Package ‘salmonMSE’

April 16, 2026

Title Management Strategy Evaluation for Salmon Species

Version 2.0.0

Date 2026-04-13

Maintainer Quang Huynh <`quang@bluematterscience.com`>

Description Simulation tools to evaluate the long-term effects of salmon management strategies, including a combination of habitat, harvest, and habitat actions. The stochastic age-structured operating model accommodates complex life histories, including freshwater survival across early life stages, juvenile survival and fishery exploitation in the marine life stage, partial maturity by age class, and fitness impacts of hatchery programs on natural spawning populations. 'salmonMSE' also provides an age-structured conditioning model to develop operating models fitted to data.

License GPL (>= 3)

Depends R (>= 3.5.0)

Imports abind, MSEtool (>= 3.7.2), RTMB (>= 1.9), dplyr, ggplot2, grDevices, gsl, methods, reshape2, rlang, rmarkdown, stats, utils

Suggests ggrepel, rstan, scales, testthat (>= 3.0.0), tmbstan

Encoding UTF-8

LazyData yes

LazyLoad yes

RoxygenNote 7.3.3

URL `https://docs.salmonmse.com/`, `https://github.com/Blue-Matter/salmonMSE`

BugReports `https://github.com/Blue-Matter/salmonMSE/issues`

NeedsCompilation no

Author Quang Huynh [aut, cre] (ORCID: <`https://orcid.org/0000-0001-7835-4376`>)

Repository CRAN

Date/Publication 2026-04-16 07:00:02 UTC

Contents

AHA *All-H Analyzer*

Description

Wrapper function for an implementation of All-H Analyzer (AHA) in R. Can be used to compare outputs between AHA and salmonMSE.

Usage

```
AHA(SOM, ngen = 100, silent = FALSE)
```

Arguments

`SOM`	An object of class SOM
`ngen`	Integer, the number of generations for which to run the simulation
`silent`	Logical, indicates whether to silence messages to the R console

Value

A named list containing vectors of state variables (by simulation, population, and generation). See SMSE object description.

References

Hatchery Scientific Review Group. 2020. All-H Analyzer Tool Guide and Documentation. May 2020.

`Bio-class` *Class* `"Bio"`

Description

The component of the operating model that controls biological dynamics, i.e., natural production.

Details

Various parameters can be stochastic (length `nsim`) or input as a single numeric (value identical across all simulations).

Slots

`Name` Character. Identifying name

`maxage` Integer. The maximum age of the population age structure.

`n_g` Integer. Number of life history groups within a cohort. Life history groups (LHGs) are subunits of a cohort that have different marine survival. Default is 1.

`p_LHG` The proportion of the total egg production assigned to each life history group within a cohort. For example, if `Bio@n_g <- 2`, then `Bio@p_LHG <- c(0.9, 0.1)`, then 90 percent of the egg production in the first population is assigned to the first life history group and ten percent to the second LHG. Default is `rep(1/Bio@n_g, Bio@n_g)`

`p_mature` Either vector by age (length `maxage`) or an array with dimension `[nsim, maxage, proyears]`. The proportion mature by age.

`SRrel` Character, stock-recruit relationship for density-dependent smolt production from fry. Either "BH" (Beverton-Holt) or "Ricker". Not used if habitat component is used. See `Habitat` object.

`capacity` Vector length `nsim`. Only used if `SRrel = "BH"`. The asymptote, i.e., maximum juvenile production, of the Beverton-Holt stock-recruit function for density-dependent natural smolt production from egg production. **Units of smolts.** Not used if habitat component is used.

`kappa` Vector length `nsim`. The adult productivity ratio for the stock-recruit function. **Units of recruits per spawner.** Natural per-capita production of recruits as the population approaches zero (density-independent component). Not used if habitat component is used.

`Smax` Vector length `nsim`. Only used if `SRrel = "Ricker"`. The spawner abundance that maximizes smolt production in the Ricker stock-recruit function. **Units of spawners.** Not used if habitat component is used.

`phi` Optional, vector length `nsim`. Egg production per smolt at unfished replacement. **Units of egg per smolt**. Converts productivity from units of recruits/spawner (`kappa`) to smolts/egg (`alpha`), where `alpha = kappa/phi`. In simple models, `phi` is the product of marine survival, fecundity, and proportion female. If not provided, `phi` will be calculated from `Mjuv_NOS`, `p_mature`, `s_enroute`, `p_female`, `fec`, and `p_LHG` corresponding to the first year and weighted by life history groups. Not used if habitat component is used.

`tau` Optional, vector length `nsim`. Spawner per smolt at unfished replacement, only used if `SRrel = "Ricker"`. **Units of spawner per smolt**. Used to convert `Smax` to `Emax`, the corresponding egg production that maximizes smolt production, where `Emax = Smax * phi/tau` and the Ricker parameter `beta = 1/Emax`. In simple models, `tau` is the product of marine survival and proportion female. If not provided, `tau` will be calculated from `Mjuv_NOS`, `p_mature`, `s_enroute`, `p_female`, and `p_LHG` corresponding to the first year and weighted by life history groups. Not used if habitat component is used.

`Mjuv_NOS` Either vector by age (length `maxage-1`) or an array with dimension `[nsim, maxage-1, proyears, n_g]`. Natural mortality of immature natural origin fish, the value for the first age represents natural mortality from age 1 to 2, second age is mortality from age 2 to 3, and so on. To replicate the SAR parameter of a stage-specific model, set `Mjuv_NOS[a] = -log(SAR)` where `a` is the age class prior to maturation (and zero for all other ages).

`fec` Vector by age (length `maxage`) or an array with dimension `[nsim, maxage, proyears]`. Female fecundity of natural origin spawners.

`p_female` Numeric. The proportion of females in the spawning population. Default is 0.5. Can also be a vector `[maxage]` (for situations where older spawners are predominantly female)

`s_enroute` Numeric. Survival of escapement to the spawning grounds (for spawning and for broodtake). Default is 1.

Creating Object

Objects can be created by calls of the form `new("Bio")`

Examples

```
showClass("Bio")
```

`calc_ref` *Reference points*

Description

Calculate MSY and Sgen reference points for the operating model. Uses the biological parameters (maturity, natural mortality) in the last year of the projection.

- `calc_MSY()` calculates the MSY reference points from a set of biological and fishery parameters
- `calc_Sgen()` calculates the Sgen, the spawner abundance that would reach the spawner abundance at MSY after one generation without fishing
- `calc_ref()` is a wrapper function that calculates MSY and Sgen for an operating model

Usage

```
calc_ref(SOM, rel_F, check = TRUE, maximize = c("MSY", "MER"))

calc_MSY(
  Mjuv,
  fec,
  p_female,
  rel_F,
  vulPT,
  vulT,
  p_mature,
  s_enroute,
  n_g = 1,
  p_LHG = 1,
  SRRpars,
  maximize = c("MSY", "MER"),
  F_search = c(1e-08, 5)
)

calc_Sgen(
  Mjuv,
  fec,
  p_female,
  rel_F,
  vulPT,
  vulT,
  p_mature,
  s_enroute,
  n_g = 1,
  p_LHG = 1,
  SRRpars,
```

```
  SMSY,
  F_search = c(1e-08, 100),
  nyears
)
```

Arguments

`SOM`	An object of class SOM
`rel_F`	Numeric length 2, indicates the relative effort in the preterminal and terminal fisheries, with a maximum value of 1. The default is `c(0, 1)` which indicates a yield calculation with only the terminal fishery.
`check`	Logical, whether to check the SOM object using `check_SOM()`
`maximize`	Character, whether the MSY calculation is the optimum that maximizes catch (`"MSY"`) or excess recruitment (`"MER"`). The two methods should be equivalent when `rel_F = c(0, 1)`.
`Mjuv`	Numeric `maxage` for juvenile natural mortality. Can be a matrix `[maxage, n_g]`.
`fec`	Numeric `maxage` for fecundity. Can be a matrix `[maxage, n_g]`.
`p_female`	Numeric for proportion female spawners
`vulPT`	Numeric `maxage` for preterminal vulnerability at age
`vulT`	Numeric `maxage` for terminal vulnerability at age
`p_mature`	Numeric `maxage` for maturity proportions at age. Can be a matrix `[maxage, n_g]`.
`s_enroute`	Numeric for en-route survival of escapement to spawning grounds
`n_g`	Integer, number of life history groups within a cohort
`p_LHG`	Numeric `n_g` for proportion of the total egg production assigned to each life history group within a cohort
`SRRpars`	Data frame, one row, that contains the stock recruit parameters that predicts density-dependent survival at the egg-smolt life stage
`F_search`	Numeric, length 2 for the range of F values to search for the instantaneous fishing mortality that produces MSY
`SMSY`	Numeric, spawning abundance at MSY
`nyears`	Integer, number of years to project the population with no fishing to reach `SMSY`. Default is the minimum age of maturity.

Value

- `calc_MSY` returns a vector of various state variables (catch, exploitation rate, egg production, spawners) at MSY
- `calc_Sgen` returns a numeric
- `calc_ref` returns a list by stock, each containing a matrix of MSY state variables and Sgen by simulation

See Also

`calc_Smsy_Ricker()`

Examples

```
ref_simple <- calc_ref(simple_SOM)
ref_multi <- calc_ref(multi_SOM)
```

`calc_smolt` *Smolt production*

Description

Calculate smolt production from base stock-recruit parameters and fitness loss

Usage

```
calc_smolt(
  N1,
  N2 = N1,
  kappa,
  capacity,
  Smax,
  phi = 1,
  tau = 1,
  fitness_loss = 1,
  SRrel = c("BH", "Ricker"),
  per_recruit = FALSE
)
```

Arguments

`N1`	Egg production for the density-independent component of the stock-recruit relationship. Can be the number of spawners if `phi = 1` and `Smax` is in units of spawners.
`N2`	Egg production for the density-dependent component of the stock-recruit relationship (only used if `per_recruit = FALSE`)
`kappa`	Base productivity parameter
`capacity`	Base capacity parameter if `SRrel = "BH"`
`Smax`	Base Smax parameter if `SRrel = "Ricker"`
`phi`	Unfished egg per smolt (`1/phi` corresponds to the one-to-one adult/spawner replacement line)
`tau`	Unfished spawner per smolt
`fitness_loss`	Survival term to reduce smolt production due to fitness, between 0-1
`SRrel`	Character for the stock-recruit function
`per_recruit`	Logical, whether N1 is a per recruit quantity (TRUE) or in absolute numbers (FALSE)

Value

Numeric

`calc_Smsy_Ricker` *Ricker reference points*

Description

Compute reference points (Umsy, Smsy, and Sgen) from Ricker stock-recruit function based on Scheuerell (2016).

Usage

```
calc_Smsy_Ricker(loga, b)

calc_Umsy_Ricker(loga)

calc_Sgen_Ricker(loga, b)
```

Arguments

`loga`	Numeric, alpha parameter (returns per spawner) in the Ricker function: $R = S\exp(\log(a) - bS)$ where `S` is the number of spawners and `R` is the return
`b`	Numeric, beta parameter

Value

All three functions return a numeric

References

Scheuerell, M.D. 2016. An explicit solution for calculating optimum spawning stock size from Ricker's stock recruitment model. PeerJ 4:e1623. doi:10.7717/peerj.1623

See Also

`calc_ref()`

Examples

```
a <- 3
Smax <- 100
b <- 1/Smax

calc_Smsy_Ricker(log(a), b)
calc_Umsy_Ricker(log(a))
calc_Sgen_Ricker(log(a), b)
```

---

calc_SRR *Calculate abundance from density-dependent mortality*

Description

Calculates the abundance of survivors after applying either a Beverton-Holt or Ricker stock-recruit relationship.

Usage

```
calc_SRR(N1, N2 = N1, p, capacity, type = c("BH", "Ricker", "HS"))
```

Arguments

`N1`	Numeric, the initial abundance that scales the density-independent survival term
`N2`	Numeric, the initial abundance that scales the density-dependent survival term
`p`	Numeric, the productivity parameter that sets the maximum survival as the initial abundance approaches zero
`capacity`	Numeric, the capacity parameter that set the maximum survivors
`type`	Character, the functional form of the stock-recruit relationship

Details

The Beverton-Holt stock recruit relationship is of the following form:

$$\text{Smolt} = \frac{\alpha N_1}{1 + \beta N_2}$$

where $\alpha = P$, $\beta = P/C$.

The Ricker stock recruit relationship is of the following form:

$$\text{Smolt} = \alpha N_1 \exp(-\beta N_2)$$

where $\alpha = P$, $\beta = P/(Ce)$, e is Euler's number.

Productivity P is in terms of abundance per unit of N_1 and N_2.

The hockey stick is of the following form:

$$\text{Smolt} = \begin{cases} pN_1 & , N_1 \leq \frac{N_1}{N_2} \times C \\ \frac{N_1}{N_2} \times C & , \text{otherwise} \end{cases}$$

Value

Numeric, the abundance of survivors

See Also

`calc_SRRpars()`

Examples

```
N1 <- 100
N2 <- 200
p <- 10
capacity <- 250

calc_SRR(N1, N2, p, capacity, type = "BH")

# Validation
SRRpars <- calc_SRRpars(p, capacity, type = "BH")
a <- SRRpars[1]
b <- SRRpars[2]
a * N1/(1 + b * N2)
```

`calc_SRRpars` *Convert density-dependent survival parameters*

Description

Converts from capacity/productivity parameters to alpha/beta stock-recruit parameters where productivity is in terms of smolts per spawner and alpha is terms of smolts per egg.

Usage

```
calc_SRRpars(p, capacity, f = 1, p_female = 1, type = c("BH", "Ricker", "HS"))
```

Arguments

`p`	Numeric, the productivity parameter that sets the maximum survival as the initial abundance approaches zero
`capacity`	Numeric, the capacity parameter that set the maximum survivors
`f`	Fecundity, the spawning output per mature female
`p_female`	The proportion of females per spawner
`type`	Character, the functional form of the stock-recruit relationship

Details

$$\alpha = \frac{P}{f \times p_{female}}$$

For the Beverton-Holt stock recruit relationship:

$$\beta = \frac{\alpha}{C}$$

For the Ricker stock recruit relationship:

$$\beta = \frac{\alpha}{Ce}$$

, e is Euler's number.

Value

Numeric vector length 2 for alpha and beta value, respectively

See Also

`calc_SRR()`

Examples

```
N1 <- 100
N2 <- 200
p <- 10
capacity <- 250

calc_SRRpars(p, capacity, type = "BH")
```

`check_SOM` *Check inputs to SOM object*

Description

Ensures that the slots in the SOM object have the correct dimensions. Function will update some slots to their full dimensions.

Usage

```
check_SOM(SOM, silent = FALSE)
```

Arguments

`SOM`	SOM object
`silent`	Logical, whether to report progress in console

Value

Updated SOM object with full dimensions in various slots

Examples

```
SOM_checked <- check_SOM(simple_SOM, silent = TRUE)
```

`CM2SOM` *Convert conditioning model to operating model*

Description

Creates an operating model from MCMC samples and data inputs of the conditioning model. Management actions for habitat, hatchery production, and harvest still need to be specified in the operating model.

Note: the function assumes the maturity values in the last conditioning year for the projection, which are likely not well informed by CWT data. Consider updating the maturity using some historical average (e.g., across most recent completed brood years).

Usage

```
CM2SOM(stanfit, sims, nsim = 2, seed = 1, proyears = 40)
```

Arguments

`stanfit`	Output from `sample_CM()`
`sims`	Optional, a vector of integers indicating the MCMC iterations to convert to operating model simulations. Otherwise, use argument `nsim` in order to sample a subset of the MCMC.
`nsim`	Integer, total number of simulations in the operating model. Only used if `sims` is missing.
`seed`	Integer, seed for sampling the MCMC output. Only used if `sims` is missing.
`proyears`	Integer, the number of projection years in the operating model

Value

SOM object.

`compare` *Compare scenarios in markdown*

Description

Generate a markdown report for multiple model runs to compare scenarios. 3-5 is likely the ideal number of scenarios for comparison.

Usage

```
compare(
  SMSE_list,
  names,
  col_vec,
  filename = "SMSEcompare",
  dir = tempdir(),
  open_file = TRUE,
  render_args = list(),
  ...
)
```

Arguments

`SMSE_list`	List of SMSE objects
`names`	Character vector `length(SMSE_list)` to label individual model runs
`col_vec`	Character vector `length(SMSE_list)` for custom colour schemes for comparing across model scenarios in figures
`filename`	Character string for the name of the markdown and HTML files.
`dir`	The directory in which the markdown and HTML files will be saved.
`open_file`	Logical, whether the HTML document is opened after it is rendered.
`render_args`	List of arguments to pass to `rmarkdown::render()`.
`...`	Additional arguments (not used)

Value

Returns invisibly the output of `rmarkdown::render()`, typically the path of the output file

See Also

`report()`

`compare_spawners` *Compare simulation runs*

Description

Create figures that compare results across two dimensions

Usage

```
compare_spawners(SMSE_list, Design, prop = FALSE, FUN = median)

compare_fitness(SMSE_list, Design, FUN = median)

compare_escapement(SMSE_list, Design, FUN = median)
```

Arguments

`SMSE_list`	A list of SMSE objects returned by `salmonMSE()`
`Design`	A data frame with two columns that describes the factorial design of the simulations. Used to label the figure. Rows correspond to each object in `SMSE_list`. There two columns are variables against which to plot the result. See example in `https://docs.salmonmse.com/articles/decision-table.html`.
`prop`	Logical, whether to plot absolute numbers over proportions
`FUN`	Summarizing function across simulations, typically `stats::median()` or `base::mean()`

Details

- `compare_spawners()` generates a time series of the composition of spawners
- `compare_fitness()` generates a time series of metrics (fitness, PNI, pHOS, and pWILD) related to hatchery production
- `compare_escapement()` generates a time series of the proportion of spawners and broodtake to escapement

Value

A ggplot object

`compare_statevar_ts` *Compare state variables from simulation runs*

Description

Compare outputs from multiple simulations to evaluate performance across states of nature and/or management levers (identified by colour):

- `compare_statevar_ts()` produces a time series for all simulations, or with medians and 95th percentile intervals
- `compare_statevar_hist()` produces a histogram or density plot across all simulations for a particular year

Usage

```
compare_statevar_ts(
  SMSE_list,
  var = "PNI",
  s = 1,
  figure = TRUE,
  xlab = "Projection Year",
  quant = FALSE,
  ylab = var,
  ylim,
```

```
  agg.fun = sum,
  names,
  col_vec,
  ...
)

compare_statevar_hist(
  SMSE_list,
  var = "PNI",
  s = 1,
  y,
  figure = TRUE,
  xlab = var,
  names,
  col_vec,
  type = c("density", "hist"),
  ...
)
```

Arguments

`SMSE_list`	List of SMSE objects for multiple model runs returned by `salmonMSE()`
`var`	Character. Slot for the state variable in `SMSE` object. See `slotNames(SMSE)` for options. Additional supported options are: `"ESS"` (egg-smolt survival), `"pbrood"` (broodtake to escapement ratio), `"pNOSesc"` (NOS/natural escapement), `"pHOSesc"` (HOS/hatchery escapement), `Total Spawners` (NOS + HOS), `NOS/SMSY`, `S/SMSY`, and `NOS/Sgen`.
`s`	Integer. Population index for multi-population model (e.g., `s = 1` is the first population in the model)
`figure`	Logical, whether to generate a figure (set to FALSE if only using the function to return the data matrix)
`xlab`	Character. Name of time variable for the figure
`quant`	Logical, whether to plot individual simulations (FALSE) or the median with 95 percent confidence intervals (TRUE)
`ylab`	Character. Name of the state variable for the figure
`ylim`	Vector. Y-axis limits
`agg.fun`	Function. Defines how to aggregate state variables that are reported by age. Typically, `sum` is used but `max` is also possible for reporting apical exploitation rates.
`names`	Character vector `length(SMSE_list)` to label individual model runs
`col_vec`	Character vector `length(SMSE_list)` for custom colour schemes for comparing across model scenarios in figures
`...`	Additional arguments to base plot function
`y`	Integer. Projection year for the state variable to plot the histogram. If missing, the last projection year is used.
`type`	Character, whether to generate a density figure or histogram

Value

An array invisibly. Also generates base graphics if `figure = TRUE`

See Also

`plot_statevar_ts()` `compare()`

`fit_CM` *Fit conditioning model to historical data*

Description

Bayesian stock reconstruction model of natural and hatchery origin fish population. Maturity and age-1 natural mortality are estimated from coded wire tag catch and escapement at age. A separate series of observed escapement, and hatchery releases reconstructs the population of interest, informed by natural mortality and maturity from CWT (Korman and Walters 2024). The model estimates time-varying maturity rate as well as time-varying ocean survival as a linear model of covariates (separate covariates for age 1 vs. ages 2+). The model can include either a preterminal juvenile fishery, terminal return fishery, or both (see Data and start sections of the documentation).

`fit_CM()` generates the RTMB model from data which can then be passed to `sample_CM()` to run the MCMC in Stan. Generate a markdown report with `report_CM()`.

More information is available on the salmonMSE website

Usage

```
fit_CM(
  data,
  start = list(),
  map = list(),
  lower = list(),
  upper = list(),
  do_fit = TRUE,
  silent = TRUE,
  control = list(eval.max = 1e+05, iter.max = 1e+05),
  ...
)

sample_CM(fit, ...)
```

Arguments

`data`	A list containing data inputs. See details.
`start`	An optional list containing parameter starting values. See details.
`map`	An optional list that describes how parameters are fixed in the model. See `TMB::MakeADFun()`.

`lower`	Named list containing lower bounds for parameters. See details.
`upper`	Named list containing upper bounds for parameters. See details.
`do_fit`	Logical, whether to do the fit and estimate the Hessian.
`silent`	Logical, whether to silence output from RTMB to the console.
`control`	List, `control` argument to pass to `stats::nlminb()`.
`...`	For `fit_CM`, arguments to `RTMB::MakeADFun()`. For `sample_CM`, arguments to `rstan::sampling()`
`fit`	List of output from `fit_CM()`

Value

- `fit_CM()` returns a named list containing the RTMB model (`obj`), nlminb output (`opt`), standard errors (`SD`), and parameter bounds (`lower` and `upper`)
- `sample_CM()` returns a `stanfit` object containing the MCMC chains

Data

Data should passed through a named list with the following entries.

- `Nages` Integer, number of age classes in the model
- `Ldyr` Integer, number of years in the model
- `lht` Integer, life history type. Should be 1 for now
- `n_r` Integer, number of release strategies for CWT, subset of a hatchery-origin brood year that differ in maturity rate. Default is 1.
- `cwtrelease` Matrix `[Ldyr, n_r]`, coded wire tag (CWT) releases by **release year and release strategy**
- `cwtesc` Array `[Ldyr, Nages, n_r]`. CWT escapement **by release year, age, and release strategy** (assumes broodyear is release year minus 1). Poisson likelihood.
- `cwtcatPT` Array `[Ldyr, Nages, n_r]`. CWT preterminal catch (juvenile fish), **by release year, age, and release strategy**. Poisson likelihood. Set all values to zero to turn off parameters related to the preterminal fishery.
- `cwtcatT` Array `[Ldyr, Nages, n_r]`. CWT terminal catch (returning, mature fish), **by release year, age, and release strategy**. Poisson likelihood. Set all values to zero to turn off parameters related to the terminal fishery.
- `bvulPT` Vector length `Nages`. Prior mean for the vulnerability at age to the preterminal fishery.
- `bvulT` Vector length `Nages`. Prior mean for the vulnerability at age to the terminal fishery.
- `RelRegFPT` Vector `Ldyr`. Trend in relative regional preterminal fishing mortality. Fishing mortality is estimated by estimating a scaling coefficient and annual deviations from this vector. Default is `rep(1, d$Ldyr)` (no prior trend) if `cwtcatPT` is provided, otherwise zero.
- `RelRegFT` Vector `Ldyr`. Trend in relative regional terminal fishing mortality. Default is `rep(1, d$Ldyr)` (no prior trend) if `cwtcatT` is provided, otherwise zero.
- `bmatt` Vector length `Nages`. Proportion maturity at age, base values for calculating the unfished replacement line. Also the prior means if year-specific maturity rates are estimated.

- `mobase`. Vector length `Nages`. Natural mortality at age, base values for calculating the unfished replacement line and the the equilibrium spawners at age.
- `covariate1` *Optional.* Matrix `Ldyr, ncov1` of linear covariates that predict natural mortality for age 1.
- `covariate` *Optional.* Matrix `Ldyr, ncov` of linear covariates that predict natural mortality for ages 2+.
- `hatchsurv` Numeric, survival of hatchery releases into the smolt life stage. Density-independent. Default is 1. If less than 1, then hatchery origin fish have lower survival to age 2 (after first year of marine life stage) compared to natural origin fish.
- `gamma` *Optional.* Numeric, the relative spawning success of hatchery origin spawners. Default is 1.
- `ssum` Numeric, proportion of spawners that is female. Can also be a vector `Nages`
- `fec` Vector length `Nages`. Fecundity, egg production at age
- `r_matt` Integer, the release strategy for which to use maturity parameter for the natural system. Default is 1.
- `obsescape` Vector length `Ldyr` by **return year**, total observed escapement from fisheries, i.e., return to river (all ages and both hatchery/natural fish). Lognormal likelihood.
- `propwildspawn` Vector length `Ldyr` by **return year**, proportion of the escapement that spawn (accounts for en-route mortality and broodtake)
- `hatchrelease` Vector length `Ldyr+1` by **release year**, number of hatchery releases for the population. Default is zero.
- `obs_pHOS` *Optional.* Vector length `Ldyr` by **return year**, observations of proportion of hatchery origin spawners (census) (between 0-1) *by brood year.* Fitted to model with logistic-normal likelihood.
- `pHOS_sd` Numeric, logistic-normal standard deviation of pHOS observations. Default is 1.
- `pHOS_init` Numeric, initial pHOS for equilibrium abundance in the first year of the model. Default is 0.
- `s_enroute` Numeric, survival of escapement to spawning grounds. Default is 1.
- `so_mu` Numeric, the prior mean for spawners at unfished replacement in logspace. Default is `log(3 * max(data$obsescape))`.
- `so_sd` Numeric, the prior standard deviation for spawners at unfished replacement in logspace. Default is 0.5.
- `finitPT` Numeric, initial preterminal fishing mortality for calculating the equilibrium juvenile proportions at age in the first year of the model. Default is 0. Set to `"estimate"` to allow the model to estimate the equilibrium condition.
- `finitT` Numeric, initial terminal fishing mortality for calculating the equilibrium juvenile proportions at age in the first year of the model. Default is 0. Set to `"estimate"` to allow the model to estimate the equilibrium condition.
- `spawn_init` Numeric, initial spawners to calculate equilibrium abundance in the first year of the model. Default is `obsescape[1]`.

- `cwtExp` Numeric, the CWT expansion factor, typically the reciprocal of the catch sampling rate (higher factors for lower sampling rate). The model scales down the CWT predictions to match the observations. In other words, the model assumes that the CWT catch and escapement are not expanded. For example, `cwtExp = 10` divides the CWT predictions by 10 for the likelihood. Default is 1. The Poisson distribution is used for the likelihood of the CWT observations, and the expansion parameter can be used to downweight the CWT likelihood relative to the escapement time series. However it requires adjustments of the CWT catches prior to fitting to ensure the proper population scale. If the expanded catch is 100, then the input CWT catch should be 10 and 50 with `cwtExp` of 10 and 2, respectively, to maintain the same population scale. The Poisson variance scales with the mean and is higher with `cwtExp = 2`.
- `fitness` Logical, whether to calculate fitness effects on survival. Default is `FALSE`.
- `theta` Vector length 2, the optimum phenotype value for the natural and hatchery environments. Default is 100 and 80, respectively. See online article for more information.
- `rel_loss` Vector length 3, the loss in fitness apportioned between the egg, fry (both prior to density-dependence), and smolt (after density-dependence) life stage. The three values should sum to 1.
- `zbar_start` Vector length 2, the mean phenotype of the spawners and broodtake in the natural and hatchery environment, respectively, at the start of the model. Default values of 100 and 100, implying maximum fitness at for the natural environment at the start of the model.
- `fitness_variance` Numeric. The variance (omega-squared) of the fitness function. Assumed identical between the natural and hatchery environments. Default is 100.
- `phenotype_variance` Numeric. The variance (sigma-squared) of the phenotypic trait (with optimum theta). Assumed identical between the natural and hatchery environments. Default is 10.
- `heritability` Numeric. The heritability (h-squared) of the phenotypic trait. Between 0-1. Default is 0.5.
- `fitness_floor` Numeric. The minimum fitness value in the natural and hatchery environments. Default is 0.5.

start

Starting values for parameters can be provided through a named list:

- `log_cr` Numeric, log of the compensation ratio (productivity). Default is 3.
- `log_so` Numeric, unfished spawners in logspace. Default is `log(3 * max(data$obsescape))`.
- `moadd` Numeric, additive term to base natural mortality rate for age 1 juveniles. Default is zero.
- `wt` Vector `Ldyr`. Annual deviates in natural mortality during the freshwater life stage (affects egg to smolt survival). Estimated with normal prior with mean zero and standard deviation `p$wt_sd`. Default is zero.
- `wto` Vector `Ldyr`. Annual deviates in natural mortality for age 1 juveniles (marine life stage). Estimated with normal prior with mean zero and standard deviation `p$wto_sd`. Default is zero.
- `log_FbasePT` Numeric, scaling coefficient to estimate preterminal fishing mortality from `data$RelRegFPT`. Default is `log(0.1)`.

- `log_FbaseT` Numeric, scaling coefficient to estimate preterminal fishing mortality from `data$RelRegFT`. Default is `log(0.1)`.
- `log_fanomalyPT` Vector `Ldyr`. Annual lognormal deviates from `exp(log_FbasePT) * data$RelRegFPT` to estimate preterminal fishing mortality. Estimated with normal prior with mean zero and standard deviation `p$fanomaly_sd`. Default is zero.
- `log_fanomalyT` Vector `Ldyr`. Annual lognormal deviates from `exp(log_FbaseT) * data$RelRegFT` to estimate terminal fishing mortality. Estimated with normal prior with mean zero and standard deviation `p$fanomalyPT_sd`. Default is zero.
- `lnE_sd` Numeric, lognormal standard deviation of the observed escapement. Estimated with hierarchical `gamma(2, 5)` prior. Default is 0.1.
- `wt_sd` Numeric, lognormal standard deviation of the egg to smolt (freshwater) natural mortality deviates. Estimated with hierarchical `gamma(2, 5)` prior. Default is 1.
- `wto_sd` Numeric, lognormal standard deviation of the age 1 (marine) natural mortality deviates. Estimated with hierarchical `gamma(2, 5)` prior. Default is 1.
- `fanomalyPT_sd` Numeric, lognormal standard deviation of `fanomalyPT`. Estimated with hierarchical `gamma(2, 5)` prior. Default is 1.
- `fanomalyT_sd` Numeric, lognormal standard deviation of `fanomalyT`. Estimated with hierarchical `gamma(2, 5)` prior. Default is 1.
- `logit_vulPT` Vector `Nages-2` of preterminal vulnerability at age in logit space. Fixed to zero and one at age 1 and the maximum age, respectively. Default is `qlogis(data$bvul_PT[-c(1, data$Nages)])`.
- `logit_vulT` Vector `Nages-2` of terminal vulnerability at age in logit space. Fixed to zero and one at age 1 and the maximum age, respectively. Default is `qlogis(data$bvul_T[-c(1, data$Nages)])`.
- `logit_matt` Matrix `Ldyr, Nages-2` maturity by year and age in logit space. Maturity is fixed to zero and one at age 1 and the maximum age, respectively. Default is `matrix(qlogis(data$bmatt[-c(1, data$Nages)]), data$Ldyr, data$Nages-2, byrow = TRUE)`.
- `sd_matt` Vector `Nages-2`. Logit standard deviation of maturity (`logit_matt`) by age class. Default is 0.5.
- `b1` Vector `ncov1` of coefficients for linear covariates that predict natural mortality for age 1. Default is zero.
- `b` Vector `ncov` of coefficients for linear covariates that predict natural mortality for ages 2+. Default is zero.

Bounds

By default, the standard deviation parameters and parameters in normal space (e.g., `FbasePT`, `Fbase_T`) have a lower bound of zero. `moadd` has a lower bound of zero by default, but it is feasible that this parameter can be negative as well. Deviation parameters centred around zero are bounded between -3 to 3. The `log_cr` parameter has a lower bound of zero.

All other parameters are unbounded.

Covariates on natural mortality

Natural mortality is modeled as the sum of a base value M^{base}, additional scaling factor for age 1 M^{add}, a linear system of covariates X and coefficients b:

$$M_{y,a} = \begin{cases} M_a^{\text{base}} + M^{\text{add}} + \sum_j b_j^1 X_{y,j}^1 & a = 1 \\ M_a^{\text{base}} + \sum_j b_j X_{y,j} & a = 2, \ldots, A \end{cases}$$

Author(s)

Q. Huynh from Stan code provided by J. Korman and C. Walters

References

Korman, J. and Walters, C. 2024. A life cycle model for Chinook salmon population dynamics. Canadian Contractor Report of Hydrography and Ocean Sciences 62: vi + 60 p.

See Also

`report_CM()`

`CM2SOM()`

`get_report` *Figures for conditioning model results*

Description

Functions used by the markdown report to generate summary figures from the age-structured conditoning model

Usage

```
get_report(stanfit, sims, inc_warmup = FALSE)

get_CMdata(fit)

CM_trace(stanfit, vars, inc_warmup = FALSE)

CM_pairs(stanfit, vars = c("log_so", "log_cr"), inc_warmup = FALSE)

CM_fit_esc(report, d, year)

CM_fit_pHOS(report, d, year)

CM_fit_CWTesc(report, d, year1 = 1, rs_names)

CM_fit_CWTcatch(report, d, PT = TRUE, year1 = 1, rs_names)
```

```
CM_maturity(
  report,
  d,
  year1 = 1,
  r = 1,
  brood = TRUE,
  annual = FALSE,
  rs_names
)

CM_vul(report, type = c("vulPT", "vulT"))

CM_SRR(report, year1 = 1)

CM_prod(report, d, year1 = 1)

CM_Srep(report, d, year1 = 1, type = c("spawner", "egg"))

CM_M(report, year1 = 1, ci = TRUE)

CM_Megg(report, year1 = 1, ci = TRUE, surv = FALSE)

CM_Njuv(report, year1 = 1, ci = TRUE)

CM_recr(report, year1 = 1, ci = TRUE)

CM_esc(report, year1 = 1, ci = TRUE)

CM_F(report, PT = TRUE, year1 = 1, ci = TRUE)

CM_surv(report, year1 = 1, ci = TRUE)

CM_wt(stanfit, year1 = 1, ci = TRUE)

CM_surv2(report, year1 = 1, ci = TRUE, ylab = "Survival to age 2")

CM_wto(stanfit, year1 = 1, ci = TRUE)

CM_ER(
  report,
  brood = TRUE,
  type = c("PT", "T", "all"),
  year1 = 1,
  ci = TRUE,
  at_age = TRUE,
  r = 1
)
```

```
CM_CWT_ER(
  report,
  brood = TRUE,
  type = c("PT", "T", "all"),
  year1 = 1,
  ci = TRUE,
  rs_names
)

CM_covariate(x, names, year1 = 1, b, ylab = "Covariate")
```

Arguments

`stanfit`	Output from `sample_CM()`
`sims`	Optional integer vector for subset of MCMC iterations
`inc_warmup`	Logical, whether to include warmup MCMC samples
`fit`	Output from `[fit_CM()]`
`vars`	Character vector for variable names (see `names(stanfit@sim$samples[[1]])`). Regex and partial matching supported because it is passed to the `pattern` argument of `grepl()`
`report`	List, output of state variables from individual MCMC samples, obtained with `get_report()`
`d`	List of data variables, obtained with `get_CMdata()`
`year`	Vector of years
`year1`	Numeric, first year of model
`rs_names`	Character vector of hatchery release strategies
`PT`	Logical, whether to plot preterminal catch, otherwise (plot terminal catch)
`r`	Integer, the release strategy for the figure (only if `annual = FALSE`)
`brood`	Logical, whether to show results by brood year or return year (FALSE)
`annual`	Logical, whether to show panel figure by individual year (TRUE) or a single time series figure
`type`	Character, indicates type of variable to plot
`ci`	Logical whether to show posterior intervals in addition to posterior median
`surv`	Logical, whether to plot survival (values between 0 - 1) or instantaneous mortality rates
`ylab`	Character y axis label
`at_age`	Logical, whether to make figure by individual age
`x`	Matrix of covariates by year x covariate
`names`	Character of covariate names
`b`	Matrix of fixed effect coefficients by simulation x covariate. If missing only the covariates (`x`) are plotted, otherwise, the dot product `sum(x * b)` is calculated by individual simulation and quantiles are plotted

Value

- `get_report()` returns the list of state variables by individual MCMC samples
- `get_CMdata()` returns the list of data variables used in the conditioning model
- `CM_trace()` returns a ggplot showing the MCMC trace plot (aka wormplot)
- `CM_pairs()` returns output from `graphics::pairs()`, a matrix of scatterplots of MCMC posterior samples
- `CM_fit_esc()` returns base graphics with fit to total escapement time series
- `CM_fit_pHOS()` returns base graphics with fit to pHOS (census) observations
- `CM_fit_CWTesc()` returns ggplot of fit to CWT escapement at age
- `CM_fit_CWTcatch()` returns ggplot of fit to CWT catch at age
- `CM_maturity()` returns ggplot of estimated maturity at age
- `CM_vul()` returns ggplot of estimated fishery vulnerability at age
- `CM_SRR()` returns ggplot of estimated stock-recruit relationship (density-dependent juvenile production from egg production) with average relationship and realized annual values. Years correspond to return years of egg production (predicts juvenile production of the following calendar year).
- `CM_prod()` returns ggplot of realized productivity in the absence of fishery harvest, annual values are based on natural mortality and maturity at age
- `CM_Srep()` returns ggplot of realized spawner or egg production at replacement, annual values are based on natural mortality and maturity at age
- `CM_M()` returns ggplot of estimated natural mortality time series by age (marine stage)
- `CM_Megg()` returns ggplot of egg-juvenile mortality time series
- `CM_Njuv()` returns ggplot of juvenile abundance
- `CM_recr()` returns ggplot of recruitment (mature return)
- `CM_esc()` returns ggplot of escapement (after terminal harvest)
- `CM_F()` returns ggplot of instantaneous fishing mortality
- `CM_surv()` returns ggplot of natural survival (converting from instantaneous units of natural mortality)
- `CM_wt()` returns ggplot of annual deviations in egg-juvenile mortality from the Ricker function
- `CM_surv2()` returns ggplot of annual survival to age 2, which includes age-1 mortality (marine life stage) for both natural and hatchery origin fish. Hatchery fish experience additional mortality specified by release mortality.

- `CM_wt()` returns ggplot of annual deviations in age 1 natural mortality (first year in marine life stage, deviations from time series average)
- `CM_ER()` returns ggplot of exploitation rate either by individual age or aggregate values using adult equivalents
- `CM_CWT_ER()` returns ggplot of CWT exploitation rate (by release strategy)
- `CM_covariate()` returns ggplot of mortality covariates

`glossary` *salmonMSE glossary*

Description

Glossary of terms and parameters used in salmonMSE

Examples

```
data(glossary)
glossary[1:2, ]
```

`Habitat-class` *Class* `"Habitat"`

Description

The component of the operating model that controls survival in the freshwater environment. Includes changes in survival from either environmental/climate effects or habitat mitigation.

Slots

`Name` Character. Identifying name

`use_habitat` Logical. If `TRUE`, utilize stage-specific density-dependent functions from egg production from incubation mortality, egg-to-fry production, and fry-to-smolt production with annual deviations. Otherwise, the density-dependence is modeled for egg-to-smolt survival. See `Bio` object.

`prespawn_rel` Character, density-dependent function for pre-spawn mortality, e.g., for spawners to reach spawning sites. Choices are "BH" (Beverton-Holt) or "HS" (hockey stick). Default is "BH".

`prespawn_prod` Numeric, productivity for pre-spawn mortality. Default is 1. Default if `Inf`.

`prespawn_capacity` Numeric, capacity for pre-spawn mortality. Default is `Inf`, i.e., density-independence. Default is `Inf`.

`egg_rel` Character, density-dependent function for egg production from total spawning output. Choices are "BH" (Beverton-Holt) or "HS" (hockey stick). Default is "BH".

`egg_prod` Numeric, productivity for egg production from total spawning output (incubation). Default is 1. Default if `Inf`.

`egg_capacity` Numeric, capacity for egg production from total spawning output (incubation). Default is `Inf`, i.e., density-independence. Default is `Inf`.

`fry_rel` Character, density-dependent function for egg-to-fry production. Choices are "BH" (Beverton-Holt) or "HS" (hockey stick). Default is "BH".

`fry_prod` Numeric between 0-1, productivity for egg production from total spawning output, i.e., maximum survival as egg production approaches zero. Default is 0.4.

`fry_capacity` Numeric, capacity for fry production from egg production. Default is `Inf`, i.e., for density-independence. Default is `Inf`.

`fry_sdev` Matrix `[nsim, proyears]`, deviations from the density-dependent egg-fry survival. Can be utilized to incorporate time-varying environmental, climate, or habitat mitigation effects. Default is `matrix(1, nsim, proyears)`.

`smolt_rel` Character, density-dependent function for fry-to-smolt production. Choices are "BH" (Beverton-Holt) or "HS" (hockey stick). Default is "BH".

`smolt_prod` Numeric between 0-1, productivity for smolt production from fry, i.e., maximum survival as fry production approaches zero. Default is 1.

`smolt_capacity` Numeric, capacity for smolt production from fry production. Set to `Inf` for density-independence. Default is `Inf`.

`smolt_sdev` Matrix `[nsim, proyears]`, deviations from the density-dependent fry-smolt survival. Can be utilized to incorporate time-varying environmental, climate, or habitat mitigation effects. Default is `matrix(1, nsim, proyears)`.

Creating Object

Objects can be created by calls of the form `new("Habitat")`

Examples

```
showClass("Habitat")
```

`Harvest-class` *Class* `"Harvest"`

Description

The component of the operating model that controls marine harvest.

Slots

`Name` Character. Identifying name

`type_PT` Character. Whether to manage preterminal fishery catch from exploitation rate ("u") or catch target ("catch"). Default is "u", `type_T` Character. Whether to manage terminal fishery catch from exploitation rate ("u") or catch target ("catch"). Default is "u",

`u_preterminal` Numeric. If `type_PT = "u"`, the exploitation rate of the immature stock in the pre-terminal fishery. This will be converted to an instantaneous fishing mortality rate, i.e., `F_preterminal = -log(1 - u_preterminal)`.

`u_terminal` Numeric. If `type_T = "u"`, The exploitation rate of the return in the terminal fishery. This will be converted to an instantaneous fishing mortality rate, i.e., `F_terminal = -log(1 - u_terminal)`.

`K_PT` Numeric. If `type_PT = "catch"`, the catch target of the immature stock in the pre-terminal fishery.

`K_T` Numeric. If `type_T = "catch"`, the catch target of the return in the terminal fishery.

`MSF_PT` Logical. Whether to implement mark-selective fishing in the preterminal fishery, with no retention on unmarked fish.

`MSF_T` Logical. Whether to implement mark-selective fishing in the terminal fishery, with no retention on unmarked fish.

`release_mort` Vector length 2. The proportion of released fish that die after release, in the preterminal and terminal fishery. Implemented to model mark-selective fishing. Not used if either `MSF_PT` or `MSF_T` is `FALSE`.

`vulPT` Vector length `maxage` or matrix `[nsim, maxage]`. Vulnerability schedule (between 0-1) in the preterminal fishery. Values indicate the proportion of fishing mortality experienced by each age class, where `F_preterminal = -log(1 - u_preterminal)`.

`vulT` Vector length `maxage` or matrix `[nsim, maxage]`. Vulnerability schedule (between 0-1) in the terminal fishery. Values indicate the proportion of fishing mortality experienced by each age class, where `F_terminal = -log(1 - u_terminal)`.

Creating Object

Objects can be created by calls of the form `new("Harvest")`

Examples

```
showClass("Harvest")
```

`Hatchery-class` *Class* `"Hatchery"`

Description

The component of the operating model that controls the hatchery management and in-river removals.

Details

Various parameters can be stochastic (length `nsim`) or input as a single numeric (value identical across all simulations).

A description of the fitness parameters and hatchery dynamics is available in the online documentation.

Slots

`Name` Character. Identifying name

`n_r` Integer. Number of release strategies, sub-groups of fish with different survival schedules. Default is one.

`n_yearling` Vector length `n_r`. The target number of yearlings released by release strategy. No hatchery is modeled if `sum(n_yearling, n_subyearling) = 0`. are zero. Default is zero. Yearlings are differentiated from sub-yearlings only in the hatchery survival parameter, e.g., lower egg-yearling survival than for egg-subyearling, so more brood is needed for yearling releases.

`n_subyearling` Vector length `n_r`. The target number of subyearlings released. No hatchery is modeled if `sum(n_yearling, n_subyearling) = 0` are zero. Default is zero. Yearlings are differentiated from sub-yearlings only in the hatchery survival parameter, e.g., lower egg-yearling survival than for egg-subyearling, so more brood is needed for yearling releases.

`yearling_DD` Logical, whether freshwater yearling survival after release is density-dependent, in competition with natural-origin juveniles. Default is FALSE. Likely relevant only for stream types where there is substantial residence time before outmigration.

`subyearling_DD` Logical, whether freshwater subyearling survival after release is density-dependent, in competition with natural-origin juveniles. Default is FALSE. Likely relevant only for stream types where there is substantial residence time before outmigration.

`s_prespawn` Numeric. The survival of broodtake prior to egg production. `1 - s_prespawn` is the proportion of fish not used for hatchery purposes, e.g., mortality or other resesarch purposes. Used to back-calculate the broodtake from `n_yearling` and `n_subyearling`. Default is 1.

`s_egg_smolt` Numeric. The survival of eggs to the smolt life stage (for yearling release). Used to back-calculate the broodtake from `n_yearling` and `n_subyearling`. Default is 1.

`s_egg_subyearling` Numeric. The survival of eggs to subyearling life stage (for subyearling release). Used to back-calculate the broodtake from `n_yearling` and `n_subyearling`. Default is 1.

`Mjuv_HOS` Either vector by age (length `maxage-1`) or an array with dimension `[nsim, maxage-1, proyears, n_r]`. Natural mortality of immature hatchery origin fish. To replicate the SAR parameter of a stage-specific model, set `Mjuv_HOS[a] = -log(SAR)` for the age class prior to maturation (and zero for all other ages).

`p_mature_HOS` Vector by age (length `maxage`) or an array with dimension `[nsim, maxage, proyears, n_r]` for the maturity of hatchery spawners. Default is set equal to `Bio@p_mature` for all release strategies.

`stray_external` Matrix by age (length `maxage`) and release strategy `n_r` that denotes the annual number of hatchery origin strays from other populations/systems not included in the operating model. Default is zero. External strays are added at the escapement life stage, assumed unmarked. For multi-population models with straying within the system, see also `SOM@stray` matrix.

`gamma` Numeric. The relative reproductive success of hatchery origin spawners (relative to natural origin spawners). Default is 1.

`m` Numeric. The mark rate of hatchery origin fish, which affects selective broodtake and fishery retention if mark-selective fishing is utilized. Set m = 1 for AHA compatibility with `ptarget_NOB`. Default is zero.

`f_brood` Function that calculates the natural origin brood and hatchery origin brood from the escapement (after en-route mortality). Function should be of the form `function(NO, HO, stray, m) {return(list(NOB, HOB_marked, HOB_unmarked, HOB_stray))}`. Allows for bespoke rules for broodtake.

`brood_import` **Not used if** `f_brood` **is provided** Matrix by age (length `maxage`) for the number of annual imported hatchery origin broodstock. Egg production is weighted by `fec_brood`. To meet hatchery production target releases, imported brood and local marked brood are used indiscriminately. Default imported brood is zero.

`pmax_esc` **Not used if** `f_brood` **is provided** Numeric. The maximum proportion of total escapement (after en route mortality) that could be used as broodtake. Set to 1 for AHA compatibility. Default is 0.75.

`pmax_NOB` **Not used if** `f_brood` **is provided** Numeric. The maximum proportion of the natural origin escapement (after en route mortality and `pmax_esc`) to be used as broodtake. If broodstock is limited by `pmax_esc < 1`, then this parameter should be 1. Default is 1.

`ptarget_NOB` **Not used if** `f_brood` **is provided** Numeric. The target proportion of the natural origin broodtake relative to the overall broodtake, assuming the mark rate is 1 and natural origin fish can be identified in the hatchery. The realized proportion may be lower if there are insufficient natural origin escapement. If the mark rate < 1, then this target proportion identifies the proportion of unmarked fished in the broodtake. If mark rate = 0, then pNOB is equal to the proportion in the escapement. Default is 0.9.

`phatchery` **Not used if** `f_brood` **is provided** Numeric. Optional parameter (default is `NA`). If set to a numeric between 0-1, this value is the proportion of the hatchery origin escapement that return to the hatchery, for example, by removal from spawning grounds or swim-in facilities. These fish are available for broodtake. None of these fish will spawn in the natural environment. With the default option, `NA` allows all hatchery origin escapement to be available for brood (the remainder go to the spawning grounds).

`premove_HOS` Numeric or function. The target proportion of the hatchery origin fish to be removed from the spawning grounds (in order to ensure a high proportion of NOS). The proportion of hatchery fish removed is discounted by the mark rate, i.e., `p = premove_HOS * m`. The removed hatchery-origin fish do not spawn and are not available for broodtake. A value less than one can represent imperfect implementation of weir removal. Default is zero. This slot can take a function that returns the proportion (p) based on hatchery-origin and natural-origin abundance (after brood removal), thus allowing for bespoke rules for in-river harvest. The function should be of the form: `function(NO, HO, m) {return(p)}`.

`premove_NOS` Numeric or function. The target proportion of the natural origin fish to be removed from the spawning grounds, for example, through an in-river fishery. The proportion of natural fish removed is adjusted by the mark rate, i.e., `p = premove_NOS * (1-m)`. Default is zero. This slot can take a function that returns the proportion (p) based on hatchery-origin and natural-origin abundance (after brood removal), thus allowing for bespoke rules for in-river harvest. The function should be of the form: `function(NO, HO, m) {return(p)}`.

`fec_brood` Vector of length `maxage` or an array with dimension `[nsim, maxage, proyears]`. The fecundity schedule of broodtake to calculate the total egg production for the hatchery. If missing, uses `Bio@fec`.

`fitness_type` Character vector length 2. The fitness function to apply in the natural and hatchery environment, respectively. For each, either "Ford" or "none".

`theta` Vector length 2. The optimum phenotype value for the natural and hatchery environments.

`rel_loss` Vector length 3. The loss in fitness apportioned among the egg, fry, and smolt life stages which reduces survival. Theoretically, the three values should sum to 1. Alternatively, set to zero to set fitness loss to zero for that specific life stage (survival is one).

`zbar_start` Vector length 2. The mean phenotype value in the natural and hatchery populations at the start of the projection. Alternatively, an array by dimension `[nsim, maxage, 2]`, where the age slot corresponds to cohort.

`fitness_variance` Numeric. The variance (omega-squared) of the fitness function. Assumed identical between the natural and hatchery environments. Default is 100.

`phenotype_variance` Numeric. The variance (sigma-squared) of the phenotypic trait (theta). Assumed identical between the natural and hatchery environments. Default is 10.

`heritability` Numeric or vector length `[nsim]`. The heritability (h-squared) of the phenotypic trait. Between 0-1. Default is 0.5

`fitness_floor` Numeric. The minimum fitness value in the natural and hatchery environments, i.e., fitness cannot drop below this threshold. Default is 0.5.

Creating Object

Objects can be created by calls of the form `new("Hatchery")`

Examples

```
showClass("Hatchery")
```

`Historical-class` *Class* `"Historical"`

Description

Optional component of the operating model that specifies the historical dynamics. The default option starts the projection with 1000 natural-origin juveniles and 1000 hatchery-origin juveniles (if there is hatchery production) in the oldest age class, representing single-brood year returns since there is only one age class in the population. Specify the abundance in all age class to simulate multiple brood-year returns.

Slots

`Name` Character. Identifying name

`InitNjuv_NOS` Array by `[nsim, maxage, n_g]`. The abundance of immature natural origin fish at the beginning of the projection. Default assumes 1000 in the oldest age class, which creates a population with single brood year returns.

`InitNjuv_HOS` Array by `[nsim, maxage, n_r]`. The abundance of immature hatchery origin fish at the beginning of the projection. Default assumes 1000 in the oldest age class, which creates a population with single brood year returns.

Creating Object

Objects can be created by calls of the form `new("Historical")`

Examples

```
showClass("Historical")
```

`make_Harvest_MMP` *Internal salmonMSE functions for converting operating model inputs and outputs*

Description

- `SOM2MOM()` converts a salmon operating model (SOM) to a multi-stock operating model (MSEtool::MOM)
- `make_Stock()` creates the MSEtool::Stock object (openMSE) corresponding to salmon life stage
- `make_Fleet()` creates the MSEtool::Fleet object (openMSE) corresponding to the fishery that interacts with the various salmon life stages
- `multiHist2SHist()` converts the openMSE historical reconstruction into a salmon Hist object (SHist)
- `MMSE2SMSE()` converts the openMSE projection output, along with additional state variables recorded in salmonMSE_env, into a salmon MSE object (SMSE)
- `make_Harvest_MMP()` creates a multi-stock management procedure for the harvest component of the operating model by specifying exploitation rates through updating the formal arguments for `Harvest_MMP()`

`salmonMSE()` is the wrapper function that coordinates the simulation and the output.

Usage

```
make_Harvest_MMP(SOM, check = TRUE)

MMSE2SMSE(MMSE, SOM, Harvest_MMP, N, stateN, Ford, H, stateH)

SOM2MOM(SOM, check = TRUE)

make_Stock(
  SOM,
  s = 1,
  g = 1,
  r = 1,
  NOS = TRUE,
  stage = c("immature", "return", "escapement")
)

make_Fleet(SOM, s, NOS = TRUE, stage = c("immature", "return", "escapement"))

multiHist2SHist(multiHist, SOM, check = TRUE)
```

Arguments

`SOM`	An object of class SOM
`check`	Logical, whether to check the `SOM` object using `check_SOM()`
`MMSE`	Object of class MSEtool::MMSE returned from MSEtool
`Harvest_MMP`	Optional harvest function created by `make_Harvest_MMP()`
`N`	Data frame of natural origin abundance at age saved in the salmonMSE_env environment during the simulation
`stateN`	Data frame of natural origin state variables saved in the salmonMSE_env environment during the simulation
`Ford`	Data frame of phenotypic trait values saved in the salmonMSE_env environment during the simulation
`H`	Data frame of hatchery origin abundance at age saved in the salmonMSE_env environment during the simulation
`stateH`	Data frame of hatchery origin state variables saved in the salmonMSE_env environment during the simulation
`s`	Integer, the population integer for which to create the Stock or Fleet object
`g`	Integer, the life history group for which to create the Stock object. Not relevant if `NOS = FALSE`
`r`	Integer, the hatchery release group for which to create the Stock object. Not relevant if `NOS = TRUE`
`NOS`	Logical, whether the Stock or Fleet object corresponds to natural origin or hatchery origin fish
`stage`	Character indicating the corresponding salmon life stage of the Stock or Fleet object
`multiHist`	Class multiHist object returned from MSEtool

Value

`make_Harvest_MMP`: Function of class "MMP" by updating the formal arguments for `Harvest_MMP()`

`MMSE2SMSE`: SMSE object

`SOM2MOM`: MSEtool::MOM object

`make_Stock`: List containing a MSEtool::Stock object and accompanying custom parameters list

`make_Stock`: List containing a MSEtool::Fleet object and accompanying custom parameters list

`multiHist2SHist`: SHist object

`plot_decision_table` *Decision table of performance metrics*

Description

Generates a coloured table of a performance metric across two axes, which may be a population dynamics variable (e.g., productivity) or a management action (e.g., hatchery production levels or harvest strategy). See example at `https://docs.salmonmse.com/articles/decision-table.html`. More examples below.

- `plot_decision_table()` is a simple figure where colour range is intended to continuously transition from pink to white to green corresponding to values of 0, 0.5, and 1, respectively.
- `plot_decision_table2()` is converts performance metrics values into bins and provides more user control in the colour scheme

Usage

```
plot_decision_table(
  x,
  y,
  z,
  title,
  xlab,
  ylab,
  scenario,
  ncol = NULL,
  dir = "v"
)

plot_decision_table2(
  x,
  y,
  z,
  title,
  xlab,
  ylab,
  zlab,
  scenario,
  ncol = NULL,
  dir = "v",
  bin = c(0, 0.05, 0.25, 0.5, 0.75, 0.95),
  bin_labels = c("0-0.04", "0.05-0.24", "0.25-0.49", "0.5-0.74", "0.75-0.94", "0.95-1"),
  bin_col = c("purple4", "deeppink", "pink", "white", "green", "green4"),
  cell_border = FALSE,
  add_values = FALSE
)
```

Arguments

`x`	Atomic, vector of values for the x axis (same length as z). Will be converted to factors
`y`	Atomic, vector of values for the y axis (same length as z). Will be converted to factors
`z`	Numeric, vector of values for the performance metric
`title`	Character, optional title of figure
`xlab`	Character, optional x-axis label
`ylab`	Character, optional y-axis label
`scenario`	Atomic, vector of faceting variables (same length as z) used to generate a grid of decision tables
`ncol`	Integer, number of columns for decision table grid, only used if `scenario` is provided
`dir`	Character, either "h" or "v" to describe how the grid of tables should be organized (horizontally or vertically)
`zlab`	Character, optional color legend
`bin`	Numeric vector of bins to sort values of `z`
`bin_labels`	Character vector for bin names for the figure
`bin_col`	Character vector of colors for the bins in the figure
`cell_border`	Logical, whether to add borders for each cell in the figure
`add_values`	Logical, whether to add the values of `z` in the figure

Value

ggplot object

See Also

`plot_statevar_ts()` `plot_tradeoff()`

Examples

```
# Simple decision table
results <- data.frame(
  PNI = c(0.7, 0.23, 0.05, 0.9, 0.85, 0.74, 0.95, 0.92, 0.9),
  pNOB = rep(c(0.5, 0.75, 1), each = 3),
  ER = rep(c(0.2, 0.3, 0.4), 3),
  scenario = "High productivity"
)
plot_decision_table(
  x = results$ER,
  y = results$pNOB,
  z = results$PNI,
  title = "PNI",
  xlab = "Exploitation rate",
```

```
  ylab = "pNOB target"
)

# Multiple decision tables organized by scenario
# Continuing from above
results_low <- results
results_low$scenario <- "Low productivity"
results_low$PNI <- 0.5 * results$PNI

results_all <- rbind(results, results_low)
plot_decision_table(
  x = results_all$ER,
  y = results_all$pNOB,
  z = results_all$PNI,
  title = "PNI",
  xlab = "Exploitation rate",
  ylab = "pNOB target",
  scenario = results_all$scenario
)
# Example of binned decision table

df <- expand.grid(
  SAR = seq(0.005, 0.03, 0.005),
  ER = seq(0, 0.5, 0.1)
)
df$value <- ifelse(5 * df$SAR + 0.2 > df$ER, 0.75, 0.05)
df$value <- ifelse(df$SAR < 0.02 & df$ER > 0.4, 0.04, df$value)

plot_decision_table2(
  x = df$SAR,
  y = df$ER,
  z = df$value,
  xlab = "Marine survival",
  ylab = "Exploitation rate",
  zlab = "Probability\nof objective"
)
```

`plot_LHG` *Plot life history groups and release strategies*

Description

Plot the annual proportions of life history groups (natural origin fish) or release strategies (hatchery origin) at various life stages

Usage

```
plot_LHG(
  SMSE,
  var = "NOS",
```

```
    type = c("prop", "abs"),
    s = 1,
    FUN = median,
    figure = TRUE,
    xlab = "Projection Year",
    ylab,
    name,
    ylim
  )

  plot_RS(
    SMSE,
    var = "HOS",
    type = c("prop", "abs"),
    s = 1,
    FUN = median,
    figure = TRUE,
    xlab = "Projection Year",
    ylab,
    name,
    ylim
  )
```

Arguments

`SMSE`	Class SMSE object returned by `salmonMSE()`
`var`	Character. Slot for the state variables in `SMSE@Misc$LHG[[1]]` or `SMSE@Misc$RS[[1]]`.
`type`	Character to indicate whether to plot proportion or absolute numbers
`s`	Integer. Population index for multi-population model (e.g., `s = 1` is the first population in the model)
`FUN`	Summarizing function across simulations, typically `median()` or `mean()`
`figure`	Logical, whether to generate a figure (set to FALSE if only using the function to return the data matrix)
`xlab`	Character. Name of time variable for the figure
`ylab`	Character. Name of the state variable for the figure
`name`	Character. Vector of names for the life history groups or release strategies
`ylim`	Vector length 2, y-axis limits

Value

Base graphics figure, barplot of distribution or total numbers by LHG or RS. Returns invisibly the matrix of plotted values

See Also

`plot_statevar_ts()`

`plot_statevar_ts` *Plot core output from salmonMSE*

Description

Various functions that plot the state variables from salmonMSE projections:

- `plot_statevar_ts()` produces a time series for all simulations, or with medians and 95th percentile intervals
- `plot_statevar_hist()` produces a histogram across all simulations for a particular year
- `plot_spawners()` produces a summary barplot of spawners, including NOS, HOS, and wild spawners
- `plot_escapement()` produces a summary figure of the proportion of spawners and broodtake to escapement
- `plot_fitness()` produces a summary figure of metrics (fitness, PNI, pHOS, and pWILD) related to hatchery production
- `plot_fishery()` produces a summary figure of metrics related to the fishery, e.g., median catch, exploitation rate or harvest rate

Usage

```
plot_statevar_ts(
  SMSE,
  var = "PNI",
  s = 1,
  figure = TRUE,
  xlab = "Projection Year",
  quant = FALSE,
  ylab = var,
  ylim,
  agg.fun = sum,
  ...
)

plot_statevar_hist(SMSE, var = "PNI", s = 1, y, figure = TRUE, xlab = var, ...)

plot_spawners(SMSE, s = 1, prop = TRUE, FUN = median, figure = TRUE, ylim)

plot_fitness(SMSE, s = 1, FUN = median, figure = TRUE, ylim)

plot_escapement(SMSE, s = 1, FUN = median, figure = TRUE, ylim)

plot_fishery(
  SMSE,
  s = 1,
  type = c("catch", "exploit", "harvest"),
```

```
  FUN = median,
  figure = TRUE,
  ylim,
  ylab,
  ...
)

plot_Kobe(
  SMSE,
  s = 1,
  FUN = median,
  figure = TRUE,
  xlim,
  ylim,
  xlab = expression(NOS/S[MSY]),
  ylab = expression(U/U[MSY]),
  type = c("T", "PT")
)
```

Arguments

`SMSE`	Class SMSE object returned by `salmonMSE()`
`var`	Character. Slot for the state variable in `SMSE` object. See `slotNames(SMSE)` for options. Additional supported options are: `"ESS"` (egg-smolt survival), `"pbrood"` (broodtake to escapement ratio), `"pNOSesc"` (NOS/natural escapement), `"pHOSesc"` (HOS/hatchery escapement), `Total Spawners` (NOS + HOS), `NOS/SMSY`, `S/SMSY`, and `NOS/Sgen`.
`s`	Integer. Population index for multi-population model (e.g., `s = 1` is the first population in the model)
`figure`	Logical, whether to generate a figure (set to FALSE if only using the function to return the data matrix)
`xlab`	Character. Name of time variable for the figure
`quant`	Logical, whether to plot individual simulations (FALSE) or the median with 95 percent confidence intervals (TRUE)
`ylab`	Character. Name of the state variable for the figure
`ylim`	Vector. Y-axis limits
`agg.fun`	Function. Defines how to aggregate state variables that are reported by age. Typically, `sum` is used but `max` is also possible for reporting apical exploitation rates.
`...`	Additional arguments to base plot function
`y`	Integer. Projection year for the state variable to plot the histogram. If missing, the last projection year is used.
`prop`	Logical, whether to plot proportions or absolute numbers
`FUN`	Summarizing function across simulations, typically `median()` or `mean()`

`type`	For `plot_Kobe`, the fishery state variable to plot. Whether to plot the exploitation rate for the terminal (T) or pre-terminal fishery (PT).
`xlim`	Vector. X-axis limits

Value

Functions return the matrix of plotted values invisibly. Figure plotted from base graphics

See Also

`plot_decision_table()` `plot_LHG()` `compare_statevar_ts()`

`plot_tradeoff` *Tradeoff figure*

Description

Generates a tradeoff figure, a comparison between two performance metrics, across two variables which may represent a population dynamics variable (e.g., productivity) or a management action (e.g., hatchery production levels or harvest strategy). See example at `https://docs.salmonmse.com/articles/decision-table.html`

Usage

```
plot_tradeoff(
  pm1,
  pm2,
  x1,
  x2,
  xlab,
  ylab,
  x1lab,
  x2lab,
  scenario,
  scenario_rows,
  scenario_cols,
  ncol = NULL,
  dir = "v",
  add_letters = FALSE
)
```

Arguments

`pm1`	Numeric or matrix. A vector of values for the first performance metric on the x-axis. Alternatively, provide a three column matrix corresponding to the lower bound, central tendency, and upper bound.

`pm2`	Numeric or matrix. A vector of values for the second performance metric on the y-axis (same length as pm1). Alternatively, provide a three column matrix corresponding to the lower bound, central tendency, and upper bound.
`x1`	Atomic, vector of values for the first grouping variable. Various levels are represented by colours. Same length as pm1.
`x2`	Numeric, vector of values for the second grouping variable. Various levels are represented by shapes. Same length as pm1.
`xlab`	Character, optional x-axis label
`ylab`	Character, optional y-axis label
`x1lab`	Character, optional label for the first grouping variable
`x2lab`	Character, optional label for the second grouping variable
`scenario`	Atomic, vector of scenario names (same length as `pm1`, `pm2`) used to identify and create separate panels
`scenario_rows`	Atomic, vector of scenario variables (same length as `pm1`, `pm2`) used to identify the rows in a grid of tradeoff figures. Use with `scenario_cols`
`scenario_cols`	Atomic, vector of scenario variables (same length as `pm1`, `pm2`) used to identify the columns in a grid of tradeoff figures. Use with `scenario_cols`
`ncol`	Integer, number of columns in the grid, only used if `scenario` is provided
`dir`	Character, either "h" or "v" to describe how the grid of tables should be organized (horizontally or vertically) , only used if `scenario` is provided
`add_letters`	Logical, whether to identify separate panels by letters (only used if scenarios are identified). Helpful for publication-level figures

Value

ggplot object

See Also

`plot_statevar_ts()` `plot_decision_table()`

Examples

```
# Single tradeoff panel
results <- data.frame(
  PNI = c(0.7, 0.23, 0.05, 0.9, 0.85, 0.74, 0.95, 0.92, 0.9),
  Catch = c(10, 14, 12, 8, 7, 7, 8.2, 7.1, 6.9),
  pNOB = rep(c(0.5, 0.75, 1), each = 3),
  ER = rep(c(0.2, 0.3, 0.4), 3),
  scenario = "High productivity"
)

plot_tradeoff(
  pm1 = results$PNI,
  pm2 = results$Catch,
  x1 = results$ER,
  x2 = results$pNOB,
```

```
  xlab = "PNI",
  ylab = "Catch",
  x1lab = "Exploitation\nrate",
  x2lab = "pNOB\ntarget"
)

# Multiple panels, continuing from above code
results_low <- results
results_low$scenario <- "Low productivity"
results_low$PNI <- 0.5 * results$PNI
results_low$Catch <- c(0.9, 0.8, 0.7) * results$Catch

results_all <- rbind(results, results_low)
g <- plot_tradeoff(
  pm1 = results_all$PNI,
  pm2 = results_all$Catch,
  x1 = results_all$ER,
  x2 = results_all$pNOB,
  xlab = "PNI",
  ylab = "Catch",
  x1lab = "Exploitation\nrate",
  x2lab = "pNOB\ntarget",
  scenario = results_all$scenario
)

# Change legend with ggplot2
library(ggplot2)
g + scale_shape_manual(values = c(1, 4, 16))
```

`PNI50` *Example performance metrics*

Description

Functions that evaluate return probabilities of outcomes from the simulations.

Usage

```
PNI50(SMSE, Ref = 0.5, Yrs = NULL)

PNI80(SMSE, Ref = 0.8, Yrs = NULL)

WILD50(SMSE, Ref = 0.5, Yrs = NULL)

SMSY85(SMSE, Ref = 0.85, Yrs = NULL)

Sgen100(SMSE, Ref = 1, Yrs = NULL)
```

Arguments

`SMSE`	SMSE object returned by `salmonMSE()`
`Ref`	Threshold for the performance metric, used to calculate the probability that the metric exceeds this value
`Yrs`	Numeric vector of length 2 to indicate the year range over which to summarize performance. If NULL, the performance is summarized over all projection years.

Details

- `PNI50` calculates the probability that PNI exceeds 0.50 (threshold for an integrated-transition population, Withler et al. 2018)
- `PNI80` calculates the probability that PNI exceeds 0.80 (threshold for an integrated-wild population, Withler et al. 2018)
- `WILD50` calculates the probability that at least 50 percent of natural spawners are wild
- `SMSY85` calculates the probability that NOS/SMSY exceeds 0.85
- `Sgen100` calculates the probability that NOS/Sgen exceeds 1

Value

A vector of probabilities corresponding to population

References

Withler et al. 2018. Genetically Based Targets for Enhanced Contributions to Canadian Pacific Chinook Salmon Populations. DFO Can. Sci. Advis. Sec. Res. Doc. 2018/019. xii + 88 p.

`report` *Generate markdown reports*

Description

Generate a markdown report for outcomes from a single operating model projection

Usage

```
## S4 method for signature 'SMSE'
report(
  object,
  name = object@Name,
  filename = "SMSE",
  dir = tempdir(),
  open_file = TRUE,
  render_args = list(),
  ...
)
```

Arguments

`object`	SMSE object
`name`	Character string for the model name to include in the report, e.g., model run number.
`filename`	Character string for the name of the markdown and HTML files.
`dir`	The directory in which the markdown and HTML files will be saved.
`open_file`	Logical, whether the HTML document is opened after it is rendered.
`render_args`	List of arguments to pass to `rmarkdown::render()`.
`...`	Additional arguments (not used)

Value

Returns invisibly the output of `rmarkdown::render()`, typically the path of the output file

See Also

`compare()`

`report_CM` *Conditioning model markdown report*

Description

Generate a markdown report to plot time series and MCMC posteriors of estimates from the conditioning model. See `get_report()` for the various plotting functions used in the report.

Usage

```
report_CM(
  stanfit,
  year,
  cov1_names,
  cov_names,
  rs_names,
  name,
  filename = "CM",
  dir = tempdir(),
  open_file = TRUE,
  render_args = list(),
  ...
)
```

Arguments

`stanfit`	Output from `sample_CM()`
`year`	Optional vector of calendar years
`cov1_names`	Optional character vector for names of covariates that predict age-1 natural mortality
`cov_names`	Optional character vector for names of covariates that predict age-2+ natural mortality
`rs_names`	Optional character vector for names of hatchery release strategies
`name`	Optional character string for the model name to include in the report, e.g., model run number
`filename`	Character string for the name of the markdown and HTML files
`dir`	The directory in which the markdown and HTML files will be saved.
`open_file`	Logical, whether the HTML document is opened after it is rendered
`render_args`	List of arguments to pass to `rmarkdown::render()`
`...`	Additional arguments (not used)

Details

Report excludes MCMC values from warmup iterations

Value

Returns invisibly the output of `rmarkdown::render()`, typically the path of the output file

See Also

`fit_CM()` `get_report()`

`salmonMSE` *Run salmonMSE*

Description

`salmonMSE()` runs a salmon management strategy evaluation through the following steps:

- Converts a salmon operating model (SOM) to a multi-stock operating model (MSEtool::MOM) via `SOM2MOM()`
- Creates a harvest management procedure specifying the harvest control rule
- Generates the historical reconstruction of the state variables
- Runs projection (if `Hist = FALSE`)
- Converts the openMSE output, along with additional state variables recorded in salmonMSE_env, into a salmon MSE object (SMSE) via `MMSE2SMSE()`

Usage

```
salmonMSE(SOM, Hist = FALSE, silent = FALSE, trace = FALSE, convert = TRUE)
```

Arguments

`SOM`	An object of class SOM
`Hist`	Logical, whether to stop the function stop after historical simulations?
`silent`	Logical, whether to report progress in console
`trace`	Logical, whether to report additional messages from openMSE
`convert`	Logical, whether to convert the output into a salmon MSE (SHist or SMSE, depending on `Hist`) object

Value

If `Hist` = `TRUE`: if `convert` = `TRUE`, a SHist object or if `convert` = `FALSE`, a multiHist object (list).

If `Hist` = `FALSE`: if `convert` = `TRUE`, a SMSE object or if `convert` = `FALSE`, a MSEtool::MMSE object.

Examples

```
## Not run:
SMSE <- salmonMSE(simple_SOM)

## End(Not run)
```

`salmonMSE_env` *Environment to store salmon specific variables, e.g, fitness, during the simulation*

Description

Environment to store salmon specific variables, e.g, fitness, during the simulation

Usage

```
salmonMSE_env
```

Format

An object of class `environment` of length 5.

`SHist-class` *Class* `"SHist"`

Description

Stores the outputs from the historical reconstruction of salmon operating models. Primarily used for development and debugging.

Slots

`Name` Character. Identifying name

`nyears` Integer, number of historica years

`nsim` Integer. The number of simulations

`nstocks` Integer. The number of stocks

`Snames` Character. Stock names

`Egg_NOS` Array `[nsim, nstocks, nyears]`. Spawning output, i.e., egg production, of natural origin spawners.

`Egg_HOS` Array `[nsim, nstocks, nyears]`. Spawning output of hatchery origin spawners.

`Smolt` Array `[nsim, nstocks, nyears]`. Natural smolt production (sum of offspring of natural and hatchery spawners).

`Smolt_Rel` Array `[nsim, nstocks, proyears]`. Smolts that are offspring of broodtake, i.e., hatchery releases.

`Njuv_NOS` Array `[nsim, nstocks, nage, nyears]`. Abundance of juvenile natural origin fish at the beginning of the year.

`Njuv_HOS` Array `[nsim, nstocks, nage, nyears]`. Abundance of juvenile hatchery origin fish at the beginning of the year.

`Return_NOS` Array `[nsim, nstocks, nage, nyears]`. Mature fish that will be natural origin spawners.

`Return_HOS` Array `[nsim, nstocks, nage, nyears]`. Mature fish that will be hatchery origin spawners.

`Escapement_NOS` Array `[nsim, nstocks, nage, nyears]`. The escapement of mature fish that will be natural origin spawners.

`Escapement_HOS` Array `[nsim, nstocks, nage, nyears]`. The escapement of mature fish that will be hatchery origin spawners.

`NOS` Array `[nsim, nstocks, proyears]`. Natural origin spawners.

`HOS` Array `[nsim, nstocks, proyears]`. Hatchery origin spawners.

`HOS_effective` Array `[nsim, nstocks, proyears]`. Hatchery origin spawners discounted by `gamma`.

`KPT_NOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery kept catch of natural origin spawners.

`KT_NOS` Array `[nsim, nstocks, proyears]`. Terminal fishery kept catch of natural origin spawners.

`KPT_HOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery kept catch of hatchery origin spawners.

`KT_HOS` Array `[nsim, nstocks, proyears]`. Terminal fishery kept catch of hatchery origin spawners.

`DPT_NOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery released catch (live and dead) of natural origin spawners.

`DT_NOS` Array `[nsim, nstocks, proyears]`. Terminal fishery released catch (live and dead) of natural origin spawners.

`DPT_HOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery released catch (live and dead) of hatchery origin spawners.

`DT_HOS` Array `[nsim, nstocks, proyears]`. Terminal fishery released catch (live and dead) hatchery origin spawners.

`UPT_NOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery harvest rate (from kept catch) of natural origin spawners.

`UT_NOS` Array `[nsim, nstocks, proyears]`. Terminal fishery harvest rate of natural origin spawners.

`UPT_HOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery harvest rate of hatchery origin spawners.

`UT_HOS` Array `[nsim, nstocks, proyears]`. Terminal fishery harvest rate of hatchery origin spawners.

`ExPT_NOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery exploitation rate (from kept catch and dead releases) of natural origin spawners.

`ExT_NOS` Array `[nsim, nstocks, proyears]`. Terminal fishery exploitation rate of natural origin spawners.

`ExPT_HOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery exploitation rate of hatchery origin spawners.

`ExT_HOS` Array `[nsim, nstocks, proyears]`. Terminal fishery exploitation rate of hatchery origin spawners.

`Misc` List. Miscellaneous output

Examples

```
showClass("SHist")
```

`simple_SOM` *Example objects from salmonMSE*

Description

S4 objects containing operating model inputs to run the projection.

Usage

```
simple_SOM

multi_SOM
```

Format

An object of class SOM.

An object of class `SOM` of length 1.

Source

See salmonMSE website

Examples

```
data(simple_SOM)

data(multi_SOM)
```

`SMSE-class` *Class* `"SMSE"`

Description

Stores the outputs from the simulation of salmon operating models.

Details

In generation t, proportionate natural influence (PNI) is defined as:

$$\mathrm{PNI}_t = \frac{p_t^{\mathrm{NOB}}}{p_t^{\mathrm{NOB}} + p_t^{\mathrm{HOSeff}}}$$

with $p^{\mathrm{HOSeff}} = \mathrm{HOSeff}/(\mathrm{NOS} + \mathrm{HOSeff})$.

The proportion of wild salmon is defined as:

$$p_t^{\text{WILD}} = q_t^{\text{HOScen}} \frac{(q_{t-1}^{\text{HOScen}})^2}{(q_{t-1}^{\text{HOScen}})^2 + 2\gamma \times p_{t-1}^{\text{HOScen}} q_{t-1}^{\text{HOScen}} + \gamma^2 (p_{t-1}^{\text{HOScen}})^2}$$

where $q = 1 - p$ and $p^{\text{HOScen}} = \text{HOS}/(\text{NOS} + \text{HOS})$.

Slots

`Name` Character. Identifying name

`proyears` Integer. The number of projected years

`nsim` Integer. The number of simulations

`nstocks` Integer. The number of stocks

`Snames` Character. Stock names

`Egg_NOS` Array `[nsim, nstocks, proyears]`. Spawning output, i.e., egg production, of natural origin spawners.

`Egg_HOS` Array `[nsim, nstocks, proyears]`. Spawning output of hatchery origin spawners.

`Fry_NOS` Array `[nsim, nstocks, proyears]`. Fry that are offspring of natural origin spawners.

`Fry_HOS` Array `[nsim, nstocks, proyears]`. Fry that are offspring of hatchery origin spawners.

`Smolt_NOS` Array `[nsim, nstocks, proyears]`. Smolts that are offspring of natural origin spawners.

`Smolt_HOS` Array `[nsim, nstocks, proyears]`. Smolts that are offspring of hatchery origin spawners.

`Smolt_Rel` Array `[nsim, nstocks, proyears]`. Smolts that are offspring of broodtake, i.e., hatchery releases.

`Njuv_NOS` Array `[nsim, nstocks, nage, proyears]`. Abundance of juvenile natural origin fish at the beginning of the year.

`Njuv_HOS` Array `[nsim, nstocks, nage, proyears]`. Abundance of juvenile hatchery origin fish at the beginning of the year.

`Return_NOS` Array `[nsim, nstocks, nage, proyears]`. Mature fish that will be natural origin spawners.

`Return_HOS` Array `[nsim, nstocks, nage, proyears]`. Mature fish that will be hatchery origin spawners.

`Escapement_NOS` Array `[nsim, nstocks, nage, proyears]`. The escapement of mature fish that will be natural origin spawners.

`Escapement_HOS` Array `[nsim, nstocks, nage, proyears]`. The escapement of mature fish that will be hatchery origin spawners.

`NOB` Array `[nsim, nstocks, proyears]`. Natural origin broodtake.

`HOB` Array `[nsim, nstocks, proyears]`. Hatchery origin broodtake (local + strays).

`HOB_stray` Array `[nsim, nstocks, proyears]`. Hatchery origin broodtake (strays only).

`HOB_import` Array `[nsim, nstocks, proyears]`. Imported hatchery origin broodtake used for hatchery production.

`NOS` Array `[nsim, nstocks, nage, proyears]`. Natural origin spawners.

`HOS` Array `[nsim, nstocks, nage, proyears]`. Hatchery origin spawners (local + strays).

`HOS_stray` Array `[nsim, nstocks, nage, proyears]`. Hatchery origin spawners (strays only).

`HOS_effective` Array `[nsim, nstocks, nage, proyears]`. Hatchery origin spawners (local + strays) discounted by `gamma`.

`KPT_NOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery kept catch of natural origin spawners.

`KT_NOS` Array `[nsim, nstocks, proyears]`. Terminal fishery kept catch of natural origin spawners.

`KPT_HOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery kept catch of hatchery origin spawners.

`KT_HOS` Array `[nsim, nstocks, proyears]`. Terminal fishery kept catch of hatchery origin spawners.

`DPT_NOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery released catch (live and dead) of natural origin spawners.

`DT_NOS` Array `[nsim, nstocks, proyears]`. Terminal fishery released catch (live and dead) of natural origin spawners.

`DPT_HOS` Array `[nsim, nstocks, proyears]`. Pre-terminal fishery released catch (live and dead) of hatchery origin spawners.

`DT_HOS` Array `[nsim, nstocks, proyears]`. Terminal fishery released catch (live and dead) hatchery origin spawners.

`UPT_NOS` Array `[nsim, nstocks, nage, proyears]`. Pre-terminal fishery harvest rate (from kept catch) of natural origin spawners.

`UT_NOS` Array `[nsim, nstocks, nage, proyears]`. Terminal fishery harvest rate of natural origin spawners.

`UPT_HOS` Array `[nsim, nstocks, nage, proyears]`. Pre-terminal fishery harvest rate of hatchery origin spawners.

`UT_HOS` Array `[nsim, nstocks, nage, proyears]`. Terminal fishery harvest rate of hatchery origin spawners.

`ExPT_NOS` Array `[nsim, nstocks, nage, proyears]`. Pre-terminal fishery exploitation rate (from kept catch and dead releases) of natural origin spawners.

`ExT_NOS` Array `[nsim, nstocks, nage, proyears]`. Terminal fishery exploitation rate of natural origin spawners.

`ExPT_HOS` Array `[nsim, nstocks, nage, proyears]`. Pre-terminal fishery exploitation rate of hatchery origin spawners.

`ExT_HOS` Array `[nsim, nstocks, nage, proyears]`. Terminal fishery exploitation rate of hatchery origin spawners.

`fitness` Array `[nsim, nstocks, 2, proyears]`. Fitness of the population in the natural (1) and hatchery (2) environments.

`pNOB` Array `[nsim, nstocks, proyears]`. Proportion of natural fish in the brood.

`pHOS_census` Array `[nsim, nstocks, proyears]`. Proportion of spawners of hatchery origin, weighted by age class fecundity.

`pHOS_effective` Array `[nsim, nstocks, proyears]`. Proportion of spawners of hatchery origin, discounted by `gamma`, weighted by age class fecundity.

`PNI` Array `[nsim, nstocks, proyears]`. Proportionate natural influence, index of gene flow from hatchery to the natural environment.

`p_wild` Array `[nsim, nstocks, proyears]`. Proportion of wild spawners, natural spawners whose parents were also produced in the natural environment assuming non-assortative mating, defined under Canada's Wild Salmon Policy.

`Mjuv_loss` Array `[nsim, nstocks, nage, proyears]`. Realized juvenile natural mortality, which may differ from inputs due to fitness loss.

`Misc` List. Miscellaneous output:

- `Ref` for reference points
- `SHist` for the SHist object (primarily for developer use)
- `SOM` for the SOM object (updated by `check_SOM()`).
- `LHG` list `nstocks` long containing state variables by natural-origin life history group
- `RS` list `nstocks` long containing state variables by hatchery-origin release strategy

Creating Object

Objects can be created by calls of the form `new("SMSE")`

References

Withler et al. 2018. Genetically Based Targets for Enhanced Contributions to Canadian Pacific Chinook Salmon Populations. DFO Can. Sci. Advis. Sec. Res. Doc. 2018/019. xii + 88 p.

Examples

```
showClass("SMSE")
```

`SOM-class` *Class* `"SOM"`

Description

An object containing all the parameters for a salmon operating model (SOM).

Slots

`Name` Character. Identifying name

`nsim` Integer. Number of simulations

`proyears` Integer. The number of projected years

`seed` Integer. A random seed to ensure users can reproduce results exactly

`Bio` Bio object informing biological parameters and natural production. Provide a list of Bio objects for multi-population models.

`Habitat` Habitat object containing management levers for controlling survival in the freshwater environment. Provide a list of Habitat objects for multi-population models.

`Hatchery` Hatchery object containing management levers for hatchery production and in-river removals. Provide a list of Hatchery objects for multi-population models.

`Harvest` Harvest object containing management levers for marine harvest. Provide a list of Harvest objects for multi-population models.

`Historical` Historical object to inform historical reconstruction and informing starting abundance for the projection. Provide a list of Historical objects for multi-population models.

`stray` Matrix `[np, np]` where `np = length(Bio)` and row `p` indicates the re-assignment of hatchery fish to each population when they mature (at the recruitment life stage). For example, `SOM@stray <- matrix(c(0.75, 0.25, 0.25, 0.75), 2, 2)` indicates that 75 percent of mature fish return to their natal river and 25 percent stray in both populations. By default, an identity matrix is used (no straying).

Objects from the Class

Objects can be created by calls of the form `new("SOM", Bio, Habitat, Hatchery, Harvest, Historical)`.

Index