Package 'SAMtool'

July 21, 2025

Type Package

Title Stock Assessment Methods Toolkit

Version 1.8.1

Date 2025-04-28

Maintainer Quang Huynh <quang@bluematterscience.com>

Description Simulation tools for closed-loop simulation are provided for the 'MSEtool' operating model to inform data-rich fisheries. 'SAMtool' provides a conditioning model, assessment models of varying complexity with standardized reporting, model-based management procedures, and diagnostic tools for evaluating assessments inside closed-loop simulation.

License GPL-3

Depends R (>= 3.5.0), MSEtool (>= 3.0.0)

Imports TMB (>= 1.9.0), abind, dplyr, gplots, graphics, methods, pbapply, rmarkdown, snowfall, stats, utils, vars

LinkingTo TMB, RcppEigen

Suggests caret, corpcor, covr, extraDistr, ggplot2, Gmisc, knitr, mvtnorm, numDeriv, reshape2, shiny, testthat, tmbstan, usethis

LazyData yes

LazyLoad yes

RoxygenNote 7.3.2

Encoding UTF-8

URL https://openmse.com, https://samtool.openmse.com, https://github.com/Blue-Matter/SAMtool

BugReports https://github.com/Blue-Matter/SAMtool/issues

NeedsCompilation yes

Author Quang Huynh [aut, cre], Tom Carruthers [aut], Adrian Hordyk [aut]

Repository CRAN

Date/Publication 2025-04-28 22:30:06 UTC

Contents

SAMtool-package . . . 3
Assessment-class . . . 4
cDD . . . 7
check_RCMdata . . . 11
compare_models . . . 20
DD_TMB . . . 21
diagnostic . . . 26
getinds . . . 27
HCRlin . . . 28
HCR_escapement . . . 29
HCR_FB . . . 30
HCR_fixedF . . . 31
HCR_MSY . . . 32
HCR_ramp . . . 34
HCR_segment . . . 37
mahplot . . . 39
make_interim_MP . . . 40
Model-based-MP . . . 42
pcod . . . 44
plot.Assessment . . . 45
plot.prof . . . 46
plot.RCModel . . . 47
plot.retro . . . 49
plot_betavar . . . 50
plot_composition . . . 51
plot_crosscorr . . . 53
plot_lognormalvar . . . 54
plot_residuals . . . 55
plot_SR . . . 56
plot_steepness . . . 57
plot_timeseries . . . 58
posterior . . . 59
PRBcalc . . . 61
prelim_AM . . . 62
Probs . . . 63
prof-class . . . 64
profile,Assessment-method . . . 64
project-class . . . 66
projection . . . 66
RCM2MOM . . . 68
RCMdata-class . . . 69
RCModel-class . . . 71

RCM_assess . . . 73
retro-class . . . 75
retrospective . . . 75
retrospective_AM . . . 76
SCA . . . 78
SCA_CAL . . . 84
SCA_DDM . . . 87
SCA_RWM . . . 90
Shortcut . . . 94
sim-class . . . 97
simulate . . . 97
SP . . . 98
SP_production . . . 103
SSS . . . 105
summary.Assessment . . . 106
swordfish . . . 107
TAC_MSY . . . 108
userguide . . . 108
VPA . . . 109

Index **113**

`SAMtool-package` *Stock Assessment Methods Toolkit*

Description

Simulation tools for closed-loop simulation are provided for the 'MSEtool' operating model to inform data-rich fisheries. SAMtool provides an OM conditioning model, assessment models of varying complexity with standardized reporting, diagnostic tools for evaluating assessments within closed-loop simulation, and helper functions for building more complex operating models and model-based management procedures.

How to use SAMtool

The main features of SAMtool are the assessment models and the ability to make model-based management procedures by combining assessment models with harvest control rules. Such MPs can be used and tested in management strategy evaluation with MSEtool operating models. An overview of these features is available on the openMSE website.

The `RCM()` (Rapid Conditioning Model) can be used to condition operating models from real data.

The following articles are available on the openMSE website:

- Description of assessment models
- General overview of RCM

The function documentation can be viewed online.

Author(s)

Quang Huynh <quang@bluematterscience.com>

Tom Carruthers <tom@bluematterscience.com>

Adrian Hordyk <adrian@bluematterscience.com>

References

Carruthers, T.R., Punt, A.E., Walters, C.J., MacCall, A., McAllister, M.K., Dick, E.J., Cope, J. 2014. Evaluating methods for setting catch limits in data-limited fisheries. Fisheries Research. 153: 48-68.

Carruthers, T.R., Kell, L.T., Butterworth, D.S., Maunder, M.N., Geromont, H.F., Walters, C., McAllister, M.K., Hillary, R., Levontin, P., Kitakado, T., Davies, C.R. Performance review of simple management procedures. ICES Journal of Marine Science. 73: 464-482.

See Also

Useful links:

- https://openmse.com
- https://samtool.openmse.com
- https://github.com/Blue-Matter/SAMtool
- Report bugs at https://github.com/Blue-Matter/SAMtool/issues

---

`Assessment-class` *Class*-`Assessment`

Description

An S4 class that contains assessment output. Created from a function of class `Assess`.

Slots

`Model` Name of the assessment model.

`Name` Name of Data object.

`conv` Logical. Whether the assessment model converged (defined by whether TMB returned a positive-definite covariance matrix for the model).

`UMSY` Estimate of exploitation at maximum sustainable yield.

`FMSY` Estimate of instantaneous fishing mortality rate at maximum sustainable yield.

`MSY` Estimate of maximum sustainable yield.

`BMSY` Biomass at maximum sustainable yield.

`SSBMSY` Spawning stock biomass at maximum sustainable yield.

`VBMSY` Vulnerable biomass at maximum sustainable yield.

`B0` Biomass at unfished equilibrium.

`R0` Recruitment at unfished equilibrium.

`N0` Abundance at unfished equilibrium.

`SSB0` Spawning stock biomass at unfished equilibrium.

`VB0` Vulnerable biomass at unfished equilibrium.

`h` Steepness.

`U` Time series of exploitation.

`U_UMSY` Time series of relative exploitation.

`FMort` Time series of instantaneous fishing mortality.

`F_FMSY` Time series of fishing mortality relative to MSY.

`B` Time series of biomass.

`B_BMSY` Time series of biomass relative to MSY.

`B_B0` Time series of depletion.

`SSB` Time series of spawning stock biomass.

`SSB_SSBMSY` Time series of spawning stock biomass relative to MSY.

`SSB_SSB0` Time series of spawning stock depletion.

`VB` Time series of vulnerable biomass.

`VB_VBMSY` Time series of vulnerable biomass relative to MSY.

`VB_VB0` Time series of vulnerable biomass depletion.

`R` Time series of recruitment.

`N` Time series of population abundance.

`N_at_age` Time series of numbers-at-age matrix.

`Selectivity` Selectivity-at-age matrix.

`Obs_Catch` Observed catch.

`Obs_Index` Observed index.

`Obs_C_at_age` Observed catch-at-age matrix.

`Catch` Predicted catch.

`Index` Predicted index.

`C_at_age` Predicted catch-at-age matrix.

`Dev` A vector of estimated deviation parameters.

`Dev_type` A description of the deviation parameters, e.g. "log recruitment deviations".

`NLL` Negative log-likelihood. A vector for the total likelihood, integrated across random effects if applicable, components, and penalty term (applied when `U > 0.975` in any year).

`SE_UMSY` Standard error of UMSY estimate.

`SE_FMSY` Standard error of FMSY estimate.

`SE_MSY` Standard error of MSY estimate.

`SE_U_UMSY` Standard error of U/UMSY.

`SE_F_FMSY` Standard error of F/FMSY.

`SE_B_BMSY` Standard error of B/BMSY.

`SE_B_B0` Standard error of B/B0.

`SE_SSB_SSBMSY` Standard error of SSB/SSBMSY.

`SE_SSB_SSB0` Standard error of SSB/SSB0.

`SE_VB_VBMSY` Standard error of VB/VBMSY.

`SE_VB_VB0` Standard error of VB/VB0.

`SE_Dev` A vector of standard errors of the deviation parameters.

`info` A list containing the data and starting values of estimated parameters for the assessment.

`forecast` A list containing components for forecasting:

- `per_recruit` A data frame of SPR (spawning potential ratio) and YPR (yield-per-recruit), calculated for a range of exploitation rate of 0 - 0.99 or instantaneous F from 0 - 2.5 FMSY.
- `catch_eq` A function that calculates the catch for the next year (after the model terminal year) when an apical F is provided.

`obj` A list with components returned from `TMB::MakeADFun()`.

`opt` A list with components from calling `stats::nlminb()` to `obj`.

`SD` A list (class sdreport) with parameter estimates and their standard errors, obtained from `TMB::sdreport()`.

`TMB_report` A list of model output reported from the TMB executable, i.e. `obj$report()`, and derived quantities (e.g. MSY).

`dependencies` A character string of data types required for the assessment.

Author(s)

Q. Huynh

See Also

plot.Assessment summary.Assessment retrospective profile make_MP

Examples

```
output <- DD_TMB(Data = MSEtool::SimulatedData)
class(output)
```

cDD *Continuous Delay-differential assessment model*

Description

A catch and index-based assessment model. Compared to the discrete delay-difference (annual time-step in production and fishing), the delay-differential model (cDD) is based on continuous recruitment and fishing mortality within a time-step. The continuous model works much better for populations with high turnover (e.g. high F or M, continuous reproduction). This model is conditioned on catch and fits to the observed index. In the state-space version (cDD_SS), recruitment deviations from the stock-recruit relationship are estimated.

Usage

```
cDD(
  x = 1,
  Data,
  AddInd = "B",
  SR = c("BH", "Ricker"),
  rescale = "mean1",
  MW = FALSE,
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  dep = 1,
  LWT = list(),
  n_itF = 5L,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 5000, eval.max = 10000),
  ...
)

cDD_SS(
  x = 1,
  Data,
  AddInd = "B",
  SR = c("BH", "Ricker"),
  rescale = "mean1",
  MW = FALSE,
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  fix_sigma = FALSE,
  fix_tau = TRUE,
  dep = 1,
```

```
  LWT = list(),
  n_itF = 5L,
  integrate = FALSE,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 5000, eval.max = 10000),
  inner.control = list(),
  ...
)
```

Arguments

x	An index for the objects in Data when running in closed loop simulation. Otherwise, equals to 1 when running an assessment.
Data	An object of class MSEtool::Data.
AddInd	A vector of integers or character strings indicating the indices to be used in the model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd.
SR	Stock-recruit function (either "BH" for Beverton-Holt or "Ricker").
rescale	A multiplicative factor that rescales the catch in the assessment model, which can improve convergence. By default, "mean1" scales the catch so that time series mean is 1, otherwise a numeric. Output is re-converted back to original units.
MW	Logical, whether to fit to mean weight. In closed-loop simulation, mean weight will be grabbed from Data@Misc[[x]]$MW, otherwise calculated from Data@CAL.
start	Optional list of starting values. Entries can be expressions that are evaluated in the function. See details.
prior	A named list for the parameters of any priors to be added to the model. See below.
fix_h	Logical, whether to fix steepness to value in Data@steep in the assessment model.
dep	The initial depletion in the first year of the model. A tight prior is placed on the model objective function to estimate the equilibrium fishing mortality corresponding to the initial depletion. Due to this tight prior, this F should not be considered to be an independent model parameter. Set to zero to eliminate this prior.
LWT	A named list of likelihood weights. For LWT$Index, a vector of likelihood weights for each survey, while for LWT$MW a numeric.
n_itF	Integer, the number of iterations to solve F conditional on the observed catch.
silent	Logical, passed to TMB::MakeADFun(), whether TMB will print trace information during optimization. Used for diagnostics for model convergence.

opt_hess	Logical, whether the hessian function will be passed to stats::nlminb() during optimization (this generally reduces the number of iterations to convergence, but is memory and time intensive and does not guarantee an increase in convergence rate). Ignored if integrate = TRUE.
n_restart	The number of restarts (calls to stats::nlminb()) in the optimization procedure, so long as the model hasn't converged. The optimization continues from the parameters from the previous (re)start.
control	A named list of parameters regarding optimization to be passed to stats::nlminb().
...	Additional arguments (not currently used).
fix_sigma	Logical, whether the standard deviation of the index is fixed. If TRUE, sigma is fixed to value provided in start (if provided), otherwise, value based on Data@CV_Ind.
fix_tau	Logical, the standard deviation of the recruitment deviations is fixed. If TRUE, tau is fixed to value provided in start (if provided), otherwise, equal to 1.
integrate	Logical, whether the likelihood of the model integrates over the likelihood of the recruitment deviations (thus, treating it as a state-space variable). Otherwise, recruitment deviations are penalized parameters.
inner.control	A named list of arguments for optimization of the random effects, which is passed on to TMB::newton() via TMB::MakeADFun().

Details

For start (optional), a named list of starting values of estimates can be provided for:

- R0 Unfished recruitment. Otherwise, Data@OM$R0[x] is used in closed-loop, and 400% of mean catch otherwise.
- h Steepness. Otherwise, Data@steep[x] is used, or 0.9 if empty.
- Kappa Delay-differential Kappa parameter. Otherwise, calculated from biological parameters in the Data object.
- F_equilibrium Equilibrium fishing mortality leading into first year of the model (to determine initial depletion). By default, 0.
- tau Lognormal SD of the recruitment deviations (process error) for DD_SS. By default, Data@sigmaR[x].
- sigma Lognormal SD of the index (observation error). By default, Data@CV_Ind[x]. Not used if multiple indices are used.
- sigma_W Lognormal SD of the mean weight (observation error). By default, 0.1.

Multiple indices are supported in the model. Data@Ind, Data@VInd, and Data@SpInd are all assumed to be biomass-based. For Data@AddInd, Data@I_units are used to identify a biomass vs. abundance-based index.

Value

An object of Assessment containing objects and output from TMB.

Priors

The following priors can be added as a named list, e.g., `prior = list(M = c(0.25, 0.15), h = c(0.7, 0.1)`. For each parameter below, provide a vector of values as described:

- `R0` - A vector of length 3. The first value indicates the distribution of the prior: `1` for lognormal, `2` for uniform on `log(R0)`, `3` for uniform on R0. If lognormal, the second and third values are the prior mean (in normal space) and SD (in log space). Otherwise, the second and third values are the lower and upper bounds of the uniform distribution (values in normal space).
- `h` - A vector of length 2 for the prior mean and SD, both in normal space. Beverton-Holt steepness uses a beta distribution, while Ricker steepness uses a normal distribution.
- `M` - A vector of length 2 for the prior mean (in normal space) and SD (in log space). Lognormal prior.
- `q` - A matrix for nsurvey rows and 2 columns. The first column is the prior mean (in normal space) and the second column for the SD (in log space). Use `NA` in rows corresponding to indices without priors.

See online documentation for more details.

Online Documentation

Model description and equations are available on the openMSE website.

Required Data

- `cDD`: Cat, Ind, Mort, L50, vbK, vbLinf, vbt0, wla, wlb, MaxAge
- `cDD_SS`: Cat, Ind, Mort, L50, vbK, vbLinf, vbt0, wla, wlb, MaxAge

Optional Data

- `cDD`: steep
- `cDD_SS`: steep, CV_Ind, sigmaR

Author(s)

Q. Huynh

References

Hilborn, R., and Walters, C., 1992. Quantitative Fisheries Stock Assessment: Choice, Dynamics and Uncertainty. Chapman and Hall, New York.

See Also

DD_TMB plot.Assessment summary.Assessment retrospective profile make_MP

Examples

```
#### Observation-error delay difference model
res <- cDD(Data = MSEtool::Red_snapper)

### State-space version
### Also set recruitment variability SD = 0.6 (since fix_tau = TRUE)
res <- cDD_SS(Data = MSEtool::Red_snapper, start = list(tau = 0.6))

summary(res@SD) # Parameter estimates
```

---

check_RCMdata *Rapid Conditioning Model (RCM)*

Description

Intended for conditioning operating models for MSEtool. For data-limited stocks, this function can generate a range of potential depletion scenarios inferred from sparse data. From a historical time series of total catch or effort, and potentially age/length compositions and multiple indices of abundance, the RCM returns a range of values for depletion, selectivity, unfished recruitment (R0), historical fishing effort, and recruitment deviations for the operating model. This is done by sampling life history parameters provided by the user and fitting a statistical catch-at-age model (with the predicted catch equal to the observed catch). Alternatively one can do a single model fit and sample the covariance matrix to generate an operating model with uncertainty based on the model fit. Either a full catch (conditioned on catch) or effort (conditioned on effort) time series is needed but missing data (as NAs) are allowed for all other data types. check_RCMdata evaluates whether the inputs in the S4 RCMdata object are correctly formatted.

Usage

```
check_RCMdata(RCMdata, OM, condition = "catch", silent = FALSE)

RCM(OM, data, ...)

## S4 method for signature 'OM,RCMdata'
RCM(
  OM,
  data,
  condition = "catch",
  selectivity = "logistic",
  s_selectivity = NULL,
  LWT = list(),
 comp_like = c("multinomial", "lognormal", "mvlogistic", "dirmult1", "dirmult2"),
  prior = list(),
  max_F = 3,
  cores = 1L,
  integrate = FALSE,
  mean_fit = FALSE,
```

```
  drop_nonconv = FALSE,
  drop_highF = FALSE,
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  start = list(),
  map = list(),
  silent = FALSE,
  ...
)

## S4 method for signature 'OM,list'
RCM(
  OM,
  data,
  condition = "catch",
  selectivity = "logistic",
  s_selectivity = NULL,
  LWT = list(),
 comp_like = c("multinomial", "lognormal", "mvlogistic", "dirmult1", "dirmult2"),
  ESS = c(30, 30),
  prior = list(),
  max_F = 3,
  cores = 1L,
  integrate = FALSE,
  mean_fit = FALSE,
  drop_nonconv = FALSE,
  drop_highF = FALSE,
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  start = list(),
  map = list(),
  silent = FALSE,
  ...
)

## S4 method for signature 'OM,Data'
RCM(
  OM,
  data,
  condition = "catch",
  selectivity = "logistic",
  s_selectivity = NULL,
  LWT = list(),
 comp_like = c("multinomial", "lognormal", "mvlogistic", "dirmult1", "dirmult2"),
  ESS = c(30, 30),
  prior = list(),
  max_F = 3,
  cores = 1L,
  integrate = FALSE,
  mean_fit = FALSE,
```

```
  drop_nonconv = FALSE,
  drop_highF = FALSE,
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  start = list(),
  map = list(),
  silent = FALSE,
  ...
)

## S4 method for signature 'list,RCMdata'
RCM(
  OM,
  data,
  condition = "catch",
  selectivity = "logistic",
  s_selectivity = NULL,
  LWT = list(),
 comp_like = c("multinomial", "lognormal", "mvlogistic", "dirmult1", "dirmult2"),
  prior = list(),
  max_F = 3,
  integrate = FALSE,
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  start = list(),
  map = list(),
  silent = FALSE,
  ...
)
```

Arguments

RCMdata	An RCMdata object.
OM	An object of class MSEtool::OM that specifies natural mortality (M), growth (Linf, K, t0, a, b), stock-recruitment relationship, steepness, maturity parameters (L50 and L50_95), and standard deviation of recruitment variability (Perr). Alternatively, provide a named list of biological inputs, see "StockPars" section below.
condition	String to indicate whether the RCM is conditioned on "catch" (where F are estimated parameters), "catch2" (where F is solved internally using Newton's method), or "effort" (F is proportional to an index series in `data@Ehist`. Can be fleet specific, in which case some combination of "catch" and "effort" are permissible.
silent	Logical to indicate whether informative messages will be reported to console.
data	Data inputs formatted in a RCMdata (preferred) or MSEtool::Data object. Use of a list is deprecated. See Data section below.
...	Other arguments to pass in for starting values of parameters and fixing parameters. See details.

`selectivity` A character vector of length nfleet to indicate `"logistic_length"`, `"dome_length"`, `"logistic_age"`, `"dome_age"`, or `"free"` selectivity for each fleet in `Chist`. If there is time-varying selectivity, this is a character vector of length nsel_block (see Data section below). "free" indicates independent selectivity parameters for each age, and additional modifications for fixing selectivity parameters will likely be needed. See Additional arguments section.

`s_selectivity` A vector of length nsurvey to indicate the selectivity of the corresponding columns in `data$Index`. Use `"B"` for total biomass, or `"SSB"` for spawning biomass (by default, "B" is used). Use numbers if the survey selectivity follows a fleet (corresponding to the columns in data$Chist, e.g., 1 = first fleet/column and so on). If the survey selectivity is otherwise independent of anything else in the model, use `"logistic_length"`, `"dome_length"`, `"logistic_age"`, `"dome_age"`, or `"free"` to specify the functional form of selectivity, and see Additional arguments section for setup of survey selectivity parameters and Articles section for more information.

`LWT` A named list of likelihood weights for the RCM. See below.

`comp_like` A string indicating the statistical distribution for the composition data, either `"multinomial"` (default), `"lognormal"`, `"mvlogistic"` (multivariate logistic), `"dirmult1"` (Dirichlet multinomial, linear version), or `"dirmult2"` (saturating version; see Thorson et al. 2017).

`prior` A named list for the parameters of any priors to be added to the model. See below.

`max_F` The maximum F for any fleet in the scoping model (higher F's in the model are penalized in the objective function). This argument will also update `OM@maxF`. See also `drop_highF`.

`cores` Integer for the number of CPU cores (set greater than 1 for parallel processing).

`integrate` Logical, whether to treat recruitment deviations as penalized parameters in the likelihood (FALSE) or random effects to be marginalized out of the likelihood (TRUE).

`mean_fit` Logical, whether to run an additional with mean values of life history parameters from the OM.

`drop_nonconv` Logical, whether to drop non-converged fits of the RCM, including fits where F = NA.

`drop_highF` Logical, whether to drop fits of the RCM where F = `max_F`.

`control` A named list of arguments (e.g, max. iterations, etc.) for optimization, to be passed to the control argument of `stats::nlminb()`.

`start` A list of starting values for the TMB model. See details.

`map` A list of `map` argument to TMB models to override defaults. See MakeADFun and details.

`ESS` A vector of length two. A shortcut method to setting the maximum multinomial sample size of the age and length compositions. Not used when data are provided in a RCMdata object.

Details

Fleet selectivity is fixed to values sampled from `OM` if no age or length compositions are provided.

Survey selectivity is estimable only if `IAA` or `IAL` is provided. Otherwise, the selectivity should be mirrored to a fleet (vulnerable biomass selectivity) or indexed to total or spawning biomass (see `s_selectivity`).

Parameters that were used in the fitting model are placed in the `RCM@OM@cpars` list.

If the operating model `OM` uses time-varying growth or M, then those trends will be used in the RCM as well. Non-stationary productivity creates ambiguity in the calculation and interpretation of depletion and MSY reference points.

The easiest way to turn off time-varying growth/M is by setting: `OM@Msd <- OM@Linfsd <- OM@Ksd <- c(0, 0)`.

To play with alternative fits by excluding indices, for example, or other optional data, set the corresponding likelihood weight to zero. The model will still generate the inferred index but the data won't enter the likelihood. See section on likelihood weights.

Value

An object of class RCModel (see link for description of output).

`check_RCMdata` returns a list of updated RCMdata object, OM, and StockPars and FleetPars from the Hist object generated from the OM.

Online Documentation

Several articles are available for RCM:

- General overview of approach
- Mathematical description
- Setup of selectivity settings and index catchability (useful for more data-rich cases)
- Description of priors

Priors

The following priors can be added as a named list, e.g., `prior = list(M = c(0.25, 0.15), h = c(0.7, 0.1)`. For each parameter below, provide a vector of values as described:

`R0` A vector of length 3. The first value indicates the distribution of the prior: `1` for lognormal, `2` for uniform on `log(R0)`, `3` for uniform on R0. If lognormal, the second and third values are the prior mean (in normal space) and SD (in log space). Otherwise, the second and third values are the lower and upper bounds of the uniform distribution (values in normal space).

`h` A vector of length 2 for the prior mean and SD, both in normal space. Beverton-Holt steepness uses a beta distribution, while Ricker steepness uses a normal distribution.

`M` A vector of length 2 for the prior mean (in normal space) and SD (in log space). Lognormal prior.

`q` A matrix for nsurvey rows and 2 columns. The first column is the prior mean (in normal space) and the second column for the SD (in log space). Use `NA` in rows corresponding to indices without priors.

See online documentation for more details.

Data

One of indices, age compositions, or length compositions should be provided in addition to the historical catch or effort. Not all arguments are needed to run the model (some have defaults, while others are ignored if not applicable depending on the data provided).

The `data` variable can be an object of class RCMdata. See help file for description of inputs.

Alternatively, the `data` input can be a MSEtool::Data S4 object which will retrieve data from the following slots:

`Data@Cat` catch series (single fleet with the Data S4 object)

`Data@Effort` effort series

`Data@CAA` fishery age composition

`Data@CAL`, `Data@CAL_mids` fishery length composition and corresponding length bins

`Data@Ind`, `Data@SpInd`, `Data@VInd`, `Data@AddInd` indices of abundance

`Data@CV_Ind`, `Data@CV_SpInd`, `Data@CV_VInd`, `Data@CV_AddInd` annual coefficients of variation for the corresponding indices of abundance. CVs will be converted to lognormal standard deviations.

`Data@ML` fishery mean lengths

`Data@AddIndV`, `Data@AddIndType`, `Data@AddIunits` Additional information for indices in `Data@AddInd`: selectivity and units (i.e., biomass or abundance).

There is no slot in the Data S4 object for the equilibrium catch/effort. These can be passed directly in the function call, i.e., `RCM(OM, Data, C_eq = C_eq, ...)`.

Data list (deprecated)

Use of a list is deprecated. For backwards compatibility, here is the list of supported entries:

`Chist` A vector of historical catch, should be of length OM@nyears. If there are multiple fleets: a matrix of `OM@nyears` rows and `nfleet` columns. Ideally, the first year of the catch series represents unfished conditions (see also `C_eq`).

`C_sd` A vector or matrix of standard deviations (lognormal distribution) for the catches in `Chist`. If not provided, the default is 0.01. Only used if `condition = "catch"`.

`Ehist` A vector of historical effort, should be of length `OM@nyears` (see also `E_eq`).

`Index` A vector of values of an index (of length `OM@nyears`). If there are multiple indices: a matrix of historical indices of abundances, with rows indexing years and columns indexing the index.

`I_sd` A vector or matrix of standard deviations (lognormal distribution) for the indices corresponding to the entries in `Index`. If not provided, this function will use values from `OM@Iobs`.

`I_type` Obsolete as of version 2.0. See `s_selectivity` argument.

`CAA` Fishery age composition matrix with `nyears` rows and `OM@maxage+1` columns. If multiple fleets: an array with dimension: `nyears, OM@maxage, and nfleet`.

`CAL` Fishery length composition matrix with nyears rows and columns indexing the length bin. If multiple fleets: an array with dimension: `nyears, length bins, and nfleet`.

MS A vector of fishery mean size (MS, either mean length or mean weight) observations (length OM@nyears), or if multiple fleets: matrix of dimension: nyears, nfleet. Generally, mean lengths should not be used if CAL is also provided, unless mean length and length comps are independently sampled.

MS_type A character (either "length" (default) or "weight") to denote the type of mean size data.

MS_cv The coefficient of variation of the observed mean size. If there are multiple fleets, a vector of length nfleet. Default is 0.2.

s_CAA Survey age composition data, an array of dimension nyears, maxage+1, nsurvey.

s_CAL Survey length composition data, an array of dimension nyears, length(length_bin), nsurvey.

length_bin A vector for the midpoints of the length bins for CAL and s_CAL. All bin widths should be equal in size.

C_eq A numeric vector of length nfleet for the equilibrium catch for each fleet in Chist prior to the first year of the operating model. Zero (default) implies unfished conditions in year one. Otherwise, this is used to estimate depletion in the first year of the data. Alternatively, if one has a full CAA matrix, one could instead estimate "artificial" rec devs to generate the initial numbers-at-age (and hence initial depletion) in the first year of the model (see additional arguments).

C_eq_sd A vector of standard deviations (lognormal distribution) for the equilibrium catches in C_eq. If not provided, the default is 0.01. Only used if condition = "catch".

E_eq The equilibrium effort for each fleet in Ehist prior to the first year of the operating model. Zero (default) implies unfished conditions in year one. Otherwise, this is used to estimate depletion in the first year of the data.

abs_I Optional, an integer vector to indicate which indices are in absolute magnitude. Use 1 to set q = 1, otherwise use 0 to estimate q.

I_units Optional, an integer vector to indicate whether indices are biomass based (1) or abundance-based (0). By default, all are biomass-based.

age_error Optional, a square matrix of maxage + 1 rows and columns to specify ageing error. The aa-th column assigns a proportion of the true age in the a-th row to observed age. Thus, all rows should sum to 1. Default is an identity matrix (no ageing error).

sel_block Optional, for time-varying fleet selectivity (in time blocks), a integer matrix of nyears rows and nfleet columns to assigns a selectivity function to a fleet for certain years.

StockPars

When an operating model is provided, the RCM function will generally fit to each simulation of biological parameters.

Alternatively for a single fit to data independent of any operating model, provide a named list containing the following (naming conventions follow internal operating model variables):

- SRrel Integer, stock-recruit function (1 = Beverton-Holt, 2 = Ricker, 3 = Mesnil-Rochet hockey stick)
- R0 Numeric, starting value for unfished recruitment parameter
- M_ageArray Matrix [maxage+1, nyears] for natural mortality
- Len_age Matrix [maxage+1, nyears + 1] for length at age

- Linf Numeric. Asymptotic length. Only used for the upper bound for the size of full selectivity (if selectivity functions are length-based)
- LatASD Matrix [maxage+1, nyears + 1] for the standard deviation in length at age
- Wt_age Matrix [maxage+1, nyears + 1] for stock weight at age
- Mat_age Matrix [maxage+1, nyears + 1] for maturity at age
- Fec_Age Matrix [maxage+1, nyears + 1] for fecundity at age. Frequently the product of maturity and weight at age
- ageMarray Numeric, age of 50 percent maturity. Used to average the initial years for the unfished replacement line of the stock recruit relationship and steepness/R0. Irrelevant if fecundity and natural mortality are not time-varying (set to 1).
- spawn_time_frac Numeric, fraction of the year when spawning occurs
- hs Numeric, steepness of the stock recruit relationship
- procsd Numeric, lognormal recruitment deviation standard deviation

Additional arguments

For RCM, additional arguments can be passed to the model via ...:

plusgroup Logical for whether the maximum age is a plusgroup or not. By default, TRUE.

fix_dome Logical for whether the dome selectivity parameter for fleets is fixed. Used primarily for backwards compatibility, this is overridden by the map argument.

resample Logical, whether the OM conditioning parameters (recruitment, fishing mortality, SSB, selectivity, etc.) are obtained by sampling the Hessian matrix from a single model fit. By default FALSE. This feature requires identical biological parameters among simulations.

pbc_recdev Vector of length nyears. Proportion of the bias correction to apply annually to the recruitment deviations (if estimated). The bias correction from logspace to normal space is exp(log_rec_dev[y] - 0.5 * pbc_recdev[y] * sigmaR^2). Default proportion is 1.

pbc_earlyrecdev Vector of length maxage. Proportion of the bias correction to apply to the abundance deviations in the first year of the model (if estimated). The bias correction from logspace to normal space is exp(log_early_rec_dev[a] - 0.5 * pbc_recdev[a] * sigmaR^2). Default proportion is 1.

start

Starting values can be specified in a named list for the following:

vul_par A matrix of 3 rows and nfleet columns for starting values for fleet selectivity. The three rows correspond to LFS (length of full selectivity), L5 (length of 5 percent selectivity), and Vmaxlen (selectivity at length Linf). By default, the starting values are values from the OM object. If any selectivity = "free", then this matrix needs to be of maxage+1 rows where the row specifies the selectivity at age. See Articles section.

ivul_par A matrix of 3 rows and nsurvey columns for starting values for fleet selectivity. Same setup as vul_par. Values in the column are ignored if s_selectivity is mapped to a fishing fleet (add NA placeholders in that case). If any s_selectivity = "free", then this matrix needs to be of maxage+1 rows where the row specifies the selectivity at age.

log_rec_dev A numeric vector of length nyears for the starting values of the log-recruitment deviations.

log_early_rec_dev A numeric vector of length OM@maxage for the starting values of the recruitment deviations controlling the abundance-at-age in the first year of the model.

q A numeric vector of length nsurvey for index catchability. See online article for more information.

map

Parameters can be fixed with the map argument (also a named list, corresponding to the start list). Each vector or matrix in the map argument will be the same dimension as in the start entry. If an entry is NA, the corresponding parameter is fixed in the model to the starting value. Otherwise, an integer for each independent parameter, i.e., shared or mirrored parameters get the same integer entry.

vul_par An integer matrix of the same dimension as start$vul_par. By default, selectivity is fixed if there are no age or length composition for that fleet or survey, otherwise estimated. Unused cells in the start$vul_par matrix should be given NA in the map matrix.

ivul_par The map argument for the survey selectivity parameters (same dimension as start$ivul_par). Placeholder parameters should have a map value of NA.

log_early_rec_dev A vector of length OM@maxage that indexes which recruitment deviates for the cohorts in the first year of the model are fixed (using NA) or estimated (a separate integer). By default, no deviates are estimated (all are NA).

log_rec_dev A vector of length OM@nyears that indexes which recruitment deviates are fixed (using NA) or estimated (a separate integer). By default, all these deviates are estimated.

q A vector of length nsurvey for index catchability. q should be an estimated parameter when sharing across surveys (perhaps with differing selectivity). Otherwise, it is solved analytically where individual parameters are independent of other indices. Use RCMdata@abs_I for fixing the catchability to 1. See online article for more information.

Likelihood weights

LWT is an optional named list containing the likelihood weights (values >= 0) with the possible options:

- Chist, CAA, CAL, MS, C_eq: A vector of length nfleet for each.
- Index, IAA, IAL: A vector of length nsurvey for each.

By default, all likelihood weights are equal to one if not specified by the user.

Annual multinomial sample sizes for the age and length comps can now be provided directly in the RCMdata object. For a list or MSEtool::Data object, use the ESS argument.

Author(s)

Q. Huynh

References

Thorson et al. 2017. Model-based estimates of effective sample size in stock assessment models using the Dirichlet-multinomial distribution. Fish. Res. 192:84-93. doi:10.1016/j.fishres.2016.06.005

See Also

plot.RCModel RCModel compare_RCM pcod RCM2MOM posterior

Examples

```
# An example that conditions a Pacific cod operating model. There are 48 simulations,
# where values of natural mortality and steepness are sampled from distributions.
# The model is fitted with priors on the index catchability. Maturity and selectivity
# are knife-edge at the age of 2 years. See online tutorial for more information.

data(pcod)
mat_ogive <- pcod$OM@cpars$Mat_age[1, , 1]
out <- RCM(OM = pcod$OM, data = pcod$data,
           condition = "catch", mean_fit = TRUE,
           selectivity = "free", s_selectivity = rep("SSB", ncol(pcod$data@Index)),
           start = list(vul_par = matrix(mat_ogive, length(mat_ogive), 1)),
           map = list(vul_par = matrix(NA, length(mat_ogive), 1),
                      log_early_rec_dev = rep(1, pcod$OM@maxage)),
           prior = pcod$prior)
plot(out, s_name = colnames(pcod$data@Index))

# Alternative OM with age-3 maturity and selectivity instead.
out_age3 <- local({
  pcod$OM@cpars$Mat_age[, 2, ] <- 0
  mat_ogive_age3 <- pcod$OM@cpars$Mat_age[1, , 1]
  RCM(OM = pcod$OM, data = pcod$data,
      condition = "catch", mean_fit = TRUE,
      selectivity = "free", s_selectivity = rep("SSB", ncol(pcod$data@Index)),
      start = list(vul_par = matrix(mat_ogive_age3, length(mat_ogive_age3), 1)),
      map = list(vul_par = matrix(NA, length(mat_ogive_age3), 1),
                 log_early_rec_dev = rep(1, pcod$OM@maxage)),
      prior = pcod$prior)
})

compare_RCM(out, out_age3, scenario = list(names = c("Age-2 maturity", "Age-3 maturity")),
            s_name = colnames(pcod$data@Index))

Hist <- runMSE(out@OM, Hist = TRUE)
```

`compare_models` *Compare output from several assessment models*

Description

Plot biomass, recruitment, and fishing mortality time series from several . This function can be used to compare outputs among different assessment models from the same Data object.

Usage

```
compare_models(..., label = NULL, color = NULL)
```

Arguments

...	Objects of class Assessment.
label	A character vector of the models for the legend.
color	A vector of colors for each assessment model.

Value

A set of figures of biomass, recruitment, and fishing mortality estimates among the models.

Author(s)

Q. Huynh

Examples

```
res <- cDD_SS(x = 3, Data = MSEtool::SimulatedData)
res2 <- SCA(x = 3, Data = MSEtool::SimulatedData)
res3 <- SP(x = 3, Data = MSEtool::SimulatedData)

compare_models(res, res2, res3)
```

DD_TMB *Delay - Difference Stock Assessment in TMB*

Description

A simple delay-difference assessment model using a time-series of catches and a relative abundance index and coded in TMB. The model can be conditioned on either (1) effort and estimates predicted catch or (2) catch and estimates a predicted index. In the state-space version DD_SS, recruitment deviations from the stock-recruit relationship are estimated.

Usage

```
DD_TMB(
  x = 1,
  Data,
  condition = c("catch", "effort"),
  AddInd = "B",
  SR = c("BH", "Ricker"),
  rescale = "mean1",
  MW = FALSE,
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  dep = 1,
  LWT = list(),
  n_itF = 3L,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 5000, eval.max = 10000),
  ...
)

DD_SS(
  x = 1,
  Data,
  condition = c("catch", "effort"),
  AddInd = "B",
  SR = c("BH", "Ricker"),
  rescale = "mean1",
  MW = FALSE,
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  fix_sd = FALSE,
  fix_tau = TRUE,
  dep = 1,
  LWT = list(),
  n_itF = 3L,
  integrate = FALSE,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 5000, eval.max = 10000),
  inner.control = list(),
  ...
)
```

Arguments

x	An index for the objects in Data when running in closed loop simulation. Otherwise, equals to 1 when running an assessment.
Data	An object of class MSEtool::Data.
condition	A string to indicate whether to condition the model on catch or effort (ratio of catch and index).
AddInd	A vector of integers or character strings indicating the indices to be used in the model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd.
SR	Stock-recruit function (either "BH" for Beverton-Holt or "Ricker").
rescale	A multiplicative factor that rescales the catch in the assessment model, which can improve convergence. By default, "mean1" scales the catch so that time series mean is 1, otherwise a numeric. Output is re-converted back to original units.
MW	Logical, whether to fit to mean weight. In closed-loop simulation, mean weight will be grabbed from Data@Misc[[x]]$MW, otherwise calculated from Data@CAL.
start	Optional list of starting values. Entries can be expressions that are evaluated in the function. See details.
prior	A named list for the parameters of any priors to be added to the model. See below.
fix_h	Logical, whether to fix steepness to value in Data@steep in the assessment model. Automatically false if a prior is used.
dep	The initial depletion in the first year of the model. A tight prior is placed on the model objective function to estimate the equilibrium fishing mortality rate that corresponds to the initial depletion. Due to this tight prior, this F should not be considered to be an independent model parameter. Set to zero to eliminate this prior.
LWT	A named list of likelihood weights. For LWT$Index, a vector of likelihood weights for each survey, while for LWT$MW a numeric.
n_itF	Integer, the number of iterations to solve F within an annual time step when conditioning on catch.
silent	Logical, passed to TMB::MakeADFun(), whether TMB will print trace information during optimization. Used for diagnostics for model convergence.
opt_hess	Logical, whether the hessian function will be passed to stats::nlminb() during optimization (this generally reduces the number of iterations to convergence, but is memory and time intensive and does not guarantee an increase in convergence rate). Ignored if integrate = TRUE.
n_restart	The number of restarts (calls to stats::nlminb()) in the optimization procedure, so long as the model hasn't converged. The optimization continues from the parameters from the previous (re)start.
control	A named list of parameters regarding optimization to be passed to stats::nlminb().
...	Additional arguments (not currently used).

fix_sd	Logical, whether the standard deviation of the data in the likelihood (index for conditioning on catch or catch for conditioning on effort). If TRUE, the SD is fixed to value provided in start (if provided), otherwise, value based on either Data@CV_Cat or Data@CV_Ind.
fix_tau	Logical, the standard deviation of the recruitment deviations is fixed. If TRUE, tau is fixed to value provided in start (if provided), otherwise, equal to 1.
integrate	Logical, whether the likelihood of the model integrates over the likelihood of the recruitment deviations (thus, treating it as a random effects/state-space variable). Otherwise, recruitment deviations are penalized parameters.
inner.control	A named list of arguments for optimization of the random effects, which is passed on to TMB::newton() via TMB::MakeADFun().

Details

For start (optional), a named list of starting values of estimates can be provided for:

- R0 Unfished recruitment. Otherwise, Data@OM$R0[x] is used in closed-loop, and 400% of mean catch otherwise.
- h Steepness. Otherwise, Data@steep[x] is used, or 0.9 if empty.
- M Natural mortality. Otherwise, Data@Mort[x] is used.
- k Age of knife-edge maturity. By default, the age of 50% maturity calculated from the slots in the Data object.
- Rho Delay-difference rho parameter. Otherwise, calculated from biological parameters in the Data object.
- Alpha Delay-difference alpha parameter. Otherwise, calculated from biological parameters in the Data object.
- q_effort Scalar coefficient when conditioning on effort (to scale to F). Otherwise, 1 is the default.
- F_equilibrium Equilibrium fishing mortality rate leading into first year of the model (to determine initial depletion). By default, 0.
- omega Lognormal SD of the catch (observation error) when conditioning on effort. By default, Data@CV_Cat[x].
- tau Lognormal SD of the recruitment deviations (process error) for DD_SS. By default, Data@sigmaR[x].
- sigma Lognormal SD of the index (observation error) when conditioning on catch. By default, Data@CV_Ind[x]. Not used if multiple indices are used.
- sigma_W Lognormal SD of the mean weight (observation error). By default, 0.1.

Multiple indices are supported in the model. Data@Ind, Data@VInd, and Data@SpInd are all assumed to be biomass-based. For Data@AddInd, Data@I_units are used to identify a biomass vs. abundance-based index.

Similar to many other assessment models, the model depends on assumptions such as stationary productivity and proportionality between the abundance index and real abundance. Unsurprisingly the extent to which these assumptions are violated tends to be the biggest driver of performance for this method.

Value

An object of Assessment containing objects and output from TMB.

Priors

The following priors can be added as a named list, e.g., `prior = list(M = c(0.25, 0.15), h = c(0.7, 0.1)`. For each parameter below, provide a vector of values as described:

- `R0` - A vector of length 3. The first value indicates the distribution of the prior: `1` for lognormal, `2` for uniform on `log(R0)`, `3` for uniform on R0. If lognormal, the second and third values are the prior mean (in normal space) and SD (in log space). Otherwise, the second and third values are the lower and upper bounds of the uniform distribution (values in normal space).
- `h` - A vector of length 2 for the prior mean and SD, both in normal space. Beverton-Holt steepness uses a beta distribution, while Ricker steepness uses a normal distribution.
- `M` - A vector of length 2 for the prior mean (in normal space) and SD (in log space). Lognormal prior.
- `q` - A matrix for nsurvey rows and 2 columns. The first column is the prior mean (in normal space) and the second column for the SD (in log space). Use `NA` in rows corresponding to indices without priors.

See online documentation for more details.

Online Documentation

Model description and equations are available on the openMSE website.

Required Data

- `DD_TMB`: Cat, Ind, Mort, L50, vbK, vbLinf, vbt0, wla, wlb, MaxAge
- `DD_SS`: Cat, Ind, Mort, L50, vbK, vbLinf, vbt0, wla, wlb, MaxAge

Optional Data

- `DD_TMB`: steep
- `DD_SS`: steep, CV_Cat

Author(s)

T. Carruthers & Z. Siders. Zach Siders coded the TMB function.

References

Carruthers, T, Walters, C.J,, and McAllister, M.K. 2012. Evaluating methods that classify fisheries stock status using only fisheries catch data. Fisheries Research 119-120:66-79.

Hilborn, R., and Walters, C., 1992. Quantitative Fisheries Stock Assessment: Choice, Dynamics and Uncertainty. Chapman and Hall, New York.

See Also

plot.Assessment summary.Assessment retrospective profile make_MP

Examples

```
#### Observation-error delay difference model
res <- DD_TMB(x = 3, Data = MSEtool::SimulatedData)

# Provide starting values
start <- list(h = 0.95)
res <- DD_TMB(x = 3, Data = MSEtool::SimulatedData, start = start)

summary(res@SD) # Parameter estimates

### State-space version
### Set recruitment variability SD = 0.3 (since fix_tau = TRUE)
res <- DD_SS(x = 3, Data = MSEtool::SimulatedData, start = list(tau = 0.3))
```

`diagnostic` *Diagnostic of assessments in MSE: did Assess models converge during MSE?*

Description

Diagnostic check for convergence of Assess models during closed-loop simulation. Use when the MP was created with make_MP with argument `diagnostic = "min"` or `"full"`. This function summarizes and plots the diagnostic information.

Usage

```
diagnostic(MSE, MP, gradient_threshold = 0.1, figure = TRUE)

diagnostic_AM(...)
```

Arguments

`MSE`	An object of class MSE created by `MSEtool::runMSE()`.
`MP`	Optional, a character vector of MPs that use assessment models.
`gradient_threshold`	The maximum magnitude (absolute value) desired for the gradient of the likelihood.
`figure`	Logical, whether a figure will be drawn.
`...`	Arguments to pass to `diagnostic`.

Value

A matrix with diagnostic performance of assessment models in the MSE. If `figure = TRUE`, a set of figures: traffic light (red/green) plots indicating whether the model converged (defined if a positive-definite Hessian matrix was obtained), the optimizer reached pre-specified iteration limits (as passed to `stats::nlminb()`), and the maximum gradient of the likelihood in each assessment run. Also includes the number of optimization iterations function evaluations reported by `stats::nlminb()` for each application of the assessment model.

Author(s)

Q. Huynh

See Also

retrospective_AM

Examples

```
OM <- MSEtool::testOM; OM@proyears <- 20
myMSE <- runMSE(OM, MPs = "SCA_4010")
diagnostic(myMSE)

# How to get all the reporting
library(dplyr)
conv_statistics <- lapply(1:myMSE@nMPs, function(m) {
  lapply(1:myMSE@nsim, function(x) {
    myMSE@PPD[[m]]@Misc[[x]]$diagnostic %>%
      mutate(MP = myMSE@MPs[m], Simulation = x)
 }) %>% bind_rows()
}) %>% bind_rows()
```

---

getinds *Characterize posterior predictive data*

Description

Characterize posterior predictive data

Usage

```
getinds(
  PPD,
  styr,
  res = 6,
  tsd = c("Cat", "Cat", "Cat", "Ind", "ML"),
  stat = c("slp", "AAV", "mu", "slp", "slp")
)
```

Arguments

PPD	An object of class Data stored in the Misc slot of an MSE object following a call of runMSE(PPD = TRUE).
styr	Positive integer, the starting year for calculation of quantities
res	Positive integer, the temporal resolution (chunks - normally years) over which to calculate quantities
tsd	Character vector of names of types of data: Cat = catch, Ind = relative abundance index, ML = mean length in catches
stat	Character vector of types of quantity to be calculated: slp = slope(log(x)), AAV = average annual variability, mu = mean(log(x))

Value

A 3D array of results (type of data/stat (e.g. mean catches),time period (chunk), simulation)

Author(s)

T. Carruthers

References

Carruthers and Hordyk 2018

HCRlin *Generic linear harvest control rule based on biomass*

Description

A general function used by HCR_ramp that adjusts the output (e.g., F) by a linear ramp based on the value of the OCP relative to target and limit values.

Usage

```
HCRlin(OCP_val, LOCP, TOCP, relF_min = 0, relF_max = 1)
```

Arguments

OCP_val	The value of the operational control point (OCP).
LOCP	Numeric, the limit value for the OCP in the HCR.
TOCP	Numeric, the target value for the OCP in the HCR.
relF_min	The relative maximum value (e.g. a multiple of FMSY) if OCP < LOCP.
relF_max	The relative maximum value (e.g. a multiple of FMSY) if OCP > TOCP.

Value

Numeric adjustment factor.

Author(s)

T. Carruthers

Examples

```
#40-10 linear ramp
Brel <- seq(0, 1, length.out = 200)
plot(Brel, HCRlin(Brel, 0.1, 0.4), xlab = "Estimated B/B0", ylab = "Relative change in F",
     main = "A 40-10 harvest control rule", type = 'l', col = 'blue')
abline(v = c(0.1,0.4), col = 'red', lty = 2)
```

HCR_escapement *Fixed escapement harvest control rule*

Description

A simple control rule that allows fishing when the operational control point (OCP) is above some threshold. By default, this function sets the TAC at F = 100% FMSY when spawning depletion > 0.1.

Usage

```
HCR_escapement(
  Assessment,
  reps = 1,
  OCP_type = "SSB_SSB0",
  OCP_threshold = 0.2,
  Ftarget_type = "FMSY",
  relF_max = 1,
  ...
)
```

Arguments

Assessment	An object of class Assessment with estimates of FMSY or UMSY and vulnerable biomass in terminal year.
reps	The number of stochastic samples of the TAC recommendation.
OCP_type	The type of operational control points (OCPs) for the harvest control rule used to determine whether there is fishing. By default, use ("SSB_SSB0" for spawning depletion. Other biomass OCPs include "SSB_SSBMSY" for spawning biomass relative to MSY and "SSB_dSSB0", for dynamic depletion (dynamic SSB0 is the historical reconstructed biomass with F = 0). For F-based OCPs, the terminal year fishing mortality relative F01 or Fmax (using yield-per-recruit) or F-SPR% (see SPR_OCP argument) can be used.
OCP_threshold	The value of the OCP above which fishing can occur.
Ftarget_type	The type of F used for the target fishing mortality rate.
relF_max	The relative value of Ftarget if OCP > OCP_treshold.
...	Miscellaneous arguments.

Details

The catch advice is calculated using the catch equation of the corresponding assessment. See `Assessment@forecast$catch_eq`, a function that returns the catch advice for a specified `Ftarget`.

Value

An object of class MSEtool::Rec with the TAC recommendation.

Author(s)

Q. Huynh

References

Deroba, J.J. and Bence, J.R. 2008. A review of harvest policies: Understanding relative performance of control rules. Fisheries Research 94:210-223.

See Also

make_MP HCR_ramp

Examples

```
# create an MP to run in closed-loop MSE (fishes at FMSY when B/B0 > 0.2)
SP_escapement <- make_MP(SP, HCR_escapement)

# The MP which fishes at 75% of FMSY
SP_escapement75 <- make_MP(SP, HCR_escapement, relF_max = 0.75)

# The MP which fishes at FMSY when BMSY > 0.5
SP_BMSY_escapement <- make_MP(SP, HCR_escapement, OCP_type = "SSB_SSBMSY",
                              OCP_threshold = 0.5, relF_max = 1)


myOM <- MSEtool::runMSE(MSEtool::testOM, MPs = c("FMSYref", "SP_escapement", "SP_BMSY_escapement"))
```

HCR_FB *A Harvest Control Rule using B/BMSY and F/FMSY to adjust TAC or TAE.*

Description

A Harvest Control Rule using B/BMSY and F/FMSY to adjust TAC or TAE.

Usage

```
HCR_FB(Brel, Frel, Bpow = 2, Bgrad = 1, Fpow = 1, Fgrad = 1)
```

Arguments

Brel	improper fraction: an estimate of Biomass relative to BMSY
Frel	improper fraction: an estimate of Fishing mortality rate relative to FMSY
Bpow	non-negative real number: controls the shape of the biomass adjustment, when zero there is no adjustment
Bgrad	non-negative real number: controls the gradient of the biomass adjustment
Fpow	non-negative real number: controls the adjustment speed relative to F/FMSY. When set to 1, next recommendation is FMSY. When less than 1 next recommendation is between current F and FMSY.
Fgrad	improper fraction: target Fishing rate relative to FMSY

Value

a TAC or TAE adjustment factor.

Author(s)

T. Carruthers

References

Made up for this package

Examples

```
res <- 100
Frel <- seq(1/2, 2, length.out = res)
Brel <- seq(0.05, 2, length.out=res)
adj <- array(HCR_FB(Brel[rep(1:res, res)], Frel[rep(1:res, each = res)],
                    Bpow = 2, Bgrad = 1, Fpow = 1, Fgrad = 0.75), c(res, res))
contour(Brel, Frel, adj, nlevels = 20, xlab = "B/BMSY", ylab = "F/FMSY",
        main = "FBsurface TAC adjustment factor")
abline(h = 1, col = 'red', lty = 2)
abline(v = 1, col = 'red', lty = 2)
legend('topright', c("Bpow = 2", "Bgrad = 1", "Fpow = 1", "Fgrad = 0.75"), text.col = 'blue')
```

HCR_fixedF *Simple fixed F harvest control rule*

Description

A simple control rule that explicitly specifies the target apical F independent of any model.

Usage

```
HCR_fixedF(Assessment, reps = 1, Ftarget = 0.1)
```

Arguments

Assessment	An object of class Assessment with estimates of next year's abundance or biomass.
reps	The number of replicates of the TAC recommendation (not used).
Ftarget	The value of F.

Details

The catch advice is calculated using the catch equation of the corresponding assessment. See Assessment@forecast$catch_eq, a function that returns the catch advice for a specified Ftarget.

Value

An object of class MSEtool::Rec with the TAC recommendation.

Author(s)

Q. Huynh

See Also

make_MP HCR_ramp#'

Examples

```
# create an MP to run in closed-loop MSE (fishes at F = 0.2)
F0.2 <- make_MP(SP, HCR_fixedF, Ftarget = 0.2)


myOM <- MSEtool::runMSE(MSEtool::testOM, MPs = c("FMSYref", "F0.2"))
```

HCR_MSY — *Harvest control rule to fish at some fraction of maximum sustainable yield*

Description

A simple control rule that specifies the total allowable catch (TAC) as a function of the abundance of the first projection year and some fraction of FMSY/UMSY.

Usage

```
HCR_MSY(Assessment, reps = 1, MSY_frac = 1, ...)
```

Arguments

Assessment	An object of class Assessment with estimates of FMSY or UMSY and vulnerable biomass in terminal year.
reps	The number of stochastic samples of the TAC recommendation.
MSY_frac	The fraction of FMSY or UMSY for calculating the TAC (e.g. MSY_frac = 0.75 fishes at 75% of FMSY).
...	Miscellaneous arguments.

Details

The catch advice is calculated using the catch equation of the corresponding assessment. See Assessment@forecast$catch_eq, a function that returns the catch advice for a specified Ftarget.

Value

An object of class MSEtool::Rec with the TAC recommendation.

Author(s)

Q. Huynh

References

Punt, A. E, Dorn, M. W., and Haltuch, M. A. 2008. Evaluation of threshold management strategies for groundfish off the U.S. West Coast. Fisheries Research 94:251-266.

See Also

make_MP HCR_ramp

Examples

```
# create an MP to run in closed-loop MSE (fishes at UMSY)
SPMSY <- make_MP(SP, HCR_MSY)

# The MP which fishes at 75% of FMSY
SP75MSY <- make_MP(SP, HCR_MSY, MSY_frac = 0.75)


myOM <- MSEtool::runMSE(MSEtool::testOM, MPs = c("FMSYref", "SPMSY", "SP75MSY"))
```

HCR_ramp *Linearly ramped harvest control rules*

Description

An output control rule with a ramp that reduces the target F (used for the TAC recommendation) linearly as a function of an operational control point (OCP) such as spawning depletion or spawning biomass. The reduction in F is linear when the OCP is between the target OCP (TOCP) and the limit OCP (LOCP). The target F is maximized at or above the TOCP. Below the LOCP, the target F is minimized. For example, the TOCP and LOCP for 40% and 10% spawning depletion, respectively, in the 40-10 control rule. Ftarget is FMSY above the TOCP and zero below the LOCP. This type of control rule can generalized with more control points (>2) in HCR_segment. Class HCR objects are typically used with function make_MP.

Usage

```
HCR_ramp(
  Assessment,
  reps = 1,
 OCP_type = c("SSB_SSB0", "SSB_SSBMSY", "SSB_dSSB0", "F_FMSY", "F_F01", "F_FSPR"),
  Ftarget_type = c("FMSY", "F01", "Fmax", "FSPR", "abs"),
  LOCP = 0.1,
  TOCP = 0.4,
  relF_min = 0,
  relF_max = 1,
  SPR_OCP = 0.4,
  SPR_targ = 0.4,
  ...
)

HCR40_10(Assessment, reps = 1, Ftarget_type = "FMSY", SPR_targ = 0.4, ...)

HCR60_20(Assessment, reps = 1, Ftarget_type = "FMSY", SPR_targ = 0.4, ...)

HCR80_40MSY(Assessment, reps = 1, Ftarget_type = "FMSY", SPR_targ = 0.4, ...)
```

Arguments

Assessment	An object of class Assessment with estimates of FMSY or UMSY, vulnerable biomass, and spawning biomass depletion in terminal year.
reps	The number of stochastic samples of the TAC recommendation.
OCP_type	The type of operational control points (OCPs) for the harvest control rule used to determine the reduction in F. See below.
Ftarget_type	The type of F used for the target fishing mortality rate. See below.
LOCP	Numeric, the limit value for the OCP in the HCR.

TOCP	Numeric, the target value for the OCP in the HCR.
relF_min	The relative value of Ftarget (i.e., as a proportion) if OCP < LOCP.
relF_max	The relative value of Ftarget if OCP > TOCP.
SPR_OCP	The value of spawning potential ratio for the OCP if OCP_type = "F_FSPR". By default, 0.4 (F40%).
SPR_targ	The target value of spawning potential ratio if Ftarget_type = "FSPR". By default, 0.4 (F40%).
...	Miscellaneous arguments.

Details

The catch advice is calculated using the catch equation of the corresponding assessment. See Assessment@forecast$catch_eq, a function that returns the catch advice for a specified Ftarget.

Operational control points (OCP_type)

The following are the available options for harvest control rule inputs, and the source of those values in the Assessment object:

- **Default** "SSB_SSB0": Spawning depletion. Uses the last value in Assessment@SSB_SSB0 vector.
- "SSB_SSBMSY": Spawning biomass relative to MSY. Uses the last value in Assessment@SSB_SSBMSY vector.
- "SSB_dSSB0": Dynamic depletion (SSB relative to the historical reconstructed biomass with F = 0). Uses the last value in Assessment@SSB/Assessment@TMB_report$dynamic_SSB0.
- "F_FMSY": Fishing mortality relative to MSY. Uses the last value in Assessment@F_FMSY.
- "F_F01": Fishing mortality relative to F_0.1 (yield per recruit), calculated from the data frame in Assessment@forecast[["per_recruit"]].
- "F_FSPR": Fishing mortality relative to F_SPR% (the F that produces the spawning potential ratio specified in "SPR_OCP", calculated from the data frame in Assessment@forecast[["per_recruit"]].

Fishing mortality target (Ftarget_type)

The type of F for which the corresponding catch is calculated in the HCR is specified here. The source of those values in the Assessment object is specified:

- **Default** "FMSY": Fishing mortality relative to MSY. Uses the value in Assessment@FMSY.
- "F01": Fishing mortality relative to F_0.1 (yield per recruit), calculated from the data frame in Assessment@forecast[["per_recruit"]].
- "Fmax": Fishing mortality relative to F_max (maximizing yield per recruit), calculated from the data frame in Assessment@forecast[["per_recruit"]].
- "FSPR": Fishing mortality relative to F_SPR% (the F that produces the spawning potential ratio specified in "SPR_targ", calculated from data frame in Assessment@forecast[["per_recruit"]].
- "abs": Fishing mortality is independent of any model output and is explicitly specified in relF.

Value

An object of class MSEtool::Rec with the TAC recommendation.

Functions

- `HCR_ramp()`: Generic ramped-HCR function where user specifies OCP and corresponding limit and target points, as well as minimum and maximum relative F target.
- `HCR40_10()`: Common U.S. west coast control rule (LOCP and TOCP of 0.1 and 0.4 spawning depletion, respectively)
- `HCR60_20()`: More conservative than `HCR40_10`, with LOCP and TOCP of 0.2 and 0.6 spawning depletion, respectively).
- `HCR80_40MSY()`: 0.8 and 0.4 SSBMSY as the LOCP and TOCP, respectively.

Author(s)

Q. Huynh & T. Carruthers

References

Deroba, J.J. and Bence, J.R. 2008. A review of harvest policies: Understanding relative performance of control rules. Fisheries Research 94:210-223.

Edwards, C.T.T. and Dankel, D.J. (eds.). 2016. Management Science in Fisheries: an introduction to simulation methods. Routledge, New York, NY. 460 pp.

Punt, A. E, Dorn, M. W., and Haltuch, M. A. 2008. Evaluation of threshold management strategies for groundfish off the U.S. West Coast. Fisheries Research 94:251-266.

Restrepo, V.R. and Power, J.E. 1999. Precautionary control rules in US fisheries management: specification and performance. ICES Journal of Marine Science 56:846-852.

See Also

HCR_segment HCR_MSY HCRlin make_MP

Examples

```
# 40-10 linear ramp
Brel <- seq(0, 1, length.out = 200)
plot(Brel, HCRlin(Brel, 0.1, 0.4),
    xlab = expression("Operational control point: Estimated"~SSB/SSB[0]),
    ylab = expression(F[target]~~": proportion of"~~F[MSY]),
    main = "40-10 harvest control rule", type = "l")
abline(v = c(0.1, 0.4), col = "red", lty = 2)

# create a 40-10 MP to run in closed-loop MSE
DD_40_10 <- make_MP(DD_TMB, HCR40_10)

# Alternatively,
DD_40_10 <- make_MP(DD_TMB, HCR_ramp, OCP_type = "SSB_SSB0", LOCP = 0.1, TOCP = 0.4)
```

```
# An SCA with LOCP and TOCP at 0.4 and 0.8, respectively, of SSB/SSBMSY
SCA_80_40 <- make_MP(SCA, HCR_ramp, OCP_type = "SSB_SSBMSY", LOCP = 0.4, TOCP = 0.8)

# A conservative HCR that fishes at 75% of FMSY at B > 80% BMSY but only reduces F
# to 10% of FMSY if B < 40% BMSY.
SCA_conservative <- make_MP(SCA, HCR_ramp, OCP_type = "SSB_SSBMSY", LOCP = 0.4, TOCP = 0.8,
relF_min = 0.1, relF_max = 0.75)

# Figure of this conservative HCR
Brel <- seq(0, 1, length.out = 200)
Frel <- HCRlin(Brel, 0.4, 0.8, 0.1, 0.75)
plot(Brel, Frel,
    xlab = expression("Operational control point: Estimated"~SSB/SSB[MSY]),
    ylab = expression(F[target]~":"~~F/F[MSY]),
    ylim = c(0, 1), type = "l")
abline(v = c(0.4, 0.8), col = "red", lty = 2)

# A harvest control rule as a function of BMSY, with F independent of model output,
# i.e., specify F in relF argument (here maximum F of 0.1)
SCA_80_40 <- make_MP(SCA, HCR_ramp, OCP_type = "SSB_SSBMSY", LOCP = 0.4, TOCP = 0.8,
                     relF_min = 0, relF_max = 0.1)
```

`HCR_segment` *Segmented harvest control rules*

Description

A linear segmented output control rule where the target F (used for the TAC recommendation) is a function of an operational control point (OCP) such as spawning depletion or spawning biomass. The segments of the HCR are specified by arguments `OCP` and `relF`. Beyond the range of `OCP`, the response will be flat. HCR_ramp uses `HCR_segment` with two control points.

Usage

```
HCR_segment(
  Assessment,
  reps = 1,
 OCP_type = c("SSB_SSB0", "SSB_SSBMSY", "SSB_dSSB0", "F_FMSY", "F_F01", "F_FSPR"),
  Ftarget_type = c("FMSY", "F01", "Fmax", "FSPR", "abs"),
  OCP = c(0.1, 0.4),
  relF = c(0, 1),
  SPR_OCP,
  SPR_targ,
  ...
)
```

Arguments

Assessment	An object of class Assessment with estimates of FMSY or UMSY, vulnerable biomass, and spawning biomass depletion in terminal year.
reps	The number of stochastic samples of the TAC recommendation.
OCP_type	The type of operational control points (OCPs) for the harvest control rule used to determine the reduction in F. See below.
Ftarget_type	The type of F used for the target fishing mortality rate. See below.
OCP	Numeric vector of operational control points for the HCR (in increasing order).
relF	Numeric vector of Ftarget corresponding to the values in OCP.
SPR_OCP	The value of spawning potential ratio for the OCP if OCP_type = "F_FSPR". By default, 0.4 (F40%).
SPR_targ	The target value of spawning potential ratio if Ftarget_type = "FSPR". By default, 0.4 (F40%).
...	Miscellaneous arguments.

Details

The catch advice is calculated using the catch equation of the corresponding assessment. See Assessment@forecast$catch_eq, a function that returns the catch advice for a specified Ftarget.

Operational control points (OCP_type)

The following are the available options for harvest control rule inputs, and the source of those values in the Assessment object:

- **Default** "SSB_SSB0": Spawning depletion. Uses the last value in Assessment@SSB_SSB0 vector.
- "SSB_SSBMSY": Spawning biomass relative to MSY. Uses the last value in Assessment@SSB_SSBMSY vector.
- "SSB_dSSB0": Dynamic depletion (SSB relative to the historical reconstructed biomass with F = 0). Uses the last value in Assessment@SSB/Assessment@TMB_report$dynamic_SSB0.
- "F_FMSY": Fishing mortality relative to MSY. Uses the last value in Assessment@F_FMSY.
- "F_F01": Fishing mortality relative to F_0.1 (yield per recruit), calculated from the data frame in Assessment@forecast[["per_recruit"]].
- "F_FSPR": Fishing mortality relative to F_SPR% (the F that produces the spawning potential ratio specified in "SPR_OCP", calculated from the data frame in Assessment@forecast[["per_recruit"]].

Fishing mortality target (Ftarget_type)

The type of F for which the corresponding catch is calculated in the HCR is specified here. The source of those values in the Assessment object is specified:

- **Default** "FMSY": Fishing mortality relative to MSY. Uses the value in Assessment@FMSY.
- "F01": Fishing mortality relative to F_0.1 (yield per recruit), calculated from the data frame in Assessment@forecast[["per_recruit"]].
- "Fmax": Fishing mortality relative to F_max (maximizing yield per recruit), calculated from the data frame in Assessment@forecast[["per_recruit"]].

- "FSPR": Fishing mortality relative to F_SPR% (the F that produces the spawning potential ratio specified in "SPR_targ", calculated from data frame in Assessment@forecast[["per_recruit"]].
- "abs": Fishing mortality is independent of any model output and is explicitly specified in relF.

Value

An object of class MSEtool::Rec with the TAC recommendation.

Author(s)

Q. Huynh

Examples

```
# This is an MP with a 40-10 harvest control rule (using FMSY)
DD_40_10 <- make_MP(DD_TMB, HCR_segment, OCP_type = "SSB_SSB0", OCP = c(0.1, 0.4), relF = c(0, 1))
#'
# This is an MP with a 40-10 harvest control rule with a maximum F of 0.1
DD_40_10 <- make_MP(DD_TMB, HCR_segment, OCP_type = "SSB_SSB0",
                    Ftarget_type = "abs", OCP = c(0.1, 0.4), relF = c(0, 0.1))
```

---

mahplot *Plot statistical power of the indicator with increasing time blocks*

Description

Plot statistical power of the indicator with increasing time blocks

Usage

```
mahplot(outlist, res = 6, maxups = 5, MPs)
```

Arguments

outlist	A list object produced by the function PRBcalc
res	Integer, the resolution (time blocking) for the calculation of PPD
maxups	Integer, the maximum number of update time blocks to plot
MPs	Character vector of MP names

Value

Density plots of Mahalanobis distance.

Author(s)

T. Carruthers

References

Carruthers and Hordyk 2018

`make_interim_MP` *Make a custom management procedure (MP)*

Description

Function operator that creates a management procedure (MP) by combining an assessment model (function of class `Assess`) with a harvest control rule (function of class `HCR`). The resulting function can then be tested in closed-loop simulation via `MSEtool::runMSE()`.

- Use `make_MP` to specify constant TAC between assessments; the frequency of assessments is specified in `OM@interval`.
- Use `make_projection_MP` to set catches according to a schedule set by projections, specify assessment frequency in argument `assessment_interval` and ensure that `OM@interval <- 1`.
- Use `make_interim_MP` to use an interim procedure to adjust the TAC between assessments using an index (Huynh et al. 2020), with the frequency of assessments specified in argument `assessment_interval` when making the MP; ensure that `OM@interval <- 1`.

Usage

```
make_interim_MP(
  .Assess = "SCA",
  .HCR = "HCR_MSY",
  AddInd = "VB",
  assessment_interval = 5,
  type = c("buffer", "mean", "loess", "none"),
  type_par = NULL,
  diagnostic = c("min", "full", "none"),
  ...
)

make_projection_MP(
  .Assess = "SCA",
  .HCR = "HCR_MSY",
  assessment_interval = 5,
  Ftarget = expression(Assessment@FMSY),
  proj_args = list(process_error = 1, p_sim = 1),
  diagnostic = c("min", "full", "none"),
  ...
)

make_MP(.Assess, .HCR, diagnostic = c("min", "full", "none"), ...)
```

Arguments

.Assess	Assessment model, a function of class Assess.
.HCR	Harvest control rule, a function of class HCR. Currently not used in projection MPs.
AddInd	A vector of integers or character strings indicating the indices to be used in the assessment model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd. For the interim procedure, the function will use the first index in AddInd.
assessment_interval	The time interval for when the assessment model is applied (number of years). In all other years, the interim procedure is applied.
type	How the index is used to calculate the TAC in the interim procedure. See details.
type_par	A control parameter for the interim procedure. See details.
diagnostic	A character string describing if any additional diagnostic information from the assessment models will be collected during the closed-loop simulation. "min" (minimal) will collect information on convergence (default) and "full" will also collect the model estimates of biomass and F generated by .Assess. "none" skips this step.
...	Additional arguments to be passed to .Assess and .HCR.
Ftarget	An expression that the MP will evaluate to identify the F used in the projection. See projection and example.
proj_args	Additional arguments for projection.

Details

make_interim_MP creates an MP that runs the interim procedure (updating the TAC according to index observations in between periodic assessment intervals. **Always ensure to set:** OM@interval <- 1. The assessment frequency is specified in argument assessment_interval.

In the year when the assessment is applied, the TAC is set by fitting the model and then running the harvest control rule. Between assessments, the TAC is updated as

$$\text{TAC}_{y+1} = C_{\text{ref}}(I_y + b \times s)/(I_{\text{ref}} + b \times s)$$

where Cref is the TAC calculated from the most recent assessment, Iref is the value of the index when Cref was calculated (see Equations 6 and 7 of Huynh et al. 2020). The value of I_y depends on type, with b and s equal zero unless type = "buffer":

- "buffer" - I_y is the most recent index with b is specifed by type_par (default = 1), and s is the standard deviation of index residuals from the most recent assessment.
- "mean" - I_y is the mean value of the index over the most recent type_par years (default = 3).
- "loess" - I_y is the most recent index predicted by a loess smoother applied over the entire time series of the index. Use type_par to adjust the span parameter (default = 0.75).
- "none" - I_y is the most recent index. Index values are not adjusted in the interim procedure.

Value

A function of class MP.

References

Huynh et al. 2020. The interim management procedure approach for assessed stocks: Responsive management advice and lower assessment frequency. Fish Fish. 21:663–679. doi:10.1111/faf.12453

See Also

HCR_ramp HCR_MSY diagnostic retrospective_AM

Examples

```
# Interim MPs
MP_buffer_5 <- make_interim_MP(assessment_interval = 5)
MP_buffer_10 <- make_interim_MP(assessment_interval = 10)
OM <- MSEtool::testOM
OM@interval <- 1

MSE <- MSEtool::runMSE(OM, MPs = c("MP_buffer_5", "MP_buffer_10"))

# A statistical catch-at-age model with a 40-10 control rule
SCA_40_10 <- make_MP(SCA, HCR40_10)

# An SCA that will produce convergence diagnostics
SCA_40_10 <- make_MP(SCA, HCR40_10, diagnostic = "min")

# MP with an SCA that uses a Ricker stock-recruit function.
SCA_Ricker <- make_MP(SCA, HCR_MSY, SR = "Ricker")
show(SCA_Ricker)


# TAC is calculated annually from triennial assessments with projections between
# assessments with F = 0.75 FMSY
# Projections by default assume no process error.
OM <- MSEtool::testOM
OM@interval <- 1
pMP <- make_projection_MP(SCA, assessment_interval = 3,
                          Ftarget = expression(0.75 * Assessment@FMSY),
                          proj_args = list(process_error = 1))
```

Model-based-MP *Model-based management procedures*

Description

A suite of model-based management procedures (MPs) included in the package. Additional MPs, with specific model configurations (e.g., stock-recruit function or fixing certain parameters) or alternative ramped harvest control rules can be created with make_MP and the available Assess and HCR objects with constant TAC between assessment years.

Usage

```
SCA_MSY(x, Data, reps = 1, diagnostic = "min")

SCA_75MSY(x, Data, reps = 1, diagnostic = "min")

SCA_4010(x, Data, reps = 1, diagnostic = "min")

DDSS_MSY(x, Data, reps = 1, diagnostic = "min")

DDSS_75MSY(x, Data, reps = 1, diagnostic = "min")

DDSS_4010(x, Data, reps = 1, diagnostic = "min")

SP_MSY(x, Data, reps = 1, diagnostic = "min")

SP_75MSY(x, Data, reps = 1, diagnostic = "min")

SP_4010(x, Data, reps = 1, diagnostic = "min")

SSS_MSY(x, Data, reps = 1, diagnostic = "min")

SSS_75MSY(x, Data, reps = 1, diagnostic = "min")

SSS_4010(x, Data, reps = 1, diagnostic = "min")
```

Arguments

x	A position in the Data object.
Data	An object of class Data
reps	Numeric, the number of stochastic replicates for the management advice.
diagnostic	Character string describing the assessment diagnostic to save, see make_MP.

Value

An object of class MSEtool::Rec which contains the management recommendation.

Functions

- SCA_MSY(): A statistical catch-at-age model with a TAC recommendation based on fishing at FMSY, and default arguments for configuring SCA.
- SCA_75MSY(): An SCA with a TAC recommendation based on fishing at 75% of FMSY.

- SCA_4010(): An SCA with a 40-10 control rule.
- DDSS_MSY(): A state-space delay difference model with a TAC recommendation based on fishing at FMSY, and default arguments for configuring DD_SS.
- DDSS_75MSY(): A state-space delay difference model with a TAC recommendation based on fishing at 75% of FMSY.
- DDSS_4010(): A state-space delay difference model with a 40-10 control rule.
- SP_MSY(): A surplus production model with a TAC recommendation based on fishing at FMSY, and default arguments for configuring SP.
- SP_75MSY(): A surplus production model with a TAC recommendation based on fishing at 75% of FMSY.
- SP_4010(): A surplus production model with a 40-10 control rule.
- SSS_MSY(): Simple stock synthesis (terminal depletion fixed to 0.4 in SSS) with a TAC recommendation based on fishing at FMSY.
- SSS_75MSY(): Simple stock synthesis (terminal depletion fixed to 0.4) with with a TAC recommendation based on fishing at 75% FMSY.
- SSS_4010(): Simple stock synthesis (terminal depletion fixed to 0.4) with a 40-10 control rule.

Examples

```
MSEtool::avail("MP", package = "SAMtool")


myMSE <- MSEtool::runMSE(MSEtool::testOM, MPs = c("FMSYref", "SCA_4010"))
```

---

pcod *Pacific cod in Area 5ABCD (Hecate Strait and Queen Charlotte Sound), British Columbia, Canada*

Description

A list containing an operating model, data set, and priors for updating the operating model using the conditioning model RCM.

Usage

```
pcod
```

Format

A list containing an object of class MSEtool::OM, RCMdata, and a list of priors for index catchability.

References

Forrest, R.E., Anderson, S.C., Grandin, C.J., and Starr, P.J. 2020. Assessment of Pacific Cod (Gadus macrocephalus) for Hecate Strait and Queen Charlotte Sound (Area 5ABCD), and West Coast Vancouver Island (Area 3CD) in 2018. DFO Can. Sci. Advis. Sec. Res. Doc. 2020/070. v + 215 p.

DFO. 2021. Status Update of Pacifc Cod (Gadus macrocephalus) for West Coast Vancouver Island (Area 3CD), and Hecate Strait and Queen Charlotte Sound (Area 5ABCD) in 2020. DFO Can. Sci. Advis. Sec. Sci. Resp. 2021/002.

See Also

RCM

Examples

```
data(pcod)
```

`plot.Assessment` *Plot Assessment object*

Description

Produces HTML file (via markdown) figures of parameter estimates and output from an Assessment object.

Usage

```
## S4 method for signature 'Assessment,missing'
plot(
  x,
  filename = paste0("report_", x@Model),
  dir = tempdir(),
  ret_yr = 0L,
  open_file = TRUE,
  quiet = TRUE,
  render_args = list(),
  ...
)

## S4 method for signature 'Assessment,retro'
plot(
  x,
  y,
  filename = paste0("report_", x@Model),
  dir = tempdir(),
  open_file = TRUE,
```

```
  quiet = TRUE,
  render_args = list(),
  ...
)
```

Arguments

x	An object of class Assessment.
filename	Character string for the name of the markdown and HTML files.
dir	The directory in which the markdown and HTML files will be saved.
ret_yr	If greater than zero, then a retrospective analysis will be performed and results will be reported. The integer here corresponds to the number of peels (the maximum number of terminal years for which the data are removed).
open_file	Logical, whether the HTML document is opened after it is rendered.
quiet	Logical, whether to silence the markdown rendering function.
render_args	Arguments to pass to render.
...	Other arguments.
y	An object of class retro.

Value

Returns invisibly the output from render.

See Also

retrospective

Examples

```
output <- DD_TMB(Data = Simulation_1)


plot(output)
```

---

plot.prof *Plot profile object*

Description

Generates a profile plot generated by profile. If a two-parameter profile is performed, then a contour plot of the likelihood surface is returned.

Usage

```
## S4 method for signature 'prof,missing'
plot(x, contour_levels = 20, ...)
```

Arguments

x	An object of class prof returned by profile.
contour_levels	Integer, passed to nlevels argument of contour.
...	Miscellaneous. Not used.

Value

A likelihood profile plot, either a one-dimensional line plot or a two-dimensional contour plot.

Author(s)

Q. Huynh

---

plot.RCModel *Plot RCM scope output*

Description

Produces HTML file (via markdown) figures of parameter estimates and output from an Assessment object. Plots histograms of operating model parameters that are updated by the RCM scoping function, as well as diagnostic plots for the fits to the RCM for each simulation. compare_RCM plots a short report that compares output from multiple RCM objects, assuming the same model structure, i.e., identical matrix and array dimensions among models, but different data weightings, data omissions, etc.

Usage

```
## S4 method for signature 'RCModel,missing'
plot(
  x,
  compare = FALSE,
  filename = "RCM",
  dir = tempdir(),
  sims = 1:x@OM@nsim,
  Year = NULL,
  Age = NULL,
  f_name = NULL,
  s_name = NULL,
  MSY_ref = c(0.5, 1),
  bubble_adj = 1.5,
  scenario = list(),
  title = NULL,
  open_file = TRUE,
  quiet = TRUE,
  render_args,
  ...
```

```
)

compare_RCM(
  ...,
  compare = FALSE,
  filename = "compare_RCM",
  dir = tempdir(),
  Year = NULL,
  Age = NULL,
  f_name = NULL,
  s_name = NULL,
  MSY_ref = c(0.5, 1),
  bubble_adj = 1.5,
  scenario = list(),
  title = NULL,
  open_file = TRUE,
  quiet = TRUE,
  render_args
)
```

Arguments

x	An object of class RCModel (output from RCM).
compare	Logical, if TRUE, the function will run runMSE to compare the historical period of the operating model and the RCM output.
filename	Character string for the name of the markdown and HTML files.
dir	The directory in which the markdown and HTML files will be saved.
sims	A logical vector of length x@OM@nsim or a numeric vector indicating which simulations to keep.
Year	Optional, a vector of years for the historical period for plotting. Useful if seasonal time steps are used.
Age	Optional, a vector of ages for plotting. Useful if seasonal time steps are used.
f_name	Character vector for fleet names.
s_name	Character vector for survey names.
MSY_ref	A numeric vector for reference horizontal lines for B/BMSY plots.
bubble_adj	A number to adjust the size of bubble plots (for residuals of age and length comps).
scenario	Optional, a named list to label each simulation in the RCM for plotting, e.g.: list(names = c("low M", "high M"), col = c("blue", "red")).
title	Optional character string for an alternative title for the markdown report.
open_file	Logical, whether the HTML document is opened after it is rendered.
quiet	Logical, whether to silence the markdown rendering function.
render_args	A list of other arguments to pass to render.
...	For compare_RCM, multiple RCM objects for comparison.

Value

Returns invisibly the output from render.

See Also

RCModel RCM

`plot.retro` *Methods for retro object*

Description

plot and summary functions for retro object.

Usage

```
## S4 method for signature 'retro,missing'
plot(x, color = NULL)

## S4 method for signature 'retro'
summary(object)
```

Arguments

`x`	An object of class retro.
`color`	An optional character vector of colors for plotting.
`object`	An object of class retro.

Value

A series of plots showing retrospective patterns in fishing mortality, spawning biomass, recruitment, etc.

Author(s)

Q. Huynh

Examples

```
res <- SP(Data = swordfish)
ret <- retrospective(res, figure = FALSE)

summary(ret)
plot(ret)
```

plot_betavar *Plots a beta variable*

Description

Plots the probability distribution function of a beta variable from the mean and standard deviation in either transformed (logit) or untransformed space.

Usage

```
plot_betavar(m, sd, label = NULL, is_logit = FALSE, color = "black")
```

Arguments

m	A vector of means of the distribution.
sd	A vector of standard deviations of the distribution.
label	Name of the variable to be used as x-axis label.
is_logit	Logical that indicates whether the means and standard deviations are in logit (TRUE) or normal (FALSE) space.
color	A vector of colors.

Value

A plot of the probability distribution function. Vertical dotted line indicates mean of distribution. This function can plot multiple curves when multiple means and standard deviations are provided.

Author(s)

Q. Huynh

See Also

plot_lognormalvar() plot_steepness()

Examples

```
mu <- 0.5
stddev <- 0.1
plot_betavar(mu, stddev) # mean of plot should be 0.5

#logit parameters
mu <- 0
stddev <- 0.1
plot_betavar(mu, stddev, is_logit = TRUE) # mean of plot should be 0.5
```

plot_composition *Plot composition data*

Description

Plots annual length or age composition data.

Usage

```
plot_composition(
  Year = 1:nrow(obs),
  obs,
  fit = NULL,
 plot_type = c("annual", "bubble_data", "bubble_residuals", "mean", "heat_residuals",
    "hist_residuals"),
  N = rowSums(obs),
  CAL_bins = NULL,
  ages = NULL,
  ind = 1:nrow(obs),
  annual_ylab = "Frequency",
  annual_yscale = c("proportions", "raw"),
  bubble_adj = 1.5,
  bubble_color = c("#99999999", "white"),
  fit_linewidth = 3,
  fit_color = "red"
)
```

Arguments

Year	A vector of years.
obs	A matrix of either length or age composition data. For lengths, rows and columns should index years and length bin, respectively. For ages, rows and columns should index years and age, respectively.
fit	A matrix of predicted length or age composition from an assessment model. Same dimensions as obs.
plot_type	Indicates which plots to create. Options include annual distributions, bubble plot of the data, and bubble plot of the Pearson residuals, and annual means.
N	Annual sample sizes. Vector of length nrow(obs).
CAL_bins	A vector of lengths corresponding to the columns in obs. and fit. Ignored for age data.
ages	An optional vector of ages corresponding to the columns in obs.
ind	A numeric vector for plotting a subset of rows (which indexes year) of obs and fit.
annual_ylab	Character string for y-axis label when plot_type = "annual".

annual_yscale	For annual composition plots (plot_type = "annual"), whether the raw values ("raw") or frequencies ("proportions") are plotted.
bubble_adj	Numeric, for adjusting the relative size of bubbles in bubble plots (larger number = larger bubbles).
bubble_color	Colors for negative and positive residuals, respectively, for bubble plots.
fit_linewidth	Argument lwd for fitted line.
fit_color	Color of fitted line.

Value

Plots depending on plot_type. Invisibly returns a matrix or list of values that were plotted.

Author(s)

Q. Huynh

Examples

```
plot_composition(obs = SimulatedData@CAA[1, 1:16, ])
plot_composition(
  obs = SimulatedData@CAA[1, , ],
  plot_type = "bubble_data",
  ages = 0:SimulatedData@MaxAge
)

SCA_fit <- SCA(x = 2, Data = SimulatedData)
plot_composition(
  obs = SimulatedData@CAA[1, , ], fit = SCA_fit@C_at_age,
  plot_type = "mean", ages = 0:SimulatedData@MaxAge
)

plot_composition(
  obs = SimulatedData@CAA[1, 1:16, ], fit = SCA_fit@C_at_age[1:16, ],
  plot_type = "annual", ages = 0:SimulatedData@MaxAge
)

plot_composition(
  obs = SimulatedData@CAA[1, , ], fit = SCA_fit@C_at_age,
  plot_type = "bubble_residuals", ages = 0:SimulatedData@MaxAge
)

plot_composition(
  obs = SimulatedData@CAA[1, , ], fit = SCA_fit@C_at_age,
  plot_type = "heat_residuals", ages = 0:SimulatedData@MaxAge
)

plot_composition(
  obs = SimulatedData@CAA[1, , ], fit = SCA_fit@C_at_age,
  plot_type = "hist_residuals", ages = 0:SimulatedData@MaxAge
)
```

plot_crosscorr *Produce a cross-correlation plot of the derived data arising from getinds(MSE_object)*

Description

Produce a cross-correlation plot of the derived data arising from getinds(MSE_object)

Usage

```
plot_crosscorr(
  indPPD,
  indData,
  pp = 1,
  dnam = c("CS", "CV", "CM", "IS", "MLS"),
  res = 1
)
```

Arguments

indPPD	A 3D array of results arising from running getind on an MSE of the Null operating model (type of data/stat (e.g. mean catches),time period (chunk), simulation)
indData	A 3D array of results arising from running getind on an MSE of the Alternative operating model (type of data/stat (e.g. mean catches),time period (chunk), simulation)
pp	Positive integer, the number of time chunks (blocks of years normally, second dimension of indPPD and indData) to produce the plot for.
dnam	A character vector of names of the data for plotting purposes (as long as dimension 1 of indPPD and indData).
res	The size of the temporal blocking that created indPPD and indData - this is just used for labelling purposes

Value

A cross-correlation plot (ndata-1) x (ndata-1)

Author(s)

T. Carruthers

References

Carruthers and Hordyk 2018

plot_lognormalvar *Plots a lognormal variable*

Description

Plots the probability distribution function of a lognormal variable from the mean and standard deviation in either transformed (normal) or untransformed space.

Usage

```
plot_lognormalvar(m, sd, label = NULL, logtransform = FALSE, color = "black")
```

Arguments

m	A vector of means of the distribution.
sd	A vector of standard deviations of the distribution.
label	Name of the variable to be used as x-axis label.
logtransform	Indicates whether the mean and standard deviation are in lognormal (TRUE) or normal (FALSE) space.
color	A vector of colors.

Value

A plot of the probability distribution function. Vertical dotted line indicates mean of distribution. This function can plot multiple curves when multiple means and standard deviations are provided.

Author(s)

Q. Huynh

See Also

plot_betavar() plot_steepness()

Examples

```
mu <- 0.5
stddev <- 0.1
plot_lognormalvar(mu, stddev) # mean of plot should be 0.5

#logtransformed parameters
mu <- 0
stddev <- 0.1
plot_lognormalvar(mu, stddev, logtransform = TRUE) # mean of plot should be 1
```

plot_residuals *Plot residuals*

Description

Plots figure of residuals (or any time series with predicted mean of zero).

Usage

```
plot_residuals(
  Year,
  res,
  res_sd = NULL,
  res_sd_CI = 0.95,
  res_upper = NULL,
  res_lower = NULL,
  res_ind_blue = NULL,
  draw_zero = TRUE,
  zero_linetype = 2,
  label = "Residual"
)
```

Arguments

Year	A vector of years for the data.
res	A vector of residuals.
res_sd	A vector of year specific standard deviation for res.
res_sd_CI	The confidence interval for the error bars based for res_sd.
res_upper	A vector of year-specific upper bounds for the error bars of the residual (in lieu of argument res_CV).
res_lower	A vector of year-specific lower bounds for the error bars of the residual (in lieu of argument res_CV).
res_ind_blue	Indices of obs for which the plotted residuals and error bars will be blue.
draw_zero	Indicates whether a horizontal line should be drawn at zero.
zero_linetype	Passes argument lty (e.g. solid line = 1, dotted = 2) to draw_zero.
label	Character string that describes the data to label the y-axis.

Value

A plot of model residuals by year (optionally, with error bars).

Author(s)

Q. Huynh

See Also

plot_timeseries()

---

plot_SR *Plot stock-recruitment function*

Description

Plot stock-recruitment (with recruitment deviations if estimated).

Usage

```
plot_SR(
  Spawners,
  expectedR,
  R0 = NULL,
  S0 = NULL,
  rec_dev = NULL,
  trajectory = FALSE,
  y_zoom = NULL,
  ylab = "Recruitment"
)
```

Arguments

Spawners	A vector of the number of the spawners (x-axis).
expectedR	A vector of the expected recruitment (from the stock-recruit function) corresponding to values of Spawners.
R0	Virgin recruitment.
S0	Virgin spawners.
rec_dev	If recruitment deviations are estimated, a vector of estimated recruitment (in normal space) corresponding to values of Spawners.
trajectory	Indicates whether arrows will be drawn showing the trajectory of spawners and recruitment deviations over time.
y_zoom	If recruitment deviations are plotted, the y-axis limit relative to maximum expected recruitment expectedR. If NULL, all recruitment values are plotted.
ylab	Character string for label on y-axis.

Value

A stock-recruit plot

Author(s)

Q. Huynh

plot_steepness *Plots probability distribution function of stock-recruit steepness*

Description

Plots the probability distribution function of steepness from the mean and standard deviation.

Usage

```
plot_steepness(
  m,
  sd,
  is_transform = FALSE,
  SR = c("BH", "Ricker"),
  color = "black"
)
```

Arguments

m	The mean of the distribution (vectorized).
sd	The standard deviation of the distribution (vectorized).
is_transform	Logical, whether the mean and standard deviation are in normal space (FALSE) or transformed space.
SR	The stock recruitment relationship (determines the range and, if relevant, transformation of steepness).
color	A vector of colors.

Value

A plot of the probability distribution function. Vertical dotted line indicates mean of distribution.

Note

The function samples from a beta distribution with parameters alpha and beta that are converted from the mean and standard deviation. Then, the distribution is transformed from 0 - 1 to 0.2 - 1.

Author(s)

Q. Huynh

See Also

plot_lognormalvar() plot_betavar()

Examples

```
mu <- 0.8
stddev <- 0.1
plot_steepness(mu, stddev)
```

`plot_timeseries` *Plot time series of data*

Description

Plot time series of observed (with lognormally-distributed error bars) vs. predicted data.

Usage

```
plot_timeseries(
  Year,
  obs,
  fit = NULL,
  obs_CV = NULL,
  obs_CV_CI = 0.95,
  obs_upper = NULL,
  obs_lower = NULL,
  obs_ind_blue = NULL,
  fit_linewidth = 3,
  fit_color = "red",
  label = "Observed data"
)
```

Arguments

`Year`	A vector of years for the data.
`obs`	A vector of observed data.
`fit`	A vector of predicted data (e.g., from an assessment model).
`obs_CV`	A vector of year-specific coefficient of variation in the observed data.
`obs_CV_CI`	The confidence interval for the error bars based for `obs_CV`.
`obs_upper`	A vector of year-specific upper bounds for the error bars of the observed data (in lieu of argument `obs_CV`).
`obs_lower`	A vector of year-specific lower bounds for the error bars of the observed data (in lieu of argument `obs_CV`).
`obs_ind_blue`	Indices of `obs` for which the plotted points and error bars will be blue.
`fit_linewidth`	Argument `lwd` for fitted line.
`fit_color`	Color of fitted line.
`label`	Character string that describes the data to label the y-axis.

Value

A plot of annual observed data and predicted values from a model.

Author(s)

Q. Huynh

See Also

plot_residuals()

Examples

```
data(Red_snapper)
plot_timeseries(Red_snapper@Year, Red_snapper@Cat[1, ],
obs_CV = Red_snapper@CV_Cat, label = "Catch")
```

---

posterior *Sample posterior of TMB models in SAMtool*

Description

A convenient wrapper function (posterior) to sample the posterior using MCMC in rstan and returns a stanfit object for diagnostics. Use RCMstan to update the RCM and the enclosed operating model with MCMC samples..

Usage

```
posterior(x, ...)

## S4 method for signature 'RCModel'
posterior(
  x,
  priors_only = FALSE,
  laplace = FALSE,
  chains = 2,
  iter = 2000,
  warmup = floor(iter/2),
  thin = 5,
  seed = 34,
  init = "last.par.best",
  cores = chains,
  ...
)

## S4 method for signature 'Assessment'
posterior(x, priors_only = FALSE, ...)

RCMstan(RCModel, stanfit, sim, cores = 1, silent = FALSE)
```

Arguments

x	An object of class Assessment or RCModel.
...	Additional arguments to pass to rstan::sampling via tmbstan::tmbstan.
priors_only	Logical, whether to set the likelihood to zero and sample the priors only.
laplace	Logical, whether to do the Laplace approximation for random parameters.
chains	The numer of MCMC chains.
iter	The number of iterations for each chain, including warmup.
warmup	The number of burnin iterations
thin	The frequency at which iterations are kept (e.g., 5 saves every fifth iteration)
seed	Seed for random number generator during the MCMC.
init	The initial values of parameters for starting the MCMC chain. See tmbstan::tmbstan.
cores	The number of cores for running in parallel, e.g., one core per MCMC chain. Used in RCMstan for reconstructing the population.
RCModel	An object of class RCModel
stanfit	An object of class stanfit returned by posterior.
sim	A matrix of RCModel@OM@nsim rows and 2 columns that specifies the samples used to update the operating model. The first column specifies the chain and the second columns specifies the MCMC iteration.
silent	Logical to indicate if progress messages should be printed to console.

Value

posterior returns an object of class stanfit. See class?stanfit.

RCMstan returns an updated RCModel.

Online Documentation

A vignette on the steps to run the MCMC is available on the openMSE website.

Author(s)

Q. Huynh

PRBcalc *Calculate mahalanobis distance (null and alternative MSEs) and statistical power for all MPs in an MSE*

Description

Calculate mahalanobis distance (null and alternative MSEs) and statistical power for all MPs in an MSE

Usage

```
PRBcalc(
  MSE_null,
  MSE_alt,
  tsd = c("Cat", "Cat", "Cat", "Ind", "ML"),
  stat = c("slp", "AAV", "mu", "slp", "slp"),
  dnam = c("C_S", "C_V", "C_M", "I_S", "ML_S"),
  res = 6,
  alpha = 0.05,
  plotCC = FALSE,
  removedat = FALSE,
  removethresh = 0.025
)
```

Arguments

MSE_null	An object of class MSE representing the null hypothesis
MSE_alt	An object of class MSE representing the alternative hypothesis
tsd	Character string of data types: Cat = catch, Ind = relative abundance index, ML = mean length in catches
stat	Character string defining the quantity to be calculated for each data type, slp = slope(log(x)), AAV = average annual variability, mu = mean(log(x))
dnam	Character string of names for the quantities calculated
res	Integer, the resolution (time blocking) for the calculation of PPD
alpha	Probability of incorrectly rejecting the null operating model when it is valid
plotCC	Logical, should the PPD cross correlations be plotted?
removedat	Logical, should data not contributing to the mahalanobis distance be removed?
removethresh	Positive fraction: the cumulative percentage of removed data (removedat=TRUE) that contribute to the mahalanobis distance

Value

A list object with two hierarchies of indexing, first by MP, second has two positions as described in Probs: (1) mahalanobis distance, (2) a matrix of type 1 error (first row) and statistical power (second row), by time block.

Author(s)

T. Carruthers

References

Carruthers, T.R, and Hordyk, A.R. In press. Using management strategy evaluation to establish indicators of changing fisheries. Canadian Journal of Fisheries and Aquatic Science.

`prelim_AM` *Preliminary Assessments in MSE*

Description

Evaluates the likely performance of Assessment models in the operating model. This function will apply the assessment model for Data generated during the historical period of the MSE, and report the convergence rate for the model and total time elapsed in running the assessments.

Usage

```
prelim_AM(x, Assess, ncpus = NULL, ...)
```

Arguments

`x`	Either a `Hist`, `Data` or `OM` object.
`Assess`	An Assess function of class `Assess`.
`ncpus`	Numeric, the number of CPUs to run the Assessment model (will run in parallel if greater than 1).
`...`	Arguments to be passed to `Assess`, e.g., model configurations.

Value

Returns invisibly a list of Assessment objects of length `OM@nsim`. Messages via console.

Author(s)

Q. Huynh

Examples

```
prelim_AM(MSEtool::SimulatedData, SP)
```

Probs *Calculates mahalanobis distance and rejection of the Null operating model*

Description

Calculates mahalanobis distance and rejection of the Null operating model, used by wrapping function PRBcalc.

Usage

```
Probs(indPPD, indData, alpha = 0.05, removedat = FALSE, removethresh = 0.05)
```

Arguments

indPPD	A 3D array of results arising from running getind on an MSE of the Null operating model (type of data/stat (e.g. mean catches),time period (chunk), simulation)
indData	A 3D array of results arising from running getind on an MSE of the Alternative operating model (type of data/stat (e.g. mean catches),time period (chunk), simulation)
alpha	Positive fraction: rate of type I error, alpha
removedat	Logical, should data not contributing to the mahalanobis distance be removed?
removethresh	Positive fraction: the cumulative percentage of removed data (removedat=TRUE) that contribute to the mahalanobis distance

Value

A list object. Position 1 is an array of the mahalanobis distances. Dimension 1 is length 2 for the Null OM (indPPD) and the alternative OM (indData). Dimension 2 is the time block (same length as indPPD dim 2). Dimension 3 is the simulation number (same length at indPPD dim 3.), Position 2 is a matrix (2 rows, ntimeblock columns) which is (row 1) alpha: the rate of false positives, and row 2 the power (1-beta) the rate of true positives

Author(s)

T. Carruthers

References

Carruthers and Hordyk 2018

prof-class *Class*-prof

Description

An S4 class that contains output from profile.

Slots

`Model` Name of the assessment model.

`Name` Name of Data object.

`Par` Character vector of parameters that were profiled.

`MLE` Numeric vector of the estimated values of the parameters (corresponding to `Par`) from the assessment.

`grid` A data.frame of the change in negative log-likelihood (`nll`) based on the profile of the parameters.

Author(s)

Q. Huynh

See Also

plot.prof profile

profile,Assessment-method *Profile likelihood of assessment models*

Description

Profile the likelihood for parameters of assessment models.

Usage

```
## S4 method for signature 'Assessment'
profile(fitted, figure = TRUE, ...)

## S4 method for signature 'RCModel'
profile(fitted, figure = TRUE, ...)
```

Arguments

fitted, Assessment	An object of class Assessment.
figure	Logical, indicates whether a figure will be plotted.
...	A sequence of values of the parameter(s) for the profile. See details and example below. See details for name of arguments to be passed on.

Details

For the following assessment models, possible sequence of values for profiling are:

- DD_TMB and DD_SS: R0 and h
- SP and SP_SS: FMSY and MSY
- DD and cDD_SS: R0 and h
- SCA and SCA_Pope: R0 and h
- SCA2: meanR
- VPA: F_term
- SSS: R0

For RCM: D (spawning biomass depletion), R0, and h are used. If the Mesnil-Rochet stock-recruit function is used, can also profile MRRmax and MRgamma.

Value

An object of class prof that contains a data frame of negative log-likelihood values from the profile and, optionally, a figure of the likelihood surface.

Author(s)

Q. Huynh

Examples

```
output <- SCA(Data = MSEtool::SimulatedData)

# Profile R0 only
pro <- profile(output, R0 = seq(1000, 2000, 50))

# Profile both R0 and steepness
pro <- profile(output, R0 = seq(1000, 2000, 100), h = seq(0.8, 0.95, 0.025))

# Ensure your grid is of proper resolution. A grid that is too coarse
# will likely distort the shape of the likelihood surface.
```

project-class *Class*-project

Description

An S4 class for the output from projection.

Slots

`Model` Name of the assessment model.

`Name` Name of Data object.

`FMort` A matrix of fishing mortality over `p_sim` rows and `p_years` columns.

`B` An matrix of biomass with `p_sim` rows and `p_years` columns.

`SSB` A matrix of spawning biomass with `p_sim` rows and `p_years` columns.

`VB` A matrix of vulnerable biomass with `p_sim` rows and `p_years` columns.

`R` A matrix of recruitment over `p_sim` rows and `p_years` columns.

`N` A matrix of abundance over `p_sim` rows and `p_years` columns.

`Catch` A matrix of simulated observed catch over `p_sim` rows and `p_years` columns.

`Index` An array of simulated observed index of dimension `c(p_sim, p_years, nsurvey)`.

`C_at_age` An array for catch-at-age with dimension `c(p_sim, p_years, n_age)`.

Author(s)

Q. Huynh

See Also

projection

projection *Projections for assessment models*

Description

This function takes an assessment model and runs a stochastic projection based on future F or catch.

Usage

```
projection(
  Assessment,
  constrain = c("F", "Catch"),
  Ftarget,
  Catch,
  p_years = 50,
  p_sim = 200,
  obs_error,
  process_error,
  max_F = 3,
  seed = 499
)
```

Arguments

Assessment	An object of class Assessment.
constrain	Whether to project on future F or catch. By default, projects on F.
Ftarget	The projection F, either of length 1 for constant F for the entirety of the projection or length p_years.
Catch	The projection catch, either of length 1 for constant catch for the entirety of the projection or length p_years.
p_years	Integer for the number of projection years.
p_sim	Integer for the number of simulations for the projection.
obs_error	A list of length two. In the first entry, a vector of length nsurvey giving the standard deviations of each future index, or alternatively an array of dimension p_sim, p_years, and nsurvey giving the deviates. The second entry is the standard deviation of the projected catch. Alternatively, a matrix of simulation and year-specific error structure for the catch (p_sim rows and p_year columns; a matrix of ones indicates perfect data).
process_error	Numeric, standard deviation for process error (e.g., recruitment or biomass deviates). If NULL, uses values from assessment model. Alternatively, a matrix of simulation and year-specific recruitment deviates (p_sim rows and p_year columns, a matrix of ones indicates no recruitment deviates).
max_F	The maximum allowable F if the projection is constrained on catch.
seed	An integer to set the seed for the sampling observation and process error deviates.

Value

An object of class project that contains future predicted values of F, catch, biomass, recruitment, etc.

Examples

```
myAssess <- SP(Data = swordfish)
do_projection <- projection(myAssess, Ftarget = myAssess@FMSY)
```

RCM2MOM *Convert RCM to a multi-fleet operating model (MOM)*

Description

The RCM (Rapid Conditioning Model) returns a single-fleet operating model, implying constant effort among fleets for projections. Here, we convert the single-fleet OM to a multi-fleet OM, preserving the multiple fleet structure used in the conditioning model for projections. This allows for testing management procedures that explicitly specify fleet allocation in the management advice.

Usage

```
RCM2MOM(RCModel)
```

Arguments

RCModel	Output from RCM, a class RCModel object.

Value

A class MSEtool::MOM object.

Author(s)

Q. Huynh

Examples

```
data(pcod)
mat_ogive <- pcod$OM@cpars$Mat_age[1, , 1]
OM <- MSEtool::SubCpars(pcod$OM, 1:3)
out <- RCM(OM = pcod$OM, data = pcod$data,
           condition = "catch", mean_fit = TRUE,
           selectivity = "free", s_selectivity = rep("SSB", ncol(pcod$data@Index)),
           start = list(vul_par = matrix(mat_ogive, length(mat_ogive), 1)),
           map = list(vul_par = matrix(NA, length(mat_ogive), 1),
                      log_early_rec_dev = rep(1, pcod$OM@maxage)),
           prior = pcod$prior)
MOM <- RCM2MOM(out)
```

RCMdata-class *Class*-RCMdata

Description

An S4 class for the data inputs into RCM.

Slots

`Chist` Either a vector of historical catch, should be of length `OM@nyears`, or if there are multiple fleets, a matrix of `OM@nyears` rows and `nfleet` columns. Ideally, the first year of the catch series represents unfished conditions (see also slot `C_eq`).

`C_sd` Same dimension as `Chist`. Lognormal distribution standard deviations (by year and fleet) for the catches in `Chist`. If not provided, the default is 0.01. Not used if `RCM(condition = "catch2")`.

`Ehist` A vector of historical effort, should be of length `OM@nyears`, or if there are multiple fleets: a matrix of `OM@nyears` rows and `nfleet` columns. See also slot `E_eq`).

`C_wt` Optional weight at age for the catch `Chist`. Array with dimension `[OM@nyears+1, OM@maxage+1, nfleet]`.

`CAA` Fishery age composition matrix with `nyears` rows and `OM@maxage+1` columns, or if multiple fleets: an array with dimension: `nyears, OM@maxage+1, nfleet`. Enter `NA` for years without any data. Raw numbers will be converted to annual proportions (see slot `CAA_ESS` for sample sizes).

`CAA_ESS` Annual sample size (for the multinomial distribution) of the fishery age comps. A vector of length `OM@nyears`, or if there are multiple fleets: a matrix of `OM@nyears` rows and `nfleet` columns. Enter zero for years without observations. An annual cap to the ESS, e.g., 50, can be calculated with something like: `pmin(apply(CAA, c(1, 3), sum, na.rm = TRUE), 50)`. By default,

`CAL` Fishery length composition matrix with `nyears` rows and `n_bin` columns (indexing the length bin), or if multiple fleets: an array with dimension: `nyears, n_bin, nfleets`. Enter `NA` for years without any data. Raw numbers will be converted to annual proportions (see slot `CAL_ESS` for sample sizes).

`CAL_ESS` Annual sample size (for the multinomial distribution) of the fishery length comps. Same dimension as `CAA_ESS`.

`length_bin` • A vector (length `n_bin`) for the midpoints of the length bins for `CAL` and `IAL`, as well as the population model, if all bin widths are equal in size. If length bins are unequal in width, then provide a vector of the boundaries of the length bins (vector of length `n_bin + 1`).

`MS` Mean mean size (either mean length or mean weight) observations from the fishery. Same dimension as `Chist`. Generally, mean lengths should not be used alongside `CAL`, unless mean length and length comps are independently sampled.

`MS_type` A character (either `"length"` (default) or `"weight"`) to denote the type of mean size data.

`MS_cv` The coefficient of variation of the observed mean size. If there are multiple fleets, a vector of length `nfleet`. Default is 0.2.

`Index` Index of abundance. Enter `NA` for missing values. A vector length `OM@nyears`, or if there are multiple surveys: a matrix of `OM@nyears` rows and `nsurvey` columns.

`I_sd` A vector or matrix of standard deviations (lognormal distribution) for the indices corresponding to the entries in `Index`. Same dimension as `Index`. If not provided, this function will use values from `OM@Iobs`.

`I_wt` Optional weight at age for the index `Index`. Array with dimension `[OM@nyears, OM@maxage+1, nsurvey]`.

`IAA` Index age composition data, an array of dimension `nyears, maxage+1, nsurvey`. Raw numbers will be converted to annual proportions (see `IAA_ESS` for sample sizes).

`IAA_ESS` Annual sample size (for the multinomial distribution) of the index age comps. A vector of length `OM@nyears`. If there are multiple indices: a matrix of `OM@nyears` rows and `nsurvey` columns.

`IAL` Index length composition data, an array of dimension `nyears, n_bin, nsurvey`. Raw numbers will be converted to annual proportions (see slot `IAL_ESS` to enter sample sizes).

`IAL_ESS` Annual sample size (for the multinomial distribution) of the index length comps. Same dimension as `IAA_ESS`.

`C_eq` Vector of length `nfleet` for the equilibrium catch for each fleet in `Chist` prior to the first year of the operating model. Zero (default) implies unfished conditions in year one. Otherwise, this is used to estimate depletion in the first year of the data. Alternatively, if one has a full CAA matrix, one could instead estimate "artificial" rec devs to generate the initial numbers-at-age (and hence initial depletion) in the first year of the model (see additional arguments in RCM).

`C_eq_sd` • A vector of standard deviations (lognormal distribution) for the equilibrium catches in `C_eq`. Same dimension as `C_eq`. If not provided, the default is 0.01. Only used if `RCM(condition = "catch")`.

`E_eq` The equilibrium effort for each fleet in `Ehist` prior to the first year of the operating model. Zero (default) implies unfished conditions in year one. Otherwise, this is used to estimate depletion in the first year of the data.

`abs_I` An integer vector length `nsurvey` to indicate which indices are in absolute magnitude. Use 1 to set q = 1, otherwise use 0 (default) to estimate q.

`I_units` An integer vector of length `nsurvey` to indicate whether indices are biomass based (1) or abundance-based (0). By default, all are biomass-based.

`I_delta` A vector of length `nsurvey` to indicate the timing of the indices within each time step (0-1, for example 0.5 is the midpoint of the year). By default, zero is used. Can also be a matrix by `nyears, nsurvey`. Use -1 if the survey operates continuously, the availability would be `N * (1 - exp(-Z))/Z`.

`age_error` A square matrix of `maxage + 1` rows and columns to specify ageing error. The `aa`-th column assigns a proportion of animals of true age `aa` to observed age `a` in the `a`-th row. Thus, all rows should sum to 1. Default is an identity matrix (no ageing error).

`sel_block` For time-varying fleet selectivity (in time blocks), a integer matrix of `nyears` rows and `nfleet` columns to assign a selectivity function to a fleet for certain years. By default, constant selectivity for each individual fleet. See the selectivity article for more details.

`Misc` A list of miscellaneous inputs. Used internally.

Author(s)

Q. Huynh

See Also

RCM

`RCModel-class` *Class*-`RCModel`

Description

An S4 class for the output from RCM.

Slots

`OM` An updated operating model, class MSEtool::OM.

`SSB` A matrix of estimated spawning biomass with `OM@nsim` rows and `OM@nyears+1` columns.

`NAA` An array for the predicted numbers at age with dimension `OM@nsim`, `OM@nyears+1`, and `OM@maxage+1`.

`CAA` An array for the predicted catch at age with dimension `OM@nsim`, `OM@nyears`, `OM@maxage`, and nfleet.

`CAL` An array for the predicted catch at length with dimension `OM@nsim`, `OM@nyears`, length bins, and nfleet.

`conv` A logical vector of length `OM@nsim` indicating convergence of the RCM in the i-th simulation.

`report` A list of length `OM@nsim` with more output from the fitted RCM. Within each simulation, a named list containing items of interest include:

- B - total biomass - vector of length nyears+1
- EPR0 - annual unfished spawners per recruit - vector of length nyears
- ageM - age of 50% maturity - integer
- EPR0_SR - unfished spawners per recruit for the stock-recruit relationship (mean EPR0 over the first `ageM` years) - numeric
- R0 - unfished recruitment for the stock-recruit relationship - numeric
- h - steepness for the stock-recruit relationship - numeric
- Arec - stock-recruit alpha parameter - numeric
- Brec - stock-recruit beta parameter - numeric
- E0_SR - unfished spawning biomass for the stock-recruit relationship (product of EPR0_SR and R0) - numeric
- CR_SR - compensation ratio, the product of Arec and EPR0_SR - numeric
- E0 - annual unfished spawning biomass (intersection of stock-recruit relationship and unfished spawners per recruit) - vector of length nyears
- R0_annual - annual unfished recruitment (annual ratio of E0 and EPR0) - vector of length nyears
- h_annual - annual steepness (calculated from EPR0 and Arec) - vector of length nyears
- CR - annual compensation ratio, the product of alpha and annual unfished spawners per recruit (EPR0) - vector of length nyears
- R - recruitment - vector of length nyears+1

- R_early - recruitment for the cohorts in first year of the model - vector n_age-1 (where n_age = maxage + 1)
- VB - vulnerable biomass - matrix of nyears x nfleet
- N - abundance at age - matrix of nyears+1 x n_age
- F - apical fishing mortality - matrix of nyears x nfleet
- F_at_age - fishing mortality at age - matrix of nyears x n_age
- F_equilibrium - equilibrium fishing mortality prior to first year - vector of length nfleet
- M - natural mortality - matrix of nyears x n_age
- Z - total mortality - matrix of nyears x n_age
- q - index catchability - vector of length nsurvey
- ivul - index selectivity at age - array of dim nyears+1, n_age, nsurvey
- ivul_len - corresponding index selectivity at length - matrix of nbins x nsurvey
- Ipred - predicted index values - matrix of nyears x nsurvey
- IAApred - predicted index catch at age - array of dim nyears, n_age, nsurvey
- vul - fleet selectivity at age - array of dim nyears+1, n_age, nfleet (or nsel_block)
- vul_len - corresponding fleet selectivity at length - matrix of nbins x nfleet (or nsel_block)
- IALpred - predicted index catch at length - array of dim nyears, nbins, nsurvey
- MLpred - predicted mean length - matrix of nyears x nfleet
- MWpred - predicted mean weight - matrix of nyears x nfleet
- CAApred - predicted catch at age - array of nyears, n_age, nfleet
- CALpred - predicted catch at length - array of nyears, nbins, nfleet
- Cpred - predicted catch in weight - matrix of nyears x nfleet
- CN - predicted catch in numbers - matrix of nyears x nfleet
- dynamic_SSB0 - the dynamic unfished spawning biomass calcaluated by projecting the historical model with zero catches - vector of length nyears+1
- SPR_eq - equilibrium spawning potential ratio calculated from annual F-at-age - vector of length nyears
- SPR_dyn - dynamic (transitional) spawning potential ratio calculated from cumulative survival of cohorts - vector of length nyears
- nll - total objective function of the model - numeric
- nll_fleet - objective function values for each annual data point(s) from fleets - array of nyears x nfleet x 5 (for Catch, equilibrium catch, CAA, CAL, and mean size)
- nll_index - objective function values for each annual data point(s) in the index - array of nyears x nsurvey x 3 (for Index, IAA, and IAL)
- prior - penalty value added to the objective function from priors - numeric
- penalty - additional penalty values added to the objective function due to high F - numeric
- conv - whether the model converged (whether a positive-definite Hessian was obtained) - logical

`mean_fit` A list of output from fit to mean values of life history parameters in the operating model. The named list consists of:

- obj - a list with components returned from `TMB::MakeADFun()`.
- opt - a list with components from calling `stats::nlminb()` to `obj`.

- SD - a list (class sdreport) with parameter estimates and their standard errors, obtained from `TMB::sdreport()`.
- report - a list of model output reported from the TMB executable, i.e. `obj$report()`. See Misc.

`data` A RCMdata object containing data inputs for the RCM.

`config` A list describing configuration of the RCM:

- drop_sim - a vector of simulations that were dropped for the output

`Misc` Slot for miscellaneous information for the user. Currently unused.

Author(s)

Q. Huynh

See Also

plot.RCModel RCM

`RCM_assess` *The rapid conditioning model as an assessment function*

Description

In beta testing. A function that uses RCM as an assessment function for use in MPs. More function arguments will be added to tinker with model settings and data inputs.

Usage

```
RCM_assess(
  x = 1,
  Data,
  AddInd = "B",
  SR = c("BH", "Ricker"),
  selectivity = c("logistic", "dome"),
  CAA_multiplier = 50,
  prior = list(),
  LWT = list(),
  StockPars = "Data",
  ...
)
```

Arguments

`x`	A position in the Data object (by default, equal to one for assessments).
`Data`	An object of class Data

AddInd	A vector of integers or character strings indicating the indices to be used in the model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd. Vulnerability to the survey is fixed in the model.
SR	Stock-recruit function (either "BH" for Beverton-Holt or "Ricker").
selectivity	Whether to model "logistic" or "dome" selectivity for the fishery.
CAA_multiplier	Numeric for data weighting of catch-at-age matrix. If greater than 1, then this is the maximum multinomial sample size in any year. If less than one, then the multinomial sample size is this fraction of the sample size.
prior	A named list for the parameters of any priors to be added to the model. See documentation in SCA.
LWT	A named list (Index, CAA, Catch) of likelihood weights for the data components. For the index, a vector of length survey. For CAL and Catch, a single value.
StockPars	Either a string ("Data" or "OM") to indicate whether to grab biological parameters from the Data object, or operating model. Alternatively, a named list to provide custom parameters for the assessment.
...	Additional arguments (to be added).

Data

Currently uses catch, CAA, and indices of abundance in the corresponding slots in the Data object.

StockPars

Biological parameters can be used from the (1) Data object, (2) operating model, or (3) provided directly in the StockPars argument.

Options 2 and 3 allow for time-varying growth, maturity, and natural mortality. Natural mortality can also be age-varying.

StockPars can be a named list of parameters used to provide inputs to the assessment model:

- Wt_age - annual weight at age, array [sim, ages, year]
- Mat_age - annual maturity at age, array [sim, ages, year]
- hs - Stock-recruit steepness, vector of length [sim]
- M_ageArray - annual natural mortality, array [sim, ages, year]

Examples

```
r <- RCM_assess(Data = SimulatedData)
myMP <- make_MP(RCM_assess, HCR_MSY)
myMP(x = 1, Data = SimulatedData)
```

retro-class *Class*-retro

Description

An S4 class that contains output from retrospective.

Slots

`Model` Name of the assessment model.

`Name` Name of Data object.

`TS_var` Character vector of time series variables, e.g. recruitment, biomass, from the assessment.

`TS` An array of time series assessment output of dimension, indexed by: peel (the number of terminal years removed from the base assessment), years, and variables (corresponding to `TS_var`).

`Est_var` Character vector of estimated parameters, e.g. R0, steepness, in the assessment.

`Est` An array for estimated parameters of dimension, indexed by: peel, variables (corresponding to `Est_var`), and value (length 2 for estimate and standard error).

Author(s)

Q. Huynh

See Also

plot.retro summary.retro plot.Assessment

retrospective *Retrospective analysis of assessment models*

Description

Perform a retrospective analysis, successive removals of most recent years of data to evaluate resulting parameter estimates.

Usage

```
retrospective(x, ...)

## S4 method for signature 'Assessment'
retrospective(x, nyr = 5, figure = TRUE)

## S4 method for signature 'RCModel'
retrospective(x, nyr = 5, figure = TRUE)
```

Arguments

x	An S4 object of class Assessment or RCModel.
...	More arguments.
nyr	The maximum number of years to remove for the retrospective analysis.
figure	Indicates whether plots will be drawn.

Value

A list with an array of model output and of model estimates from the retrospective analysis.

Figures showing the time series of biomass and exploitation and parameter estimates with successive number of years removed. For a variety of time series output (SSB, recruitment, etc.) and estimates (R0, steepness, etc.), also returns a matrix of Mohn's rho (Mohn 1999).

Author(s)

Q. Huynh

References

Mohn, R. 1999. The retrospective problem in sequential population analysis: an investigation using cod fishery and simulated data. ICES Journal of Marine Science 56:473-488.

Examples

```
output <- SP(Data = swordfish)
get_retro <- retrospective(output, nyr = 5, figure = FALSE)
```

---

retrospective_AM *retrospective_AM (retrospective of Assessment model in MSE)*

Description

Plots the true retrospective of an assessment model during the closed-loop simulation. A series of time series estimates of SSB, F, and VB are plotted over the course of the MSE are plotted against the operating model (true) values (in black).

Usage

```
retrospective_AM(MSE, MP, sim = 1, plot_legend = FALSE)
```

Arguments

MSE	An object of class MSEtool::MSE.
MP	Character. The name of the management procedure created by make_MP() containing the assessment model.
sim	Integer between 1 and MSE@nsim. The simulation number for which the retrospectives will be plotted.
plot_legend	Logical. Whether to plot legend to reference year of assessment in the MSE.

Details

For assessment models that utilize annual exploitation rates (u), the instantaneous fishing mortality rates are obtained as F = -log(1 - u).

Value

A series of figures for SSB, depletion, fishing mortality, and vulnerable biomass (VB) estimated in the MP over the course of the closed-loop simulation against the values generated in the operating model (both historical and projected).

Note

This function only plots retrospectives from a single simulation in the MSE. Results from one figure may not be indicative of general assessment behavior and performance overall.

Author(s)

Q. Huynh

See Also

diagnostic

Examples

```
SP_40_10 <- make_MP(SP, HCR_MSY, diagnostic = "full")
OM <- MSEtool::testOM; OM@proyears <- 20
myMSE <- MSEtool::runMSE(OM = OM, MPs = "SP_40_10")
retrospective_AM(myMSE, MP = "SP_40_10", sim = 1)

# How to get all the estimates
library(dplyr)
assess_estimates <- lapply(1:myMSE@nMPs, function(m) {
  lapply(1:myMSE@nsim, function(x) {
    report <- myMSE@PPD[[m]]@Misc[[x]]$Assessment_report
    if (is.null(report)) {
      return(data.frame())
    } else {
      mutate(report, MP = myMSE@MPs[m], Simulation = x)
    }
  }) %>% bind_rows()
```

```
}) %>% bind_rows()
```

SCA *Statistical catch-at-age (SCA) model*

Description

A generic statistical catch-at-age model (single fleet, single season) that uses catch, index, and catch-at-age composition data. SCA parameterizes R0 and steepness as leading productivity parameters in the assessment model. Recruitment is estimated as deviations from the resulting stock-recruit relationship. In SCA2, the mean recruitment in the time series is estimated and recruitment deviations around this mean are estimated as penalized parameters (SR = "none", similar to Cadigan 2016). The standard deviation is set high so that the recruitment is almost like free parameters. Unfished and MSY reference points are not estimated, it is recommended to use yield per recruit or spawning potential ratio in harvest control rules. SCA_Pope is a variant of SCA that fixes the expected catch to the observed catch, and Pope's approximation is used to calculate the annual exploitation rate (U; i.e., catch_eq = "Pope").

Usage

```
SCA(
  x = 1,
  Data,
  AddInd = "B",
  SR = c("BH", "Ricker", "none"),
  vulnerability = c("logistic", "dome"),
  catch_eq = c("Baranov", "Pope"),
  CAA_dist = c("multinomial", "lognormal"),
  CAA_multiplier = 50,
  rescale = "mean1",
  max_age = Data@MaxAge,
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  fix_F_equilibrium = TRUE,
  fix_omega = TRUE,
  fix_tau = TRUE,
  LWT = list(),
  early_dev = c("comp_onegen", "comp", "all"),
  late_dev = "comp50",
  integrate = FALSE,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  inner.control = list(),
```

```
  ...
)

SCA2(
  x = 1,
  Data,
  AddInd = "B",
  vulnerability = c("logistic", "dome"),
  CAA_dist = c("multinomial", "lognormal"),
  CAA_multiplier = 50,
  rescale = "mean1",
  max_age = Data@MaxAge,
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  fix_F_equilibrium = TRUE,
  fix_omega = TRUE,
  fix_tau = TRUE,
  LWT = list(),
  common_dev = "comp50",
  integrate = FALSE,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  inner.control = list(),
  ...
)

SCA_Pope(
  x = 1,
  Data,
  AddInd = "B",
  SR = c("BH", "Ricker", "none"),
  vulnerability = c("logistic", "dome"),
  CAA_dist = c("multinomial", "lognormal"),
  CAA_multiplier = 50,
  rescale = "mean1",
  max_age = Data@MaxAge,
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  fix_U_equilibrium = TRUE,
  fix_tau = TRUE,
  LWT = list(),
  early_dev = c("comp_onegen", "comp", "all"),
  late_dev = "comp50",
  integrate = FALSE,
```

```
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  inner.control = list(),
  ...
)
```

Arguments

x	A position in the Data object (by default, equal to one for assessments).
Data	An object of class Data
AddInd	A vector of integers or character strings indicating the indices to be used in the model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd. Vulnerability to the survey is fixed in the model.
SR	Stock-recruit function (either "BH" for Beverton-Holt, "Ricker", or "none" for constant mean recruitment).
vulnerability	Whether estimated vulnerability is "logistic" or "dome" (double-normal). See details for parameterization.
catch_eq	Whether to use the Baranov equation or Pope's approximation to calculate the predicted catch at age in the model.
CAA_dist	Whether a multinomial or lognormal distribution is used for likelihood of the catch-at-age matrix. See details.
CAA_multiplier	Numeric for data weighting of catch-at-age matrix if CAA_hist = "multinomial". Otherwise ignored. See details.
rescale	A multiplicative factor that rescales the catch in the assessment model, which can improve convergence. By default, "mean1" scales the catch so that time series mean is 1, otherwise a numeric. Output is re-converted back to original units.
max_age	Integer, the maximum age (plus-group) in the model.
start	Optional list of starting values. Entries can be expressions that are evaluated in the function. See details.
prior	A named list for the parameters of any priors to be added to the model. See below.
fix_h	Logical, whether to fix steepness to value in Data@steep in the model for SCA. This only affects calculation of reference points for SCA2.
fix_F_equilibrium	Logical, whether the equilibrium fishing mortality prior to the first year of the model is estimated. If TRUE, F_equilibrium is fixed to value provided in start (if provided), otherwise, equal to zero (assumes unfished conditions).
fix_omega	Logical, whether the standard deviation of the catch is fixed. If TRUE, omega is fixed to value provided in start (if provided), otherwise, value based on Data@CV_Cat.

fix_tau — Logical, the standard deviation of the recruitment deviations is fixed. If TRUE, tau is fixed to value provided in start (if provided), otherwise, value based on Data@sigmaR.

LWT — A named list (Index, CAA, Catch) of likelihood weights for the data components. For the index, a vector of length survey. For CAL and Catch, a single value.

early_dev — Numeric or character string describing the years for which recruitment deviations are estimated in SCA. By default, equal to "comp_onegen", where rec devs are estimated one full generation prior to the first year when catch-at-age (CAA) data are available. With "comp", rec devs are estimated starting in the first year with CAA. With "all", rec devs start at the beginning of the model. If numeric, the number of years after the first year of the model for which to start estimating rec devs. Use negative numbers for years prior to the first year.

late_dev — Typically, a numeric for the number of most recent years in which recruitment deviations will not be estimated in SCA (recruitment in these years will be based on the mean predicted by stock-recruit relationship). By default, "comp50" uses the number of ages (smaller than the mode) for which the catch-at-age matrix has less than half the abundance than that at the mode.

integrate — Logical, whether the likelihood of the model integrates over the likelihood of the recruitment deviations (thus, treating it as a random effects/state-space variable). Otherwise, recruitment deviations are penalized parameters.

silent — Logical, passed to TMB::MakeADFun(), whether TMB will print trace information during optimization. Used for diagnostics for model convergence.

opt_hess — Logical, whether the hessian function will be passed to stats::nlminb() during optimization (this generally reduces the number of iterations to convergence, but is memory and time intensive and does not guarantee an increase in convergence rate). Ignored if integrate = TRUE.

n_restart — The number of restarts (calls to stats::nlminb()) in the optimization procedure, so long as the model hasn't converged. The optimization continues from the parameters from the previous (re)start.

control — A named list of arguments for optimization to be passed to stats::nlminb().

inner.control — A named list of arguments for optimization of the random effects, which is passed on to TMB::newton().

... — Other arguments to be passed, including yind (an expression for the vector of years to include in the model, useful for debugging for data lags), M_at_age (set to TRUE to specify a matrix of M by year and age from the operating model and the bias parameter), IAA_hist (an array of index age proportions by year, age, survey), and IAA_n (a matrix of multinomial sample size by year and survey).

common_dev — Typically, a numeric for the number of most recent years in which a common recruitment deviation will be estimated (in SCA2, uninformative years will have a recruitment closer to the mean, which can be very misleading, especially near the end of the time series). By default, "comp50" uses the number of ages (smaller than the mode) for which the catch-at-age matrix has less than half the abundance than that at the mode.

fix_U_equilibrium — Logical, same as fix_F_equilibrium for SCA_Pope.

Details

The basic data inputs are catch (by weight), index (by weight/biomass), and catch-at-age matrix (by numbers).

With `catch_eq = "Baranov"` (default in SCA and SCA2), annual F's are estimated parameters assuming continuous fishing over the year, while an annual exploitation rate from pulse fishing in the middle of the year is estimated in `SCA_Pope` or `SCA(catch_eq = "Pope")`.

The annual sample sizes of the catch-at-age matrix is provided to the model (used in the likelihood for catch-at-age assuming a multinomial distribution) and is manipulated via argument `CAA_multiplier`. This argument is interpreted in two different ways depending on the value provided. If `CAA_multiplier > 1`, then this value will cap the annual sample sizes to that number. If `CAA_multiplier <= 1`, then all the annual samples sizes will be re-scaled by that number, e.g. `CAA_multiplier = 0.1` multiplies the sample size to 10% of the original number. By default, sample sizes are capped at 50.

Alternatively, a lognormal distribution with inverse proportion variance can be used for the catch at age (Punt and Kennedy, 1994, as cited by Maunder 2011).

For `start` (optional), a named list of starting values of estimates can be provided for:

- `R0` Unfished recruitment, except when `SR = "none"` where it is mean recruitment. By default, 150% `Data@OM$R0[x]` is used as the start value in closed-loop simulation, and 400% of mean catch otherwise.
- `h` Steepness. Otherwise, `Data@steep[x]` is used, or 0.9 if empty.
- `M` Natural mortality. Otherwise, `Data@Mort[x]` is used.
- `vul_par` Vulnerability parameters, see next paragraph.
- `F` A vector of length `nyears` for year-specific fishing mortality.
- `F_equilibrium` Equilibrium fishing mortality leading into first year of the model (to determine initial depletion). By default, 0.
- `U_equilibrium` Same as F_equilibrium when `catch_eq = "Pope"`. By default, 0.
- `omega` Lognormal SD of the catch (observation error) when `catch_eq = "Baranov"`. By default, `Data@CV_Cat[x]`.
- `tau` Lognormal SD of the recruitment deviations (process error). By default, `Data@sigmaR[x]`.

Vulnerability can be specified to be either logistic or dome. If logistic, then the parameter vector `vul_par` is of length 2:

- `vul_par[1]` corresponds to `a_95`, the age of 95% vulnerability. `a_95` is a transformed parameter via logit transformation to constrain `a_95` to less than 75% of the maximum age: `a_95 = 0.75 * max_age * plogis(x[1])`, where `x` is the estimated vector.
- `vul_par[2]` corresponds to `a_50`, the age of 50% vulnerability. Estimated as an offset, i.e., `a_50 = a_95 - exp(x[2])`.

With dome vulnerability, a double Gaussian parameterization is used, where `vul_par` is an estimated vector of length 4:

- `vul_par[1]` corresponds to `a_asc`, the first age of full vulnerability for the ascending limb. In the model, `a_asc` is estimated via logit transformation to constrain `a_95` to less than 75% of the maximum age: `a_asc = 0.75 * maxage * plogis(x[1])`, where `x` is the estimated vector.

- vul_par[2] corresponds to a_50, the age of 50% vulnerability for the ascending limb. Estimated as an offset, i.e., a_50 = a_asc - exp(x[2]).
- vul_par[3] corresponds to a_des, the last age of full vulnerability (where the descending limb starts). Generated via logit transformation to constrain between a_asc and max_age, i.e., a_des = (max_age - a_asc) * plogis(x[3]) + a_asc. By default, fixed to a small value so that the dome is effectively a three-parameter function.
- vul_par[4] corresponds to vul_max, the vulnerability at the maximum age. Estimated in logit space: vul_max = plogis(x[4]).

Vague priors of vul_par[1] ~ N(0, sd = 3), vul_par[2] ~ N(0, 3), vul_par[3] ~ Beta(1.01, 1.01) are used to aid convergence when parameters may not be well estimated, for example, when vulnerability » 0.5 for the youngest age class.

Value

An object of class Assessment.

Priors

The following priors can be added as a named list, e.g., prior = list(M = c(0.25, 0.15), h = c(0.7, 0.1). For each parameter below, provide a vector of values as described:

- R0 - A vector of length 3. The first value indicates the distribution of the prior: 1 for lognormal, 2 for uniform on log(R0), 3 for uniform on R0. If lognormal, the second and third values are the prior mean (in normal space) and SD (in log space). Otherwise, the second and third values are the lower and upper bounds of the uniform distribution (values in normal space).
- h - A vector of length 2 for the prior mean and SD, both in normal space. Beverton-Holt steepness uses a beta distribution, while Ricker steepness uses a normal distribution.
- M - A vector of length 2 for the prior mean (in normal space) and SD (in log space). Lognormal prior.
- q - A matrix for nsurvey rows and 2 columns. The first column is the prior mean (in normal space) and the second column for the SD (in log space). Use NA in rows corresponding to indices without priors.

See online documentation for more details.

Online Documentation

Model description and equations are available on the openMSE website.

Required Data

- SCA, SCA_Pope, and SCA_Pope: Cat, Ind, Mort, L50, L95, CAA, vbK, vbLinf, vbt0, wla, wlb, MaxAge

Optional Data

- SCA: Rec, steep, sigmaR, CV_Ind, CV_Cat
- SCA2: Rec, steep, CV_Ind, CV_Cat
- SCA_Pope: Rec, steep, sigmaR, CV_Ind

Author(s)

Q. Huynh

References

Cadigan, N.G. 2016. A state-space stock assessment model for northern cod, including under-reported catches and variable natural mortality rates. Canadian Journal of Fisheries and Aquatic Science 72:296-308.

Maunder, M.N. 2011. Review and evaluation of likelihood functions for composition data in stock-assessment models: Estimating the effective sample size. Fisheries Research 209:311-319.

Punt, A.E. and Kennedy, R.B. 1997. Population modelling of Tasmanian rock lobster, Jasus edwardsii, resources. Marine and Freshwater Research 48:967-980.

See Also

plot.Assessment summary.Assessment retrospective profile make_MP

Examples

```
res <- SCA(Data = MSEtool::SimulatedData)
res2 <- SCA2(Data = MSEtool::SimulatedData)

# Downweight the index
res3 <- SCA(Data = MSEtool::SimulatedData, LWT = list(Index = 0.1, CAA = 1))

compare_models(res, res2)
```

SCA_CAL *Age-structured model using fishery length composition*

Description

A single-fleet assessment that fits to catch, indices of abundance, and fishery length compositions. See SCA for all details.

Usage

```
SCA_CAL(
  x = 1,
  Data,
  AddInd = "B",
  SR = c("BH", "Ricker", "none"),
  vulnerability = c("logistic", "dome"),
  catch_eq = c("Baranov", "Pope"),
  CAL_dist = c("multinomial", "lognormal"),
  CAL_multiplier = 50,
  rescale = "mean1",
```

```
  max_age = Data@MaxAge,
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  fix_F_equilibrium = TRUE,
  fix_omega = TRUE,
  fix_tau = TRUE,
  LWT = list(),
  early_dev = c("comp_onegen", "comp", "all"),
  late_dev = "comp50",
  integrate = FALSE,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  inner.control = list(),
  ...
)
```

Arguments

x	A position in the Data object (by default, equal to one for assessments).
Data	An object of class Data
AddInd	A vector of integers or character strings indicating the indices to be used in the model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd. Vulnerability to the survey is fixed in the model.
SR	Stock-recruit function (either "BH" for Beverton-Holt, "Ricker", or "none" for constant mean recruitment).
vulnerability	Whether estimated vulnerability is "logistic" or "dome" (double-normal). See details for parameterization.
catch_eq	Whether to use the Baranov equation or Pope's approximation to calculate the predicted catch at age in the model.
CAL_dist	Character, the statistical distribution for the likelihood of the catch-at-length.
CAL_multiplier	Numeric for data weighting of catch-at-length matrix if CAL_hist = "multinomial". A value smaller than one rescales annual sample sizes to this fraction of the original sample size. Values greater than one generates a cap of the annual sample size to this value.
rescale	A multiplicative factor that rescales the catch in the assessment model, which can improve convergence. By default, "mean1" scales the catch so that time series mean is 1, otherwise a numeric. Output is re-converted back to original units.
max_age	Integer, the maximum age (plus-group) in the model.
start	Optional list of starting values. Entries can be expressions that are evaluated in the function. See details.

prior — A named list for the parameters of any priors to be added to the model. See below.

fix_h — Logical, whether to fix steepness to value in Data@steep in the model for SCA. This only affects calculation of reference points for SCA2.

fix_F_equilibrium — Logical, whether the equilibrium fishing mortality prior to the first year of the model is estimated. If TRUE, F_equilibrium is fixed to value provided in start (if provided), otherwise, equal to zero (assumes unfished conditions).

fix_omega — Logical, whether the standard deviation of the catch is fixed. If TRUE, omega is fixed to value provided in start (if provided), otherwise, value based on Data@CV_Cat.

fix_tau — Logical, the standard deviation of the recruitment deviations is fixed. If TRUE, tau is fixed to value provided in start (if provided), otherwise, value based on Data@sigmaR.

LWT — A named list (Index, CAA, Catch) of likelihood weights for the data components. For the index, a vector of length survey. For CAL and Catch, a single value.

early_dev — Numeric or character string describing the years for which recruitment deviations are estimated in SCA. By default, equal to "comp_onegen", where rec devs are estimated one full generation prior to the first year when catch-at-age (CAA) data are available. With "comp", rec devs are estimated starting in the first year with CAA. With "all", rec devs start at the beginning of the model. If numeric, the number of years after the first year of the model for which to start estimating rec devs. Use negative numbers for years prior to the first year.

late_dev — Typically, a numeric for the number of most recent years in which recruitment deviations will not be estimated in SCA (recruitment in these years will be based on the mean predicted by stock-recruit relationship). By default, "comp50" uses the number of ages (smaller than the mode) for which the catch-at-age matrix has less than half the abundance than that at the mode.

integrate — Logical, whether the likelihood of the model integrates over the likelihood of the recruitment deviations (thus, treating it as a random effects/state-space variable). Otherwise, recruitment deviations are penalized parameters.

silent — Logical, passed to TMB::MakeADFun(), whether TMB will print trace information during optimization. Used for diagnostics for model convergence.

opt_hess — Logical, whether the hessian function will be passed to stats::nlminb() during optimization (this generally reduces the number of iterations to convergence, but is memory and time intensive and does not guarantee an increase in convergence rate). Ignored if integrate = TRUE.

n_restart — The number of restarts (calls to stats::nlminb()) in the optimization procedure, so long as the model hasn't converged. The optimization continues from the parameters from the previous (re)start.

control — A named list of arguments for optimization to be passed to stats::nlminb().

inner.control — A named list of arguments for optimization of the random effects, which is passed on to TMB::newton().

... Other arguments to be passed, including `yind` (an expression for the vector of years to include in the model, useful for debugging for data lags), `M_at_age` (set to TRUE to specify a matrix of M by year and age from the operating model and the bias parameter), `IAA_hist` (an array of index age proportions by year, age, survey), and `IAA_n` (a matrix of multinomial sample size by year and survey).

Online Documentation

Model description and equations are available on the openMSE website.

Author(s)

Q. Huynh

`SCA_DDM` *SCA models with time-varying natural mortality*

Description

A modification of `SCA` that incorporates density-dependent effects on M based on biomass depletion (Forrest et al. 2018). Set the bounds of M in the `M_bounds` argument, a length-2 vector where the first entry is M0, the M as B/B0 >= 1, and the second entry is M1, the M as B/B0 approaches zero. Note that M0 can be greater than M1 (compensatory) or M0 can be less than M1 (depensatory).

Usage

```
SCA_DDM(
  x = 1,
  Data,
  AddInd = "B",
  SR = c("BH", "Ricker", "none"),
  vulnerability = c("logistic", "dome"),
  catch_eq = c("Baranov", "Pope"),
  CAA_dist = c("multinomial", "lognormal"),
  CAA_multiplier = 50,
  rescale = "mean1",
  max_age = Data@MaxAge,
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  fix_F_equilibrium = TRUE,
  fix_omega = TRUE,
  fix_tau = TRUE,
  LWT = list(),
  early_dev = c("comp_onegen", "comp", "all"),
  late_dev = "comp50",
  M_bounds = NULL,
```

```
  integrate = FALSE,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  inner.control = list(),
  ...
)
```

Arguments

x	A position in the Data object (by default, equal to one for assessments).
Data	An object of class Data
AddInd	A vector of integers or character strings indicating the indices to be used in the model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd. Vulnerability to the survey is fixed in the model.
SR	Stock-recruit function (either "BH" for Beverton-Holt, "Ricker", or "none" for constant mean recruitment).
vulnerability	Whether estimated vulnerability is "logistic" or "dome" (double-normal). See details for parameterization.
catch_eq	Whether to use the Baranov equation or Pope's approximation to calculate the predicted catch at age in the model.
CAA_dist	Whether a multinomial or lognormal distribution is used for likelihood of the catch-at-age matrix. See details.
CAA_multiplier	Numeric for data weighting of catch-at-age matrix if CAA_hist = "multinomial". Otherwise ignored. See details.
rescale	A multiplicative factor that rescales the catch in the assessment model, which can improve convergence. By default, "mean1" scales the catch so that time series mean is 1, otherwise a numeric. Output is re-converted back to original units.
max_age	Integer, the maximum age (plus-group) in the model.
start	Optional list of starting values. Entries can be expressions that are evaluated in the function. See details.
prior	A named list for the parameters of any priors to be added to the model. See below.
fix_h	Logical, whether to fix steepness to value in Data@steep in the model for SCA. This only affects calculation of reference points for SCA2.
fix_F_equilibrium	Logical, whether the equilibrium fishing mortality prior to the first year of the model is estimated. If TRUE, F_equilibrium is fixed to value provided in start (if provided), otherwise, equal to zero (assumes unfished conditions).
fix_omega	Logical, whether the standard deviation of the catch is fixed. If TRUE, omega is fixed to value provided in start (if provided), otherwise, value based on Data@CV_Cat.

fix_tau	Logical, the standard deviation of the recruitment deviations is fixed. If TRUE, tau is fixed to value provided in start (if provided), otherwise, value based on Data@sigmaR.
LWT	A named list (Index, CAA, Catch) of likelihood weights for the data components. For the index, a vector of length survey. For CAL and Catch, a single value.
early_dev	Numeric or character string describing the years for which recruitment deviations are estimated in SCA. By default, equal to "comp_onegen", where rec devs are estimated one full generation prior to the first year when catch-at-age (CAA) data are available. With "comp", rec devs are estimated starting in the first year with CAA. With "all", rec devs start at the beginning of the model. If numeric, the number of years after the first year of the model for which to start estimating rec devs. Use negative numbers for years prior to the first year.
late_dev	Typically, a numeric for the number of most recent years in which recruitment deviations will not be estimated in SCA (recruitment in these years will be based on the mean predicted by stock-recruit relationship). By default, "comp50" uses the number of ages (smaller than the mode) for which the catch-at-age matrix has less than half the abundance than that at the mode.
M_bounds	A numeric vector of length 2 to indicate the M as B/B0 approaches zero and one, respectively. By default, set to 75% and 125%, respectively, of Data@Mort[x].
integrate	Logical, whether the likelihood of the model integrates over the likelihood of the recruitment deviations (thus, treating it as a random effects/state-space variable). Otherwise, recruitment deviations are penalized parameters.
silent	Logical, passed to TMB::MakeADFun(), whether TMB will print trace information during optimization. Used for diagnostics for model convergence.
opt_hess	Logical, whether the hessian function will be passed to stats::nlminb() during optimization (this generally reduces the number of iterations to convergence, but is memory and time intensive and does not guarantee an increase in convergence rate). Ignored if integrate = TRUE.
n_restart	The number of restarts (calls to stats::nlminb()) in the optimization procedure, so long as the model hasn't converged. The optimization continues from the parameters from the previous (re)start.
control	A named list of arguments for optimization to be passed to stats::nlminb().
inner.control	A named list of arguments for optimization of the random effects, which is passed on to TMB::newton().
...	Other arguments to be passed, including yind (an expression for the vector of years to include in the model, useful for debugging for data lags), M_at_age (set to TRUE to specify a matrix of M by year and age from the operating model and the bias parameter), IAA_hist (an array of index age proportions by year, age, survey), and IAA_n (a matrix of multinomial sample size by year and survey).

Details

See SCA for more information on all arguments.

Value

An object of class Assessment.

Online Documentation

Model description and equations are available on the openMSE website.

Author(s)

Q. Huynh

References

Forrest, R.E., Holt, K.R., and Kronlund, A.R. 2018. Performance of alternative harvest control rules for two Pacific groundfish stocks with uncertain natural mortality: Bias, robustness and trade-offs. Fisheries Research 2016: 259-286.

See Also

SCA SCA_RWM plot.Assessment summary.Assessment retrospective profile make_MP

Examples

```
res <- SCA_DDM(Data = MSEtool::SimulatedData)
```

SCA_RWM *SCA with random walk in M*

Description

SCA_RWM is a modification of SCA that incorporates a random walk in M in logit space (constant with age). Set the variance (start$tau_M) to a small value (0.001) in order to fix M for all years, which is functionally equivalent to SCA.

Usage

```
SCA_RWM(
  x = 1,
  Data,
  AddInd = "B",
  SR = c("BH", "Ricker", "none"),
  vulnerability = c("logistic", "dome"),
  catch_eq = c("Baranov", "Pope"),
  CAA_dist = c("multinomial", "lognormal"),
  CAA_multiplier = 50,
  rescale = "mean1",
  max_age = Data@MaxAge,
```

```
  start = NULL,
  prior = list(),
  fix_h = TRUE,
  fix_F_equilibrium = TRUE,
  fix_omega = TRUE,
  fix_tau = TRUE,
  LWT = list(),
  early_dev = c("comp_onegen", "comp", "all"),
  late_dev = "comp50",
  refyear = expression(length(Data@Year)),
  M_bounds = NULL,
  integrate = FALSE,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  inner.control = list(),
  ...
)
```

Arguments

x	A position in the Data object (by default, equal to one for assessments).
Data	An object of class Data
AddInd	A vector of integers or character strings indicating the indices to be used in the model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd. Vulnerability to the survey is fixed in the model.
SR	Stock-recruit function (either "BH" for Beverton-Holt, "Ricker", or "none" for constant mean recruitment).
vulnerability	Whether estimated vulnerability is "logistic" or "dome" (double-normal). See details for parameterization.
catch_eq	Whether to use the Baranov equation or Pope's approximation to calculate the predicted catch at age in the model.
CAA_dist	Whether a multinomial or lognormal distribution is used for likelihood of the catch-at-age matrix. See details.
CAA_multiplier	Numeric for data weighting of catch-at-age matrix if CAA_hist = "multinomial". Otherwise ignored. See details.
rescale	A multiplicative factor that rescales the catch in the assessment model, which can improve convergence. By default, "mean1" scales the catch so that time series mean is 1, otherwise a numeric. Output is re-converted back to original units.
max_age	Integer, the maximum age (plus-group) in the model.
start	Optional list of starting values. Entries can be expressions that are evaluated in the function. See details.

prior — A named list for the parameters of any priors to be added to the model. See below.

fix_h — Logical, whether to fix steepness to value in Data@steep in the model for SCA. This only affects calculation of reference points for SCA2.

fix_F_equilibrium — Logical, whether the equilibrium fishing mortality prior to the first year of the model is estimated. If TRUE, F_equilibrium is fixed to value provided in start (if provided), otherwise, equal to zero (assumes unfished conditions).

fix_omega — Logical, whether the standard deviation of the catch is fixed. If TRUE, omega is fixed to value provided in start (if provided), otherwise, value based on Data@CV_Cat.

fix_tau — Logical, the standard deviation of the recruitment deviations is fixed. If TRUE, tau is fixed to value provided in start (if provided), otherwise, value based on Data@sigmaR.

LWT — A named list (Index, CAA, Catch) of likelihood weights for the data components. For the index, a vector of length survey. For CAL and Catch, a single value.

early_dev — Numeric or character string describing the years for which recruitment deviations are estimated in SCA. By default, equal to "comp_onegen", where rec devs are estimated one full generation prior to the first year when catch-at-age (CAA) data are available. With "comp", rec devs are estimated starting in the first year with CAA. With "all", rec devs start at the beginning of the model. If numeric, the number of years after the first year of the model for which to start estimating rec devs. Use negative numbers for years prior to the first year.

late_dev — Typically, a numeric for the number of most recent years in which recruitment deviations will not be estimated in SCA (recruitment in these years will be based on the mean predicted by stock-recruit relationship). By default, "comp50" uses the number of ages (smaller than the mode) for which the catch-at-age matrix has less than half the abundance than that at the mode.

refyear — An expression for the year for which M is used to report MSY and unfished reference points. By default, terminal year. If multiple years are provided, then the mean M over the specified time period is used.

M_bounds — A numeric vector of length 2 to indicate the minimum and maximum M in the random walk as a proportion of the starting M (start$M). The default min and max are 75% and 125%, respectively.

integrate — Logical, whether the likelihood of the model integrates over the likelihood of the recruitment deviations (thus, treating it as a random effects/state-space variable). Otherwise, recruitment deviations are penalized parameters.

silent — Logical, passed to TMB::MakeADFun(), whether TMB will print trace information during optimization. Used for diagnostics for model convergence.

opt_hess — Logical, whether the hessian function will be passed to stats::nlminb() during optimization (this generally reduces the number of iterations to convergence, but is memory and time intensive and does not guarantee an increase in convergence rate). Ignored if integrate = TRUE.

n_restart	The number of restarts (calls to stats::nlminb()) in the optimization procedure, so long as the model hasn't converged. The optimization continues from the parameters from the previous (re)start.
control	A named list of arguments for optimization to be passed to stats::nlminb().
inner.control	A named list of arguments for optimization of the random effects, which is passed on to TMB::newton().
...	Other arguments to be passed, including yind (an expression for the vector of years to include in the model, useful for debugging for data lags), M_at_age (set to TRUE to specify a matrix of M by year and age from the operating model and the bias parameter), IAA_hist (an array of index age proportions by year, age, survey), and IAA_n (a matrix of multinomial sample size by year and survey).

Details

The model estimates year-specific M (constant with age) as a random walk in logit space, bounded by a proportion of start$M (specified in M_bounds).

The starting value for the first year M (start$M) is Data@Mort[x] and is fixed, unless a prior is provided (prior$M). The fixed SD of the random walk (tau_M) is 0.05, by default.

Steepness and unfished recruitment in the estimation model, along with unfished reference points, correspond to spawners per recruit using the first year M. With argument refyear, new unfished reference points and steepness values are calculated. See examples.

Alternative values can be provided in the start list (see examples):

- R0 Unfished recruitment, except when SR = "none" where it is mean recruitment. By default, 150% Data@OM$R0[x] is used as the start value in closed-loop simulation, and 400\
- h Steepness. Otherwise, Data@steep[x] is used, or 0.9 if empty.
- M Natural mortality in the first year. Otherwise, Data@Mort[x] is used.
- vul_par Vulnerability parameters, see next paragraph.
- F A vector of length nyears for year-specific fishing mortality.
- F_equilibrium Equilibrium fishing mortality leading into first year of the model (to determine initial depletion). By default, 0.
- omega Lognormal SD of the catch (observation error) when catch_eq = "Baranov". By default, Data@CV_Cat[x].
- tau Lognormal SD of the recruitment deviations (process error). By default, Data@sigmaR[x].
- tau_M The fixed SD of the random walk in M. By default, 0.05.

See SCA for all other information about the structure and setup of the model.

The SCA builds in a stock-recruit relationship into the model. Annual unfished and MSY reference points are calculated and reported in TMB_report of the Assessment object.

Value

An object of class Assessment.

Online Documentation

Model description and equations are available on the openMSE website.

Author(s)

Q. Huynh

See Also

SCA SCA_DDM

Examples

```
res <- SCA_RWM(Data = MSEtool::SimulatedData, start = list(M_start = 0.4, tau_M = 0.05))
res2 <- SCA(Data = MSEtool::SimulatedData)
res3 <- SCA_RWM(Data = MSEtool::SimulatedData, start = list(M_start = 0.4, tau_M = 0.001))

# Use mean M in most recent 5 years for reporting reference points
res_5r <- SCA_RWM(Data = MSEtool::SimulatedData,
                  refyear = expression(seq(length(Data@Year) - 4, length(Data@Year))),
                  start = list(M_start = 0.4, tau_M = 0.001))
res_5r@SSB0 # SSB0 reported (see also res_5r@TMB_report$new_E0)
res_5r@TMB_report$E0 # SSB0 of Year 1 M


compare_models(res, res2, res3)
```

Shortcut — *Assessment emulator as a shortcut to model fitting in closed-loop simulation*

Description

Functions (class Assessment) that emulate a stock assessment by sampling the operating model biomass, abundance, and fishing mortality (with observation error, autocorrelation, and bias) instead of fitting a model. This output can then be passed onto a harvest control rule (HCR function). `Shortcut` is the base function that samples the OM with an error distribution. `Shortcut2`, the more preferable option, fits SCA in the last historical year of the operating model, estimates the error parameters using a vector autoregressive model of the residuals, and then generates model "estimates" using predict.varest. `Perfect` assumes no error in the assessment model and is useful for comparing the behavior of different harvest control rules. To utilize the shortcut method in closed-loop simulation, use make_MP with these functions as the Assessment model. **N.B. the functions do not work with** `runMSE(parallel = TRUE)` for MSEtool v3.4.0 and earlier.

Usage

```
Shortcut(
  x = 1,
  Data,
  method = c("B", "N", "RF"),
  B_err = c(0.3, 0.7, 1),
  N_err = c(0.3, 0.7, 1),
  R_err = c(0.3, 0.7, 1),
  F_err = c(0.3, 0.7, 1),
  VAR_model,
  ...
)

Shortcut2(
  x,
  Data,
  method = "N",
  SCA_args = list(),
  VAR_args = list(type = "none"),
  ...
)

Perfect(x, Data, ...)
```

Arguments

`x`	An index for the objects in `Data` when running in runMSE. Otherwise, equals to 1 When running an assessment interactively.
`Data`	An object of class Data.
`method`	Indicates where the error in the OM is located. For "B", OM biomass is directly sampled with error. For "N", OM abundance-at-age is sampled and biomass subsequently calculated. For "RF", recruitment and F are sampled to calculate abundance and biomass. There is no error in biological parameters for "N" and "RF". By default, "B" is used for `Shortcut` and "N" for `Shortcut2`.
`B_err`	If `method = "B"`, a vector of length three that specifies the standard deviation (in logspace), autocorrelation, and bias (1 = unbiased) for biomass.
`N_err`	Same as B_err, but for abundance when `method = "N"`.
`R_err`	Same as B_err, but for recruitment when `method = "RF"`.
`F_err`	Same as B_err. Always used regardless of `method` to report F and selectivity for HCR.
`VAR_model`	An object returned by VAR to generate emulated assessment error. Used by `Shortcut2`.
`...`	Other arguments (not currently used).
`SCA_args`	Additional arguments to pass to SCA. Currently, arguments `SR` and `vulnerability` are obtained from the operating model.

VAR_args — Additional arguments to pass to VAR. By default, argument type = "none" (stationary time series with mean zero is assumed).

Details

Currently there is no error in FMSY (frequently the target F in the HCR).

See Wiedenmann et al. (2015) for guidance on the magnitude of error for the shortcut emulator.

Value

An object of class Assessment.

Author(s)

Q. Huynh

References

Wiedenmann, J., Wilberg, M.J., Sylvia, A., and Miller, T.J. 2015. Autocorrelated error in stock assessment estimates: Implications for management strategy evaluation. Fisheries Research 172: 325-334.

Examples

```
Shortcut_4010 <- make_MP(Shortcut, HCR40_10)
Shortcut_Nerr <- make_MP(Shortcut, HCR40_10, method = "N", N_err = c(0.1, 0.1, 1)) # Highly precise!

# Fits SCA first and then emulate it in the projection period
Shortcut2_4010 <- make_MP(Shortcut2, HCR40_10)


# Compare the shortcut method vs. fitting an SCA model with a 40-10 control rule
MSE <- runMSE(testOM, MPs = c("Shortcut_4010", "SCA_4010"))


# Compare the performance of three HCRs
Perfect_4010 <- make_MP(Perfect, HCR40_10)
Perfect_6020 <- make_MP(Perfect, HCR60_20)
Perfect_8040MSY <- make_MP(Perfect, HCR_ramp, OCP_type = "SSB_SSBMSY", TOCP = 0.8, LOCP = 0.4)


MSE <- runMSE(testOM, MPs = c("Perfect_4010", "Perfect_6020", "Perfect_8040MSY"))
```

sim-class *Class*-sim

Description

An S4 class that contains output from simulate.

Slots

`Model` Name of the assessment model.

`data` List of data from the assessment.

`data_sim` List of simulated data values. Each value returns an array.

`process_sim` List of simulated process error.

`est` Estimates from the original model fit.

`est_sim` Estimates from the simulated data.

Author(s)

Q. Huynh

simulate *Generate simulated data from TMB models in SAMtool*

Description

A convenient wrapper function (`simulate`) to simulate data (and process error) from the likelihood function.

Usage

```
simulate(object, ...)

## S4 method for signature 'Assessment'
simulate(
  object,
  nsim = 1,
  seed = NULL,
  process_error = FALSE,
  refit = FALSE,
  cores = 1,
  ...
)

## S4 method for signature 'RCModel'
```

```
simulate(
  object,
  nsim = 1,
  seed = NULL,
  process_error = FALSE,
  refit = FALSE,
  cores = 1,
  ...
)
```

Arguments

object	An object of class Assessment or RCModel containing the fitted model.
...	Additional arguments
nsim	Number of simulations
seed	Used for the random number generator
process_error	Logical, indicates if process error is re-sampled in the simulation.
refit	Logical, whether to re-fit the model for each simulated dataset.
cores	The number of CPUs for parallel processing for model re-fitting if refit = TRUE.

Details

Process error, e.g., recruitment deviations, will be re-sampled in the simulation.

Value

A sim object returning the original data, simulated data, original parameters, parameters estimated from simulated data, and process error used to simulate data. then a nested list of model output (opt, SD, and report).

Author(s)

Q. Huynh

SP *Surplus production model with FMSY and MSY as leading parameters*

Description

A surplus production model that uses only a time-series of catches and a relative abundance index and coded in TMB. The base model, SP, is conditioned on catch and estimates a predicted index. Continuous surplus production and fishing is modeled with sub-annual time steps which should approximate the behavior of ASPIC (Prager 1994). The Fox model, SP_Fox, fixes BMSY/K = 0.37 (1/e). The state-space version, SP_SS estimates annual deviates in biomass. An option allows for setting a prior for the intrinsic rate of increase. The function for the spict model (Pedersen and Berg, 2016) is available in MSEextra.

Usage

```
SP(
  x = 1,
  Data,
  AddInd = "B",
  rescale = "mean1",
  start = NULL,
  prior = list(),
  fix_dep = TRUE,
  fix_n = TRUE,
  LWT = NULL,
  n_seas = 4L,
  n_itF = 3L,
  Euler_Lotka = 0L,
  SR_type = c("BH", "Ricker"),
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 5000, eval.max = 10000),
  ...
)

SP_SS(
  x = 1,
  Data,
  AddInd = "B",
  rescale = "mean1",
  start = NULL,
  prior = list(),
  fix_dep = TRUE,
  fix_n = TRUE,
  fix_sigma = TRUE,
  fix_tau = TRUE,
  LWT = NULL,
  early_dev = c("all", "index"),
  n_seas = 4L,
  n_itF = 3L,
  Euler_Lotka = 0L,
  SR_type = c("BH", "Ricker"),
  integrate = FALSE,
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 5000, eval.max = 10000),
  inner.control = list(),
  ...
)
```

```
SP_Fox(x = 1, Data, ...)
```

Arguments

x	An index for the objects in Data when running in runMSE. Otherwise, equals to 1 When running an assessment interactively.
Data	An object of class Data.
AddInd	A vector of integers or character strings indicating the indices to be used in the model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd.
rescale	A multiplicative factor that rescales the catch in the assessment model, which can improve convergence. By default, "mean1" scales the catch so that time series mean is 1, otherwise a numeric. Output is re-converted back to original units.
start	Optional list of starting values. Entries can be expressions that are evaluated in the function. See details.
prior	A named list for the parameters of any priors to be added to the model. See details.
fix_dep	Logical, whether to fix the initial depletion (ratio of biomass to carrying capacity in the first year of the model). If TRUE, uses the value in start, otherwise equal to 1 (unfished conditions).
fix_n	Logical, whether to fix the exponent of the production function. If TRUE, uses the value in start, otherwise equal to n = 2, where the biomass at MSY is half of carrying capacity.
LWT	A vector of likelihood weights for each survey.
n_seas	Integer, the number of seasons in the model for calculating continuous surplus production.
n_itF	Integer, the number of iterations to solve F conditional on the observed catch given multiple seasons within an annual time step. Ignored if n_seas = 1.
Euler_Lotka	Integer. If greater than zero, the function will calculate a prior for the intrinsic rate of increase to use in the estimation model (in lieu of an explicit prior in argument prior). The value of this argument specifies the number of stochastic samples used to calculate the prior SD. See section on priors below.
SR_type	If use_r_prior = TRUE, the stock-recruit relationship used to calculate the stock-recruit alpha parameter from steepness and unfished spawners-per-recruit. Used to develop the r prior.
silent	Logical, passed to TMB::MakeADFun(), whether TMB will print trace information during optimization. Used for diagnostics for model convergence.
opt_hess	Logical, whether the hessian function will be passed to stats::nlminb() during optimization (this generally reduces the number of iterations to convergence, but is memory and time intensive and does not guarantee an increase in convergence rate). Ignored if integrate = TRUE.

n_restart	The number of restarts (calls to stats::nlminb()) in the optimization procedure, so long as the model hasn't converged. The optimization continues from the parameters from the previous (re)start.
control	A named list of parameters regarding optimization to be passed to stats::nlminb().
...	For SP_Fox, additional arguments to pass to SP.
fix_sigma	Logical, whether the standard deviation of the index is fixed. If TRUE, sigma is fixed to value provided in start (if provided), otherwise, value based on Data@CV_Ind.
fix_tau	Logical, the standard deviation of the biomass deviations is fixed. If TRUE, tau is fixed to value provided in start (if provided), otherwise, equal to 0.1.
early_dev	Character string describing the years for which biomass deviations are estimated in SP_SS. By default, deviations are estimated in each year of the model ("all"), while deviations could also be estimated once index data are available ("index").
integrate	Logical, whether the likelihood of the model integrates over the likelihood of the biomass deviations (thus, treating it as a state-space variable).
inner.control	A named list of arguments for optimization of the random effects, which is passed on to newton via TMB::MakeADFun().

Details

For start (optional), a named list of starting values of estimates can be provided for:

- MSY Maximum sustainable yield.. Otherwise, 300% of mean catch by default.
- FMSY Steepness. Otherwise, Data@Mort[x] or 0.2 is used.
- dep Initial depletion (B/B0) in the first year of the model. By default, 1.
- n The production function exponent that determines BMSY/B0. By default, 2 so that BMSY/B0 = 0.5.
- sigma Lognormal SD of the index (observation error). By default, 0.05. Not used with multiple indices.
- tau Lognormal SD of the biomass deviations (process error) in SP_SS. By default, 0.1.

Multiple indices are supported in the model.

Tip: to create the Fox model (Fox 1970), just fix n = 1. See example.

Value

An object of Assessment containing objects and output from TMB.

Priors

The following priors can be added as a named list, e.g., prior = list(r = c(0.25, 0.15), MSY = c(50, 0.1). For each parameter below, provide a vector of values as described:

- r - A vector of length 2 for the lognormal prior mean (normal space) and SD (lognormal space).

- MSY - A vector of length 2 for the lognormal prior mean (normal space) and SD (lognormal space).

In lieu of an explicit r prior provided by the user, set argument Euler_Lotka = TRUE to calculate the prior mean and SD using the Euler-Lotka method (Equation 15a of McAllister et al. 2001). The Euler-Lotka method is modified to multiply the left-hand side of equation 15a by the alpha parameter of the stock-recruit relationship (Stanley et al. 2009). Natural mortality and steepness are sampled in order to generate a prior distribution for r. See vignette("Surplus_production") for more details.

Online Documentation

Model description and equations are available on the openMSE website.

Required Data

- SP: Cat, Ind
- SP_SS: Cat, Ind

Optional Data

SP_SS: CV_Ind

Note

The model uses the Fletcher (1978) formulation and is parameterized with FMSY and MSY as leading parameters. The default conditions assume unfished conditions in the first year of the time series and a symmetric production function (n = 2).

Author(s)

Q. Huynh

References

Fletcher, R. I. 1978. On the restructuring of the Pella-Tomlinson system. Fishery Bulletin 76:515:521.

Fox, W.W. 1970. An exponential surplus-yield model for optimizing exploited fish populations. Transactions of the American Fisheries Society 99:80-88.

McAllister, M.K., Pikitch, E.K., and Babcock, E.A. 2001. Using demographic methods to construct Bayesian priors for the intrinsic rate of increase in the Schaefer model and implications for stock rebuilding. Can. J. Fish. Aquat. Sci. 58: 1871-1890.

Pedersen, M. W. and Berg, C. W. 2017. A stochastic surplus production model in continuous time. Fish and Fisheries. 18:226-243.

Pella, J. J. and Tomlinson, P. K. 1969. A generalized stock production model. Inter-Am. Trop. Tuna Comm., Bull. 13:419-496.

Prager, M. H. 1994. A suite of extensions to a nonequilibrium surplus-production model. Fishery Bulletin 92:374-389.

Stanley, R.D., M. McAllister, P. Starr and N. Olsen. 2009. Stock assessment for bocaccio (Sebastes paucispinis) in British Columbia waters. DFO Can. Sci. Advis. Sec. Res. Doc. 2009/055. xiv + 200 p.

See Also

SP_production plot.Assessment summary.Assessment retrospective profile make_MP

Examples

```
data(swordfish)

#### Observation-error surplus production model
res <- SP(Data = swordfish)

# Provide starting values, assume B/K = 0.875 in first year of model
# and symmetrical production curve (n = 2)
start <- list(dep = 0.875, n = 2)
res <- SP(Data = swordfish, start = start)


plot(res)
profile(res, FMSY = seq(0.1, 0.4, 0.01))
retrospective(res)


#### State-space version
res_SS <- SP_SS(Data = swordfish, start = list(dep = 0.875, sigma = 0.1, tau = 0.1))


plot(res_SS)


#### Fox model
res_Fox <- SP(Data = swordfish, start = list(n = 1), fix_n = TRUE)
res_Fox2 <- SP_Fox(Data = swordfish)

#### SP with r prior calculated internally (100 stochastic samples to get prior SD)
res_prior <- SP(Data = SimulatedData, Euler_Lotka = 100)

#### Pass an r prior to the model with mean = 0.35, lognormal sd = 0.10
res_prior2 <- SP(Data = SimulatedData, prior = list(r = c(0.35, 0.10)))

#### Pass MSY prior to the model with mean = 1500, lognormal sd = 0.05
res_prior3 <- SP(Data = SimulatedData, prior = list(MSY = c(1500, 0.05)))
```

SP_production *Find the production parameter based on depletion that produces MSY*

Description

For surplus production models, this function returns the production exponent n corresponding to BMSY/K (Fletcher 1978).

Usage

```
SP_production(depletion, figure = TRUE)
```

Arguments

depletion	The hypothesized depletion that produces MSY.
figure	Local, plots figure of production function as a function of depletion (B/K)

Value

The production function exponent n (numeric).

Note

May be useful for parameterizing n in SP and SP_SS.

Author(s)

Q. Huynh

References

Fletcher, R. I. 1978. On the restructuring of the Pella-Tomlinson system. Fishery Bulletin 76:515:521.

See Also

SP SP_SS

Examples

```
SP_production(0.5)
SP_production(0.5)
```

SSS *Simple Stock Synthesis*

Description

A simple age-structured model (SCA_Pope) fitted to a time series of catch going back to unfished conditions. Terminal depletion (ratio of current total biomass to unfished biomass) is by default fixed to 0.4. Selectivity is fixed to the maturity ogive, although it can be overridden with the start argument. The sole parameter estimated is R0 (unfished recruitment), with no process error.

Usage

```
SSS(
  x = 1,
  Data,
  dep = 0.4,
  SR = c("BH", "Ricker"),
  rescale = "mean1",
  start = NULL,
  prior = list(),
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  ...
)
```

Arguments

x	A position in the Data object (by default, equal to one for assessments).
Data	An object of class Data
dep	Depletion value to use in the model. Can be an expression that will be evaluated inside the function.
SR	Stock-recruit function (either "BH" for Beverton-Holt or "Ricker").
rescale	A multiplicative factor that rescales the catch in the assessment model, which can improve convergence. By default, "mean1" scales the catch so that time series mean is 1, otherwise a numeric. Output is re-converted back to original units.
start	Optional named list of starting values. Entries can be expressions that are evaluated in the function: • R0 Unfished recruitment • vul_par A length-two vector for the age of 95% and 50% fleet selectivity. Fixed to maturity otherwise.
prior	A named list for the parameters of any priors to be added to the model. See details in SCA_Pope.

silent	Logical, passed to TMB::MakeADFun(), whether TMB will print trace information during optimization. Used for diagnostics for model convergence.
opt_hess	Logical, whether the hessian function will be passed to stats::nlminb() during optimization (this generally reduces the number of iterations to convergence, but is memory and time intensive and does not guarantee an increase in convergence rate).
n_restart	The number of restarts (calls to stats::nlminb()) in the optimization procedure, so long as the model hasn't converged. The optimization continues from the parameters from the previous (re)start.
control	A named list of arguments for optimization to be passed to stats::nlminb().
...	Other arguments to be passed (not currently used).

Details

In SAMtool, SSS is an implementation of SCA_Pope with fixed final depletion (in terms of total biomass, not spawning biomass) assumption.

Value

An object of class Assessment.

Author(s)

Q. Huynh

References

Cope, J.M. 2013. Implementing a statistical catch-at-age model (Stock Synthesis) as a tool for deriving overfishing limits in data-limited situations. Fisheries Research 142:3-14.

Examples

```
res <- SSS(Data = Red_snapper)

SSS_MP <- make_MP(SSS, HCR40_10, dep = 0.3) # Always assume depletion = 0.3
```

---

summary.Assessment *Summary of Assessment object*

Description

Returns a summary of parameter estimates and output from an Assessment object.

Usage

```
## S4 method for signature 'Assessment'
summary(object)
```

Arguments

object	An object of class Assessment

Value

A list of parameters.

Examples

```
output <- DD_TMB(Data = MSEtool::SimulatedData)
summary(output)
```

---

swordfish *North Atlantic Swordfish dataset*

Description

An S4 object containing catch and index time series for North Atlantic swordfish.

Usage

```
swordfish
```

Format

An object of class MSEtool::Data.

Source

ASPIC Software at https://www.mhprager.com/aspic.html

Examples

```
data(swordfish)
```

TAC_MSY *Calculate MSY-based TAC from Assessment object*

Description

A function to calculate the total allowable catch (TAC). Based on the MSY (maximum sustainable yield) principle, the TAC is the product of either UMSY or FMSY and the available biomass, i.e. vulnerable biomass, in terminal year.

Usage

```
TAC_MSY(Assessment, reps, MSY_frac = 1)
```

Arguments

Assessment	An Assessment object with estimates of UMSY or FMSY and terminal year vulnerable biomass.
reps	The number of stochastic draws of UMSY or FMSY.
MSY_frac	The fraction of FMSY or UMSY for calculating the TAC (e.g. MSY_frac = 0.75 fishes at 75% of FMSY).

Value

A vector of length reps of stochastic samples of TAC recommendation. Returns NA's if missing either UMSY/FMSY or vulnerable biomass.

Note

calculate_TAC is deprecated as of version 1.2 in favor of TAC_MSY because the latter has a more informative name.

See Also

HCR_MSY HCR40_10 HCR60_20

userguide *Get the SAMtool vignettes*

Description

A convenient function to open a web browser with the openMSE documentation vignettes

Usage

```
userguide()
```

Value

Displays a browser webpage to the openMSE website.

Examples

```
userguide()
```

VPA *Virtual population analysis (VPA)*

Description

A VPA model that back-calculates abundance-at-age assuming that the catch-at-age is known without error and tuned to an index. The population dynamics equations are primarily drawn from VPA-2BOX (Porch 2018). MSY reference points and per-recruit quantities are then calculated from the VPA output.

Usage

```
VPA(
  x = 1,
  Data,
  AddInd = "B",
  expanded = FALSE,
  SR = c("BH", "Ricker"),
  vulnerability = c("logistic", "dome", "free"),
  start = list(),
  fix_h = TRUE,
  fix_Fratio = TRUE,
  fix_Fterm = FALSE,
  LWT = NULL,
  shrinkage = list(),
  n_itF = 5L,
  min_age = "auto",
  max_age = "auto",
  refpt = list(),
  silent = TRUE,
  opt_hess = FALSE,
  n_restart = ifelse(opt_hess, 0, 1),
  control = list(iter.max = 2e+05, eval.max = 4e+05),
  ...
)
```

Arguments

x	A position in the Data object (by default, equal to one for assessments).
Data	An object of class Data
AddInd	A vector of integers or character strings indicating the indices to be used in the model. Integers assign the index to the corresponding index in Data@AddInd, "B" (or 0) represents total biomass in Data@Ind, "VB" represents vulnerable biomass in Data@VInd, and "SSB" represents spawning stock biomass in Data@SpInd.
expanded	Whether the catch at age in Data has been expanded. If FALSE, then the catch in weight should be provided in Data@Cat so that the function can calculate annual expansion factors.
SR	Stock-recruit function (either "BH" for Beverton-Holt or "Ricker") for calculating MSY reference points.
vulnerability	Whether the terminal year vulnerability is "logistic" or "dome" (double-normal). If "free", independent F's are calculated in the terminal year (subject to the assumed ratio of F of the plus-group to the previous age class). See details for parameterization.
start	Optional list of starting values. Entries can be expressions that are evaluated in the function. See details.
fix_h	Logical, whether to fix steepness to value in Data@steep. This only affects calculation of MSY and unfished reference points.
fix_Fratio	Logical, whether the ratio of F of the plus-group to the previous age class is fixed in the model.
fix_Fterm	Logical, whether to fix the value of the terminal F.
LWT	A vector of likelihood weights for each survey.
shrinkage	A named list of up to length 2 to constrain parameters: • vul - a length two vector that constrains the vulnerability-at-age in the most recent years. The first number is the number of years in which vulnerability will be constrained (as a random walk in log space), the second number is the standard deviation of the random walk. The default • R - a length two vector that constrains the recruitment estimates in the most recent years. The first number is the number of years in which recruitment will be constrained (as a random walk in log space), the second number is the standard deviation of the random walk.
n_itF	The number of iterations for solving F in the model (via Newton's method).
min_age	An integer to specify the smallest age class in the VPA. By default, the youngest age with non-zero CAA in the terminal year is used.
max_age	An integer to specify the oldest age class in the VPA. By default, the oldest age with non-zero CAA for all years is used.
refpt	A named list of how many years to average parameters for calculating reference points, yield per recruit, and spawning potential ratio: • vul An integer for the number of most recent years to average the vulnerability schedule (default is 3).

- `R` A length two for the quantile used to calculate recruitment in the year following the terminal year and the number of years from which that quantile is used, i.e., `c(0.5, 5)` is the default that calculates median recruitment from the most recent 5 years of the model.

`silent` Logical, passed to `TMB::MakeADFun()`, whether TMB will print trace information during optimization. Used for diagnostics for model convergence.

`opt_hess` Logical, whether the hessian function will be passed to `stats::nlminb()` during optimization (this generally reduces the number of iterations to convergence, but is memory and time intensive and does not guarantee an increase in convergence rate). Ignored if `integrate = TRUE`.

`n_restart` The number of restarts (calls to `stats::nlminb()`) in the optimization procedure, so long as the model hasn't converged. The optimization continues from the parameters from the previous (re)start.

`control` A named list of arguments for optimization to be passed to `stats::nlminb()`.

`...` Other arguments to be passed.

Details

The VPA is initialized by estimating the terminal F-at-age. Parameter `Fterm` is the apical terminal F if a functional form for vulnerability is used in the terminal year, i.e., when `vulnerability = "logistic"` or `"free"`. If the terminal F-at-age are otherwise independent parameters, `Fterm` is the F for the reference age which is half the maximum age. Once terminal-year abundance is estimated, the abundance in historical years can be back-calculated. The oldest age group is a plus-group, and requires an assumption regarding the ratio of F's between the plus-group and the next youngest age class. The F-ratio can be fixed (default) or estimated.

For `start` (optional), a named list of starting values of estimates can be provided for:

- `Fterm` The terminal year fishing mortality. This is the apical F when `vulnerability = "logistic"` or `"free"`.
- `Fratio` The ratio of F in the plus-group to the next youngest age. If not provided, a value of 1 is used.
- `vul_par` Vulnerability parameters in the terminal year. This will be of length 2 vector for `"logistic"` or length 4 for `"dome"`, see SCA for further documentation on parameterization. For option `"free"`, this will be a vector of length A-2 where A is the number of age classes in the model. To estimate parameters, vulnerability is initially set to one at half the max age (and subsequently re-calculated relative to the maximum F experienced in that year). Vulnerability in the plus-group is also constrained by the Fratio.

MSY and depletion reference points are calculated by fitting the stock recruit relationship to the recruitment and SSB estimates. Per-recruit quantities are also calculated, which may be used in harvest control rules.

Value

An object of class Assessment. The F vector is the apical fishing mortality experienced by any age class in a given year.

Additional considerations

The VPA tends to be finicky to implement straight out of the box. For example, zeros in plusgroup age in the catch-at-age model will crash the model, as well as if the catch-at-age values are close to zero. The model sets F-at-age to 1e-4 if any catch-at-age value < 1e-4.

It is recommended to do some preliminary fits with the VPA before running simulations en masse. See example below.

Shrinkage, penalty functions that stabilize model estimates of recruitment and selectivity year-over-year near the end of the time series, alters the behavior of the model. This is something to tinker with in your initial model fits, and worth evaluating in closed-loop simulation.

Online Documentation

Model description and equations are available on the openMSE website.

References

Porch, C.E. 2018. VPA-2BOX 4.01 User Guide. NOAA Tech. Memo. NMFS-SEFSC-726. 67 pp.

Examples

```
OM <- MSEtool::testOM

# Simulate logistic normal age comps with CV = 0.1
# (set CAA_ESS < 1, which is interpreted as a CV)
OM@CAA_ESS <- c(0.1, 0.1)
Hist <- MSEtool::Simulate(OM, silent = TRUE)

# VPA max age is 15 (Hist@Data@MaxAge)
m <- VPA(x = 2, Data = Hist@Data, vulnerability = "dome")

# Use age-9 as the VPA max age instead
m9 <- VPA(x = 2, Data = Hist@Data, vulnerability = "dome", max_age = 9)

compare_models(m, m9)
```

Index

∗ **datasets**
pcod, 44
swordfish, 107
∗ **evaluation**
SAMtool-package, 3
∗ **fisheries**
SAMtool-package, 3
∗ **management**
SAMtool-package, 3
∗ **strategy**
SAMtool-package, 3

Assessment, *9*, *21*, *25*, *29*, *32–35*, *38*, *45–47*, *60*, *62*, *65*, *67*, *76*, *83*, *90*, *93*, *96*, *98*, *101*, *106*, *107*, *111*
Assessment *(*Assessment-class*)*, 4
Assessment-class, 4

calculate_TAC *(*TAC_MSY*)*, 108
cDD, 7
cDD_SS *(*cDD*)*, 7
check_RCMdata, 11
compare_models, 20
compare_RCM, *20*
compare_RCM *(*plot.RCModel*)*, 47
contour, *47*

Data-rich-MP *(*Model-based-MP*)*, 42
DD_SS, *44*
DD_SS *(*DD_TMB*)*, 21
DD_TMB, *10*, 21
DDSS_4010 *(*Model-based-MP*)*, 42
DDSS_75MSY *(*Model-based-MP*)*, 42
DDSS_MSY *(*Model-based-MP*)*, 42
diagnostic, 26, *42*, *77*
diagnostic_AM *(*diagnostic*)*, 26

getinds, 27

HCR40_10, *108*
HCR40_10 *(*HCR_ramp*)*, 34
HCR60_20, *108*
HCR60_20 *(*HCR_ramp*)*, 34
HCR80_40MSY *(*HCR_ramp*)*, 34
HCR_escapement, 29
HCR_FB, 30
HCR_fixedF, 31
HCR_MSY, 32, *36*, *42*, *108*
HCR_ramp, *28*, *30*, *32*, *33*, 34, *37*, *42*
HCR_segment, *34*, *36*, 37
HCRlin, 28, *36*

loess, *41*

mahplot, 39
make_interim_MP, 40
make_MP, *6*, *10*, *26*, *30*, *32–34*, *36*, *43*, *84*, *90*, *94*, *103*
make_MP *(*make_interim_MP*)*, 40
make_MP(), *77*
make_projection_MP *(*make_interim_MP*)*, 40
MakeADFun, *14*
Model-based-MP, 42
MP *(*Model-based-MP*)*, 42
MSEextra, *98*
MSEtool::Data, *8*, *13*, *16*, *19*, *23*, *107*
MSEtool::MOM, *68*
MSEtool::MSE, *77*
MSEtool::OM, *13*, *44*, *71*
MSEtool::Rec, *30*, *32*, *33*, *36*, *39*, *43*
MSEtool::runMSE(), *26*, *40*

newton, *101*

pcod, *20*, 44
Perfect *(*Shortcut*)*, 94
plot,Assessment,missing-method *(*plot.Assessment*)*, 45
plot,Assessment,retro-method *(*plot.Assessment*)*, 45
plot,prof,missing-method *(*plot.prof*)*, 46

plot,RCModel,missing-method (plot.RCModel), 47
plot,retro,missing-method (plot.retro), 49
plot.Assessment, *6*, *10*, *26*, 45, *75*, *84*, *90*, *103*
plot.prof, 46, *64*
plot.RCModel, *20*, 47, *73*
plot.retro, 49, *75*
plot_betavar, 50
plot_betavar(), *54*, *57*
plot_composition, 51
plot_crosscorr, 53
plot_lognormalvar, 54
plot_lognormalvar(), *50*, *57*
plot_residuals, 55
plot_residuals(), *59*
plot_SR, 56
plot_steepness, 57
plot_steepness(), *50*, *54*
plot_timeseries, 58
plot_timeseries(), *56*
posterior, *20*, 59
posterior,Assessment-method (posterior), 59
posterior,RCModel-method (posterior), 59
posterior.Assessment (posterior), 59
posterior.RCModel (posterior), 59
PRBcalc, *39*, 61, *63*
predict.varest, *94*
prelim_AM, 62
Probs, *61*, 63
prof, *47*, *65*
prof (prof-class), 64
prof-class, 64
profile, *6*, *10*, *26*, *46*, *47*, *64*, *84*, *90*, *103*
profile (profile,Assessment-method), 64
profile,Assessment-method, 64
profile,RCModel-method (profile,Assessment-method), 64
project, *67*
project (project-class), 66
project-class, 66
projection, *41*, *66*, 66

RCM, *44*, *45*, *48*, *49*, *68–71*, *73*
RCM (check_RCMdata), 11
RCM(), *3*
RCM,list,RCMdata-method (check_RCMdata), 11
RCM,OM,Data-method (check_RCMdata), 11
RCM,OM,list-method (check_RCMdata), 11
RCM,OM,RCMdata-method (check_RCMdata), 11
RCM2MOM, *20*, 68
RCM_assess, 73
RCMdata, *13*, *14*, *16*, *19*, *44*, *73*
RCMdata (RCMdata-class), 69
RCMdata-class, 69
RCModel, *15*, *20*, *48*, *49*, *60*, *68*, *76*, *98*
RCModel (RCModel-class), 71
RCModel-class, 71
RCMstan (posterior), 59
render, *46*, *48*, *49*
retro, *46*, *49*
retro (retro-class), 75
retro-class, 75
retrospective, *6*, *10*, *26*, *46*, *75*, 75, *84*, *90*, *103*
retrospective,Assessment-method (retrospective), 75
retrospective,RCModel-method (retrospective), 75
retrospective_AM, *27*, *42*, 76
runMSE, *95*, *100*

SAMtool (SAMtool-package), 3
SAMtool-package, 3
SCA, *43*, *74*, 78, *84*, *89*, *90*, *93–95*, *111*
SCA2 (SCA), 78
SCA_4010 (Model-based-MP), 42
SCA_75MSY (Model-based-MP), 42
SCA_CAL, 84
SCA_DDM, 87, *94*
SCA_MSY (Model-based-MP), 42
SCA_Pope, *105*, *106*
SCA_Pope (SCA), 78
SCA_RWM, *90*, 90
Shortcut, 94
Shortcut2 (Shortcut), 94
sim, *98*
sim (sim-class), 97
sim-class, 97
simulate, *97*, 97
simulate,Assessment-method (simulate), 97
simulate,RCModel-method (simulate), 97

simulate.Assessment *(simulate)*, 97
simulate.RCModel *(simulate)*, 97
SP, *44*, 98, *104*
SP_4010 *(Model-based-MP)*, 42
SP_75MSY *(Model-based-MP)*, 42
SP_Fox *(SP)*, 98
SP_MSY *(Model-based-MP)*, 42
SP_production, *103*, 103
SP_SS, *104*
SP_SS *(SP)*, 98
SSS, *44*, 105
SSS_4010 *(Model-based-MP)*, 42
SSS_75MSY *(Model-based-MP)*, 42
SSS_MSY *(Model-based-MP)*, 42
stats::nlminb(), *6*, *9*, *14*, *23*, *27*, *72*, *81*, *86*, *89*, *92, 93*, *100, 101*, *106*, *111*
summary,Assessment-method *(summary.Assessment)*, 106
summary,retro-method *(plot.retro)*, 49
summary.Assessment, *6*, *10*, *26*, *84*, *90*, *103*, 106
summary.retro, *75*
summary.retro *(plot.retro)*, 49
swordfish, 107

TAC_MSY, 108
TMB::MakeADFun(), *6*, *8, 9*, *23, 24*, *72*, *81*, *86*, *89*, *92*, *100, 101*, *106*, *111*
TMB::newton(), *9*, *24*, *81*, *86*, *89*, *93*
TMB::sdreport(), *6*, *73*

userguide, 108

VAR, *95, 96*
VPA, 109