Package 'LeMaRns'

May 7, 2026

Title Length-Based Multispecies Analysis by Numerical Simulation

Version 0.1.2

Date 2019-12-09

Description Set up, run and explore the outputs of the Length-based Multispecies model (LeMans; Hall et al. 2006 <doi:10.1139/f06-039>), focused on the marine environment.

Depends R (>= 3.5.0)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.0.0

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp, abind, methods

Suggests knitr, rmarkdown

VignetteBuilder knitr

Collate 'NS_par.R' 'RcppExports.R' 'calc_M1.R' 'calc_Q.R' 'calc_growth.R' 'calc_indicators.R' 'calc_mature.R' 'param_setup.R' 'run_LeMans.R' 'calc_output.R' 'calc_phi.R' 'calc_prefs.R' 'calc_ration_growthfac.R' 'calc_recruits.R' 'calc_suit_vect.R' 'get_Catch.R' 'get_SSB.R' 'get_indicators.R' 'other.R' 'plot_SSB.R' 'plot_indicators.R'

NeedsCompilation yes

Author Michael A. Spence [aut, cre] (ORCID: <https://orcid.org/0000-0002-3445-7979>), Hayley J. Bannister [aut] (ORCID: <https://orcid.org/0000-0002-2546-5168>), Johnathan E. Ball [aut], Paul J. Dolder [aut] (ORCID: <https://orcid.org/0000-0002-4179-712X>), Robert B. Thorpe [aut] (ORCID: <https://orcid.org/0000-0001-8193-6932>), Christopher A. Griffiths [ctb]

Maintainer Michael A. Spence <michael.spence@cefas.co.uk>

Repository CRAN

Date/Publication 2019-12-09 15:10:02 UTC

Contents

calc_growth *Calculate growth*

Description

Calculates the number of individuals of each species in each length class for the next time step.

Usage

```
calc_growth(N, phi, nfish, nsc)
```

Arguments

N	A matrix with dimensions nsc and nfish representing the number of individuals in each length class for the current time step.
phi	A matrix with dimensions nsc and nfish representing the proportion of individuals that leave each length class.
nfish	A numeric value representing the number of species in the model.
nsc	A numeric value representing the number of length classes in the model.

Value

A matrix with dimensions nsc and nfish representing the number of individuals of each species in each length class for the next time step.

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf) * 1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                  phi_min=0.1) # fixed phi_min
phi <- tmp$phi
phi_min <- tmp$phi_min
```

```
# Calculate growth increments
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt

# Get an initial population
N0 <- get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept=1e10, slope=-5)

# Calculate growth
growth <- calc_growth(N0, phi, nfish, nsc)
```

`calc_LFI` *Calculate community indicators*

Description

Calculates the Large Fish Indicator (LFI), Typical Length (TyL) or Length Quantile (LQ) for the community or a subset of the species.

Usage

```
calc_LFI(wgt, N, bry, prop)

calc_LQ(wgt, u_bound, N, prob)

calc_TyL(wgt, mid, N)
```

Arguments

`wgt`	A matrix with dimensions `nsc` and `nfish` representing the weight of each species in each length class.
`N`	A matrix with dimensions `nsc` and `nfish` representing the number of individuals in each length class for the current time step.
`bry`	A numeric vector representing the length classes that are larger than `length_LFI`.
`prop`	A numeric value between 0 and 1 representing how far along, as a proportion, the value of the large indicator threshold is in the length class that contains it.
`u_bound`	A numeric vector of length `nsc` representing the upper bounds of the length classes.
`prob`	A numeric value or vector between 0 and 1 denoting the LQ to be calculated.
`mid`	A numeric vector of length `nfish` representing the mid-point of the length classes in the model.

Value

calc_LFI returns a numeric value or vector representing the proportion of the biomass in the length classes above bry and in prop of the biomass in the length class bry.

calc_TYL returns a numeric value or vector representing the biomass-weighted geometric mean length of the community.

calc_LQ returns a numeric value or vector representing the length at which biomass exceeds a given proportion prob of the total biomass.

See Also

get_indicators

Examples

```
# Set up and run the model
NS_params <- LeMansParam(NS_par, tau=NS_tau, eta=rep(0.25, 21), L50=NS_par$Lmat, other=1e11)
effort <- matrix(0.5, 10, dim(NS_params@Qs)[3])
model_run <- run_LeMans(NS_params, years=10, effort=effort)

# Calculate the LFI for 40cm
bry <- which(NS_params@l_bound<=40 & NS_params@u_bound>40)
length_LFI <- 40
prop <- (length_LFI-NS_params@l_bound[bry])/(NS_params@u_bound[bry]-NS_params@l_bound[bry])
LFI <- calc_LFI(model_run@N[,,101], NS_params@wgt, bry, prop)

# Calculate TyL for the final time step
calc_TyL(wgt=NS_params@wgt, mid=NS_params@mid, N=model_run@N[,,101])

# Calculate the LQ for the final time step
calc_LQ(wgt=NS_params@wgt, u_bound=NS_params@u_bound, N=model_run@N[,,101], prob=0.5)
```

---

calc_M1 *Calculate background mortality*

Description

Calculates the background mortality of each species in each length class per time step.

Usage

```
calc_M1(
  nsc,
  sc_Linf,
  phi_min,
  natmort_opt = rep("std_RNM", length(sc_Linf)),
  Nmort = rep(0.8, length(sc_Linf)),
  prop = rep(0.75, length(sc_Linf))
```

```
)

get_M1(nsc, sc_Linf, natmort_opt, Nmort, prop)

calc_M1_stdRNM(nsc, sc_Linf, Nmort = 0.8, prop = 0.75)

calc_M1_lin(nsc, sc_Linf, Nmort = 0.8)

calc_M1_constRNM(nsc, sc_Linf, Nmort = 0.8)
```

Arguments

nsc	A numeric value representing the number of length classes in the model.
sc_Linf	A numeric vector of length nfish for calc_M1 and 1 for get_M1, calc_M1_stdRNM, calc_M1_constRNM and calc_M1_lin, representing the length class at which each species reaches its asymptotic length.
phi_min	A numeric value representing the time step of the model.
natmort_opt	A character vector of length nfish for calc_M1 and 1 for get_M1, describing the mortality function to be used for each species. By default, natmort_opt takes the value "std_RNM", but it can also take "constant" or "linear". See 'Details' for more information.
Nmort	A numeric vector of length nfish representing the maximum background mortality of each species. The default is 0.8 for all species.
prop	A numeric vector of length nfish representing the proportion of length classes that have a non-zero background mortality. This parameter is required only when the natmort_opt mortality function is used. The default is 0.75.

Details

The background mortality is defined as the number of individuals that die per year, but not from predation or fishing

```
N*exp(-M1)
```

where N is the total number of individuals. This function allows three different models for background mortality: (1) "constant", which gives M1=Nmort for all length classes up to and including the sc_Linf-th class; (2) "std_RNM", which gives M1=Nmort for all length classes between floor(sc_Linf*prop) and the sc_Linf-th class and M1=0 for the rest; and (3) "linear", which gives M1=0 for the first length class, followed by a linear increase in M1 up to and including the sc_Linf-th length class M1=Nmort and M1=0 for the rest.

Value

calc_M1 returns a matrix of dimensions nsc and nfish representing the background mortality of each species for each length class.

get_M1 returns a numeric vector of length nsc representing the background mortality for each length class.

calc_M1_stdRNM returns a numeric vector of length nsc representing the background mortality for each length class. M1=Nmort for all length classes between floor(sc_Linf*prop) and the sc_Linf-th class and M1=0 for the rest.

calc_M1_lin returns a numeric vector of length nsc representing the background mortality for each length class. M1=0 for the first length class. M1 then increases linearly up to and including the sc_Linf-th length class. For all length classes above sc_Linf, M1=0.

calc_M1_constRNM returns a numeric vector of length nsc representing the background mortality for each length class. M1=Nmort for all length classes up to and including the sc_Linf-th class. For all length classes above sc_Linf, M1=0.

References

Thorpe, R.B., Jennings, S., Dolder, P.J. (2017). Risks and benefits of catching pretty good yield in multispecies mixed fisheries. *ICES Journal of Marine Science*, 74(8):2097-2106.

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                  phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Calculate growth increments
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf

# Calculate background mortality with natmort_opt="std_RNM", Nmort=0.8 and prop=0.75 for all species
M1 <- calc_M1(nsc, sc_Linf, phi_min)

# Get background mortality with natmort_opt="std_RNM", Nmort=0.8 and prop=0.75 for all species
natmort_opt <- "std_RNM"
Nmort <- 0.8
prop <- 0.75
get_M1(nsc, sc_Linf, natmort_opt, Nmort, prop)

# Calculate standard residual background mortality
```

```
M1_stdRNM <- calc_M1_stdRNM(nsc, sc_Linf)

# Calculate linear background mortality
M1_lin <- calc_M1_lin(nsc, sc_Linf,Nmort=0.3)

# Calculate constant residual background mortality
M1_constRNM <- calc_M1_constRNM(nsc, sc_Linf)
```

`calc_M2` *Calculate predation mortality*

Description

Calculates the predation mortality for each species in each length class.

Usage

```
calc_M2(N, ration, wgt, nfish, nsc, other, sc_Linf, suit_M2)
```

Arguments

`N`	A matrix with dimensions `nsc` and `nfish` representing the number of individuals in each length class for the current time step.
`ration`	A matrix with dimensions `nsc` and `nfish` representing the amount of food required for fish of a given species and length class to grow according to the von Bertalanffy growth curve in a time step.
`wgt`	A matrix with dimensions `nsc` and `nfish` representing the weight of each species in each length class.
`nfish`	A numeric value representing the number of species in the model.
`nsc`	A numeric value representing the number of length classes in the model.
`other`	A numeric value representing the amount of other food (g) available from prey that is not explicitly represented in the model.
`sc_Linf`	A numeric vector of length `nsc` representing the length class at which each species reaches its asymptotic length.
`suit_M2`	A list object of length `nfish`. Each element in the list is an array of dimensions `nsc`, `nsc` and `nfish` containing a value between zero and 1 representing prey preference and prey suitability for each species and length class.

Details

The predation mortality of the `i`th species in the `j`th length class is

```
sum_m(sum_n(I[j,i]*N[j,i]*suit_M2[[m]][n,j,i]/
(sum_k(sum_l(suit_M2[[m]][n,l,k]wgt[l,k]N[l,k]))+other)))
```

where `sum_m` represents the sum over all `m`, `sum_n` represents the sum over all `n`, `sum_l` represents the sum over all `l` and `sum_k` represents the sum over all `k`. This equation corresponds to a Holling type-II functional response. See equation 8 of Hall et al. (2006) for more details.

Value

A matrix with dimensions `nsc` and `nfish` representing the the predation mortality for each species in each length class.

References

Hall, S. J., Collie, J. S., Duplisea, D. E., Jennings, S., Bravington, M., & Link, J. (2006). A length-based multispecies model for evaluating community responses to fishing. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(6):1344-1359.

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                 phi_min=0.1) # fixed phi_min
phi <- tmp$phi
phi_min <- tmp$phi_min

# Calculate growth increments
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
ration <- tmp$ration
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt
g_eff <- tmp$g_eff

# Calculate predator-prey size preferences
prefs <- calc_prefs(pred_mu=-2.25, pred_sigma=0.5, wgt, sc_Linf)

# Calculate prey preference and prey suitability
suit_M2 <- calc_suit_vect(nsc, nfish, sc_Linf, prefs, NS_tau)


# Get an initial population
N0 <- get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept=1e10, slope=-5)

# Calculate the predation mortality
M2 <- calc_M2(N0, ration, wgt, nfish, nsc, other=1e12, sc_Linf, suit_M2)
```

`calc_mature` *Calculate the proportion of mature individuals*

Description

Calculates the proportion of individuals that are mature for each species and length class.

Usage

```
calc_mature(Lmat, nfish, mid, kappa, sc_Linf, eps = 1e-05, force_mature = TRUE)
```

Arguments

`Lmat`	A numeric vector of length `nfish` representing the length at which 50% of individuals are mature for each species.
`nfish`	A numeric value representing the number of species in the model.
`mid`	A numeric vector of length `nfish` representing the mid-point of the length classes in the model.
`kappa`	A numeric vector of length `nfish` representing the rate of change from immaturity to maturity for each species.
`sc_Linf`	A numeric vector of length `nsc` representing the length class at which each species reaches its asymptotic length.
`eps`	A numeric value specifying a numerical offset. The default is `1e-5`.
`force_mature`	A logical statement indicating whether to force all fish in the largest length class to be mature. The default is `TRUE`.

Details

The proportion of individuals in the `j`th length class of the `i`th species that are mature is described by a logistic model

```
1/(1+exp(-kappa[i]*(mid[j]-Lmat[i])))
```

Value

A matrix with dimensions `nsc` and `nfish` and elements in the range 0-1 representing the proportion of individuals that are mature for each species and length class.

References

Hall, S. J., Collie, J. S., Duplisea, D. E., Jennings, S., Bravington, M., & Link, J. (2006). A length-based multispecies model for evaluating community responses to fishing. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(6):1344-1359.

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                 phi_min=0.1) # fixed phi_min
phi <- tmp$phi
phi_min <- tmp$phi_min

# Calculate growth increments
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf

# Calculate the proportion of mature individuals
mature <- calc_mature(Lmat, nfish, mid, kappa=rep(10, nfish), sc_Linf)
```

`calc_phi` *Calculate the proportion of individuals that leave each length class for each species each length class each time step.*

Description

Calculate the proportion of individuals of each species that leave each length class at each time step.

Usage

```
calc_phi(
  k,
  Linf,
  nsc,
  nfish,
  u_bound,
  l_bound,
  calc_phi_min = FALSE,
  phi_min = 0.1
)
```

Arguments

k	A numeric vector of length nfish representing the von Bertalanffy growth parameter (1/yr) for each species.
Linf	A numeric vector of length nfish representing the asymptotic length of each species.
nsc	A numeric value representing the number of length classes in the model.
nfish	A numeric value representing the number of species in the model.
u_bound	A numeric vector of length nsc representing the upper bounds of the length classes.
l_bound	A numeric vector of length nsc representing the lower bounds of the length classes.
calc_phi_min	A logical statement indicating whether phi_min should be calculated within the function. The default is FALSE.
phi_min	A fixed numeric value of phi_min, which represents the time step of the model. This is only required if calc_phi_min=FALSE. The default is 0.1.

Details

Calculates the time (yrs) for an average fish to grow from the lower to the upper bound of a length class assuming von Bertalanffy growth. The values are scaled to the fastest growing fish and length class combination in order to calculate the proportion of individuals leaving each length class in a time step.

Value

A list object containing phi and phi_min. phi is a matrix of dimensions nsc and nfish representing the proportion of individuals of each species that leave each length class. phi_min is a numeric value representing the time step of the model.

References

Hilborn, R. & Walters, C.J. (1992). Quantitative Fisheries Stock Assessment. Springer.

von Bertalanffy, L. (1957). Quantitative Laws in Metabolism and Growth. *The Quarterly Review of Biology*, 32:217-231.

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
```

```
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Calculate the proportion of individuals that leave each length class
# with and without a fixed value for phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                  phi_min=0.1) # fixed phi_min
phi <- tmp$phi
phi_min <- tmp$phi_min

tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=TRUE) # without fixed phi_min
phi <- tmp$phi
phi_min <- tmp$phi_min
```

`calc_prefs` *Calculate predator-prey size preferences*

Description

Calculates the size preference of each predator species in each length class for each prey species in each length class.

Usage

```
calc_prefs(pred_mu, pred_sigma, wgt, sc_Linf)
```

Arguments

`pred_mu`	A numeric value representing the preferred predator-prey mass ratio.
`pred_sigma`	A numeric value representing the width of the weight preference function.
`wgt`	A matrix with dimensions `nsc` and `nfish` representing the weight of each species in each length class.
`sc_Linf`	A numeric vector of length `nsc` representing the length class at which each species reaches its asymptotic length.

Details

A predator of species `i` in length class `j` has a size preference for species `k` in length class `l` equal to

`exp(-(log10(wgt[l, k]/wgt[j, i])-pred_mu)^2/(2*pred_sigma))`.

Value

An array of dimensions `nsc`, `nfish`, `nsc` and `nfish`. The first and second dimensions represent the prey species whereas the third and fourth dimensions represent the predator species.

References

Hall, S. J., Collie, J. S., Duplisea, D. E., Jennings, S., Bravington, M., & Link, J. (2006). A length-based multispecies model for evaluating community responses to fishing. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(6):1344-1359.

Ursin, E. (1973). On the prey size preferences of cod and dab. *Meddelelser fra Danmarks Fiskeri-og Havundersgelser*, 7:85-98.

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                  phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Calculate growth increments
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt

# Calculate predator-prey size preferences
prefs <- calc_prefs(pred_mu=-2.25, pred_sigma=0.5, wgt, sc_Linf)
```

---

calc_Q *Calculate gear catchability*

Description

Calculates the catchability for each fishing gear.

Usage

```
calc_Q(
  curve = rep("logistic", nfish),
```

```
  species = ((0:(length(curve) - 1))%%nfish) + 1,
  max_catchability = rep(1, length(curve)),
  gear_name = paste("gear_", 1:length(curve), sep = ""),
  custom = NULL,
  nsc,
  nfish,
  l_bound,
  u_bound,
  mid,
  eps = 1e-05,
  species_names,
  ...
)

get_Q(curve, species, gear_name, nsc, nfish, l_bound, u_bound, mid, eps, ...)

logistic_catch(species, nsc, nfish, mid, eps, L50 = 50, eta = 0.25, ...)

log_gaussian_catch(species, nsc, nfish, mid, eps, Lmu = 50, Lsigma = 1, ...)

knife_edge_catch(species, nsc, nfish, l_bound, u_bound, Lmin = 50, ...)
```

Arguments

curve	A character vector of almost any length describing the type of curve to be used to determine the catchability of each species by the fishing gear. By default, curve is a character vector of length nfish that takes the value "logistic" in each element, but it can also take "log-gaussian" or "knife-edge". If a custom curve is required for a particular species and/or fishing gear, the curve must be specified in custom. See 'Details' for more information.
species	A numeric or character vector of the same length as curve describing the species that each element of curve relates to. By default, species is a numeric vector of length curve that takes the values (0:(length(curve)-1))%%(nfish)+1.
max_catchability	A numeric vector of length curve describing the maximum catchability for each catchability curve.
gear_name	A character vector of the same length as curve and species describing the fishing gear that each element of curve and species relates to. By default, gear_name is a character vector of length curve that takes the value paste("gear_", 1:length(curve), sep = "").
custom	An array of dimensions nsc, nfish and the number of custom catchability curves that are required. custom represents the catchability of each species by the gears specified using custom catchability curves. By default, custom is set to NULL.
nsc	A numeric value representing the number of length classes in the model.
nfish	A numeric value representing the number of fish species in the model.

l_bound	A numeric vector of length nsc representing the lower bounds of the length classes.
u_bound	A numeric vector of length nsc representing the upper bounds of the length classes.
mid	A numeric vector of length nfish representing the mid-point of the length classes.
eps	A numeric value specifying a numerical offset. The default is 1e-5.
species_names	A character vector of length nfish that is the name of the species in the same order as the species parameters.
...	Vectors of the same length as curve representing the parameters of the catchability curves. See 'Details' for more information.
L50	A numeric value representing the length at 50% of the maximum catchability of the catchability curve. This is used with the logistic_catch function. The default value is 50.
eta	A numeric value representing the steepness of the slope of the catchability curve. This is used with the logistic_catch function. The default value is 0.25
Lmu	A numeric value representing the length at the maximum catchability of the catchability curve. This is used with the log_gaussian_catch function. The default value is 50.
Lsigma	A numeric value representing the standard deviation of the catchability curve. See 'Details' for more information. This is used with the log_gaussian_catch function. The default value is 1.
Lmin	A numeric value representing the minimum length that is caught by the catchability curve. This is used with the knife_edge_catch function. The default value is 50.

Details

This function allows three different models for catchability, all of which are rescaled so that their maximum catchability is equal to one:

(1) "logistic" is proportional to

`1/(1+exp(-eta*(mid-L50)))`;

(2) "log-gaussian" is proportional to

`dlnorm(mid, log(Lmu), Lsigma)`;

and (3) "knife-edge" is equal to 1 for the length classes indexed by `max(which(l_bound<Lmin)):nsc` and 0 for all other length classes. This means that all of the individuals in the length class containing Lmin will have a catchability of 1.

In calc_Q the catchability is rescaled so that the maximum catchability is one.

Value

calc_Q returns an array of dimensions nsc, nfish and gear representing the catchability of each species by each of the fishing gears, scaled from 0 to max_catchability.

get_Q returns a catchability curve for a given species and gear scaled from 0 to 1. If an invalid catchability curve is selected, NULL is returned and a warning message is shown.

logistic_catch returns a matrix with dimensions nsc and nfish with all elements set to zero excluding one column that represents the logistic catchability curve for the selected species scaled from 0 to 1.

log_gaussian_catch returns a matrix with dimensions nsc and nfish with all elements set to zero excluding one column that represents the log-gaussian catchability curve for the selected species scaled from 0 to 1.

knife_edge_catch returns a matrix with dimensions nsc and nfish with all elements set to zero excluding one column that represents the knife-edge catchability curve for the selected species scaled from 0 to 1.

References

Hall, S. J., Collie, J. S., Duplisea, D. E., Jennings, S., Bravington, M., & Link, J. (2006). A length-based multispecies model for evaluating community responses to fishing. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(6):1344-1359.

Thorpe, R.B., Jennings, S., Dolder, P.J. (2017). Risks and benefits of catching pretty good yield in multispecies mixed fisheries. *ICES Journal of Marine Science*, 74(8):2097-2106.

See Also

dlnorm

Examples

```
# Set up the inputs to the function
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Calculate gear catchability
Qs <- calc_Q(curve=rep("logistic", nfish), species=NS_par$species_names,
             max_catchability=rep(1, nfish), gear_name=NS_par$species_names,
             nsc=nsc, nfish=nfish, mid=mid, l_bound=l_bound, u_bound=u_bound,
             species_names=NS_par$species_names, eta=rep(0.25, nfish), L50=Lmat)

# Calculate logistic catchability for the first species
logistic_catch(species=1, nsc, nfish, mid, eps=1e-5,L50=Lmat[1], eta=0.25)
```

```
# Calculate log-gaussian catchability for the first species
log_gaussian_catch(species=1, nsc, nfish, mid, eps=1e-5, Lmu=50, Lsigma=1)

# Calculate knife-edge catchability for the first species
knife_edge_catch(species=1, nsc, nfish, l_bound, u_bound, Lmin=50)
```

`calc_ration_growthfac` *Calculate growth increments*

Description

Calculates the amount of food required for fish of a given species and length class to grow according to the von Bertalanffy growth curve in a time step.

Usage

```
calc_ration_growthfac(
  k,
  Linf,
  nsc,
  nfish,
  l_bound,
  u_bound,
  mid,
  W_a,
  W_b,
  phi_min,
  vary_growth = TRUE,
  growth_eff = 0.5,
  growth_eff_decay = 0.11
)
```

Arguments

`k`	A numeric vector of length `nfish` representing the von Bertalanffy growth parameter (`1/yr`) for each species.
`Linf`	A numeric vector of length `nfish` representing the asymptotic length of each species.
`nsc`	A numeric value representing the number of length classes in the model.
`nfish`	A numeric value representing the number of species in the model.
`l_bound`	A numeric vector of length `nsc` representing the lower bounds of the length classes.
`u_bound`	A numeric vector of length `nsc` representing the upper bounds of the length classes.
`mid`	A numeric vector of length `nfish` representing the mid-point of the length classes.

W_a A numeric vector of length nfish representing the parameter a in the length-weight conversion. See 'Details' for more information.

W_b A numeric vector of length nfish representing the parameter b in the length-weight conversion. See 'Details' for more information.

phi_min A numeric value representing the time step of the model.

vary_growth A logical statement indicating whether growth efficiency should vary for each species (vary_growth=TRUE) or be fixed at the value given by fixed_growth (vary_growth=FALSE). The default is FALSE.

growth_eff If vary_growth==TRUE, growth_eff is a numeric representing the growth efficiencies of a fish of length 0. If vary_growth==FALSE, growth_eff is a numeric value of length 1 representing a fixed growth efficiency for all fish. The default is 0.5.

growth_eff_decay A numeric value specifying the rate at which growth efficiency decreases as length approaches Linf. The default is 0.11.

Details

The weight increments of the ith species in the jth length class is calculated by determining the amount an individual will grow in one time step, phi_min, if it were to follow the von Bertalanffy growth curve

L22=(Linf[i]-mid[j])*(1-exp(-k[i]*phi_min)).

The weight of a fish at the mid-point of the size class is calculated using the length-weight relationship

wgt[j,i] = a[i]*mid[j]^b[i],

and similarly the expected change in weight of the the fish is calculated as

growth_inc = (W_a[i]*L22^W_b[i]).

It also has a growth efficiency

g_eff[j, i]=(1-(wgt[j,i]/(W_a[i]*Linf[i]^W_b[i]))^growth_eff_decay)*growth_eff

if vary_growth==TRUE or g_eff[j, i]=growth_eff otherwise.

ration is then calculated by

growth_inc*(1/g_eff[j, i]).

Value

A list object containing ration, sc_Linf, wgt and g_eff. ration is a matrix with dimensions nsc and nfish representing the amount of food required for fish of a given species and length class to grow according to the von Bertalanffy growth curve in a time step. sc_Linf is a numeric vector of length nfish representing the length class at which each species reaches Linf. wgt is a matrix with dimensions nsc and nfish representing the weight of each species in each length class. g_eff is a matrix with dimensions nsc and nfish representing the growth efficiency of each species in each length class.

References

von Bertalanffy, L. (1957). Quantitative Laws in Metabolism and Growth. *The Quarterly Review of Biology*, 32:217-231

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                  phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Calculate growth increments
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
ration <- tmp$ration
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt
g_eff <- tmp$g_eff
```

`calc_recruits` *Calculate the number of recruits*

Description

Calculates the number of recruits of each species in a year.

Usage

```
calc_recruits(SSB, stored_rec_funs, rec_args)
```

Arguments

`SSB`	A numeric vector of length `nfish` representing the Spawning Stock Biomass (SSB) of each species.

stored_rec_funs
: A list object of length nfish that is used to store the generated stock recruitment functions.

rec_args
: A list object of length nfish that is used to store the parameters of the stock recruitment functions.

Value

A numeric vector of length nfish representing the number of new recruits of each species.

References

Barrowman, N.J., Myers, R.A. (2000). Still more spawner-recruit curves: the hockey stick and its generalisations. *Canadian Journal of Fisheries and Aquatic Science*, 57:665–676.

Beverton, R.J.H., Holt, S.J. (1957). On the Dynamics of Exploited Fish Populations, volume 19 of Fisheries Investigations (Series 2). United Kingdom Ministry of Agriculture and Fisheries.

Hall, S. J., Collie, J. S., Duplisea, D. E., Jennings, S., Bravington, M., & Link, J. (2006). A length-based multispecies model for evaluating community responses to fishing. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(6):1344-1359.

Ogle, D.H. (2016). Introductory Fisheries Analyses with R. CRC Press.

Ricker, W.E. (1954). Stock and recruitment. *Journal of the Fisheries Research Board of Canada*, 11:559-623.

Thorpe, R.B., Le Quesne, W.J.F., Luxford, F., Collie, J.S., Jennings, S. (2015). Evaluation and management implications of uncertainty in a multispecies size-structured model of population and community responses to fishing. *Methods in Ecology and Evolution*, 6:49-58.

See Also

get_rec_fun, make_rec_fun, rec_BH, rec_Ricker, rec_hockey, rec_const, rec_linear and calc_SSB

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
```

```
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                  phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Run calc_ration_growthfac()
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt

# Calculate maturity
mature <- calc_mature(Lmat, nfish, mid, kappa=rep(10, nfish), sc_Linf)

# Create recruitment functions
stored_rec_funs <- get_rec_fun(rep("hockey-stick", nfish))
recruit_params <- do.call("Map", c(c, list(a=NS_par$a, b=NS_par$b)))

# Get an initial population
N0 <- get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept=1e10, slope=-5)

# Calculate the SSB
SSB <- calc_SSB(mature, N0, wgt)

# Calculate the number of recruits
R <- calc_recruits(SSB, stored_rec_funs, recruit_params)
```

`calc_SSB` *Calculate Spawning Stock Biomass (SSB) or total biomass*

Description

Calculates the spawning stock biomass (SSB) or total biomass of each species in the model.

Usage

```
calc_SSB(wgt, mature, N)

calc_biomass(wgt, N)
```

Arguments

`wgt`	A matrix with dimensions `nsc` and `nfish` representing the weight of each species in each length class.
`mature`	A matrix with dimensions `ncs` and `nfish` with elements in the range 0-1 representing the proportion of mature individuals of each species in each length class.
`N`	A matrix with dimensions `nsc` and `nfish` representing the number of individuals in each length class for the current time step.

Details

The Spawning Stock Biomass is equal to `colSums(mature*N*wgt)` and the total biomass is equal to `colSums(mature*N*wgt)`.

Value

`calc_SSB` returns a numeric vector of length `nfish` representing the SSB of each species (g).

`calc_biomass` returns a numeric vector of length `length(species)` where the `j`th element is the biomass (g) of the `j`th species.

See Also

`calc_recruits`

Examples

```
# Set up and run the model
NS_params <- LeMansParam(NS_par, tau=NS_tau, eta=rep(0.25, 21), L50=NS_par$Lmat, other=1e12)
effort <- matrix(0.5, 10, dim(NS_params@Qs)[3])
model_run <- run_LeMans(NS_params, years=10, effort=effort)

# Calculate SSB
calc_SSB(wgt=NS_params@wgt, mature=NS_params@mature, N=model_run@N[,,101])

# Calculate biomass in the last time step
calc_biomass(wgt=NS_params@wgt, N=model_run@N[,,101])
```

`calc_suit_vect` *Combines prey preference and prey suitability*

Description

Calculates a combined value for prey preference and prey suitability standardised to a value between 0 and 1.

Usage

```
calc_suit_vect(nsc, nfish, sc_Linf, prefs, tau)
```

Arguments

`nsc`	A numeric value representing the number of length classes in the model.
`nfish`	A numeric value representing the number of species in the model.
`sc_Linf`	A numeric vector of length `nsc` representing the length class at which each species reaches its asymptotic length.

prefs	An array of dimensions nsc, nfish, nsc and nfish. The first and second dimensions represent the prey whereas the the third and fourth dimensions represent the predator.
tau	A matrix of dimensions nfish and nfish. Row indices represent predators and column indices represent prey. A value of 1 at location i, j indicates prey j is eaten by predator i.

Details

tau values are assigned to an array of dimensions nsc, nfish, nsc and nfish and multiplied by the array prefs. This creates an array of dimensions nsc, nfish, nsc and nfish indicating prey suitability. Prey suitability is then standardised to sum to 1 for each predator species in each length class.

Value

A list object of length nfish. Each element in the list is an array of dimensions nsc, nsc and nfish containing a value between 0 and 1 that represents prey preference and prey suitability for each species and length class.

References

Hall, S. J., Collie, J. S., Duplisea, D. E., Jennings, S., Bravington, M., & Link, J. (2006). A length-based multispecies model for evaluating community responses to fishing. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(6):1344-1359.

See Also

calc_M2.

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min
```

```
# Calculate growth increments
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt

# Calculate predator-prey size preferences
prefs <- calc_prefs(pred_mu=-2.25, pred_sigma=0.5, wgt, sc_Linf)

# Calculate prey preference and prey suitability
suit_M2 <- calc_suit_vect(nsc, nfish, sc_Linf, prefs, NS_tau)
```

`comb_LeMans_run` *Combine two* `LeMans_outputs` *objects*

Description

Combines two LeMans_outputs objects.

Usage

```
comb_LeMans_run(LeMans_run_x, LeMans_run_y, cont = TRUE)
```

Arguments

`LeMans_run_x`	A LeMans_outputs object.
`LeMans_run_y`	A LeMans_outputs object.
`cont`	A logical statement indicating whether or not `LeMans_run_y` is a continuation of `LeMans_run_x`. The default is `TRUE`.

Details

If `cont==T`, the first years output from `LeMans_run_y` is removed as this will be the same as the last year of `LeMans_run_x`.

Value

A `LeMans_outputs` object.

See Also

LeMans_outputs, `run_LeMans`

Examples

```
# Set up the inputs to the model
NS_params <- LeMansParam(NS_par, tau=NS_tau, eta=rep(0.25, 21), L50=NS_par$Lmat, other=NS_other)

# Define fishing effort
effort <- matrix(0.5, 10, dim(NS_params@Qs)[3])

# Run the model for the first time
model_run1 <- run_LeMans(NS_params, years=10, effort=effort)

# Run the model for another 12 years
effort1 <- matrix(0.5, 12, dim(NS_params@Qs)[3])
model_run2 <- run_LeMans(N=model_run1@N[,,101], NS_params, years=12, effort=effort1)

# Combine the two model runs into a single run
out <- comb_LeMans_run(model_run1, model_run2, cont=TRUE)
```

---

get_annual_catch *Get annual catch for each species, catch per unit effort or catch per gear*

Description

Get annual catch for each species, the Catch Per Unit Effort (CPUE) or the Catch Per Gear (CPG)

Usage

```
get_annual_catch(inputs, outputs, ...)

## S4 method for signature 'missing,missing'
get_annual_catch(
  Catch,
  years = (dim(Catch)[3] + (inc_first - 1)) * phi_min,
  phi_min = 0.1,
  inc_first = FALSE
)

## S4 method for signature 'LeMans_param,missing'
get_annual_catch(
  inputs,
  Catch,
  years = (dim(Catch)[3] + (inc_first - 1)) * inputs@phi_min,
  inc_first = FALSE
)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_annual_catch(
  inputs,
```

```
  outputs,
  years = (dim(outputs@Catch)[3] + (inc_first - 1)) * inputs@phi_min,
  inc_first = FALSE
)

## S4 method for signature 'missing,LeMans_outputs'
get_annual_catch(
  outputs,
  years = (dim(outputs@Catch)[3] + (inc_first - 1)) * phi_min,
  phi_min = 0.1,
  inc_first = FALSE
)

get_CPUE(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_CPUE(inputs, outputs, effort, years = nrow(effort), inc_first = FALSE)

## S4 method for signature 'LeMans_param,missing'
get_CPUE(inputs, Catch, effort, years = nrow(effort), inc_first = FALSE)

## S4 method for signature 'missing,LeMans_outputs'
get_CPUE(
  outputs,
  Qs,
  effort,
  years = nrow(effort),
  phi_min = 0.1,
  inc_first = FALSE
)

## S4 method for signature 'missing,missing'
get_CPUE(
  Catch,
  Qs,
  effort,
  years = nrow(effort),
  phi_min = 0.1,
  inc_first = FALSE
)

get_CPG(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_CPG(inputs, outputs, effort, years = nrow(effort), inc_first = FALSE)

## S4 method for signature 'LeMans_param,missing'
get_CPG(inputs, Catch, effort, years = nrow(effort), inc_first = FALSE)
```

```
## S4 method for signature 'missing,LeMans_outputs'
get_CPG(
  outputs,
  Qs,
  effort,
  years = nrow(effort),
  phi_min = 0.1,
  inc_first = FALSE
)

## S4 method for signature 'missing,missing'
get_CPG(
  Catch,
  Qs,
  effort,
  years = nrow(effort),
  phi_min = 0.1,
  inc_first = FALSE
)
```

Arguments

inputs	A LeMans_param object containing the parameter values of the current LeMans model.
outputs	A LeMans_outputs object containing the outputs of the model run.
...	Additional arguments.
Catch	An array with dimensions nsc, nfish and tot_time representing the number of individuals in each length class for each time step.
years	A numeric value representing the number of years included in Catch.
phi_min	A numeric value representing the time step of the model.
inc_first	A logical statement indicating whether the first time step of Catch should be included. The default is FALSE.
effort	A matrix with dimensions years and the number of fishing gears, representing fishing effort in each year for each gear.
Qs	An array of dimensions nsc, nfish and gear representing the catchability of each species by each of the fishing gears.

Value

get_annual_catch returns a matrix with dimensions years and length(species) where the i,jth element represents the total catch (g) of the jth species in the ith year.

get_CPUE returns a matrix with dimensions tot_time and nfish where the i,jth element represents the CPUE in the ith time step for the jth species.

get_CPG returns an array with dimensions nfish, dim(Qs[3]) and the number of time steps, where the i,j,kth element denotes the total catch of the ith species by the jth gear in the kth time step.

Examples

```
# Set up and run the model
NS_params <- LeMansParam(NS_par, tau=NS_tau, eta=rep(0.25, 21), L50=NS_par$Lmat, other=1e12)
effort <- matrix(0.5, 10, dim(NS_params@Qs)[3])
model_run <- run_LeMans(NS_params, years=10, effort=effort)

# Get annual catch
get_annual_catch(inputs=NS_params, outputs=model_run)

# Calculate the CPUE
get_CPUE(inputs=NS_params, outputs=model_run, effort=effort)

# Calculate Catch Per Gear (CPG)
get_CPG(inputs=NS_params, outputs=model_run, effort=effort)
```

---

get_indicators *Calculate indicators*

Description

Calculates Mean Maximum Length (MML), the Large Fish Indicator (LFI), Typical Length (TyL) and Length Quantiles (LQ) of the whole community or a subset of the species.

Usage

```
get_indicators(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_indicators(
  inputs,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  prob = 0.5,
  length_LFI = 40
)

## S4 method for signature 'LeMans_param,missing'
get_indicators(
  inputs,
  N,
  species = 1:dim(N)[2],
  time_steps = 1:dim(N)[3],
  prob = 0.5,
  length_LFI = 40
)

## S4 method for signature 'missing,LeMans_outputs'
```

```
get_indicators(
  wgt,
  mid,
  l_bound,
  u_bound,
  Linf,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  species_names = NULL,
  prob = 0.5,
  length_LFI = 40
)

## S4 method for signature 'missing,missing'
get_indicators(
  wgt,
  mid,
  l_bound,
  u_bound,
  Linf,
  N,
  species = 1:dim(N)[2],
  time_steps = 1:dim(N)[3],
  species_names = NULL,
  prob = 0.5,
  length_LFI = 40
)

get_LFI(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_LFI(
  inputs,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  length_LFI = 40
)

## S4 method for signature 'LeMans_param,missing'
get_LFI(
  inputs,
  N,
  species = 1:dim(N)[2],
  time_steps = 1:dim(N)[3],
  length_LFI = 40
)
```

```
## S4 method for signature 'missing,LeMans_outputs'
get_LFI(
  wgt,
  l_bound,
  u_bound,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  species_names = NULL,
  length_LFI = 40
)

## S4 method for signature 'missing,missing'
get_LFI(
  wgt,
  l_bound,
  u_bound,
  N,
  species = 1:dim(N)[2],
  time_steps = 1:dim(N)[3],
  species_names = NULL,
  length_LFI = 40
)

get_MML(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_MML(
  inputs,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3]
)

## S4 method for signature 'LeMans_param,missing'
get_MML(inputs, N, species = 1:dim(N)[2], time_steps = 1:dim(N)[3])

## S4 method for signature 'missing,LeMans_outputs'
get_MML(
  wgt,
  Linf,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  species_names = NULL
)
```

```
## S4 method for signature 'missing,missing'
get_MML(
  wgt,
  Linf,
  N,
  species = 1:dim(N)[2],
  time_steps = 1:dim(N)[3],
  species_names = NULL
)

get_TyL(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_TyL(
  inputs,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3]
)

## S4 method for signature 'LeMans_param,missing'
get_TyL(inputs, N, species = 1:dim(N)[2], time_steps = 1:dim(N)[3])

## S4 method for signature 'missing,LeMans_outputs'
get_TyL(
  wgt,
  mid,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  species_names = NULL
)

## S4 method for signature 'missing,missing'
get_TyL(
  wgt,
  mid,
  N,
  species = 1:dim(N)[2],
  time_steps = 1:dim(N)[3],
  species_names = NULL
)

get_LQ(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_LQ(
  inputs,
```

```
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  prob = 0.5
)

## S4 method for signature 'LeMans_param,missing'
get_LQ(inputs, N, species = 1:dim(N)[2], time_steps = 1:dim(N)[3], prob = 0.5)

## S4 method for signature 'missing,LeMans_outputs'
get_LQ(
  wgt,
  u_bound,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  species_names = NULL,
  prob = 0.5
)

## S4 method for signature 'missing,missing'
get_LQ(
  wgt,
  u_bound,
  N,
  species = 1:dim(N)[2],
  time_steps = 1:dim(N)[3],
  species_names = NULL,
  prob = 0.5
)
```

Arguments

inputs	A LeMans_param object containing the parameter values of the current LeMans model.
outputs	A LeMans_outputs object containing the outputs of the model run.
...	Additional arguments.
species	A numeric value or vector or a character string representing the species that you wish to use to calculate the indicators. The default is 1:dim(N)[2].
time_steps	A numeric vector of the time steps that you wish to use to calculate the indicators. The default is 1:dim(N)[3].
prob	A numeric value or vector between 0 and 1 denoting the length quantiles to be calculated. The default is 0.5.
length_LFI	A numeric vector representing the thresholds to be used to calculate the LFI. The default value is 40.
N	An array with dimensions nsc, nfish and tot_time representing the number of individuals in each length class for each time step.

wgt	A matrix with dimensions nsc and nfish representing the weight of each species in each length class.
mid	A numeric vector of length nfish representing the mid-point of the length classes in the model.
l_bound	A numeric vector of length nsc representing the lower bounds of the length classes.
u_bound	A numeric vector of length nsc representing the upper bounds of the length classes.
Linf	A numeric vector of length nfish representing the asymptotic length of each species.
species_names	A character vector of length nfish that denotes the names of the species in the model.

Details

The LFI represents the proportion of biomass with a length larger than length_LFI. The MML is the biomass weighted mean of Linf:

sum(biomass[species]*Linf[species])/sum(biomass[species])

where biomass is a numeric vector of length nfish representing the biomass of each species. TyL is the biomass-weighted geometric mean length of the community:

exp(sum(biomass_*log(mid))/sum(Bio_l))

where biomass_ is a numeric vector of length nsc representing the biomass of all the species in each length class. The LQ is the length at which the biomass exceeds a given proportion prob of the total biomass.

Value

get_indicators returns a list object with names 'LFI', 'MML', 'TYL' and 'LQ'. If length(length_LFI)>1, 'LFI' is a matrix with dimensions length(time_steps) by length(length_LFI) where the i,jth element represents the LFI using the jth length_LFI in the ith time_steps. If length(length_LFI)==1, the function will return a numeric vector of length length(time_steps). 'MML' is a numeric vector of length time_steps where each element is the MML for the species in species. 'TYL' is a numeric vector of length time_steps where each element is the TyL for the species in species. If length(prob)==1, 'LQ' is a matrix with dimensions length(time_steps) by length(prob) where the i,jth element is the LQ using thejth prob in the ith time_steps. If length(prob)==1, the function will return a numeric vector of length length(time_steps).

If length(length_LFI)==1, get_LFI returns a matrix with dimensions length(time_steps) by length(length_LFI) where the i,jth element is the LFI using the jth length_LFI in the ith time_steps. If length(length_LFI)==1, the function will return a numeric vector of length length(time_steps).

get_MML returns a numeric vector of length time_steps where each element is the MML for the species in species.

If length(prob)>1, get_LQ returns a matrix with dimensions length(time_steps) and length(prob) where the i,jth element is the LQ using the the jth prob in the ith time_steps. If length(prob)==1, the function will return a numeric vector of length length(time_steps).

See Also

plot_indicators

Examples

```
# Set up and run the model
NS_params <- LeMansParam(NS_par, tau=NS_tau, eta=rep(0.25, 21), L50=NS_par$Lmat, other=1e12)
effort <- matrix(0.5, 10, dim(NS_params@Qs)[3])
model_run <- run_LeMans(NS_params, years=10, effort=effort)

# Calculate the indicators
get_indicators(inputs=NS_params, outputs=model_run)

# Calculate the LFI
get_LFI(inputs=NS_params, outputs=model_run)

# Calculate MML
get_MML(inputs=NS_params, outputs=model_run)

# Calculate TyL
get_TyL(inputs=NS_params, outputs=model_run)

# Calculate LQs
get_LQ(inputs=NS_params, outputs=model_run)
```

---

get_N0 *Generate a starting value for N*

Description

Generate a starting value for N, which represents the number of individuals in each length class for each species.

Usage

```
get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept = 1e+10, slope = -5)
```

Arguments

nsc	A numeric value representing the number of length classes in the model.
nfish	A numeric value representing the number of fish species in the model.
mid	A numeric vector of length nfish representing the mid-point of the length classes.
wgt	A matrix with dimensions nsc and nfish representing the weight of each species in each length class.
sc_Linf	A numeric vector of length nsc representing the length class at which each species reaches its asymptotic length.

intercept	A numeric value representing the number of individuals in the first length class. The default is 1e10.
slope	A numeric value representing the slope of the community size spectrum. The default is -5.

Details

The total number of individuals in the community in each length class is equal to intercept*mid^slope. Within each length class, the number of individuals of each species is determined using the proportion of each species' biomass that is found in that length class.

Value

A matrix with dimensions nsc and nfish representing the number of individuals in each length class.

References

Andersen, K.H., Blanchard, J.L., Fulton, E.A., Gislason, H., Jacobsen, N.S., van Kooten, T. (2016). Assumptions behind size-based ecosystem models are realistic. *ICES Journal of Marine Science*, 73(6):1651-1655.

See Also

run_LeMans

Examples

```
# Set up the inputs to the function
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                 phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Calculate growth increments
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
```

```
wgt <- tmp$wgt

# Get an initial population
get_N0(nsc, nfish, mid, wgt, sc_Linf)
```

`get_rec_fun` *Collate the stock recruitment functions*

Description

Collates the stock recruitment functions for all of the species in the model.

Usage

```
get_rec_fun(rec_fun = "hockey-stick")
```

Arguments

`rec_fun`	A character vector representing the stock recruitment function to be applied to each species. The default is `"hockey-stick"`, but `rec_fun` can also take `"Ricker"`, `"Beverton-Holt"`, `"constant"`, or `"linear"` for each species.

Details

For `"Beverton-Holt"`, the stock recruitment function is defined as `a*SSB/(1+b*SSB)`; for `"Ricker"` it is defined as `a*SSB*exp(-b*SSB)`; for `"hockey-stick"` it is defined as `min(a*SSB, b)`; for `"constant"` it is defined as `a`, and for `"linear"` it is defined as `a*SSB`. In all cases, `SSB` is the Spawning Stock Biomass in 1000s of tonnes and `a` and `b` are parameters of the specific stock recruitment functions.

Value

A list object of length `rec_fun` where each element includes the stock recruitment function for a given species. If an invalid recruitment function is selected, `NULL` is returned and a warning message is shown.

References

Barrowman, N.J., Myers, R.A. (2000). Still more spawner-recruit curves: the hockey stick and its generalisations. *Canadian Journal of Fisheries and Aquatic Science*, 57:665–676.

Beverton, R.J.H., Holt, S.J. (1957). On the Dynamics of Exploited Fish Populations, volume 19 of Fisheries Investigations (Series 2). United Kingdom Ministry of Agriculture and Fisheries.

Hall, S. J., Collie, J. S., Duplisea, D. E., Jennings, S., Bravington, M., & Link, J. (2006). A length-based multispecies model for evaluating community responses to fishing. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(6):1344-1359.

Ogle, D.H. (2016). Introductory Fisheries Analyses with R. CRC Press.

Ricker, W.E. (1954). Stock and recruitment. *Journal of the Fisheries Research Board of Canada*, 11:559-623.

Thorpe, R.B., Le Quesne, W.J.F., Luxford, F., Collie, J.S., Jennings, S. (2015). Evaluation and management implications of uncertainty in a multispecies size-structured model of population and community responses to fishing. *Methods in Ecology and Evolution*, 6:49-58.

See Also

calc_recruits, make_rec_fun, rec_BH, rec_Ricker, rec_hockey, rec_const, rec_linear and calc_SSB

Examples

```
nfish <- nrow(NS_par)
stored_rec_funs <- get_rec_fun(rep("hockey-stick", nfish))
```

---

get_SSB *Calculate Spawning Stock Biomass (SSB) or total biomass*

Description

Calculates the Spawning Stock Biomass (SSB) or the total biomass of each species in the model.

Usage

```
get_SSB(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_SSB(
  inputs,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3]
)

## S4 method for signature 'LeMans_param,missing'
get_SSB(inputs, N, species = 1:dim(N)[2], time_steps = 1:dim(N)[3])

## S4 method for signature 'missing,LeMans_outputs'
get_SSB(
  wgt,
  mature,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  species_names = NULL
)
```

```
## S4 method for signature 'missing,missing'
get_SSB(
  wgt,
  mature,
  N,
  species = 1:dim(N)[2],
  time_steps = 1:dim(N)[3],
  species_names = NULL
)

get_biomass(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
get_biomass(
  inputs,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3]
)

## S4 method for signature 'LeMans_param,missing'
get_biomass(inputs, N, species = 1:dim(N)[2], time_steps = 1:dim(N)[3])

## S4 method for signature 'missing,LeMans_outputs'
get_biomass(
  wgt,
  outputs,
  species = 1:dim(outputs@N)[2],
  time_steps = 1:dim(outputs@N)[3],
  species_names = NULL
)

## S4 method for signature 'missing,missing'
get_biomass(
  wgt,
  N,
  species = 1:dim(N)[2],
  time_steps = 1:dim(N)[3],
  species_names = NULL
)
```

Arguments

inputs	A LeMans_param object containing the parameter values of the current LeMans model.
outputs	A LeMans_outputs object containing the outputs of the model run.
...	Additional arguments.

species	A numeric value or vector or a character string or vector of the species that you wish to calculate the mean maximum length. The default is 1:dim(N)[2].
time_steps	A numeric vector of the time steps that you wish to calculate the mean maximum length. The default is 1:dim(N)[3].
N	An array with dimensions nsc, nfish and tot_time representing the number of individuals in each length class for each time step.
wgt	A matrix with dimensions nsc and nfish representing the weight of each species in each length class.
mature	A matrix with dimensions ncs and nfish with elements in the range 0-1 representing the proportion of mature individuals of each species in each length class.
species_names	A character vector of length nfish that denotes the names of the species in the model.

Details

The SSB for each species in species is equal to:

colSums(N[, species]*wgt[, species]*mature[, species]).

The biomass for each species in species is equal to:

colSums(N[, species]*wgt[, species])

Value

If length(species)>1, get_SSB returns a matrix with dimensions length(time_steps) by length(species) where the i, jth element is the SSB (g) of the jth species in the ith time_steps. If length(species)==1, the function will return a numeric vector of length length(time_steps).

If length(species)>1, get_biomass returns a matrix with dimensions length(time_steps) by length(species) where the i, jth element is the biomass (g) of the jth species in the ith time_steps. If length(species)==1, the function will return a numeric vector of length length(time_steps).

Examples

```
# Set up and run the model
NS_params <- LeMansParam(NS_par, tau=NS_tau, eta=rep(0.25, 21), L50=NS_par$Lmat, other=1e12)
effort <- matrix(0.5, 10, dim(NS_params@Qs)[3])
model_run <- run_LeMans(NS_params, years=10, effort=effort)

# Calculate SSB
get_SSB(inputs=NS_params, outputs=model_run)

# Calculate biomass
get_biomass(inputs=NS_params, outputs=model_run)
```

LeMansParam *A constructor for the* LeMansParam *class*

Description

A constructor for the LeMansParam class.

Usage

```
LeMansParam(df, gdf, ...)

## S4 method for signature 'ANY,ANY'
LeMansParam(
  df,
  gdf,
  nfish = nrow(df),
  nsc = 32,
  pred_mu = -2.25,
  pred_sigma = 0.5,
  other = 1e+12,
  bounds = NULL,
  calc_phi_min = FALSE,
  phi_min = 0.1,
  vary_growth = TRUE,
  growth_eff = 0.5,
  growth_eff_decay = 0.11,
  eps = 1e-05,
  force_mature = TRUE,
  species_names = paste("species", 1:nfish, sep = "_"),
  kappa = rep(10, nfish),
  tau = matrix(1, nfish, nfish),
  rec_fun = rep("hockey-stick", nfish),
  recruit_params = list(a = 18.835 - 4.133 * df$Linf, b = rep(1000/nfish, nfish)),
  natmort_opt = rep("std_RNM", nfish),
  Nmort = rep(0.8, nfish),
  prop = rep(3/4, nfish),
  curve = rep("logistic", nfish),
  catch_species = ((0:(length(curve) - 1))%%nfish) + 1,
  max_catchability = rep(1, length(curve)),
  gear_name = paste("gear_", 1:length(curve), sep = ""),
  custom = NULL,
  ...
)

## S4 method for signature 'missing,ANY'
LeMansParam(
  gdf,
```

```
  nfish = length(Linf),
  nsc = 32,
  pred_mu = -2.25,
  pred_sigma = 0.5,
  other = 1e+12,
  bounds = NULL,
  calc_phi_min = FALSE,
  phi_min = 0.1,
  vary_growth = FALSE,
  growth_eff = 0.5,
  growth_eff_decay = 0.11,
  eps = 1e-05,
  force_mature = TRUE,
  Linf,
  k,
  W_a,
  W_b,
  Lmat,
  species_names = paste("species", 1:nfish, sep = "_"),
  kappa = rep(10, nfish),
  tau = matrix(1, nfish, nfish),
  rec_fun = rep("hockey-stick", nfish),
  recruit_params = list(a = 18.835 - 4.133 * Linf, b = rep(1000/nfish, nfish)),
  natmort_opt = rep("std_RNM", nfish),
  Nmort = rep(0.8, nfish),
  prop = rep(3/4, nfish),
  curve = rep("logistic", nfish),
  catch_species = ((0:(length(curve) - 1))%%nfish) + 1,
  max_catchability = rep(1, length(curve)),
  gear_name = paste("gear_", 1:length(curve), sep = ""),
  custom = NULL,
  ...
)

## S4 method for signature 'missing,missing'
LeManSParam(
  df,
  gdf,
  nfish = length(Linf),
  nsc = 32,
  pred_mu = -2.25,
  pred_sigma = 0.5,
  other = 1e+12,
  bounds = NULL,
  calc_phi_min = TRUE,
  phi_min = 0.1,
  vary_growth = FALSE,
  growth_eff = 0.5,
```

```
  growth_eff_decay = 0.11,
  eps = 1e-05,
  force_mature = TRUE,
  species_names = paste("species", 1:nfish, sep = "_"),
  Linf,
  k,
  W_a,
  W_b,
  Lmat,
  kappa = rep(10, nfish),
  tau = matrix(1, nfish, nfish),
  rec_fun = rep("hockey-stick", nfish),
  recruit_params = list(a = 18.835 - 4.133 * Linf, b = rep(1000/nfish, nfish)),
  natmort_opt = rep("std_RNM", nfish),
  Nmort = rep(0.8, nfish),
  prop = rep(3/4, nfish),
  curve = rep("logistic", nfish),
  catch_species = ((0:(length(curve) - 1))%%nfish) + 1,
  max_catchability = rep(1, length(curve)),
  gear_name = paste("gear_", 1:length(curve), sep = ""),
  custom = NULL,
  ...
)
```

Arguments

df	A data frame with nfish rows and the following columns: Linf, W_a, W_b, k and Lmat. See below for definitions of each of these parameters.
gdf	A data frame with nfish rows and the following columns: curve, catch_species, max_catchability, k and gear_name. See below for definitions of each of these parameters.
...	Additional arguments for calculating catchability curves. See calc_Q for more details.
nfish	A numeric value representing the number of fish species in the model.
nsc	A numeric value representing the number of length classes in the model.
pred_mu	A numeric value representing the preferred predator-prey mass ratio.
pred_sigma	A numeric value representing the width of the weight preference function.
other	A numeric value representing the amount of other food (g) available from prey that is not explicitly represented in the model. The default is 1e12.
bounds	An optional argument specifying the bounds of the length classes.
calc_phi_min	A logical statement indicating whether phi_min should be calculated within the function. The default is FALSE.
phi_min	A fixed numeric value of phi_min, which represents the time step of the model. This parameter is only required if calc_phi_min=FALSE. The default is 0.1.

vary_growth A logical statement indicating whether growth efficiency should vary for each species (vary_growth=TRUE) or be fixed at the value given by fixed_growth (vary_growth=FALSE). The default is TRUE.

growth_eff If vary_growth==TRUE, growth_eff is a numeric representing the growth efficiencies of a fish of length 0. If vary_growth==FALSE, growth_eff is a numeric value of length 1 representing a fixed growth efficiency for all fish. The default is 0.5.

growth_eff_decay A numeric value specifying the rate at which growth efficiency decreases as length increases to Linf. The default is 0.11.

eps A numeric value specifying a numerical offset. The default value is 1e-5.

force_mature A logical statement indicating whether to force maturity for all fish in the largest length class. The default is TRUE.

species_names A numeric or character vector of length nfish that denotes the names of the species in the model.

kappa A numeric vector of length nfish representing the rate of change from immature to mature fish.

tau A matrix with dimensions nfish and nsc. Row indices represent predators and column indices represent prey. A value of 1 at location i, j indicates prey j is eaten by predator i.

rec_fun A character vector representing the stock recruitment function to be applied to each species. The default value is "hockey-stick" but rec_fun can take a value of "Ricker", "Beverton-Holt", "constant", or "linear" for each species.

recruit_params A list object of length nfish specifying the parameters for the recruitment function.

natmort_opt A character vector of length 1 describing the mortality function to be used for the species. The default value is "std_RNM" but can take a value of "constant" or "linear". See calc_M1 for more information.

Nmort A numeric vector of length nfish representing the maximum background mortality of each species. The default is 0.8 for all species.

prop A numeric vector of length nfish representing the proportion of length classes that have a non-zero background mortality. This is required only when the natmort_opt mortality function is used. The default is 3/4.

curve A character vector of almost any length describing the type of curve to be used to determine the catchability of each species by fishing gear. By default, curve is a character vector of length nfish that takes the value "logistic" in each element, but it can also take a value of "log-gaussian" or "knife-edge". If a custom curve is required for a particular species and/or fishing gear, the curve must be specified in custom.

catch_species A numeric value or character string describing the species to apply the catchability curve to.

max_catchability A numeric vector of length curve describing the maximum catchability for each catchability curve.

gear_name	A character vector of the same length as curve and species describing the fishing gear that each element of curve and species relates to. By default, gear_name is a character vector of length curve that takes the value paste("gear_", 1:length(curve), sep = "").
custom	An array with dimensions nsc, nfish and the number of custom catchability curves that are required. custom represents the catchability of each species by the gears specified using custom catchability curves. By default, custom is set to NULL.
Linf	A numeric vector of length nfish representing the mid-point of the length classes.
k	k A numeric vector of length nfish representing the von Bertalanffy growth parameter (1/yr) for each species.
W_a	A numeric vector representing the parameter a in the length-weight conversion.
W_b	A numeric vector representing the parameter b in the length-weight conversion.
Lmat	A numeric vector of length nsc representing the length at which 50% of the individuals are mature.

Details

Converts objects of class data frame or vector to class LeMansParams for use in the LeMans model. Linf, W_a, W_b, k and Lmat are required as either a data frame or as vectors.

Value

An object of class LeMansParam for use in the LeMans model.

See Also

run_LeMans, LeMans_param

Examples

```
# To run the model with all inputs specified explicitly:
# Set up species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50% of individuals are mature (cm).

NS_params <- LeMansParam(species_names=NS_par$species_names, Linf=Linf, k=k, W_a=W_a, W_b=W_b,
Lmat=Lmat, tau=NS_tau, recruit_params=list(a=NS_par$a, b=NS_par$b), eta=rep(0.25, 21), L50=Lmat,
other=NS_other)

##############################################
# Alternatively:
NS_params <- LeMansParam(NS_par, tau=NS_tau, eta=rep(0.25, 21), L50=NS_par$Lmat, other=NS_other)
```

LeMans_outputs-class *An S4 class representing the outputs of the LeMans model*

Description

An S4 class representing the outputs of the LeMans model

Slots

`N` An array with dimensions `nsc`, `nfish` and `tot_time` representing the number of individuals in each length class at each time step of the model.

`Catch` An array with dimensions `nsc`, `nfish` and `tot_time` representing the biomass of each species that is caught in each length class at each time step of the model.

`M2` An array with dimensions `nsc`, `nfish` and `tot_time` representing the predation mortality of each species in each length class at each time step of the model.

`R` A matrix with dimensions `tot_time` and `nfish` representing the number of recruits of each species at each time step of the model.

LeMans_param-class *An S4 class representing the inputs of the LeMans model*

Description

An S4 class representing the inputs of the LeMans model

Details

Sets the class of a LeMansParam object, which is then used to run the LeMans model.

Slots

`nsc` A numeric value representing the number of length classes in the model.

`nfish` A numeric value representing the number of fish species in the model.

`phi_min` A numeric value representing the time step of the model.

`l_bound` A numeric vector of length `nsc` representing the lower bounds of the length classes.

`u_bound` A numeric vector of length `nsc` representing the upper bounds of the length classes.

`mid` A numeric vector of length `nfish` representing the mid-point of the length classes.

`species_names` A numeric or character vector of length `nfish` that denotes the names of the species in the model.

`Linf` A numeric vector of length `nfish` representing the mid-point of the length classes.

`W_a` A numeric vector representing the parameter `a` in the length-weight conversion.

`W_b` A numeric vector representing the parameter `b` in the length-weight conversion.

k k A numeric vector of length nfish representing the von Bertalanffy growth parameter (1/yr) for each species.

Lmat A numeric vector of length nsc representing the length at which 50% of the individuals are mature.

mature A matrix with dimensions nsc and nfish with elements in the range 0-1 representing the proportion of mature individuals of each species in each length class.

sc_Linf A numeric vector of length nsc representing the length class at which each species reaches its asymptotic length.

wgt A matrix with dimensions nsc and nfish representing the weight of each species in each length class.

phi A matrix with dimensions nsc and nfish representing the proportion of individuals that leave each length class.

ration A matrix with dimensions nsc and nfish representing the amount of food required for fish of a given species and length class to grow according to the von Bertalanffy growth curve in a time step.

other A numeric value representing the amount of other food (g) available from prey that is not explicitly represented in the model.

M1 A matrix of dimensions nsc and nfish representing the natural mortality of each species for each length class.

suit_M2 A list object of length nfish. Each element in the list is an array of dimensions nsc, nsc and nfish containing a value between 0 and 1 representing prey preference and prey suitability for each species and length class.

Qs An array of dimensions nsc, nfish and length(gear) representing the catchability of each species by each of the fishing gears.

stored_rec_funs A list object of length rec_fun where each element includes the stock recruitment function for a given species. If an invalid recruitment function is selected, NULL is returned and a warning message is shown.

eps A numeric value specifying a numerical offset. The default value is 1e-5.

recruit_params A list object of length nfish specifying the parameters for the recruitment function.

---

make_rec_fun *Generate the stock recruitment functions*

Description

Generates the stock recruitment function for a given species.

Usage

```
make_rec_fun(rec_fun = "hockey-stick")
```

Arguments

rec_fun	A character vector representing the stock recruitment function to be applied to each species. The default is "hockey-stick", but rec_fun can also take "Ricker", "Beverton-Holt", "constant", or "linear" for each species.

Details

For "Beverton-Holt", the stock recruitment function is defined as a*SSB/(1+b*SSB); for "Ricker" it is defined as a*SSB*exp(-b*SSB); for "hockey-stick" it is defined as min(a*SSB, b); for "constant" it is defined as a, and for "linear" it is defined as a*SSB. In all cases, SSB is the Spawning Stock Biomass in 1000s of tonnes and a and b are parameters of the specific stock recruitment functions.

Value

The stock recruitment function for a given species. If an invalid recruitment function is selected, NULL is returned and a warning message is shown.

References

Barrowman, N.J., Myers, R.A. (2000). Still more spawner-recruit curves: the hockey stick and its generalisations. *Canadian Journal of Fisheries and Aquatic Science*, 57:665–676.

Beverton, R.J.H., Holt, S.J. (1957). On the Dynamics of Exploited Fish Populations, volume 19 of Fisheries Investigations (Series 2). United Kingdom Ministry of Agriculture and Fisheries.

Hall, S. J., Collie, J. S., Duplisea, D. E., Jennings, S., Bravington, M., & Link, J. (2006). A length-based multispecies model for evaluating community responses to fishing. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(6):1344-1359.

Ogle, D.H. (2016). Introductory Fisheries Analyses with R. CRC Press.

Ricker, W.E. (1954). Stock and recruitment. *Journal of the Fisheries Research Board of Canada*, 11:559-623.

Thorpe, R.B., Le Quesne, W.J.F., Luxford, F., Collie, J.S., Jennings, S. (2015). Evaluation and management implications of uncertainty in a multispecies size-structured model of population and community responses to fishing. *Methods in Ecology and Evolution*, 6:49-58.

See Also

calc_recruits, get_rec_fun, rec_BH, rec_Ricker, rec_hockey, rec_const, rec_linear and calc_SSB

Examples

```
# Run the function
make_rec_fun("Ricker")
```

NS_eta — *The steepness of the slope of the catchability curve*

Description

The steepness of the slope of the catchability curve associated with NS_mixed_fish.

Usage

```
NS_eta
```

Format

A numeric value representing the steepness of the slope of the catchability curve.

References

Thorpe, R.B., Dolder, P.J. , Reeves, S., Robinson, P., Jennings, S. (2015). Assessing fishery and ecological consequences of alternative management options for multispecies fisheries *ICES Journal of Marine Science*, 73(6):1503-1512.

NS_L50 — *The length at 50% of the maximum catchability of the catchability curve*

Description

The length at 50% of the maximum catchability of the catchability curve associated with NS_mixed_fish.

Usage

```
NS_L50
```

Format

A numeric value representing the length at 50% of the maximum catchability of the catchability curve.

References

Thorpe, R.B., Dolder, P.J. , Reeves, S., Robinson, P., Jennings, S. (2015). Assessing fishery and ecological consequences of alternative management options for multispecies fisheries *ICES Journal of Marine Science*, 73(6):1503-1512.

NS_mixed_fish *Gear selectivity data frame*

Description

A gear selectivity data frame for the 21 species in NS_par.

Usage

```
NS_mixed_fish
```

Format

A data frame with 21 rows and 4 variables, including:

`catch_species` A character string describing the species to apply the catchability curve to.

`curve` A character vector describing the type of curve to be used to determine the catchability of each species by the fishing gear.

`gear_name` A character string describing the name of the gear.

`max_catchability` A numeric vector describing the maximum catchability for each catchability curve.

References

Thorpe, R.B., Dolder, P.J. , Reeves, S., Robinson, P., Jennings, S. (2015). Assessing fishery and ecological consequences of alternative management options for multispecies fisheries *ICES Journal of Marine Science*, 73(6):1503-1512.

NS_other *Other food for the North Sea*

Description

Other food for the North Sea dataset, NS_par.

Usage

```
NS_other
```

Format

A numerical value representing other food. To be used with NS_par.

NS_par *North Sea data*

Description

Data for the 21 species in the North Sea version of the LeMans model.

Usage

```
NS_par
```

Format

A data frame with 21 rows and 8 variables, including:

`species_names` A numeric or character vector of length `nfish` that denotes the names of the species in the model.

`Linf` A numeric vector of length `nfish` representing the asymptotic length of each species (cm).

`W_a` A numeric vector of length `nfish` representing the parameter `a` in the length-weight conversion.

`W_b` A numeric vector of length `nfish` representing the parameter `b` in the length-weight conversion.

`k` A numeric vector of length `nfish` representing the von Bertalanffy growth parameter `(1/yr)` for each species.

`Lmat` A numeric vector of length `nsc` representing the length at which 50% of the individuals are mature (cm).

`a` A numeric value representing the density independent part of the hockey-stick recruitment curve.

`b` A numeric value representing the density dependent part of the hockey-stick recruitment curve.

References

Thorpe, R.B., Le Quesne, W.J.F., Luxford, F., Collie, J.S., Jennings, S. (2015). Evaluation and management implications of uncertainty in a multispecies size-structured model of population and community responses to fishing. *Methods in Ecology and Evolution*, 6:49-58.

NS_tau *North Sea interaction matrix*

Description

A predator-prey interaction matrix for the 21 species in `NS_par`.

Usage

```
NS_tau
```

Format

A matrix with 21 rows and 21 columns:

Row indices represent predators and column indices represent prey. A value of 1 at location `i`, `j` indicates prey `j` is eaten by predator `i`.

References

Thorpe, R.B., Le Quesne, W.J.F., Luxford, F., Collie, J.S., Jennings, S. (2015). Evaluation and management implications of uncertainty in a multispecies size-structured model of population and community responses to fishing. *Methods in Ecology and Evolution*, 6:49-58.

plot_indicators *Plot indicators*

Description

Plots Mean Maximum Length (MML), the Large Fish Indicator (LFI), Typical Length (TyL) and the Length Quantiles (LQ) of the whole community or a subset of the species.

Usage

```
plot_indicators(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
plot_indicators(inputs, outputs, species, time_steps, prob, length_LFI, ...)

## S4 method for signature 'LeMans_param,missing'
plot_indicators(inputs, N, species, time_steps, prob, length_LFI, ...)

## S4 method for signature 'missing,LeMans_outputs'
plot_indicators(
  wgt,
  mid,
```

```
  l_bound,
  u_bound,
  Linf,
  species,
  outputs,
  time_steps,
  species_names = NULL,
  prob,
  length_LFI,
  ...
)

## S4 method for signature 'missing,missing'
plot_indicators(
  wgt,
  mid,
  l_bound,
  u_bound,
  Linf,
  N,
  species,
  time_steps,
  species_names = NULL,
  prob,
  length_LFI,
  MML,
  TyL,
  LFI,
  LQ,
  units = "cm",
  ...
)

plot_LFI(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
plot_LFI(
  inputs,
  outputs,
  species,
  time_steps,
  species_names,
  length_LFI,
  LFI,
  ...
)

## S4 method for signature 'LeMans_param,missing'
```

```
plot_LFI(inputs, N, species, time_steps, species_names, length_LFI, LFI, ...)

## S4 method for signature 'missing,LeMans_outputs'
plot_LFI(
  wgt,
  l_bound,
  u_bound,
  outputs,
  species,
  time_steps,
  species_names,
  length_LFI,
  LFI,
  ...
)

## S4 method for signature 'missing,missing'
plot_LFI(
  wgt,
  l_bound,
  u_bound,
  N,
  species,
  time_steps,
  species_names,
  length_LFI,
  LFI,
  units = "cm",
  ...
)

plot_MML(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
plot_MML(inputs, outputs, species, species_names, time_steps, MML, ...)

## S4 method for signature 'LeMans_param,missing'
plot_MML(inputs, N, species, time_steps, species_names, MML, ...)

## S4 method for signature 'missing,LeMans_outputs'
plot_MML(wgt, Linf, outputs, species, time_steps, species_names, MML, ...)

## S4 method for signature 'missing,missing'
plot_MML(
  wgt,
  Linf,
  N,
  species,
```

```
  time_steps,
  species_names,
  MML,
  units = "cm",
  ...
)

plot_TyL(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
plot_TyL(inputs, outputs, species, time_steps, species_names, TyL, ...)

## S4 method for signature 'LeMans_param,missing'
plot_TyL(inputs, N, species, time_steps, species_names, TyL, ...)

## S4 method for signature 'missing,LeMans_outputs'
plot_TyL(wgt, mid, outputs, species, time_steps, species_names, TyL, ...)

## S4 method for signature 'missing,missing'
plot_TyL(
  wgt,
  mid,
  N,
  species,
  time_steps,
  species_names,
  TyL,
  units = "cm",
  ...
)

plot_LQ(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
plot_LQ(inputs, outputs, species, time_steps, species_names, LQ, prob, ...)

## S4 method for signature 'LeMans_param,missing'
plot_LQ(inputs, N, species, species_names, time_steps, LQ, prob, ...)

## S4 method for signature 'missing,LeMans_outputs'
plot_LQ(
  wgt,
  u_bound,
  outputs,
  species,
  time_steps,
  species_names,
  LQ,
```

```
  prob,
  ...
)

## S4 method for signature 'missing,missing'
plot_LQ(
  wgt,
  u_bound,
  N,
  species,
  time_steps,
  species_names,
  LQ,
  prob,
  units = "cm",
  ...
)
```

Arguments

inputs	A LeMans_param object containing the parameter values of the current LeMans model.
outputs	A LeMans_outputs object containing the outputs of the model run.
...	Additional arguments.
species	A numeric value or vector or a character string or vector denoting the species to be used to calculate the indicators. This option is only required if MML, LFI, TyL and LQ are not provided. The default is 1:dim(N)[2].
time_steps	A numeric vector denoting the time steps to be used to calculate and/or plot the indicators. The default is 1:dim(N)[3] or 1:length(indicator).
prob	A numeric value or vector between 0 and 1 denoting the LQ(s) to be calculated. This option is only required if LQ is not provided. The default is 0.5.
length_LFI	A numeric value or vector representing the threshold(s) to be used to calculate the LFI. This option is only required if LFI is not provided. The default is 40.
N	An array with dimensions nsc, nfish and tot_time representing the number of individuals in each length class for each time step, where nsc represents the number of length classes in the model, nfish represents the number of species in the model and tot_time represents the number of time steps that the model was run for. This option is only required if outputs or MML, LFI, TyL and LQ are not provided.
wgt	A matrix with dimensions nsc and nfish representing the weight of each species in each length class. This option is only required if inputs or MML, LFI, TyL and LQ are not provided.
mid	A numeric vector of length nfish representing the mid-point of the length classes in the model, where nfish represents the number of species in the model. This option is only required if inputs or TyL are not provided.

l_bound	A numeric vector of length nsc representing the lower bounds of the length classes, where nsc represents the number of length classes in the model. This option is only required if inputs or LFI are not provided.
u_bound	A numeric vector of length nsc representing the upper bounds of the length classes, where nsc represents the number of length classes in the model. This option is only required if inputs, LFI and/or LQ are not provided.
Linf	A numeric vector of length nfish representing the asymptotic length of each species, where nfish represents the number of species in the model. This option is only required if inputs or MML are not provided.
species_names	A character vector of length 1:dim(N)[2] that denotes the names of the species in the model. This option is only required if inputs is not provided.
MML	A numeric vector representing the outputs of the function get_MML(). This option is only required if inputs and outputs are not provided.
TyL	A numeric vector representing the outputs of the function get_TyL(). This option is only required if inputs and outputs are not provided.
LFI	A numeric vector or matrix representing the outputs of the function get_LFI(). This option is only required if inputs and outputs are not provided.
LQ	A numeric vector or matrix representing the outputs of the function get_LQ(). This option is only required if inputs and outputs are not provided.
units	A character string denoting the units of length used in the model. The default is "cm".

Value

plot_indicators returns a set of four line plots depicting changes in the MML, LFI, TyL and LQ(s) of the community (including only the selected species) over time.

plot_LFI returns a line plot depicting the LFI of the community (including only the selected species) over time.

plot_MML returns a line plot depicting the MML of the community (including only the selected species) over time.

plot_TYL returns a line plot depicting the TyL of the community (including only the selected species) over time.

plot_LQ returns a line plot depicting the LQ(s) of the community (including only the selected species) over time.

See Also

get_indicators

Examples

```
# Set up and run the model
NS_params <- LeMansParam(NS_par, tau=NS_tau, eta=rep(0.25, 21), L50=NS_par$Lmat, other=1e12)
effort <- matrix(0.5, 10, dim(NS_params@Qs)[3])
model_run <- run_LeMans(NS_params, years=10, effort=effort)
```

```
# Calculate the indicators
tmp <- get_indicators(NS_params, model_run)
MML <- tmp$MML
LFI <- tmp$LFI
TyL <- tmp$TYL
LQ <- tmp$LQ

# Plot the indicators
plot_indicators(MML=MML, LFI=LFI, TyL=TyL, LQ=LQ)

# Plot the LFI
plot_LFI(LFI=LFI)

# Plot MML
plot_MML(MML=MML)

# Plot the TyL
plot_TyL(TyL=TyL)

# Plot the LQs
plot_LQ(LQ=LQ)
```

---

plot_SSB *Plot Spawning Stock Biomass (SSB)*

Description

Plots community and/or species-specific Spawning Stock Biomass (SSB) or total biomass.

Usage

```
plot_SSB(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
plot_SSB(inputs, outputs, species, time_steps, species_names, SSB, ...)

## S4 method for signature 'LeMans_param,missing'
plot_SSB(inputs, N, species, time_steps, species_names, SSB, ...)

## S4 method for signature 'missing,LeMans_outputs'
plot_SSB(wgt, mature, outputs, species, time_steps, species_names, SSB, ...)

## S4 method for signature 'missing,missing'
plot_SSB(
  wgt,
  mature,
  N,
  species,
  species_names,
```

```
  time_steps,
  SSB,
  full_plot_only = TRUE,
  units = "g",
  ...
)

plot_biomass(inputs, outputs, ...)

## S4 method for signature 'LeMans_param,LeMans_outputs'
plot_biomass(inputs, outputs, species, time_steps, species_names, biomass, ...)

## S4 method for signature 'LeMans_param,missing'
plot_biomass(inputs, N, species, time_steps, biomass, species_names, ...)

## S4 method for signature 'missing,LeMans_outputs'
plot_biomass(wgt, outputs, species, time_steps, biomass, species_names, ...)

## S4 method for signature 'missing,missing'
plot_biomass(
  wgt,
  N,
  species,
  time_steps,
  species_names,
  biomass,
  full_plot_only = TRUE,
  units = "g",
  ...
)
```

Arguments

inputs	A LeMans_param object containing the parameter values of the current LeMans model. This option is only required if SSB is not provided.
outputs	A LeMans_outputs object containing the outputs of the model run. This option is only required if SSB is not provided.
...	Additional arguments.
species	A numeric value or vector or a character string or vector denoting the species to be used in the plot(s). The default is 1:dim(N)[2].
time_steps	A numeric vector denoting the time steps to be used to calculate and/or plot SSB. The default is 1:dim(N)[3] or 1:length(SSB).
species_names	A character vector of length 1:dim(N)[2] that denotes the names of the species in the model. This option is only required if inputs is not provided.
SSB	A numeric vector or a matrix representing the outputs of the function get_SSB(). This option is only required if inputs and outputs are not provided.

N	An array with dimensions nsc, nfish and tot_time representing the number of individuals in each length class for each time step, where nsc represents the number of length classes in the model, nfish represents the number of species in the model and tot_time represents the number of time steps that the model was run for. This option is only required if outputs and SSB are not provided.
wgt	A matrix with dimensions nsc and nfish representing the weight of each species in each length class. This option is only required if inputs and SSB are not provided.
mature	A matrix with dimensions nsc and nfish and elements in the range 0-1 representing the proportion of individuals that are mature for each species and length class, where nsc represents the number of length classes in the model and nfish represents the number of species in the model. This option is only required if outputs and SSB are not provided.
full_plot_only	A logical statement indicating whether a single plot depicting the SSB of all of the selected species should be produced (full_plot_only=TRUE) or multiple plots depicting the SSB of individual species should be produced (full_plot_only=FALSE). The default is TRUE.
units	A character string denoting the units of weight used in the model. The default is "g".
biomass	A numeric vector or a matrix representing the outputs of the function get_biomass(). This option is only required if inputs and outputs are not provided.

Value

plot_SSB returns line plots of the in SSB of the selected species through time.

plot_biomass returns line plots of the changes in biomass of the selected species through time.

See Also

get_SSB, get_biomass

Examples

```
# Set up and run the model
NS_params <- LeMansParam(NS_par, tau=NS_tau, eta=rep(0.25, 21), L50=NS_par$Lmat, other=1e12)
effort <- matrix(0.5, 10, dim(NS_params@Qs)[3])
model_run <- run_LeMans(NS_params, years=10, effort=effort)

# Calculate SSB
SSB <- get_SSB(NS_params, model_run)

# Plot SSB
plot_SSB(SSB=SSB)

# Calculate biomass
biomass <- get_biomass(NS_params, model_run)

# Plot biomass
plot_biomass(biomass=biomass)
```

rec_BH *The Beverton-Holt stock recruitment function*

Description

Calculates the number of recruits as given by the Beverton-Holt stock recruitment function.

Usage

```
rec_BH(SSB, rec_args)
```

Arguments

SSB	A numeric value representing the Spawning Stock Biomass (SSB) of a given species (g).
rec_args	A list object of length nfish, with each element in the list including a value of a and b for each species. a is a positive numeric value, often referred to as the *density-independent* parameter. The default is 1. b is a positive numeric value, often referred to as the *density-dependent* parameter. The default is 0.001.

Details

The Beverton-Holt stock recruitment function is defined as a*(SSB/1e9))/(1+b*(SSB/1e9)).

Value

A numeric value representing the number of recruits of a given species.

References

Beverton, R.J.H., Holt, S.J. (1957). On the Dynamics of Exploited Fish Populations, volume 19 of Fisheries Investigations (Series 2). United Kingdom Ministry of Agriculture and Fisheries.

See Also

calc_recruits, make_rec_fun, get_rec_fun, rec_Ricker, rec_hockey, rec_const, rec_linear and calc_SSB

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2
```

```
# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                  phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Run calc_ration_growthfac()
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt

# Calculate maturity
mature <- calc_mature(Lmat, nfish, mid, kappa=rep(10, nfish), sc_Linf)

# Create recruitment functions
stored_rec_funs <- get_rec_fun(rep("hockey-stick", nfish))
recruit_params <- do.call("Map", c(c, list(a=NS_par$a, b=NS_par$b)))

# Get an initial population
N0 <- get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept=1e10, slope=-5)

# Calculate the SSB
SSB <- calc_SSB(mature, N0, wgt)

# Run the function
rec_BH(SSB[1], recruit_params[[1]])
```

`rec_const` *The constant stock recruitment function*

Description

Calculates the number of recruits as given by the constant stock recruitment function.

Usage

```
rec_const(SSB, rec_args)
```

Arguments

`SSB`	A numeric value representing the Spawning Stock Biomass (SSB) of a given species (g).
`rec_args`	A list object of length `nfish`, with each element in the list including a value of `a` for each species. `a` is a positive numeric value, often referred to as the *density-independent* parameter. The default is 1.

Details

The number of recruits is defined as the value of a.

Value

A numeric value representing the number of recruits of a given species.

See Also

calc_recruits, make_rec_fun, get_rec_fun, rec_BH, rec_Ricker, rec_hockey, rec_linear and calc_SSB

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                  phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Run calc_ration_growthfac()
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt

# Calculate maturity
mature <- calc_mature(Lmat, nfish, mid, kappa=rep(10, nfish), sc_Linf)

# Create recruitment functions
stored_rec_funs <- get_rec_fun(rep("hockey-stick", nfish))
recruit_params <- do.call("Map", c(c, list(a=NS_par$a, b=NS_par$b)))

# Get an initial population
N0 <- get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept=1e10, slope=-5)

# Calculate the SSB
SSB <- calc_SSB(mature, N0, wgt)
```

```
rec_const(SSB[1], recruit_params[[1]])
```

`rec_hockey` *The hockey-stick stock recruitment function*

Description

Calculates the number of recruits as given by the hockey-stick stock recruitment function.

Usage

```
rec_hockey(SSB, rec_args)
```

Arguments

`SSB`	A numeric value representing the Spawning Stock Biomass (SSB) of a given species (g).
`rec_args`	A list object of length `nfish`, with each element in the list including a value of `a` and `b` for each species. `a` is a positive numeric value, often referred to as the *density-independent* parameter. The default is 1. `b` is a positive numeric value, often referred to as the *density-dependent* parameter. The default is 0.001.

Details

The stock recruitment function is defined as `min(a*(SSB/1e9), b)`.

Value

A numeric value representing the number of recruits of a given species.

References

Barrowman, N.J., Myers, R.A. (2000). Still more spawner-recruit curves: the hockey stick and its generalisations. *Canadian Journal of Fisheries and Aquatic Science*, 57:665–676.

Thorpe, R.B., Le Quesne, W.J.F., Luxford, F., Collie, J.S., Jennings, S. (2015). Evaluation and management implications of uncertainty in a multispecies size-structured model of population and community responses to fishing. *Methods in Ecology and Evolution*, 6:49-58.

See Also

`calc_recruits`, `make_rec_fun`, `get_rec_fun`, `rec_BH`, `rec_Ricker`, `rec_const`, `rec_linear` and `calc_SSB`

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Run calc_ration_growthfac()
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt

# Calculate maturity
mature <- calc_mature(Lmat, nfish, mid, kappa=rep(10, nfish), sc_Linf)

# Create recruitment functions
stored_rec_funs <- get_rec_fun(rep("hockey-stick", nfish))
recruit_params <- do.call("Map", c(c, list(a=NS_par$a, b=NS_par$b)))

# Get an initial population
N0 <- get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept=1e10, slope=-5)

# Calculate the SSB
SSB <- calc_SSB(mature, N0, wgt)

rec_hockey(SSB[1], recruit_params[[1]])
```

---

rec_linear *The density-independent stock recruitment function*

Description

Calculates the number of recruits as given by the density-independent stock recruitment function.

Usage

```
rec_linear(SSB, rec_args)
```

Arguments

SSB	A numeric value representing the Spawning Stock Biomass (SSB) of a given species (g).
rec_args	A list object of length nfish, with each element in the list including a value of a for each species. a is a positive numeric value, often referred to as the *density-independent* parameter. The default is 1.

Details

The number of recruits is defined as a*(SSB/1e9).

Value

A numeric value representing the number of recruits of a given species.

References

Ogle, D.H. (2016). Introductory Fisheries Analyses with R. CRC Press.

See Also

calc_recruits, make_rec_fun, get_rec_fun, rec_BH, rec_Ricker, rec_hockey, rec_const and calc_SSB

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                 phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Run calc_ration_growthfac()
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt
```

```
# Calculate maturity
mature <- calc_mature(Lmat, nfish, mid, kappa=rep(10, nfish), sc_Linf)

# Create recruitment functions
stored_rec_funs <- get_rec_fun(rep("hockey-stick", nfish))
recruit_params <- do.call("Map", c(c, list(a=NS_par$a, b=NS_par$b)))

# Get an initial population
N0 <- get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept=1e10, slope=-5)

# Calculate the SSB
SSB <- calc_SSB(mature, N0, wgt)

rec_linear(SSB[1], recruit_params[[1]])
```

`rec_Ricker` *The Ricker stock recruitment function*

Description

Calculates the number of recruits as given by the Ricker stock recruitment function.

Usage

```
rec_Ricker(SSB, rec_args)
```

Arguments

`SSB`	A numeric value representing the Spawning Stock Biomass (SSB) of a given species (g).
`rec_args`	A list object of length `nfish`, with each element in the list including a value of `a` and `b` for each species. `a` is a positive numeric value, often referred to as the *density-independent* parameter. The default is 1. `b` is a positive numeric value, often referred to as the *density-dependent* parameter. The default is 0.001.

Details

The Ricker stock recruitment function is defined as `a*(SSB/1e9))*exp(-b*(SSB/1e9))`.

Value

A numeric value representing the number of recruits of a given species.

References

Hall, S. J., Collie, J. S., Duplisea, D. E., Jennings, S., Bravington, M., & Link, J. (2006). A length-based multispecies model for evaluating community responses to fishing. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(6):1344-1359.

Ricker, W.E. (1954). Stock and recruitment. *Journal of the Fisheries Research Board of Canada*, 11:559-623.

See Also

calc_recruits, make_rec_fun, get_rec_fun, rec_BH, rec_hockey, rec_const, rec_linear and calc_SSB

Examples

```
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                  phi_min=0.1) # fixed phi_min
phi_min <- tmp$phi_min

# Run calc_ration_growthfac()
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt

# Calculate maturity
mature <- calc_mature(Lmat, nfish, mid, kappa=rep(10, nfish), sc_Linf)

# Create recruitment functions
stored_rec_funs <- get_rec_fun(rep("hockey-stick", nfish))
recruit_params <- do.call("Map", c(c, list(a=NS_par$a, b=NS_par$b)))

# Get an initial population
N0 <- get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept=1e10, slope=-5)

# Calculate the SSB
SSB <- calc_SSB(mature, N0, wgt)

rec_Ricker(SSB[1], recruit_params[[1]])
```

---

run_LeMans *Project the LeMans model*

Description

Project the LeMans model forward in time.

Usage

```
run_LeMans(params, ...)

## S4 method for signature 'missing'
run_LeMans(
  N0,
  Fs,
  tot_time,
  nsc,
  nfish,
  phi_min,
  mature,
  sc_Linf,
  wgt,
  phi,
  ration,
  other,
  M1,
  suit_M2,
  stored_rec_funs,
  recruit_params,
  eps = 1e-05
)

## S4 method for signature 'LeMans_param'
run_LeMans(
  params,
  years = 10,
  N0 = NULL,
  effort = matrix(0, years, dim(params@Qs)[3]),
  Fs,
  intercept = 1e+10,
  slope = -5,
  tot_time
)
```

Arguments

params	A LeMans_param object containing the parameter values of the current LeMans model.
...	Additional arguments.
N0	A matrix with dimensions nsc and nfish representing the number of individuals in each length class when the model is initialised.

Fs	An array with dimensions nsc, nfish and tot_time representing the fishing mortality of each species in each length class at each time step.
tot_time	A numeric value representing the number of time steps to run the model for.
nsc	A numeric value representing the number of length classes in the model.
nfish	A numeric value representing the number of fish species in the model.
phi_min	A numeric value representing the time step of the model.
mature	A matrix with dimensions nsc and nfish and elements in the range 0-1 representing the proportion of individuals that are mature for each species and length class.
sc_Linf	A numeric vector of length nsc representing the length class at which each species reaches its asymptotic length.
wgt	A matrix with dimensions nsc and nfish representing the weight of each species in each length class.
phi	A matrix with dimensions nsc and nfish representing the proportion of individuals that leave each length class.
ration	A matrix with dimensions nsc and nfish representing the amount of food required for fish of a given species and length class to grow according to the von Bertalanffy growth curve in a time step.
other	A numeric value representing the amount of other food (g) available from prey that is not explicitly represented in the model.
M1	A matrix of dimensions nsc and nfish representing the natural mortality of each species for each length class.
suit_M2	A list object of length nfish. Each element in the list is an array of dimensions nsc, nsc and nfish containing a value between zero and 1 representing prey preference and prey suitability for each species and length class.
stored_rec_funs	A list object of length nfish where each element includes the stock recruitment function for each species. If an invalid recruitment function is selected, NULL is returned and a warning message is shown.
recruit_params	A list object of length nfish specifying the parameters for the recruitment function.
eps	A numeric value specifying a numerical offset. The default value is 1e-5.
years	A numeric value representing the number of years that the model is run for. The default is 10.
effort	A matrix with dimensions years and the number of fishing gears, representing fishing effort in each year for each gear. This parameter is required only if Fs is missing.
intercept	A numeric value representing the number of individuals in the first length class. This parameter is only required if N0 is missing. The default is 1e10.
slope	A numeric value representing the slope of the community size spectrum. This parameter is only required if N0 is missing. The default is -5.

Value

An object of class LeMans_outputs.

See Also

LeMans_outputs, LeMans_param, LeMansParam

Examples

```
# Run the model with all inputs specified explicitly:
# Set up the inputs to the function - species-independent parameters
nfish <- nrow(NS_par)
nsc <- 32
maxsize <- max(NS_par$Linf)*1.01 # the biggest size is 1% bigger than the largest Linf
l_bound <- seq(0, maxsize, maxsize/nsc); l_bound <- l_bound[-length(l_bound)]
u_bound <- seq(maxsize/nsc, maxsize, maxsize/nsc)
mid <- l_bound+(u_bound-l_bound)/2

# Set up the inputs to the function - species-specific parameters
Linf <- NS_par$Linf # the von-Bertalanffy asymptotic length of each species (cm).
W_a <- NS_par$W_a # length-weight conversion parameter.
W_b <- NS_par$W_b # length-weight conversion parameter.
k <- NS_par$k # the von-Bertalnaffy growth parameter.
Lmat <- NS_par$Lmat # the length at which 50\% of individuals are mature (cm).

# Get phi_min
tmp <- calc_phi(k, Linf, nsc, nfish, u_bound, l_bound, calc_phi_min=FALSE,
                 phi_min=0.1) # fixed phi_min
phi <- tmp$phi
phi_min <- tmp$phi_min

# Calculate growth increments
tmp <- calc_ration_growthfac(k, Linf, nsc, nfish, l_bound, u_bound, mid, W_a, W_b, phi_min)
ration <- tmp$ration
sc_Linf <- tmp$sc_Linf
wgt <- tmp$wgt
g_eff <- tmp$g_eff

# Calculate maturity
mature <- calc_mature(Lmat, nfish, mid, kappa=rep(10, nfish), sc_Linf)

# Create recruitment functions
stored_rec_funs <- get_rec_fun(rep("hockey-stick", nfish))
recruit_params <- do.call("Map", c(c, list(a=NS_par$a, b=NS_par$b)))

# Calculate background mortality
M1 <- calc_M1(nsc, sc_Linf, phi_min)

# Calculate predator-prey size preferences
prefs <- calc_prefs(pred_mu=-2.25, pred_sigma=0.5, wgt, sc_Linf)

# Calculate prey preference and prey suitability
```

```
suit_M2 <- calc_suit_vect(nsc, nfish, sc_Linf, prefs, NS_tau)

# Calculate catchability
Qs <- calc_Q(curve=rep("logistic", nfish), species=NS_par$species_names,
             max_catchability=rep(1, nfish), gear_name=NS_par$species_names,
             nsc=nsc, nfish=nfish, mid=mid, l_bound=l_bound, u_bound=u_bound,
             species_names=NS_par$species_names, eta=rep(0.25, nfish), L50=Lmat)

# Get an initial population
N0 <- get_N0(nsc, nfish, mid, wgt, sc_Linf, intercept=1e10, slope=-5)
years <- 10 # run the model for 10 years
tot_time <- years*phi_min # total number of time steps

# Define fishing effort to be 0.5 for all species
effort <- matrix(0.5, tot_time, dim(Qs)[3])

# Calculate F
Fs <- array(0, dim=c(nsc, nfish, tot_time))
for (j in 1:ncol(effort)) {
  for (ts in 1:tot_time) {
    Fs[,,ts] <- Fs[,,ts]+effort[ts, j]*Qs[,,j]
  }
}

# Run the model
model_run <- run_LeMans(N0=N0, tot_time=tot_time, Fs=Fs, nsc=nsc, nfish=nfish,
                        phi_min=phi_min, mature=mature, sc_Linf=sc_Linf, wgt=wgt,
                        phi=phi, ration=ration, other=NS_other, M1=M1, suit_M2=suit_M2,
                        stored_rec_funs=stored_rec_funs, recruit_params=recruit_params,
                        eps=1e-05)

##############################################
# Alternatively:
NS_params <- LeMansParam(NS_par,tau=NS_tau,eta=rep(0.25,21),L50=NS_par$Lmat,other=NS_other)

# Define fishing effort
effort <- matrix(0.5, 10, dim(NS_params@Qs)[3])

# Run the model
model_run <- run_LeMans(NS_params, years=10, effort=effort)
```

Index