Package: FLjjm (via r-universe)

August 30, 2024

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

Version 0.1.9.9011

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, 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 7.3.1

NeedsCompilation no

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

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

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

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

RemoteRef HEAD

RemoteSha da26edc7dd86099b09a3bf953f01d7bd4287c3c0

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 *Build an FLIndices object from a jjm.output list*

Description

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=1	Stock for which indices are to be loaded.

Value

An FLIndices object.

Examples

```
idx <- readFLIsjjm("h1_1.07",
  path = system.file("ext-data", "single_stock", package="FLjjm"))
summary(idx)
plot(idx)
```

buildFLRPsjjm *Construct an FLPar contaning reference points from a jjm.output object*

Description

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

`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)
```

`buildFLSsjjm` *Build Multiple FLStock Objects from a jjm.output Object*

Description

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

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

Usage

```
buildFLSsjjm(out)
```

```
buildFLSsjjm(out)
```

Arguments

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

Value

An FLStocks object containing multiple FLStock objects.

An FLStocks object containing multiple FLStock objects.

See Also

`FLStock`, `FLStocks`

`FLStock`, `FLStocks`

`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 *Update Observations in CJM Fishery Operation Model*

Description

Updates stock and index observations in a CJM fishery operation model based on provided arguments, tracking information, and F3 selectivity.

Usage

```
cjm.oem(stk, deviances, observations, args, tracking, F3sel)
```

Arguments

stk	Stock data.
deviances	Deviances for the model.
observations	Observations data.
args	Argument list containing model parameters.
tracking	Tracking information.
F3sel	F3 selectivity data.

Value

A list with updated stock data, indices, observations, and tracking information.

cjm2.oem *Update Observations in Fishery Operation Model*

Description

This function updates stock observations and indices in a fishery operation model (OM) based on provided arguments and tracking information.

Usage

```
cjm2.oem(om, deviances, observations, args, tracking)
```

Arguments

om	Fishery operation model object.
deviances	Deviances for the model.
observations	Observations data.
args	Argument list containing model parameters.
tracking	Tracking information.

Value

A list with updated stock data, indices, observations, and tracking information.

`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.

cperfect.oem *Create a Perfect Observation Error Model*

Description

This function creates a perfect observation error model for a stock assessment, adjusting the stock data to match observations.

Usage

```
cperfect.oem(
  stk,
  deviances,
  observations,
  args,
  tracking,
  biomass = FALSE,
  ...
)
```

Arguments

`stk`	Stock data.
`deviances`	Deviances for the model.
`observations`	Observations data.
`args`	Argument list.
`tracking`	Tracking information.
`biomass`	Boolean, if TRUE, use biomass instead of numbers.
`...`	Additional arguments.

Value

A list with updated stock data, index, observations, and tracking.

`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,
  ctl,
  path = tempfile(),
  mp = FALSE,
  clean = mp,
  lengthcomp_F3 = NULL
)
```

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.
`lengthcomp_F3`	Optional length composition data for F3.

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.

perfcjm.sa *Perfect CJM Stock Assessment*

Description

A placeholder function for perfect CJM stock assessment, potentially for testing or method development purposes.

Usage

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

Arguments

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

Value

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

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. 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=1`	Stock to extract, of relevance on 2 stocks model runs, *numeric*.

Value

An object of class FLBiol.

See Also

jjmR::readJJM FLCore::FLBiol buildFLBjjm

Examples

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

---

readFLFsjjm — *Create an FLFisheries from a JJMS model run directory*

Description

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*.

Value

An object of class FLIndices.

See Also

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

Examples

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

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

Description

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*.

Value

An object of class FLIndices.

See Also

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

Examples

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

readFLoemjjm *Create an FLoem from a JJMS model run directory*

Description

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 = 1, ...)
```

Arguments

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

Value

An object of class FLoem.

See Also

jjmR::readJJM mse::FLoem buildFLSjjm buildFLIsjjm

Examples

```
# 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
```

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

Description

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*.

Value

An object of class FLBiol.

See Also

jjmR::readJJM FLombf buildFLBjjm buildFLFsjjm

Examples

```
# 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)
```

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.

`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