Package: jjmR (via r-universe)

June 6, 2026

Type Package

Title Graphics and diagnostics tools for SPRFMO's Joint Jack Mackerel model

Version 1.2020.1

Date 2020-10-13

Description Graphics and diagnostics tools for SPRFMO's Joint Jack Mackerel model.

Author Ricardo Oliveros-Ramos [cre, aut], Wencheng Lau-Medrano [aut], Giancarlo Moron [aut], Josymar Torrejon-Magallanes [aut], Niels Hintzen [aut], Enrique Ramos-Vasquez [ctb], Jim Ianelli [aut]

Maintainer Ricardo Oliveros-Ramos <roliveros@imarpe.gob.pe>

License GPL-2

Depends R (>= 4.1.0)

Imports colorspace, doBy, dplyr, foreach, forcats, ggplot2, ggridges, graphics, grDevices, grid, gridExtra, icesAdvice, knitr, lattice, latticeExtra, magrittr, patchwork, PBSmodelling, purrr, rmarkdown, scales, methods, stats, tibble, tidyr, utils

Encoding UTF-8

RoxygenNote 7.3.2

Config/pak/sysreqs bwidget cmake make libicu-dev libjpeg-dev libpng-dev libuv1-dev libxml2-dev

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

Date/Publication 2026-05-22 10:22:01 UTC

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

RemoteRef HEAD

RemoteSha 80ca41cd8891e1221dba5c22e034036bde7a6c4f

Contents

`.getJjmConfig` *Create a jjm.config object*

Description

Construct a `jjm.config` object from data and control lists.

Usage

```
.getJjmConfig(data, control, ...)
```

Arguments

`data`	List of raw data elements for the JJM model.
`control`	List of control parameters, including `modelName`.
`...`	Unused.

Value

An object of class `jjm.config`.

`annex_diagnostics` *Annex diagnostic plotting helpers*

Description

Helpers for building benchmark annex diagnostic figures from JJM model outputs, including input metadata, composition fits, index diagnostics, retrospective plots, selectivity, stock status, stock-recruitment, Kobe plots, and historical retrospectives.

Usage

```
jjm_label(nm, stock_name = NULL)
jjm_pfa_palette(type = c("pfa", "fleet", "survey", "peel"))
extr_catch(mod, stock = 1)
extr_wtat_fsh(mod, stock = 1)
extr_wtat_srv(mod, stock = 1)
extr_agefits_fsh(mod, stock = 1)
extr_lenfits_fsh(mod, stock = 1)
extr_agefits_srv(mod, stock = 1)
extr_indices(mod, stock = 1)
extr_meanage_fsh(mod, stock = 1)
extr_meanage_srv(mod, stock = 1)
extr_meanlen_fsh(mod, stock = 1)
```

```
extr_effn(mod, stock = 1)
extr_catch_cv(mod, stock = 1)
extr_fished_unfished(mod, stock = 1)
extr_retro(retro, var = "SSB", stock = 1)
extr_q_breaks(mod)
extr_cv(mod, stock = 1)
extr_sel_fsh(mod, stock = 1)
extr_sel_srv(mod, stock = 1)
extr_sel_breaks(mod, type = c("fsh", "srv"))
extr_sr(mod, stock = 1)
extr_kobe(mod, stock = 1)
extr_francis(agefits, effn_df, grp_var = "fleet")
extr_q_trajectory(mod, stock = 1)
extr_survey_skill(mod, stock = 1)
has_comp(df, grp_var = NULL, grp_filter = NULL)
comp_fig_height(df, grp_var = NULL, grp_filter = NULL, ncol = 5,
  panel_h = 2.2, row2_h = 2.5, row3_h = 2.2)
facet_fig_height(n_panels, ncol = 2, panel_h = 3.5, min_h = 4)
plot_catch_fleet(mod, stock = 1, title = NULL)
calc_mohn_rho(df, n_peel = 5)
plot_retro(retro, var = "SSB", stock = 1, title = NULL)
plot_wtat(df, facet_var, title = NULL, years = NULL)
plot_wtat_cohort(df, facet_var, title = NULL, years = NULL)
plot_comp_fits(df, x_var, grp_var, grp_filter = NULL, x_lab = "Age",
  title = NULL)
plot_indices_v2(mod, stock = 1, title = NULL)
plot_comp_ridge(df, x_var = "age", facet_var = "fleet", x_lab = "Age",
  title = NULL)
plot_sel_ridge(df, facet_var, title = NULL, breaks_df = NULL)
plot_fit_diagnostics(df, title = NULL)
plot_effn(mod, stock = 1, title = NULL)
plot_meanstat(df, facet_var, y_lab = "Mean age (years)", title = NULL)
plot_fished_unfished(mod, stock = 1, title = NULL)
plot_summary_sheet(mod, stock = 1, title = NULL)
plot_sr(mod, stock = 1, title = NULL)
plot_kobe(mod, stock = 1, title = NULL)
strong_cohorts(mod, stock = 1, n = 3)
annotate_cohorts(years, colour = PFA_GREEN, label_size = 3)
plot_francis(mod, stock = 1, title = NULL)
plot_q_trajectory(mod, stock = 1, title = NULL)
plot_survey_skill(mod, stock = 1, title = NULL)
plot_metadata_overview(mod, title = NULL)
read_historic_retro(file,
  exclude_assessment_types = c("benchmark", "mod1.4"))
plot_historic_retro(data, quantity = c("stock", "reference"),
  title = NULL)
```

Arguments

mod	A jjm.output object returned by readJJM.
stock	Stock index.
retro	A retrospective object loaded from a JJM retrospective RData file.
df, agefits, effn_df, data	Data frames or tibbles returned by the corresponding extractor functions.
title	Optional plot title.
var	Retrospective variable, such as "SSB", "R", or "F".
type, quantity	Character option selecting a palette, selectivity type, or historical retrospective plot group.
grp_var	Grouping variable name.
grp_filter	Optional group value to filter.
ncol	Number of panels per row.
panel_h	Height per panel row.
row2_h	Height for the bubble residual row.
row3_h	Height for the marginal residual row.
n_panels	Number of panels.
min_h	Minimum figure height.
n_peel	Number of retrospective peels to include.
facet_var	Facet variable name.
x_var	X variable name.
x_lab	X-axis label.
breaks_df	Optional selectivity break-year data.
y_lab	Y-axis label.
n	Number of cohort years to return.
colour	Annotation colour.
label_size	Annotation label size.
exclude_assessment_types	Assessment types to remove before plotting historical retrospectives.
nm, stock_name	Model name and optional stock label used to construct a display label.
file	Path to a historical retrospective CSV file.
years	Years to annotate or plot.

Value

Extractor functions return tibbles or lists. Plot functions return **ggplot2** or **patchwork** objects.

Examples

```
## Not run:
mod <- readJJM("h1_1.14", path = "assessment/config",
  input = "assessment/input", output = "assessment/results")

plot_metadata_overview(mod)
plot_catch_fleet(mod, title = jjm_label("h1_1.14"))
plot_indices_v2(mod)

## End(Not run)
```

`changeNameModel` *Change the internal name of a model*

Description

This function internally replaces the name of a JJM output object with a user-specified string. Mostly useful for plots.

Usage

```
changeNameModel(modList, nameVector)
```

Arguments

`modList`	A list of JJM model output objects.
`nameVector`	Character vector of replacement model names.

Examples

```
## Not run:
recmods <- compareModels(c("mod1.00.hl","mod1.00.ll","mod1.00.hs","mod1.00.ls"))

changeNameModel(recmods,c( "h=0.8, full series","h=0.8, short series","h=0.65, full series","h=0.65, short series

## End(Not run)
```

coerce *Coerce a vector to character or numeric*

Description

Coerce a vector to character or numeric

Usage

```
coerce(x, to = c("chr", "num", "num_chr"))
```

Arguments

x	Vector to coerce
to	One of "chr", "num", "num_chr"

combineModels *Combine outputs*

Description

This function takes model objects (class `outputs`) of JJM and generate an object with combined models.

Usage

```
combineModels(...)
```

Arguments

...	One or more output objects, to be combined to list of models.

Examples

```
## Not run:
mod1 <- runJJM(modelName = "mod2.1")
mod2 <- runJJM(modelName = "mod2.2")
mod3 <- runJJM(modelName = "mod2.3")

mod_123 = combineModels(mod1, mod2, mod3)

## End(Not run)
```

compareModels *Compare combined JJM outputs*

Description

This function takes a vector of model names, reads in the JJM runs, and combines them. Basically a wrapper function for `combineModels`. Assumes model runs are in the same folder.

Usage

```
compareModels(mods)
```

Arguments

`mods`	Character vector of model names to compare.

Examples

```
## Not run:

mod_123 = compareModels(c("h1_0.00", "h1_0.01", "h1_0.02")

## End(Not run)
```

diagnostics *Generate Assessment plots from single model*

Description

Function to generate plots from results of readJJM function

Usage

```
diagnostics(object, ...)
```

Arguments

`object`	Object ob class outputs.
`...`	Extra arguments

Examples

```
## Not run:
model = readJJM(modelName = "mod2.4")
diagnostics(object = model)

## End(Not run)
```

`fixed_bmsy` *Calculate or input a fixed Bmsy value for the jjm model Updates the msy_mt table in the jjm output with new B/Bmsy ratios.*

Description

Calculate or input a fixed Bmsy value for the jjm model Updates the msy_mt table in the jjm output with new B/Bmsy ratios.

Usage

```
fixed_bmsy(mod, refpt = NULL)
```

Arguments

`mod`	jjm object that is a list of lists
`refpt`	A number (or vector of numbers if multiple stocks) to input as Bmsy. If not filled, calculated as the average of the Bmsy estimated for the last ten years (as determined in SCW14 benchmark 2022)

Value

A model

Examples

```
# fixed_bmsy(mod_h1_1.00, refpt=5500) # To input a fixed Bmsy
# fixed_bmsy(mod_h1_1.00) # To calculate the Bmsy
```

`get_age_fits` *Get fits to indices*

Description

Get fits to indices

Usage

```
get_age_fits(models)
```

Arguments

`models`	an object of class jjm.output

Value

a tidy dataframe of age fits

get_catchabilities *Get estimated catchability coefficients*

Description

Get estimated catchability coefficients

Usage

```
get_catchabilities(models)
```

Arguments

models	an object of class jjm.output

Value

a data frame of estimated catchabilities

get_fishing_mortality *Get fishing mortality at age*

Description

Get fishing mortality at age

Usage

```
get_fishing_mortality(models)
```

Arguments

models	an object of class jjm.output

Value

a tidy dataframe of fishing mortality at age

get_index_fits *Get fits to indices*

Description

Get fits to indices

Usage

```
get_index_fits(models)
```

Arguments

models	an object of class jjm.output

Value

a tidy dataframe of index fits

get_len_fits *Get fits to length compositions*

Description

Get fits to length compositions

Usage

```
get_len_fits(models)
```

Arguments

models	an object of class jjm.output

Value

a tidy dataframe of length composition fits

get_msy_mt *Get and tidy msy_my table*

Description

Get and tidy msy_my table

Usage

```
get_msy_mt(models)
```

Arguments

models	series of class jjm.output

Value

a tidy msy_mt

Examples

```
## Not run:

mod0.00 <- readJJM("h2_0.00", path = "config", input = "input")
get_msy_mt(mod0.00)

## End(Not run)
```

get_recruits *Get estimated recruits*

Description

Get estimated recruits

Usage

```
get_recruits(models)
```

Arguments

models	an object of class jjm.output

Value

a tidy dataframe of recruits

get_selectivities *Get and tidy selectivity-at-age ogives over time by model and fleet*

Description

Get and tidy selectivity-at-age ogives over time by model and fleet

Usage

```
get_selectivities(models)
```

Arguments

models	an object of class jjm.output

Value

a tidy data frame of selectivity estimates

Examples

```
## Not run:
h1.mod <- jjmR::readJJM("h2_0.02", path = "config", input = "input")
selectivities <- get_selectivities(h1.mod)

## End(Not run)
```

get_totals *Get total metrics (biomass, spawning biomass, and recruitment)*

Description

Get total metrics (biomass, spawning biomass, and recruitment)

Usage

```
get_totals(models)
```

Arguments

models	an object of class jjm.output

Value

a dataframe of total values

geth *Add hypothesis number to a model name*

Description

Add hypothesis number to a model name

Usage

```
geth(mod, h = hyp)
```

Arguments

mod	A character string of a model name.
h	A character string containing the hypothesis to use.

Value

A character string containing the hypothesis name and the model name.

Examples

```
geth("1.00", "h1")
```

getter *Get elements of a list by string matching on name*

Description

Get elements of a list by string matching on name

Usage

```
getter(x, pattern = "^sel_", things = NA)
```

Arguments

x	the object
pattern	the string pattern (regex supported) to search for
things	Optional preselected object list; defaults to NA.

Value

an object subset to matches in names with strings

gray_ramp *Compute a gray color ramp*

Description

Compute a gray color ramp

Usage

```
gray_ramp(n)
```

Arguments

n	Number of shades

Value

Character vector of hex grays

kobe *Kobe plot*

Description

This function create a kobe plot from JJM model outputs

Usage

```
kobe(
  obj,
  add = FALSE,
  col = "black",
  stock = 1,
  Bref = 1,
  Fref = 1,
  Blim = Bref,
  Flim = Fref,
  xlim = NULL,
  ylim = NULL,
  engine = "ggplot",
  ...
)
```

Arguments

obj	a jjm model outputs object.
add	boolean, add to an existing kobe plot?
col	color for the lines and points.
stock	Number of the stock chosen for the kobe plot.
Bref	Reference point for B/B_MSY, default=1.
Fref	Reference point for F/F_MSY, default=1.
Blim	Limit reference point for B/B_MSY, default=0.5.
Flim	Limit reference point for F/F_MSY, default=1.5.
xlim	'x' axis limits.
ylim	'y' axis limits.
engine	Plotting engine, either "ggplot" or "lattice".
...	Additional parameters passed to plot.

Examples

```
## Not run:
kobe(model)

## End(Not run)
```

---

melt_matrix *Melt an age-or-length matrix into a long tibble*

Description

Melt an age-or-length matrix into a long tibble

Usage

```
melt_matrix(mat, years = NULL, ages = NULL, class = "age")
```

Arguments

mat	A matrix with years in column 1 and data in subsequent columns
years	Vector of years (if not the first column)
ages	Ages or lengths
class	Name of the grouping variable (e.g. "age" or "length")

Value

A tibble with columns 'year', 'value', and '<class>'

`plot.jjm.diag` *Plot method for jjm.diag objects*

Description

Creates diagnostic plots from the JJM model diagnostics based on specified parameters. Can generate plots for model inputs (data) or outputs across different models and stocks.

Usage

```
## S3 method for class 'jjm.diag'
plot(
  x,
  what = c("data", "output"),
  model = NULL,
  stock = NULL,
  var = NULL,
  fleet = NULL,
  plot = TRUE,
  ...
)
```

Arguments

`x`	An object of class 'jjm.diag'
`what`	Character vector specifying which type of diagnostics to plot. Options are "data" (or deprecated "input") and "output". Default is c("data", "output").
`model`	Character vector of model names to include. If NULL (default), all models are used.
`stock`	Character vector of stock names to include. If NULL (default), all stocks are used.
`var`	Character vector of variable names to include. If NULL (default), all variables are used.
`fleet`	Character vector or numeric vector of fleet indices to include. If NULL (default), all fleets are used.
`plot`	Logical. If TRUE (default), plots are displayed. If FALSE, plot objects are returned.
`...`	Additional arguments passed to internal plotting functions

Value

If plot=TRUE, returns invisibly. If plot=FALSE, returns a list of plot objects.

Examples

```
## Not run:
diagnostics <- diag.jjm("path/to/model")
# Plot all diagnostics
plot(diagnostics)

# Plot only data diagnostics for a specific model and stock
plot(diagnostics, what = "data", model = "Model1", stock = "Stock1")

# Return plot objects without displaying
plot_objects <- plot(diagnostics, plot = FALSE)

## End(Not run)
```

---

plot.jjm.diagnostics *Plot diagnostics*

Description

Plot diagnostics

Usage

```
## S3 method for class 'jjm.diagnostics'
plot(x, which = names(x), ...)
```

Arguments

x	A summary.jjm.diagnostics
which	Vector of named diagnostics to render
...	Additional arguments. Currently ignored.

---

plot.jjm.output *Plot method for jjm.output objects*

Description

Creates various plots from the JJM model output based on the specified type.

Usage

```
## S3 method for class 'jjm.output'
plot(
  x,
  what = "biomass",
  stack = TRUE,
  endvalue = FALSE,
  total = FALSE,
  combine = FALSE,
  cols = NULL,
  poslegend = "right",
  scen = 1,
  ...
)
```

Arguments

x	An object of class 'jjm.output'
what	Character string specifying the type of plot. Options include: • "biomass" - Plot biomass over time • "recruitment" - Plot recruitment over time • "ssb" - Plot spawning stock biomass over time • "noFishTB" - Plot unfished total biomass • "ftot" - Plot total fishing mortality • "kobe" - Create a Kobe plot • "catchProj" - Plot catch projections • "ssbProj" - Plot SSB projections • "totalProj" - Plot total projections • "catchProjScen" - Plot catch projection scenarios • "ssbProjScen" - Plot SSB projection scenarios • "ratioSSB_F" - Plot ratio of SSB to F • "ratioSSB" - Plot ratio of SSB • "selectivity" - Plot selectivity curves
stack	Logical. If TRUE, stack multiple series in one plot. Default is TRUE.
endvalue	Logical. If TRUE, display end values on plot. Default is FALSE.
total	Logical. If TRUE, include total values. Default is FALSE.
combine	Logical. If TRUE, combine multiple plots. Default is FALSE.
cols	Optional vector of colors for plotting
poslegend	Position of the legend. Default is "right".
scen	Integer. Scenario number for projection plots. Default is 1.
...	Additional arguments passed to plotting functions

Value

A plot object (typically a grid or ggplot object)

Examples

```
## Not run:
model <- read.jjm("path/to/model")
plot(model, what = "biomass")
plot(model, what = "ssb", stack = FALSE)
plot(model, what = "kobe")

## End(Not run)
```

---

plot_fishery_weight *Plot weight at age in the fishery*

Description

Plot weight at age in the fishery

Usage

```
plot_fishery_weight(out, ages)
```

Arguments

out	A jjm output list
ages	Vector of ages

---

plot_selectivities *Plot selectivities by age, year, fleet, and model*

Description

Plot selectivities by age, year, fleet, and model

Usage

```
plot_selectivities(
  sels,
  fleet = "fsh",
  alpha = 0.4,
  scale = 4,
  size = 0.5,
  years = "all"
)
```

Arguments

sels	selectivity data frame generated by get_selectivities
fleet	fleets to plot: "fsh" (fishery), "ind" (survey), or "all" (both)
alpha	Line opacity.
scale	Scaling factor for the panel strip color gradient.
size	Line width.
years	Years to plot, or "all" for every year.

Value

a ggplot2 plot object

Examples

```
## Not run:

oldnewMods <- combineModels(mod0.00,mod_prev)
selectivities <- get_selectivities(oldnewMods)
plot_selectivities(selectivities)


## End(Not run)
```

---

print.jjm.config *Print a jjm.config object*

Description

Print a jjm.config object

Usage

```
## S3 method for class 'jjm.config'
print(x, ...)
```

Arguments

x	A jjm.config object.
...	Additional arguments (ignored).

Value

Invisibly returns x.

print.jjm.diag *Print method for jjm.diag objects*

Description

Prints a summary of the JJM diagnostic object, displaying information about available input and output plots for each model and stock.

Usage

```
## S3 method for class 'jjm.diag'
print(x, ...)
```

Arguments

x	An object of class 'jjm.diag'
...	Additional arguments passed to print methods

Value

Invisibly returns the input object (for use in pipelines)

Examples

```
## Not run:
diagnostics <- diag.jjm("path/to/model")
print(diagnostics)

## End(Not run)
```

print.jjm.output *Print method for jjm.output objects*

Description

Prints a summary of the JJM model output object, displaying basic information about the model, stocks, and data sources.

Usage

```
## S3 method for class 'jjm.output'
print(x, ...)
```

Arguments

x	An object of class 'jjm.output'
...	Additional arguments passed to print methods

Value

Invisibly returns the input object (for use in pipelines)

Examples

```
## Not run:
model <- read.jjm("path/to/model")
print(model)

## End(Not run)
```

---

print.summary.jjm.config

Print a summary of jjm.config

Description

Print a summary of jjm.config

Usage

```
## S3 method for class 'summary.jjm.config'
print(x, ...)
```

Arguments

x	A summary.jjm.config object.
...	Additional arguments (ignored).

Value

Invisibly returns x.

---

print.summary.jjm.diag

Print method for summary.jjm.diag objects

Description

Prints the detailed diagnostics information from a summary.jjm.diag object, showing both input and output plot details.

Usage

```
## S3 method for class 'summary.jjm.diag'
print(x, ...)
```

Arguments

x	An object of class 'summary.jjm.diag'
...	Additional arguments passed to print methods

Value

Invisibly returns the input object (for use in pipelines)

---

print.summary.jjm.output *Print method for summary.jjm.output objects*

Description

Prints the tables and plots generated by the summary method for jjm.output objects.

Usage

```
## S3 method for class 'summary.jjm.output'
print(x, ...)
```

Arguments

x	An object of class 'summary.jjm.output'
...	Additional arguments passed to print methods

Value

Invisibly returns the input object (for use in pipelines)

---

readExFiles *Read external files*

Description

Read external files

Usage

```
readExFiles(
  fileName,
  type,
  path = NULL,
  version = "2015MS",
  parameters = FALSE,
  parData,
  nameFishery,
  nameIndex,
  nAges,
  nStock = NULL
)
```

Arguments

fileName	filename
type	type
path	path
version	version of JJM, default to "2015MS" (2015 SC multi-stock).
parameters	parameters
parData	parData
nameFishery	nameFishery
nameIndex	nameIndex
nAges	nAges
nStock	nStock

---

readJJM *Read a model or list of models*

Description

Function to read models and list if models and generate results

Usage

```
readJJM(
  model,
  path = NULL,
  output = "results",
  input = NULL,
  version = "2015MS",
  ...
)
```

Arguments

model	String with the name of model that will be readed or run.
path	Directory where the 'admb' folder is located.
output	Path to the model outputs directory.
input	Path to model inputs directory.
version	version of JJM, default to "2015MS" (2015 SC multi-stock).
...	Extra arguments

Examples

```
## Not run:
readJJM(model = "mod2.4")

## End(Not run)
```

---

readJJMConfig *Read dat and ctl files from disk to create a jjm.config object.*

Description

Store in an R object (of class jjm.config) the dat and ctl files needed to run a model.

Usage

```
readJJMConfig(model, path, input = NULL, ...)
```

Arguments

model	Model object or outputs
path	Path to the ctl file
input	Path to the input files
...	Additional arguments passed to other functions.

Examples

```
## Not run:
readJJMConfig(mod1)

## End(Not run)
```

report *Create a report from JJM outputs*

Description

Function to create and save reports in PDF and MS Word formats.

Usage

```
report(object, format, output, tangle = FALSE, tidy = TRUE, ...)
```

Arguments

object	The object to create the report with, can be of classes 'jjm.output' or 'jjm.diag' as created with readJJM or diagnostics.
format	Format for the report: either "pdf", "html" or "word".
output	Path to save the report, by default the working directory.
tangle	Boolean, if TRUE the R script to create the report is produced.
tidy	Boolean, if TRUE the intermediate files (Rmd, tex) are deleted.
...	Extra arguments

Examples

```
## Not run:
report(mod0.0)

## End(Not run)
```

retro *Run a retrospective analysis diagnostic for a JJM model*

Description

Run a retrospective analysis for a model

Usage

```
retro(
  model,
  n = 5,
  output = "results",
  exec = NULL,
  parallel = FALSE,
  temp = NULL,
```

```
  wait = TRUE,
  iprint = 100,
  ...
)
```

Arguments

model	An object of class jjm.output
n	Number of years to run a retrospective analysis.
output	Path to save results.
exec	Path to JJM executable file.
parallel	Boolean flag to run models in parallel.
temp	Folder to run retrospective analysis. If NULL, a temporal folder is used.
wait	Boolean, passed to runJJM, should we wait for the parameter estimation?
iprint	Command line argument passed to jjm.
...	Additional arguments passed to other functions.

Examples

```
## Not run:
retro(mod1)

## End(Not run)
```

---

runit *Fit, run, read and plot a JJM model*

Description

Shortcut to fit, run, read and plot a JJM model

Usage

```
runit(
  mod,
  est = FALSE,
  exec = NULL,
  path = "config",
  input = "input",
  output = "results",
  version = "2015MS",
  pdf = FALSE,
  portrait = TRUE,
  ...
)
```

Arguments

mod	A character specifying the name of a model (by it's ctl filename).
est	Boolean, should we run the parameter estimation for a model?
exec	Path to the JJM executable file. By default, 'jjms' will be used.
path	Directory where the configuration files will be written.
input	Input
output	Folder to save the outputs, 'arc' by default.
version	version of JJM, default to "2015MS" (2015 SC multi-stock).
pdf	Produce outputs in a pdf file?
portrait	Orientation of the pdf output, default TRUE.
...	Additional arguments passed to runJJM.

Examples

```
## Not run:
writeJJM(mod1)

## End(Not run)
```

---

runJJM *Run a JJM model*

Description

Function to run one or several JJM models

Usage

```
runJJM(
  models,
  path = NULL,
  output = "results",
  input = NULL,
  exec = NULL,
  version = NULL,
  useGuess = FALSE,
  guess = NULL,
  piner = NULL,
  iprint = 100,
  wait = TRUE,
  parallel = FALSE,
  temp = NULL,
  ...
)
```

Arguments

models	String with the name of the models to be run.
path	Directory where the 'admb' folder is located.
output	Folder to save the outputs, 'arc' by default.
input	Input
exec	Path to the jjm executable
version	version of JJM, default to "2015MS" (2015 SC multi-stock).
useGuess	boolean, to use an initial guess for the parameters?
guess	File with the initial guess for the parameters. If NULL, will use model.par in the output folder.
piner	A number to start the profiling on the meanlogrec
iprint	iprint parameter for the JJM model, 100 by default.
wait	boolean, wait for the model to finish? Forced to be TRUE.
parallel	Should model run in parallel? A cluster need to be setup to be used with foreach.
temp	character, path for a temporal directory to run models, if NULL a temporal folder is automaticaly created.
...	Arguments passed from system function.

Examples

```
## Not run:
model = runJJM(models = "mod2.4")

## End(Not run)
```

---

runJJM.default *Run JJM models (default method)*

Description

Executes one or more JJM model runs, optionally in parallel.

Usage

```
## Default S3 method:
runJJM(
  models,
  path = NULL,
  output = "results",
  input = NULL,
  exec = NULL,
  version = NULL,
  useGuess = FALSE,
```

```
  guess = NULL,
  piner = NULL,
  iprint = 100,
  wait = TRUE,
  parallel = FALSE,
  temp = NULL,
  ...
)
```

Arguments

models	Character vector of model filenames.
path	Optional directory containing model files.
output	Directory for model results (created if missing).
input	Directory for input files (default = model directory).
exec	Path to JJM executable.
version	Executable version (ignored if exec provided).
useGuess	Logical; whether to use starting parameter guesses.
guess	Vector of guess filenames or logical.
piner	Additional runner parameters.
iprint	Printing control for JJM output.
wait	Logical; whether to wait for completion before returning.
parallel	Logical; whether to run models in parallel.
temp	Directory for temporary files (default = tempdir()).
...	Additional arguments passed to internal runner.

Value

Invisibly returns the path to temp used for input files.

---

runJJM.jjm.config *Run JJM models for jjm.config*

Description

Run JJM models for jjm.config

Usage

```
## S3 method for class 'jjm.config'
runJJM(
  models,
  path = NULL,
  output = "results",
  input = NULL,
  exec = NULL,
  version = NULL,
  useGuess = FALSE,
  guess = NULL,
  piner = NULL,
  iprint = 100,
  wait = TRUE,
  parallel = FALSE,
  temp = NULL,
  ...
)
```

Arguments

models	Character vector of model filenames.
path	Optional directory containing model files.
output	Directory for model results (created if missing).
input	Directory for input files (default = model directory).
exec	Path to JJM executable.
version	Executable version (ignored if exec provided).
useGuess	Logical; whether to use starting parameter guesses.
guess	Vector of guess filenames or logical.
piner	Additional runner parameters.
iprint	Printing control for JJM output.
wait	Logical; whether to wait for completion before returning.
parallel	Logical; whether to run models in parallel.
temp	Directory for temporary files (default = tempdir()).
...	Additional arguments passed to internal runner.

---

runJJM.jjm.output *Run JJM models for jjm.output*

Description

Run JJM models for jjm.output

Usage

```
## S3 method for class 'jjm.output'
runJJM(
  models,
  path = NULL,
  output = "results",
  input = NULL,
  exec = NULL,
  version = NULL,
  useGuess = FALSE,
  guess = NULL,
  piner = NULL,
  iprint = 100,
  wait = TRUE,
  parallel = FALSE,
  temp = NULL,
  ...
)
```

Arguments

models	Character vector of model filenames.
path	Optional directory containing model files.
output	Directory for model results (created if missing).
input	Directory for input files (default = model directory).
exec	Path to JJM executable.
version	Executable version (ignored if exec provided).
useGuess	Logical; whether to use starting parameter guesses.
guess	Vector of guess filenames or logical.
piner	Additional runner parameters.
iprint	Printing control for JJM output.
wait	Logical; whether to wait for completion before returning.
parallel	Logical; whether to run models in parallel.
temp	Directory for temporary files (default = tempdir()).
...	Additional arguments passed to internal runner.

---

summary.jjm.config *Summarize a jjm.config object*

Description

Summarize a jjm.config object

Usage

```
## S3 method for class 'jjm.config'
summary(object, ...)
```

Arguments

object	A jjm.config object.
...	Additional arguments (ignored).

Value

An object of class summary.jjm.config containing model names.

---

summary.jjm.diag *Summary method for jjm.diag objects*

Description

Creates a detailed summary of JJM diagnostics, including information about input and output plots for model evaluation.

Usage

```
## S3 method for class 'jjm.diag'
summary(object, ...)
```

Arguments

object	An object of class 'jjm.diag'
...	Additional arguments passed to internal functions

Value

An object of class 'summary.jjm.diag' containing detailed diagnostics

Examples

```
## Not run:
diagnostics <- diag.jjm("path/to/model")
diag_summary <- summary(diagnostics)
print(diag_summary)

## End(Not run)
```

summary.jjm.diagnostics *Summarize all diagnostics from a jjm run*

Description

Summarize all diagnostics from a jjm run

Usage

```
## S3 method for class 'jjm.diagnostics'
summary(object, ...)
```

Arguments

object	A list returned by runJJM
...	Additional arguments. Currently ignored.

summary.jjm.output *Summary method for jjm.output objects*

Description

Generates a detailed summary of the JJM model output, including parameter tables, likelihood tables, projection tables (if requested), and summary plots.

Usage

```
## S3 method for class 'jjm.output'
summary(
  object,
  Projections = FALSE,
  Fmult = NULL,
  BiomProj = NULL,
  CapProj = NULL,
  MRS = NULL,
  ...
)
```

Arguments

object	An object of class 'jjm.output'
Projections	Logical. Whether to include projection tables. Default is FALSE.
Fmult	Optional vector of F multipliers for projections
BiomProj	Optional biomass projection parameters
CapProj	Optional capacity projection parameters
MRS	Optional Maximum Relative Selectivity parameters
...	Additional arguments passed to internal plotting functions

Value

An object of class 'summary.jjm.output' containing tables and plots

Examples

```
## Not run:
model <- read.jjm("path/to/model")
model_summary <- summary(model, Projections = TRUE)
print(model_summary)

## End(Not run)
```

---

theme_jjm *ggplot2 theme for jjmR*

Description

ggplot2 theme for jjmR

Usage

```
theme_jjm(base_size = 12, ...)
```

Arguments

base_size	Base font size.
...	Additional arguments passed to ggplot2::theme_bw().

Value

a ggplot2 theme object

Examples

```
library(ggplot2)
ggplot(mtcars, aes(mpg)) + geom_histogram() + theme_jjm()
```

tidy_JJM *Tidy results of JJM model*

Description

Tidy results of JJM model

Usage

```
tidy_JJM(models)
```

Arguments

models	an object of class jjm.output

Value

a list of tidy dataframes

Examples

```
## Not run:
mod0.00 <- readJJM("h2_0.00", path = "config", input = "input")
tidy_jjm_results <- tidy_JJM(mod0.00)


## End(Not run)
```

writeJJM *Write dat and ctl files from a JJM model stored in R*

Description

Function write to the disk dat and ctl files

Usage

```
writeJJM(object, path, ...)
```

Arguments

object	An object of class jjm.config or jjm.output.
path	Directory where the configuration files will be written.
...	Additional arguments

Examples

```
## Not run:
writeJJM(mod1)

## End(Not run)
```

Index