim@totalWnum$cv

      time space technical     value
1 2006 - 1   all   OTB_DEF       NaN
2 2006 - 2   all   GTR_DEF 0.0000000
3 2006 - 2   all   OTB_DEF 0.3214746
4 2006 - 3   all   GTR_DEF 0.8864942
5 2006 - 3   all   OTB_DEF 0.3364609
6 2006 - 4   all   GTR_DEF       NaN
7 2006 - 4   all   OTB_DEF       NaN

> disEstim <- dbeCalc(disEstim,type="CI",vrbl="w",update=TRUE)
> disEstim <- totVolume(disEstim,s7dsurv_csc,s7dsurv_cec,type="fd",val="nAtLength")
> # We look only at a part of the object (length structure quarter 3)
> disEstim@lenStruc$estim[disEstim@lenStruc$estim$time == "2006 - 3",]

       time space technical length     value
37 2006 - 3   all   GTR_DEF    120    0.0000
38 2006 - 3   all   GTR_DEF    140    0.0000
39 2006 - 3   all   GTR_DEF    150    0.0000
40 2006 - 3   all   GTR_DEF    160    0.0000
41 2006 - 3   all   GTR_DEF    170    0.0000
42 2006 - 3   all   GTR_DEF    180    0.0000
43 2006 - 3   all   GTR_DEF    190  243.7500
44 2006 - 3   all   GTR_DEF    200    0.0000
45 2006 - 3   all   GTR_DEF    210  243.7500
46 2006 - 3   all   GTR_DEF    220    0.0000
47 2006 - 3   all   GTR_DEF    230  731.2500
48 2006 - 3   all   GTR_DEF    240  121.8750
49 2006 - 3   all   OTB_DEF    120    0.0000
50 2006 - 3   all   OTB_DEF    140    0.0000
51 2006 - 3   all   OTB_DEF    150    0.0000
52 2006 - 3   all   OTB_DEF    160    0.0000
53 2006 - 3   all   OTB_DEF    170    0.0000
54 2006 - 3   all   OTB_DEF    180    0.0000
55 2006 - 3   all   OTB_DEF    190  714.1313
56 2006 - 3   all   OTB_DEF    200  210.7273
57 2006 - 3   all   OTB_DEF    210 6801.3983
58 2006 - 3   all   OTB_DEF    220 3968.6970
59 2006 - 3   all   OTB_DEF    230 5391.1061
60 2006 - 3   all   OTB_DEF    240    0.0000

> disEstim <- dbeCalc(disEstim,type="CV",vrbl="l",update=TRUE)
> # We look only at a part of the object (CV of length structure quarter 3)
> disEstim@lenNum$cv[disEstim@lenNum$cv$time == "2006 - 3",]

       time space technical length     value
37 2006 - 3   all   GTR_DEF    120       NaN
38 2006 - 3   all   GTR_DEF    140       NaN
39 2006 - 3   all   GTR_DEF    150       NaN
40 2006 - 3   all   GTR_DEF    160       NaN
41 2006 - 3   all   GTR_DEF    170       NaN
42 2006 - 3   all   GTR_DEF    180       NaN
43 2006 - 3   all   GTR_DEF    190 1.0888089
44 2006 - 3   all   GTR_DEF    200       NaN
45 2006 - 3   all   GTR_DEF    210 0.6645740
46 2006 - 3   all   GTR_DEF    220       NaN
47 2006 - 3   all   GTR_DEF    230 0.8787815
48 2006 - 3   all   GTR_DEF    240 1.0888089
49 2006 - 3   all   OTB_DEF    120       NaN
50 2006 - 3   all   OTB_DEF    140       NaN
51 2006 - 3   all   OTB_DEF    150       NaN
```

```
52 2006 - 3   all   OTB_DEF    160       NaN
53 2006 - 3   all   OTB_DEF    170       NaN
54 2006 - 3   all   OTB_DEF    180       NaN
55 2006 - 3   all   OTB_DEF    190 0.5158496
56 2006 - 3   all   OTB_DEF    200 0.5133192
57 2006 - 3   all   OTB_DEF    210 0.4032550
58 2006 - 3   all   OTB_DEF    220 0.3655572
59 2006 - 3   all   OTB_DEF    230 0.3969691
60 2006 - 3   all   OTB_DEF    240       NaN
```

```
> disEstim <- dbeCalc(disEstim,type="CI",vrbl="l",update=TRUE)
> # We look only at a part of the object (CV of length structure quarter 3)
> disEstim@lenNum$ci[disEstim@lenNum$ci$time == "2006 - 3",]
```

```
       time space technical length     value      inf        sup
37 2006 - 3   all   GTR_DEF    120    0.0000    0.000     0.0000
38 2006 - 3   all   GTR_DEF    140    0.0000    0.000     0.0000
39 2006 - 3   all   GTR_DEF    150    0.0000    0.000     0.0000
40 2006 - 3   all   GTR_DEF    160    0.0000    0.000     0.0000
41 2006 - 3   all   GTR_DEF    170    0.0000    0.000     0.0000
42 2006 - 3   all   GTR_DEF    180    0.0000    0.000     0.0000
43 2006 - 3   all   GTR_DEF    190  243.7500    0.000   763.9189
44 2006 - 3   all   GTR_DEF    200    0.0000    0.000     0.0000
45 2006 - 3   all   GTR_DEF    210  243.7500    0.000   561.2444
46 2006 - 3   all   GTR_DEF    220    0.0000    0.000     0.0000
47 2006 - 3   all   GTR_DEF    230  731.2500    0.000  1990.7405
48 2006 - 3   all   GTR_DEF    240  121.8750    0.000   381.9594
49 2006 - 3   all   OTB_DEF    120    0.0000    0.000     0.0000
50 2006 - 3   all   OTB_DEF    140    0.0000    0.000     0.0000
51 2006 - 3   all   OTB_DEF    150    0.0000    0.000     0.0000
52 2006 - 3   all   OTB_DEF    160    0.0000    0.000     0.0000
53 2006 - 3   all   OTB_DEF    170    0.0000    0.000     0.0000
54 2006 - 3   all   OTB_DEF    180    0.0000    0.000     0.0000
55 2006 - 3   all   OTB_DEF    190  714.1313    0.000  1436.1514
56 2006 - 3   all   OTB_DEF    200  210.7273    0.000   422.7373
57 2006 - 3   all   OTB_DEF    210 6801.3983 1425.809 12176.9874
58 2006 - 3   all   OTB_DEF    220 3968.6970 1125.209  6812.1849
59 2006 - 3   all   OTB_DEF    230 5391.1061 1196.582  9585.6299
60 2006 - 3   all   OTB_DEF    240    0.0000    0.000     0.0000
```

The DCR indicator for precision are, respectively for the weight estimates and length structure estimates

```
> disEstim@totalWnum$DCRcvIndicator
```

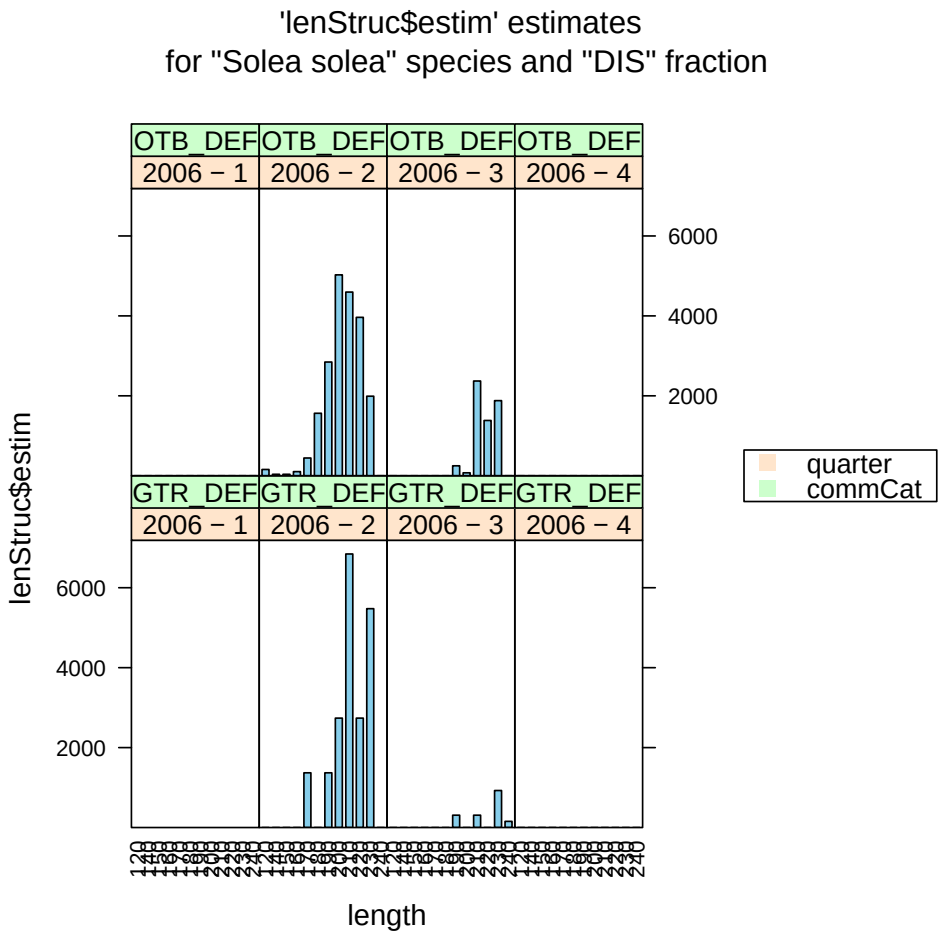
```
[1] 0.2694736
```

```
> disEstim@lenNum$DCRcvIndicator
```

```
[1] 0.3254903
```

Plots of the length structure of discards and related CVs is done through the call of the generic plot method

```
> dbePlot(disEstim, elmt="lenStruc$estim")
> dbePlot(disEstim, elmt="lenVar")
```



9 Raising length structure from market sampling

9.1 Analytical method

The first step is to create the empty object, that will be given the appropriate values for the descritor fields.

```
> lanEstim <- dbeObject(desc="Sole VIId Landingss",species="Solea solea",catchCat="LAN",strataDesc=strD
+ methodDesc="analytical")
```

Here we will consider the simpler case, which is one species in the commercial sampling tables (here Solea solea) matches the commercial naming of the taxon in the commercial landings (table cl). In this case, the only arguments to pass to the function are the dbe object, the consolidated cs and cl datasets. Note that there is the possibility to split the number of strata displayed per page in the plot function with the use of the argument `layout=c(ncol,nrow)`. For details on the use of the function `update`, see section 3.3.1.

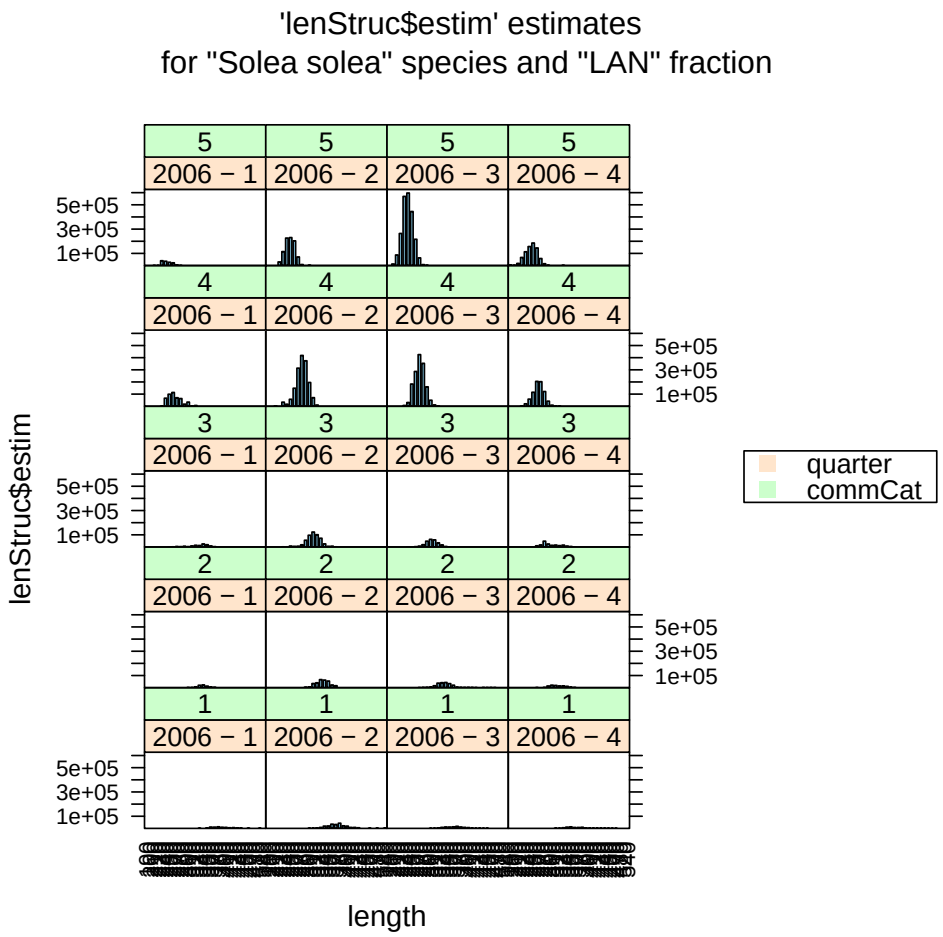
```
> lanEstim <- RaiseLgth(lanEstim, s7d_csc, s7d_clc)
> lanEstim <- dbeCalc(lanEstim, type="CI",probs=c(0.025,0.975),update=TRUE)
> lanEstim <- dbeCalc(lanEstim,type="CV",
+ update=TRUE)
> head(lanEstim@lenNum$cv)

      time space technical length      value
1 2006 - 1   all         1    330 1.27475015
```

```
2 2006 - 1   all          1     350 0.50652970
3 2006 - 1   all          1     360 0.18734961
4 2006 - 1   all          1     370 0.11550346
5 2006 - 1   all          1     380 0.26174092
6 2006 - 1   all          1     390 0.07719868
```

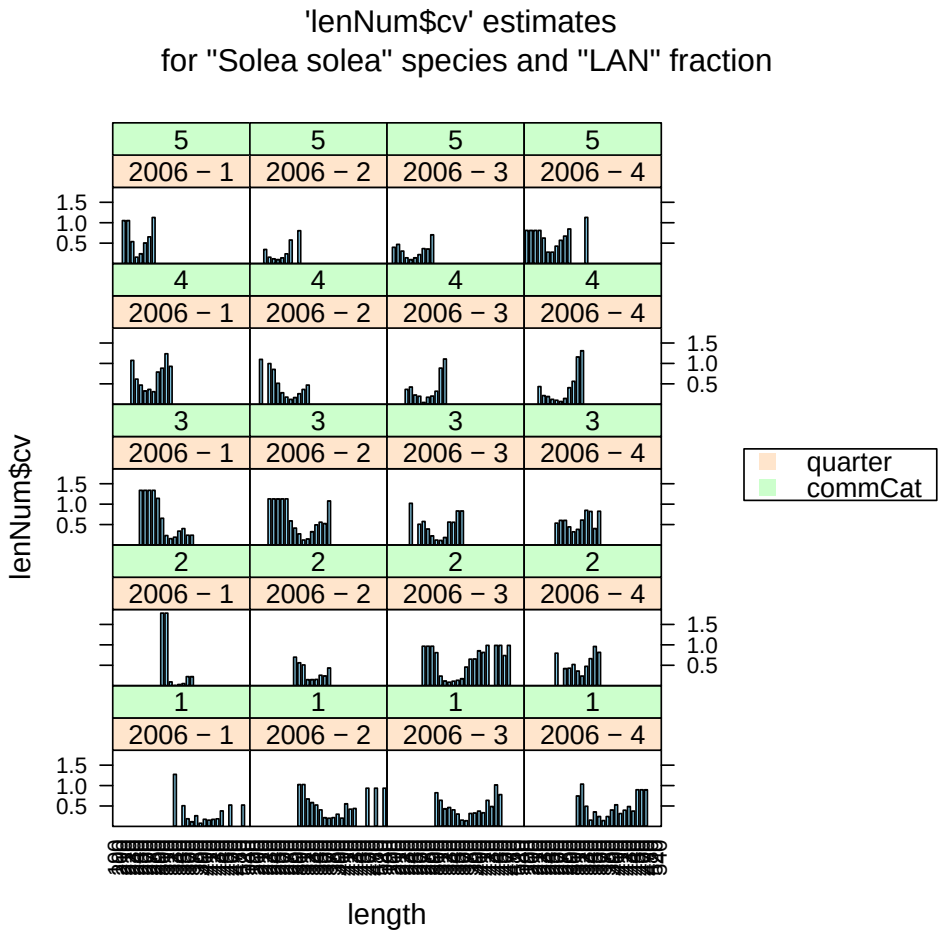
```
> lanEstim@lenNum$DCRcvIndicator
```

```
[1] 0.2396934
```

```
> dbePlot(lanEstim,elmt="lenStruc$estim")
> update(trellis.last.object(),par.strip.text=list(cex=0.6))
```



```
> dbePlot(lanEstim,elmt="lenNum$cv")
> update(trellis.last.object(),par.strip.text=list(cex=0.6))
```



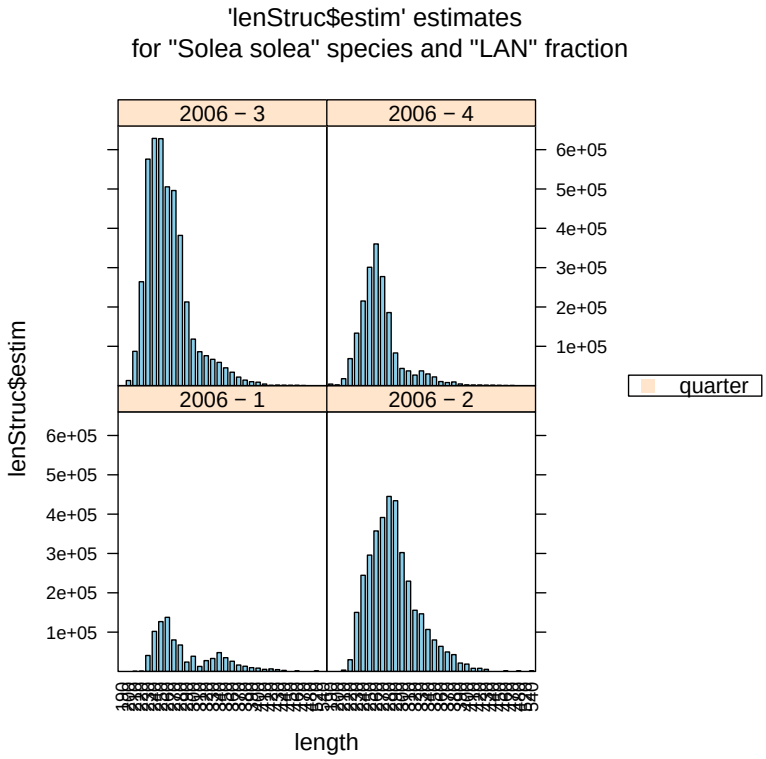
The `RaiseLgth` method is designed to cope with more complex situation like the name of the species (Solea solea) does not match the name of the commercial taxon (Sole). The code to use is then the following

When the commercial sampling dataset contains several species which all together match a commercial taxon, as is the case for e.g. the rays or the anglerfich, the code to use is then the following

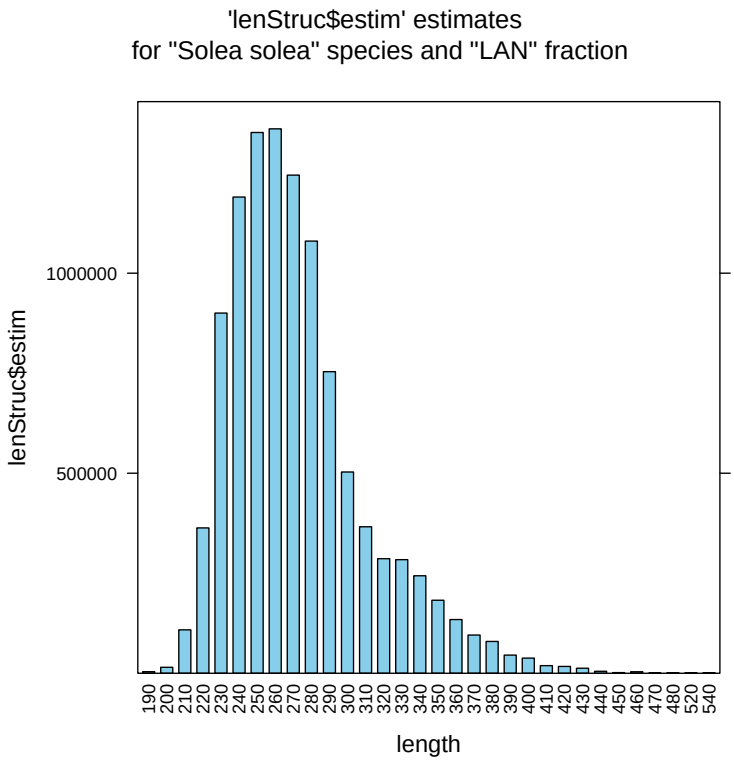
The same occurs when the commercial sampling dataser contains Males and Females as is the case for e.g. megrim or Nephrops, and they are mixed in the commercial landings, the code to use is then the following You have noticed that from one dbe object, you may generate as many objects as there are calls to `RaiseLgth`. The internal code of `RaiseLgth` will proceed to the appropriate sum of sampling weights before subsetting the commercial sampling dataset according to the argument given and raise to the total landings. Eventually, you get a dbe object for each of the sampled species.

The next step of the analysis is to consider the estimates at the desired resolution, ie yearly, quarterly, for the entire area, all gears combined, ... The call to `stratAggreg` will perform the sum over the demanded strata for all the parameters contained in the dbeObject. Beside the name of the dbeObject, the arguments to be passed to the function are the three levels of stratification. The bolean value TRUE means that you demand an aggregation of the parameters over all the strata. It is suggested here to allocate the result of the function to a new dbeObject, in order to be able to keep the original for other types of aggregation if needed.

```
> # Quarterly estimates
> lanEstim_Q <- stratAggreg(lanEstim,timeStrata=FALSE,spaceStrata=TRUE,techStrata=TRUE)
> dbePlot(lanEstim_Q,elmt="lenStruc$estim")
> update(trellis.last.object(),par.strip.text=list(cex=0.6))
```




```
> # Yearly estimates
> lanEstim_Y <- stratAggreg(lanEstim,timeStrata=TRUE,spaceStrata=TRUE,techStrata=TRUE)
> dbePlot(lanEstim_Y,elmt="lenStruc$estim")
> update(trellis.last.object(),par.strip.text=list(cex=0.6))
```

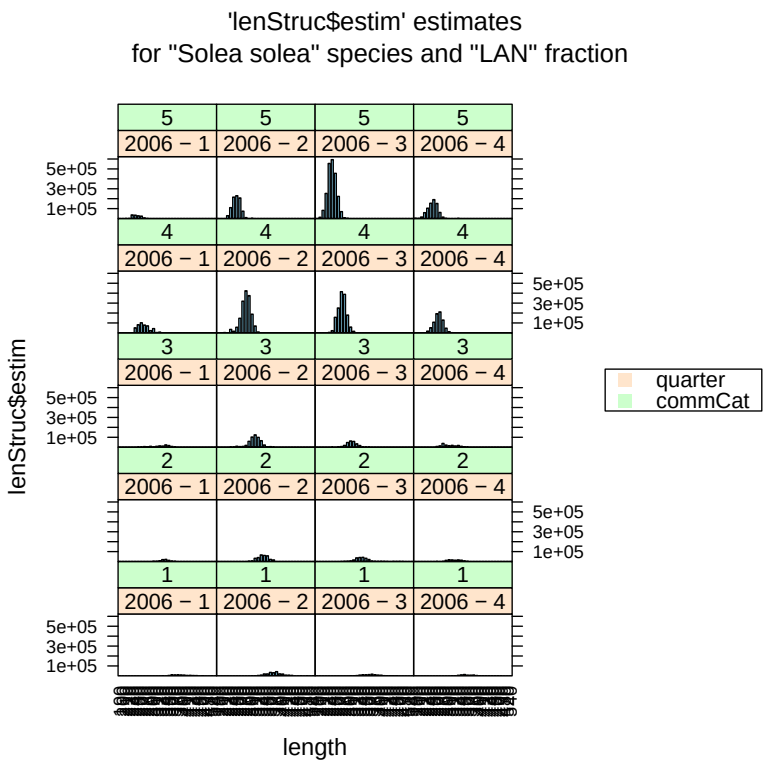


9.2 resampling method

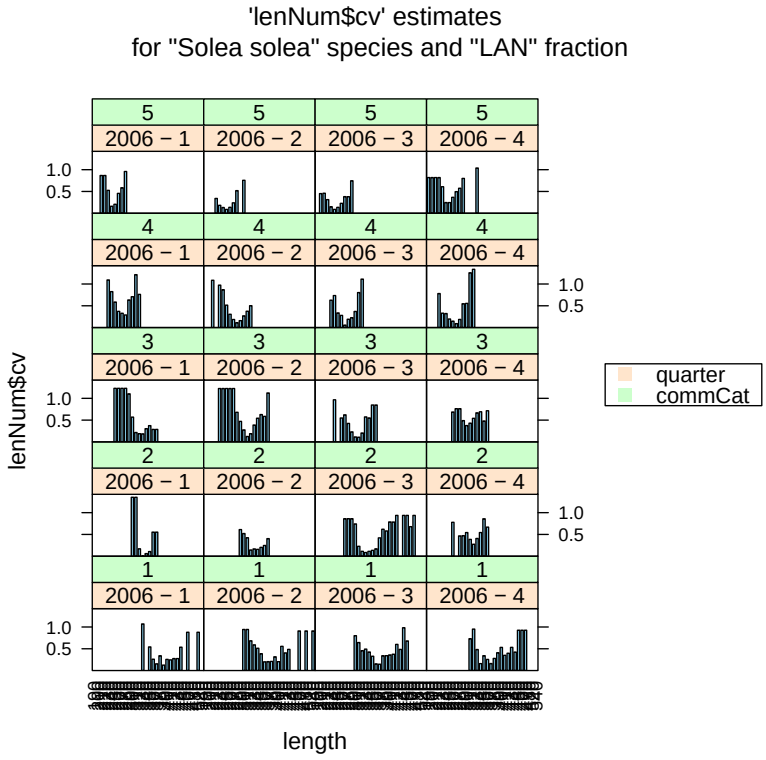
```
> lanEstimB <- dbeObject(desc="Sole VIId Landingss",species="Solea solea",
+ catchCat="LAN",strataDesc=strD, methodDesc="bootstrap")
```

Here we will consider the simpler case, which is one species in the commercial sampling tables (here Solea solea) matches the commercial naming of the taxon in the commercial landings (table cl). In this case, the only arguments to pass to the function are the dbe object, the consolidated cs and cl datasets, and the number of replicates for bootstrap. In the testing phase, consider a 'small' B, since you have to count between 1 and 3 seconds per replicates depending on the size of the dataset and the performance of your machine.

```
> lanEstimB <- RaiseLgthBoot(lanEstimB, s7d_csc, s7d_clc, B=100)

[1] "iter"
[1] 0
[1] 50
[1] 100
[1] 100
[1] "iterations complete"

> lanEstimB <- dbeCalc(lanEstimB, type="CI",probs=c(0.025,0.975),update=TRUE)
> lanEstimB <- dbeCalc(lanEstimB,type="CV",
+ update=TRUE)
> head(lanEstimB@lenNum$cv)

      time space technical length value
1 2006 - 1   all         1    190   NaN
2 2006 - 1   all         1    200   NaN
3 2006 - 1   all         1    210   NaN
4 2006 - 1   all         1    220   NaN
5 2006 - 1   all         1    230   NaN
6 2006 - 1   all         1    240   NaN

> lanEstimB@lenNum$DCRcvIndicator

[1] 0.2572031

> dbePlot(lanEstimB,elmt="lenStruc$estim")
```

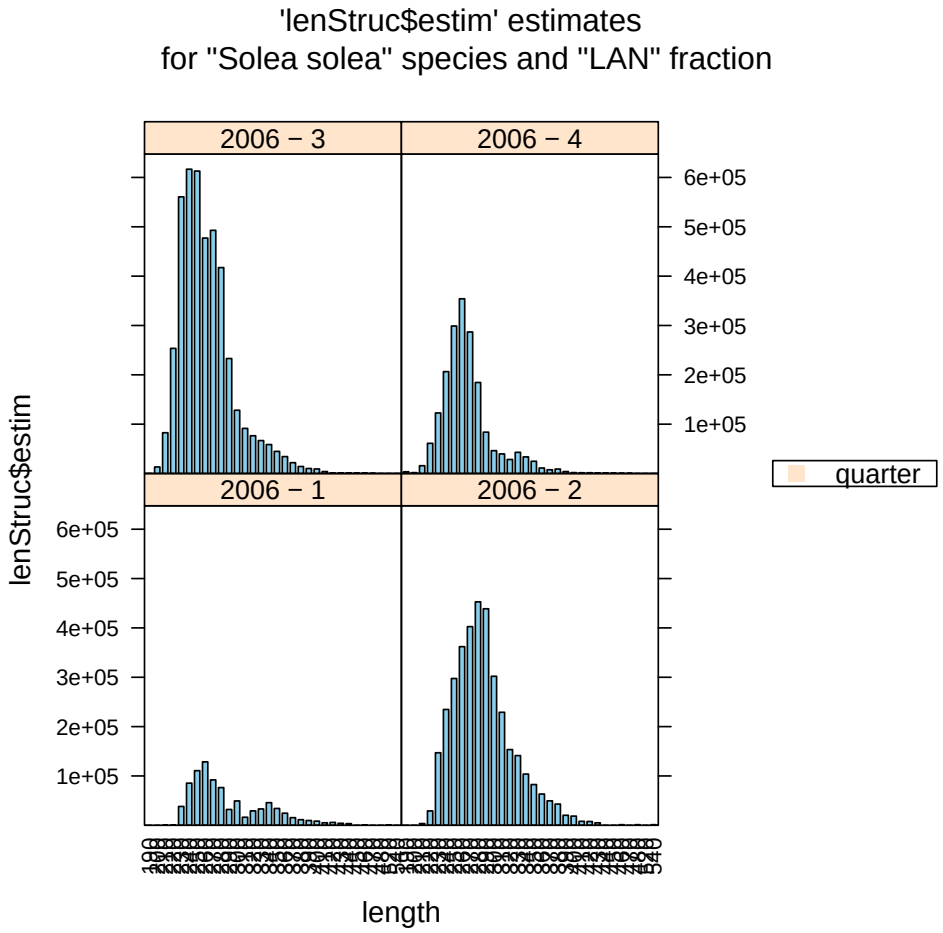


```
> dbePlot(lanEstimB,elmt="lenNum$cv")
```



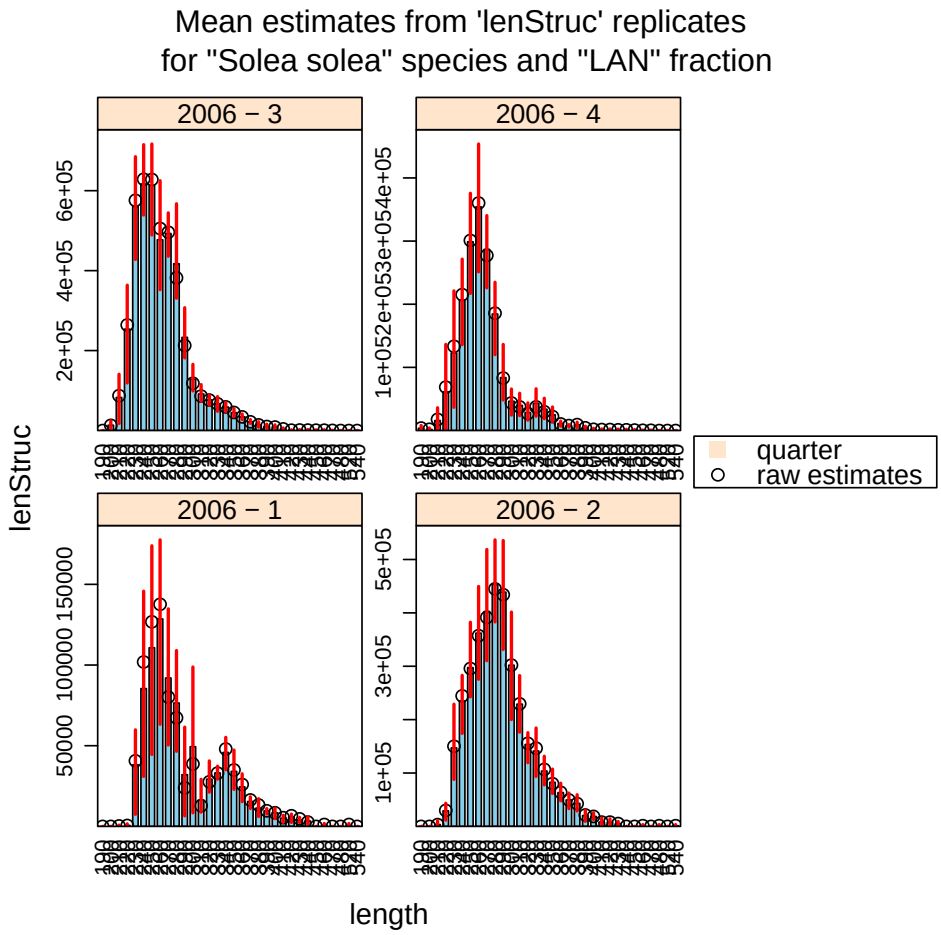
The next step of the analysis is to consider the estimates at the desired resolution, ie yearly, quarterly, for the entire area, all gears combined, ... The call to `stratAggreg` will perform the sum over the demanded strata for all the parameters contained in the dbeObject. Beside the name of the dbeObject, the arguments to be passed to the function are the three levels of stratification. The bolean value TRUE means that you demand an aggregation of the parameters over all the strata. It is suggested here to allocate the result of the function to a new dbeObject, in order to be able to keep the original for other types of aggregation if needed.

```
> # Quarterly estimates
> lanEstimB_Q <- stratAggreg(lanEstimB,timeStrata=FALSE,spaceStrata=TRUE,techStrata=TRUE)
> print(dbePlot(lanEstimB_Q,elmt="lenStruc$estim"))
```



The use of the resampling method will fill the replicate slot in the dbeObject. A function `dbePlotRep` has been developped to show the confidence intervals on the plot estimates.

```
> # Quarterly estimates
> print(dbePlotRep(lanEstimB_Q,Slot="lenStruc"))
```



The figure above reads as follows:

1. The bars are the estimates given by the mean of the replicates
2. The empty dots are the results of the preliminary call to the function `RaiseLgth`
3. The thick lines give the confidence intervals calculated following the argument `probs` passed through the function `dbePlotRep`. By default `probs==c(0.05,0.95)`.

The `RaiseLgth` method is designed to cope with more complex situation like the name of the species (Solea solea) does not match the name of the commercial taxon (Sole). The code to use is then the following
When the commercial sampling dataset contains several species which all together match a commercial taxon, as is the case for e.g. the rays or the anglerfich, the code to use is then the following
The same occurs when the commercial sampling dataser contains Males and Females as is the case for e.g. megrim or Nephrops, and they are mixed in the commercial landings, the code to use is then the following You have noticed that from one dbe object, you may generate as many objects as there are calls to `RaiseLgth`. The internal code of `RaiseLgth` will proceed to the appropriate sum of sampling weights before subsetting the commercial sampling dataset according to the argument given and raise to the total landings. Eventually, you get a dbe object for each of the sampled species.

9.3 Addition of dbeObjects

There is the possibility to add several dbeObjects, with the use `+`. Mainly, there can be two examples of use of the addition: (i) adding dbeObjects from different countries to generate international estimates and (ii) adding estimates of the retained part and the discarded part to estimate the catches. The addition concerns only the slots containing estimates, not the confidence intervals nor the CVs.

10 Age structure from market sampling

10.1 exploratory analysis of the Age Length Key

Before using the Age Length Keys (ALK), you must have a look if all the length classes in the ALK match the estimated length structures of the landings for each of the strata. For this purpose, `viewGapsAlkCons` is very informative and easy to use. In the matrix proposed the length classes with dots are length classes absent from the estimated length structure for that specific stratum, and the length classes with dash lines are length classes present in the estimated length structure but absent from the ALK for that specific stratum. The dash lines are the enemies!

```
> #
> viewGapsAlkCons(s7d_csc)

, , 2006 - 4, all

    1 2  3 4  5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
190 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
200 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
210 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
220 1 0  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
230 1 5  1 2  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
240 3 8  3 7  1 0 0 0 0 0  0  0  0  0  0  0  0  0  0
250 0 11 4 7  2 1 0 0 0 0  0  0  0  0  0  0  0  0  0
260 2 13 1 3  3 0 0 1 0 0  0  0  0  0  0  0  0  0  0
270 0 12 5 0  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
280 0 7  2 2  1 1 0 1 0 0  0  0  0  0  0  0  0  0  0
290 0 2  2 2  1 0 1 0 0 0  0  0  0  0  0  0  0  0  0
300 0 4  1 2  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
310 0 4  4 8  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
320 0 0  3 7  2 2 0 0 0 1  0  0  0  0  0  0  0  0  0
330 0 0  2 11 4 1 2 0 0 0  0  0  0  0  0  0  0  0  0
340 0 1  1 9  0 2 0 0 1 0  0  0  0  0  0  0  0  0  0
350 0 0  1 3  3 1 0 0 0 0  0  0  0  0  0  0  0  0  0
360 0 1  2 6  0 1 0 0 0 0  0  0  0  0  0  0  0  0  0
370 0 0  1 0  0 1 0 0 0 0  0  0  0  0  0  0  0  0  0
380 0 0  0 0  0 2 0 0 0 0  0  0  0  0  0  0  0  0  0
390 0 0  0 0  0 2 1 0 0 0  0  0  0  0  0  0  0  0  0
400 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
410 0 0  0 2  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
420 0 0  0 0  1 0 0 0 0 0  0  0  0  0  0  1  0  0  0
430 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
440 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
450 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
460 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
470 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
480 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
490 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
500 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
510 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
520 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
530 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
540 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
```

When dash lines are encountered, the COST methods propose ways of filling the gaps which are

- `type = "stepIncr"` : Length class step is increased to specified value parameter (default value=10)
- `type ="fillMiss"` : All gaps (with size <= value) are filled out with the sum of surrounding recorded classes (default value=1)
- `type = "sExtrGrp"` : The 'value' first classes are grouped (default value=1)
- `type = "lExtrGrp"` : The 'value' last classes are grouped (default value=1)

- `type = "sFillMiss"` : The 'value' empty classe(s) prior to first recorded length class is filled out with the latter (default value=1)
- `type = "lFillMiss"` : The 'value' empty classe(s) following last recorded length class is filled out with the latter (default value=1)
- `type = "sFillAge"` : The 'value' empty classe(s) prior to first recorded length class is filled out with 1 individual of minimal age (default value=1)

Here is a go at filling the gaps we have, playing with the arguments type and value : For simplicity, we will look only at the 4th quarter output. Information is missing in the 2 first lines (length classes 190, 200), then on 1 line in the middle (length class 400), and finally 5 lines on the upper part pf the ALK (length classes 430 to 470) which are not on the border of the matrix. Note that you can make as many tries as you want with argument update=FALSE; once you are sure of what you are doing, you have to change the update argument to TRUE, in order to update the ca tables with new records. First, let's deal with the 2 first missing length classes.

```
> s7d_csc <- alkLgthRec(s7d_csc,type="sFillMiss",value=2,
+ preview=FALSE, postview=TRUE, update=TRUE)

, , 2006 - 4, all

    1 2  3 4  5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
190 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
200 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
210 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
220 1 0  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
230 1 5  1 2  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
240 3 8  3 7  1 0 0 0 0 0  0  0  0  0  0  0  0  0  0
250 0 11 4 7  2 1 0 0 0 0  0  0  0  0  0  0  0  0  0
260 2 13 1 3  3 0 0 1 0 0  0  0  0  0  0  0  0  0  0
270 0 12 5 0  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
280 0 7  2 2  1 1 0 1 0 0  0  0  0  0  0  0  0  0  0
290 0 2  2 2  1 0 1 0 0 0  0  0  0  0  0  0  0  0  0
300 0 4  1 2  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
310 0 4  4 8  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
320 0 0  3 7  2 2 0 0 0 1  0  0  0  0  0  0  0  0  0
330 0 0  2 11 4 1 2 0 0 0  0  0  0  0  0  0  0  0  0
340 0 1  1 9  0 2 0 0 1 0  0  0  0  0  0  0  0  0  0
350 0 0  1 3  3 1 0 0 0 0  0  0  0  0  0  0  0  0  0
360 0 1  2 6  0 1 0 0 0 0  0  0  0  0  0  0  0  0  0
370 0 0  1 0  0 1 0 0 0 0  0  0  0  0  0  0  0  0  0
380 0 0  0 0  0 2 0 0 0 0  0  0  0  0  0  0  0  0  0
390 0 0  0 0  0 2 1 0 0 0  0  0  0  0  0  0  0  0  0
400 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
410 0 0  0 2  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
420 0 0  0 0  1 0 0 0 0 0  0  0  0  0  0  1  0  0  0
430 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
440 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
450 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
460 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
470 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
480 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
490 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
500 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
510 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
520 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
530 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
540 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
```

Now the length clases 190 and 200 have been filled with the value of the first length class containing information, i.e. length class 210. We are going now to fill the length class 400 by summing the 2 surrounding length classes:

```
> s7d_csc <- alkLgthRec(s7d_csc,type="fillMiss",value=1,
+ preview=FALSE, postview=TRUE, update=TRUE)

, , 2006 - 4, all

    1 2  3 4  5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
190 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
200 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
210 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
220 1 0  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
230 1 5  1 2  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
240 3 8  3 7  1 0 0 0 0 0  0  0  0  0  0  0  0  0  0
250 0 11 4 7  2 1 0 0 0 0  0  0  0  0  0  0  0  0  0
260 2 13 1 3  3 0 0 1 0 0  0  0  0  0  0  0  0  0  0
270 0 12 5 0  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
280 0 7  2 2  1 1 0 1 0 0  0  0  0  0  0  0  0  0  0
290 0 2  2 2  1 0 1 0 0 0  0  0  0  0  0  0  0  0  0
300 0 4  1 2  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
310 0 4  4 8  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
320 0 0  3 7  2 2 0 0 0 1  0  0  0  0  0  0  0  0  0
330 0 0  2 11 4 1 2 0 0 0  0  0  0  0  0  0  0  0  0
340 0 1  1 9  0 2 0 0 1 0  0  0  0  0  0  0  0  0  0
350 0 0  1 3  3 1 0 0 0 0  0  0  0  0  0  0  0  0  0
360 0 1  2 6  0 1 0 0 0 0  0  0  0  0  0  0  0  0  0
370 0 0  1 0  0 1 0 0 0 0  0  0  0  0  0  0  0  0  0
380 0 0  0 0  0 2 0 0 0 0  0  0  0  0  0  0  0  0  0
390 0 0  0 0  0 2 1 0 0 0  0  0  0  0  0  0  0  0  0
400 0 0  0 2  0 2 1 0 0 0  0  0  0  0  0  0  0  0  0
410 0 0  0 2  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
420 0 0  0 0  1 0 0 0 0 0  0  0  0  0  0  1  0  0  0
430 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
440 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
450 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
460 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
470 - -  - -  - - - - - -  -  -  -  -  -  -  -  -  -
480 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
490 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
500 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
510 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
520 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
530 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
540 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
```

This is easy to check that the length class 400 corresponds now to the sum of length class 390 and length class 410. The difficult part is to fill the length classes 430 to 470. There are 2 alternatives, one is creating a plus length group at length 420, i.e. all the length classes above are summed in the hl table and you will need to rebuild a new dbe object. This is done using the argument type="lExtrGrp". The second solution that we show here is to fill the last 5 lines with the information contained in the last length class (420):

```
> s7d_csc <- alkLgthRec(s7d_csc,type="lFillMiss",value=5,
+ preview=FALSE, postview=TRUE, update=TRUE)

, , 2006 - 4, all

    1 2  3 4  5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
190 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
200 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
210 0 3  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
220 1 0  0 1  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
230 1 5  1 2  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
240 3 8  3 7  1 0 0 0 0 0  0  0  0  0  0  0  0  0  0
250 0 11 4 7  2 1 0 0 0 0  0  0  0  0  0  0  0  0  0
```

```
260 2 13 1 3  3 0 0 1 0 0  0  0  0  0  0  0  0  0  0
270 0 12 5 0  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
280 0 7  2 2  1 1 0 1 0 0  0  0  0  0  0  0  0  0  0
290 0 2  2 2  1 0 1 0 0 0  0  0  0  0  0  0  0  0  0
300 0 4  1 2  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
310 0 4  4 8  1 1 0 0 0 0  0  0  0  0  0  0  0  0  0
320 0 0  3 7  2 2 0 0 0 1  0  0  0  0  0  0  0  0  0
330 0 0  2 11 4 1 2 0 0 0  0  0  0  0  0  0  0  0  0
340 0 1  1 9  0 2 0 0 1 0  0  0  0  0  0  0  0  0  0
350 0 0  1 3  3 1 0 0 0 0  0  0  0  0  0  0  0  0  0
360 0 1  2 6  0 1 0 0 0 0  0  0  0  0  0  0  0  0  0
370 0 0  1 0  0 1 0 0 0 0  0  0  0  0  0  0  0  0  0
380 0 0  0 0  0 2 0 0 0 0  0  0  0  0  0  0  0  0  0
390 0 0  0 0  0 2 1 0 0 0  0  0  0  0  0  0  0  0  0
400 0 0  0 2  0 2 1 0 0 0  0  0  0  0  0  0  0  0  0
410 0 0  0 2  0 0 0 0 0 0  0  0  0  0  0  0  0  0  0
420 0 0  0 0  1 0 0 0 0 0  0  0  0  0  0  1  0  0  0
430 0 0  0 0  1 0 0 0 0 0  0  0  0  0  0  1  0  0  0
440 0 0  0 0  1 0 0 0 0 0  0  0  0  0  0  1  0  0  0
450 0 0  0 0  1 0 0 0 0 0  0  0  0  0  0  1  0  0  0
460 0 0  0 0  1 0 0 0 0 0  0  0  0  0  0  1  0  0  0
470 0 0  0 0  1 0 0 0 0 0  0  0  0  0  0  1  0  0  0
480 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
490 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
500 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
510 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
520 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
530 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
540 . .  . .  . . . . . .  .  .  .  .  .  .  .  .  .
```

Once we are happy with the result, we can estimate the age structure. The danger with using ALK with length classes missing, is that the estimated total numberage will be inferior to the estimated total numberlength. This discrepancy must be avoided in any case.

10.2 Estimating age structure - Analytical

Contrarily to `RaiseLgth`, there are no special cases for `RaiseAge` as it consists 'only' in multiplying the ALK with the length structure previously estimated. The argument type refers to the way the ALK has been created. If the protocol was to collect n otoliths per length class, the type is said "fixed". If the protocol was to collect one otolith every n fish measured, the type is said "proportional". In the latter case, it is preferable to raise directly the numberage without going through an ALK (see following section). For filling the age slots, we use the same dbeObject as the one used for raising length (section 7.1).

```
> lanEstim <- RaiseAge(lanEstim, s7d_csc, type="fixed")
> lanEstim <- dbeCalc(lanEstim,type="CV",
+ vrbl="a", update=TRUE)
> head(lanEstim@ageNum$cv)
```

```
      time space technical age     value
1 2006 - 1   all         1   4 1.2324282
2 2006 - 1   all         1   5 0.3797815
3 2006 - 1   all         1   6 0.3029222
4 2006 - 1   all         1   7 0.2763075
5 2006 - 1   all         1   8 0.1709542
6 2006 - 1   all         1   9 0.3300700
```

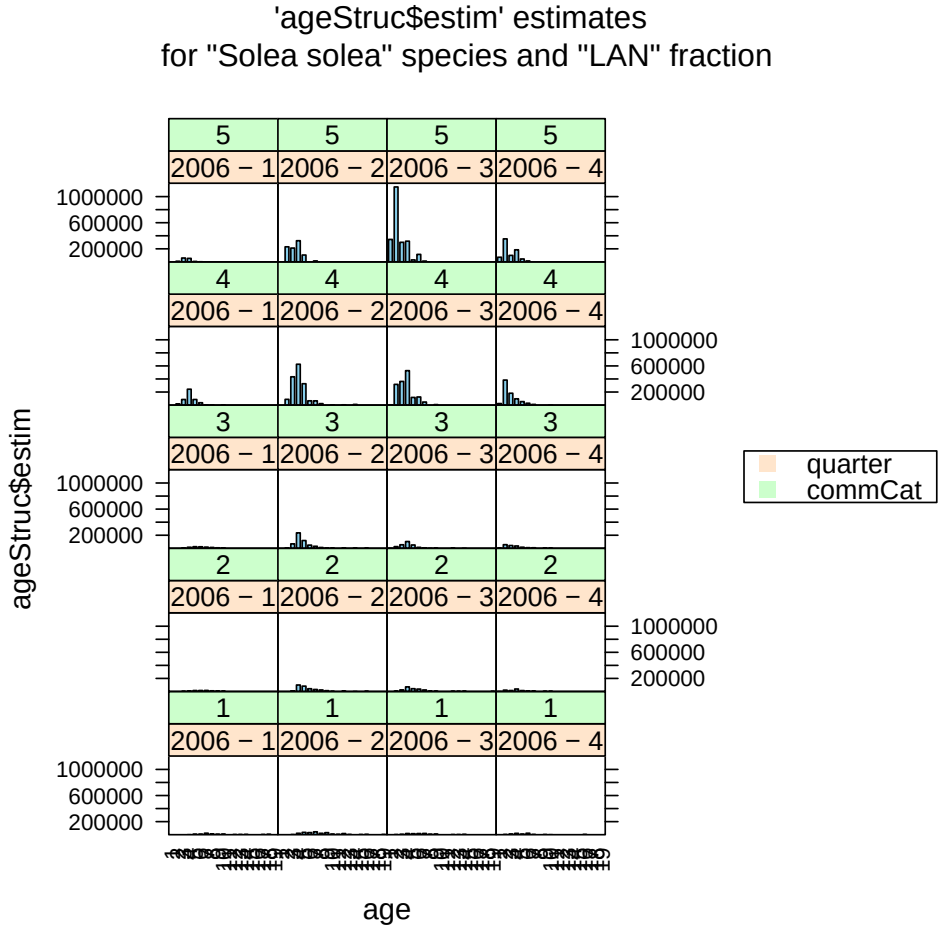
```
> lanEstim@ageNum$DCRcvIndicator
```

```
[1] 0.2277553
```

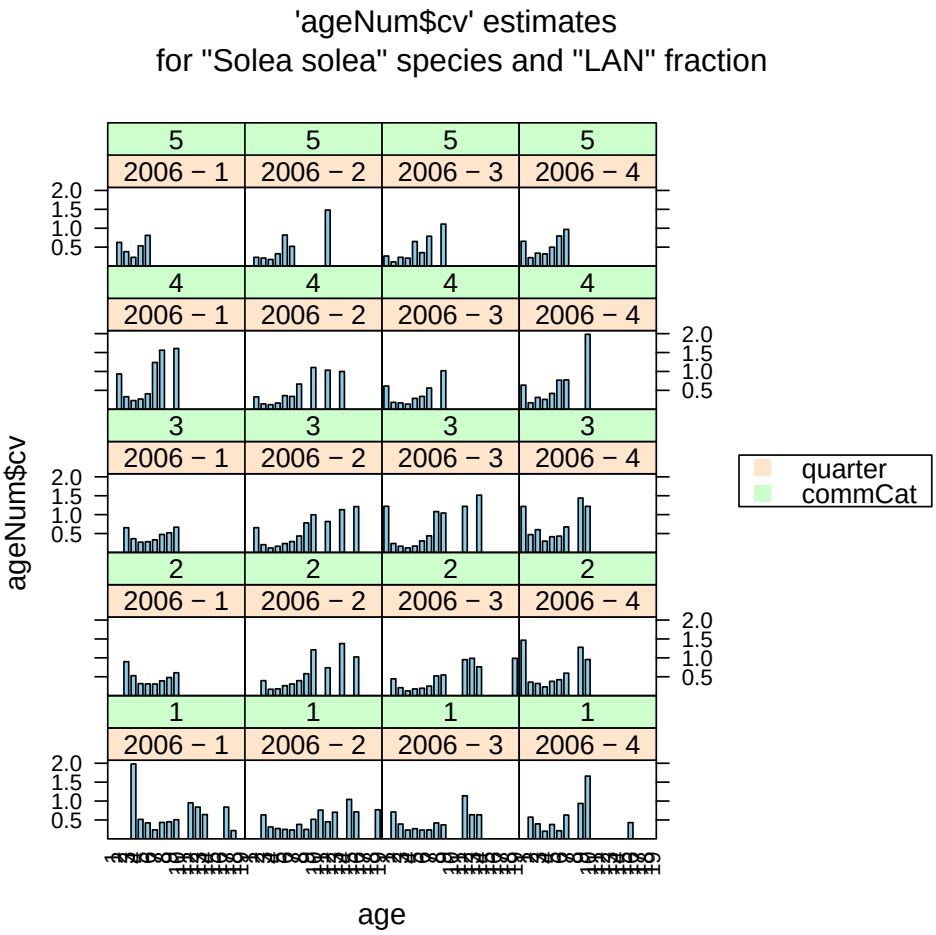
```
> lanEstim <- dbeCalc(lanEstim,type="CI",
+ vrbl="a", probs=c(0.025,0.975),update=TRUE)
> head(lanEstim@ageNum$ci)
```

```
      time space technical age       value       inf        sup
1 2006 - 1   all         1   4    97.37879     0.000   332.5987
2 2006 - 1   all         1   5  5672.46684  1450.121  9894.8129
3 2006 - 1   all         1   6  9884.29727  4015.826 15752.7684
4 2006 - 1   all         1   7 11168.44178  5120.141 17216.7426
5 2006 - 1   all         1   8 21547.90468 14327.978 28767.8317
6 2006 - 1   all         1   9  8260.66720  2916.633 13604.7012
```

```
> dbePlot(lanEstim,elmt="ageStruc$estim")
> update(trellis.last.object(),par.strip.text=list(cex=0.6))
```



```
> dbePlot(lanEstim,elmt="ageNum$cv")
> update(trellis.last.object(),par.strip.text=list(cex=0.6))
```

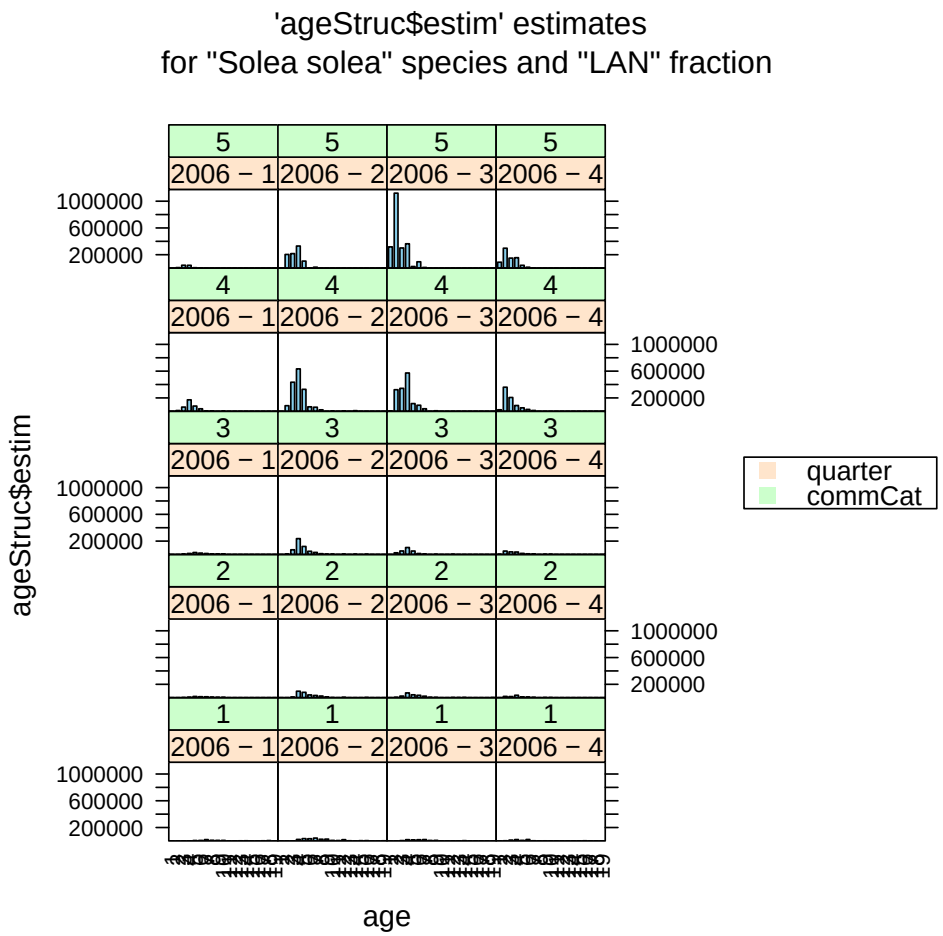


10.3 Estimating age structure - Bootstrap

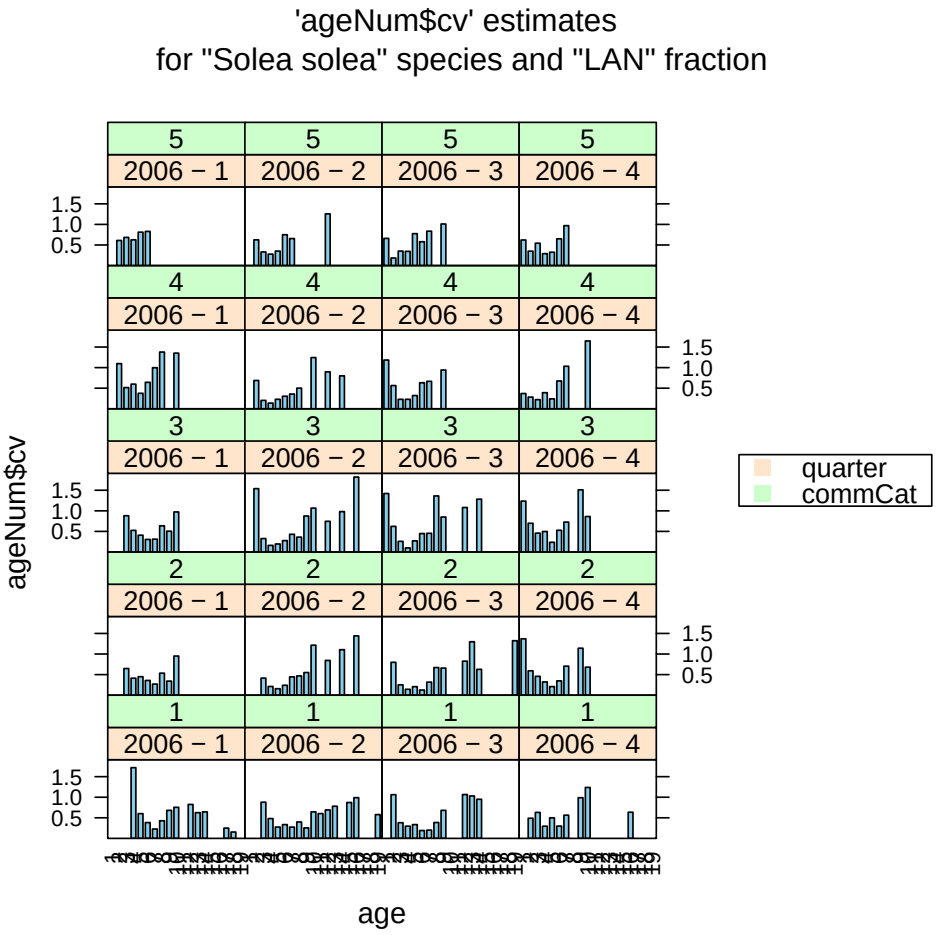
The method used needs the same argument as `RaiseLgth`, plus the number of replicates for the bootstrap. In the testing phase, it is advisable to use a small number of replicates. For filling the age slots, we use the same dbeObject as the one used for raising length (section 7.2).

```
> lanEstimB <- RaiseAgeBoot(lanEstimB, s7d_csc, type="fixed", B=100)

[1] "dbeOutput object methodDesc slot has been changed from bootstrap to bootstrap samples"
[1] "iter"
[1] 0
[1] 50
[1] 100
[1] 100
[1] "iterations complete"


> lanEstimB <- dbeCalc(lanEstimB,type="CV",
+ vrbl="a", update=TRUE)
> head(lanEstimB@ageNum$cv)

      time space technical age     value
1 2006 - 1   all         1   1       NaN
2 2006 - 1   all         1   2       NaN
3 2006 - 1   all         1   3       NaN
4 2006 - 1   all         1   4 1.6616153
5 2006 - 1   all         1   5 0.5368487
6 2006 - 1   all         1   6 0.5202834
```

```
> lanEstimB@ageNum$DCRcvIndicator

[1] 0.3333867

> lanEstimB <- dbeCalc(lanEstimB,type="CI",
+ vrbl="a", probs=c(0.025,0.975),update=TRUE)
> head(lanEstim@ageNum$ci)

      time space technical age       value       inf        sup
1 2006 - 1   all         1   4    97.37879     0.000   332.5987
2 2006 - 1   all         1   5  5672.46684  1450.121  9894.8129
3 2006 - 1   all         1   6  9884.29727  4015.826 15752.7684
4 2006 - 1   all         1   7 11168.44178  5120.141 17216.7426
5 2006 - 1   all         1   8 21547.90468 14327.978 28767.8317
6 2006 - 1   all         1   9  8260.66720  2916.633 13604.7012

> lanEstimB,elmt="ageStruc$estim"
> update(trellis.last.object(),par.strip.text=list(cex=0.6))
```



```
> dbePlot(lanEstimB,elmt="ageNum$cv")
> update(trellis.last.object(),par.strip.text=list(cex=0.6))
```



11 Age structure without ALK

A dedicated method has been implemented `N_at_Age_direct` for the cases where the estimation of age is done without the need to use an ALK. This is the case where the collection of otolith is based on a random sampling of the population, or where age reading may be done visually on the individual (scallops). The arguments to be passed to the method are an empty dbeOutput object, a csData set and a clData set. The method estimates the age structure by strata and the related variances at age.

```
> str21 <- strIni(timeStrata=c("quarter"),techStrata=c("commCat"))
> cs.cons <- csDataCons(cs.vali, str21)
> cl.cons <- clDataCons(cl.vali, str21)
> pleLan<-dbeObject(desc="Plaice IV",species="Pleuronectes platessa",catchCat="LAN",
+ strataDesc=str21, methodDesc="analytical")
> pleLan <- N_at_Age_direct(pleLan,cs.cons,cl.cons)
> head(pleLan@ageStruc$estim)

  time space technical age  value
1    1   all         1   1      0
2    1   all         1   2      0
3    1   all         1   3      0
4    1   all         1   4  28159
5    1   all         1   5 140795
6    1   all         1   6 238072

> head(pleLan@ageVar)

  time space technical age     value
1    1   all         1   1         0
```

```
2    1   all         1   2         0
3    1   all         1   3         0
4    1   all         1   4  70191835
5    1   all         1   5 313101123
6    1   all         1   6 474140262
```

As for the age structure with ALK (section 8), all methods working on a dbeOutput object are available. it is therefore possible to calculate the CVs, the confidence intervals and the DCR precision estimates, and plot the results.

```
> pleLan <- dbeCalc(pleLan,type="CV",
+ vrbl="a", probs=c(0.025,0.975),update=TRUE)
> head(pleLan@ageNum$cv)
```

```
  time space technical age      value
1    1   all         1   1        NaN
2    1   all         1   2        NaN
3    1   all         1   3        NaN
4    1   all         1   4 0.29752679
5    1   all         1   5 0.12567679
6    1   all         1   6 0.09146293
```

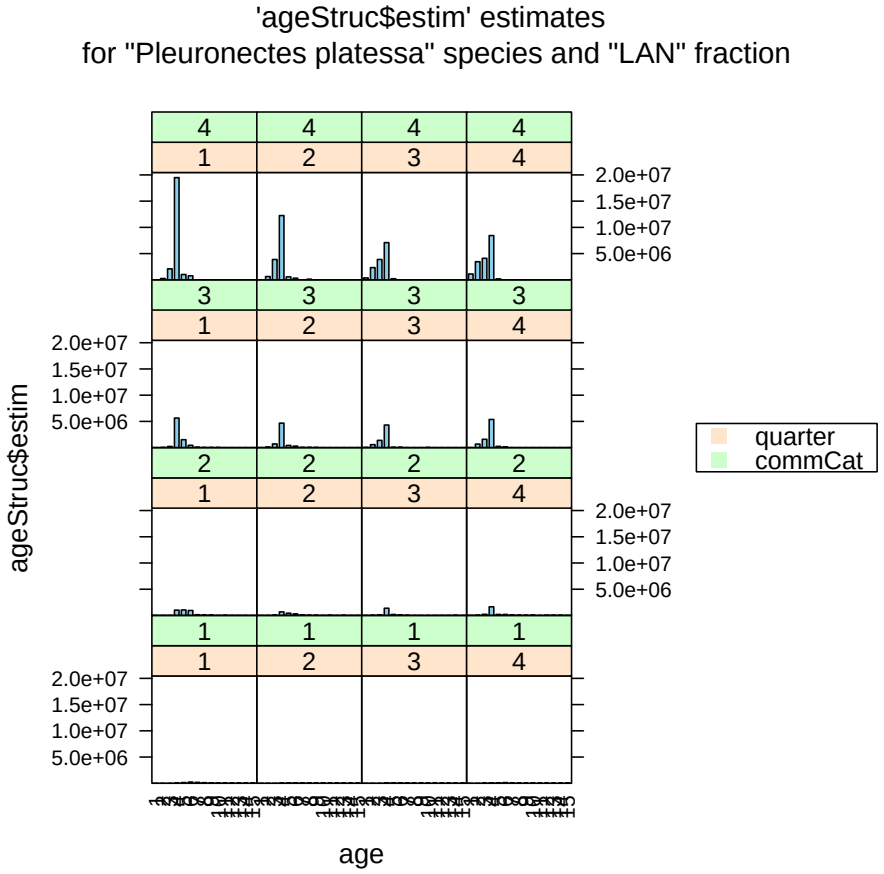
```
> pleLan@ageNum$DCRcvIndicator
```

```
[1] 0.08845803
```

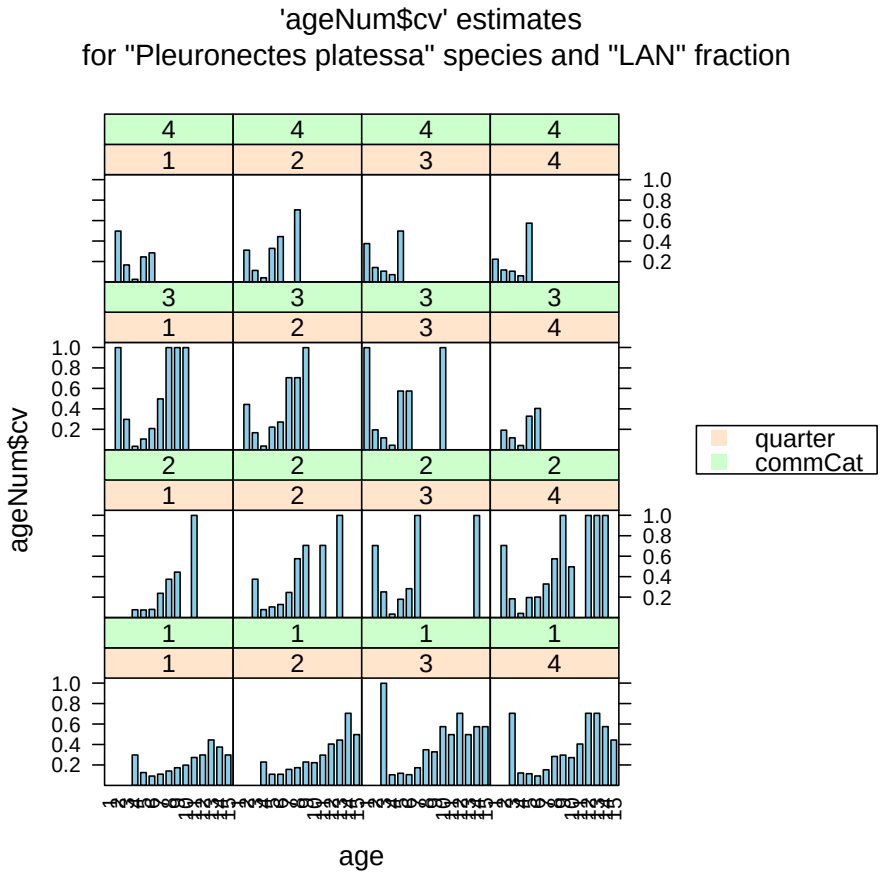
```
> pleLan <- dbeCalc(pleLan,type="CI",
+ vrbl="a", update=TRUE)
> head(pleLan@ageNum$ci)
```

```
  time space technical age  value       inf       sup
1    1   all         1   1      0      0.00      0.00
2    1   all         1   2      0      0.00      0.00
3    1   all         1   3      0      0.00      0.00
4    1   all         1   4  28159  11738.31  44579.69
5    1   all         1   5 140795 106114.10 175475.90
6    1   all         1   6 238072 195394.25 280749.75
```

```
> dbePlot(pleLan,elmt="ageStruc$estim")
```




```
> dbePlot(pleLan,elmt="ageNum$cv")
```



12 Estimating biological parameters

A specific consolidated object is built for biological parameters estimates.

```
> strBP <- strIni(timeStrata="quarter")
> s7dbp_csc <- csDataCons(s7d_csv,strBP)
```

This section describes how to calculate the biological parameters estimates based on an analytical or a bootstrap method. As a user, you will specify which method you want to run, by specifying `"analytical"` or `"bootstrap"` in the argument `methodDesc` at the creation of the `dbeOutput` object, and running respectively `bpEstim` and `bpBoot`. Following this, all the calls to calculation or plotting methods will be exactly the same whatever the method used. There are some supplementary possibilities when using the bootstrap method, like defining the number of replicates or examine the distribution of the replicates. In this manual, we are not going to replicate all the figures, which are similar, for analytical and bootstrap but we will do the calculation analytically for individual weight and sex-ratio and the bootstrap for maturity.

12.1 Individual weight

A new 'dbeOutput' object is created with individual 'weight' as a parameter.

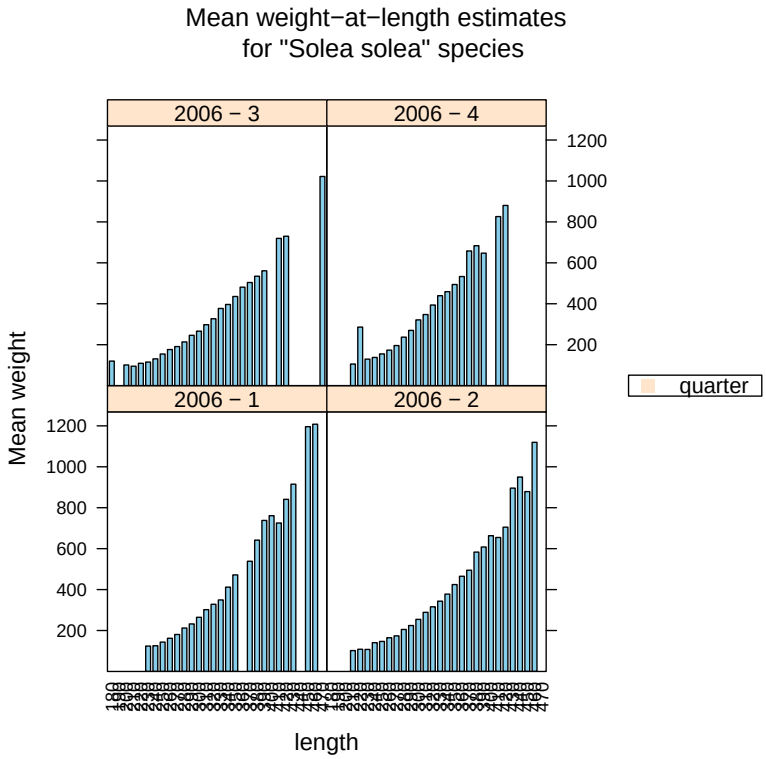
```
> wtEstim_An <- dbeObject(desc="Sole 2006 Individual Weights",species="Solea solea",
+ catchCat="LAN",param="weight",strataDesc=strBP,methodDesc="analytical")
```

Analytical estimates of mean weight-at-length and age, and associated variances at-length and at-age can be computed with `bpEstim` method.

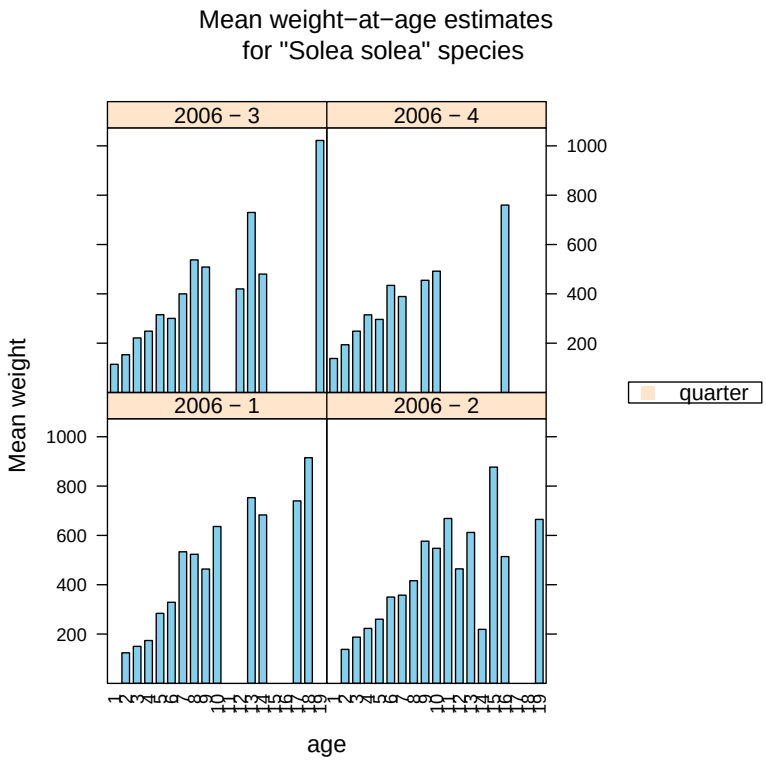
```
> wtEstim_An <- bpEstim(wtEstim_An,s7dbp_csc,adjust=TRUE)
```

If 'adjust=FALSE', length distribution recorded in 'ca' table is supposed to be representative of the catch. Hence, all calculations are made within CA. If 'adjust=TRUE', previous hypothesis is not valid anymore, and estimates-at-age are calculated by injecting 'hl' information (length sampling data).
The function `dbePlot` is designed to plot the main dataframes contained in the `dbeOutput` object. It is then possible to plot the length structure (lenStruc$estim dataframe), the age structure (ageStruc$estim dataframe) but also the CVs (lenNum$cv and ageNum$cv dataframes), the confidence intervals (lenNum$ci and ageNum$ci dataframes). As the same object is used for containing all the dbe output, by lenStruc, ageStruc, lenNum, ageNum, we understand mean weight-at-length and mean weight-at-age here.

```
> #plot of mean weight-at-length data
>
> dbePlot(wtEstim_An,elmt="lenStruc$estim",step=10,ylab="Mean weight",main="Mean weight-at-length
+ estimates \n for \"Solea solea\" species")
```

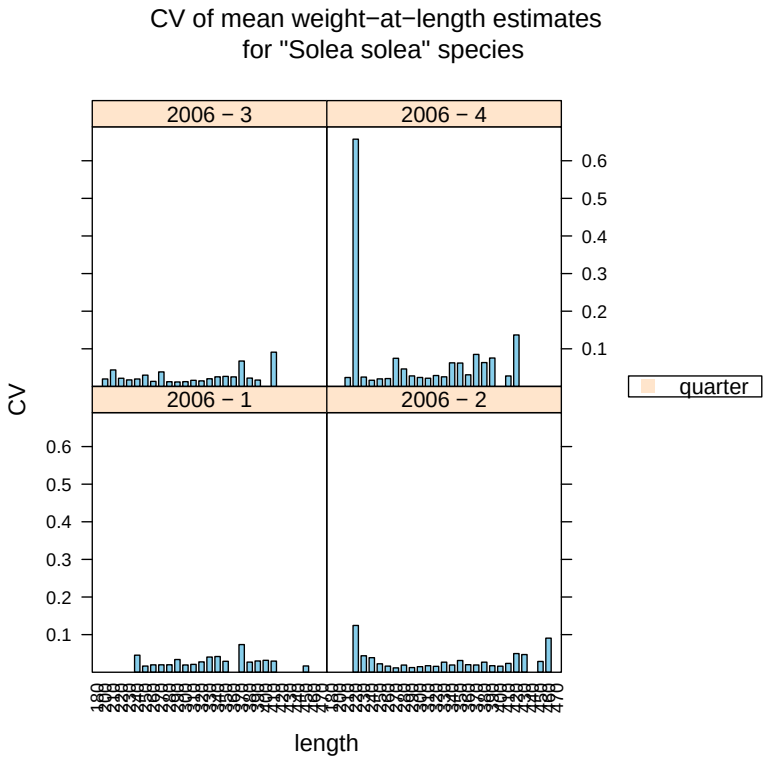



```
> #plot of mean weight-at-age data
>
> dbePlot(wtEstim_An,elmt="ageStruc$estim",ylab="Mean weight",main="Mean weight-at-age
+ estimates \n for \"Solea solea\" species")
```



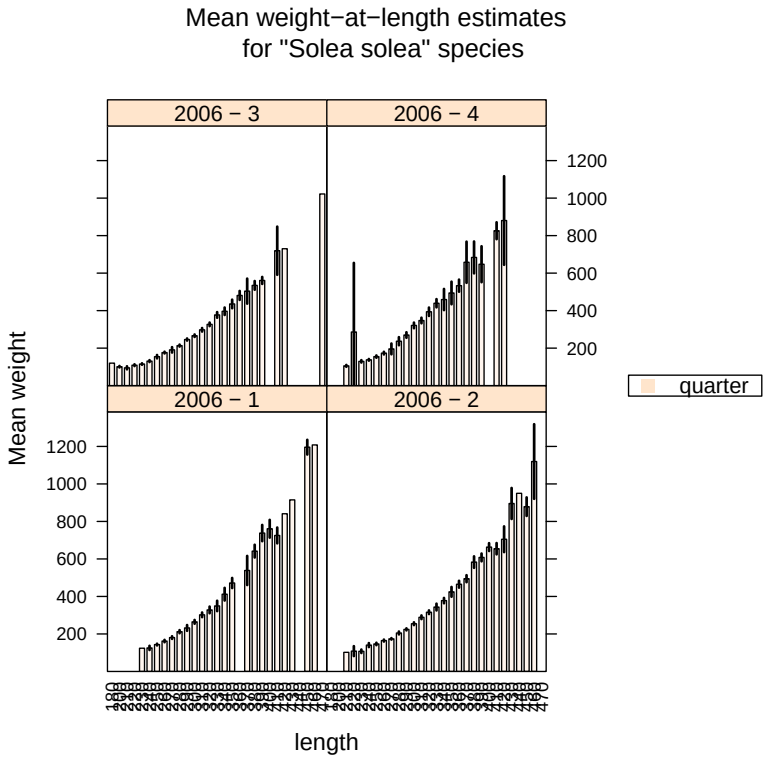
A DCR precision indicator will be calculated when the outcome of the ICES WKPRECISE will be known (October 2008). It is anticipated in the structure of the 'dbeOutput' object that such an information is needed and will be provided. This DCR precision estimates will not necessitate a new method, but will be an update of the current CV calculation method. For the moment, what is computed is the weighted mean over all strata and length/age classes.

```
> #cv at length
> wtEstim_An <- dbeCalc(wtEstim_An,type="CV",vrbl="l",update=TRUE)
> #DCR indicator
> wtEstim_An@lenNum$DCRcvIndicator

[1] 0.03557204

> #plot of cv at length
> dbePlot(wtEstim_An,elmt="lenNum$cv",ylab="CV",main="CV of mean weight-at-length
+ estimates \n for \"Solea solea\" species")
```



The confidence intervals have been calculated based on the normality assumption and the student distribution. With the aim of proposing realistic results, COST has replaced the negative values sometimes occuring for lower bounds by 0.

```
> #ci at length
> wtEstim_An <- dbeCalc(wtEstim_An,type="CI",vrbl="l",update=TRUE)
> dbePlot(wtEstim_An,elmt="lenNum$ci",col="seashell",l.col="black",l.lwd=2,ylab="Mean weight",
+ main="Mean weight-at-length estimates \n for \"Solea solea\" species")
```



12.2 Maturity stages

Another `dbeOutput` object is created with maturity as main parameter.

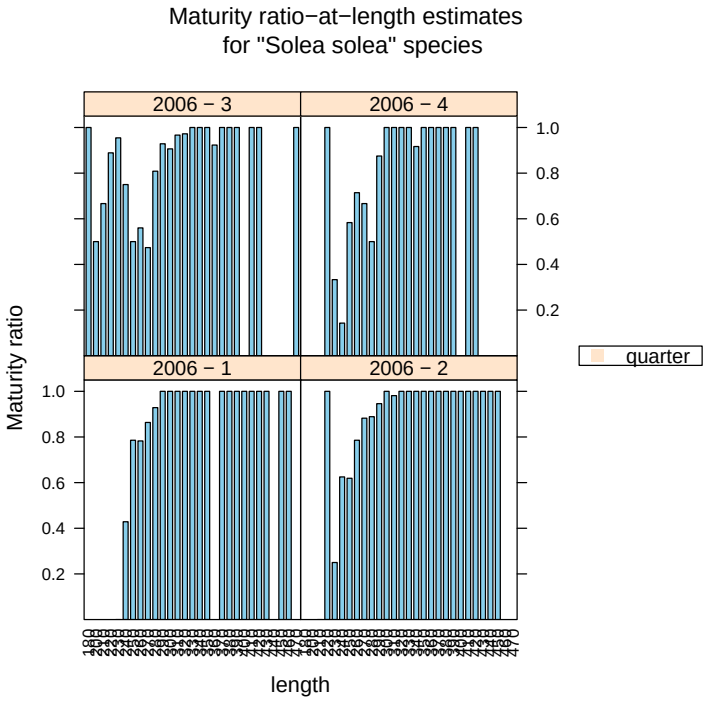
```
> matEstim_Boot <- dbeObject(desc="Sole 2006 Maturity",species="Solea solea",catchCat="LAN",
+ param="maturity",strataDesc=strBP, nboot=10, methodDesc="bootstrap")
```

Analytical estimates of maturity ratio-at-length and age, and associated variances at-length and at-age are computed.

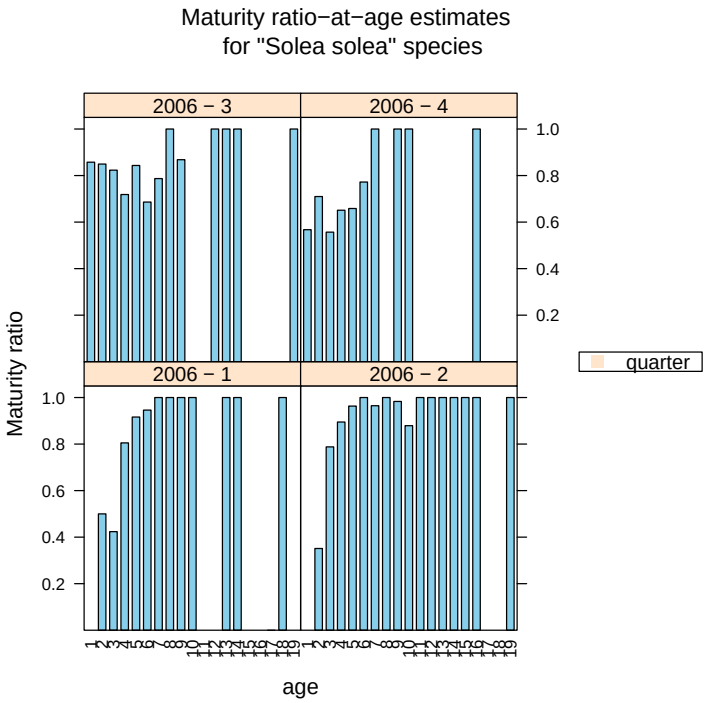
```
> matEstim_Boot <- bpBoot(matEstim_Boot,s7dbp_csc,adjust=TRUE)
```

The function `dbePlot` is designed to plot the main dataframes contained in the `dbeOutput` object. It is then possible to plot the length structure (lenStruc$estim dataframe), the age structure (ageStruc$estim dataframe) but also the CVs (lenNum$cv and ageNum$cv dataframes), the confidence intervals (lenNum$ci and ageNum$ci dataframes). As the same object is used for containing all the dbe output, by lenStruc, ageStruc, lenNum, ageNum, we understand maturity-at-length and maturity-at-age here.

```
> #plot of maturity ratio-at-length data
>
> dbePlot(matEstim_Boot,elmt="lenStruc$estim",ylab="Maturity ratio",main="Maturity ratio-at-length
+ estimates \n for \"Solea solea\" species")
```

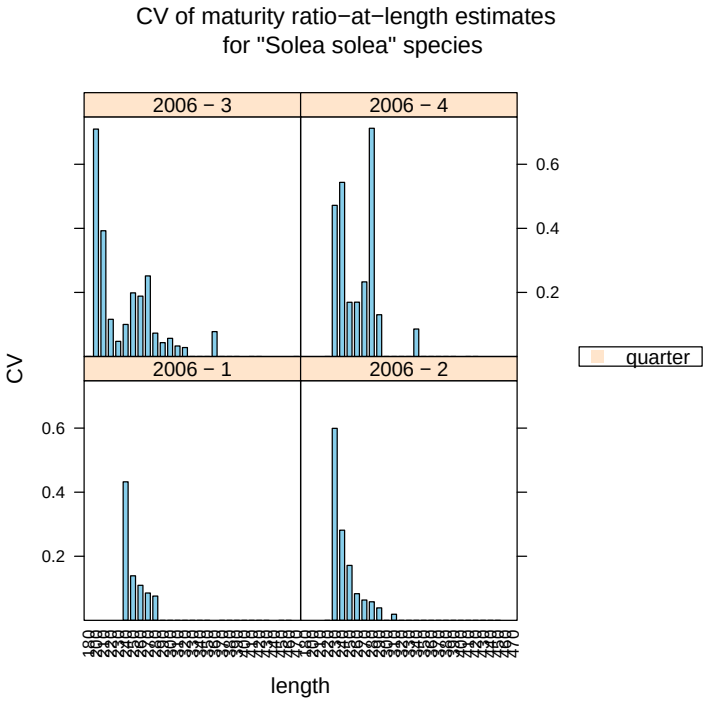



```
> #plot of maturity ratio-at-age data
>
> dbePlot(matEstim_Boot,elmt="ageStruc$estim",ylab="Maturity ratio",main="Maturity ratio-at-age
+ estimates \n for \"Solea solea\" species")
```



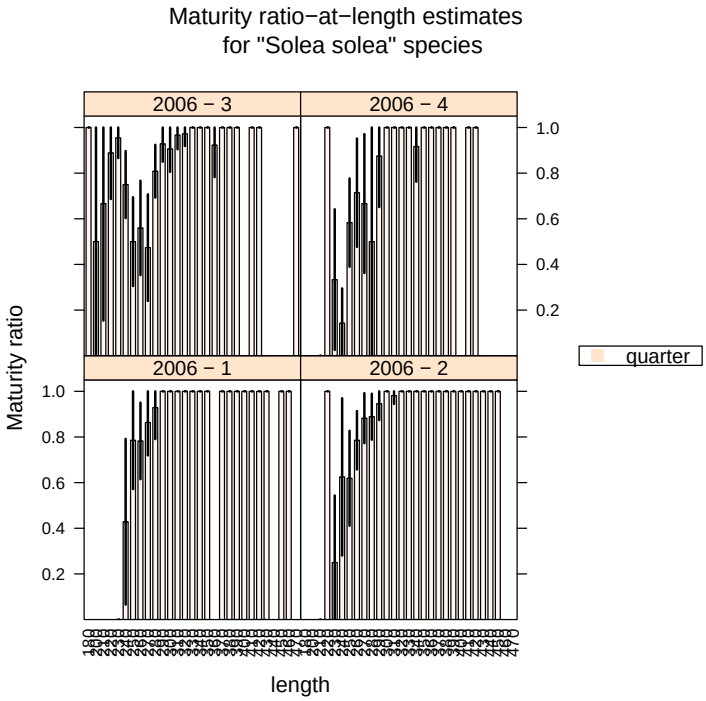
A DCR precision indicator will be calculated when the outcome of the ICES WKPRECISE will be known (October 2008). It is anticipated in the structure of the 'dbeOutput' object that such an information is needed and will be provided. This DCR precision estimates will not necessitate a new method, but will be an update of the current CV calculation method. For the moment, what is computed is the weighted mean over all strata and length/age classes.

```
> #cv at length
> matEstim_Boot <- dbeCalc(matEstim_Boot,type="CV",vrbl="l",update=TRUE)
> #DCR indicator
> matEstim_Boot@lenNum$DCRcvIndicator

[1] 0.03851852

> #plot of cv at length
> dbePlot(matEstim_Boot,elmt="lenNum$cv",ylab="CV",main="CV of maturity ratio-at-length
+ estimates \n for \"Solea solea\" species")
```



The confidence intervals have been calculated based on the normality assumption and the student distribution. With the aim of proposing realistic results, COST has replaced the negative values sometimes occuring for lower bounds by 0, and truncated to 1 all upper bounds superior to this value.

```
> #ci at length
> matEstim_Boot <- dbeCalc(matEstim_Boot,type="CI",vrbl="l",update=TRUE)
> dbePlot(matEstim_Boot,elmt="lenNum$ci",col="seashell",l.col="black",l.lwd=2,ylab="Maturity ratio",
+ main="Maturity ratio-at-length estimates \n for \"Solea solea\" species")
```



12.3 Sex-ratio

A third `dbeOutput` object is created for sex-ratio parameter.

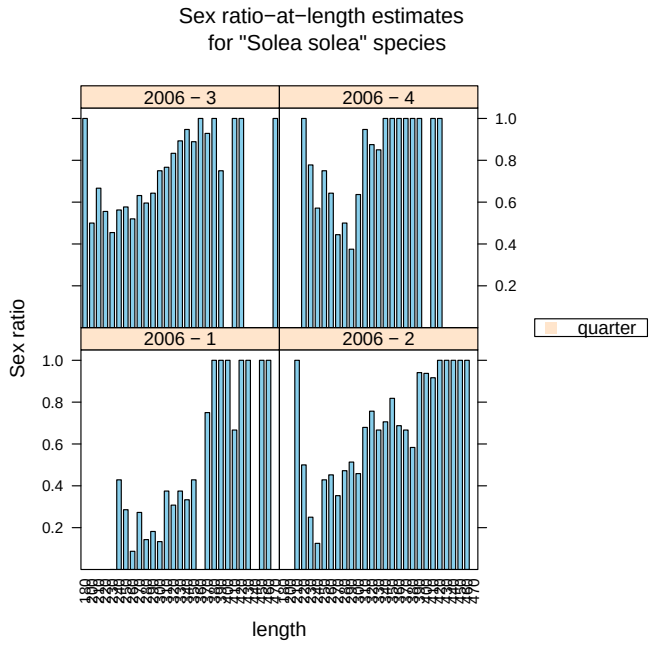
```
> srEstim_An <- dbeObject(desc="Sole 2006 Sex-ratio",species="Solea solea",catchCat="LAN",param="sex",
+ strataDesc=strBP, methodDesc="analytical")
```

Analytical estimates of sex ratio-at-length and age, and associated variances at-length and at-age are computed.

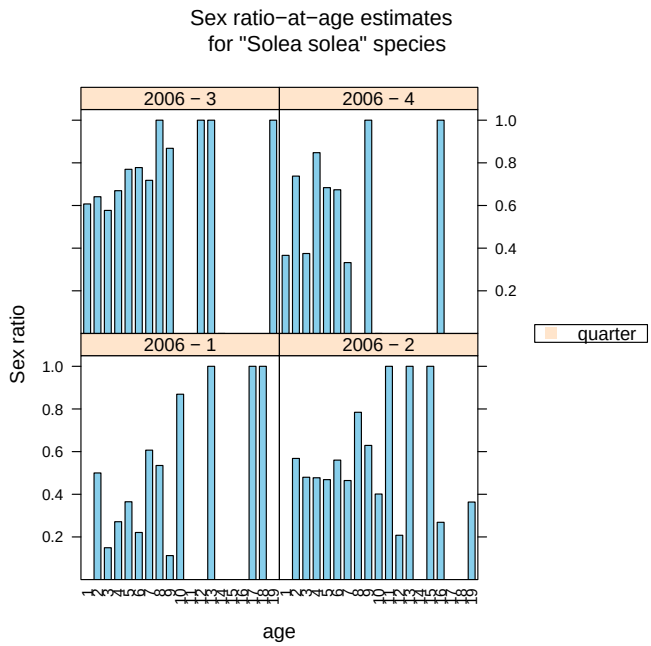
```
> srEstim_An <- bpEstim(srEstim_An,s7dbp_csc,adjust=TRUE)
```

The function 'dbePlot' is designed to plot the main dataframes contained in the 'dbeOutput' object. It is then possible to plot the length structure (lenStruc$estimdataframe), theagestructure(ageStruc$estim dataframe) but also the CVs (lenNum$cvandageNum$cv dataframes), the confidence intervals (lenNum$ciandageNum$ci dataframes). As the same object is used for containing all the dbe output, by lenStruc, ageStruc, lenNum, ageNum, we understand sex-ratio-at-length and sex-ratio-at-age here.

```
> #plot of sex ratio-at-length data
>
> dbePlot(srEstim_An,elmt="lenStruc$estim",ylab="Sex ratio",main="Sex ratio-at-length
+ estimates \n for \"Solea solea\" species")
```



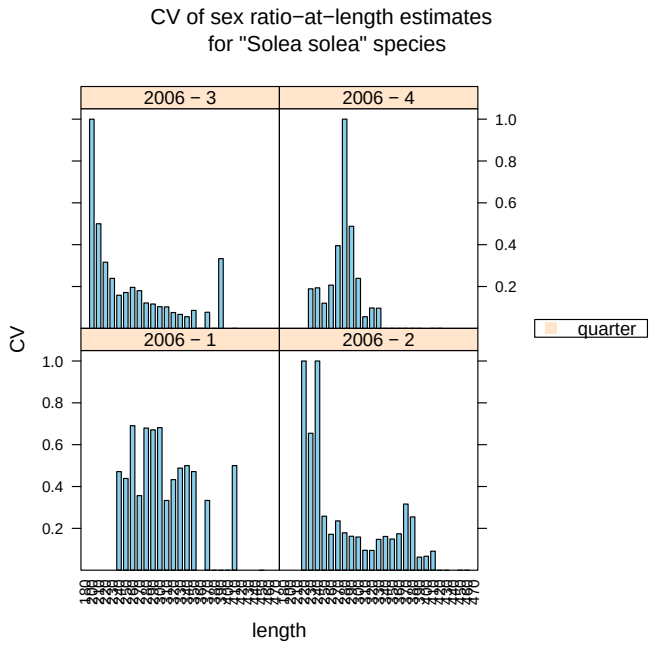

```
> #plot of sex ratio-at-age data
>
> dbePlot(srEstim_An,elmt="ageStruc$estim",ylab="Sex ratio",main="Sex ratio-at-age
+ estimates \n for \"Solea solea\" species")
```



A DCR precision indicator will be calculated when the outcome of the ICES WKPRECISE will be known (October 2008). It is anticipated in the structure of the 'dbeOutput' object that such an information is needed and will be provided. This DCR precision estimates will not necessitate a new method, but will be an update of the current CV calculation method. For the moment, what is computed is the weighted mean over all strata and length/age classes.

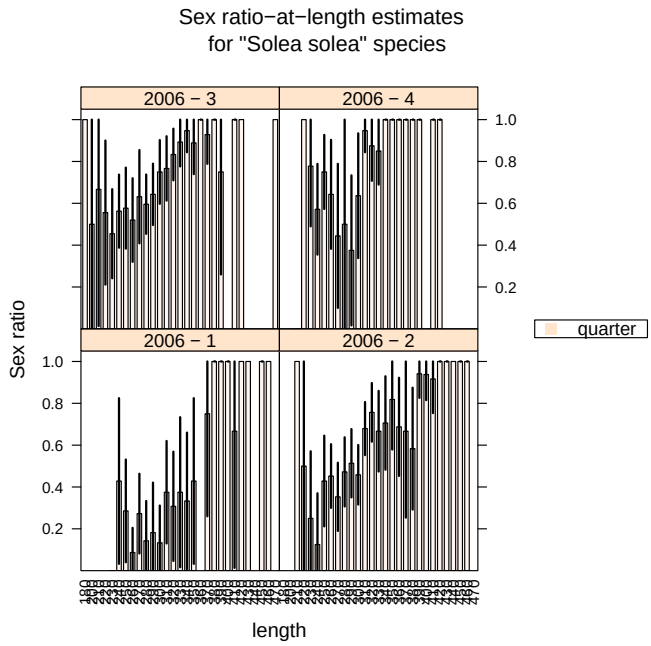
```
> #cv at length
> srEstim_An <- dbeCalc(srEstim_An,type="CV",vrbl="l",update=TRUE)
> #DCR indicator
> srEstim_An@lenNum$DCRcvIndicator
```

```
[1] 0.1165799

> #plot of cv at length
> dbePlot(srEstim_An,elmt="lenNum$cv",ylab="CV",main="CV of sex ratio-at-length
+ estimates \n for \"Solea solea\" species")
```



The confidence intervals have been calculated based on the normality assumption and the student distribution. With the aim of proposing realistic results, COST has replaced the negative values sometimes occuring for lower bounds by 0, and truncated to 1 all upper bounds superior to this value.

```
> #ci at length
> srEstim_An <- dbeCalc(srEstim_An,type="CI",vrbl="l",update=TRUE)
> dbePlot(srEstim_An,elmt="lenNum$ci",col="seashell",l.col="black",l.lwd=2,ylab="Sex ratio",
+ main="Sex ratio-at-length estimates \n for \"Solea solea\" species")
```



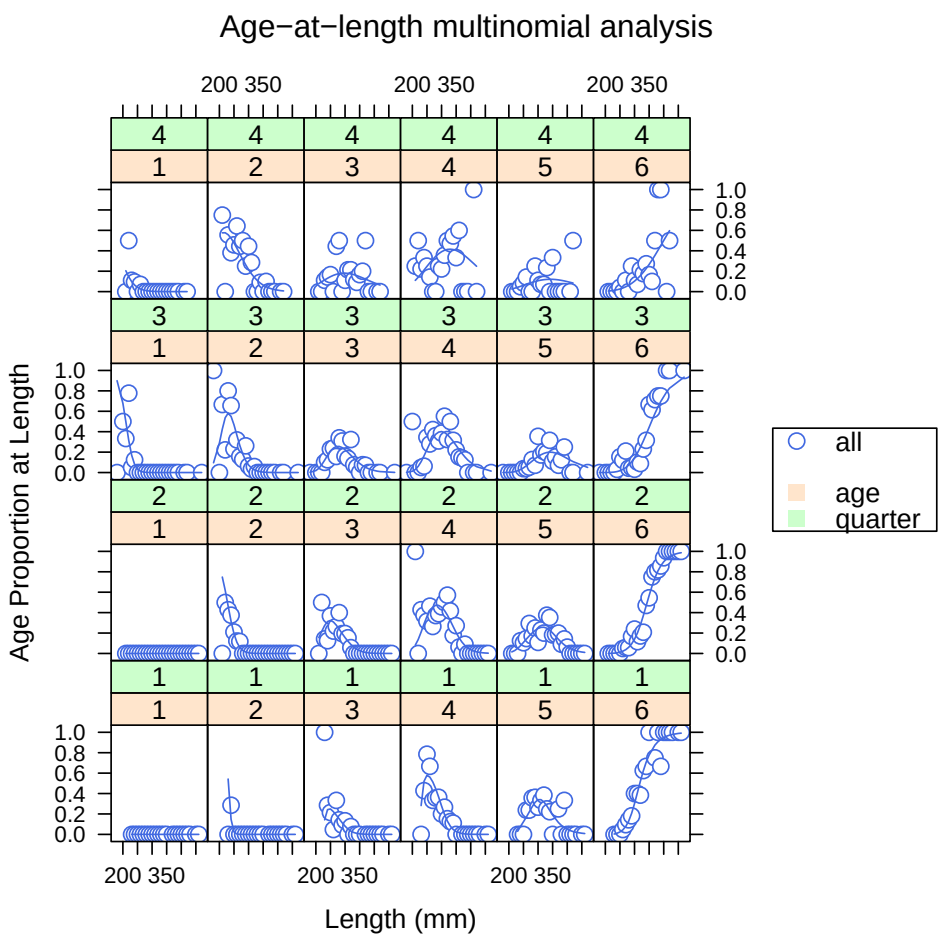
13 Modelling

13.1 Modelling ALK

A method computing a multinomial analysis of age-at-length data was implemented. It can be used to compare different age-length keys within different strata. A 'plot' of the output object can then reveal differences between strata, or, on the contrary, underline similarities. Considering *ageLenMulti* method output as the full multinomial log-linear model within specified stratification, various strata effects studies can then be carried out from this multinomial nested models or/and contrasts redefinition. All details are in

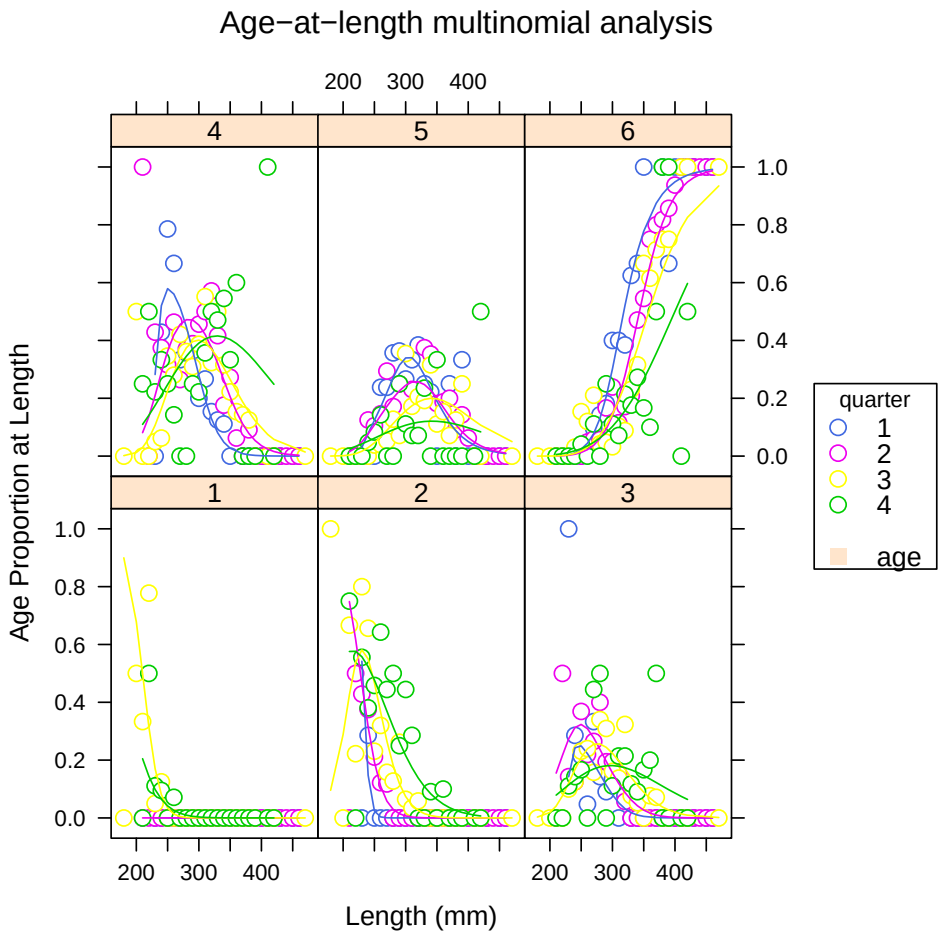
Modelling ALK information from `s7d_csv` object within time stratification ('quarter') :

```
> aLM <- ageLenMulti(s7d_csv,strBP,age.plus=6)

# weights:  54 (40 variable)
initial  value 2820.229405
iter  10 value 2514.656594
iter  20 value 2512.541593
iter  30 value 1965.958769
iter  40 value 1949.053749
iter  50 value 1946.701936
iter  60 value 1945.660644
iter  70 value 1945.572960
iter  80 value 1945.546960
final  value 1945.546271
converged

> plot(aLM)
```



To make graphical analysis easier, a zoom on one stratum can be displayed by panel :

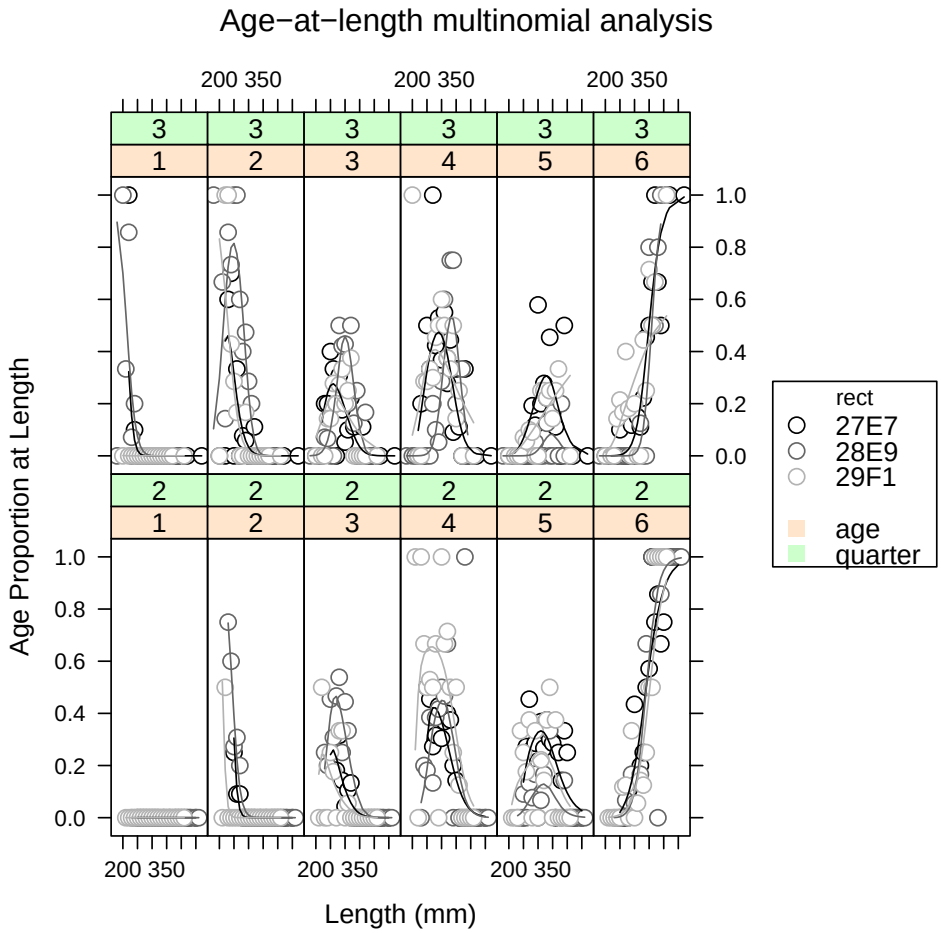
```
> plot(aLM,grps="timeStrata")
```



In that case, each stratum seems to be significantly different from the others.
Let's try to add a spatial stratification ('rect'), and make a restriction upon time stratification (only quarters 2 and 3 are kept):

```
> strM <- strIni(timeStrata="quarter",spaceStrata="rect")
> aLM2 <- ageLenMulti(s7d_csv,strM,elmts=list(tp=c("2","3"),sp="all",tc="all"),age.plus=6)

# weights:  78 (60 variable)
initial  value 2012.145884
iter  10 value 1783.672007
iter  20 value 1782.240158
iter  30 value 1781.861617
iter  40 value 1288.064170
iter  50 value 1271.114342
iter  60 value 1269.492364
iter  70 value 1268.774582
iter  80 value 1268.377385
iter  90 value 1268.259771
iter 100 value 1268.243016
final  value 1268.243016
stopped after 100 iterations
```

```
> plot(aLM2,grps="spaceStrata",l.col=c("black","grey40","grey70"))
```



13.2 Bayesian estimates

The Bayesian estimates program is run using the function run.bayes. The parameters are as follows:

1. **csdata** : The cs data object (in Cost format) to be analysed (no default). Important limitation: The mbe program will at present only work with data types `M` and `S`, ie market sampling where the trip is known, and at-sea sampling.
2. **cldata** : The cl data object (in Cost format) to be analysed (no default)
3. **fit**: A fit object from a previous estimation, or NULL if estimation is to be done on csdata (default NULL)
4. **do.predict**: Logical T or F. If true estimates of catch at age etc will be made, if false only fitting will be done (default T)
5. **species**. Character eg "Melanogrammus aeglefinus". No default.
6. **Season.definition**: Character "quarter" or "mont". Default "quarter"
7. **burnin**: Integer. How many mcmcsamples to run before storing results. Default 2000
8. **thin**: Integer. What proportion of mcmc sample sto keep, eg thin=10 means keep every 10th (after burnin). Default 1 (ie keep all)
9. **nmcmc**: Integer. Number of mcmc samples to keep after burnin (so if thin is 10 and nmcmc=1000, 10000 samples are run in all). Default 1000.
10. **ageMin**: Integer. Minimum age to be considered. Can be lower than lowest observed age in data. Default 0.
11. **ageMax**: Integer. Maximum age to be considered. Can be higher than highest observed age in data. Default 20. All ages greater than or equal to ageMax will be merged into a plus group.

12. **wgl.eqn**: Equation relating weight to length if no weights in data. If Null weight given length relationship is estimated from data. Default NULL.

13. **age.covariates**: List with logical elements season, gear and area. Defines which covariates to include in the age and length-given-age models. No default.

14. **weight.covariates**: List with logical elements season, gear and area. Defines which covariates to include in the weight-given-length model. No default.

15. **prediction.cells**: List with elements areas, gears and seasons. Defines over which levels of covariates results should be summed. Each element should be either a vector of names (eg `prediction.cells$areas=c("VI","VII")`) or `"ALL"`. No default.

16. **arealist**. Vector. Areas to include in analysis if more than those in the data (see below). Default NULL.

17. **length.list**: List with elements minl, maxl and int. Defines the length intervals for which output will be calculated, eg if minl=10, maxl=100 and int=10, the catch at length will be given in intervals of length 10 with midpoints from 10 to 100. The units are whatever the length units in the data are.

18. **adj**. Adjacency structure for areas. List with elements num, adj and areanames. See below.

Details: The first two parameters are the data to be analysed. All data for the species in parameter 5 will be included. If a smaller subset is to be analysed, the data must be subsetted before running run.bayes. If wgl.eqn is not NULL, it must contain a list of this form:

```
> wgl.eqn<-list(slope=list(const=2.8268,seas=NULL,gear=NULL,area=NULL),
+ int=list(const=-4.15625,seas=c(-0.0107,-0.0237,-0.0774,-0.0569,-0.0435,
+         -0.0368,0.0211,0.0397,0.0458,0.0817,0.0458,0.0148), gear=NULL,area=NULL))
```

The elements `slope` and `int` must each be lists containing vectors seas, gear and area. These must either be NULL, or contain one value per level of the covariate.
age.covariates and weight.covariates do not have to be the same. If for example age.covariates = list(season=T,gear=T,area=F) then the season and gear covariates will be fitted, but not the area.
The adjacency structure must be defined. For some areas this is done within the program. In order to find which areas these are type unique(unlist(adjacent.areas)). If the areas in the data are not all included here, then adj cannot be NULL. It must have the following form:

```
> adj <- list(num=c(2,1,1),adj=c(2,3,1,1),areanames=c("area1","area2","area3"))
```

The first parameter defines the number of neighbours each area has. In this case there are only three areas, the3 first has two neighbours and the other two only one. The adj parameter defines which these neigbouring areas are, so area 1 is adjacent to areas 2 and 3, area 2 is adjacent to area 1, and area 3 is adjacent to area 1. The areanames list must be identical to the arealist parameter. This defines the full list of areas that should be included in the analysis. Because the areas are modeled with a spatial smoothing method, it is possible to include areas for which there is no data (this is not true of gears or seasons). However , the areas must all be contiguous. If this is not the case it is necessary to include extra areas in order to join up the isolated areas. arealist must include all the areas in the data.

```
> library(COSTmbe)
> library(COSTeda)
> library(COSTdata)
> library(COSTdbe)
> data(CEFAS2006cs)
> data(CEFAS2006cl)
> species<-"HAD"
> use.areas<-c("VIIE", "VIIH", "VIIG", "VIIF", "IVB") #list of areas to include in analysis:
> season.definition<-"quarter"
> ageMin<-0 # minimum age in model
> ageMax<-9 # maximum age in model
> wgl.eqn<-NULL
> age.covariates<-list(season=T,gear=F,area=F) # covariates to include in age and legnth-given-age mode
> weight.covariates<-list(season=T,gear=F,area=F) #covariates to include in weight-given-length model.
> prediction.cells<-list(areas="ALL",gears="ALL",seasons="ALL") # catch at age etc will be summed over
```

```
> length.list<-list(minl=50,maxl=1000,int=50)
> num.areas <- c(2,3,3,3,0)
> adj.areas <- c(4,2,1,4,3,2,1,4,3,1,2)
> adj <- list(num=num.areas,adj=adj.areas,areanames=use.areas)
> fit<-run.bayes(csdata=CEFAS2006cs,cldata=CEFAS2006cl,fit=NULL,do.predict=T,
+ species,season.definition=season.definition,burnin=2000,thin=1,nmcmc=1000,
+ ageMin=ageMin,ageMax=ageMax,wgl.eqn=wgl.eqn,age.covariates,weight.covariates,
+ prediction.cells,use.areas,length.list,adj)

Start MCMC runs - age and lga model
Success fitting age and lga model
Start MCMC runs - wgl model
Success fitting wgl model
Finished cost.fit
Start calling caa_predict
Finished calling caa_predict
```

The output of the run.bayes function is a list with three elements:

1. **fit**. This is a structure containing all the mcmc samples of all the fitted parameters. This can be used to get predictions for other cells without re-running the fit.

2. **predict**. This contains the estimates of catch at age etc. It has 10 elements:

   (a) **cov**. This is list defining the strata over which results have been summed.

   (b) **avec**. The age classes used in the analysis

   (c) **l.int**. The length classes used in the analysis (on a log scale) As input to run.bayes.

   (d) **totcatch.land**. An array number of length classes x number of age classes x number of mcmc samples. This contains a set of matrices of numbers landed for each combination of age class and length class, with one matrix per mcmc sample. To get landings at age use apply(totcatch.land, 2:3, sum)

   (e) **totcatch.disc**: As above for discards. Note that this will be created even if there are no discard data, in which case it should be treated with extreme caution.

   (f) **mean.lga.land**. A matrix number of age classes x number of mcmc samples. The mean length of landings in each age class.

   (g) **mean.lga.disc**. As above for discards.

   (h) **mean.wga.land**. mean weight at age for landings.

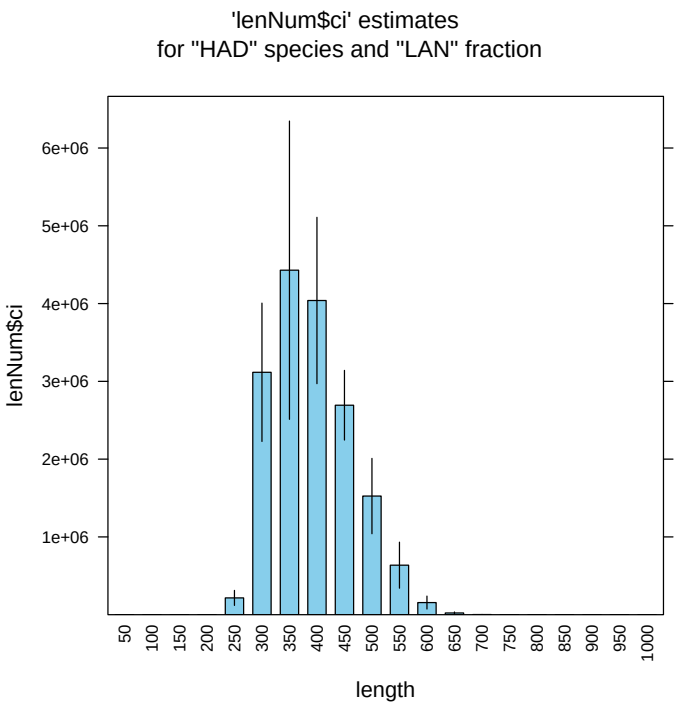
   (i) **mean.wga.disc**. As above for discards.

   (j) The proportion of the total catch landed, in each stratum (this is the only output that is not combined over all strata).

3. **dbeObject.list**. This is a list of 6 objects in dbeOutput format, for compatibility with other COST functions. They contain the numbers at age landed and discarded, the mean weight at age landed and discarded, and the mean length at age landed and discarded.

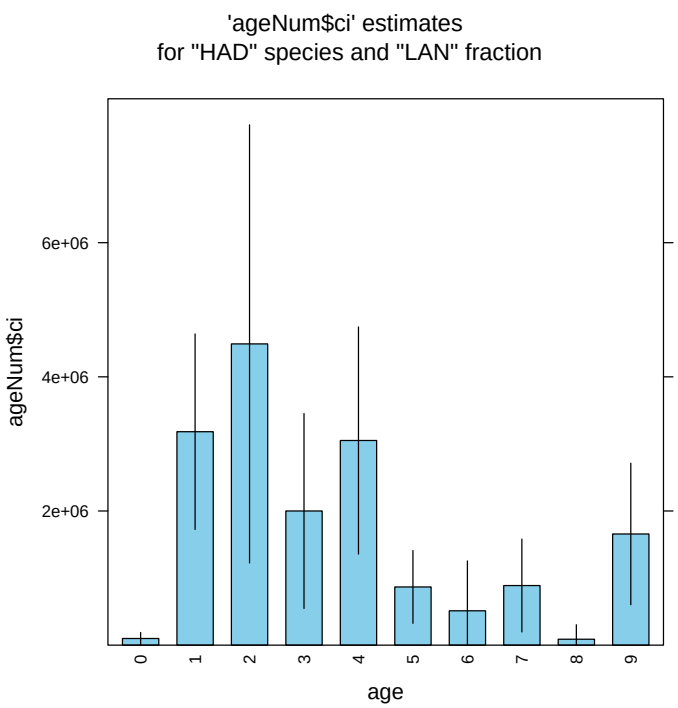
Using the methods developed for the `dbeObjects` allows the calculation of confidence intervals, coefficients of variation and the plotting of the results. Below is the length structure of the total landings with the confidence intervals.

```
> fit$dbeObject.list$dbeTotcatch.land <- dbeCalc(fit$dbeObject.list$dbeTotcatch.land,
+ vrbl="l", type="CI",probs=c(0.025,0.975),update=TRUE)
> fit$dbeObject.list$dbeTotcatch.land <- dbeCalc(fit$dbeObject.list$dbeTotcatch.land,
+ vrbl="l", type="CV", update=TRUE)
> dbePlot(fit$dbeObject.list$dbeTotcatch.land,elmt="lenNum$ci")
```



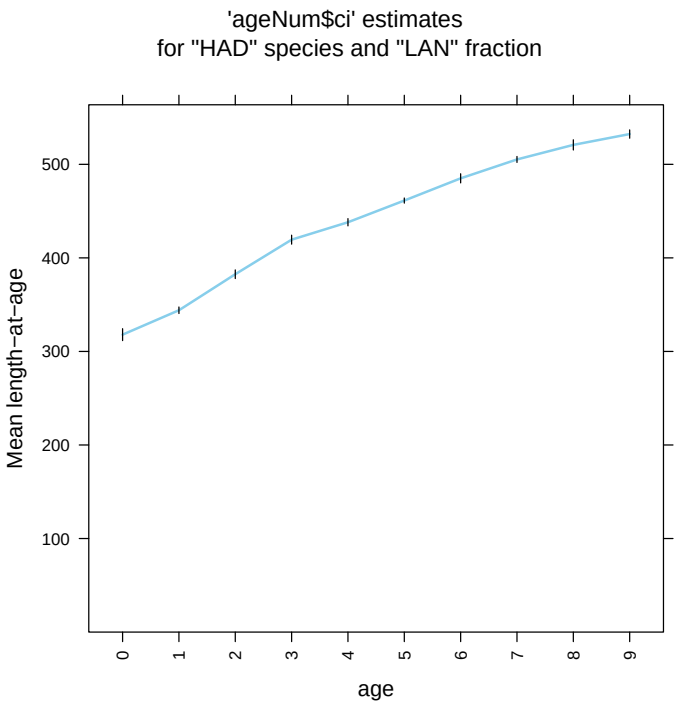
Below is the age structure of the total landings with the confidence intervals.

```
> fit$dbeObject.list$dbeTotcatch.land <- dbeCalc(fit$dbeObject.list$dbeTotcatch.land,
+ vrbl="a", type="CI",probs=c(0.025,0.975),update=TRUE)
> fit$dbeObject.list$dbeTotcatch.land <- dbeCalc(fit$dbeObject.list$dbeTotcatch.land,
+ vrbl="a", type="CV", update=TRUE)
> dbePlot(fit$dbeObject.list$dbeTotcatch.land,elmt="ageNum$ci")
```



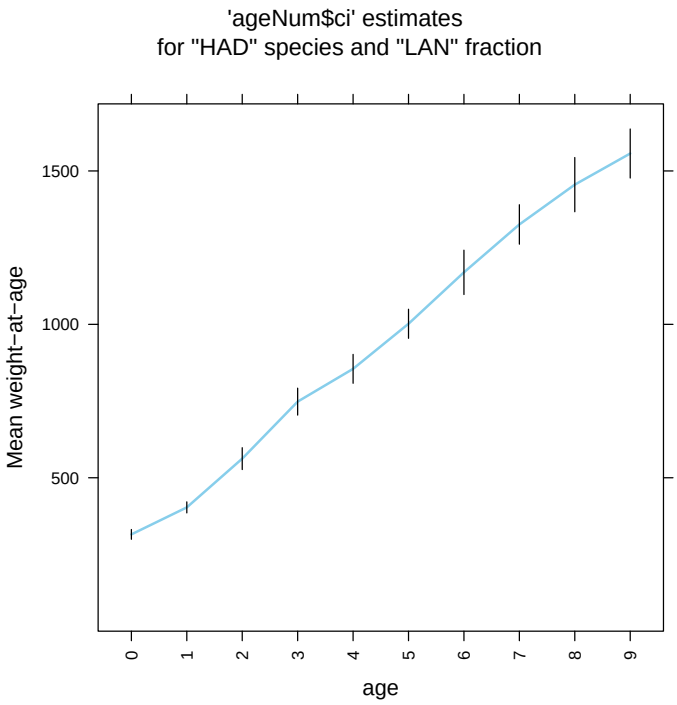
Below is mean length-at-age of the landings, with the confidence intervals.

```
> fit$dbeObject.list$dbeMean.lga.land <- dbeCalc(fit$dbeObject.list$dbeMean.lga.land,
+ vrbl="a", type="CI",probs=c(0.025,0.975),update=TRUE)
> fit$dbeObject.list$dbeMean.lga.land <- dbeCalc(fit$dbeObject.list$dbeMean.lga.land,
+ vrbl="a", type="CV", update=TRUE)
> dbePlot(fit$dbeObject.list$dbeMean.lga.land,elmt="ageNum$ci", ylab="Mean length-at-age",type="line",
```



Below is mean weight-at-age of the landings, with the confidence intervals.

```
> fit$dbeObject.list$dbeMean.wga.land <- dbeCalc(fit$dbeObject.list$dbeMean.wga.land,
+ vrbl="a", type="CI",probs=c(0.025,0.975),update=TRUE)
> fit$dbeObject.list$dbeMean.wga.land <- dbeCalc(fit$dbeObject.list$dbeMean.wga.land,
+ vrbl="a", type="CV", update=TRUE)
> dbePlot(fit$dbeObject.list$dbeMean.wga.land,elmt="ageNum$ci", ylab="Mean weight-at-age",type="line",
```

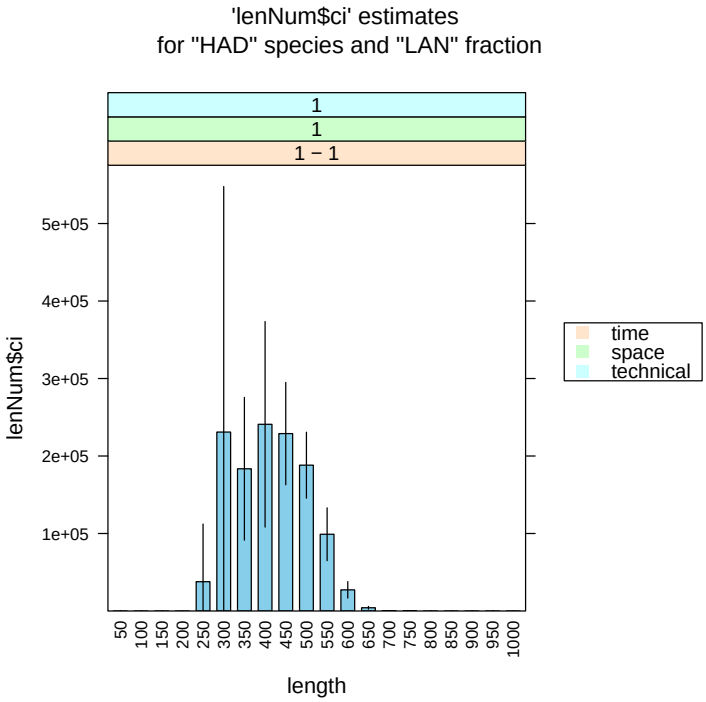


Estimating only one strata necessitates rerunning `run.bayes` with the strata given in the argument `prediction.cells`. Below is the code for estimating all parameters for area "IVB" and quarter 1.

```
> prediction.cells<-list(areas="IVB",gears="ALL",seasons="1")
> fit.4b.Q1<-run.bayes(csdata=CEFAS2006cs,cldata=CEFAS2006cl,fit=fit$fit,do.predict=T,
+  species,season.definition=season.definition,burnin=2000,thin=1,nmcmc=1000,
+  ageMin=ageMin,ageMax=ageMax,wgl.eqn=wgl.eqn,age.covariates,weight.covariates,
+  prediction.cells,use.areas,length.list,adj)
```

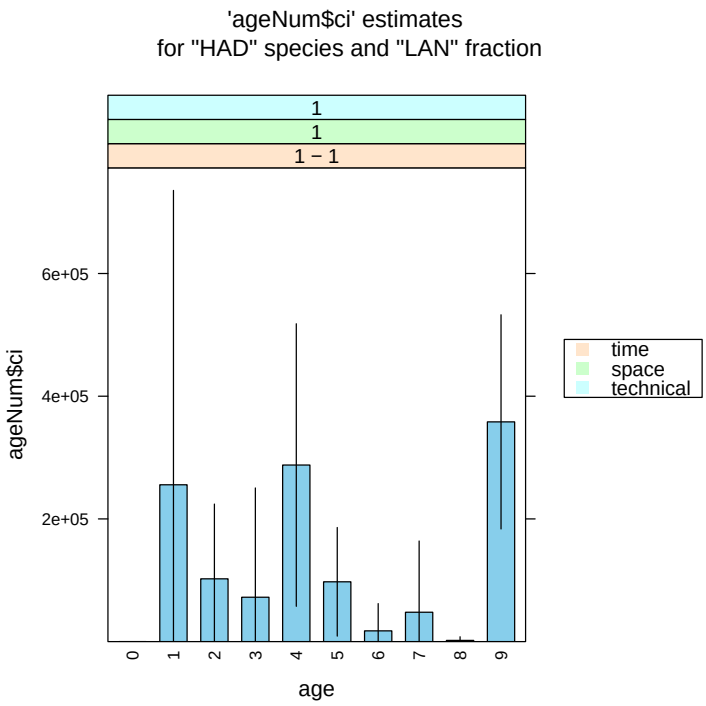
Resulting graphs for length and age structures of the landings are shown below.

```
> fit.4b.Q1$dbeObject.list$dbeTotcatch.land <- dbeCalc(fit.4b.Q1$dbeObject.list$dbeTotcatch.land,
+ vrbl="l", type="CI",probs=c(0.025,0.975),update=TRUE)
> fit.4b.Q1$dbeObject.list$dbeTotcatch.land <- dbeCalc(fit.4b.Q1$dbeObject.list$dbeTotcatch.land,
+ vrbl="l", type="CV", update=TRUE)
> dbePlot(fit.4b.Q1$dbeObject.list$dbeTotcatch.land,elmt="lenNum$ci")
```



Below is the age structure of the total landings with the confidence intervals.

```
> fit.4b.Q1$dbeObject.list$dbeTotcatch.land <- dbeCalc(fit.4b.Q1$dbeObject.list$dbeTotcatch.land,
+ vrbl="a", type="CI",probs=c(0.025,0.975),update=TRUE)
> fit.4b.Q1$dbeObject.list$dbeTotcatch.land <- dbeCalc(fit.4b.Q1$dbeObject.list$dbeTotcatch.land,
+ vrbl="a", type="CV", update=TRUE)
> dbePlot(fit.4b.Q1$dbeObject.list$dbeTotcatch.land,elmt="ageNum$ci")
```



14 Exportation of the estimates

Before exporting estimates, user may need to combine strata and, therefore, aggregate the data over these strata. This can be done with `stratAggreg` method, applied to an input `dbeOutput` object. For example, if we want to transform a quarterly data to a yearly data, we do :

```
> disEstimAg <- stratAggreg(disEstim,timeStrata=TRUE,spaceStrata=FALSE,techStrata=FALSE)
> head(disEstimAg@lenStruc$estim)
```

```
  time space technical length value
1  all   all   GTR_DEF    120     0
2  all   all   GTR_DEF    140     0
3  all   all   GTR_DEF    150     0
4  all   all   GTR_DEF    160     0
5  all   all   GTR_DEF    170  1156
6  all   all   GTR_DEF    180     0
```

There are many ways to export the estimates listed in *dbeOutput* object, either in their original format, or possibly passing through a specific conversion process. Thus, it is possible to export these data into InterCatch Exchange Format, into FLR format, or simply into CSV tables.

14.1 to InterCatch

This can be achieved in two steps... :

A list with HI, SI, SD dataframes is created by `makeICdf` command :

```
> #Create list with HI, SI, SD dataframes
> Est.ICdf <- makeICdf (disEstim, Country = "FRA", Species = "Solea solea",
+ ReportingCategory = "R", SamplesOrigin = "M")
```

Output object looks like :

```
> lapply(Est.ICdf,head)

$HI
  RecordType Country Year SeasonType Season   Fleet AreaType FishingArea DepthRange
1         HI     FRA 2006    Quarter      1 OTB_DEF     <NA>         all         NA
2         HI     FRA 2006    Quarter      2 GTR_DEF     <NA>         all         NA
3         HI     FRA 2006    Quarter      2 OTB_DEF     <NA>         all         NA
4         HI     FRA 2006    Quarter      3 GTR_DEF     <NA>         all         NA
5         HI     FRA 2006    Quarter      3 OTB_DEF     <NA>         all         NA
6         HI     FRA 2006    Quarter      4 GTR_DEF     <NA>         all         NA
  UnitEffort Effort AreaQualifier
1         NA     -9            NA
2         NA     -9            NA
3         NA     -9            NA
4         NA     -9            NA
5         NA     -9            NA
6         NA     -9            NA

$SI
  RecordType Country Year SeasonType Season   Fleet AreaType FishingArea DepthRange
1         SI     FRA <NA>    Quarter      1 OTB_DEF     <NA>         all         NA
2         SI     FRA <NA>    Quarter      2 GTR_DEF     <NA>         all         NA
3         SI     FRA <NA>    Quarter      2 OTB_DEF     <NA>         all         NA
4         SI     FRA <NA>    Quarter      3 GTR_DEF     <NA>         all         NA
5         SI     FRA <NA>    Quarter      3 OTB_DEF     <NA>         all         NA
6         SI     FRA <NA>    Quarter      4 GTR_DEF     <NA>         all         NA
      Species Stock CatchCategory ReportingCategory DataToFrom Usage SamplesOrigin
1 Solea solea    NA             D                 R         NA    NA             M
2 Solea solea    NA             D                 R         NA    NA             M
3 Solea solea    NA             D                 R         NA    NA             M
4 Solea solea    NA             D                 R         NA    NA             M
5 Solea solea    NA             D                 R         NA    NA             M
6 Solea solea    NA             D                 R         NA    NA             M
  QualityFlag UnitCATON     CATON OffLandings   varCATON InfoFleet InfoStockCoordinator
1          NA        kg    0.0000          NA       0.00        NA                   NA
2          NA        kg 1537.4800          NA       0.00        NA                   NA
3          NA        kg 3879.5824          NA 1555475.74        NA                   NA
4          NA        kg  152.4656          NA   18268.20        NA                   NA
5          NA        kg 1768.8974          NA  354221.20        NA                   NA
6          NA        kg    0.0000          NA       0.00        NA                   NA
  InfoGeneral
1          NA
2          NA
3          NA
4          NA
5          NA
6          NA

$SD
  RecordType Country Year SeasonType Season   Fleet AreaType FishingArea DepthRange
1         SD     FRA 2006    Quarter      1 OTB_DEF     <NA>         all         NA
2         SD     FRA 2006    Quarter      1 OTB_DEF     <NA>         all         NA
3         SD     FRA 2006    Quarter      1 OTB_DEF     <NA>         all         NA
```

```
4         SD     FRA 2006    Quarter      1 OTB_DEF     <NA>         all         NA
5         SD     FRA 2006    Quarter      1 OTB_DEF     <NA>         all         NA
6         SD     FRA 2006    Quarter      1 OTB_DEF     <NA>         all         NA
      Species Stock CatchCategory ReportingCategory Sex CANUMtype AgeLength PlusGroup
1 Solea solea    NA             D                 R   N    length       120        -9
2 Solea solea    NA             D                 R   N    length       140        -9
3 Solea solea    NA             D                 R   N    length       150        -9
4 Solea solea    NA             D                 R   N    length       160        -9
5 Solea solea    NA             D                 R   N    length       170        -9
6 Solea solea    NA             D                 R   N    length       180        -9
  SampledCatch NumSamplesLngt NumLngtMeas NumSamplesAge NumAgeMeas unitMeanWeight
1           -9             -9          -9            -9         -9              g
2           -9             -9          -9            -9         -9              g
3           -9             -9          -9            -9         -9              g
4           -9             -9          -9            -9         -9              g
5           -9             -9          -9            -9         -9              g
6           -9             -9          -9            -9         -9              g
  unitCANUM UnitAgeOrLength UnitMeanLength Maturity NumberCaught MeanWeight MeanLength
1         n              mm             mm       NA            0         -9         -9
2         n              mm             mm       NA            0         -9         -9
3         n              mm             mm       NA            0         -9         -9
4         n              mm             mm       NA            0         -9         -9
5         n              mm             mm       NA            0         -9         -9
6         n              mm             mm       NA            0         -9         -9
  varNumLanded varWgtLanded varLgtLanded
1            0           -9          -9
2            0           -9          -9
3            0           -9          -9
4            0           -9          -9
5            0           -9          -9
6            0           -9          -9
```

It is then exported with `makeICfile` command :

```
> # Not run
> makeICfile (Est.ICdf, filename ="IC_Est.csv")
```

...or directly :

```
> makeICdf(disEstim, filename ="IC_Est.csv", output.df = FALSE, Country = "FRA",
+ Species = "Solea solea", ReportingCategory = "R", SamplesOrigin = "M")
```

14.2 to FLR

The export method to FLR will further be included in a specific package (COSTlink) dedicated to the linkage with other projects and the production of data at the appropriate format. The reason for externalising the method in a specific package is that the use of the export method to FLR necessitates loading the FLCore library. The FLR `FLQuant` method has been supplemented with a new function recognising a dbeObject and reorganising the dbeOutput information in the FLR format. For the example below, we make use of the dbeObject previously calculated in section 6, called `lanEstim`.

```
> library(FLCore)
> flq <- FLQuant(lanEstim)
```

You can check the results

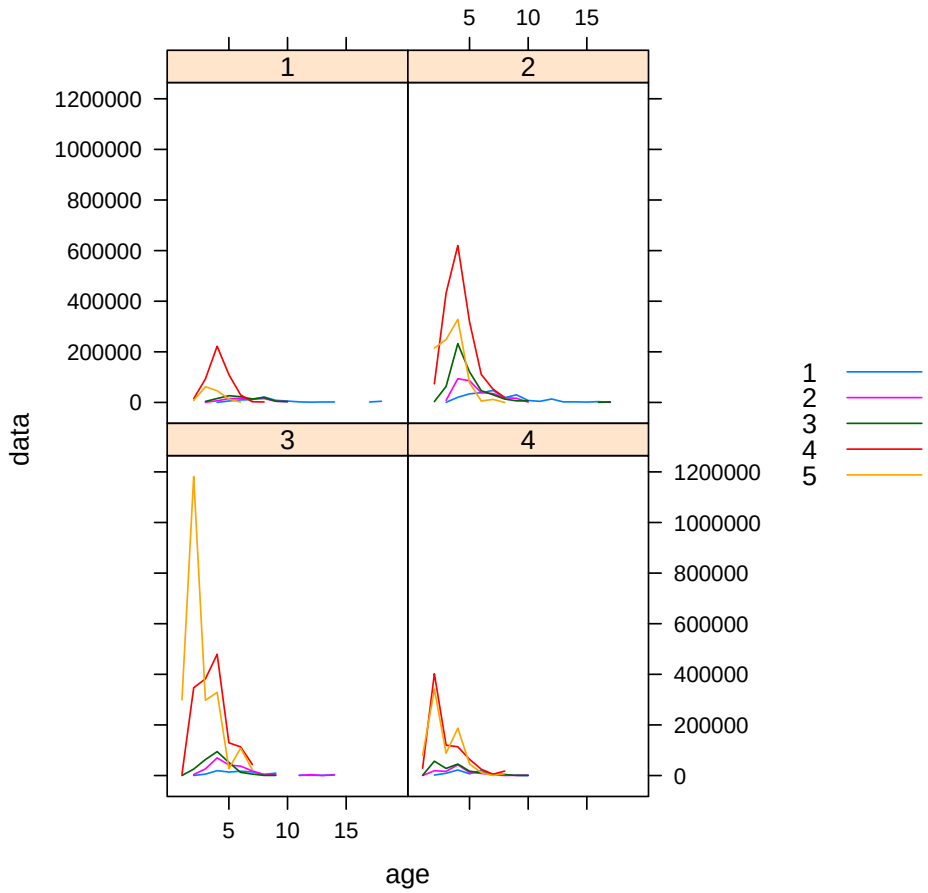
```
> head(lanEstim@ageStruc$estim)
```

```
      time space technical age       value
1 2006 - 1   all         1   4    97.37879
2 2006 - 1   all         1   5  5672.46684
3 2006 - 1   all         1   6  9884.29727
4 2006 - 1   all         1   7 11168.44178
5 2006 - 1   all         1   8 21547.90468
6 2006 - 1   all         1   9  8260.66720
```

```
> flq["4","2006","1","1","all",]

An object of class "FLQuant"
, , unit = 1, season = 1, area = all

   year
age 2006
  4 97.379

units:  NA

> flq["9","2006","1","1","all",]

An object of class "FLQuant"
, , unit = 1, season = 1, area = all

   year
age 2006
  9 8260.7

units:  NA
```

Welcome to the wonderful world of FLR!

```
> xyplot(data~age|season,groups=unit, data=flq, auto.key=list(points=F,
+ lines=T, space="right"), type="l", as.table=T)
```



14.3 to CSV tables

When `dbeOutput` estimates don't need any conversion nor formatting, a direct export can be processed using R export functions such as `write.table` or `write.csv`. These functions are preferably applied to dataframes, so they are specifically convenient for elements contained in *dbeOutput* object. For example, exporting length structure estimates table from `disEstim` object into .csv files can be done with :

```
> write.table(disEstim@lenStruc$estim,file="disEst.csv",sep=";",row.names=FALSE)
> write.csv(disEstim@lenStruc$estim,file="disEst.csv")
```

15 Simulation program

The simulation program works as follows:

1. Take a real data set and fit the Bayesian model. Save one set of parameters from the joint posterior distribution (`initial.byfit`).
2. Find the true values of relevant catch-at-age etc given this set of parameters (`true`)
3. Define the sampling structure of the data to be simulated ("setup.data")
4. Simulate a new set of data from the parameters in step 1.
5. Estimate catch-at-age and other parameters using the Bayesian model or the bootstrap or analytical methods.
6. Compare the estimates with the truth.
7. Go to step 4 and repeat a large number of times.

In more detail :

Step 1 The model is fitted as described in the relevant section. Any number of samples can be saved (by changing the numit.outer parameter), but it is simplest only to save 1. The relevant saved parameters are:

initial.byfit$age This contains the parameters for the age model.

There are more variables than necessary for the simulation. The relevant ones are:

initial.byfitageInteffConst This is a matrix of dimension number of age groups by number of samples (eg 16 x 1).

```
> initial.byfit <- cost.fit(obj, data.COST, burnin = 500, numit.inner = 1,
+     numit.outer = 1, constr = 1, seed = 213421, ageMin = ageMin, ageMax = ageMax,
+     nSeason = nseas, agemodel = setup.data$agemodel, lgamodel = setup.data$lgamodel,
+     lgarel = "log-linear", model1 = T, model2 = F)
```

```
> initial.byfit$age$Int$eff$Const
            [,1]
 [1,]  0.3286831
 [2,]  5.6522236
 [3,]  7.8441188
 [4,]  8.1418695
 [5,]  6.4349999
 [6,]  5.1684546
 [7,]  2.3857818
 [8,]  1.2647051
 [9,] -0.9462614
[10,] -2.3359494
[11,] -3.8145808
[12,] -4.9834769
[13,] -5.7408312
[14,] -6.1945261
[15,] -6.9465912
[16,] -6.2586193
```

initial.byfitageInteffseas An array dimension (number of ages) x (number of seasons) x (number of samples) eg 16 x 4 x 1 (if season fitted in the model).

```
> initial.byfit$age$Int$eff$seas

, , 1

             [,1]        [,2]         [,3]        [,4]
 [1,] -3.994371222 -4.40722450  1.990049597  6.41154612
 [2,] -0.976375910 -0.91972205 -0.175251087  2.07134905
 [3,] -0.476791786 -0.92121826  0.142793698  1.25521635
 [4,]  0.381432461 -0.32946798 -0.080411076  0.02844659
 [5,] -0.018680605 -0.36594352 -0.005017805  0.38964193
 [6,]  1.172615370 -0.30438330 -0.248127337 -0.62010473
 [7,]  1.137881717  0.05084039 -0.437010825 -0.75171129
 [8,] -0.072179032  0.82631417 -0.194944224 -0.55919092
 [9,] -0.001598716  1.15407742 -0.004348519 -1.14813018
[10,]  0.120764502  0.54770192  0.730086105 -1.39855253
[11,] -0.046102686  1.27306676 -0.801949030 -0.42501505
[12,]  0.544996029 -0.21057475  0.093355430 -0.42777671
[13,]  0.377638797  0.95631135 -0.565947536 -0.76800262
[14,]  1.011929450  0.52268319 -0.010080786 -1.52453185
[15,]  0.296917166  1.48742708  0.172471466 -1.95681571
[16,]  0.541924465  0.64011206 -0.605668070 -0.57636845
```

initial.byfitageInteffgear An array dimension (number of ages) x (number of gears) x (number of samples) eg 16 x 4 x 1 (if gear fitted in the model).

initial.byfitageInteffarea An array dimension (number of ages) x (number of areas) x (number of samples) eg 16 x 4 x 1 (if area fitted in the model).

initial.byfitageInt$haul.tau The precision of the haul effect for the age model.

initial.byfitageInt$area.tau The precision of the area effect for the age model (if fitted)

initial.byfitageInt$cell.tau The precision of the cell effect for the age model (if at least two covariates fitted).

From these parameters it is possible to construct the probability at age for any random trip in any cell. Numbers at age in the catch (discards + landings) can then be simulated.

initial.byfit$lga This contains the parameters for the length-given-age model.

initial.byfitlgaInteffConst Matrix 1 x number of samples (eg 1x1). The constant for the regression.

initial.byfitlgaInteffseas Matrix (number of seasons) x (number of samples). Season effects for regression (if season fitted)

initial.byfitlgaInteffgear Matrix (number of gears) x (number of samples). Gear effects for regression (if gear fitted)

initial.byfitlgaInteffarea Matrix (number of areas) x (number of samples). Area effects for regression (if area fitted)

initial.byfitlgaSlpeffConst Slope for regression.

initial.byfitlgag.a Additional parameters for Schnute-Richards model (if used).

initial.byfitlgatau.obs The within haul residual precision for the regression.

initial.byfitlgaInt$haul.tau The precision of the haul effect for the length-given-age model (if fitted).

initial.byfitlgaInt$area.tau The precision of the area effect for the length-given-age model (if fitted)

initial.byfitlgaInt$cell.tau The precision of the cell effect for the length-given-age model (if at least two covariates fitted).

initial.byfit$wgl contains similar parameters for the weight-given-length model.

These parameters allow the length of fish in each trip to be simulated, once the ages have been simulated.

initial.byfit$COST.list$cens.mcmc This contains the parameters for the discard function. k and m are common to all trips. r gives the fitted r values for the trips in the data. Mu gives the posterior means of k,m and r, and tau gives their posterior precisions.

These parameters allow the simulated fish to be divided into discards and landings, in each trip.

Step 2 The function `predict.fit.COST` is used to obtain the true value of the landings-at-age, discards-at-age, length-given-age and weight-given-age for landings and discards, given the parameters used for the simulation. These are combined with the parameter values into a single object "true" with the function "getparams".

```
> trueCAA <- predict.fit.COST(res2, initial.byfit$COST.list, t.year, t.seas,
+     t.gear, t.area, landings, 1:nseas, t2.year = NULL, t2.seas = NULL,
+     t2.gear = NULL, t2.area = NULL, burnin = 0, nMC = 100, l.int = l.int,
+     par.haulsize = NULL)
> true <- getparams(initial.byfit, trueCAA, simno)

> true

$age
$age$p
 [1] 1.308579e-02 5.257950e-02 3.483988e-01 4.704720e-01 8.069830e-02 3.183996e-02
 [7] 2.074592e-03 7.201844e-04 1.011875e-04 2.083150e-05 5.933295e-06 1.034673e-06
[13] 7.335166e-07 4.600046e-07 3.373360e-07 3.734047e-07

$age$Const
 [1]  0.3286831  5.6522236  7.8441188  8.1418695  6.4349999  5.1684546  2.3857818
 [8]  1.2647051 -0.9462614 -2.3359494 -3.8145808 -4.9834769 -5.7408312 -6.1945261
[15] -6.9465912 -6.2586193

$age$HZ
NULL

$age$seas
              [,1]        [,2]         [,3]        [,4]
 [1,] -3.994371222 -4.40722450  1.990049597  6.41154612
 [2,] -0.976375910 -0.91972205 -0.175251087  2.07134905
 [3,] -0.476791786 -0.92121826  0.142793698  1.25521635
 [4,]  0.381432461 -0.32946798 -0.080411076  0.02844659
 [5,] -0.018680605 -0.36594352 -0.005017805  0.38964193
 [6,]  1.172615370 -0.30438330 -0.248127337 -0.62010473
 [7,]  1.137881717  0.05084039 -0.437010825 -0.75171129
 [8,] -0.072179032  0.82631417 -0.194944224 -0.55919092
 [9,] -0.001598716  1.15407742 -0.004348519 -1.14813018
[10,]  0.120764502  0.54770192  0.730086105 -1.39855253
[11,] -0.046102686  1.27306676 -0.801949030 -0.42501505
[12,]  0.544996029 -0.21057475  0.093355430 -0.42777671
[13,]  0.377638797  0.95631135 -0.565947536 -0.76800262
[14,]  1.011929450  0.52268319 -0.010080786 -1.52453185
[15,]  0.296917166  1.48742708  0.172471466 -1.95681571
[16,]  0.541924465  0.64011206 -0.605668070 -0.57636845

$age$area
NULL

$age$gear
NULL

$age$tau
[1] 0.1691056


$lga
```

```
$lga$Const
[1] 1.834174

$lga$HZ
NULL

$lga$seas
[1] -0.002420796 -0.027392739 -0.010433800  0.040247334

$lga$area
NULL

$lga$gear
NULL

$lga$tau
NULL

$lga$tau.obs
[1] 103.0814

$lga$slope
[1] 2.521875


$wgl
$wgl$Const
NULL

$wgl$HZ
NULL

$wgl$seas
NULL

$wgl$area
NULL

$wgl$gear
NULL

$wgl$tau
NULL

$wgl$tau.obs
NULL

$wgl$slope
NULL


$caa
$caa$wga.land
 [1]  176.1754  288.7956  391.2644  513.7533  726.3994  944.2284 1273.2320 1638.6882
 [9] 2031.3373 2434.0469 2892.9771 3380.1577 3912.3066 4443.3088 5017.8641 5625.5335

$caa$CAA.land
 [1] 7.911152e-07 9.560019e-02 7.252273e+00 3.306740e+01 2.078847e+01 1.301032e+01
 [7] 2.296644e+00 9.947056e-01 6.173868e-02 7.218197e-02 4.531295e-03 1.943103e-03
[13] 1.014100e-03 4.936956e-04 3.884751e-03 2.153413e-03

$caa$wga.disc
```

```
 [1]   48.99168  139.03464  245.49229  330.07333  398.70901  449.75150  499.66864
 [8]  529.34045  531.86641 8046.68290 2152.18306 3409.17836 4015.35153    0.00000
[15]    0.00000    0.00000

$caa$CAA.disc
 [1]  7.062092e+00  1.803046e+01  3.917956e+01  2.481232e+01  2.481476e+00  2.689178e-01
 [7]  7.427006e-03  1.653432e-04  1.484506e-06  2.211204e-07  7.174624e-10  1.113383e-10
[13]  2.556066e-12 -5.898411e-14 -1.779698e-12 -1.216982e-12

$caa$planded
[1] 0.4419515 0.5227803 0.4852686 0.4066259

$caa$lga.land
          [,1]
 [1,] 22.63059
 [2,] 26.94080
 [3,] 29.94523
 [4,] 32.91374
 [5,] 37.14550
 [6,] 40.74704
 [7,] 45.29856
 [8,] 49.53884
 [9,] 53.44930
[10,] 56.98796
[11,] 60.57917
[12,] 64.00993
[13,] 67.40528
[14,] 70.51219
[15,] 73.61401
[16,] 76.65171

$caa$lga.disc
           [,1]
 [1,]  14.27987
 [2,]  20.55337
 [3,]  25.31219
 [4,]  28.20750
 [5,]  30.18147
 [6,]  31.52214
 [7,]  32.70549
 [8,]  33.39417
 [9,]  33.83363
[10,] 124.93891
[11,]  52.77725
[12,]  64.32565
[13,]  68.06714
[14,]   0.00000
[15,]   0.00000
[16,]   0.00000
```

Step 3 The function `setup` defines the numbers of trips to be sampled, and various important parameters of them, such as the seasons they will be sampled from etc.

```
> setup.data <- setup(datalist, species, ageseas = T, agegear = F, agearea = F,
+     lgaseas = T, lgagear = F, lgaarea = F, wglseas = T, wglgear = F,
+     wglarea = F, ntrip = 50, ndisc = 30, div = c(4, 9), ageMin, ageMax,
+     nseas)
```

The parameters are:

Datalist This is the original data. Not currently in COST format.

Species eg"Haddock".

ageseas,agegear etc Logical parameters saying whether the various covariates have been included in the model

ntrip The total number of trips to sample

ndisc The number of sampled trips which will be observer data. These will include discard data.

div A vector used in determining how many size classes there will be in the sampled hauls

ageMin, ageMax Observed minimum and maximum ages in the data

nseas Number of seasons. Usually nseas: Number of seasons. Usually 4 or 12.

The output is:

structure The sampled trips are randomly assigned covariates, such that each level of each covariate has at least one sample (eg one sample in every season, one from every gear etc). These are defined in the "seas", "gear" and "area" variables. "cell" is a unique combination of the covariates. "dis" defines whether the sample is an observer trip, and so includes discard data. "n.size" is the number of size classes in the landings on each trip, "n.land" is the number of fish landed in each trip, "n.haul" is the number of hauls in each trip, "landratio" is the mean ratio of the total catch to the landings.

sim.models The numbers of size classes and hauls, and the number of fish in the trip are necessary for the simulations, but are not part of the model. An ad-hoc approach has therefore been used to simulate them, based on the observed data. The methods are as follows:

Number of size classes The observed numbers of landings (on a log scale) are divided into classes based on the `div` parameter, eg if this is c(4,9), then the classes are 0<4, 4-9 and >9. The proportions of times the number of size classes falls into each of these classes is then put into a matrix, eg for class 1 there might be 50% of hauls with 1 size class, and 50% with 2. This matrix is returned as setup.data$sim.models$psize. The number of size classes is then randomly assigned depending on the number of landings.

Number of hauls This is assigned in a similar way to the number of size classes. The matrix of proportions is setup.data$sim.models$phaul.

```
> setup.data$sim.models$phaul
     [,1] [,2] [,3]       [,4]      [,5] [,6] [,7]      [,8]       [,9] [,10]     [,11]
[1,]    0 0.50 0.00 0.00000000 0.5000000 0.00 0.00 0.0000000 0.00000000  0.00 0.0000000
[2,]    0 0.08 0.12 0.04000000 0.0400000 0.04 0.04 0.0000000 0.00000000  0.08 0.0800000
[3,]    0 0.00 0.00 0.04545455 0.0909091 0.00 0.00 0.0909091 0.04545455  0.00 0.0909091
          [,12] [,13]      [,14] [,15]     [,16] [,17] [,18] [,19]     [,20]      [,21]
[1,] 0.00000000  0.00 0.00000000  0.00 0.0000000  0.00  0.00  0.00 0.0000000 0.00000000
[2,] 0.08000000  0.04 0.04000000  0.12 0.0000000  0.08  0.04  0.04 0.0000000 0.04000000
[3,] 0.04545455  0.00 0.04545455  0.00 0.1363636  0.00  0.00  0.00 0.0909091 0.04545455
     [,22]     [,23] [,24] [,25]      [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33]
[1,]     0 0.0000000     0     0 0.00000000     0     0     0     0     0     0     0
[2,]     0 0.0000000     0     0 0.00000000     0     0     0     0     0     0     0
[3,]     0 0.1363636     0     0 0.04545455     0     0     0     0     0     0     0
     [,34] [,35]      [,36]      [,37]
[1,]     0     0 0.00000000 0.00000000
[2,]     0     0 0.00000000 0.00000000
[3,]     0     0 0.04545455 0.04545455
```

Number of landings This is simulated from a regression of the log(number of landings) on gear, based on the observed data.

The other parameters in `setup.data` repeat the input.

Step 4 Given all the parameters defined in the previous steps, the lengths, ages and weights of the sampled fish can be simulated.

```
> sim.data <- cost.simloop(initial.byfit, setup.data, simno = 1, burnin = burnin,
+     nmcmc = nmcmc, l.int, Int, Slp, landings, nHaul, nseas, fit = FALSE)
```

The numbers to be sampled are defined in the `cost.simulate` function. At present the function must be edited to change these. They are:

```
> nlsamp.land <- 100
> nasamp.land <- 2
> nlsamp.disc <- 30
> nasamp.disc <- 2
```

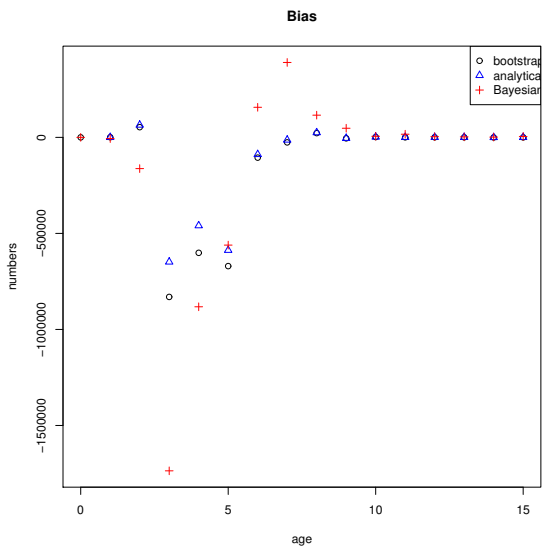
This means that 100 landing lengths are sampled within each size class. Of these 2 from each length class (combined over all size classes) are sampled. The equivalent values for discards are 30 and 2, where length sampling is within each haul, and age sample is from lengths combined over discards from all hauls.

Step 5 The simulated data are converted into COST format, and so can be analysed by any COST function.

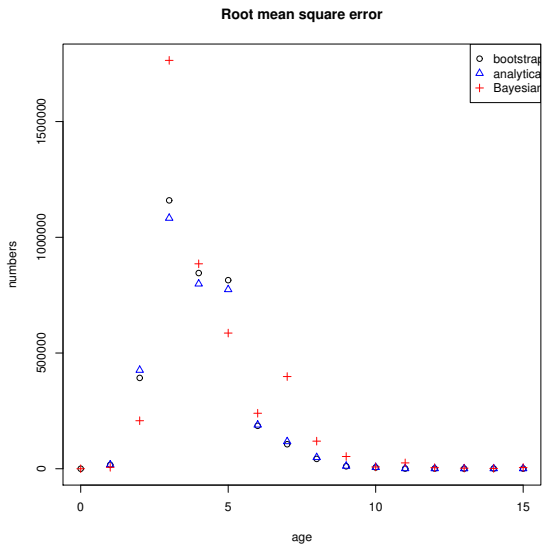
Step 6 For any of the parameters, the mean squared error, and the coverage of confidence intervals etc can be found.

There are as yet no formal methods for simulating data or summarising results, but the kind of plots that may be available are the following:

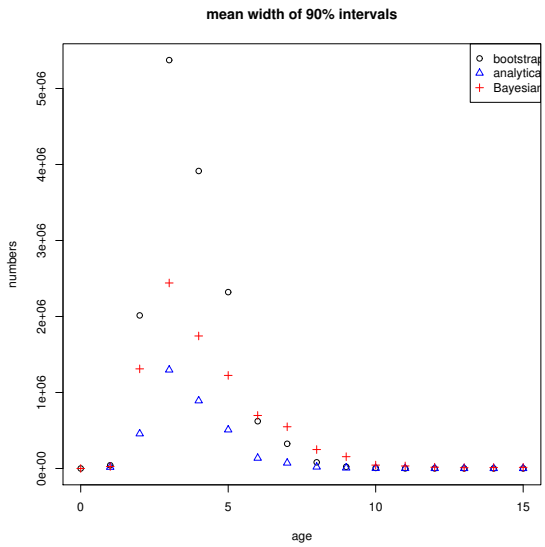
Bias : This shows the mean difference between the point estimates of parameters and the true values.



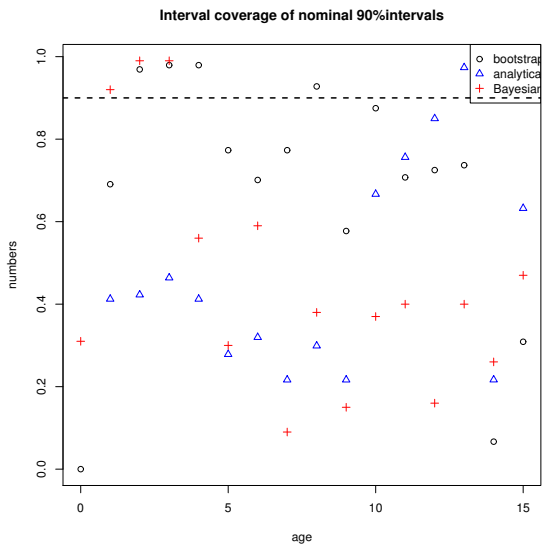
Root mean squared error : This is equal to the square root of (bias squared + variance)



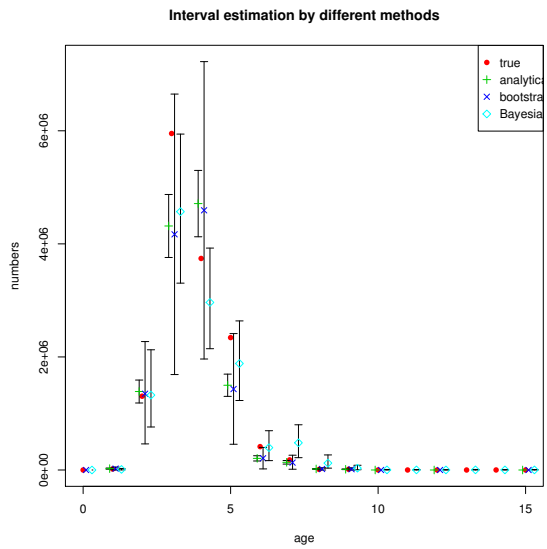
Mean width of intervals : Plotted for each age group



Coverage of intervals : For a nominal 90% interval (for example), this shows the estimated true coverage.



Also shown below is a plot of a typical result, with the true value and the means and 90bootstrap and analytical methods used the same simulated data set, but the Bayesian method used a different set.



Index

Symboles

+ 67

A

ageLenMulti 87
alkLgthRec 69

B

bioPar.plot 18
boxplot 14
bpBoot 78, 81
bpEstim 78, 84

C

ca table 4
ce table 6
ceData 3, 6, 7
ceDataCons 40
ceDataVal 23
cl table 6
clData 3, 6, 7
clDataCons 40
clDataVal 23
convert.icesarea.statsq 34
convert.lon.lat.statsq 34
convert.statsq.samplingarea 30
cost.simloop 104
cost.simulate 104
csData 3, 4, 7
csDataCons 40
csDataVal 23
csPlot.design 11
cstSummary 8

D

data 4, 24
dbeCalc 49, 52, 60, 71
dbecalc 64
dbeObject 43, 60, 63, 78, 81, 92
dbeObject.list 91
dbeOutput 95
dbePlot 52, 78
dbePlotRep 66, 67
DCRcvIndicator 59
delta.out 22
deltCalc 21, 23
dim 8, 12
disCorrPlot 47

E

Europe 34

F

finecoast 34
fit 91
FLQuant 97

G

GSAareas 36

H

head 8
hh table 4
hl table 4

I

ICESAreaRects 36
initial.byfit 99

L

landisVol 14, 16
lenDisPlot 14
lendisPlot 16
library 3
load 41

M

makeICdf 96
makeICfile 97
mergecsData 11

N

N_at_Age_direct 75
names 7
newcoast 34
newhl 11
NHcoast 34

P

plot 14
predict 91
predict.fit.COST 101
prediction.cells 94

R

RaiseAge 71
RaiseAgeBoot 73
RaiseLgth 60, 62, 67
RaiseLgthBoot 64
Raising Procedure Key 47
relativeValue 39
Report 42
run.bayes 90, 94

S

sampPar 14
save 41
setup 103
setup.data 104
sl table 4
slotNames 7
spacePlot 24, 25, 27, 29, 30, 33
spaceStrata 21
str 8, 43
stratAggreg 62, 65, 95
strDef 14
strIni 14, 21, 37, 38, 78
subset 12
subSetSpp 13
subsetSpp 12
subsetTarget 13

summary 8

T
tabConsist 37
tabconsist 12
tail 8
tapply 25
techStrata 21
text 26
timeStrata 21
totVolume 49, 52, 55, 57
tr table 4

U
update 16

V
viewGapsAlkCons 68

W
write.csv 99
write.table 99

X
xyplot 16

ANNEX I – Outcome examples of the reporting method

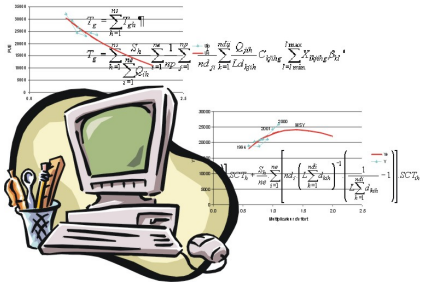
A – Reporting a Validated object

B – Reporting a dbeOutput object

C – Reporting a script of analysis

Sampling Information (status : validated data)

stock	Hake in GSA 09
year	2006
country	ITA



1 Information about Market Sampling in shore(landed/sold)

Space	Technical/Time	1	2	3	4
GSA9	GTR	NA	6	10	14
	OTB	3	NA	3	6

Table 1: Number of measured samples by national metier, quarter, zone

Space	Technical/Time	1	2	3	4
GSA9	GTR	NA	11270	7320	47010
	OTB	10600	NA	11200	40450

Table 2: Total measured weight by national metier, quarter, zone (g)

Space	Technical/Time	1	2	3	4
GSA9	GTR	3	17	18	33
	OTB	4	NA	4	8

Table 3: Number of sampled trips by national metier, quarter, zone

Space	Technical/Time	1	2	3	4
GSA9	GTR	NA	28	19	100
	OTB	79	NA	62	147

Table 4: Number of measured fishs by national metier, quarter, zone

2 Information about Biological Sampling

Space	Technical/Time	1	3	4
GSA9	all	55	54	88

Table 5: Number of sampled fish by quarter, zone (age)

Space	Technical/Time	1	3	4
GSA9	all	3286.9	NA	10032

Table 6: Total sampled weight by quarter, zone (age)

3 Information about Sampling at sea(caught/discarded)

Space	Technical/Time	1	2	3	4
GSA9	OTB	2	16	23	20

Table 7: Number of sampled trips by national metier, quarter, zone (at-sea sampling)

Space	Technical/Time	1	2	3	4
GSA9	OTB	5	28	58	44

Table 8: Number of sampled fish by national metier, quarter, zone

Space	Technical/Time	1	2	3	4
GSA9	OTB	990	NA	NA	NA

Table 9: Total fishing duration of sampled hauls by national metier, quarter, zone (mn)

Space	Technical/Time	1	2	3	4
GSA9	OTB	64	2831	3084	2203

Table 10: Number of measured fishs by national metier, quarter, zone (at-sea sampling)

List of Tables

1 Number of measured samples by national metier, quarter, zone 2
2 Total measured weight by national metier, quarter, zone (g) 2
3 Number of sampled trips by national metier, quarter, zone 2
4 Number of measured fishs by national metier, quarter, zone 2
5 Number of sampled fish by quarter, zone (age) 3
6 Total sampled weight by quarter, zone (age) 3
7 Number of sampled trips by national metier, quarter, zone (at-sea sampling) 4
8 Number of sampled fish by national metier, quarter, zone 4
9 Total fishing duration of sampled hauls by national metier, quarter, zone (mn) 4
10 Number of measured fishs by national metier, quarter, zone (at-sea sampling) 4

Merluccius merluccius estimate information

July 31, 2009

Desc	Output object
Species	Merluccius merluccius
catchCat	LAN
param	NA
time	NA
space	NA
technical	foCatNat

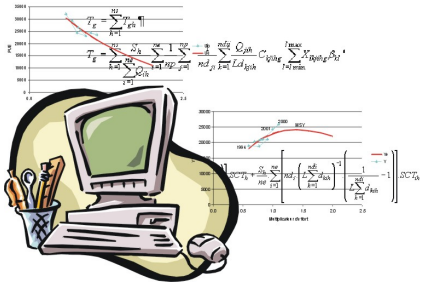


Table 1: Length estimates, intervals and cv

	Time	Space	Technical	Length	Estimates	inf	sup	variation	cv
1	all	all	GTR	150	879.97	0.00	3105.22	1289025.32	1.29
2	all	all	GTR	160	30452.19	0.00	105876.22	1480891845.29	1.26
3	all	all	GTR	190	29201.24	0.00	101968.17	1378389202.61	1.27
4	all	all	GTR	195	30081.21	0.00	102883.52	1379730055.88	1.23
5	all	all	GTR	200	32212.12	0.00	107772.61	1486255258.38	1.20
6	all	all	GTR	215	31332.15	0.00	106296.23	1462885049.77	1.22
7	all	all	GTR	220	30452.19	0.00	105381.90	1461544196.49	1.26
8	all	all	GTR	230	1759.93	0.00	6172.66	5068956.93	1.28
9	all	all	GTR	245	40200.82	0.00	113048.00	1381431469.84	0.92
10	all	all	GTR	250	5939.77	4495.58	7383.97	542943.18	0.12
11	all	all	GTR	255	5059.81	2956.41	7163.21	1151722.29	0.21
12	all	all	GTR	260	16059.38	10913.59	21205.17	6892998.20	0.16
13	all	all	GTR	265	5059.81	2956.41	7163.21	1151722.29	0.21
14	all	all	GTR	270	11879.54	8298.98	15460.11	3337385.70	0.15
15	all	all	GTR	275	5939.77	4110.07	7769.48	871497.44	0.16
16	all	all	GTR	280	66982.16	0.00	209348.13	5276139015.35	1.08
17	all	all	GTR	285	73942.74	13475.17	134410.30	951806756.75	0.42
18	all	all	GTR	290	121706.48	1201.40	242211.56	3780197977.39	0.51
19	all	all	GTR	295	5939.77	4110.07	7769.48	871497.44	0.16
20	all	all	GTR	300	182840.99	46817.12	318864.85	4816527343.61	0.38
21	all	all	GTR	305	16059.38	10913.59	21205.17	6892998.20	0.16
22	all	all	GTR	310	132946.33	32318.24	233574.43	2635981197.69	0.39
23	all	all	GTR	320	154798.56	32954.82	276642.31	3864651133.40	0.40
24	all	all	GTR	325	68002.97	7608.06	128397.87	949520647.73	0.45
25	all	all	GTR	330	63163.15	2041.07	124285.23	972523354.28	0.49
26	all	all	GTR	335	10999.58	8144.42	13854.74	2122091.72	0.13
27	all	all	GTR	340	30081.21	0.00	84532.64	771831252.12	0.92
28	all	all	GTR	345	87964.56	76700.98	99228.15	33026065.89	0.07
29	all	all	GTR	350	219614.06	36320.75	402907.38	8745750003.21	0.43
30	all	all	GTR	355	144967.95	77560.58	212375.33	1182819913.84	0.24
31	all	all	GTR	360	8579.67	3409.51	13749.83	6958450.34	0.31
32	all	all	GTR	365	68002.97	7608.06	128397.87	949520647.73	0.45
33	all	all	GTR	370	186409.57	5644.15	367175.00	8506179673.70	0.49
34	all	all	GTR	375	29201.24	0.00	83606.58	770525256.58	0.95
35	all	all	GTR	380	4399.83	0.00	13344.50	20827267.01	1.04
36	all	all	GTR	385	5059.81	2956.41	7163.21	1151722.29	0.21
37	all	all	GTR	390	37271.92	19637.32	54906.53	80953436.48	0.24
38	all	all	GTR	395	63462.29	0.00	172293.30	3083252748.63	0.87
39	all	all	GTR	400	1759.93	0.00	5418.40	3484191.11	1.06
40	all	all	GTR	405	30081.21	0.00	84531.27	771792495.81	0.92
41	all	all	GTR	410	29201.24	0.00	83606.58	770525256.58	0.95
42	all	all	GTR	415	5059.81	2956.41	7163.21	1151722.29	0.21
43	all	all	GTR	420	38151.89	20233.43	56070.35	83580544.41	0.24
44	all	all	GTR	430	30081.21	0.00	84532.99	771841170.52	0.92
45	all	all	GTR	440	32212.12	14346.60	50077.64	83087413.95	0.28
46	all	all	GTR	445	1759.93	0.00	5469.16	3581540.52	1.08
47	all	all	GTR	465	879.97	0.00	3105.22	1289025.32	1.29
48	all	all	GTR	490	64713.24	7529.08	121897.39	851246261.29	0.45
49	all	all	GTR	500	1759.93	0.00	5463.45	3570532.81	1.07
50	all	all	GTR	510	30452.19	12968.99	47935.38	79569282.42	0.29
51	all	all	GTR	535	5059.81	2956.41	7163.21	1151722.29	0.21
52	all	all	GTR	545	5059.81	2956.41	7163.21	1151722.29	0.21

Continued on next page

	Time	Space	Technical	Length	Estimates	inf	sup	variation	cv
53	all	all	GTR	550	5059.81	2956.41	7163.21	1151722.29	0.21
54	all	all	GTR	560	5059.81	2956.41	7163.21	1151722.29	0.21
55	all	all	GTR	570	5059.81	2956.41	7163.21	1151722.29	0.21
56	all	all	GTR	585	5059.81	2956.41	7163.21	1151722.29	0.21
57	all	all	GTR	600	879.97	0.00	3105.22	1289025.32	1.29
58	all	all	GTR	610	5059.81	2956.41	7163.21	1151722.29	0.21
59	all	all	OTB	70	15809.09	0.00	50074.96	305652175.79	1.11
60	all	all	OTB	75	21078.78	0.00	66766.62	543381645.84	1.11
61	all	all	OTB	80	31618.18	0.00	100149.93	1222608703.14	1.11
62	all	all	OTB	85	97489.38	0.00	308795.61	11623273018.07	1.11
63	all	all	OTB	90	113298.47	0.00	358870.58	15698635361.88	1.11
64	all	all	OTB	95	110663.62	0.00	350524.75	14976956613.50	1.11
65	all	all	OTB	100	118568.16	0.00	375562.23	17192934887.95	1.11
66	all	all	OTB	105	110663.62	0.00	350524.75	14976956613.50	1.11
67	all	all	OTB	110	68521.81	0.00	202965.60	4705278891.69	1.00
68	all	all	OTB	115	78287.15	0.00	227488.32	5794931139.27	0.97
69	all	all	OTB	120	164742.84	0.00	435848.13	19132856784.73	0.84
70	all	all	OTB	125	117145.08	0.00	264677.71	5666044684.38	0.64
71	all	all	OTB	130	117122.94	47106.62	187139.25	1276151651.59	0.31
72	all	all	OTB	135	207203.64	100871.38	313535.90	2943295764.35	0.26
73	all	all	OTB	140	296651.07	147152.03	446150.11	5818092787.51	0.26
74	all	all	OTB	145	280523.05	155222.86	405823.24	4087024708.91	0.23
75	all	all	OTB	150	400290.37	275682.50	524898.24	4041985644.00	0.16
76	all	all	OTB	155	732871.84	567815.64	897928.03	7091979292.87	0.11
77	all	all	OTB	160	1172843.14	848288.52	1497397.77	27420755923.31	0.14
78	all	all	OTB	165	1249495.43	1011776.21	1487214.64	14710668174.44	0.10
79	all	all	OTB	170	1483384.58	1013539.14	1953230.01	57466380026.18	0.16
80	all	all	OTB	175	1351250.06	839688.47	1862811.65	68123927533.36	0.19
81	all	all	OTB	180	1507035.94	1080065.63	1934006.25	47456878995.82	0.14
82	all	all	OTB	185	985518.44	751615.11	1219421.76	14242184566.55	0.12
83	all	all	OTB	190	1101220.69	781636.39	1420804.98	26587326854.70	0.15
84	all	all	OTB	195	840550.20	583689.90	1097410.51	17175041663.98	0.16
85	all	all	OTB	200	679905.73	545387.99	814423.46	4710455690.12	0.10
86	all	all	OTB	205	662116.05	496228.61	828003.48	7163591195.25	0.13
87	all	all	OTB	210	561455.39	455267.30	667643.48	2935320095.90	0.10
88	all	all	OTB	215	420583.91	286973.07	554194.75	4647155723.01	0.16
89	all	all	OTB	220	420856.23	300699.09	541013.36	3758399512.55	0.15
90	all	all	OTB	225	259551.95	199752.53	319351.38	930888927.15	0.12
91	all	all	OTB	230	306612.89	200834.08	412391.70	2912736403.31	0.18
92	all	all	OTB	235	248894.20	156347.38	341441.03	2229599451.85	0.19
93	all	all	OTB	240	185424.26	100323.45	270525.07	1885259932.58	0.23
94	all	all	OTB	245	188714.78	143996.63	233432.93	520560911.27	0.12
95	all	all	OTB	250	168574.77	106274.43	230875.11	1010379846.44	0.19
96	all	all	OTB	255	84347.10	46647.16	122047.03	369985753.60	0.23
97	all	all	OTB	260	116966.55	72688.99	161244.10	510353495.39	0.19
98	all	all	OTB	265	84497.14	60322.11	108672.16	152137989.93	0.15
99	all	all	OTB	270	82465.89	59995.48	104936.29	131439433.63	0.14
100	all	all	OTB	275	39076.59	20435.62	57717.56	90456709.97	0.24
101	all	all	OTB	280	47789.97	31878.77	63701.17	65903680.54	0.17
102	all	all	OTB	285	39599.81	27634.35	51565.28	37270312.75	0.15
103	all	all	OTB	290	35481.83	21116.32	49847.34	53721230.26	0.21
104	all	all	OTB	295	33976.56	17549.35	50403.76	70247576.65	0.25
105	all	all	OTB	300	35347.56	18238.96	52456.16	76196090.55	0.25

Continued on next page

	Time	Space	Technical	Length	Estimates	inf	sup	variation	cv
106	all	all	OTB	305	34091.48	15215.38	52967.59	92753120.97	0.28
107	all	all	OTB	310	45588.46	23177.24	67999.67	130747872.32	0.25
108	all	all	OTB	315	16522.63	8869.44	24175.83	15247181.41	0.24
109	all	all	OTB	320	15948.18	6643.55	25252.81	22537328.07	0.30
110	all	all	OTB	325	10576.32	1473.08	19679.56	21572276.59	0.44
111	all	all	OTB	330	25153.88	7569.72	42738.05	80491028.29	0.36
112	all	all	OTB	335	8279.34	0.00	18567.03	27551126.10	0.63
113	all	all	OTB	340	17147.65	5368.67	28926.63	36117638.30	0.35
114	all	all	OTB	345	2461.60	296.97	4626.23	1219753.26	0.45
115	all	all	OTB	350	19182.12	4976.90	33387.33	52529047.11	0.38
116	all	all	OTB	355	11104.19	4271.34	17937.04	12153671.68	0.31
117	all	all	OTB	360	8090.83	0.00	18253.65	26886383.46	0.64
118	all	all	OTB	365	3663.73	0.00	8538.60	6186274.37	0.68
119	all	all	OTB	370	7950.22	0.00	17735.24	24924528.65	0.63
120	all	all	OTB	375	4237.39	0.00	9904.92	8361655.57	0.68
121	all	all	OTB	380	5249.45	1631.81	8867.09	3406860.46	0.35
122	all	all	OTB	385	7287.46	0.00	17274.52	25964488.12	0.70
123	all	all	OTB	390	8091.02	554.82	15627.23	14784597.95	0.48
124	all	all	OTB	395	419.92	0.00	1294.26	199004.57	1.06
125	all	all	OTB	400	2345.78	0.00	5516.24	2616681.55	0.69
126	all	all	OTB	405	8822.35	0.00	19578.58	30117822.71	0.62
127	all	all	OTB	410	4058.33	0.00	9481.74	7656801.91	0.68
128	all	all	OTB	420	1172.89	0.00	3374.62	1261921.11	0.96
129	all	all	OTB	425	8507.78	0.00	27142.34	90394487.78	1.12
130	all	all	OTB	430	2070.52	0.00	4605.48	1672815.83	0.62
131	all	all	OTB	435	839.84	0.00	2568.77	778141.46	1.05
132	all	all	OTB	445	1267.68	0.00	3873.80	1768047.07	1.05
133	all	all	OTB	455	3334.32	0.00	7052.72	3599276.51	0.57
134	all	all	OTB	470	1592.81	0.00	4104.08	1641687.99	0.80
135	all	all	OTB	475	1172.89	0.00	3588.90	1519508.69	1.05
136	all	all	OTB	480	419.92	0.00	1284.38	194535.36	1.05
137	all	all	OTB	485	539.67	0.00	1616.56	301887.39	1.02
138	all	all	OTB	500	1172.89	0.00	3588.90	1519508.69	1.05
139	all	all	OTB	505	419.92	0.00	1284.38	194535.36	1.05
140	all	all	OTB	520	419.92	0.00	1284.38	194535.36	1.05
141	all	all	OTB	530	9137.66	0.00	27816.95	90829003.04	1.04
142	all	all	OTB	540	4253.89	0.00	13571.17	22598621.95	1.12
143	all	all	OTB	560	4253.89	0.00	13571.17	22598621.95	1.12
144	all	all	OTB	580	4253.89	0.00	13571.17	22598621.95	1.12
145	all	all	OTB	600	781.93	0.00	2413.26	692774.11	1.06
146	all	all	OTB	610	2733.40	346.06	5120.74	1483656.96	0.45
147	all	all	OTB	660	839.84	0.00	2104.08	416068.76	0.77
148	all	all	OTB	680	930.63	0.00	2317.47	500670.60	0.76
149	all	all	OTB	685	1172.89	0.00	3530.68	1447152.63	1.03
150	all	all	OTB	690	419.92	0.00	1284.38	194535.36	1.05
151	all	all	OTB	700	930.63	0.00	2317.47	500670.60	0.76
152	all	all	OTB	730	1172.89	0.00	3530.68	1447152.63	1.03

Table 2: Number of samples and mesured fish length by strata

	Time	Space	Technical	nSampling	nMeas
1	all	all	GTR	20	176.00
2	all	all	OTB	120	8724.00

List of Tables

1 Length estimates, intervals and cv 2
2 Number of samples and mesured fish length by strata 4

script.r

31 juillet 2009

```
>  # Overview of COST available methods
> # for exploratory data analysis
> # by an example on Market Sampling in Mediterranean sea
> #
> # COST team
> # 18 May 2009
> #
> # Versions used
> # -------------
> # R 2.8.1
> # COSTcore 1.3-3
> # COSTeda 1.3-1
> # COSTdbe 1.2
> library(COSTcore)
> library(COSTeda)
> library(COSTdbe)
> library(COSTdata)
> # ---------- A : IMPORTING THE DATASET --------------------
> #
> ## First set up a directory where the cvs files are to be created
> ## e.g. in a directory called "COST/Examples" on the C drive
> ## then specify the path to this and set it as the working directory for this session
> path <- (".")      # This is 'My documents' folder
> setwd(path)
> # ---- A.: Creating the raw dataset  -----
> #
> # This dataset is already in COSTdata package, and the right COST format
> data(SIBM2006cs)
> #Mediterranean dataset is loaded in the R session
> ls()

 [1] "area.list"     "ca"            "ce"            "cl"            "cov.list"
 [6] "example.data"  "fieldCorr"     "fieldTitle"    "gear.list"     "hh"
[11] "hl"            "l.int"         "landings"      "path"          "pred.cov.real"
[16] "seas.list"     "SIBM_NLa"      "SIBM2006cl"    "SIBM2006cs"    "SIBMconsCL"
[21] "SIBMconsCS"    "SIBMval"       "sl"            "species"       "strD"
[26] "timeStrata"    "tr"

> # ---------- B : VERIFYING THE CONSISTENCY --------------------
> #
> #
> head(SIBM2006cs)  # you see the raw data object, i.e. an object of class "csData"

An object of class Ş̆csDataŤ̆
Slot "desc":
[1] "Unknown stock"

Slot "tr":
  sampType landCtry vslFlgCtry year     proj trpCode vslLen vslPwr vslSize vslType
1        M      ITA        ITA 2006 CAMPBIOL       1     NA     NA      NA       1
2        M      ITA        ITA 2006 CAMPBIOL       2     NA     NA      NA       1
3        M      ITA        ITA 2006 CAMPBIOL       3     NA     NA      NA       1
```

```
4        M      ITA        ITA 2006 CAMPBIOL       4     21    299      53       1
5        S      ITA        ITA 2006 CAMPBIOL       5     NA     NA      NA       1
6        M      ITA        ITA 2006 CAMPBIOL       6      8     31       3       4
  harbour foNum daysAtSea vslId sampCtry sampMeth
1    <NA>     2         1    14      ITA Observer
2    <NA>     2         1     7      ITA Observer
3    <NA>     2         1     7      ITA Observer
4    <NA>     5         1     3      ITA Observer
5    <NA>     2         1    15      ITA Observer
6    <NA>     1         1    17      ITA Observer

Slot "hh":
  sampType landCtry vslFlgCtry year     proj trpCode staNum foVal aggLev catReg sppReg
1        M      ITA        ITA 2006 CAMPBIOL       1    999     V      T    Lan    Par
2        M      ITA        ITA 2006 CAMPBIOL       2    999     V      T    Lan    Par
3        M      ITA        ITA 2006 CAMPBIOL       3    999     V      T    Lan    Par
4        M      ITA        ITA 2006 CAMPBIOL       4    999     V      T    Lan    All
5        S      ITA        ITA 2006 CAMPBIOL       5      1     V      H    All    All
6        S      ITA        ITA 2006 CAMPBIOL       5      2     V      H    All    All
        date        time foDur   latIni   lonIni   latFin   lonFin area rect subRect
1 2006-01-17        <NA>    NA       NA       NA       NA       NA GSA9   99    <NA>
2 2006-02-13        <NA>    NA       NA       NA       NA       NA GSA9   99    <NA>
3 2006-03-10        <NA>    NA       NA       NA       NA       NA GSA9   99    <NA>
4 2006-03-14        <NA>     0       NA       NA       NA       NA GSA9   99    <NA>
5 2006-03-20 0.229166667   285 41.73067 11.98167 41.53333 11.13333 GSA9   99    <NA>
6 2006-03-20 0.458333333   225 42.55950 12.09000 42.70000 11.93750 GSA9   99    <NA>
  foDep waterDep foCatNat                              foCatEu5       foCatEu6 meshSize
1    NA       NA      OTB Mixed demersal and deep-water species OTB_DEF_40_0_0       40
2    NA       NA      OTB Mixed demersal and deep-water species OTB_DEF_40_0_0       40
3    NA       NA      OTB Mixed demersal and deep-water species OTB_DEF_40_0_0       40
4    NA       NA      OTB Mixed demersal and deep-water species OTB_DEF_40_0_0       40
5   310      310      OTB Mixed demersal and deep-water species OTB_DEF_40_0_0       40
6   570      570      OTB Mixed demersal and deep-water species OTB_DEF_40_0_0       40
  selDev meshSizeSelDev
1      0              0
2      0              0
3      0              0
4      0              0
5      0              0
6      0              0

Slot "sl":
  sampType landCtry vslFlgCtry year     proj trpCode staNum                   spp
1        M      ITA        ITA 2006 CAMPBIOL       1    999   Aristeus antennatus
2        M      ITA        ITA 2006 CAMPBIOL       1    999   Aristeus antennatus
3        M      ITA        ITA 2006 CAMPBIOL       1    999      Eledone moschata
4        M      ITA        ITA 2006 CAMPBIOL       1    999 Merluccius merluccius
5        M      ITA        ITA 2006 CAMPBIOL       1    999   Nephrops norvegicus
6        M      ITA        ITA 2006 CAMPBIOL       2    999      Eledone cirrhosa
  catchCat landCat commCatScl  commCat subSampCat  sex    wt subSampWt lenCode
1      LAN     HUC      Local UNSORTED          0 <NA>  4000         0      mm
2      LAN     HUC      Local UNSORTED          2 <NA> 25000      5000      mm
3      LAN     HUC      Local UNSORTED          0 <NA> 12000         0     scm
4      LAN     HUC      Local UNSORTED          0 <NA>  4000         0     scm
5      LAN     HUC      Local UNSORTED          0 <NA>  8000         0      mm
6      LAN     HUC      Local UNSORTED          2 <NA>  8500      1500     scm

Slot "hl":
  sampType landCtry vslFlgCtry year     proj trpCode staNum                 spp catchCat
1        M      ITA        ITA 2006 CAMPBIOL       1    999 Aristeus antennatus      LAN
2        M      ITA        ITA 2006 CAMPBIOL       1    999 Aristeus antennatus      LAN
3        M      ITA        ITA 2006 CAMPBIOL       1    999 Aristeus antennatus      LAN
```

```
4        M      ITA        ITA 2006 CAMPBIOL       1    999 Aristeus antennatus      LAN
5        M      ITA        ITA 2006 CAMPBIOL       1    999 Aristeus antennatus      LAN
6        M      ITA        ITA 2006 CAMPBIOL       1    999 Aristeus antennatus      LAN
  landCat commCatScl  commCat subSampCat     sex lenCls lenNum
1     HUC      Local UNSORTED          2 UNSEXED     27      1
2     HUC      Local UNSORTED          2 UNSEXED     29      1
3     HUC      Local UNSORTED          2 UNSEXED     30      1
4     HUC      Local UNSORTED          2 UNSEXED     31      2
5     HUC      Local UNSORTED          2 UNSEXED     32      1
6     HUC      Local UNSORTED          2 UNSEXED     33      2

Slot "ca":
  sampType landCtry vslFlgCtry year     proj trpCode staNum quarter month
1        M      ITA        ITA 2006 CAMPBIOL     117    999       4    11
2        M      ITA        ITA 2006 CAMPBIOL     117    999       4    11
3        M      ITA        ITA 2006 CAMPBIOL     117    999       4    11
4        M      ITA        ITA 2006 CAMPBIOL     117    999       4    11
5        M      ITA        ITA 2006 CAMPBIOL     117    999       4    11
6        M      ITA        ITA 2006 CAMPBIOL     117    999       4    11
                    spp sex catchCat landCat commCatScl  commCat         stock area rect
1 Merluccius merluccius   M      LAN     HUC      Local UNSORTED Mediterranean GSA9   99
2 Merluccius merluccius   M      LAN     HUC      Local UNSORTED Mediterranean GSA9   99
3 Merluccius merluccius   M      LAN     HUC      Local UNSORTED Mediterranean GSA9   99
4 Merluccius merluccius   M      LAN     HUC      Local UNSORTED Mediterranean GSA9   99
5 Merluccius merluccius   M      LAN     HUC      Local UNSORTED Mediterranean GSA9   99
6 Merluccius merluccius   M      LAN     HUC      Local UNSORTED Mediterranean GSA9   99
  subRect lenCls age fishId lenCode ageMeth plusGrp otoWt otoSide indWt matMeth
1    <NA>    175   1      1     scm    <NA>       -     0       L    40    <NA>
2    <NA>    215   2      2     scm    <NA>       -     0       L    68    <NA>
3    <NA>    200   2      3     scm    <NA>       -     0       L    58    <NA>
4    <NA>    290   3      4     scm    <NA>       -     0       L   168    <NA>
5    <NA>    310   3      5     scm    <NA>       -     0       L   226    <NA>
6    <NA>    270   3      6     scm    <NA>       -     0       L   142    <NA>
            matScale matStage
1 Holden & Raitt 1-5        1
2 Holden & Raitt 1-5        1
3 Holden & Raitt 1-5        1
4 Holden & Raitt 1-5        1
5 Holden & Raitt 1-5        1
6 Holden & Raitt 1-5        1
```

> str(SIBM2006cs) ## the structure of the object

```
Formal class 'csData' [package "COSTcore"] with 6 slots
  ..@ desc: chr "Unknown stock"
  ..@ tr  :'data.frame':        148 obs. of  16 variables:
  .. ..$ sampType  : chr [1:148] "M" "M" "M" "M" ...
  .. ..$ landCtry  : chr [1:148] "ITA" "ITA" "ITA" "ITA" ...
  .. ..$ vslFlgCtry: chr [1:148] "ITA" "ITA" "ITA" "ITA" ...
  .. ..$ year      : int [1:148] 2006 2006 2006 2006 2006 2006 2006 2006 2006 2006 ...
  .. ..$ proj      : chr [1:148] "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" ...
  .. ..$ trpCode   : chr [1:148] "1" "2" "3" "4" ...
  .. ..$ vslLen    : int [1:148] NA NA NA 21 NA 8 0 8 8 5 ...
  .. ..$ vslPwr    : int [1:148] NA NA NA 299 NA 31 100 31 31 19 ...
  .. ..$ vslSize   : int [1:148] NA NA NA 53 NA 3 50 3 3 1 ...
  .. ..$ vslType   : chr [1:148] "1" "1" "1" "1" ...
  .. ..$ harbour   : chr [1:148] NA NA NA NA ...
  .. ..$ foNum     : int [1:148] 2 2 2 5 2 1 3 1 1 1 ...
  .. ..$ daysAtSea : int [1:148] 1 1 1 1 1 1 1 1 1 1 ...
  .. ..$ vslId     : int [1:148] 14 7 7 3 15 17 2 17 17 22 ...
  .. ..$ sampCtry  : chr [1:148] "ITA" "ITA" "ITA" "ITA" ...
  .. ..$ sampMeth  : chr [1:148] "Observer" "Observer" "Observer" "Observer" ...
  ..@ hh  :'data.frame':        222 obs. of  29 variables:
```

```
.. ..$ sampType      : chr [1:222] "M" "M" "M" "M" ...
.. ..$ landCtry      : chr [1:222] "ITA" "ITA" "ITA" "ITA" ...
.. ..$ vslFlgCtry    : chr [1:222] "ITA" "ITA" "ITA" "ITA" ...
.. ..$ year          : int [1:222] 2006 2006 2006 2006 2006 2006 2006 2006 2006 2006 ...
.. ..$ proj          : chr [1:222] "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" ...
.. ..$ trpCode       : chr [1:222] "1" "2" "3" "4" ...
.. ..$ staNum        : int [1:222] 999 999 999 999 1 2 999 1 2 3 ...
.. ..$ foVal         : chr [1:222] "V" "V" "V" "V" ...
.. ..$ aggLev        : chr [1:222] "T" "T" "T" "T" ...
.. ..$ catReg        : chr [1:222] "Lan" "Lan" "Lan" "Lan" ...
.. ..$ sppReg        : chr [1:222] "Par" "Par" "Par" "All" ...
.. ..$ date          : chr [1:222] "2006-01-17" "2006-02-13" "2006-03-10" "2006-03-14" ...
.. ..$ time          : chr [1:222] NA NA NA NA ...
.. ..$ foDur         : int [1:222] NA NA NA 0 285 225 720 150 130 200 ...
.. ..$ latIni        : num [1:222] NA NA NA NA 41.7 ...
.. ..$ lonIni        : num [1:222] NA NA NA NA 12 ...
.. ..$ latFin        : num [1:222] NA NA NA NA 41.5 ...
.. ..$ lonFin        : num [1:222] NA NA NA NA 11.1 ...
.. ..$ area          : chr [1:222] "GSA9" "GSA9" "GSA9" "GSA9" ...
.. ..$ rect          : chr [1:222] "99" "99" "99" "99" ...
.. ..$ subRect       : chr [1:222] NA NA NA NA ...
.. ..$ foDep         : int [1:222] NA NA NA NA 310 570 NA NA NA NA ...
.. ..$ waterDep      : int [1:222] NA NA NA NA 310 570 NA NA NA NA ...
.. ..$ foCatNat      : chr [1:222] "OTB" "OTB" "OTB" "OTB" ...
.. ..$ foCatEu5      : chr [1:222] "Mixed demersal and deep-water species" "Mixed demersal and deep-wa
.. ..$ foCatEu6      : chr [1:222] "OTB_DEF_40_0_0" "OTB_DEF_40_0_0" "OTB_DEF_40_0_0" "OTB_DEF_40_0_0
.. ..$ meshSize      : int [1:222] 40 40 40 40 40 40 22 40 40 40 ...
.. ..$ selDev        : chr [1:222] "0" "0" "0" "0" ...
.. ..$ meshSizeSelDev: int [1:222] 0 0 0 0 0 0 0 0 0 0 ...
..@ sl  :'data.frame':        5250 obs. of  17 variables:
.. ..$ sampType  : chr [1:5250] "M" "M" "M" "M" ...
.. ..$ landCtry  : chr [1:5250] "ITA" "ITA" "ITA" "ITA" ...
.. ..$ vslFlgCtry: chr [1:5250] "ITA" "ITA" "ITA" "ITA" ...
.. ..$ year      : int [1:5250] 2006 2006 2006 2006 2006 2006 2006 2006 2006 2006 ...
.. ..$ proj      : chr [1:5250] "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" ...
.. ..$ trpCode   : chr [1:5250] "1" "1" "1" "1" ...
.. ..$ staNum    : int [1:5250] 999 999 999 999 999 999 999 999 999 999 ...
.. ..$ spp       : chr [1:5250] "Aristeus antennatus" "Aristeus antennatus" "Eledone moschata" "Merlu
.. ..$ catchCat  : chr [1:5250] "LAN" "LAN" "LAN" "LAN" ...
.. ..$ landCat   : chr [1:5250] "HUC" "HUC" "HUC" "HUC" ...
.. ..$ commCatScl: chr [1:5250] "Local" "Local" "Local" "Local" ...
.. ..$ commCat   : chr [1:5250] "UNSORTED" "UNSORTED" "UNSORTED" "UNSORTED" ...
.. ..$ subSampCat: chr [1:5250] "0" "2" "0" "0" ...
.. ..$ sex       : chr [1:5250] NA NA NA NA ...
.. ..$ wt        : int [1:5250] 4000 25000 12000 4000 8000 8500 21000 1000 3500 5000 ...
.. ..$ subSampWt : int [1:5250] 0 5000 0 0 0 1500 0 0 500 0 ...
.. ..$ lenCode   : chr [1:5250] "mm" "mm" "scm" "scm" ...
..@ hl  :'data.frame':        8385 obs. of  16 variables:
.. ..$ sampType  : chr [1:8385] "M" "M" "M" "M" ...
.. ..$ landCtry  : chr [1:8385] "ITA" "ITA" "ITA" "ITA" ...
.. ..$ vslFlgCtry: chr [1:8385] "ITA" "ITA" "ITA" "ITA" ...
.. ..$ year      : int [1:8385] 2006 2006 2006 2006 2006 2006 2006 2006 2006 2006 ...
.. ..$ proj      : chr [1:8385] "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" ...
.. ..$ trpCode   : chr [1:8385] "1" "1" "1" "1" ...
.. ..$ staNum    : int [1:8385] 999 999 999 999 999 999 999 999 999 999 ...
.. ..$ spp       : chr [1:8385] "Aristeus antennatus" "Aristeus antennatus" "Aristeus antennatus" "Ar
.. ..$ catchCat  : chr [1:8385] "LAN" "LAN" "LAN" "LAN" ...
.. ..$ landCat   : chr [1:8385] "HUC" "HUC" "HUC" "HUC" ...
.. ..$ commCatScl: chr [1:8385] "Local" "Local" "Local" "Local" ...
.. ..$ commCat   : chr [1:8385] "UNSORTED" "UNSORTED" "UNSORTED" "UNSORTED" ...
.. ..$ subSampCat: chr [1:8385] "2" "2" "2" "2" ...
.. ..$ sex       : chr [1:8385] "UNSEXED" "UNSEXED" "UNSEXED" "UNSEXED" ...
```

```
  .. ..$ lenCls    : int [1:8385] 27 29 30 31 32 33 34 35 37 38 ...
  .. ..$ lenNum    : int [1:8385] 1 1 1 2 1 2 3 2 1 4 ...
  ..@ ca  :'data.frame':        566 obs. of  31 variables:
  .. ..$ sampType  : chr [1:566] "M" "M" "M" "M" ...
  .. ..$ landCtry  : chr [1:566] "ITA" "ITA" "ITA" "ITA" ...
  .. ..$ vslFlgCtry: chr [1:566] "ITA" "ITA" "ITA" "ITA" ...
  .. ..$ year      : int [1:566] 2006 2006 2006 2006 2006 2006 2006 2006 2006 2006 ...
  .. ..$ proj      : chr [1:566] "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" "CAMPBIOL" ...
  .. ..$ trpCode   : chr [1:566] "117" "117" "117" "117" ...
  .. ..$ staNum    : int [1:566] 999 999 999 999 999 999 999 999 999 999 ...
  .. ..$ quarter   : int [1:566] 4 4 4 4 4 4 4 4 4 4 ...
  .. ..$ month     : int [1:566] 11 11 11 11 11 11 11 11 11 11 ...
  .. ..$ spp       : chr [1:566] "Merluccius merluccius" "Merluccius merluccius" "Merluccius merluccius
  .. ..$ sex       : chr [1:566] "M" "M" "M" "M" ...
  .. ..$ catchCat  : chr [1:566] "LAN" "LAN" "LAN" "LAN" ...
  .. ..$ landCat   : chr [1:566] "HUC" "HUC" "HUC" "HUC" ...
  .. ..$ commCatScl: chr [1:566] "Local" "Local" "Local" "Local" ...
  .. ..$ commCat   : chr [1:566] "UNSORTED" "UNSORTED" "UNSORTED" "UNSORTED" ...
  .. ..$ stock     : chr [1:566] "Mediterranean" "Mediterranean" "Mediterranean" "Mediterranean" ...
  .. ..$ area      : chr [1:566] "GSA9" "GSA9" "GSA9" "GSA9" ...
  .. ..$ rect      : chr [1:566] "99" "99" "99" "99" ...
  .. ..$ subRect   : chr [1:566] NA NA NA NA ...
  .. ..$ lenCls    : int [1:566] 175 215 200 290 310 270 315 245 265 350 ...
  .. ..$ age       : int [1:566] 1 2 2 3 3 3 3 2 2 4 ...
  .. ..$ fishId    : int [1:566] 1 2 3 4 5 6 7 8 9 10 ...
  .. ..$ lenCode   : chr [1:566] "scm" "scm" "scm" "scm" ...
  .. ..$ ageMeth   : chr [1:566] NA NA NA NA ...
  .. ..$ plusGrp   : chr [1:566] "-" "-" "-" "-" ...
  .. ..$ otoWt     : num [1:566] 0 0 0 0 0 0 0 0 0 0 ...
  .. ..$ otoSide   : chr [1:566] "L" "L" "L" "L" ...
  .. ..$ indWt     : num [1:566] 40 68 58 168 226 142 210 84 112 272 ...
  .. ..$ matMeth   : chr [1:566] NA NA NA NA ...
  .. ..$ matScale  : chr [1:566] "Holden & Raitt 1-5" "Holden & Raitt 1-5" "Holden & Raitt 1-5" "Holden
  .. ..$ matStage  : chr [1:566] "1" "1" "1" "1" ...
```

```
> cstSummary(SIBM2006cs, tab = "sl")   # we see several species sampled, mainly 'Merluccius merluccius'
```

```
 sampType landCtry   vslFlgCtry   year           proj         trpCode     staNum
 S:4707   ITA:5250   ITA:5250   2006:5250   CAMPBIOL:5250   51     : 269   1  :2236
 M: 543                                                    130    : 213   2  :1300
                                                           46     : 199   3  : 669
                                                           128    : 194   999: 543
                                                           108    : 180   4  : 271
                                                           40     : 166   5  : 143
                                                           49     : 160   6  :  45
                                                           58     : 153   8  :  23
                                                           37     : 141   7  :  20
                                                           133    : 126
                                                           50     : 126
                                                           74     : 126
                                                           44     : 114
                                                           39     : 111
                                                           142    : 109
                                                           47     : 105
                                                           7      : 102
                                                           48     : 101
                                                           109    : 100
                                                           (Other):2455
                            spp       catchCat   landCat    commCatScl       commCat
 Merluccius merluccius        : 308   DIS:2641   DIS:2641   Local:5250   DIS     :2641
 Mullus barbatus              : 194   LAN:2609   HUC:2609                UNSORTED:2609
 Pagellus erythrinus          : 151
 Arnoglossus laterna          : 113
```

```
 Squilla mantis               : 113
 Eledone cirrhosa             : 110
 Parapenaeus longirostris     : 104
 Trachurus mediterraneus      : 103
 Citharus linguatula          :  89
 Octopus vulgaris             :  89
 Conger conger                :  80
 Engraulis encrasicolus       :  74
 Mullus surmuletus            :  74
 Sardina pilchardus           :  67
 Cepola  macrophthalma        :  66
 Trisopterus minutus capelanus:  66
 Chelidonichthys lucernus     :  65
 Lepidorhombus boscii         :  64
 Phycis blennoides            :  63
 (Other)                      :3257
 subSampCat   sex             wt           subSampWt       lenCode
 0:4895     NA's:5250   Min.   :     0   Min.   :    0.0   scm:4547
 1: 176                 1st Qu.:   100   1st Qu.:    0.0   mm : 703
 2: 144                 Median :   400   Median :    0.0
 3:  35                 Mean   :  2425   Mean   :  464.2
                        3rd Qu.:  1760   3rd Qu.:    0.0
                        Max.   :273600   Max.   :25500.0
```

```
> # ---- B.1 : Subsetting the dataset ---------------------------------
> # Subsetting for keeping only market sampling data
> SIBM2006cs <- subset(SIBM2006cs, sampType %in% "M")
> # resulting dataset summary :
> cstSummary(SIBM2006cs, tab = "tr", except = c("trpCode"))
```

```
 sampType landCtry vslFlgCtry   year          proj     vslLen    vslPwr   vslSize
 M:87     ITA:87   ITA:87     2006:87   CAMPBIOL:87   8   :12   31  :12   3   :12
                                                      21  : 6   19  : 6   1   : 6
                                                      5   : 6   299 : 5   53  : 5
                                                      13  : 4   113 : 4   10  : 4
                                                      NA's:59   287 : 1   37  : 1
                                                                NA's:59   NA's:59
```

```
 vslType harbour   foNum  daysAtSea vslId   sampCtry     sampMeth
 4:71    NA's:87   1:72   1:86      10:19   ITA:87   Observer:87
 1:16              2: 7   2: 1      17:12
                   5: 5             5 :11
                   3: 3             9 : 8
                                    11: 6
                                    22: 6
                                    3 : 5
                                    7 : 5
                                    1 : 4
                                    14: 3
                                    18: 2
                                    8 : 2
                                    12: 1
                                    24: 1
                                    26: 1
```

```
                                   6 : 1

> cstSummary(SIBM2006cs, tab = "hh", except = c("trpCode","date"))

 sampType landCtry vslFlgCtry   year          proj    staNum   foVal  aggLev catReg
 M:87     ITA:87   ITA:87     2006:87   CAMPBIOL:87   999:87   V:87   T:87   Lan:87




 sppReg     time     foDur        latIni        lonIni        latFin        lonFin
 All:76   14  : 1   0   :17   Min.   : NA   Min.   : NA   Min.   : NA   Min.   : NA
 Par:11   NA's:86   720 : 9   1st Qu.: NA   1st Qu.: NA   1st Qu.: NA   1st Qu.: NA
                    660 : 3   Median : NA   Median : NA   Median : NA   Median : NA
                    NA's:58   Mean   :NaN   Mean   :NaN   Mean   :NaN   Mean   :NaN
                              3rd Qu.: NA   3rd Qu.: NA   3rd Qu.: NA   3rd Qu.: NA
                              Max.   : NA   Max.   : NA   Max.   : NA   Max.   : NA
                              NA's   : 87   NA's   : 87   NA's   : 87   NA's   : 87
   area    rect    subRect    foDep    waterDep  foCatNat
 GSA9:87   99:87   NA's:87   40  : 2   40  : 2   GTR:71
                             70  : 2   70  : 2   OTB:16
                             48  : 1   48  : 1
                             80  : 1   80  : 1
                             NA's:81   NA's:81


                                  foCatEu5            foCatEu6  meshSize selDev
 Demersal species                     :71   GTR_DEF_22_0_0:71   22:71    0:87
 Mixed demersal and deep-water species:16   OTB_DEF_40_0_0:16   40:16




 meshSizeSelDev
 0:87

> cstSummary(SIBM2006cs, tab = "sl", except = c("trpCode"))

 sampType landCtry  vslFlgCtry   year           proj     staNum
 M:543    ITA:543   ITA:543    2006:543   CAMPBIOL:543   999:543


                        spp      catchCat  landCat   commCatScl      commCat    subSampCat
```

```
 Merluccius merluccius   : 77   LAN:524   HUC:524   Local:543   UNSORTED:524   0:468
 Mullus barbatus         : 34   DIS: 19   DIS: 19               DIS     : 19   1: 39
 Pagellus erythrinus     : 34                                                  2: 28
 Sepia officinalis       : 23                                                  3:  8
 Octopus vulgaris        : 22
 Mullus surmuletus       : 17
 Diplodus annularis      : 14
 Solea solea             : 13
 Uranoscopus scaber      : 13
 Chelidonichthys lucernus: 12
 Diplodus vulgaris       : 12
 Scorpaena porcus        : 12
 Squilla mantis          : 11
 Penaeus kerathurus      : 10
 Eledone cirrhosa        :  8
 Lithognathus mormyrus   :  8
 Conger conger           :  7
 Pagellus acarne         :  7
 Palinurus elephas       :  7
 (Other)                 :202
   sex            wt           subSampWt     lenCode
 NA's:543   Min.   :    15   Min.   :    0   scm:499
            1st Qu.:   260   1st Qu.:    0   mm : 44
            Median :  1420   Median :    0
            Mean   :  6457   Mean   :  635
            3rd Qu.:  5000   3rd Qu.:  100
            Max.   :204000   Max.   :20000
```

```
> cstSummary(SIBM2006cs, tab = "hl", except = c("trpCode","lenCls"))
```

```
 sampType landCtry   vslFlgCtry   year            proj      staNum
 M:1101   ITA:1101   ITA:1101   2006:1101   CAMPBIOL:1101   999:1101
```

```
                    spp      catchCat   landCat    commCatScl       commCat
 Merluccius merluccius:435   LAN:1100   HUC:1100   Local:1101   UNSORTED:1100
 Sepia officinalis    :189   DIS:   1   DIS:   1                DIS     :   1
 Mullus barbatus      :184
 Pagellus erythrinus  :106
 Mullus surmuletus    : 79
 Aristeus antennatus  : 55
 Nephrops norvegicus  : 25
 Octopus vulgaris     : 16
 Eledone cirrhosa     :  9
 Loligo vulgaris      :  3
 subSampCat      sex          lenNum
 0:432      UNSEXED:542   Min.   : 1.000
 2:342      F      :338   1st Qu.: 1.000
 1:248      M      :216   Median : 1.000
 3: 79      N      :  5   Mean   : 2.249
                          3rd Qu.: 3.000
                          Max.   :15.000
```

```
> cstSummary(SIBM2006cs, tab = "ca", sizeMax = 15)
```

```
 sampType landCtry  vslFlgCtry   year           proj     trpCode  staNum    quarter
 M:270    ITA:270   ITA:270    2006:270   CAMPBIOL:270   77 :52   999:270   4:150
```

```
                                                                  139:51             3: 72
                                                                  4  :48            1: 48
                                                                  117:47
                                                                  116:30
                                                                  78 :20
                                                                  136:14
                                                                  120: 8
```

```
month                       spp           sex      catchCat  landCat   commCatScl
11:85   Merluccius merluccius:195   F      :131   LAN:270   HUC:270   Local:270
9 :72   Mullus barbatus      : 75   M      :124
12:65                               UNSEXED: 15
3 :48
```

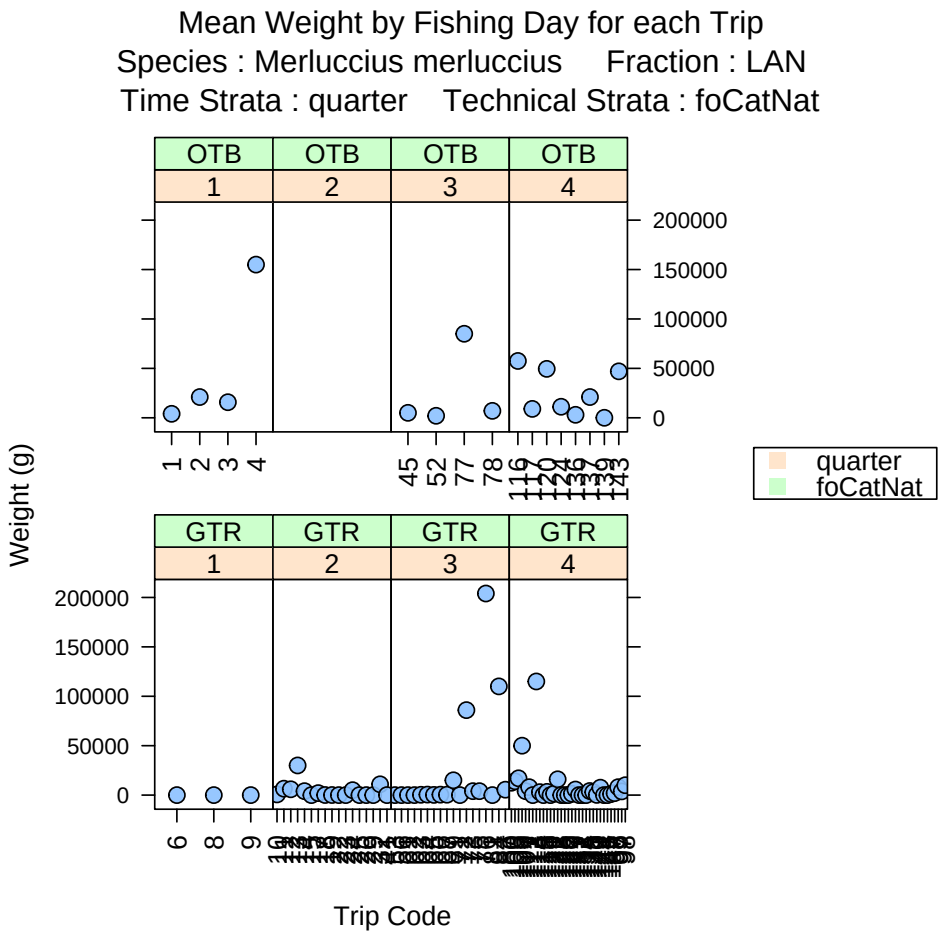
```
    commCat               stock       area     rect     subRect        lenCls    age
UNSORTED:270   Mediterranean:270   GSA9:270   99:270   NA's:270   210    : 19   1:127
                                                                   200    : 18   2:111
                                                                   205    : 14   3: 20
                                                                   215    : 14   4: 11
                                                                   150    : 13   6:  1
                                                                   160    : 12
                                                                   220    : 12
                                                                   225    : 11
                                                                   180    : 10
                                                                   190    : 10
                                                                   230    : 10
                                                                   175    :  9
                                                                   195    :  9
                                                                   235    :  8
                                                                   (Other):101
    fishId       lenCode   ageMeth    plusGrp     otoWt   otoSide     indWt
Min.   :  1.00   scm:270   NA's:270   -:270   Min.   :0   L:270   Min.   : 18.10
1st Qu.: 17.00                                1st Qu.:0           1st Qu.: 41.10
Median : 36.50                                Median :0           Median : 63.30
Mean   : 49.88                                Mean   :0           Mean   : 83.70
3rd Qu.: 78.75                                3rd Qu.:0           3rd Qu.: 88.25
Max.   :134.00                                Max.   :0           Max.   :630.40
                                                                  NA's   : 71.00
```

```
 matMeth                   matScale   matStage
 NA's:270   Holden & Raitt 1-5:270   1   :117
                                     2   : 55
                                     0   : 25
                                     3   :  3
                                     NA's: 70
```

```
> #no data in ca for quarter nř2 --> we will recode it at consolidation
> # Finally, we subset for keeping landings data
> SIBM2006cs <- subsetSpp(SIBM2006cs, catchCat %in% "LAN")
> # we can use 'dfApply' function to operate on a field over crossed modalities.
> #For example, which species are subcategorized.
> dfApply(SIBM2006cs@sl,"wt","spp","subSampCat",sum,na.rm=TRUE)
```

	spp	0	1	2	3
1	Alloteuthis spp.	42950	NA	NA	NA
2	Aphia minuta	26000	NA	NA	NA
3	Aristeus antennatus	39000	NA	25000	3500
4	Belone belone	960	NA	NA	NA
5	Boops boops	47500	NA	NA	NA
6	Chelidonichthys lucernus	4710	NA	NA	NA
7	Chelidonichthys obscurus	140	NA	NA	NA
8	Citharus linguatula	2450	NA	NA	NA
9	Conger conger	25780	NA	NA	NA
10	Coryphaena hippurus	800	NA	NA	NA
11	Dentex dentex	3200	NA	NA	NA
12	Diplodus annularis	13505	NA	NA	NA
13	Diplodus puntazzo	880	NA	NA	NA
14	Diplodus sargus	590	NA	NA	NA
15	Diplodus vulgaris	5400	NA	NA	NA
16	Eledone cirrhosa	202500	NA	8500	NA
17	Eledone moschata	13300	NA	NA	NA
18	Engraulis encrasicolus	281000	NA	NA	NA
19	Labrus merula	1380	NA	NA	NA
20	Labrus viridis	890	NA	NA	NA
21	Lepidopus caudatus	15000	NA	NA	NA
22	Lepidorhombus boscii	2000	NA	NA	NA
23	Lepidotrigla cavillone	60	NA	NA	NA
24	Lithognathus mormyrus	25380	NA	NA	NA
25	Liza aurata	380	NA	NA	NA
26	Liza saliens	200	NA	NA	NA
27	Loligo sp.	24200	NA	NA	NA
28	Loligo vulgaris	15740	NA	NA	NA
29	Lophius budegassa	1600	NA	NA	NA
30	Lophius spp.	15000	NA	NA	NA
31	Merluccius merluccius	789915	260995	228455	60
32	Mugil cephalus	15000	NA	NA	NA
33	Mugilidae	1360	NA	NA	NA
34	Mullus barbatus	183950	44865	7530	3090
35	Mullus surmuletus	34350	860	NA	NA
36	Muraena helena	12800	NA	NA	NA
37	Nephrops norvegicus	38000	NA	25000	NA
38	Oblada melanura	380	NA	NA	NA
39	Octopus macropus	520	NA	NA	NA
40	Octopus vulgaris	194390	NA	NA	NA
41	Ommastrephidae	31000	NA	NA	NA
42	Pagellus acarne	5530	NA	NA	NA
43	Pagellus erythrinus	47970	9675	12730	6255
44	Pagrus pagrus	240	NA	NA	NA
45	Palinurus elephas	15250	NA	NA	NA
46	Parapenaeus longirostris	25000	NA	NA	NA
47	Penaeus kerathurus	33760	NA	NA	NA
48	Phycis phycis	11420	NA	NA	NA

```
49            Pomatomus saltator |   6400     NA     NA   NA
50                 Raja asterias |   4400     NA     NA   NA
51                   Sarda sarda |  91000     NA     NA   NA
52            Sardina pilchardus |  15900     NA     NA   NA
53                   Sarpa salpa |   1940     NA     NA   NA
54                 Sciaena umbra |   4200     NA     NA   NA
55             Scomber japonicus |  30820     NA     NA   NA
56              Scomber scombrus |  13440     NA     NA   NA
57          Scophthalmus rhombus |   4930     NA     NA   NA
58              Scorpaena notata |     40     NA     NA   NA
59              Scorpaena porcus |  46220     NA     NA   NA
60              Scorpaena scrofa |  13620     NA     NA   NA
61        Scyliorhinus stellaris |    500     NA     NA   NA
62             Scyllarides latus |    100     NA     NA   NA
63             Sepia officinalis | 154500     NA     NA   NA
64              Seriola dumerili |   2600     NA     NA   NA
65             Serranus cabrilla |   1150     NA     NA   NA
66               Serranus scriba |   6990     NA     NA   NA
67                Solea lascaris |   2560     NA     NA   NA
68                   Solea solea |  61090     NA     NA   NA
69                 Sparus aurata |   1590     NA     NA   NA
70           Sphyraena sphyraena |  11150     NA     NA   NA
71       Spondyliosoma cantharus |   1220     NA     NA   NA
72                Squilla mantis |  58310     NA     NA   NA
73               Symphodus tinca |   1650     NA     NA   NA
74                Synodus saurus |    560     NA     NA   NA
75             Trachinus araneus |    500     NA     NA   NA
76            Trachinus radiatus |    750     NA     NA   NA
77       Trachurus mediterraneus |    790     NA     NA   NA
78           Trachurus trachurus |    960     NA     NA   NA
79 Trisopterus minutus capelanus |  53000     NA     NA   NA
80            Uranoscopus scaber |  18360     NA     NA   NA
81                    Zeus faber |    300     NA     NA   NA

> #cc strategy can be applied to 'Merluccius merluccius', 'Mullus barbatus'
> #or 'Pagellus erythrinus' species for example
> # ---- B.2 : FROM RAW TO VALIDATED DATASETS ---------------------------
> #Exploration of sampling data
> # Defining the stratification adopted  (no space stratification as there's data in only one area)
> strD <- strIni(timeStrata="quarter", techStrata="foCatNat")
> #a view of landed weights, sampled by trips, within chosen stratification
> obj <- landisVol(SIBM2006cs,strD,species="Merluccius merluccius")
> print(plot(obj))  #trip number are alpha.numerics

NULL

> #no data for quarter 2 and OTB gear, and some trips have uncommon sampled weights
> #(eg trip 76, foCatNat=="GTR" and quarter=="3",...)
> # ----  B.3 : Validating the dataset  ----
> #
> SIBM2006csv <- csDataVal(SIBM2006cs)
```


Mean Weight by Fishing Day for each Trip
Species : Merluccius merluccius Fraction : LAN
Time Strata : quarter Technical Strata : foCatNat
OTB
1
2
3
4
GTR
Weight (g)
Trip Code
quarter
foCatNat