Package: FLjjm (via r-universe)

June 9, 2026

Title Running the JJM Stock Assessment Model Inside the MSE FLR System

Version 0.3.5

AuthorR c(person(given = 'Iago', family = 'Mosqueira', role = c('aut', 'cre'), email = 'iago.mosqueira@wur.nl'), person(given = 'Karolina', family = 'Molla Gazi', role = c('aut'), email = 'karolina.mollagazi@wur.nl'))

Description Runs the JJM stock assessment model for Chilean Jack Mackerel inside the MSE system of FLR's mse package.

Depends R (>= 3.5.0), methods, FLCore, ggplotFL, jjmR (>= 1.2018.02)

Imports mse, FLFishery, FLasher, data.table, foreach, progressr

Suggests testthat, knitr, rmarkdown

Additional_repositories https://sprfmo.r-universe.dev, https://flr.r-universe.dev

VignetteBuilder knitr

Encoding UTF-8

License EUPL

LazyLoad Yes

LazyData No

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

NeedsCompilation no

Author Iago Mosqueira [aut, cre], Karolina Molla Gazi [aut]

Maintainer Iago Mosqueira <iago.mosqueira@wur.nl>

Repository https://jimianelli.r-universe.dev

Date/Publication 2026-06-09 07:13:50 UTC

RemoteUrl https://github.com/SPRFMO/FLjjm

RemoteRef HEAD

RemoteSha 15abe9e47b4e948467d1f02789422d19947f38e5

Contents

.combinejjmsout *Combine Fisheries Model Results*

Description

Combines results from two sources of fisheries model runs, handling both single and multiple stock scenarios. It combines data such as stock numbers, harvest, data, and control.

Usage

```
.combinejjmsout(x, y)
```

Arguments

x	First set of fisheries model results.
y	Second set of fisheries model results.

Value

A combined list of fisheries model results.

AgeToLengthComp *Convert Age Composition to Length Composition*

Description

This function converts age composition data to length composition data based on provided length-height (lh) relationships, selectivity (S_a), and other parameters.

Usage

```
AgeToLengthComp(lh, S_a, tyears, N_at, comp_sample, sample_type = "catch")
```

Arguments

lh	A data frame or list containing length-height relationship data.
S_a	Matrix of selectivity at age.
tyears	Total number of years in the data.
N_at	Matrix of numbers at age over time.
comp_sample	Vector of sample sizes for each year.
sample_type	Type of sample, default is 'catch'. Can be 'catch' or other types.

Value

A list containing matrices: probabilities being in a length bin given age (plba), probabilities being harvested at an age (page), probabilities of sampling a given length bin (plb), and length frequencies (LF).

build *Build Functions for FLjjm Package*

Description

These functions are used to build various FLR (Fisheries Library in R) objects from the output of the JJMS (Just Another Management Strategy) stock assessment model. The functions handle different types of outputs including biological, fishery, stock, and index data.

Details

The primary functions include:

- `buildFLBjjm`: Constructs an FLBiol object from jjm.output.
- `buildFLBsjjm`: Builds multiple FLBiol objects from jjm.output.
- `buildFLFsjjm`: Creates FLFisheries objects from jjm.output.
- `buildFLIsjjm`: Generates FLIndices objects from jjm.output.
- `buildFLRPsjjm`: Constructs FLPar objects with reference points from jjm.output.
- `buildFLSojjm`: Builds FLStock objects from jjm.output considering areas.
- `buildFLSjjm`: Constructs FLStock objects from jjm.output.
- `buildFLSsjjm`: Creates multiple FLStock objects from jjm.output.

Each function takes a jjm.output object as input, which is the result of running the JJMS model. They process this output into various FLR objects for further analysis and use within the FLR framework.

Author(s)

Iago Mosqueira (WMR) <iago.mosqueira@wur.nl>

See Also

`FLBiol`, `FLFisheries`, `FLIndices`, `FLStock`

buildFLBjjm *Build an FLBiol from a jjm.output object*

Description

Build an FLBiol from a jjm.output object

Usage

```
buildFLBjjm(out, stock = 1, name = "CJM")
```

Arguments

out	A *jjm.output* object, as returned by readJJM.
stock	Stock to extract, of relevance on 2-stock model runs, *numeric*.
spwn	Proportion of the year when spawning takes place.

Value

An object of class FLBiol.

See Also

FLCore::FLBiol

---

buildFLFsjjm *Build FLFisheries from JJMS Model Output*

Description

Constructs an FLFisheries object from a JJMS model output, processing data for each fishery and creating corresponding FLCatch and FLFishery objects.

Usage

```
buildFLFsjjm(out, stock = 1)
```

Arguments

out	JJMS model output object.
stock	The stock number to extract fisheries data for, default is 1.

Value

An FLFisheries object constructed from the JJMS model output.

`buildFLIsjjm` *Construct FLIndices from a jjm.output object*

Description

Transform a single-run jjm.output object into an `FLIndices` collection. This lower-level builder is used internally by `readFLIsjjm` and other wrapper helpers. It maps the jjm output structure into FLIndex objects including selectivity, index q and time ranges.

This function creates an `FLIndices`, a list of `FLIndexBiomass` objects, each of them containing the data and estimates of a single index of abundance. Slots on each `FLIndexBiomass` object are filled with the following information:

- index: `data$Index`.
- index.var: `data$Indexerr`.
- catch.n: `data$Ipropage`.
- catch.wt: `data$Iwtatage`.
- effort: Empty.
- sel.pattern: `output@sel_ind_i`, where `i` refers to the index number.
- index.q: `output@q_i`, where `i` refers to the index number.
- range 'startf' and 'endf': (`data$Imonths` - 1) / 12.

Usage

```
buildFLIsjjm(out)
```

Arguments

`out`	A one or two-stock, single run, jjm.output object.
`stock`	Numeric. Stock index to process; for single-stock runs this is 1. Default is 1.
`stock=1`	Stock for which indices are to be loaded.

Details

The function expects the jjm output to include objects named `info`, `data`, `output`, `control` and `parameters`. It will extract indices, their selectivity patterns (sel.pattern), catchability (index.q) and transform start and end months into fractional positions within a year.

Value

An `FLIndices` object built from the provided output.

An `FLIndices` object.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

readFLIsjjm

Examples

```
## Not run:
mod <- jjmR::readJJM("h1_1.07", path = system.file("ext-data","single_stock", package="FLjjm"))
idx <- buildFLIsjjm(mod)

## End(Not run)
idx <- readFLIsjjm("h1_1.07",
  path = system.file("ext-data", "single_stock", package="FLjjm"))
summary(idx)
plot(idx)
```

---

buildFLRPsjjm *Build reference points (FLPar) from jjm.output*

Description

Create an FLPar holding reference points such as MSY, SB0, SBMSY and FMSY using the output slot msy_mt in the jjm output. Units are set to match the package conventions (e.g. "1000 t" for biomass-based quantities and "f" for fishing mortality).

An object of class *FLPar* is created from the information in a jjm.output object and for a given stock, if more than one is present. The function currently extracts the following reference points from the msy_mt data.frame in the output element of jjm.output:

- MSY: Maximum Sustainable Yield, in thousands of tonnes.
- SB0: Virgin spawning biomass, in thousands of tonnes.
- SBmsy: Spawning biomass at MSY, in thousands of tonnes.
- Fmsy: Fishing mortality at MSY.

Usage

```
buildFLRPsjjm(out, stock = 1)
```

Arguments

out	A jjm.output object (single-run) or a wrapper list typically returned by jjmR::readJJM.
stock	Numeric. Stock index to select the correct element in the output list when multiple stocks are present. Default is 1.
name	Name of the ctl model file, *character*.
path	Path ot the model folder structure, *character*.
stock=1	Stock to extract, of relevance on 2 stocks model runs, *numeric*.

Details

The function reads the last row of the `msy_mt` table in the output and maps the columns msy, bzero, bmsy and fmsy to the corresponding FLPar elements. The units are set explicitly in the returned object.

Value

An `FLPar` with named reference points: MSY, SB0, SBMSY and FMSY.

An object of class FLPar.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

`jjmR::readJJM`

jjmR::readJJM FLCore::FLBiol buildFLBjjm

Examples

```
## Not run:
mod <- jjmR::readJJM("h1_1.07", path = system.file("ext-data","single_stock", package="FLjjm"))
rps <- buildFLRPsjjm(mod)
rps

## End(Not run)
rps <- readFLRPsjjm(name = "h1_1.07",
  path = system.file("ext-data", "single_stock", package="FLjjm"))
summary(rps)
```

`buildFLSjjm` *Build FLStock from jjm.output*

Description

Build a standard `FLStock` object (without explicit area breakdown) from jjm.output. The returned object contains landings, stock numbers, weights, natural mortality, maturity, and other standard FLStock slots.

Arguments

`out`	A jjm.output object (single-run) or similar list.
`stock`	Numeric. Stock index to extract (default 1).
`name`	Character. Optional name for the resulting FLStock. Default "CJM".

Details

The function constructs age and year dimnames from the input metadata (`data$age` and `data$year`) and fills relevant slots using the output matrices such as `output$N` for stock.n and `output$wt_a_pop` for stock weights. Units are set according to the package conventions (e.g. "1e6" for counts where appropriate, or "1000 t" for biomass-based numbers).

Value

An `FLStock` object constructed from the jjm model output.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

`buildFLSojjm`

Examples

```
## Not run:
mod <- jjmR::readJJM("h1_1.07", path = system.file("ext-data","single_stock", package="FLjjm"))
stk <- buildFLSjjm(mod)

## End(Not run)
```

---

buildFLSojjm *Build FLStock with areas from jjm.output*

Description

Convert a jjm.output into an `FLStock` object taking into account area-specific outputs when available. The function fills standard stock slots (landings, stock.n, stock.wt, m, mat, etc.) using the model outputs and sets appropriate dimnames for ages and years.

Arguments

`out`	A jjm.output object (single-run) or the list structure returned by `jjmR::readJJM`.
`stock`	Numeric. Index of the stock to build (default 1).
`name`	Character. Optional name to assign to the returned `FLStock`.

Details

The function expects the jjm output to contain lists named `info`, `data`, `output`, `control` and `parameters`. Age and year dimensions are derived from `data$age` and `data$year` ranges respectively. Fisheries information, if present, will be used to create catch objects per area.

Value

An FLStock object reflecting the model's output for the requested stock, including age and year dimensions derived from the model input metadata.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

buildFLSjjm, buildFLSsjjm

Examples

```
## Not run:
mod <- jjmR::readJJM("h1_1.07", path = system.file("ext-data","single_stock", package="FLjjm"))
stk <- buildFLSojjm(mod)
summary(stk)

## End(Not run)
```

---

buildFLSsjjm *Build multiple FLStock objects from jjm.output*

Description

Create a named list of FLStock objects for each stock present in a multi-stock jjm.output. This is a convenience wrapper that calls buildFLSjjm for each stock and organizes the results into a list, assigning names where available.

This function creates a collection of FLStock objects from a jjm.output object. It is particularly useful for multi-stock assessments.

Usage

```
buildFLSsjjm(out)
```

Arguments

out	A jjm.output object, typically the output of a stock assessment model.

Value

A named list of FLStock objects, one per stock present in the input.

An FLStocks object containing multiple FLStock objects.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

buildFLSjjm

FLStock, FLStocks

Examples

```
## Not run:
mod <- jjmR::readJJM("multi_stock_run", path = "/path/to/run")
stks <- buildFLSsjjm(mod)

## End(Not run)
```

---

buildjjmctl *Updating dat and ctl file lists from FLstock(s) and FLIndices*

Description

Updating dat and ctl file lists from FLstock(s) and FLIndices

Usage

```
buildjjmctl(stk, idx, dat, ctl, ...)
```

Arguments

stk	TODO:description
idx	TODO:description
dat	TODO:description
ctl	TODO:description

Value

TODO:description

Examples

```
data(cjmstk)
ctl <- buildjjmctl(stk, idx, mod$data, mod$control)
data(cjmstks)
ctl <- buildjjmctl(stks, idxs, mods$data, mods$control)
```

catch,FLStocks-method *Calculate Total Catch across FLStocks*

Description

This method calculates the total catch by summing the catch (accounting for area sums) of each FLStock object within an FLStocks object.

Usage

```
## S4 method for signature 'FLStocks'
catch(object)
```

Arguments

object	An FLStocks object.

Value

A numeric value representing the total catch.

cjm.oem *Observation Error Model wrapper for CJM within MSE workflows*

Description

Run the CJM Observation Error Model (OEM) for a given FLStock and observation configuration. This function prepares inputs, calls the sampling.oem helper and post-processes results to match the expected structure for later MSE steps.

Arguments

stk	An FLStock object used as the operating model.
deviances	A list or object with deviance specifications for the OEM sampling routine.
observations	List containing observation specifications (e.g. indices and their sampling properties) used by the OEM.
stability	Numeric. Stability parameter for the OEM (default 1).
wts	Logical. Whether to use weights (default TRUE).
jjms	Logical. Whether to run the jjms-specific OEM branch (default TRUE).
F3sel	(internal) Selection pattern for F3 (passed to the sampler).
args	List. Additional arguments controlling dimension settings such as dy, frq and other temporal helpers used by the OEM wrapper.
tracking	Logical or list. Optional tracking and diagnostic settings.

Details

This function is tailored for the CJM/OEM integration inside the MSE flow used by the package. It sets up the temporal dimension arguments (using `dy` and `frq` from `args`), calls `sampling.oem` and performs light post-processing such as dropping data columns not required for downstream routines.

Value

A list with sampled observation time series and diagnostics as prepared by the `sampling.oem` helper and post-processing steps.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

`sampling.oem`

Examples

```
## Not run:
res <- cjm.oem(stk, deviances = list(), observations = list(), args = list(dy=2022, frq=1))

## End(Not run)
```

---

cjmage2len *Convert Age Composition to Length Composition for Landings*

Description

This function converts age composition data from landings to length composition using specified biological parameters and selectivity.

Usage

```
cjmage2len(
  landings,
  selex,
  ess = 100,
  L_inf = 80.4,
  k = 0.16,
  L_0 = 18,
  M = 0.33,
  CVlen = 0.09,
  ages = an(dimnames(landings)$age),
  sample_type = "catch"
)
```

Arguments

landings	Matrix of landings at age.
selex	Matrix of selectivity at age.
ess	Effective sample size for each year, default is 100.
L_inf	Asymptotic length, default is 80.4.
k	Growth coefficient, default is 0.16.
L_0	Theoretical length at age zero, default is 18.
M	Natural mortality rate, default is 0.33.
CVlen	Coefficient of variation in length, default is 0.09.
ages	Vector of ages, default is derived from landings.
sample_type	Type of sampling, default is 'catch'.

Value

A matrix representing length composition for each year.

---

cjmfwc *Converts FLQuants with Fleets as Areas into a fwdControl Object*

Description

Converts FLQuants objects, typically returned by a Harvest Control Rule (HCR) module, into a fwdControl object with different behavior based on the number of stocks.

Usage

```
cjmfwc(flqs, quant = "catch", nstocks = 1)
```

Arguments

flqs	An FLQuants object as returned by a HCR module.
quant	The quant to use, defaults to 'catch'.
nstocks	Number of stocks, defaults to 1.

Value

A fwdControl object with the corresponding FCB slot.

exejjms *Execute JJMS Model*

Description

Executes the JJMS model with specified arguments, manages file operations for the model run, and optionally cleans up the output directory.

Usage

```
exejjms(name, path, args = "", verbose = TRUE, clean = TRUE)
```

Arguments

name	The name of the JJMS model.
path	The path to the directory where the JJMS model is located.
args	Additional arguments for the JJMS model execution.
verbose	If TRUE, prints the execution log to the console.
clean	If TRUE, cleans up the output directory after model execution.

Value

Invisible path where the model was executed.

fbar,FLStocks-method *Calculate Average Fishing Mortality across FLStocks*

Description

This method calculates the average fishing mortality (fbar) across multiple FLStock objects contained within an FLStocks object.

Usage

```
## S4 method for signature 'FLStocks'
fbar(object)
```

Arguments

object	An FLStocks object.

Value

A numeric value representing the average fishing mortality.

FLjjm *FLjjm: An R Package for Interacting with JJMS Stock Assessment Model in FLR*

Description

This package provides tools for the Fishery Library (FLR) to interact with the JJMS (Just Another Management Strategy) stock assessment model. The package includes functions for building, reading, and executing JJMS assessments within the FLR framework.

Details

The main functions include:

- `buildFLjjm`: Function to build FLjjm objects.
- `readFLjjm`: Function to read FLjjm data from files.
- `loadJJMS`: Function to load JJMS data.
- `exejjms`: Function to execute the JJMS model.
- `runjjms`: Function to run JJMS simulations.
- `jjms`: General function for JJMS model operations.

The package aims to facilitate the integration of JJMS model outputs with the FLR tools for fisheries science and management strategy evaluation.

Author(s)

Iago Mosqueira (WMR) <iago.mosqueira@wur.nl>

Karolina Molla Gazi (WMR) <karolina.mollagazi@wur.nl>

See Also

`FLjjm` in the FLjjm package for an overview of the package.

fwdmov *Forward Movement Simulation*

Description

This function performs a forward movement simulation on a fishery model object using specified control measures and rates over time.

Usage

```
fwdmov(object, control, rates, time = 0)
```

Arguments

object	The fishery model object to be simulated.
control	A data frame or matrix containing control measures for each year.
rates	A list or vector of rates used in the simulation.
time	An optional time parameter, default is 0.

Value

The updated fishery model object after applying the forward movement simulation.

---

fwdmov.om *Forward Movement Operation Model*

Description

Simulates the movement of fish populations between two areas in a fishery operation model. Adjusts population numbers based on movement rates, fishing mortality, and natural mortality.

Usage

```
fwdmov.om(om, ctrl, FCB = FCB(ctrl), rates, time = 0, ...)
```

Arguments

om	A fishery operation model object.
ctrl	Control measures data frame or matrix.
FCB	Fishing mortality, default is derived from ctrl.
rates	Movement rates matrix.
time	Time step for the simulation, default is 0.
...	Additional arguments.

Value

A list containing the updated fishery operation model object.

jjms *Run JJMS Model for Fisheries Stock Assessment*

Description

Executes the JJMS model for fisheries stock assessment. Handles both single and multiple stocks and can operate in parallel or sequentially. Updates the stock data with the results from the model runs.

Usage

```
jjms(
  stock,
  indices,
  dat = attr(stock, "dat"),
  ctl = attr(stock, "ctl"),
  path = tempfile(),
  mp = FALSE,
  clean = mp
)
```

Arguments

stock	FLStock or list of FLStocks for assessment.
indices	Indices data for the model.
dat	Data for each iteration of the model.
ctl	Control settings for each iteration.
path	Path to save temporary files, defaults to a temporary file path.
mp	If TRUE, runs in parallel; otherwise, runs sequentially.
clean	If TRUE, cleans up temporary files after execution.

Value

The updated FLStock or list of FLStocks after model execution.

jjms.sa *JJMS Stock Assessment*

Description

Performs a JJMS stock assessment, modifying and running the JJMS model with provided arguments.

Usage

```
jjms.sa(stk, idx, args, tracking, ...)
```

Arguments

stk	Stock data.
idx	Index data.
args	Argument list.
tracking	Tracking information.
...	Additional arguments.

Value

A list with the resulting stock data, tracking information, and arguments.

---

loadFLSjjms *Load FLStocks from JJMS Model*

Description

This function loads FLStocks from JJMS model directories located at a specified path. It can handle both single and multiple stock scenarios and offers an option to combine the loaded stocks into a single object.

Usage

```
loadFLSjjms(path, combine = FALSE)
```

Arguments

path	The path to the directory containing the JJMS model data.
combine	Boolean, if TRUE, combine the loaded FLStocks into a single object.

Value

FLStocks object(s) loaded from the specified path.

loadJJMS *Load JJMS Model Components*

Description

Loads a JJMS model from a given path and constructs various components of a fisheries model, including biological data, fisheries data, indices, reference points, and stock data.

Usage

```
loadJJMS(name, path)
```

Arguments

name	The name of the JJMS model.
path	The path to the directory containing the JJMS model data.

Value

A list containing biological data, fisheries data, indices, reference points, stock data, and model data and control settings.

namejjms *Get Name of JJMS Model*

Description

Reads the first configuration file in a JJMS model directory to extract the model's name.

Usage

```
namejjms(path)
```

Arguments

path	The path to the directory containing the JJMS model data.

Value

A string with the name of the JJMS model.

nStocksjjms *Get Number of Stocks in JJMS Model*

Description

Reads a JJMS model control file to determine the number of stocks.

Usage

```
nStocksjjms(name, path)
```

Arguments

name	The name of the JJMS model.
path	The path to the directory containing the JJMS model data.

Value

Numeric value representing the number of stocks in the JJMS model.

packjjmsrun *Clean Up JJMS Model Run Directory*

Description

Removes unnecessary files from a JJMS model run directory, keeping only essential files.

Usage

```
packjjmsrun(path)
```

Arguments

path	The path to the JJMS model run directory.

Value

Invisible TRUE if all files are successfully removed, FALSE otherwise.

`readFLBjjm` *Create an FLBiol from a JJMS model run directory*

Description

An object of class `FLBiol` is created from the information in the various files in a `jjms` folder structure. The function reads the model run (control, input and results folders) and constructs the biological component that can be used within the FLR framework. See `buildFLBjjm` for details about how each slot is populated.

An object of class *FLBiol* is created from the information in the various files in a *jjms* folder structure. See *buildFLBjjm* for how each slot is populated from the inputs and outputs of *jjms*.

Usage

```
readFLBjjm(name, path, stock = 1)
```

Arguments

`name`	Name of the ctl model file, *character*.
`path`	Path ot the model folder structure, *character*.
`stock`	Numeric. Stock index to extract when the run contains multiple stocks. Default is 1.
`stock=1`	Stock to extract, of relevance on 2 stocks model runs, *numeric*.

Details

The function delegates the heavy lifting to `jjmR::readJJM` to import the model run and then to `buildFLBjjm` which maps the jjm inputs/outputs to the corresponding FLBiol slots. Typical jjm outputs used include population numbers by age (N), weight-at-age, natural mortality (M) and maturity. When a multi-stock jjm run is used, specify `stock` to select the correct output component.

Value

An object of class `FLBiol` constructed from the jjm output (with slots such as `n`, `wt`, `m`, `mat` filled according to the model input and output files).

An object of class FLBiol.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

References

EUPL 1.2

See Also

buildFLBjjm, jjmR::readJJM

jjmR::readJJM FLCore::FLBiol buildFLBjjm

Examples

```
## Not run:
biol <- readFLBjjm(name = "h1_1.07",
  path = system.file("ext-data", "single_stock", package = "FLjjm"))
summary(biol)

## End(Not run)
```

---

readFLFsjjm *Create FLFisheries objects from a JJMS model run directory*

Description

Construct the FLFisheries (or individual FLFishery) objects representing fisheries in the jjms output. For each fishery the function creates a FLCatch with slots landings, landings.n, landings.wt, discards and related weights, using the model outputs C_fsh_* and wt_fsh_*.

An object of class *FLFisheries*, a list of *FLFishery* objects, is created from the information in the various files in a *jjms* folder structure. See *buildFLFsjjm* for how each slot is populated from the inputs and outputs of *jjms*.

Usage

```
readFLFsjjm(name, path)
```

Arguments

name	Name of the ctl model file, *character*.
path	Path ot the model folder structure, *character*.
stock	Numeric. Stock index to extract when more than one stock is present. Default is 1.

Details

The function extracts fishery names from out$output[[stock]]$Fshry_names and loops through the fisheries building FLQuant objects for landings numbers (C_fsh_i), weights at age (wt_fsh_i) and other slots.

Value

A list (or an FLFisheries/FLFishery object) with fisheries built for the given stock.

An object of class FLIndices.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

`buildFLFsjjm`

jjmR::readJJM FLFishery::FLFisheries FLFishery::FLFishery buildFLFsjjm

Examples

```
## Not run:
fis <- readFLFsjjm(name = "h1_1.07",
  path = system.file("ext-data", "single_stock", package = "FLjjm"))
str(fis)

## End(Not run)
## Not run:
fisheries <- readFLFsjjm(name="h1_1.07",
  path = system.file("ext-data", "single_stock", package="FLjjm"))
summary(fisheries)

## End(Not run)
```

`readFLIsjjm` *Create an FLIndices object from a JJMS model run directory*

Description

Read jjms output and construct an `FLIndices` object containing the abundance indices used/produced by the model. The function maps index selectivity patterns, catchability (q) and range/start-end fractions to FLIndex objects and combines them into an FLIndices list.

An object of class *FLIndices*, a list of *FLIndex* objects, is created from the information in the various files in a *jjms* folder structure. See *buildFLIsjjm* for how each slot is populated from the inputs and outputs of *jjms*.

Usage

```
readFLIsjjm(name, path)
```

Arguments

`name`	Name of the ctl model file, *character*.
`path`	Path ot the model folder structure, *character*.

Details

The function calls `jjmR::readJJM` to read the run and `buildFLIsjjm` to construct the FLIndices object. The mapping follows the conventions used in the package: selectivity patterns are read from `output@sel_ind_i`, index catchability from `output@q_i` and month ranges are converted to fractions using the `Imonths` input.

Value

An `FLIndices` object with one or more `FLIndex` elements representing survey/observation time series extracted from the model output.

An object of class FLIndices.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

`buildFLIsjjm`, `jjmR::readJJM`

jjmR::readJJM FLCore::FLIndices FLCore::FLIndex buildFLIsjjm

Examples

```
## Not run:
idx <- readFLIsjjm(name = "h1_1.07",
  path = system.file("ext-data", "single_stock", package = "FLjjm"))
plot(idx)

## End(Not run)
## Not run:
indices <- readFLIsjjm(name="h1_1.07",
  path = system.file("ext-data", "single_stock", package="FLjjm"))
summary(indices)

## End(Not run)
```

`readFLoemjjm` *Read FLIndices with OEM iterations and MC outputs*

Description

Read jjm output and construct indices suitable for OEM simulations. When `iter > 1` the function will attempt to read Monte Carlo evaluation output to populate index catchability (q) and other stochastic elements. The function returns indices propagated to the requested number of iterations.

An object of class *FLoem* is created from the information in the various files in a *jjms* folder structure. The *@observations* slot is created containing the past observations of catch and biology (as an *FLStock* or *FLStocks*) and for the indices of abundance (as an *FLIndices*)

Usage

```
readFLoemjjm(name, path, method = cjm.oem, iter = NULL, ...)
```

Arguments

name	Name of the ctl model file, *character*.
path	Path to the model folder structure, *character*.
method	Function. The OEM sampling function to be used; default is cjm.oem.
iter	Integer. Number of iterations to return/propgate. If NULL, defaults to 1.
...	Additional arguments passed to method.

Value

A list of FLIndex objects propagated to iter iterations and augmented with MC-derived catchabilities when available.

An object of class FLoem.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

readFLIsjjm, readMCeval

jjmR::readJJM mse::FLoem buildFLSjjm buildFLIsjjm

Examples

```
## Not run:
idx <- readFLoemjjm("h1_1.07", path = system.file("ext-data","single_stock", package="FLjjm"), iter = 10)

## End(Not run)
## Not run:
# One stock OM
oem <- readFLoemjjm(name="h1_1.07",
  path=system.file("ext-data", "single_stock", package="FLjjm"))
oem
# Two stock OM
oemtwo <- readFLoemjjm(name="h2_1.07",
  path=system.file("ext-data", "two_stock", package="FLjjm"))
oemtwo

## End(Not run)
```

`readFLomjjm` *Create an FLombf object from a JJMS model run directory*

Description

Construct an `FLombf` (operating model for biomass/forces framework) from the outputs of a jjms run. The resulting object contains three main slots populated by the corresponding `buildFL*jjm` helper functions: `biols` (from `buildFLBjjm`), `fisheries` (from `buildFLFsjjm`) and `refpts` (from `buildFLRPsjjm`).

An object of class *FLombf* is created from the information in the various files in a *jjms* folder structure. Three slots are created from the corresponding calls to *buildFLjjm* functions.

- @biols, of class *FLBiols* from a call to *buildFLBsjjm*.
- @fisheries, of class *FLFisheries*, from a call to *buildFLFsjjm*.
- @refpts, of classs *FLPars*, from a call to *buildFLRPsjjm*.

Usage

```
readFLomjjm(name, path, iter = NULL, ...)
```

Arguments

`name`	Name of the ctl model file, *character*.
`path`	Path to the model folder structure, *character*.

Details

The wrapper reads the jjm run with `jjmR::readJJM` and then assembles the subcomponents using the specialized builder functions. The function is useful to convert an existing jjms assessment run into an FLR-compatible operating model for simulation or MSE work.

Value

An `FLombf` object ready to be used as an operating model within the FLR/MSE framework.

An object of class FLBiol.

Author(s)

Iago Mosqueira iago.mosqueira@wur.nl

See Also

`buildFLBjjm`, `buildFLFsjjm`, `buildFLRPsjjm`

jjmR::readJJM FLombf buildFLBjjm buildFLFsjjm

Examples

```
## Not run:
om <- readFLomjjm(name = "h1_1.07",
  path = system.file("ext-data", "single_stock", package = "FLjjm"))
summary(om)

## End(Not run)
# One stock OM
om <- readFLomjjm(name="h1_1.07",
  path=system.file("ext-data", "single_stock", package="FLjjm"))
summary(om)
# Two stock OM
omtwo <- readFLomjjm(name="h2_1.07",
  path=system.file("ext-data", "two_stock", package="FLjjm"))
summary(omtwo)
```

`readFLRPsjjm` *Create an FLPar contaning reference points from a JJMS model run directory*

Description

An object of class *FLPar* is created from the information in the output files in a *jjms* folder structure. See *buildFLRPsjjm* to check what reference points are being extracted from the inputs outputs of *jjms*.

Usage

```
readFLRPsjjm(name, path, stock = 1)
```

Arguments

`name`	Name of the ctl model file, *character*.
`path`	Path ot the model folder structure, *character*.
`stock=1`	Stock to extract, of relevance on 2 stocks model runs, *numeric*.

Value

An object of class FLPar.

See Also

jjmR::readJJM FLCore::FLBiol buildFLBjjm

Examples

```
rps <- readFLRPsjjm(name="h1_1.07",
  path=system.file("ext-data", "single_stock", package="FLjjm"))
summary(rps)
```

readFLSjjm *Create an FLStock from a JJMS model run directory*

Description

An object of class *FLStock* is created from the information in the various files in a *jjms* folder structure. See *buildFLSjjm* for how each slot is populated from the inputs and outputs of *jjms*.

Usage

```
readFLSjjm(name, path, stock = 1, output = TRUE)
```

Arguments

name	Name of the ctl model file, *character*.
path	Path ot the model folder structure, *character*.
stock=1	Stock to extract, of relevance on 2 stocks model runs, *numeric*.

Value

An object of class FLStock.

See Also

jjmR::readJJM FLCore::FLStock buildFLSjjm

Examples

```
bio <- readFLSjjm(name="h1_1.07",
  path = system.file("ext-data", "single_stock", package="FLjjm"))
summary(bio)
```

readFLSsjjm *Create an FLStocks from a two-stock JJMS model run directory*

Description

An object of class *FLStocks* is created from the information in the various files in a *jjms* folder structure. See *buildFLSjjm* for how each slot is populated from the inputs and outputs of *jjms*.

Usage

```
readFLSsjjm(name, path, output = FALSE)
```

Arguments

name	Name of the ctl model file, *character*.
path	Path ot the model folder structure, *character*.

Value

An object of class FLStock.

See Also

jjmR::readJJM FLCore::FLStock buildFLSjjm

Examples

```
bio <- readFLSsjjm(name="h2_1.07",
  path=system.file("ext-data", "two_stock", package="FLjjm"), output=TRUE)
summary(bio)
```

---

runjjms *Run JJMS Model*

Description

Executes the JJMS model with the given model data, saving the model files to a specified path. The function creates necessary directories, writes model files, and calls the JJMS model.

Usage

```
runjjms(mod, path = tempfile(), args = "", verbose = TRUE, clean = TRUE)
```

Arguments

mod	The JJMS model data to be executed.
path	The path to save model files and run the JJMS model, defaults to a temporary file path.
args	Additional arguments for the JJMS model execution.
verbose	if TRUE, execution details are printed; otherwise, they are suppressed.

Value

The path where the JJMS model was executed.

slick_performance *Compute performance metrics for MPs and OM in slick format*

Description

This function computes performance metrics for MPs over future years and for OM over past years.

Usage

```
slick_performance(
  x,
  om,
  statistics,
  metrics = list(C = catch, SB = ssb, F = fbar)
)
```

Arguments

x	A list of MP or OM projections runs, of class 'list' or 'FLmse'.
om	An OM, of class 'FLom' or 'FLombf'.
statistics	A list of performance statistics to compute, 'list'.
metrics	Metrics to compute, 'list'.

Value

A combined data.table with performance metrics

Examples

```
# slick_performance(x = my_mps_data, om = my_om_data, statistics = my_statistics)
```

ssb,FLStocks-method *Calculate Total Spawning Stock Biomass across FLStocks*

Description

This method calculates the total spawning stock biomass (ssb) by summing the ssb of each FLStock object within an FLStocks object.

Usage

```
## S4 method for signature 'FLStocks'
ssb(object)
```

Arguments

`object`	An FLStocks object.

Value

A numeric value representing the total spawning stock biomass.

`yourFunctionName` *Construct Components of Fisheries Model*

Description

This function constructs various components of a fisheries model from a given model object. It builds biological data, fisheries data, indices, reference points, and stock data.

Usage

```
yourFunctionName(mod)
```

Arguments

`mod`	The model object from which components are to be constructed.

Value

A list containing biological data, fisheries data, indices, reference points, stock data, and model data and control settings.

Index