Package 'COSTeda'

July 31, 2009

Title Exploratory Data Analysis methods for COST.

Version 1.3.4

Date 2009

Author COST Team and various contributors.

Description COSTeda provides the ability to graphically assess sampling coverage, explore and summarize the data and identify unusual observations. Furthermore, it can be used as a descriptive quality control tool.

Depends R(>= 2.8.1), COSTcore, lattice, nnet, methods

Maintainer Mathieu Merzereaud <Mathieu.Merzereaud@ifremer.fr>

License GPL 2 or above

LazyLoad Yes

LazyData Yes

R topics documented:

`bioPar.boxplot,csData-method`

Boxplot of individual biological parameters

Description

This method implements a boxplot of individual biological data. It requires a *csData* object with *ca* table.

Usage

```
bioPar.boxplot(object,type="wl",species="all",...)
```

Arguments

`object`	A *csData* object with *ca* informations.
`type`	To be chosen between `"wl"` (weight at length), `"ml"` (maturity at length) or `"sl"` (sex at length).
`species`	A character (vector) specifying species (default value is "all" for all species) .
`...`	Further graphical arguments.

Author(s)

Mathieu Merzereaud

See Also

bioPar.plot, csPlot.design

Examples

```
data(sole)
bioPar.boxplot(sole.cs)

#4 graphs on the same page
library(grid)
grid.newpage()
pushViewport(viewport(layout = grid.layout(2, 2)))

pushViewport(viewport(layout.pos.col=1,layout.pos.row=1))
print(bioPar.plot(sole.cs),newpage=FALSE) ; popViewport(1)
pushViewport(viewport(layout.pos.col=2, layout.pos.row=1))
print(bioPar.plot(sole.cs,type="ml"),newpage=FALSE) ; popViewport(1)
pushViewport(viewport(layout.pos.col=1, layout.pos.row=2))
print(bioPar.boxplot(sole.cs,type="sl"),newpage=FALSE) ; popViewport(1)
pushViewport(viewport(layout.pos.col=2, layout.pos.row=2))
print(bioPar.boxplot(sole.cs,type="ml"),newpage=FALSE) ; popViewport(1)
```

---

bioPar.plot,csData-method

Plots of individual biological parameters

Description

This method implements a scatter plot of individual biological data. It requires a *csData* object with *ca* table.

Usage

```
bioPar.plot(object,type="wl",species="all",selection=FALSE,...)
```

Arguments

object	A *csData* object with *ca* informations.
type	To be chosen between "wl" (weight at length), "ml" (maturity at length) or "sl" (sex at length).
species	A character (vector) specifying species (default value is "all" for all species) .
selection	If TRUE, outlier(s) identification can be done by clicking on points and corresponding ca table extract is then returned.
...	Further graphical arguments.

Author(s)

Mathieu Merzereaud

See Also

`bioPar.boxplot`, `csPlot.design`, `identify`

Examples

```
data(sole)
bioPar.plot(sole.cs)
```

`GraphsPar` *Graphical parameters for plotting procedures from "COSTeda" package.*

Description

List of graphical parameters used for **COSTeda** plotting procedures.

Usage

```
GraphsPar
```

Format

Components of *gp* graphical parameters list:

l.col	character	Color(s) of lines
l.lwd	numeric	Heaviness of lines
lty	numeric	Type of lines
pch	numeric, character	Type of points
p.col	character	Color(s) of points
p.bg	character	Filling color(s) of points
p.cex	numeric	Size of points
p.lwd	numeric	Heaviness of points
cex.lab	numeric	Size of labels
cex.axis	numeric	Size of axis annotations
cex.main	numeric	Size of main title
cex.sub	numeric	Size of subtitle
col.lab	character	Color of labels
col.axis	character	Color of axis annotations
col.main	character	Color of main title
col.sub	character	Color of subtitle
font	numeric	General font
font.lab	numeric	Font of labels
font.axis	numeric	Font of axis annotations
font.main	numeric	Font of main title
font.sub	numeric	font of subtitle
col	character	General color(s)
rot	numeric	Rotation of x-axis annotations
bg	character	General background color

Details

According to plotting procedures, some of these parameters turn out to be ineffective.

Examples

```
data(GraphsPar)
names(gp)
```

---

ageLenMulti,csDataVal-method

Multinomial modelisation applied to fisheries age-at-length data

---

Description

This method implements a multinomial analysis of age-at-length data, for specified time, space and technical stratification as defined in input *strIni* object. It requires a *csDataVal* object that can be built by `csDataVal` method from available **COSTcore** package. All information is taken from *ca* table.

Usage

```
ageLenMulti(data,strDef,elmts=list(tp="all",sp="all",tc="all"),age.plus=-1,...)
```

Arguments

`data`	A *csDataVal* object with *ca* informations.
`strDef`	Optionnal. A *strIni* object. Specified stratification must match with ca fields.
`elmts`	List of strata occurrence(s) to be kept for modelisation. *tp*, *sp* and *tc* are character vectors. `"all"` includes all occurrences for the strata in ca table.
`age.plus`	Threshold for grouping age (numeric value). (-1)-value means no grouping.
`...`	Further arguments.

Details

Output object can then be called by *plot* method to display fitted result (see `plot.edaResult`). Furthermore, response matrix (*outPut$age* element) and predictors (in *outPut$dat* table) internally called by *multinom* formula expression are provided. Considering the output as the full multinomial log-linear model, various strata effects studies can then be achieved from it using multinomial nested models or/and contrasts redefinition.

Author(s)

Mathieu Merzereaud

References

Gerritsen, H.D., McGrath, D., and Lordan, C. (2006) *A simple method for comparing age-length keys reveals significant regional differences within a single stock of haddock (Melanogrammus aeglefinus).* ICES Journal of Marine Science, 63: 1096-1100.

See Also

edaResult, plot.edaResult, multinom, strIni

Examples

```
data(sole)
sole.cs.val <- csDataVal(sole.cs)
ageLenMulti(sole.cs.val,age.plus=6)
```

---

agelenPlot *Plots age given length from csData*

Description

Plots age given length from the ca table of csData objects. Optionally can plot by a grouping variable, and specific factor levels within the grouping variable. The discarded and landed fractions of the catch can be plotted separately or together.

Usage

```
agelenPlot(x,by="spp",level="all",fraction=c("DIS","LAN"),title=TRUE,supsmu=FALSE,jitter=FA
```

Arguments

x	An object of class csData or csDataVal
by	The character name of a grouping variable.
level	The level within the grouping variable, the default is "all".
fraction	The fraction of the catch to plot. DIS for discards, LAN for landings. The default is to plot both if present
title	Logical. adds a title to the outer margin
supsmu	Logical. Adds a running line smoother to the plotted points.
jitter	Logical. If "TRUE" jitters the plotted points, the default is false.
...	other arguments, particularly those to plot and lines.

Details

The possible options for the grouping variable are those within the amalgamated hl table produced by mergecsData and include for time: "year", "month" and "quarter"; for space: "area", and "rect"; for technical: "foCatNat","foCatEu5" and "foCatEu6". Other options include "proj", "trpCode", "commCat" and "sex". The default is to plot by "spp" so, for a single species data set, this will plot all length frequencies.

For plotting selected levels within the grouping variable the names of those levels can be passed as a vector to level, e.g just to plot data from the first quarter then set by ="quarter" and level=1. Grouping variables that are numeric, such as months and quarters, are specified as numerics e.g. level=c(1,3). Grouping variables that are characters are specified as character strings e.g. level=c("OTB_DEM")

Setting supsmu=TRUE fits a running line smoother, "super smoother" supsmu, to the plotted points. The span argument to this function, which takes values in the range (0-1) can be passed as an argument to ....

If `jitter=TRUE` age class values will be jittered, see `jitter`.

Other arguments that can be passed as `...` include `pch` for setting the plotting character and `col` for setting the colour or the plotted points. If `supsmu==TRUE` then the colour of the fitted lines can be set using `col.line`, `lty` sets the line type and `lwd` the line width. `pch`, `col`, `col.line`, `lwd` and `lty` can all be passed a vector as arguments to enable the differetiation between levels of a grouping variable. Setting `add=TRUE` enables overlaying of multiple plots and for example, can be used to compare the fitted lines for a number of different levels of a grouping variable. If `axes=FALSE` no axes will be plotted. `ylim` sets the limits for the y axes and `xlab` and `ylab` are for axes labels. Other graphical parameters include `main`, `sub`, `cex.main`, `cex.asis`, `cex.lab` e.t.c see `par` for more details.

The outer margin default title is L̈ength distribution of *species* by *grouping variable*: This can be turned off if `title=FALSE`. The figure titles default to the argument passed as `by` and, if specified, `level`. This can be overwritten by a call to `main`. `main=""` will result in no figure title.

`par(mfrow=c(nrow,ncol))` can be used to adjust the number of plots per page to accommodate the multiple figures generated when the grouping variable has more than one level.

Value

A named list of the grouping variable and levels plotted, each component of which is itself a list with components x and y, the coordinates of the fitted SuperSmoother. See `supsmu` for details.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`ageLenMulti` which plots age proportions at length.

Examples

```
data(cod2004cs)
# Plotting the age and length from the ca table for the cod data set
agelenPlot(cod2004cs)

# and now grouped by area and adding a smooth to the plotted points
par(mfrow=c(3,2))
agelenPlot(cod2004cs,by="area",col=2,supsmu=TRUE,col.line=3,lwd=2,jitter=TRUE)

# and on the same plot
par(mfrow=c(1,1))
areas <-1:6
agelenPlot(cod2004cs,by="area",col=areas,pch=areas,supsmu=TRUE,col.line=areas,lwd=2,add=TRUE)
```

`boxplot.edaResult` *Boxplot of "edaResult" object*

Description

This method is a generic function for producing boxplots from an object of class *edaResult*, resulting from a *COSTeda* package function.

Methods

boxplot `signature("edaResult")`: plotting procedure of an object of class *edaResult* with *desc* slot equal to "landisVol".

Usage

desc	parameter	default	description
`"landisVol"`			
	x		*edaResult* object with *desc="landisVol"* (see `landisVol` method).
	type	`"FD"`	Character. If type="FO", boxplot of catch weights per fishing operation, for each fishing day of each trip will be displayed. Otherwise (type="FD"), boxplot of catch weights per fishing day for each trip will be displayed.
	...		Further graphical parameters.

Author(s)

Mathieu Merzereaud

See Also

`edaResult`, `landisVol`, `plot.edaResult`, `boxplot`

Examples

```
#desc="landisVol"
data(sole)
ldV <- landisVol(sole.cs,species="Solea solea")

boxplot(ldV)
boxplot(ldV,type="FO")
```

`cod2004ce` *FRS commercial effort data*

Description

FRS commercial effort data in the COST data exchange format.

Usage

```
data(cod2004ce)
```

Format

Formal class 'ceData' [package "COSTcore"] objects with 2 slots:
@desc: description
@ce: data frame of 18 variables
see ceData for details of the variables

Details

Commercial effort data from vessels landing the names species into Scottish ports, the data are aggregated by month, ICES statistical rectangle, ICES area and gear type. Compiled from fisheries information network (FIN) data.

Warning

These are test data sets only and should not to be used or cited without prior permission.

Source

FRS Marine Laboratory, Aberdeen, Scotland.

Examples

```
data(cod2004ce)
```

---

cod2004cl *FRS commercial landings data*

Description

FRS commercial landings data in the COST data exchange format.

Usage

```
data(cod2004cl)
```

Format

Formal class 'clData' [package "COSTcore"] objects with 2 slots:
@desc: description
@ce: data frame of 22 variables
see clData for details of the variables

Details

Commercial landings data from vessels landing the named species into Scottish, the data are aggregated by month, ICES statistical rectangle, ICES area and gear type. Compiled from fisheries information network (FIN) data.

Warning

This is a test data set only and should not to be used or cited without prior permission.

Source

FRS Marine Laboratory, Aberdeen, Scotland.

Examples

```
data(cod2004cl)
```

`cod2004cs` *FRS commercial sampling data*

Description

FRS commercial sampling data in the COST data exchange format.

Usage

```
data(cod2004cs)
```

Format

Formal class 'csData' [package "COSTcore"] objects with 6 slots
@desc: description
@tr: data.frame of 16 variables
@hh: data.frame of 29 variables
@sl: data.frame of 17 variables
@hl: data.frame of 16 variables
@ca: data.frame of 31 variables
see csData for details of the variables

Details

Commercial sampling data from vessels landing the names species into Scottish ports, the data are aggregated by month, ICES statistical rectangle, ICES area and gear type. Compiled from FRS sampling programme data.

Warning

These data sets are for test purposes only and should not to be used or cited without prior permission.

Source

FRS Marine Laboratory, Aberdeen. Scotland.

Examples

```
data(cod2004cs)
dim(cod2004cs)
str(cod2004cs)
```

`convert.icesarea.lat.lon` *Returns the coordinates of the (approximate) centre of an ICES area*

Description

Given an ICES Subarea, Division or Subdivision (i.e. in the form "V", "Vb", "Vb2" respectively), this function returns a list containing the longitude and latitude of the approximate centre of the area. These locations are suitable for displaying numeric or character information on spatial plots.

Usage

```
convert.icesarea.lat.lon(icesarea)
```

Arguments

`icesarea`	A character vector giving ICES Subareas, Divisions or Subdivisions in Roman numeric format.

Value

A list containing:

`lon`	Longitude in decimal degrees
`lat`	Latitude in decimal degrees

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`ices.division.lines` for plotting ICES Subarea Division and Subdivision boundaries, `convert.icesarea.statsq` for obtaining the component statistical rectangles of an area.

Examples

```
data(NHcoast)
plot(NHcoast$lon,NHcoast$lat,xlim=c(-10,10),ylim=c(50,60),type="l")
ices.division.lines()
a <-convert.icesarea.lat.lon("IVb")
text(a$lon,a$lat,"This is ICES area IVb")
```

`convert.icesarea.statsq` *Converts ICES areas into ICES statistical rectangles*

Description

Converts an ICES Subarea, Division or Subdivision (i.e. in the form "V", "Vb", "Vb2" respectively) into the constituent ICES statistical rectangles.

Usage

```
convert.icesarea.statsq(areas)
```

Arguments

`areas`	A vector of ICES Subarea, Division or Subdivision names in Roman numeric format.

Details

Interrogates the look up table *ICESAreasRects*.

Value

A list containing:

`statsq`	The constituent ICES statistical rectangles for the specified areas.
`parentarea`	A vector of the same length as `statsq` giving the parent ICES area name.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`ices.division.lines` for plotting ICES Subarea, Division and Subdivision boundaries, `convert.statsq.lat.lon` for obtaining the longitude and latitude centres of statistical rectangles.

Examples

```
# the statistical rectangles that make up areas VIb and VIIg
statsqs <-convert.icesarea.statsq(c("VIb","VIIg"))
# plotting them
statsq.latlon <-convert.statsq.lat.lon(statsqs$statsq)
plot(statsq.latlon$lon,statsq.latlon$lat,pch=15,cex=1.5,xlim=c(-25,5),ylim=c(48,63))
ices.division.lines(col=2)
data(NHcoast)
lines(NHcoast$lon,NHcoast$lat)
```

`convert.lon.lat.statsq` *Finds the ICES statistical rectangle for a lon, lat point.*

Description

Finds the ICES statistical rectangle that contains the given point where the point is specified as decimal lon,lat coordinates.

Usage

```
convert.lon.lat.statsq(lon,lat)
```

Arguments

`lon`	Longitude in decimal degrees.
`lat`	Latitude in decimal degrees.

Details

Whole degree longitude increments and half degree latitude increments are rounded upwards, hence (1,60) will return statsq "49F1" the centre of which is 1.5, 60.25.

Value

A character vector of ICES statistical rectangle(s).

Warning

This function differs from convert.lat.lon.statsq in package "Liz" which rounds down whole degree longitude increments and half degree latitude increments.

Author(s)

Liz Clarke, Alastair Pout, ⟨e.d.clarke@marlab.ac.uk⟩, ⟨a.pout@marlab.ac.uk⟩

See Also

`convert.statsq.lat.lon` for the converse.

Examples

```
convert.lon.lat.statsq(c(-1,-1.56),c(65,65.3))
```

`convert.samplingarea.statsq` *Converts FRS sampling area codes to ICES statistical rectangles*

Description

Given FRS sampling area codes this function returns the constituent ICES statistical rectangles.

Usage

```
convert.samplingarea.statsq(areacode,samplingarea)
```

Arguments

`areacode`	A character or numeric vector giving the names of strata within the sampling area e.g. c(1:10) for Demersal.
`samplingarea`	A character vector giving the names of sampling area; one of "Demersal", "DemersalMet", "Mackerel", "Herring", "Scallops" and "Edible crab". The default is "Demersal".

Value

A list containing:

`statsq`	The ICES statistical rectangles within the strata
`parentarea`	The strata
`parentname`	The name of the strata

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`convert.statsq.lat.lon` for obtaining the coordinates of statistical rectangles.

Examples

```
# Demersal sampling areas 1 to 13
doarea <-c(1:13)
convert.samplingarea.statsq(doarea)

# Nephrops sampling areas
doarea <-c("AR","CL","FD","FF","FL","IL","IS","MF","NM","NO","OC","ON","OW","SM")
convert.samplingarea.statsq(doarea,samplingarea="Nephrops")
```

`convert.statsq.icesarea`
Converts ICES statistical rectangles to ICES areas

Description

Converts ICES statistical rectangles into ICES Subarea, Division or Subdivision.

Usage

```
convert.statsq.icesarea(statsqs)
```

Arguments

`statsqs`	A character vector of ICES statistical rectangles.

Details

Interrogates the look up table *ICESAreasRects*.

Value

A list containing:

`subdivs`	The ICES subdivision.
`divs`	The ICES division.
`subarea`	The ICES subarea.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`convert.icesarea.statsq` for the converse.

Examples

```
# the ICES areas for statistical rectangles "45F2" and "43D8"
convert.statsq.icesarea(c("45F2","43D8"))
```

`convert.statsq.lat.lon`

Finds the coordinates of the centre of an ICES rectangle

Description

Given an ICES statistical rectangle (i.e. in the form "54E4"), this function returns a list containing the latitude and longitude of the centre of the rectangle.

Usage

```
convert.statsq.lat.lon(statsq)
```

Arguments

`statsq`	A character vector giving ICES statistical rectangles, i.e. in the form "54E4".

Value

`lon`	Logitude centre of the statistical rectangle(s)
`lat`	Latitude centre of the statistical rectangle(s)

Author(s)

Liz Clarke ⟨e.d.clarke@marlab.ac.uk⟩

See Also

`convert.lon.lat.statsq` for the converse.

Examples

```
convert.statsq.lat.lon(c("43E3","54D9"))
```

`convert.statsq.samplingarea`

Converts ICES statistical rectangles to FRS sampling area codes

---

Description

Given ICES statistical rectangles this function returns the coresponding FRS sampling area codes.

Usage

```
convert.statsq.samplingarea(statsq,samplingarea)
```

Arguments

`statsq`	A character vector giving the names of ICES statistical rectangle e.g. "45F2".
`samplingarea`	A character vector giving the names of sampling area; one of "Demersal", "DemersalMet", "Mackerel", "Herring", "Scallops" and "Edible crab". The default is "Demersal".

Value

A list containing:

`parentarea`	The strata
`parentname`	The name of the strata
`statsq`	The ICES statistical rectangles

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`convert.samplingarea.statsq` for obtaining the statistical rectangles within a sampling area.

Examples

```
# Demersal sampling areas for "47F0" and "46F0"
convert.statsq.samplingarea(c("47F0","46F0"))
```

---

csPlot.design — *Plot.design of individual biological parameters in "ca" table from a "csData" or "csDataVal" object*

Description

Specific version of *plot.design* procedure from **graphics** package applied to *csData* or *csDataVal* biological parameters.

Usage

```
csPlot.design(object,species="all",...)
```

Arguments

`object`	A *csData* or *csDataVal* object with *ca* informations.
`species`	A character (vector) specifying species (default value is "all" for all species) .
`...`	Further arguments.

Author(s)

Mathieu Merzereaud

See Also

`plot.design`, `bioPar.plot`, `bioPar.boxplot`

Examples

```
data(sole)
sole.cs.val <- csDataVal(sole.cs)
csPlot.design(sole.cs.val)
```

`cstSummary-methods` *Summary "plus" procedure for "COST" objects.*

Description

These methods implements a special *summary* procedure for objects of class *csData/csDataVal/csDataCons*, *clData/clDataVal/clDataCons* and *ceData/ceDataVal/ceDataCons*.

Usage

```
## S4 method for signature 'csData':
cstSummary(object,tab="tr",sizeMax=20,except=NULL,...)
## S4 method for signature 'csDataVal':
cstSummary(object,tab="tr",sizeMax=20,except=NULL,...)
## S4 method for signature 'csDataCons':
cstSummary(object,tab="tr",sizeMax=20,except=NULL,...)
## S4 method for signature 'clData':
cstSummary(object,sizeMax=20,except=NULL,...)
## S4 method for signature 'clDataVal':
cstSummary(object,sizeMax=20,except=NULL,...)
## S4 method for signature 'clDataCons':
cstSummary(object,sizeMax=20,except=NULL,...)
## S4 method for signature 'ceData':
cstSummary(object,sizeMax=20,except=NULL,...)
## S4 method for signature 'ceDataVal':
cstSummary(object,sizeMax=20,except=NULL,...)
## S4 method for signature 'ceDataCons':
cstSummary(object,sizeMax=20,except=NULL,...)
```

Arguments

`object`	A *COST* object.
`tab`	Character specifying one slot of a COST *cs* object (ie "tr", "hh", "sl", "hl" or "ca").
`sizeMax`	Numeric value specifying the number of rows displayed.
`except`	Character specifying fields to omit.
`...`	Further parameters.

Methods

cstSummary `signature(csData)`: summary for a *csData* object.

cstSummary `signature(csDataVal)`: summary for a *csDataVal* object.

cstSummary `signature(csDataCons)`: summary for a *csDataCons* object.

cstSummary `signature(clData)`: summary for a *clData* object.

cstSummary `signature(clDataVal)`: summary for a *clDataVal* object.

cstSummary `signature(clDataCons)`: summary for a *clDataCons* object.

cstSummary `signature(ceData)`: summary for a *ceData* object.

cstSummary `signature(ceDataVal)`: summary for a *ceDataVal* object.

cstSummary `signature(ceDataCons)`: summary for a *ceDataCons* object.

Author(s)

Mathieu Merzereaud

See Also

`summary`

Examples

```
data(sole)
cstSummary(sole.cs,tab="ca")
cstSummary(sole.cs,tab="ca",sizeMax=28)
cstSummary(sole.cs,tab="ca",except="trpCode")
```

`deltCalc` — *Calculation of Delta index for sampling outliers detection and variance calculation*

Description

This method implements the calculation of Delta values, derived from the formulation of the variance in landings-at-length. It requires an input *csData/csDataVal* object built from **COSTcore** package. Length distribution informations are taken from *hl* table.

Usage

```
deltCalc(data,strDef,species,fraction="LAN",strategy="metier",indSamp=TRUE,...)
```

Arguments

`data`	A *csData/csDataVal* object with *tr*, *hh*, *sl* and *hl* informations.
`strDef`	A *strIni* object specifying time, space or/and technical stratification.
`species`	Field specifying species (e.g `"Solea solea"`).
`fraction`	Fate of the catch on which calculation is made. To be chosen between `"LAN"`, `"DIS"` and `"all"` for total catch.
`strategy`	To be chosen between `"metier"` and `"cc"` (for commercial categories). Sample definition differs according to chosen strategy.

`indSamp`	If `TRUE`, output is within each sample and is dedicated to outliers detection. If `FALSE`, output is within length classes and is dedicated to variance calculation.
`...`	Further arguments.

Details

For more informations about arguments, see *FishFrame/COST Exchange format specification.*

Value

An object of class *edaResult* with *desc="sampDeltaCalc".*

Author(s)

Mathieu Merzereaud

References

Vigneau, J. and Mahevas, S. (2007) *Detecting sampling outliers and sampling heterogeneity when catch-at-length is estimated using the ratio estimator.* Oxford Journals.

See Also

`edaResult`, `plot.edaResult`, `lenDisPlot`

Examples

```
data(sole)
strD <- strIni(timeStrata="quarter",techStrata="commCat")
obj <- deltCalc(sole.cs,strD,"Solea solea",strategy="cc")
```

---

demersal.sampling.lines

Adds FRS demersal sampling area boundaries to a map

Description

Adds FRS demersal sampling area boundaries to a map.

Usage

```
demersal.sampling.lines(doarea,nos=FALSE,...)
```

Arguments

`doarea`	An optional numeric vector of the sampling areas boundaries to draw, the default is all.
`nos`	Logical, adds area numbers if TRUE, default is FALSE.
`...`	Other arguments to lines and text e.g. ldw, lty.

Author(s)

Alastair Pout⟨a.pout@marlab.ac.uk⟩

`dfApply` *Aggregate-like function with a two-dimensional result*

Description

This function is an aggregate-like method that operates over grouping variables whose one constitutes field names of the resulting dataframe.

Usage

```
dfApply(tab,valueField,rowFields,colField,fun,...)
```

Arguments

`tab`	an R data.frame.
`valueField`	character specifying the field on which operates the function (only one).
`rowFields`	character specifying the grouping element(s) within rows.
`colField`	character specifying the grouping element within columns (only one).
`fun`	function to be applied.
`...`	further arguments.

Author(s)

Mathieu Merzereaud

See Also

`tabConsist`

Examples

```
data(sole)
dfApply(sole.cs@sl,"wt",c("catchCat","commCat"),"sampType",sum,na.rm=TRUE)
```

`disCorrPlot,csDataCons-method`
Scatterplot of discards volume versus an auxiliary variable

Description

Plot a scatter plot of discards volume (weight or number) versus an auxiliary variable that can be fishing time or landings volume.

Usage

```
disCorrPlot(object,species="all",landSpp=as.character(NA),aux="time",val="weight",
            sampPar=TRUE,timeStrata=FALSE,spaceStrata=FALSE,techStrata=FALSE,reg=TRUE,
            show.legend="right",...)
```

Arguments

`object`	A *csDataCons* object.
`species`	Character (vector) specifying considered discarded species (default value is "all" for all species).
`landSpp`	Character (vector) specifying considered landed species (if NA (default value), species are those described by 'species' parameter). Unused if fishing time is the auxiliary variable.
`aux`	Character defining auxiliary variable (to be chosen between "time" for fishing time, and "landings" for landings volume).
`val`	Character specifying considered variable (to be chosen between "weight" and "number").
`sampPar`	Logical specifying if given species is/are considered to be automatically sampled during the sampling process (default value is `TRUE`).
`timeStrata`	Logical. If TRUE, display is done within time strata.
`spaceStrata`	Logical. If TRUE, display is done within spatial strata.
`techStrata`	Logical. If TRUE, display is done within technical strata.
`reg`	Logical. If TRUE, a simple regression line is drawn in each panel.
`show.legend`	Logical. If TRUE, a legend is displayed.
`...`	Further graphical arguments.

Value

A dataframe with discards volume (*disVol* field) and an auxiliary variable (*auxVar*) values for each sampled haul/trip recorded in *object@hh* table, with time, space and technical strata specification (one row for one plotted point).

Author(s)

Mathieu Merzereaud

See Also

`csDataCons`

Examples

```
#consolidated object is created for a given stratification
data(sole)
strDef <- strIni(timeStrata="quarter",techStrata="foCatEu5")
object <- csDataCons(csDataVal(sole.cs),strDef)

res <- disCorrPlot(object,aux="landings",techStrata=TRUE,l.col="steelblue",bg="gold",lty=2)
```

---

disInfo-methods *Display 'Tapply' function call output applied to "COST" objects.*

Description

Tapply-like methods to display information from validated and consolidated objects on .txt file. 'cs' tables are merged one by one from 'tr' to 'hl' until all required fields are in resulting table.

Usage

```
## S4 method for signature 'csDataVal':
disInfo(object,path,field,by,fun,...,biopar=FALSE,transpose=FALSE,title="",append=TRUE)
## S4 method for signature 'csDataCons':
disInfo(object,path,field,by,fun,...,biopar=FALSE,transpose=FALSE,title="",append=TRUE)
## S4 method for signature 'clDataVal':
disInfo(object,path,field,by,fun,...,transpose=FALSE,title="",append=TRUE)
## S4 method for signature 'clDataCons':
disInfo(object,path,field,by,fun,...,transpose=FALSE,title="",append=TRUE)
## S4 method for signature 'ceDataVal':
disInfo(object,path,field,by,fun,...,transpose=FALSE,title="",append=TRUE)
## S4 method for signature 'ceDataCons':
disInfo(object,path,field,by,fun,...,transpose=FALSE,title="",append=TRUE)
```

Arguments

`object`	A *COST* object.
`path`	Output .txt file path (with .txt extension).
`field`	Character specifying field on which function is applied. If more than one field is specified (only for cs table), fields are internally concatenated before calculation.
`by`	Vector with characters specifying factor(s) within which function is applied. For consolidated objects, only `"time", "space" & "technical"` are allowed.
`fun`	Function to be applied.
`...`	Further arguments to function to be applied.
`biopar`	If `TRUE`, calculation is made upon 'ca' table. If `FALSE`, calculation is done upon 'tr', 'hh', 'sl' and 'hl' table.
`transpose`	If `TRUE`, output array is transposed.
`title`	Title to be written on .txt output file.
`append`	If `TRUE`, output will be appended to .txt file. If `FALSE`, file will be overwritten. If `NA`, result is not exported, but an output list with 'title' and 'result' elements is returned.

Methods

disInfo `signature(csDataVal)`: disInfo for a *csDataVal* object.

disInfo `signature(csDataCons)`: disInfo for a *csDataCons* object.

disInfo `signature(clDataVal)`: disInfo for a *clDataVal* object.

disInfo `signature(clDataCons)`: disInfo for a *clDataCons* object.

disInfo `signature(ceDataVal)`: disInfo for a *ceDataVal* object.

disInfo `signature(ceDataCons)`: disInfo for a *ceDataCons* object.

Author(s)

Mathieu Merzereaud

See Also

`tapply`

Examples

```
data(sole)
sole.cs.val <- csDataVal(sole.cs)

##Change path before run
#
#Path <- "C:/draft.txt"
#disInfo(sole.cs.val,Path,"lenNum","lenCls",sum,na.rm=TRUE,
#        title="Measured numbers at length",append=FALSE)
#disInfo(sole.cs.val,Path,"wt",c("quarter","foCatEu5"),sum,na.rm=TRUE,
#        title="Total sampled weights by quarter and metier")
#disInfo(sole.cs.val,Path,"wt","foCatEu5",sum,na.rm=TRUE,
#        title="Total sampled weights by metier")
#disInfo(sole.cs.val,Path,"subSampWt",c("quarter","foCatEu5","area"),sum,na.rm=TRUE,
#        title="Measured weights by quarter, metier and area")
#disInfo(sole.cs.val,Path,c("trpCode","staNum"),c("quarter"),
#        function(x) length(unique(x)),title="Total FO numbers of sampled trips by quarter")
#disInfo(sole.cs.val,Path,c("trpCode","staNum","landCat"),c("quarter"),
#        function(x) length(unique(x)),title="Sampled FO numbers by quarter")
#disInfo(sole.cs.val,Path,"lenNum",c("quarter","commCat"),sum,na.rm=TRUE,
#        title="Number of measured fish by quarter and commercial category")
##information from CA table
#disInfo(sole.cs.val,Path,"indWt",c("quarter","sex"),mean,na.rm=TRUE,biopar=TRUE,
#        title="Mean individual weight by quarter and sex")
```

---

edaResult-class *Class "edaResult"*

Description

Any object of this class is created by a procedure from *COSTeda* package. Its content depends on *desc* slot that refers to mother function. Most often it can be called by a plotting procedure (e.g *plot* or *boxplot*).

Slots

slot	desc	elements	class	description
`desc`			`character`	Distinctive reference to mother function
`outPut`			`ANY`	Information stored in the object. It depends on *desc* slot.
	csRelativeValue		`data.frame`	Stratified relative values calculated with *relativeValue* procedure applied to 'cs'.
	clceRelativeValue		`data.frame`	Stratified relative values calculated with *relativeValue* procedure applied to 'cl' or 'ce'.
	sampDeltaCalc		`list`	Result from *deltCalc* procedure.
		species	`character`	Specified species.
		fraction	`character`	Catch category.
		strategy	`character`	Chosen strategy for Delta calculation.
		timeStrata	`character`	Specified time stratification.
		spaceStrata	`character`	Specified space stratification.
		techStrata	`character`	Specified technical stratification.
		DeltaMatrix	`array`	Sum of squared delta values of each sample within each strata and length class (only if `indSamp=FALSE` in `deltCalc` function call).
		NkMatrix	`array`	Number of samples within each strata (only if `indSamp=FALSE` in `deltCalc` function call).
		WkMatrix	`array`	Sampled weight within each strata in grams (only if `indSamp=FALSE` in `deltCalc` function call).
		SampDeltaMat	`data.frame`	Data.frame of delta values within each sample (only if `indSamp=TRUE` in `deltCalc` function call).
		tab	`data.frame`	Data.frame resulting from treatment (only if `indSamp=TRUE` in `deltCalc` function call).
		DFsamp	`data.frame`	Informations about each sample (only if `indSamp=TRUE` in `deltCalc` function call).
	sampDeltaId		`list`	Result from identification process of *edaResult* object with *desc="sampDeltaCalc"* plotting procedure.
		species	`character`	Specified species.
		fraction	`character`	Catch category.
		sampId	`data.frame`	Specification of identified samples.
		tabId	`data.frame`	Length distribution data from identified samples.
		tab	`data.frame`	Resulting data from `deltCalc` procedure.
	landisVol		`list`	Result from *landisVol* procedure.
		species	`character`	Specified species.
		fraction	`character`	Catch category.
		strategy	`character`	Chosen strategy for Delta calculation.
		timeStrata	`character`	Specified time stratification.
		spaceStrata	`character`	Specified space stratification.
		techStrata	`character`	Specified technical stratification.
		`VolFO_FDTR`	`list`	Catch weight by FO for each fishing day of each trip.
		`MeanFO_FDTR`	`numeric`	Mean FO-catch weight for each fishing day of each trip.
		`VolFD_TR`	`list`	Catch weight by fishing day for each trip and each strata, raised by numbers of FO.

	MeanFD_TR	numeric	Mean FD-raised catch weight by trip and strata.
alMulti		list	Result from *ageLenMulti* procedure.
	timeStrata	character	Time stratification field.
	spaceStrata	character	Space stratification field.
	techStrata	character	Technical stratification field.
	Mm	multinomial	A **nnet** package object resulting from *multinom* procedure.
	dat	data.frame	Age-at-Length data for specified stratification. Predictors used in model formula expression were taken from this table.
	age	matrix	Extract of *dat* table. Response matrix used in model formula expression.

Author(s)

Mathieu Merzereaud

See Also

`plot.edaResult`, `boxplot.edaResult`, `relativeValue`, `deltCalc`, `landisVol`, `ageLenMulti`

Examples

```
showClass("edaResult")
```

`ices.division.lines` *Adds ICES Division lines to a map*

Description

Adds ICES Subarea and Division boundaries to an existing plot.

Usage

```
ices.division.lines(division = NULL, area = NULL, lty = 1, col = 1, lwd = 1)
```

Arguments

`division`	Optional character vector of which Divisions to do, the default is to do all.
`area`	Optional character vector of which Subareas to do, the default is to do all.
`lty`	Line type, the default is 1.
`col`	Line colour, the default is 1 (black).
`lwd`	Line width, the default is 1.

Details

If `area` is specified the constituent division lines are also plotted.
The Baltic areas are not (yet) covered.

Author(s)

Liz Clarke ⟨e.d.clarke@marlab.ac.uk⟩

Examples

```
data(NHcoast)
plot(NHcoast$lon,NHcoast$lat,xlim=c(-55,25),ylim=c(35,75),type="l")
ices.division.lines(col=2)
```

`ices.division.names` *Adds ICES Division names to a map*

Description

Adds ICES Subarea and Division names to an existing plot.

Usage

```
ices.division.names(text.cex = 1)
```

Arguments

`text.cex`	Cex size of the text

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

Examples

```
data(NHcoast)
plot(NHcoast$lon,NHcoast$lat,xlim=c(-55,25),ylim=c(35,75),type="l")
ices.division.lines(col=2)
ices.division.names(0.7)
```

`is.statsq` *Checks for ICES statistical rectangle format*

Description

Function that checks its argument against the ICES statistical rectangle format.

Usage

```
is.statsq(x)
```

Arguments

`x`	A character vector to be checked against the ICES statistical rectangle format.

Details

ICES statistical rectangles follow a four digit format e.g. "45F3", the first two digits representing latitude, the second two longitude. This function simply checks if x is 4 characters long, if the 1st, 2nd and 4th are numerals in the range 0 to 9, and the third is a capital letter. Of use in distinguishing the type of spatial stratification of a COST object between statistical rectangles, ICES areas and FAO areas.

Value

`TRUE`	if the above conditions are met.

Warning

This only checks the format of x, there may not necessarily be a ICES statistical rectangle with that exact name.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`code.list$rect`, for a full list of the ICES statistical rectangles used in the COST exchange format.

Examples

```
# 35D2 follows the ICES rectangle format
is.statsq("35D2")
# but ICES area VIa does not
is.statsq("VIa")
```

`landisVol` *Calculation upon sea-sampled catch weight at FO level, fishing day level, and trip level.*

Description

This method creates an object of class *edaResult* with *desc="landisVol"* containing volume (weights) informations about sampled and raised catch, for a given species in a specified catch category. It requires a *csData* object built from **COSTcore** package. Only sea sampling data is computed.

Usage

```
## S4 method for signature 'csData, missing':
landisVol(object,species,fraction="LAN",sampPar=TRUE,...)
## S4 method for signature 'csData, strIni':
landisVol(object,strDef,species,fraction="LAN",sampPar=TRUE,...)
```

Arguments

`object`	A *csData* object with sea-sampling information (*tr*, *hh* and *sl* required).
`strDef`	A *strIni* object specifying time (e.g `"year"`, `"quarter"`, `"month"`,...), space (e.g `"area"`, `"rect"`,...) and/or technical stratification (e.g `"foCatNat"`, `"foCatEu5"`,...).
`species`	Field specifying species (e.g `"Solea solea"`).
`fraction`	Field specifying catch category (to be chosen between `"LAN"` and `"DIS"`.).
`sampPar`	Logical indicating if, during the sampling process, specified species is considered to be automatically sampled or not (therefore, 0-values criterion differs).
`...`	Further arguments.

Value

An object of class *edaResult* with *desc="landisVol"*.

Author(s)

Mathieu Merzereaud

See Also

`edaResult`, `plot.edaResult`, `boxplot.edaResult`

Examples

```
data(sole)
df <- subset(sole.cs,sampType=="S")  #only sea sampling data is kept
obj <- landisVol(df,species="Solea solea")
```

lenDisPlot *Plot of length distribution*

Description

This method plots from a *csData/csDataVal* object the length distribution of one or several trips, for given species and catch category, at trip level or fishing operation level.

Usage

```
lenDisPlot(x,species,fraction="LAN",trpCode="all",level="trip",...)
```

Arguments

x	A *csData* or *csDataVal* object with *hl* informations.
species	Character string specifying species (e.g "Solea solea").
fraction	Field specifying catch category ("LAN", "DIS", or c("LAN","DIS") for total catch).
trpCode	Character specifying trip codes ("all" means that all trips in x object will be considered).
level	Character string specifying the level at which the length distribution is considered ("fo" for fishing operation-level, "trip" for trip-level).
...	Further graphical arguments.

Author(s)

Mathieu Merzereaud

See Also

edaResult, deltCalc, plot.edaResult

Examples

```
data(sole)
lenDisPlot(sole.cs,"Solea solea","DIS","DIL1197",level="fo")
```

lengthHist *Plots a length frequency histogram from csData*

Description

Plots histograms of the length frequency data from the hl table of csData objects. Optionally can plot by a grouping variable, and specific factor levels within the grouping variable. The discarded and landed fractions of the catch can be plotted separately or together.

Usage

```
lengthHist(x,by="spp",level="all",fraction=c("DIS","LAN"),title=TRUE,...)
```

Arguments

`x`	an object of class csData or csDataVal
`by`	the character name of a grouping variable.
`level`	the level within the grouping variable, the default is "all".
`fraction`	the fraction of the catch to plot. DIS for discards, LAN for landings. The default is to plot both if present
`title`	logical. adds a title to the outer margin
`...`	other arguments, particularly those to `hist`.

Details

The possible options for the grouping variable are those within the amalgamated hl table produced by `mergecsData` and include for time: "year", "month" and "quarter"; for space: "area", and "rect"; for technical: "foCatNat","foCatEu5" and "foCatEu6". Other options include "proj", "trpCode", "commCat" and "sex". The default is to plot by "spp" so, for a single species data set, this will plot all length frequencies.

For plotting selected levels within the grouping variable the names of those levels can be passed as a vector to `level`, e.g just to plot data from the first quarter then set `by ="quarter"` and `level=1`. Grouping variables that are numeric, such as months and quarters, are specified as numerics e.g. `level=c(1,3)`. Grouping variables that are characters are specified as character strings e.g. `level=c("OTB_DEM")`

The arguments that can be passed as `...` includes: `col` for setting the colour or the bars; `border` for setting the colour of the borders of the bars; `add=TRUE` for adding to an existing plot; `angle` for setting the angle of shading lines; `density` for setting the density of the shading lines. The argument `freq=FASLE` will plot a density histogram. If `axes=FALSE` no axes will be plotted. `ylim` sets the limits for the y axes (only applicable for frequency plots) and `xlab` and `ylab` are for axes labels. Other graphical parameters include `main`, `sub`, `cex.main`, `cex.asis`, `cex.lab` e.t.c see `hist` and `par` for more details.

The outer margin default title is L̈ength distribution of *species* by *grouping variable*:. This can be turned off if `title=FALSE`. The figure titles default to the argument passed as `by` and, if specified, `level`. This can be overwritten by a call to `main`. `main=""` will result in no figure title.

`par(mfrow=c(nrow,ncol))` can be used to adjust the number of plots per page to accommodate the multiple figures generated when the grouping variable has more than one level.

Value

A named list of the grouping variable and levels plotted, each component of which is itself a list of class "histogram" with components breaks, counts, density, intensities, mids xname and equidist. See `hist` for details.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`lenDisPlot` which plots length distributions within trips.

Examples

```
data(cod2004cs)
# Plotting the length distribution in the hl table
lengthHist(cod2004cs,col=2)

# and now grouped by the commercial category
lengthHist(cod2004cs,by="commCat",level=1,col=2,density=30,main="")
for(i in 2:6)lengthHist(cod2004cs,by="commCat",level=i,col=(i+1),density=30,main="",add=TRUE,title=FAL

# plotting by quarter
par(mfrow=c(2,2))
lengthHist(cod2004cs,by="quarter",col=2,freq=TRUE)
par(mfrow=c(1,1))
```

---

mergecsData — *Adds parent variables to the child records of a csData COST object.*

Description

This function is designed as a prerequisite for plotting variables within the tr, sl, hl and ca data frames of a csData object where the spatio-temporal and technical information within the hh data frame is required.

Usage

```
mergecsData(csobj)
```

Arguments

csobj	an object of class csData

Details

csData objects consist of 5 nested data frames: tr, hh, sl, hl and ca. Information of ICES statistical rectangle, ICES (or FAO) areas, and gear type are recorded as variables only within the hh data frame. This function adds the variables

$rect, $area, $date, $foCatNat, $foCatEu5, $foCatEu6, $quarter, $month, and $yearfromdate

to the tr, hl, sl and ca data frames and defines

$quarter, $month, and $yearfromdate

for the hh data frame.
$yearfromdate is an additional variable to check that $year in the tr, sl and hl data frames are compatible with the year derived from $date within the hh data frame.

Value

csobj	An amended csData object with the additional variables as outlined above

Warning

The returned csobj is no longer in csData format.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`spacePlot` which calls this function for plotting csData objects.

Examples

```
data(cod2004cs)
# A csData@hl table contains data on length frequencies but no
# information on where these data were obtained, when (other than the year)
# and with what gear type
dim(cod2004cs@hl)
head(cod2004cs@hl)
newcod2004cs <-mergecsData(cod2004cs)
# The merged csData@hl table contains the additional
# information on where these data were obtained, when and with what gear type
dim(newcod2004cs@hl)
head(newcod2004cs@hl)
```

`plot.edaResult` *Plot "edaResult" object*

Description

This method is a generic function for plotting *edaResult* objects. These objects result from a *COSTeda* package function call.

Methods

plot `signature("edaResult","missing")`: plotting procedure of an object of class *edaResult* with *desc* slot equal to "csRelativeValue", "clceRelativeValue", "sampDeltaCalc", "sampDeltaId", "landisVol" or "alMulti".

plot `signature("edaResult","edaResult")`: plotting procedure of two objects of class *edaResult* with *desc* slot equal to "csRelativeValue" or "clceRelativeValue".

Usage of plot.edaResult method according to 'desc' slot in input object :

desc	parameters	default value	description
`"csRelativeValue"`			
	x		*edaResult* object with *desc="csRelativeValue"* (see `relativeValue` method).
	...		Further graphical parameters.
`"clceRelativeValue"`			
	x		*edaResult* object with *desc="clceRelativeValue"* (see `relativeValue` method).
	...		Further graphical parameters.

`"sampDeltaCalc"`			
	x		*edaResult* object with *desc="sampDeltaCalc"* (see `deltCalc` method).
	elmts	list(tp="all", sp="all",tc="all")	List of levels for specified stratification to be displayed on the graph.
	strat1		Optionnal. To be chosen between `"timeStrata"`, `"spaceStrata"` and `"techStrata"`. Primary stratification for graphical display.
	strat2	`"NULL"`	Optionnal. To be chosen between `"timeStrata"`, `"spaceStrata"` and `"techStrata"`. Secondary stratification for graphical display.
	selection	`FALSE`	If `TRUE`, outliers identification is made, and an *edaResult* object with `desc="sampDeltaId"` is returned. Displayed values during identification process are taken from 'SampNum' field from `sampId` returned data.frame.
	show.legend	`"right"`	Display the legend (`""` means "no legend").
	shift	`FALSE`	If `TRUE`, displayed text is shifted.
	...		Further graphical parameters.
`"sampDeltaId"`			
	x		*edaResult* object with *desc="sampDeltaId"* (see `plot.edaResult` method for *desc="sampDeltaCalc"*).
	smpNum	`"all"`	Character specifying sample Id(s) as displayed during outliers identification process ("all" is a shortcut to display all identified samples).
	show.legend	`"right"`	Display the legend (`""` means "no legend").
	...		Further graphical parameters.
`"landisVol"`			
	x		*edaResult* object with *desc="landisVol"* (see `landisVol` method).
	type	`"FD"`	Character. If type="FO", mean FO-catch weight for each fishing day of each trip will be displayed. Otherwise (i.e type="FD"), mean fishing-day-catch weight by trip will be displayed.
	groups	`NULL`	Only for `type="FD"`. Character specifying intra-graph stratification (to be chosen between `"timeStrata"`, `"techStrata"`, `"spaceStrata"` and `NULL`).
	...		Further graphical parameters.
`"alMulti"`			
	x		*edaResult* object with *desc="alMulti"* (see `ageLenMulti` method).
	grps	`NULL`	Character or NULL. Strata to be differentiated in each panel. `NULL` means one graph per crossed strata (to be chosen between "timeStrata", "spaceStrata", "techStrata" and NULL).
	show.legend	`"right"`	Display the legend (`""` means "no legend").
	...		Further graphical parameters.

Author(s)

Mathieu Merzereaud

See Also

`edaResult`, `relativeValue`, `deltCalc`, `landisVol`, `ageLenMulti`, `plot`, `boxplot.edaResult`, `GraphsPar`

Examples

```
#desc="csRelativeValue" or/and "clceRelativeValue"
data(sole)
sole.cs.val <- csDataVal(sole.cs)
sole.cl.val <- clDataVal(sole.cl)
strD <- strIni(timeStrata="month",spaceStrata="area",techStrata="commCat")
CS <- relativeValue(sole.cs.val,strD,"nbSamp")
CL <- relativeValue(sole.cl.val,strD)

plot(CS)
plot(CS,CL)

#desc="sampDeltaCalc"
strD <- strIni(timeStrata="quarter",techStrata="commCat")
dlt <- deltCalc(sole.cs,strD,"Solea solea",strategy="cc",indSamp=TRUE)

plot(dlt,strat1="techStrata",strat2="timeStrata")


##desc="sampDeltaId"
#sl <- plot(dlt,strat1="techStrata",strat2="timeStrata",selection=TRUE)
#plot(sl)

#desc="landisVol"
df <- subset(sole.cs,sampType=="S")  #only sea sampling data is kept
ldV <- landisVol(df,strIni(techStrata="foCatEu5",timeStrata="quarter"),
                 species="Solea solea")

plot(ldV,rot=20,cex.lab=0.8)
plot(ldV,groups="techStrata")
plot(ldV,type="FO")

#desc="alMulti"
strD <- strIni(timeStrata="quarter",spaceStrata="area")
aLM <- ageLenMulti(sole.cs.val,strD,elmts=list(tp=c("2","3"),sp="all",tc="all"),age.plus=6)

plot(aLM,grps="timeStrata",l.col=c("steelblue","violetred2"))
```

`relativeValue` *Calculation of relative values of a numerical field within time, space and/or technical stratification*

Description

This method calculates relative values of a population level variable (if input object class is *clDataVal/clDataCons/ceDataVal/ceDataCons*), or a sampling level variable (if input object class is *csDataVal/csDataCons*). Calculation can be done within time, space and/or technical stratification. This stratification (as well as an optionnal strata recoding process) is defined by *strDef* parameter for input validated objects (see *strIni*). Output is an *edaResult* object with *desc*="csRelativeValue" or "clceRelativeValue". An exploratory graphic to compare two objects can be made by applying *plot* function. (see plot.edaResult)

Usage

```
## S4 method for signature 'csDataVal, missing':
relativeValue(data,field="lenNum",...)
## S4 method for signature 'csDataVal, strIni':
relativeValue(data,strDef,field="lenNum",...)
## S4 method for signature 'clDataVal, missing':
relativeValue(data,field="landWt",...)
## S4 method for signature 'clDataVal, strIni':
relativeValue(data,strDef,field="landWt",...)
## S4 method for signature 'ceDataVal, missing':
relativeValue(data,field="trpNum",...)
## S4 method for signature 'ceDataVal, strIni':
relativeValue(data,strDef,field="trpNum",...)
## S4 method for signature 'csDataCons, missing':
relativeValue(data,field="lenNum",...)
## S4 method for signature 'clDataCons, missing':
relativeValue(data,field="landWt",...)
## S4 method for signature 'ceDataCons, missing':
relativeValue(data,field="trpNum",...)
```

Arguments

`data`	A *csDataVal/clDataVal/ceDataVal/csDataCons/clDatacons/ceDataCons* object.
`strDef`	Optionnal. A *strIni* object describing stratification and recoding parameters. Specified stratification must match with 'data' parameter.
`field`	A numeric field from *data* (e.g "lenNum", "wt", "subSampWt" or "nbSamp" (number of samples in sl table) and "nbInd" (number of individuals in ca table) for 'cs', "landWt" for 'cl', or "trpNum" for 'ce').
`...`	Further arguments.

Author(s)

Mathieu Merzereaud

See Also

`strIni`

Examples

```
data(sole)
sole.cs.val <- csDataVal(sole.cs)
```

```
sole.cl.val <- clDataVal(sole.cl)
strD <- strIni(timeStrata="month",spaceStrata="area",techStrata="commCat")

CS <- relativeValue(sole.cs.val,strD,"nbSamp")
CL <- relativeValue(sole.cl.val,strD)
```

`COSTeda-datasets` *Internal datasets for COSTeda package spatial plotting*

Description

Various datasets for COSTeda mapping methods

Source

FRS Marine Laboratory, Aberdeen, Scotland.

`spacePlot` *Spatial plotting function*

Description

Function that maps a numeric variable over spatial strata, grouped by a function. The possible spatial strata are ICES statistical rectangles, ICES areas, FAO areas and GSA management areas within the Mediterranean. The grouping function can be any that can be passed as the `func` argument to tapply. If passed a COSTobject this function will plot any numeric variable within a specified object optionally by time and technical stratification. The geographic area covered extends to approximately to -50 degrees West, 70 degrees East longitude and between 20 and 85 degrees of latitude.

Usage

```
spacePlot(costobj, variable, SpaceStrata, func, TimeStrata, TechStrata,...)
```

Arguments

`costobj`	Optionally the name of a COST object of class either 'csData', 'clData' or 'ceData' or their validated and consolidated forms.
`variable`	The name of the numeric variable within the specified COST object to be plotted specified as a character string, e.g. "age" . If a COST object is missing a vector of the numeric variable to be plotted.
`SpaceStrata`	The name of the spatial strata over which `variable` is to be grouped specified as a character string. Options are ICES statistical rectangle "rect", or ICES and FAO areas "area" if a COST object is specified. If a COST object is missing a character vector of the same length as `variable` specifying the spatial strata. ICES rectangles have the form "34D2" ICES areas in have the form "IVa" FAO areas have the form "27.4.a" (FOA area 27 corresponds to ICES areas).

GSA areas within the Mediterranean are numbered "G3701" to "G3730".
FRS sampling areas are prefixed with a six letter code the first three being FRS the second three specifying the sampling shceme:
FRSDMM demersal metier sampling area two and three digit codes e.g. "FRSDMM4O" "FRSDMM4aO"
FRSDMS demersal sampling area two digit codes e.g. "FRSDMS05"
FRSMAC mackerel sampling area two digit codes e.g. "FRSMAC01"
FRSHER herring sampling area two digit codes e.g. "FRSHER01"
FRSNEP Nephrops sampling area two letter codes e.g. "FRSNEPFL"
FRSSCA scallop sampling area two letter codes e.g. "FRSSCAEC"

`func` The name of the grouping function to be applied to `variable` over `SpaceStrata`. This can be any function that can be passed as the `func` argument to tapply.

`TimeStrata` When a COST object is specified the name of the temporal strata over which `variable` is to be grouped specified as a character string.
The available options are "year", "quarter", and "month".

`TechStrata` When a COST object is specified the name of the technical strata over which `variable` is to be grouped specified as a character string.
The available options are "foCatEu5", "foCatEu6", (level 5 and level 6 respectively of the DCR matrix) and "foCatNat" a national gear specification.

`...` Additional arguments such as those to `plot`

Other arguments

Other arguments are :

xlim Vector of c(min, max) for the longitude limits of the plotting area in decimal degrees. Longitudes to the west of 0 degrees are expressed as negatives.

ylim Vector of c(min, max) for the latitude limits of the plotting area in decimal degrees.

zlim Vector of c(min, max) for the limits to `variable`

xlab Label for the x axis

ylab Label for the y axis

breaks The breaks by which to split `variable`. The default is NULL which results in seq(zmin, zmax, length.out=8) being used.

maptype One of "image", "contour", "bubble" or "values". The default is "image".
"image" plots `variable` grouped by `func` as an image plot over the chosen `SpaceStrata`.
"contour" plots `variable` grouped by `func` as contour lines over the chosen `SpaceStrata`.
"bubble" plots `variable` grouped by `func` as a scaled plotting character at the (approximate) centre of the chosen `SpaceStrata`.
"values" plots `variable` grouped by `func` as numeric values at the (approximate) centre of the chosen `SpaceStrata`.

plotmap Logical. If FALSE plots the map area but omits the "image", "contour", "bubble", or "values".

overlay Logical. If TRUE adds to the proceeding plot.

squarmap Logical. If TRUE produces a square plot area with side length equal to the longer of the x or y limits. If FALSE produces a maximum plot area map with x and y limits as specified. The default is FALSE

area.lines Logical. If TRUE adds the boundary lines of ICES or FRS demsersal sampling areas.

depths Vector of the form c(50,100,200,500). Adds the specified depth contour.

statrects Logical. If TRUE adds ICES statistical rectangle division lines.

fcoast Logical. If TRUE adds 1:250,000 scale coastline. Not available when landmass=TRUE.

landmass Logical. If TRUE adds landmass polygons.

pch Plotting character used for maptype="bubble", the default is pch=1.

colour Logical. If TRUE the colours used for image plots are rev(heatcolors(n-1)) where n is the length of breaks. If FALSE uses grey(rev(seq(0.1, 0.9, length.out =(n-1)))).

col.coast Colour to be used for plotting the coastline, the default is "blue".

col.cont Colour to be used for the contour lines, the default is "grey".

col.pch Colour to be used for the plotting character, the default is "red".

col.rect Colour to be used for statistical rectangle lines, the default is "grey".

col.land Colour to be used for the landmass polygons, the default is "white".

col.depth Colour to be used for the depth contour lines, the default is "grey".

col.text Colour to be used for values when maptype="values", the default is 1 (black).

scale Logical. If TRUE a scale is added to the plot.

scale.title Title for the scale, the default is no title.

scale.cex Cex to be used for the size of the scale box.

scaleplace Location of the scale, default is "bottomright", available options are "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center".

scale.box The type of box to be drawn around the scale, "o" the default or "n" none.

cex.max.bubble Maximum size of the plotting character when maptype="bubble", the default is 2.

threshold Only relevant when maptype="values". The value below which points are shown with a "+" symbol rather than their actual value. The default is zero.

digits.text Number of decimal places to print when maptype="values", the default is 0.

cex.text Cex to be used for printed values when maptype="values". The default is 1.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩, Liz Clarke ⟨e.d.clarke@marlab.ac.uk⟩

Examples

```
#=====================================================

# The default
spacePlot()

#  Plotting numeric variables and spatial strata....
#  by ICES statistical rectangle,

rects <-c("45E2","44E3","46E4","44E0","41E1")
values <-c(234,358,127,530,442)

spacePlot(values,rects,mean,maptype="values",statrects=TRUE,landmass=TRUE,
col.land="snow2",col.coast="blue",col.text=2,)

# and by ICES area.
```

```
areas <-c("IVa","IVb","IVc","VIa","VIIa")
values <-c(234,358,127,530,442)

spacePlot(values,areas,mean,maptype="image",statrects=TRUE,landmass=TRUE,
col.land="snow2",col.coast="blue",col.text=2,scale=T,scale.title="Values by ICES area")

#=================================================
# plotting COST objects

data(cod2004ce)
# Effort by Scottish vessels for cod in 2004:
#   number of trips by ICES statistical rectangle
spacePlot(cod2004ce,"trpNum","rect",sum,statrects=TRUE,scale.title="Number of trips")
title("Effort by Scottish vessels for cod in 2004")

# Landings by Scottish vessels of cod in 2004:
#   landed weight by ICES statistical rectangle as a bubble plot
data(cod2004cl)
spacePlot(cod2004cl,"landWt","rect",sum,maptype="bubble",statrects=TRUE,
          scale.title="Landed weight (kg)")
title("Landings by Scottish vessels of cod in 2004")


# A comparison of landed weight and number of fish measured by ICES rectangle
data(cod2004cl)
data(cod2004cs)

# first plotting out landings to get the full spatial extent.
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)")

# then overlay numbers of measured fish
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")

# redoing the landings plot to show landings < sampling

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

title(sub="Landed weight and number of fish measured, cod 2004")

#=================================================
# plotting COST objects by time and temporal stratification

# FRS market sampled cod: maximum age by rect and quarter
spacePlot(cod2004cs,"age","rect",max,TimeStrata="quarter",
statrects=TRUE,nplots=4,maptype="image",landmass=TRUE,col.land="lightgrey",col.text=2)
```

`subSetGear` *Subsets a COST object by gear*

Description

Function that subsets csData, clData or ceData by gear code.

Usage

```
subSetGear(costobj,gear)
```

Arguments

`costobj`	A COST object of csDate, clData or ceData class or their validated equivalents.
`gear`	Vector of gear codes.

Details

subSetGear returns all components of the csData object for the specified gear or gears. Gears are specified as the two or three letter codes of the fishing activity category European level 4. e.g. "OTB" = Bottom otter trawl, "SSC" = Scottish seine, "DRB" = Boat dredge etc. The specified costobj must have entries for `$foCatEu5` in the hh table if a csData object, or in the cl table if a clData object.

Value

`costobj`	Sub-setted COST object.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`subSetSpp`, `subSetTrip`, `subSetProj`, `subSetTarget` for other sub-setting functions.

Examples

```
# Sub-setting the sole csData to get all "OTB" records.
data(sole)
newsoledata <-subSetGear(sole.cs,c("OTB"))
```

`subSetProj` *Subsets a COST object by project*

Description

Function that subsets csData by project code.

Usage

```
subSetProj(costobj,proj)
```

Arguments

`costobj`	A COST object either of csDate or csDataVal class.
`proj`	Vector of proj codes.

Details

subSetProj returns all components of the csData object for the specified project or projects.

Value

`costobj`	Sub-setted COST object.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`subSetSpp`, `subSetTrip`, `subSetGear`, `subSetTarget` for other sub-setting functions.

Examples

```
# Sub-setting the sole csData to get the market sampling and observer project records.
data(sole)
newsoledata <-subSetProj(sole.cs,c("MarkSamp","Obsmer"))
```

`subSetSpp` *Subsets a COST object by species*

Description

Function that subsets a csData or clData by species (or taxon).

Usage

```
subSetSpp(costobj,spp)
```

Arguments

`costobj`	A COST object either of csDate calss or clData class or their validated equivalents.
`spp`	Character string specifying the species (or taxon) to subset by. either the Scientific name or the ASFIS X3A code are accepted.

Details

spp has to be in the ASFIS list

Value

`costobj`	Sub-setted COST object.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

Examples

```
# Assuming the data set has more than one species record which this example does not....
data(cod2004cs)
# sub-setting the cod2004 FRS sampling data for cod (this only contains cod records!)
newcod2004cs <-subSetSpp(cod2004cs,"Gadus morhua")
# or alternatively
newcod2004cs <-subSetSpp(cod2004cs,"COD")
# and for Saithe (but there are no Saithe in this example data set)
pok2004cs <-subSetSpp(cod2004cs,"POK")
```

---

subSetTarget *Subsets a COST object by target assemblage*

Description

Function that subsets csData, clData or ceData by target assemblage.

Usage

```
subSetTarget(costobj,assemblage)
```

Arguments

`costobj`	A COST object of csDate, clData or ceData class or their validated equivalents.
`assemblage`	Vector of target assemblage codes.

Details

subSetTarget returns all components of the csData object for the specified target assemblage codes. Target assemblages are specified as the final three letter code of the fishing activity category European level 5. e.g. "DEF" = Demersal fish, "MOL" = Molluscs, "SPF" = Small pelagic fish etc. The specified costobj must have entries for `$foCatEu5` in the hh table if a csData object, in the cl table if a clData object or in the ce table if a ceData object.

Value

`costobj`	Sub-setted COST object.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`subSetSpp`, `subSetTrip`, `subSetProj`, `subSetGear` for other sub-setting functions.

Examples

```
# Sub-setting the sole csData to get all records for trips targeting demersal fish "DEF" and molluscs
data(sole)
newsoledata <-subSetTarget(sole.cs,c("DEF","MOL"))
```

`subSetTrip` *Subsets a COST object by trip code*

Description

Function that subsets csData by trip.

Usage

```
subSetTrip(costobj,trpCode)
```

Arguments

`costobj`	A COST object either of csDate or csDataVal class.
`trpCode`	Vector of trip codes.

Details

subSetTrip returns all components of the csData object for the specified trip or trips.

Value

`costobj`	Sub-setted COST object.

Author(s)

Alastair Pout ⟨a.pout@marlab.ac.uk⟩

See Also

`subSetSpp`, `subSetProj`, `subSetGear`, `subSetTarget` for other sub-setting functions.

Examples

```
# Sub-setting the cod2004 FRS sampling data to get only the records for two specified trips
data(cod2004cs)
newcoddata <-subSetTrip(cod2004cs,c("01020104A","01030104A"))

# Sub-setting the sole csData to get only the records for the first 4 trips
data(sole)
newsoledata <-subSetTrip(sole.cs,c(1,2,3,4))
```

`tabConsist` *Field composition over COST objects*

Description

This function gives a description of a specified field contents in various COST objects.

Usage

```
tabConsist(lTab,field,nb=FALSE)
```

Arguments

`lTab`	A list of COST objects (within the same class).
`field`	Character specifying the field to describe (`"month"` and `"quarter"` informations are also available from *hh* table in object of class *cs*).
`nb`	Logical. If `TRUE`, number of occurrence of each factor level is displayed.

Author(s)

Mathieu Merzereaud

See Also

`dfApply`

Examples

```
data(sole)
tabConsist(list(sole.cs,sole.ce,sole.cl),"area")
```

Index