Package: FLFishery (via r-universe)

November 23, 2026

Title Classes and Methods for Simpler Fleet/Fishery Modelling

Version 0.4.1

Description A set of classes and methods for modelling of fleet dynamics. Fisheries are groups of vessels sharing an effort time series, with static or changing spatio-temporal patterns in selectivity.

X-schema.org-keywords fisheries, flr, R

Depends R(>= 4.0), methods, FLCore(>= 2.6.19.9032), ggplotFL

Imports ggplot2

Suggests knitr, testthat, rmarkdown

Additional_repositories http://flr-project.org/R

License EUPL

LazyLoad Yes

LazyData No

Collate FLFishery.R generics.R classes.R accessors.R fcb.R computation.R coerce.R constructors.R methods.R getPlural.R plot.R harvest.R oem.R data.R

VignetteBuilder knitr

RoxygenNote 7.2.2

Repository https://ices-tools-dev.r-universe.dev

Date/Publication 2026-06-27 14:24:39 UTC

RemoteUrl https://github.com/flr/FLFishery

RemoteRef HEAD

RemoteSha 72c756a2148910c28b782ecd7ecceebbfe4dc09b

Contents

as.FLStock,FLBiol-method 2
FCB,ANY-method 3

fcb2int 4
FLCatch 4
FLCatches 8
FLFisheries 10
FLFishery 11
FLFisherycpp 17
guessfcb 20
harvests 20
nsfishery 22

Index **23**

`as.FLStock,FLBiol-method` *Create an* `FLStock` *object from* `FLBiol` *and* `FLFishery` *or* `FLFisheries`

Description

A coertion method that returns a `FLStock` object by combining biological information contained in `FLBiol` and catch data from one of more `FLFishery` objects.

Usage

```
## S4 method for signature 'FLBiol'
as.FLStock(
  object,
  fisheries,
  full = TRUE,
  catch = rep(1, length(fisheries)),
  ...
)
```

Arguments

`object`	An object of class `FLBiol`.
`fisheries`	An object of classes `FLFishery` or `FLFisheries`.
`full`	Should fishing mortality be computed and added? Logical.
`catch`	Vector of the same length ad `fisheries`, indicating the position of the `FLCatch` objects that refer to the `FLBiol`. Defaults to the first one along all elements.

Details

Details: Aliquam sagittis feugiat felis eget consequat.

Value

An object of class `FLStock`.

Author(s)

The FLR Team

See Also

FLCore::FLStock

Examples

```
data(nsfishery)
as.FLStock(sol, nsfleet)
```

FCB,ANY-method *Matrix of Fishery - Catch - Biol relationships*

Description

The relationships between a fishery, its catch elements and the biological populations that catch is taken from, is part of the 'fwdControl' class. When a single 'FLBiol' and 'FLFishery', or just an 'FLStock', are projected forward, this structure is constructed on the fly. Even when multiple biols and fisheries are used, a guess is made based on name matching.

Usage

```
## S4 method for signature 'ANY'
FCB(object, ...)

## S4 method for signature 'missing'
FCB(object, ...)

## S4 method for signature 'FLBiols'
FCB(object, fisheries)

## S4 method for signature 'FLFisheries'
FCB(object, biols)
```

Arguments

object	A list or vector containing a row for the matrix.
...	Further vectors to be merged into the matrix.
fisheries	An 'FLFisheries' object to extract fishery and catch names from.
biols	An 'FLBiols' object to match with 'FLCatch' elements in 'object'.

Details

But when a more complex structure is employed, the ‘FCB‘ matrix can be given to the ‘fwdControl()‘ constructor method. Elements in this matrix can be names or numbers.

If ‘FLBiols‘ and ‘FLFisheries‘ objects are passed to ‘FCB‘, a guess is made at constructing the matrix based on the names of the various list elements.

Examples

```
# 1 fishery with catches from 2 biols
FCB(c(f=1, c=1, b=2), c(f=1, c=2, b=2))
# 2 fisheries with caches from 3 biols
FCB(c(f=1, c=1, b=1), c(f=1, c=2, b=2),
  c(f=2, c=1, b=2), c(f=2, c=2, b=2),
  c(f=2, c=3, b=3))
```

`fcb2int` *fcb2int function*

Description

Internal function not for public consumption

Usage

```
fcb2int(fcb, biols, fisheries)
```

Arguments

`fcb`	The FCB matrix
`biols`	The biols
`fisheries`	The fisheries

`FLCatch` *FLCatch constructor*

Description

Make an FLCatch object.

Catch data for a single species or stock unit is handled by the `FLCatch` class. Data is separated as landings and discards by age, in numbers, with the corresponding mean weights at age.

Usage

```
FLCatch(object, ...)

catch.sel(object, ...) <- value

discards.ratio(object, ...)

lrevenue(object, ...)

## S4 method for signature 'FLCatch'
landings.n(object)

## S4 replacement method for signature 'FLCatch,numeric'
landings.n(object) <- value

## S4 replacement method for signature 'FLCatch,numeric'
landings.wt(object) <- value

## S4 replacement method for signature 'FLCatch,numeric'
discards.n(object) <- value

## S4 replacement method for signature 'FLCatch,numeric'
discards.wt(object) <- value

## S4 replacement method for signature 'FLCatch,numeric'
catch.sel(object) <- value

## S4 replacement method for signature 'FLCatch,numeric'
price(object) <- value

## S4 replacement method for signature 'FLCatch,numeric'
catch.q(object) <- value

## S4 method for signature 'FLCatch'
landings(object)

## S4 method for signature 'FLCatch'
discards(object)

## S4 method for signature 'FLCatch'
catch(object)

## S4 method for signature 'FLCatch'
landings.n(object)

## S4 method for signature 'FLCatch'
discards.n(object)
```

```
## S4 method for signature 'FLCatch'
catch.n(object)

## S4 method for signature 'FLCatch'
catch.wt(object)

## S4 method for signature 'FLQuant'
FLCatch(object, ...)

## S4 method for signature 'missing'
FLCatch(object, ...)

## S4 method for signature 'FLCatch'
lrevenue(object)

## S4 method for signature 'FLCatch'
landings.sel(object)

## S4 method for signature 'FLCatch'
discards.sel(object)

## S4 method for signature 'FLCatch'
discards.ratio(object)

## S4 method for signature 'FLCatch,missing'
plot(x, y, ...)
```

Arguments

object	Either an FLQuant (to determine the size of the FLQuant slots) or missing
...	Other things
value	Replacement value
x	FLCatch
y	missing

Details

Make an FLCatch object.

This is class is used inside FLFishery to store the catches of a single stock or species caught by that fleet.

Value

An FLCatch object

Slots

catch.q Parameters of the catchability function, (FLPar).

catch.sel Selectivity at age as proportions over fully-selected ages, (FLQuant).

desc Description of the data contents and origin, (character).

discards.n Discards at age in numbers, (FLQuant).

discards.wt Mean weight-at-age in the discards, (FLQuant).

name Name of the object, e.g. species or stock code, (character).

landings.n Landings at age in numbers, (FLQuant).

landings.wt Mean weight-at-age in the landings, (FLQuant).

price Mean price by age per unit of weight, (FLQuant).

range Ranges of age and years, plusgroup, (numeric).

Validity

Length of dimensions 2:5 All FLQuant slots must share dimensions 2 to 5.

iter **dim of length 1 or N** The 6th dimension in all FLQuant and FLPar slots must be 1 or N, where N is the same value for the whole object.

Length of dimensions 2:5 All FLQuant slots must share dimensions 2 to 5.

You can inspect the class validity function by using `getValidity(getClassDef('FLCatch'))`

Accessors

All slots in the class have accessor and replacement methods defined that allow retrieving and substituting individual slots.

The values passed for replacement need to be of the class of that slot. A numeric vector can also be used when replacing FLQuant slots, and the vector will be used to substitute the values in the slot, but not its other attributes.

Constructor

A construction method exists for this class that can take named arguments for any of its slots. All slots are then created to match the requirements of the class validity. If an unnamed FLQuant object is provided, this is used for sizing but not stored in any slot.

Methods

Methods exist for various calculations based on values stored in the class:

landings Total landings as sum on 'age' of landings.n times landings.wt.

discards Total discards as sum on 'age' of discards.n times discards.wt.

landings.sel Selectivity at age in the landings as proportions over fully-selected ages, (FLQuant).

discards.sel Selectivity at age in the discards as proportions over fully-selected ages, (FLQuant).

catch.n Catch at age in numbers as landings.n plus discards.n.

catch.wt Weighted average of landings.wt and discards.wt.

catch Total catch as sum of landings and discards.

discards.ratio Proportion at age of discards in catch.

plot Standard plot for the FLCatch class.

Author(s)

The FLR Team

Iago Mosqueira, EC JRC.

See Also

FLComp

FLCatches, FLFishery

Examples

```
data(ple4)

# EXTRACT data from FLCore ple4, fake prices
fca <- FLCatch(name='PLE', desc='All NS PLE catches',
  landings.n=landings.n(ple4), landings.wt=landings.wt(ple4),
  discards.n=discards.n(ple4), discards.wt=discards.wt(ple4),
  price=landings.wt(ple4) * 23, catch.q=FLPar(q=1),
  catch.sel=catch.sel(ple4))

# Calculations
landings(fca)

catch.n(fca)
catch.wt(fca)
```

FLCatches *FLCatches constructor*

Description

Make an FLCatches object.

This is class is used inside `FLFishery` to store the catches of all species caught by that fleet. It is not meant to be used directly.

Usage

```
FLCatches(object, ...)

## S4 method for signature 'list'
FLCatches(object)

## S4 method for signature 'missing'
FLCatches(object, ...)
```

Arguments

`object`	Either a list of FLCatch objects or missing
`...`	Other things

Details

Make an FLCatches object.

Value

An FLCatches object

Validity

Length of dimensions 2:5 All elements must be of class `FLCatch`

Length of dimensions 2, 4 and 5 All `FLQuant` slots must share dimensions 2, 4 and 5 (year, season and area).

quant must 'age' The 1st dimension in elements must be 'age'.

You can inspect the class validity function by using `getValidity(getClassDef('FLCatches'))`

Accessors

Elements in the classes can be extracted and replaced using the list subset operators,'[', '[<-', '[[' and '[[<-'.

The values passed for replacement need to be of the class FLCatch.

Constructor

A construction method exists for this class that can take named arguments for any of its elements.

Methods

Methods exist for various operations with elements stored in the class:

plot Standard plot for the FLCatches class.

Author(s)

The FLR Team

Iago Mosqueira, EC JRC.

See Also

FLComp

FLCatch, FLFishery

FLFisheries *FLFisheries constructor*

Description

Make an FLFisheries object.

This is a containuer class for FLFishery objects.

Usage

```
FLFisheries(object, ...)

## S4 method for signature 'FLFisheries'
landings(object, by = c("fishery", "catch"), sum = TRUE)

## S4 method for signature 'FLFisheries'
discards(object, by = c("fishery", "catch"), sum = TRUE)

## S4 method for signature 'FLFisheries'
catch(object, by = c("fishery", "catch"), sum = TRUE)

## S4 method for signature 'FLFisheries'
landings.n(object, pos = unique(unlist(lapply(object, names))), reduce = TRUE)

## S4 method for signature 'FLFisheries'
discards.n(object, pos = unique(unlist(lapply(object, names))), reduce = TRUE)

## S4 method for signature 'FLFisheries'
catch.n(object, pos = unique(unlist(lapply(object, names))), reduce = TRUE)

## S4 method for signature 'FLFisheries'
landings.wt(object, pos = lapply(object, names))

## S4 method for signature 'FLFisheries'
catch.wt(object, pos = unique(unlist(lapply(object, names))), reduce = TRUE)

## S4 method for signature 'list'
FLFisheries(object, desc = character(1))

## S4 method for signature 'missing'
FLFisheries(object, ...)
```

Arguments

object	Either a list of FLFishery objects or missing
...	Other things

by	Dimension to aggregate by, "fishery" or "catch".
desc	Description

Details

Make an FLFisheries object.

Value

An FLFisheries object

Accessors

Elements in the classes can be extracted and replaced using the list subset operators,'[', '[<-', '[[' and '[[<-'.

The values passed for replacement need to be of the class FLFishery.

Constructor

A construction method exists for this class that can take named arguments for any of its elements.

Author(s)

The FLR Team

Finlay Scott, EC JRC.

See Also

FLComp

FLCatch, FLFishery

FLFishery *Classes and Methods for Modeling of Fishing Fleets*

Description

FLFishery defines a set of classes to load, manipulate and combine landings discards data from fishing fleets catching individuals of one or more stocks.

Fishing fleets consisting of a number of boats operating homogeneously can be modelled using the FLFishery class. All boats in the fleet must have a common gear configuration during each time step and area (no *metiers*).

Usage

```
FLFishery(object, ...)

capacity(object, ...)

capacity(object, ...) <- value

crewshare(object, ...)

crewshare(object, ...) <- value

hperiod(object, ...)

hperiod(object, ...) <- value

orevenue(object, ...)

orevenue(object, ...) <- value

## S4 method for signature 'FLFishery'
capacity(object)

## S4 replacement method for signature 'FLFishery,FLQuant'
capacity(object) <- value

## S4 replacement method for signature 'FLFishery,numeric'
capacity(object) <- value

## S4 method for signature 'FLFishery'
hperiod(object)

## S4 replacement method for signature 'FLFishery,FLQuant'
hperiod(object) <- value

## S4 replacement method for signature 'FLFishery,numeric'
hperiod(object) <- value

## S4 method for signature 'FLFishery,ANY'
effort(object, compute = TRUE)

## S4 replacement method for signature 'FLFishery,FLQuant'
effort(object) <- value

## S4 method for signature 'FLFishery'
vcost(object, compute = TRUE)

## S4 replacement method for signature 'FLFishery,FLQuant'
vcost(object) <- value
```

```
## S4 method for signature 'FLFishery'
fcost(object, compute = TRUE)

## S4 replacement method for signature 'FLFishery,FLQuant'
fcost(object) <- value

## S4 replacement method for signature 'FLFishery,numeric'
fcost(object) <- value

## S4 method for signature 'FLFishery'
orevenue(object)

## S4 replacement method for signature 'FLFishery,FLQuant'
orevenue(object) <- value

## S4 replacement method for signature 'FLFishery,numeric'
orevenue(object) <- value

## S4 method for signature 'FLFishery'
crewshare(object, compute = TRUE)

## S4 replacement method for signature 'FLFishery,predictModel'
crewshare(object) <- value

## S4 method for signature 'FLFishery'
landings(object, catch = seq(object))

## S4 method for signature 'FLFishery'
discards(object, catch = seq(object))

## S4 method for signature 'FLFishery'
catch(object, catch = seq(object))

## S4 method for signature 'FLFishery'
landings.n(object, pos = names(object))

## S4 method for signature 'FLFishery'
discards.n(object, pos = names(object))

## S4 method for signature 'FLFishery'
catch.n(object, pos = names(object))

## S4 method for signature 'FLFishery'
landings.wt(object, pos = names(object))

## S4 method for signature 'FLFishery'
catch.wt(object, pos = names(object))
```

```
## S4 method for signature 'list'
FLFishery(object, ...)

## S4 method for signature 'FLCatch'
FLFishery(object, ...)

## S4 method for signature 'missing'
FLFishery(object, ...)

## S4 method for signature 'FLFishery,ANY,missing,ANY'
x[i]

## S4 replacement method for signature 'FLFishery,numeric,missing,FLCatch'
x[[i]] <- value

## S4 replacement method for signature 'FLFishery,character,missing,FLCatch'
x[[i]] <- value

## S4 method for signature 'FLFishery'
summary(object)

## S4 method for signature 'FLFishery'
lrevenue(object)

## S4 method for signature 'FLFishery'
revenue(object)

## S4 method for signature 'FLFishery'
cost(object)

## S4 method for signature 'FLFishery'
profit(object)

## S4 method for signature 'FLFishery'
ccost(object)

## S4 method for signature 'FLFishery'
propagate(object, iter, fill.iter = TRUE)

## S4 method for signature 'FLFishery'
iter(obj, iter)

## S4 replacement method for signature 'FLFishery,FLFishery'
iter(object, iter) <- value
```

Arguments

object	Object on which to assign value

...	Other things
value	Object to assign
compute	Carry out formula calculation (TRUE) or return full slot (FALSE).
x	Object to be subset
i	Element to be extracted, by name (character) or position (numeric).
iter	Position (numeric) or name (character) of the iter(s) to be extracted (iter), or number of iters to be created (propagate).
fill.iter	Should the object content be copied across the new iters, logical.
obj	Object on which to apply method

Details

Fisheries are defined here as a group of vessels catching from a number of stocks

What do you say what I can ever do for you What are we gonna do to pass the time What do you care when you find that life's unfair Equality is just a state of mind Believe whatever is right what's right for you tonight You know where to draw the line

Slots

FLFishery objects inherit from FLCatches with a number of slots added.

.Data The list of FLCatch object with catch data per stock, (FLCatches).

name Name of the object, e.g. species or stock code, (character).

desc Description of the data contents and origin, (character).

range Ranges of age and years, plusgroup, (numeric).

capacity Number of boats in the fleet, (FLQuant).

effort Mean effort per boat applied by the fleet, (FLQuant).

hperiod Start and end of fishing within each time step, as proportions. An FLQuant object with dimnames 'quant=c('start', 'end')' in the first dimension.

vcost Variable costs per unit of effort, (FLQuant).

fcost Variable costs per unit of effort, (FLQuant).

orevenue Revenues obtained from sources other than landings, (FLQuant).

crewshare Formula, parameter values and inputs to calculate the crew costs, (predictModel).

Validity

You can inspect the class validity function by using getValidity(getClassDef('FLFishery'))

Accessors

All slots in the class have accessor and replacement methods defined that allow retrieving and substituting individual slots.

The values passed for replacement need to be of the class of that slot. A numeric vector can also be used when replacing FLQuant slots, and the vector will be used to substitute the values in the slot, but not its other attributes.

Constructor

A construction method exists for this class that can take named arguments for any of its slots. All slots are then created to match the requirements of the class validity. If an unnamed `FLQuant` object is provided, this is used for sizing but not stored in any slot.

Methods

Methods exist for various calculations based on values stored in the class:

ccost Calculate the total crew costs by evaluating the formula in `crewshare`.

cost Total costs, calculated.

lrevenue .

revenue .

profit .

landings .

discards .

catch .

catch.n .

catch.wt .

harvest .

Author(s)

Iago Mosqueira (EC JRC) <iago.mosqueira@ec.europa.eu>

Maintainer: Iago Mosqueira (EC JRC) <iago.mosqueira@ec.europa.eu>

Iago Mosqueira, EC JRC.

See Also

See `vignette("FLFishery", package = "FLFishery")` for an overview of the package.

FLCatches

Examples

```
## Not run: FLFishery()

data(ple4)
FLFishery(PLE=as(ple4, 'FLCatch'))
```

FLFisherycpp *An internal class for homogeneous fishing fleets*

Description

The same as the FLFishery class but all predictModel slots have been turned into FLQuant objects. The class is used for passing to C++ FLR objects.

Usage

```
## S4 method for signature 'FLFisherycpp'
capacity(object)

## S4 replacement method for signature 'FLFisherycpp,FLQuant'
capacity(object) <- value

## S4 replacement method for signature 'FLFisherycpp,numeric'
capacity(object) <- value

## S4 method for signature 'FLFisherycpp'
hperiod(object)

## S4 replacement method for signature 'FLFisherycpp,FLQuant'
hperiod(object) <- value

## S4 replacement method for signature 'FLFisherycpp,numeric'
hperiod(object) <- value

## S4 method for signature 'FLFisherycpp'
orevenue(object)

## S4 replacement method for signature 'FLFisherycpp,FLQuant'
orevenue(object) <- value

## S4 replacement method for signature 'FLFisherycpp,numeric'
orevenue(object) <- value

## S4 method for signature 'FLFisherycpp,ANY'
effort(object)

## S4 replacement method for signature 'FLFisherycpp,FLQuant'
effort(object) <- value

## S4 replacement method for signature 'FLFisherycpp,numeric'
effort(object) <- value

## S4 method for signature 'FLFisherycpp'
```

```
vcost(object)

## S4 replacement method for signature 'FLFisherycpp,FLQuant'
vcost(object) <- value

## S4 replacement method for signature 'FLFisherycpp,numeric'
vcost(object) <- value

## S4 method for signature 'FLFisherycpp'
fcost(object)

## S4 replacement method for signature 'FLFisherycpp,FLQuant'
fcost(object) <- value

## S4 replacement method for signature 'FLFisherycpp,numeric'
fcost(object) <- value

## S4 method for signature 'FLFisherycpp'
crewshare(object)

## S4 replacement method for signature 'FLFisherycpp,FLQuant'
crewshare(object) <- value

## S4 replacement method for signature 'FLFisherycpp,numeric'
crewshare(object) <- value
```

Arguments

object	Object to extract or operate on
value	Replacement value

Details

What have you done what's in your mind what do you need Where shall we go to let it out What have you seen, we don't know where you've been Life so often blows your candle out Believe in what is right, what's right for you tonight Who knows what the fuck it's all about

Slots

FLFisherycpp objects inherit from FLCatches woth a number of slots added.

.Data The list of FLCatch object with catch data per stock, (FLCatches).

name Name of the object, e.g. species or stock code, (character).

desc Description of the data contents and origin, (character).

range Ranges of age and years, plusgroup, (numeric).

capacity Number of boats in the fleet, (FLQuant).

effort Mean effort per boat applied by the fleet, (FLQuant).

hperiod Start and end of fishing within each time step, as proportions. An `FLQuant` object with dimnames 'quant=c('start', 'end')' in the first dimension.

vcost Variable costs per unit of effort, (`FLQuant`).

fcost Variable costs per unit of effort, (`FLQuant`).

orevenue Revenues obtained from sources other than landings, (`FLQuant`).

crewshare Formula, parameter values and inputs to calculate the crew costs, (`FLQuant`).

Validity

You can inspect the class validity function by using `getValidity(getClassDef('FLFishery'))`

Accessors

All slots in the class have accessor and replacement methods defined that allow retrieving and substituting individual slots.

The values passed for replacement need to be of the class of that slot. A numeric vector can also be used when replacing FLQuant slots, and the vector will be used to substitute the values in the slot, but not its other attributes.

Constructor

A construction method exists for this class that can take named arguments for any of its slots. All slots are then created to match the requirements of the class validity. If an unnamed `FLQuant` object is provided, this is used for sizing but not stored in any slot.

Methods

Methods exist for various calculations based on values stored in the class:

ccost Calculate the total crew costs by evaluating the formula in `crewshare`.

cost Total costs, calculated.

lrevenue .

revenue .

profit .

landings .

discards .

catch .

catch.n .

catch.wt .

harvest .

Author(s)

Iago Mosqueira, EC JRC.

See Also

FLCatches

Examples

```
data(ple4)
FLFishery(PLE=as(ple4, 'FLCatch'))
```

---

guessfcb *Generate an FCB matrix from FLBiols and FLFisheries*

Description

Tries to generate FCB matrix based on names. Internal function. Ignore.

Usage

```
guessfcb(biols, fisheries)
```

Arguments

biols	The FLBiols.
fisheries	The FLFisheries.

---

harvests *Methods to calculate fishing mortalities*

Description

Fishing mortalities and harvest rates can be calculated for a combination of FLBiol and FLFishery/FLFisheries using the harvest() method.

Usage

```
harvests(object, catches, ...)

## S4 method for signature 'FLBiol,FLFisheries'
harvests(object, catches, fcb = rep(1, length(catches)), units = c("f", "hr"))

## S4 method for signature 'FLBiol,FLFishery'
harvest(object, catch, fcb = 1)

## S4 method for signature 'FLBiol,FLFisheries'
harvest(object, catch, fcb = 1)
```

```
## S4 method for signature 'FLBiols,FLFisheries'
harvest(object, catch, fcb = FCB(object, catch))

## S4 method for signature 'FLBiols,FLFishery'
harvest(object, catch, fcb = FCB(object, FLFisheries(catch)))

## S4 method for signature 'FLBiol,FLCatch'
harvest(object, catch)

## S4 method for signature 'FLBiol'
fbar(
  object,
  fisheries,
  range = dims(object)[c("min", "max")],
  minfbar = range$min,
  maxfbar = range$max,
  ...
)

## S4 method for signature 'FLBiols'
fbar(
  object,
  fisheries,
  range = lapply(object, function(x) dims(x)[c("min", "max")]),
  ...
)
```

Arguments

object	Object containing population abundances, of class *FLBiol*.
catches	Object containing catches in number of the population represented by object, class *FLFisheries*.
...	Other things
fcb	A vector indicating the correspondance, by position of name, between *object* and the *FLCatch* elements inside *catches*, and of the same length.
units	Should output be in terms of fishing mortaloty ('f') or harvest rate ('hr').
catch	Object containing catches in number of the population represented by object, class *FLFishery*.

Details

The calculated fishing mortalities, or havest rates, are returned, in the case of *harvest*, disaggregated by fishery, as an *FLQuants* list. For a single *FLFishery* object, a single *FLQuant* is obtained.

Value

An object of class *FLQuant* or *FLQuants*.

Author(s)

The FLR Team

See Also

FLCore::harvest()

Examples

```
data(nsfishery)
harvests(ple, nsfleet)
harvest(ple, nsfleet[["bt"]], fcb="ple")
harvest(ple, nsfleet)
data(nsfishery)
bis <- FLBiols(ple=ple, sol=sol)
harvest(bis, nsfleet)
harvest(bis, nsfleet[[1]])
harvest(ple, nsfleet[["bt"]][["ple"]])
fbar(ple, fisheries=nsfleet, minfbar=3, maxfbar=6)
fbar(ple, fisheries=nsfleet[["bt"]], minfbar=3, maxfbar=6)
fbar(FLBiols(ple=ple, sol=sol), nsfleet)
```

---

nsfishery — *Simulated fisheries data for fisheries exploiting North sea plaice and sole.*

Description

An example data for two fisheries (*bt* and *gn*) exploiting two stocks, *ple* and *sol*, represented by two objects of class *FLBiol*.

Usage

```
nsfleet

ple

sol
```

Format

The objects of classes FLFisheries (nsfleet) and FLBiol (sol, ple).

An object of class `FLBiol` of length 1.

An object of class `FLBiol` of length 1.

Index

∗ **classes**
FLCatch, 4
FLCatches, 8
FLFisheries, 10
FLFishery, 11
FLFisherycpp, 17
∗ **datasets**
nsfishery, 22
∗ **methods**
as.FLStock,FLBiol-method, 2
harvests, 20
[,FLFishery,ANY,missing,ANY-method *(FLFishery)*, 11
[[<-,FLFishery,character,missing,FLCatch-method *(FLFishery)*, 11
[[<-,FLFishery,numeric,missing,FLCatch-method *(FLFishery)*, 11

as.FLStock,FLBiol-method, 2

capacity *(FLFishery)*, 11
capacity,FLFishery-method *(FLFishery)*, 11
capacity,FLFisherycpp-method *(FLFisherycpp)*, 17
capacity-method *(FLFishery)*, 11
capacity<- *(FLFishery)*, 11
capacity<-,FLFishery,FLQuant-method *(FLFishery)*, 11
capacity<-,FLFishery,numeric-method *(FLFishery)*, 11
capacity<-,FLFisherycpp,FLQuant-method *(FLFisherycpp)*, 17
capacity<-,FLFisherycpp,numeric-method *(FLFisherycpp)*, 17
capacity<--method *(FLFishery)*, 11
catch,FLCatch-method *(FLCatch)*, 4
catch,FLFisheries-method *(FLFisheries)*, 10
catch,FLFishery-method *(FLFishery)*, 11
catch.n,FLCatch-method *(FLCatch)*, 4
catch.n,FLFisheries-method *(FLFisheries)*, 10
catch.n,FLFishery-method *(FLFishery)*, 11
catch.q,FLCatch-method *(FLCatch)*, 4
catch.q<-,FLCatch,FLPar-method *(FLCatch)*, 4
catch.q<-,FLCatch,numeric-method *(FLCatch)*, 4
catch.sel,FLCatch-method *(FLCatch)*, 4
catch.sel<- *(FLCatch)*, 4
catch.sel<-,FLCatch,FLQuant-method *(FLCatch)*, 4
catch.sel<-,FLCatch,numeric-method *(FLCatch)*, 4
catch.sel<--method *(FLCatch)*, 4
catch.wt,FLCatch-method *(FLCatch)*, 4
catch.wt,FLFisheries-method *(FLFisheries)*, 10
catch.wt,FLFishery-method *(FLFishery)*, 11
ccost,FLFishery-method *(FLFishery)*, 11
cost,FLFishery-method *(FLFishery)*, 11
crewshare *(FLFishery)*, 11
crewshare,FLFishery-method *(FLFishery)*, 11
crewshare,FLFisherycpp-method *(FLFisherycpp)*, 17
crewshare-method *(FLFishery)*, 11
crewshare<- *(FLFishery)*, 11
crewshare<-,FLFishery,predictModel-method *(FLFishery)*, 11
crewshare<-,FLFisherycpp,FLQuant-method *(FLFisherycpp)*, 17
crewshare<-,FLFisherycpp,numeric-method *(FLFisherycpp)*, 17
crewshare<--method *(FLFishery)*, 11

desc,FLCatch-method *(FLCatch)*, 4

desc<-,FLCatch,character-method (FLCatch), 4
discards,FLCatch-method (FLCatch), 4
discards,FLFisheries-method (FLFisheries), 10
discards,FLFishery-method (FLFishery), 11
discards.n,FLCatch-method (FLCatch), 4
discards.n,FLFisheries-method (FLFisheries), 10
discards.n,FLFishery-method (FLFishery), 11
discards.n<-,FLCatch,FLQuant-method (FLCatch), 4
discards.n<-,FLCatch,numeric-method (FLCatch), 4
discards.ratio (FLCatch), 4
discards.ratio,FLCatch-method (FLCatch), 4
discards.ratio-method (FLCatch), 4
discards.sel,FLCatch-method (FLCatch), 4
discards.wt,FLCatch-method (FLCatch), 4
discards.wt<-,FLCatch,FLQuant-method (FLCatch), 4
discards.wt<-,FLCatch,numeric-method (FLCatch), 4

effort,FLFishery,ANY-method (FLFishery), 11
effort,FLFisherycpp,ANY-method (FLFisherycpp), 17
effort<-,FLFishery,FLQuant-method (FLFishery), 11
effort<-,FLFisherycpp,FLQuant-method (FLFisherycpp), 17
effort<-,FLFisherycpp,numeric-method (FLFisherycpp), 17

fbar,FLBiol-method (harvests), 20
fbar,FLBiols-method (harvests), 20
FCB,ANY-method, 3
FCB,FLBiols-method (FCB,ANY-method), 3
FCB,FLFisheries-method (FCB,ANY-method), 3
FCB,missing-method (FCB,ANY-method), 3
fcb2int, 4
fcost,FLFishery-method (FLFishery), 11
fcost,FLFisherycpp-method (FLFisherycpp), 17
fcost<-,FLFishery,FLQuant-method (FLFishery), 11
fcost<-,FLFishery,numeric-method (FLFishery), 11
fcost<-,FLFisherycpp,FLQuant-method (FLFisherycpp), 17
fcost<-,FLFisherycpp,numeric-method (FLFisherycpp), 17
FLCatch, 4, *9*, *11*
FLCatch,FLQuant-method (FLCatch), 4
FLCatch,missing-method (FLCatch), 4
FLCatch-class (FLCatch), 4
FLCatch-method (FLCatch), 4
FLCatch-methods (FLCatch), 4
FLCatches, 8, *8*, *16*, *20*
FLCatches,list-method (FLCatches), 8
FLCatches,missing-method (FLCatches), 8
FLCatches-class (FLCatches), 8
FLCatches-methods (FLCatches), 8
FLComp, *8*, *9*, *11*
FLCore::harvest(), *22*
FLFisheries, 10
FLFisheries,list-method (FLFisheries), 10
FLFisheries,missing-method (FLFisheries), 10
FLFisheries-class (FLFisheries), 10
FLFisheries-methods (FLFisheries), 10
FLFishery, *8*, *9*, 11, *11*
FLFishery,FLCatch-method (FLFishery), 11
FLFishery,list-method (FLFishery), 11
FLFishery,missing-method (FLFishery), 11
FLFishery-class (FLFishery), 11
FLFishery-method (FLFishery), 11
FLFishery-methods (FLFishery), 11
FLFisherycpp, 17
FLFisherycpp-class (FLFisherycpp), 17
FLFisherycpp-methods (FLFisherycpp), 17

guessfcb, 20

harvest (harvests), 20
harvest,FLBiol,FLCatch-method (harvests), 20
harvest,FLBiol,FLFisheries-method (harvests), 20
harvest,FLBiol,FLFishery-method (harvests), 20

harvest,FLBiols,FLFisheries-method (harvests), 20
harvest,FLBiols,FLFishery-method (harvests), 20
harvests, 20
harvests,FLBiol,FLFisheries-method (harvests), 20
harvests-method (harvests), 20
hperiod (FLFishery), 11
hperiod,FLFishery-method (FLFishery), 11
hperiod,FLFisherycpp-method (FLFisherycpp), 17
hperiod-method (FLFishery), 11
hperiod<- (FLFishery), 11
hperiod<-,FLFishery,FLQuant-method (FLFishery), 11
hperiod<-,FLFishery,numeric-method (FLFishery), 11
hperiod<-,FLFisherycpp,FLQuant-method (FLFisherycpp), 17
hperiod<-,FLFisherycpp,numeric-method (FLFisherycpp), 17
hperiod<--method (FLFishery), 11

iter,FLFishery-method (FLFishery), 11
iter<-,FLFishery,FLFishery-method (FLFishery), 11

landings,FLCatch-method (FLCatch), 4
landings,FLFisheries-method (FLFisheries), 10
landings,FLFishery-method (FLFishery), 11
landings.n,FLCatch-method (FLCatch), 4
landings.n,FLFisheries-method (FLFisheries), 10
landings.n,FLFishery-method (FLFishery), 11
landings.n<-,FLCatch,FLQuant-method (FLCatch), 4
landings.n<-,FLCatch,numeric-method (FLCatch), 4
landings.sel,FLCatch-method (FLCatch), 4
landings.wt,FLCatch-method (FLCatch), 4
landings.wt,FLFisheries-method (FLFisheries), 10
landings.wt,FLFishery-method (FLFishery), 11
landings.wt<-,FLCatch,FLQuant-method (FLCatch), 4
landings.wt<-,FLCatch,numeric-method (FLCatch), 4
lrevenue (FLCatch), 4
lrevenue,FLCatch-method (FLCatch), 4
lrevenue,FLFishery-method (FLFishery), 11
lrevenue-method (FLCatch), 4

name,FLCatch-method (FLCatch), 4
name<-,FLCatch,character-method (FLCatch), 4
nsfishery, 22
nsfleet (nsfishery), 22

orevenue (FLFishery), 11
orevenue,FLFishery-method (FLFishery), 11
orevenue,FLFisherycpp-method (FLFisherycpp), 17
orevenue-method (FLFishery), 11
orevenue<- (FLFishery), 11
orevenue<-,FLFishery,FLQuant-method (FLFishery), 11
orevenue<-,FLFishery,numeric-method (FLFishery), 11
orevenue<-,FLFisherycpp,FLQuant-method (FLFisherycpp), 17
orevenue<-,FLFisherycpp,numeric-method (FLFisherycpp), 17
orevenue<--method (FLFishery), 11

ple (nsfishery), 22
plot,FLCatch,missing-method (FLCatch), 4
price,FLCatch-method (FLCatch), 4
price<-,FLCatch,FLQuant-method (FLCatch), 4
price<-,FLCatch,numeric-method (FLCatch), 4
profit,FLFishery-method (FLFishery), 11
propagate,FLFishery-method (FLFishery), 11

range,FLCatch-method (FLCatch), 4
range<-,FLCatch,numeric-method (FLCatch), 4
revenue,FLFishery-method (FLFishery), 11

sol (nsfishery), 22

summary,FLFishery-method *(FLFishery)*, 11

vcost,FLFishery-method *(FLFishery)*, 11
vcost,FLFisherycpp-method
(FLFisherycpp), 17
vcost<-,FLFishery,FLQuant-method
(FLFishery), 11
vcost<-,FLFisherycpp,FLQuant-method
(FLFisherycpp), 17
vcost<-,FLFisherycpp,numeric-method
(FLFisherycpp), 17