

Package: stockassessment (via r-universe)

January 24, 2025

Title State-Space Assessment Model

Version 0.12.0

Date 2025-01-20

Author Anders Nielsen [aut, cre], Casper Berg [aut], Christoffer Moesgaard Albertsen [aut], Kasper Kristensen [aut], Mollie Brooks [aut], Vanessa Trijoulet [aut], Olav Nikolai Breivik [aut]

Maintainer Anders Nielsen <an@aqu.dtu.dk>

Description Fitting SAM...

License GPL-2

Imports TMB, ellipse, methods, stats, utils, grDevices, graphics, Matrix, plot.matrix

LinkingTo TMB (>= 1.9.1), RcppEigen

Encoding UTF-8

URL <https://github.com/fishfollower/SAM>

LazyData TRUE

BugReports <https://github.com/fishfollower/SAM/issues>

RemoteType github

RemoteHost <https://api.github.com>

RemoteRepo SAM

RemoteUsername fishfollower

RemoteRef HEAD

RemoteSha 7e947decf551ffb850a7164794176a61c4975569

RemoteSubdir stockassessment

RoxygenNote 7.2.3

Repository <https://flr.r-universe.dev>

RemoteUrl <https://github.com/fishfollower/SAM>

Contents

addforecast	4
addRecruitmentCurve	5
b0plot	7
b0table	8
bc	8
c.sam	9
catchbyfleetplot	9
catchbyfleetable	10
catchplot	10
catchtable	11
caytable	12
clean.void.catches	12
coef.sam	13
componentplot	13
corplot	14
corplotcommon	15
dataplot	15
defcon	16
defpar	16
deterministicReferencepoints	17
deterministicReferencepoints.sam	18
empirobscorrplot	19
equilibriumbiomassplot	20
equilibriumbiomassstable	20
faytable	21
fbarplot	22
fbartable	23
fitfromweb	24
fitplot	24
forecast	25
forecastMSY	27
fselectivityplot	28
generationlengthplot	29
generationlengthtable	29
getAllDerivedValues	30
getFleet	30
getLowerBounds	31
getResidualFleets	31
getUpperBounds	32
grad	32
hcr	33
ibc	34
icesAdviceRule	35
iibc	36
is.whole.positive.number	36
jacobian	37

jit	38
leaveout	39
lifeexpectancyplot	39
lifeexpectancytable	40
loadConf	41
logLik.sam	41
modelDescription	42
modelforecast	42
modeltable	46
modelVersionInfo	47
mohn	47
nobs.sam	48
nscodConf	48
nscodData	49
nscodParameters	50
ntable	51
obscorrplot	51
obscov	52
parplot	52
partable	53
plot.hcr	53
plot.sam	54
plot.samforecast	54
plot.samres	55
plot.samset	56
plot.samypr	56
plotby	57
plotit	58
predstdplot	61
print.hcr	62
print.sam	62
print.samcoef	63
print.samforecast	63
print.samres	64
print.samset	64
print.samypr	65
print.sam_referencepoints	65
procres	66
qtable	66
qtable.sam	67
qtableplot	67
read.data.files	68
read.ices	68
read.surveys	69
read.table.nowarn	69
recplot	70
rectable	70
reduce	71

referencepoints	72
refit	74
residplot	74
residuals.sam	75
retro	76
rmaxplot	76
rmaxtable	77
rmvnorm	78
runwithout	78
sam.fit	79
saveConf	83
sdplot	84
setS	84
setSeq	85
setup.sam.data	85
simstudy	86
simulate.sam	87
sprplot	88
sprtable	88
srplot	89
ssbplot	90
ssbtable	91
stochasticReferencepoints	91
stockassessment-deprecated	93
summary.sam	94
tableit	94
tsbplot	95
tsbtable	95
write.data.files	96
write.ices	96
write.surveys	97
yearslostplot	97
yearslosttable	98
ypr	99
yprplot	100
yprtable	100

Index**102**

addforecast

*SAM add forecasts***Description**

SAM add forecasts

Usage

```

addforecast(
  fit,
  what,
  dotcol = "black",
  dotpch = 19,
  dotcex = 1.5,
  intervalcol = gray(0.5, alpha = 0.5),
  ...
)

## S3 method for class 'samforecast'
addforecast(
  fit,
  what,
  dotcol = "black",
  dotpch = 19,
  dotcex = 1.5,
  intervalcol = gray(0.5, alpha = 0.5),
  ...
)

```

Arguments

fit	the object returned from sam.fit
what	what to plot
dotcol	color for dot
dotpch	pch for dot
dotcex	cex for dot
intervalcol	color for interval
...	extra arguments not currently used

Details

internal plotting fun

addRecruitmentCurve *Add stock-recruitment curve to srplot*

Description

Add stock-recruitment curve to srplot

Usage

```

addRecruitmentCurve(
  fit,
  CI = TRUE,
  col = rgb(0.6, 0, 0),
  cicol = rgb(0.6, 0, 0, 0.3),
  plot = TRUE,
  PI = FALSE,
  picol = rgb(0.6, 0, 0),
  pilty = 2,
  ...
)

## S3 method for class 'sam'
addRecruitmentCurve(
  fit,
  CI = TRUE,
  col = rgb(0.6, 0, 0),
  cicol = rgb(0.6, 0, 0, 0.3),
  plot = TRUE,
  PI = FALSE,
  picol = rgb(0.6, 0, 0),
  pilty = 2,
  year = NA_real_,
  lastR = NA_real_,
  ...
)

```

Arguments

<code>fit</code>	Object to show SR-curve for
<code>CI</code>	Add confidence intervals?
<code>col</code>	Color of fitted line
<code>cicol</code>	Color of confidence intervals
<code>plot</code>	Add the curve to a plot?
<code>PI</code>	Add prediction intervals?
<code>picol</code>	Color of prediction interval line
<code>pilty</code>	Line type of prediction interval line
<code>...</code>	not used
<code>year</code>	Show recruitment calculated conditional on this year (for recruitment functions that depend on year)
<code>lastR</code>	Show recruitment calculated conditional on this previous recruitment (for recruitment functions that depend on recruitment the previous year)

See Also

srplot

b0plot*SAM equilibrium biomass in the absence of fishing plot*

Description

SAM equilibrium biomass in the absence of fishing plot

Usage

```
b0plot(fit, ...)

## Default S3 method:
b0plot(fit, ...)

## S3 method for class 'samforecast'
b0plot(fit, ...)

## S3 method for class 'hcr'
b0plot(fit, ...)
```

Arguments

<code>fit</code>	the object returned from <code>sam.fit</code>
<code>...</code>	extra arguments transferred to plot including the following: add logical, plotting is to be added on existing plot ci logical, confidence intervals should be plotted cicol color to plot the confidence polygon

Details

Plot of deterministic equilibrium biomass in the absence of fishing assuming biological parameters and selectivity for that year remains unchanged in the future.

b0table	<i>B0 biomass table</i>
---------	-------------------------

Description

B0 biomass table

Usage

```
b0table(fit, ...)
```

```
## Default S3 method:
b0table(fit, ...)
```

Arguments

fit	...
...	extra arguments not currently used

Details

...

bc	<i>Spline basis for use with formula interface</i>
----	--

Description

Spline basis for use with formula interface

Usage

```
bc(x, df = 3L, knots = NULL, Boundary.knots = range(x), intercept = FALSE)
```

Arguments

x	Points to evaluate the basis in
df	Degrees of freedom
knots	Internal knots. If NULL, they are selected from quantiles of x.
Boundary.knots	Boundary knots. Defaults to range of x
intercept	Include an intercept in basis?

Value

A spline basis

c.sam	<i>Collect sam objects</i>
-------	----------------------------

Description

Collect sam objects

Usage

```
## S3 method for class 'sam'
c(...)
```

Arguments

... one or more sam fits (as returned from the [sam.fit](#) function) to be combined

Details

...

catchbyfleetplot	<i>SAM catchbyfleet plot</i>
------------------	------------------------------

Description

SAM catchbyfleet plot

Usage

```
catchbyfleetplot(fit, obs.show = FALSE, ...)
```

Arguments

fit	the object returned from sam.fit
obs.show	if observations are to be shown also
...	extra arguments transferred to plot

Details

Plot of estimated (and optionally observed) total catch in weight

catchbyfleetable	<i>CatchByFleet table</i>
------------------	---------------------------

Description

CatchByFleet table

Usage

```
catchbyfleetable(fit, obs.show = FALSE)
```

Arguments

fit	object returned from sam.fit
obs.show	logical add a column with catch sum of product rowsums(C*W)

Details

...

catchplot	<i>SAM catch plot</i>
-----------	-----------------------

Description

SAM catch plot

Usage

```
catchplot(fit, obs.show = TRUE, drop = NULL, ...)
```

```
## S3 method for class 'sam'
catchplot(fit, obs.show = TRUE, drop = NULL, ...)
```

```
## S3 method for class 'samset'
catchplot(fit, obs.show = TRUE, drop = NULL, ...)
```

```
## S3 method for class 'samforecast'
catchplot(fit, obs.show = TRUE, drop = NULL, ...)
```

```
## S3 method for class 'hcr'
catchplot(fit, obs.show = TRUE, drop = NULL, ...)
```

Arguments

<code>fit</code>	the object returned from <code>sam.fit</code>
<code>obs.show</code>	if observations are to be shown also
<code>drop</code>	number of years to be left unplotted at the end. Default (NULL) is to not show years at the end with no catch information
<code>...</code>	extra arguments transferred to plot including the following: <code>add</code> logical, plotting is to be added on existing plot <code>ci</code> logical, confidence intervals should be plotted <code>cicol</code> color to plot the confidence polygon

Details

Plot of estimated (and optionally observed) total catch in weight

<code>catchtable</code>	<i>Catch table</i>
-------------------------	--------------------

Description

Catch table

Usage

```
catchtable(fit, obs.show = FALSE, ...)

## S3 method for class 'sam'
catchtable(fit, obs.show = FALSE, ...)
```

Arguments

<code>fit</code>	object returned from <code>sam.fit</code>
<code>obs.show</code>	logical add a column with catch sum of product <code>rowsums(C*W)</code>
<code>...</code>	extra arguments not currently used

Details

...

caytable	<i>Catch-at-age in numbers table</i>
----------	--------------------------------------

Description

Catch-at-age in numbers table

Usage

```
caytable(fit, fleet = which(fit$data$fleetTypes == 0))
```

Arguments

fit	a fitted object of class 'sam' as returned from sam.fit
fleet	the fleet number(s) to return catch summed for (default is to return the sum of all residual fleets).

Details

...

clean.void.catches	<i>remove void catches</i>
--------------------	----------------------------

Description

remove void catches

Usage

```
clean.void.catches(dat, conf)
```

Arguments

dat	data for the sam model as returned from the setup.sam.data function
conf	model configuration which can be set up using the defcon function and then modified

Value

an updated dataset without the catches where F is fixed to zero

coef.sam	<i>Extract fixed coefficients of sam object</i>
----------	---

Description

Extract fixed coefficients of sam object

Usage

```
## S3 method for class 'sam'
coef(object, ...)
```

Arguments

object	sam fitted object as returned from the sam.fit function
...	extra arguments

Details

fixed coefficients of sam object

componentplot	<i>Area plot of spawning components</i>
---------------	---

Description

Area plot of spawning components

Usage

```
componentplot(fit, ...)

## S3 method for class 'sam'
componentplot(
  fit,
  onlyComponentYears = FALSE,
  ylab = "Composition",
  colSet = c("#332288", "#88CCEE", "#44AA99", "#117733", "#999933", "#DDCC77", "#661100",
    "#CC6677", "#882255", "#AA4499"),
  legend.pos = "bottom",
  bg = "white",
  ncol = length(cf),
  ...
)
```

Arguments

fit	sam fit
...	passed to legend
onlyComponentYears	If true, x axis is limited to the range with spawning component data. Otherwise, the model years are used.
ylab	Label for y axis
colSet	Colors
legend.pos	Legend position. See ?legend
bg	Background of legend. See ?legend
ncol	Number of columns in legend. See ?legend

Value

Nothing

Author(s)

Christoffer Moesgaard Albertsen

corplot	<i>Plots between-age correlations by fleet, either estimated or empirical using residuals.</i>
---------	--

Description

Plots between-age correlations by fleet, either estimated or empirical using residuals.

Usage

```
corplot(x, ...)

## S3 method for class 'sam'
corplot(x, ...)

## S3 method for class 'samres'
corplot(x, ...)
```

Arguments

x	Either a sam fit as returned by sam.fit OR the object returned from residuals.sam
...	extra arguments to plot

corplotcommon	<i>Common function for plotting correlation matrices.</i>
---------------	---

Description

Common function for plotting correlation matrices.

Usage

```
corplotcommon(x, fn, ...)
```

Arguments

x	a list of correlation matrices
fn	a vector of fleet names
...	extra arguments to plotcorr

dataplot	<i>SAM Data plot</i>
----------	----------------------

Description

SAM Data plot

Usage

```
dataplot(fit, col = NULL, fleet_type = NULL, fleet_names = NULL)
```

```
## S3 method for class 'sam'
```

```
dataplot(fit, col = NULL, fleet_type = NULL, fleet_names = NULL)
```

Arguments

fit	the object returned from sam.fit
col	color to use for each fleet, default is two sequential colors
fleet_type	character vector giving the type of data per fleet. The default uses fit\$data\$fleetTypes as follows: fit\$data\$fleetTypes==0 "Catch at age" fit\$data\$fleetTypes==1 "Catch at age with effort" fit\$data\$fleetTypes==2 or 6 "Index at age" fit\$data\$fleetTypes==3 "Biomass or catch index" fit\$data\$fleetTypes==5 "Tagging data" fit\$data\$fleetTypes==7 "Sum of fleets"
fleet_names	character vector giving fleet names. The default is given by attr(fit\$data, "fleetNames")

Details

Plot data available for the stock

defcon	<i>Setup basic minimal configuration for sam assessment</i>
--------	---

Description

Setup basic minimal configuration for sam assessment

Usage

```
defcon(dat, level = 1)
```

Arguments

dat	sam data object
level	1 or 2 (1 most basic configuration, 2 configuration with AR correlation structure on surveys)

Details

The configuration returned by defcon is intended as a help to set up a syntactically correct configuration for the sam model. The dimensions are set from the data (years, age-classes, and fleet types available). The configuration is intended to be fairly simplistic in the hope that the model configured will at least converge (not guaranteed). Most importantly: No model validation has been performed, so it should not be assumed that the returned model configuration will result in a sensible assessment of the stock. The actual model configuration is the responsibility of the user.

Value

a list containing the elements needed to configure a sam model (e.g. minAge, maxAge, maxAge-PlusGroup, keyLogFsta, ...).

defpar	<i>Setup initial values for all model parameters and random effects.</i>
--------	--

Description

Setup initial values for all model parameters and random effects.

Usage

```
defpar(dat, conf, spinoutyear = 10)
```


Arguments

dat	sam data object as returned from the function <code>setup.sam.data</code>
conf	sam configuration list, which could be read from a configuration file via the <code>loadConf</code> function. A default/dummy configuration can be generated via the <code>defcon</code> function.
spinoutyear	Technical setting only used for biological parameter process models to insure equilibrium distribution in final edge year

Details

The model parameters and random effects are not initialized in any clever way - most are simply set to zero. If convergence problems occur different initial values can be tested, but it is more likely a problem with the model configuration.

Value

a list containing initial values for all model parameters and random effects in the model.

deterministicReferencepoints

Function to calculate reference points for the embedded deterministic model of a SAM fit

Description

The function estimates reference points based on deterministic per-recruit calculations with no process variance. The following reference points are implemented:

F=x F fixed to x, e.g., "F=0.3"

StatusQuo F in the last year of the assessment

StatusQuo-y F in the y years before the last in the assessment, e.g., "StatusQuo-1"

MSY F that maximizes yield

0.xMSY Fs that gives 0.x*100% of MSY, e.g., "0.95MSY"

Max F that maximizes yield per recruit

0.xdYPR F such that the derivative of yield per recruit is 0.x times the derivative at F=0, e.g., "0.1dYPR"

0.xSPR F such that spawners per recruit is 0.x times spawners per recruit at F=0, e.g., "0.35SPR"

0.xB0 F such that biomass is 0.x times the biomass at F=0, e.g., "0.2B0"

Usage

`deterministicReferencepoints(fit, referencepoints, ...)`

Arguments

fit A fitted SAM model
referencepoints list of reference points to calculate (See details)
 ... other arguments not used

Value

List of estimated reference points

Examples

```

## Not run:
deterministicReferencepoints(fit, c("MSY", "0.95MSY", "Max", "0.35SPR", "0.1dYPR", "StatusQuo-3"))

## End(Not run)

```

deterministicReferencepoints.sam

Function to calculate reference points for the embedded deterministic model of a SAM fit

Description

Function to calculate reference points for the embedded deterministic model of a SAM fit

Usage

```

## S3 method for class 'sam'
deterministicReferencepoints(
  fit,
  referencepoints,
  catchType = "catch",
  nYears = 300,
  Fsequence = seq(0, 2, len = 50),
  aveYears = max(fit$data$years) + (-9:0),
  selYears = max(fit$data$years),
  biasCorrect = FALSE,
  newton.control = list(),
  run = TRUE,
  ...
)

```

Arguments

<code>fit</code>	A fitted SAM model
<code>referencepoints</code>	list of reference points to calculate (See details)
<code>catchType</code>	landing, catch, or discard
<code>nYears</code>	Number of years in per-recruit calculations
<code>Fsequence</code>	Sequence of F values for plotting and starting values
<code>aveYears</code>	Years to average over for biological input
<code>selYears</code>	Years to average over for selectivity
<code>biasCorrect</code>	Should bias correction be used in sdreport ?
<code>newton.control</code>	Control arguments passed to the newton optimizer (See newton)
<code>run</code>	Run estimation? If false, a list of arguments to <code>MakeADFun</code> is returned.
<code>...</code>	other arguments not used

Value

List of estimated reference points

<code>empirobscorrplot</code>	<i>Plots the residual between-age correlation matrices by fleet.</i>
-------------------------------	--

Description

Plots the residual between-age correlation matrices by fleet.

Usage

```
empirobscorrplot(res, ...)

## S3 method for class 'samres'
empirobscorrplot(res, ...)
```

Arguments

<code>res</code>	the object returned from <code>residuals.sam</code>
<code>...</code>	extra arguments to plot

```
equilibriumbiomassplot
      SAM equilibrium biomass plot
```

Description

SAM equilibrium biomass plot

Usage

```
equilibriumbiomassplot(fit, ...)

## Default S3 method:
equilibriumbiomassplot(fit, ...)

## S3 method for class 'samforecast'
equilibriumbiomassplot(fit, ...)

## S3 method for class 'hcr'
equilibriumbiomassplot(fit, ...)
```

Arguments

<code>fit</code>	the object returned from <code>sam.fit</code>
<code>...</code>	extra arguments transferred to <code>plot</code> including the following: <code>add</code> logical, plotting is to be added on existing plot <code>ci</code> logical, confidence intervals should be plotted <code>cicol</code> color to plot the confidence polygon

Details

Plot of deterministic equilibrium spawners per recruit assuming biological parameters and selectivity for that year remains unchanged in the future.

```
equilibriumbiomasstable
      equilibrium biomass table
```

Description

equilibrium biomass table

Usage

```
equilibriumbiomasstable(fit, ...)
```

```
## Default S3 method:
equilibriumbiomasstable(fit, ...)
```

Arguments

```
fit          ...
...          extra arguments not currently used
```

Details

```
...
```

faytable	<i>F-at-age table</i>
----------	-----------------------

Description

F-at-age table

Usage

```
faytable(fit, ...)
```

```
## S3 method for class 'sam'
faytable(fit, fleet = which(fit$data$fleetTypes == 0), ...)
```

Arguments

```
fit          a fitted object of class 'sam' as returned from sam.fit
...          extra arguments not currently used
fleet        the fleet number(s) to return F summed for (default is to return the sum of all
              residual fleets).
```

Details

```
...
```

fbarplot

SAM Fbar plot

Description

SAM Fbar plot

Usage

```
fbarplot(fit, ...)  
  
## S3 method for class 'sam'  
fbarplot(  
  fit,  
  partial = TRUE,  
  drop = NULL,  
  pcol = "lightblue",  
  page = NULL,  
  plot = TRUE,  
  ...  
)  
  
## S3 method for class 'samset'  
fbarplot(  
  fit,  
  partial = FALSE,  
  drop = NULL,  
  pcol = "lightblue",  
  page = NULL,  
  ...  
)  
  
## S3 method for class 'samforecast'  
fbarplot(  
  fit,  
  partial = FALSE,  
  drop = NULL,  
  pcol = "lightblue",  
  page = NULL,  
  ...  
)  
  
## S3 method for class 'hcr'  
fbarplot(  
  fit,  
  partial = FALSE,  
  drop = NULL,
```

```

    pcol = "lightblue",
    page = NULL,
    ...
)

```

Arguments

fit	the object returned from sam.fit
...	extra arguments transferred to plot including the following: add logical, plotting is to be added on existing plot ci logical, confidence intervals should be plotted cicol color to plot the confidence polygon
partial	true if included partial F's are to be plotted
drop	number of years to be left unplotted at the end. Default (NULL) is to not show years at the end with no catch information
pcol	color of partial lines
page	partial ages to plot
plot	true if fbar should be plotted

Details

Plot the defined fbar.

fbartable	<i>Fbar table</i>
-----------	-------------------

Description

Fbar table

Usage

```
fbartable(fit, ...)
```

```
## Default S3 method:
fbartable(fit, ...)
```

Arguments

fit	...
...	extra arguments not currently used

Details

...

fitfromweb

Read a fitted model from stockassessment.org

Description

Read a fitted model from stockassessment.org

Usage

```
fitfromweb(stockname, character.only = FALSE, return.all = FALSE)
```

Arguments

stockname	The short-form name of a stock on stockassessment.org. This will (currently?) not work for stocks defined via the AD Model builder version of SAM.
character.only	a logical indicating whether 'stockname' can be assumed to be a character string
return.all	a logical indicating whether everything from model.RData should be returned in an environment

Details

...

fitplot

Plots fit to data

Description

Plots fit to data

Usage

```
fitplot(fit, log = TRUE, ...)
```

```
## S3 method for class 'sam'
```

```
fitplot(fit, log = TRUE, fleets = unique(fit$data$aux[, "fleet"]), ...)
```

Arguments

fit	the object returned from sam.fit
log	should the plot be against log-obs
...	extra arguments to plot
fleets	an integer vector of fleets to plot. Default is all of them

forecast	<i>forecast function to do shortterm</i>
----------	--

Description

forecast function to do shortterm

Usage

```
forecast(
  fit,
  fscale = NULL,
  catchval = NULL,
  catchval.exact = NULL,
  fval = NULL,
  nextssb = NULL,
  landval = NULL,
  cwF = NULL,
  nosim = 1000,
  year.base = max(fit$data$years),
  ave.years = max(fit$data$years) + (-4:0),
  rec.years = max(fit$data$years) + (-9:0),
  label = NULL,
  overwriteSelYears = NULL,
  deterministic = FALSE,
  processNoiseF = TRUE,
  customWeights = NULL,
  customSel = NULL,
  lagR = FALSE,
  splitLD = FALSE,
  addTSB = FALSE,
  useSWmodel = (fit$conf$stockWeightModel >= 1),
  useCWmodel = (fit$conf$catchWeightModel >= 1),
  useMOModel = (fit$conf$matureModel >= 1),
  useNMmodel = (fit$conf$mortalityModel >= 1),
  savesim = FALSE,
  cf.cv.keep.cv = matrix(NA, ncol = 2 * sum(fit$data$fleetTypes == 0), nrow =
    length(catchval)),
  cf.cv.keep.fv = matrix(NA, ncol = 2 * sum(fit$data$fleetTypes == 0), nrow =
    length(catchval)),
  cf.keep.fv.offset = matrix(0, ncol = sum(fit$data$fleetTypes == 0), nrow =
    length(catchval)),
  estimate = median
)
```

Arguments

`fit` an assessment object of type `sam`, as returned from the function `sam.fit`

<code>fscale</code>	a vector of f-scales. See details.
<code>catchval</code>	a vector of target catches. See details.
<code>catchval.exact</code>	a vector of target catches which will be met without noise. See details.
<code>fval</code>	a vector of target f values. See details.
<code>nextssb</code>	a vector target SSB values the following year. See details
<code>landval</code>	a vector of target catches. See details.
<code>cwF</code>	a vector target custom weighted F values. <code>customWeights</code> must also be specified
<code>nosim</code>	number of simulations default is 1000
<code>year.base</code>	starting year default last year in assessment. Currently it is only supported to use last assessment year or the year before
<code>ave.years</code>	vector of years to average for weights, maturity, M and such
<code>rec.years</code>	vector of years to use to resample recruitment from
<code>label</code>	optional label to appear in short table
<code>overwriteSelYears</code>	if a vector of years is specified, then the average selectivity of those years is used (not recommended)
<code>deterministic</code>	option to turn all process noise off (not recommended, as it will likely cause bias)
<code>processNoiseF</code>	option to turn off process noise in F
<code>customWeights</code>	a vector of same length as number of age groups giving custom weights (currently only used for weighted average of F calculation)
<code>customSel</code>	supply a custom selection vector that will then be used as fixed selection in all years after the final assessment year (not recommended)
<code>lagR</code>	if the second youngest age should be reported as recruits
<code>splitLD</code>	if TRUE the result is split in landing and discards
<code>addTSB</code>	if TRUE the total stock biomass (TSB) is added
<code>useSWmodel</code>	if TRUE the catch mean weight predicted from the assessment model is used (can only be used for configurations supporting this)
<code>useCWmodel</code>	if TRUE the catch mean weight predicted from the assessment model is used (can only be used for configurations supporting this)
<code>useM0model</code>	if TRUE the proportion mature predicted from the assessment model is used (can only be used for configurations supporting this)
<code>useNMmodel</code>	if TRUE the natural mortality predicted from the assessment model is used (can only be used for configurations supporting this)
<code>savesim</code>	save the individual simulations
<code>cf.cv.keep.cv</code>	exotic option
<code>cf.cv.keep.fv</code>	exotic option
<code>cf.keep.fv.offset</code>	exotic option
<code>estimate</code>	the summary function used (typically mean or median)

Details

There are three ways to specify a scenario. If e.g. four F values are specified (e.g. `fval=c(.1,.2,.3,4)`), then the first value is used in the last assessment year (`base.year`), and the three following in the three following years. Alternatively F's can be specified by a scale, or a target catch. Only one option can be used per year. So for instance to set a catch in the first year and an F-scale in the following one would write `catchval=c(10000,NA,NA,NA)`, `fscale=c(NA,1,1,1)`. The length of the vector specifies how many years forward the scenarios run.

Value

an object of type `samforecast`

forecastMSY

Estimating Fmsy

Description

Estimating Fmsy

Usage

```
forecastMSY(
  fit,
  nYears = 100,
  nlminb.control = list(eval.max = 2000, iter.max = 2000),
  rec.years = c(),
  ave.years = max(fit$data$years) + (-9:0),
  processNoiseF = FALSE,
  ...
)

## S3 method for class 'sam'
forecastMSY(
  fit,
  nYears = 100,
  nlminb.control = list(eval.max = 2000, iter.max = 2000, trace = 1),
  rec.years = c(),
  ave.years = max(fit$data$years) + (-9:0),
  processNoiseF = FALSE,
  jacobianHScale = 0.5,
  nCatchAverageYears = 20,
  ...
)
```

Arguments

<code>fit</code>	a SAM fit
<code>nYears</code>	Number of years to forecast
<code>nlminb.control</code>	list of control variables for nlminb
<code>rec.years</code>	Numeric vector of years to use (to calculate mean and standard deviation) for recruitment. An empty vector will use the recruitment model.
<code>ave.years</code>	vector of years to average for weights, maturity, M and such. Following ICES guidelines, the default is the last 10 years.
<code>processNoiseF</code>	Should random walk process noise be used for F?
<code>...</code>	other arguments passed to forecast
<code>jacobianHScale</code>	Scale step size in jacobian calculation
<code>nCatchAverageYears</code>	Number of years to average catch over for finding MSY

References

Albertsen, C. M. and Trijoulet, V. (2020) Model-based estimates of reference points in an age-based state-space stock assessment model. *Fisheries Research*, 230, 105618. doi: 10.1016/j.fishres.2020.105618

See Also

[forecast referencepoints](#)

<code>fselectivityplot</code>	<i>SAM F-selectivity plot</i>
-------------------------------	-------------------------------

Description

SAM F-selectivity plot

Usage

```
fselectivityplot(fit, cexAge = 1, ...)
```

```
## S3 method for class 'sam'
fselectivityplot(fit, cexAge = 1, ...)
```

Arguments

<code>fit</code>	An object returned from <code>sam.fit</code>
<code>cexAge</code>	cex variable giving the size of the age numbers
<code>...</code>	extra arguments transferred to <code>barplot</code> and <code>text</code>

Details

Plots selectivity in F.

generationlengthplot *SAM generation length plot*

Description

SAM generation length plot

Usage

```
generationlengthplot(fit, ...)  
  
## Default S3 method:  
generationlengthplot(fit, ...)  
  
## S3 method for class 'samforecast'  
generationlengthplot(fit, ...)  
  
## S3 method for class 'hcr'  
generationlengthplot(fit, ...)
```

Arguments

fit	the object returned from sam.fit
...	extra arguments transferred to plot including the following: add logical, plotting is to be added on existing plot ci logical, confidence intervals should be plotted cicol color to plot the confidence polygon

Details

Plot of life expectancy

generationlengthtable *Generation length table*

Description

Generation length table

Usage

```
generationlengthtable(fit, ...)  
  
## Default S3 method:  
generationlengthtable(fit, ...)
```

Arguments

fit	...
...	extra arguments not currently used

Details

...

getAllDerivedValues	<i>Update sam fit with additional derived values</i>
---------------------	--

Description

Update sam fit with additional derived values

Usage

```
getAllDerivedValues(fit)
```

Arguments

fit	sam fit returned by sam.fit
-----	-----------------------------

Value

Updated sam fit

getFleet	<i>Extract a fleet observed or predicted value from a fitted object</i>
----------	---

Description

Extract a fleet observed or predicted value from a fitted object

Usage

```
getFleet(fit, fleet, pred = "FALSE")
```

Arguments

fit	A fitted object as returned from sam.fit
fleet	The fleet number
pred	Should it be predicted value, default is observed

Details

Extract for example the observed or predicted catch at age of fleet "fleet"

Value

A matrix of observed or predicted values for fleet "fleet"

getLowerBounds	<i>Bounds</i>
----------------	---------------

Description

Bounds

Usage

```
getLowerBounds(parameters, conf)
```

Arguments

parameters	initial values for the model in a format similar to what is returned from the defpar function
conf	model configuration in a format similar to what is returned from the defcon function

Value

a named list

getResidualFleets	<i>Extract a list of catch fleets</i>
-------------------	---------------------------------------

Description

Extract a list of catch fleets

Usage

```
getResidualFleets(fit, pred = "FALSE")
```

Arguments

fit	A fitted object as returned from sam.fit
pred	Should it be predicted value, default is observed

Value

A list of matrices of observed or predicted values for catch fleets

getUpperBounds	<i>Bounds</i>
----------------	---------------

Description

Bounds

Usage

```
getUpperBounds(parameters, conf)
```

Arguments

parameters	initial values for the model in a format similar to what is returned from the defpar function
conf	model configuration in a format similar to what is returned from the defcon function

Value

a named list

grad	<i>Calculate gradient of a function</i>
------	---

Description

Calculate gradient of a function

Usage

```
grad(
  func,
  x,
  h = abs(1e-04 * x) + 1e-04 * (abs(x) < sqrt(.Machine$double.eps/7e-07)),
  ...
)
```

Arguments

func	function
x	parameter values
h	step size
...	passed to func

Value

gradient vector

hcr	<i>Harvest control rule forecast</i>
-----	--------------------------------------

Description

The formula below is used to determine a new F based on the previous SSB.

$$F = \begin{cases} F_{cap} & SSB < B_{cap} \\ \min(F_{target}, \max(F_{origin}, (SSB - B_{origin}) \cdot (F_{target} - F_{origin}) / (B_{trigger} - B_{origin}))) & SSB \geq B_{cap} \end{cases}$$

If $B_{trigger} = B_{origin}$ and $SSB \geq B_{cap}$, F_{target} is always returned.

Usage

```
hcr(fit, ...)

## S3 method for class 'sam'
hcr(
  fit,
  nYears = 20,
  Ftarget,
  Btrigger,
  Forigin = 0,
  Borigin = 0,
  Fcap = 0,
  Bcap = 0,
  nosim = 10000,
  ave.years = max(fit$data$years) + (-4:0),
  rec.years = numeric(0),
  preForecast = list(),
  currentSSB = FALSE,
  ...
)
```

Arguments

fit	A SAM fit
...	additional arguments passed to modelforecast
nYears	Number of years to forecast
Ftarget	Target F for high SSB
Btrigger	SSB that triggers the control rule
Forigin	F used for SSB = Borigin

Borigin	Between Blim and Btrigger, F values are selected based on linear interpolation from Forigin to Ftarget
Fcap	F for $SSB < Bcap$
Bcap	SSB for which Fcap is used below
nosim	Number of simulations to do. If NULL a model forecast based on the Laplace approximation is used
ave.years	vector of years to average for weights, maturity, M and such
rec.years	vector of years to use to resample recruitment from. If an empty vector is given, recruitment is based on the fitted model.
preForecast	list of forecast parameters (i.e., fval, fscale, catchval, landval, or nextssb) to use before the HCR
currentSSB	if TRUE, SSB at the beginning of the control rule year is used. If FALSE, SSB at the beginning of the previous year is used.

Value

model forecast using a harvest control rule

hcr model forecast object

See Also

modelforecast

ibc

Integrated spline basis for use with formula interface

Description

Integrated spline basis for use with formula interface

Usage

```
ibc(x, df = 3L, knots = NULL, Boundary.knots = range(x), intercept = FALSE)
```

Arguments

x	Points to evaluate the basis in
df	Degrees of freedom
knots	Internal knots. If NULL, they are selected from quantiles of x.
Boundary.knots	Boundary knots. Defaults to range of x
intercept	Include an intercept in basis?

Value

A spline basis

icesAdviceRule *Forecast with an ICES advice rule*

Description

Forecast with an ICES advice rule

Usage

```
icesAdviceRule(
  x,
  Fmsy,
  MSYBtrigger,
  Blim,
  nosim = 10000,
  ave.years = max(x$data$years) + (-4:0),
  rec.years = numeric(0),
  preForecast = list(),
  currentSSB = FALSE,
  ...
)
```

Arguments

x	Fitted assessment model
Fmsy	ICES Fmsy which is used as target F
MSYBtrigger	ICES MSYBtrigger below which F is reduced
Blim	ICES Blim below which F is set to zero.
nosim	Number of simulations to do. If NULL a model forecast based on the Laplace approximation is used
ave.years	vector of years to average for weights, maturity, M and such
rec.years	vector of years to use to resample recruitment from. If an empty vector is given, recruitment is based on the fitted model.
preForecast	list of forecast parameters (i.e., fval, fscale, catchval, landval, or nextssb) to use before the HCR
currentSSB	if TRUE, SSB at the begining of the control rule year is used. If FALSE, SSB at the begining of the previous year is used.
...	Other arguments passes to hcr

Value

hcr object

Warning

The function does not make a short term forecast to see if fishing can continue below Blim.

References

ICES (2021) Advice on fishing opportunities. DOI: 10.17895/ices.advice.7720

See Also

[hcr](#)

`iibc` *Double integrated spline basis for use with formula interface*

Description

Double integrated spline basis for use with formula interface

Usage

```
iibc(x, df = 3L, knots = NULL, Boundary.knots = range(x), intercept = FALSE)
```

Arguments

<code>x</code>	Points to evaluate the basis in
<code>df</code>	Degrees of freedom
<code>knots</code>	Internal knots. If NULL, they are selected from quantiles of <code>x</code> .
<code>Boundary.knots</code>	Boundary knots. Defaults to range of <code>x</code>
<code>intercept</code>	Include an intercept in basis?

Value

A spline basis

`is.whole.positive.number`
Function to test if x is ...

Description

Function to test if x is ...

Usage

```
is.whole.positive.number(x, tol = .Machine$double.eps^0.5)
```

Arguments

x	number
tol	precision

Details

...

jacobian	<i>Calculate jacobian of a function</i>
----------	---

Description

Calculate jacobian of a function

Usage

```
jacobian(  
  func,  
  x,  
  h = abs(1e-04 * x) + 1e-04 * (abs(x) < sqrt(.Machine$double.eps/7e-07)),  
  ...  
)
```

Arguments

func	function
x	parameter values
h	step size
...	passed to func

Value

jacobian matrix

jit*Jitter runs*

Description

Jitter runs

Usage

```
jit(  
  fit,  
  nojit = 10,  
  par = defpar(fit$data, fit$conf),  
  sd = 0.25,  
  ncores = detectCores()  
)
```

Arguments

<code>fit</code>	a fitted model object as returned from <code>sam.fit</code>
<code>nojit</code>	a list of vectors. Each element in the list specifies a run where the fleets mentioned are omitted
<code>par</code>	initial values to jitter around. The default ones are returned from the <code>defpar</code> function
<code>sd</code>	the standard deviation used to jitter the initial values (most parameters are on a log scale, so similar to <code>cv</code>)
<code>ncores</code>	the number of cores to attempt to use

Details

...

Value

A "samset" object, which is basically a list of sam fits

leaveout	<i>leaveout run</i>
----------	---------------------

Description

leaveout run

Usage

```
leaveout(
  fit,
  fleet = as.list(2:fit$data$noFleets),
  ncores = detectCores(),
  ...
)
```

Arguments

<code>fit</code>	a fitted model object as returned from <code>sam.fit</code>
<code>fleet</code>	a list of vectors. Each element in the list specifies a run where the fleets mentioned are omitted
<code>ncores</code>	the number of cores to attempt to use
<code>...</code>	extra arguments to sam.fit

Details

...

lifeexpectancyplot	<i>SAM life expectancy plot</i>
--------------------	---------------------------------

Description

SAM life expectancy plot

Usage

```
lifeexpectancyplot(fit, atRecruit = TRUE, ...)

## Default S3 method:
lifeexpectancyplot(fit, atRecruit = TRUE, ylimAdd = fit$conf$maxAge, ...)

## S3 method for class 'samforecast'
lifeexpectancyplot(fit, atRecruit = TRUE, ylimAdd = fit$conf$maxAge, ...)

## S3 method for class 'hcr'
lifeexpectancyplot(fit, atRecruit = TRUE, ylimAdd = fit$conf$maxAge, ...)
```

Arguments

<code>fit</code>	the object returned from <code>sam.fit</code>
<code>atRecruit</code>	If true, show life expectancy given survival until <code>minAge</code> , otherwise show life expectancy at birth
<code>...</code>	extra arguments transferred to plot including the following: <code>add</code> logical, plotting is to be added on existing plot <code>ci</code> logical, confidence intervals should be plotted <code>cicol</code> color to plot the confidence polygon
<code>ylimAdd</code>	values to add when calculating <code>ylim</code> for the plot

Details

Plot of life expectancy

`lifeexpectancytable` *Life expectancy table*

Description

Life expectancy table

Usage

```
lifeexpectancytable(fit, atRecruit = TRUE, ...)

## Default S3 method:
lifeexpectancytable(fit, atRecruit = TRUE, ...)
```

Arguments

<code>fit</code>	...
<code>atRecruit</code>	If true, show life expectancy given survival until <code>minAge</code> , otherwise show life expectancy at birth
<code>...</code>	extra arguments not currently used

Details

...

loadConf	<i>Loads a model configuration from a file</i>
----------	--

Description

Loads a model configuration from a file

Usage

```
loadConf(dat, file, patch = TRUE)
```

Arguments

dat	sam data list as returned from the function <code>setup.sam.data</code>
file	the file to read the configuration from
patch	logical if TRUE missing entries will be automatically filled with default values

Details

function useful loading a model configuration. Such a configuration can be saved via the `saveConf` function

logLik.sam	<i>Log likelihood of sam object</i>
------------	-------------------------------------

Description

Log likelihood of sam object

Usage

```
## S3 method for class 'sam'  
logLik(object, ...)
```

Arguments

object	sam fitted object as returned from the <code>sam.fit</code> function
...	extra arguments

Details

log likelihood of a sam model run

modelDescription	<i>Description of model</i>
------------------	-----------------------------

Description

Description of model

Usage

```
modelDescription(fit, ...)
```

Arguments

fit	returned object from sam.fit
...	Additional parameters to be passed to ...

Details

...

modelforecast	<i>Model based forecast function</i>
---------------	--------------------------------------

Description

Model based forecast function
 Model based forecast function

Usage

```
modelforecast(fit, ...)

## S3 method for class 'sam'
modelforecast(
  fit,
  constraints = NULL,
  fscale = NULL,
  catchval = NULL,
  fval = NULL,
  nextssb = NULL,
  landval = NULL,
  findMSY = NULL,
  hcr = NULL,
  nosim = 0,
  year.base = max(fit$data$years),
```

```

ave.years = c(),
rec.years = c(),
label = NULL,
overwriteSelYears = NULL,
deterministicF = FALSE,
processNoiseF = FALSE,
resampleFirst = !is.null(nosim) && nosim > 0,
customSel = NULL,
lagR = FALSE,
splitLD = FALSE,
addTSB = FALSE,
biasCorrect = FALSE,
returnAllYears = FALSE,
nCatchAverageYears = 1,
returnObj = FALSE,
hcrConf = numeric(0),
hcrCurrentSSB = 0,
progress = TRUE,
estimate = median,
silent = TRUE,
newton_config = NULL,
...
)

```

Arguments

<code>fit</code>	an assessment object of type <code>sam</code> , as returned from the function <code>sam.fit</code>
<code>...</code>	other variables used by the methods
<code>constraints</code>	a character vector of forecast constraint specifications
<code>fscale</code>	a vector of f-scales. See details.
<code>catchval</code>	a vector of target catches. See details.
<code>fval</code>	a vector of target f values. See details.
<code>nextssb</code>	a vector target SSB values the following year. See details.
<code>landval</code>	a vector of target catches. See details.
<code>findMSY</code>	Should not be used. See forecastMSY .
<code>hcr</code>	Should not be used. See hcr .
<code>nosim</code>	number of simulations. If 0, the Laplace approximation is used for forecasting.
<code>year.base</code>	starting year default last year in assessment. Currently it is only supported to use last assessment year or the year before
<code>ave.years</code>	vector of years to average for weights, maturity, M and such
<code>rec.years</code>	vector of years to use to resample recruitment from. If the vector is empty, the stock recruitment model is used.
<code>label</code>	optional label to appear in short table

overwriteSelYears	if a vector of years is specified, then the average selectivity of those years is used (not recommended)
deterministicF	option to set F variance to (almost) zero (not recommended)
processNoiseF	option to turn off process noise in F
resampleFirst	Resample base year when nosim > 0?
customSel	supply a custom selection vector that will then be used as fixed selection in all years after the final assessment year (not recommended)
lagR	if the second youngest age should be reported as recruits
splitLD	if TRUE the result is split in landing and discards
addTSB	if TRUE the total stock biomass (TSB) is added
biasCorrect	Do bias correction of reported variables. Can be turned off to reduce running time (not recommended).
returnAllYears	If TRUE, all years are bias corrected. Otherwise, only forecast years are corrected.
nCatchAverageYears	Should not be used. See forecastMSY .
returnObj	Only return TMB object?
hcrConf	Should not be used. See hcr .
hcrCurrentSSB	Should not be used. See hcr .
progress	Show progress bar for simulations?
estimate	the summary function used (typically mean or median) for simulations
silent	Passed to MakeADFun. Should the TMB object be silent?
newton_config	Configuration for newton optimizer to find F values. See <code>?TMB::newton</code> for details. Use NULL for TMB defaults.

Details

Function to forecast the model under specified catch constraints. In the forecast, catch constraints are used to set the mean of the $\log(F)$ process. Therefore, catch constraints are not matched exactly in individual simulations (unlike the forecast function simulations). Likewise, the summary of a specific set of simulations will not match exactly due to random variability. By default, recruitment is forecasted using the estimated recruitment model. If a vector of recruitment years is given, recruitment is forecasted using a log-normal distribution with the same mean and variance as the recruitment in the years given. This is different from the forecast function, which samples from the recruitment estimates. Catch scenarios are specified by a vector of target constraints. The first value determines F in the year after the base year.

Value

an object of type `samforecast`

F based constraints

Forecasts for F values are specified by the format "F[f,a0-a1]=x" where f is the residual catch fleet and a0-a1 is an age range. For example, "F[2,2-4]=0.3" specifies that the average F for the second fleet over ages 2-4 should be 0.3. If an "*" is added to the target value, the target will be relative to the year before. For example, "F[2,2-4]=0.9*" specifies that the average F for the second fleet over ages 2-4 should be 90% of the value from the year before. If the fleet is omitted (e.g., F[2-4]), the target is for the total F. If the age range is omitted (e.g., F[2]), the fleet range of the model is used. Likewise, both fleet and age range can be omitted (e.g., F=0.3) to specify a value for total F with the range used in the model.

Catch/Landing based constraints

Forecasts for catch and landing values are specified by the format "C[f,a0-a1]=x" for catch and "L[f,a0-a1]" for landings. If the age range is omitted, all modelled ages are used. Otherwise, the format is similar to F based scenarios. If an "*" is added to the target value, the target will be relative to the year before. Further, the catch target for a fleet can be relative to the total by adding "*C" or to another fleet by adding "*C[f]" where f is the fleet number. The same age range will always be used. Likewise, relative landing targets can be specified using "*", "*L", or "*L[f]" for targets relative to last year, the total, or fleet f, respectively.

SSB/TSB at the beginning of next year based constraints

Forecasts for spawning stock biomass (SSB) and total stock biomass (TSB) values are specified by the format "SSB[a0-a1]=x" for SSB and "TSB[a0-a1]" for TSB. The format is similar to catch/landing based scenarios. However, fleets have no effect. If an "*" is added to the target value, the target will be relative to the year before. Note that SSB and TSB used for catch constraints are at the beginning of the next year to avoid dependence on future F values. Therefore, the values will differ from the output SSB and TSB estimates if propM or propF are not zero.

Combining constraints

Constraints for different fleets can be combined by "&". For example, "F[2-4]=0.5 & C[2]=10000" specifies that total Fbar over ages 2-4 should be 0.5 while the catch for the second residual catch fleet should be 10,000t. The constraints cannot affect within-fleet selectivity. Therefore, a fleet can at most have one constraint per year, and the total number of constraints cannot exceed the number of catch fleets. That is, if a constraint is given for the sum of fleets, there must be at least one fleet without any constraints. For fleets where no constraints are given, a constraint is set to keep their relative Fs constant.

Values relative to previous year

Catch constraints specified as specific values are inherently different from catch constraints specified as relative values, even if they lead to the same F. Catch constraints specified as relative values will propagate the uncertainty in, e.g, F from previous years whereas constraints specified as specific values will not. This is different from the [forecast](#) function where, for example, a forecast using fval is the same as a forecast using fscale, if they lead to the same F.

Old specification

It is also possible to specify forecast constraints in a way similar to the `forecast` function. There are four ways to specify a scenario. If e.g. four F values are specified (e.g. `fval=c(.1,.2,.3,4)`), then the first value is used in the year after the last assessment year (`base.year + 1`), and the three following in the three following years. Alternatively F's can be specified by a scale, or a target catch. Only one option can be used per year. So for instance to set a catch in the first year and an F-scale in the following one would write `catchval=c(10000,NA,NA,NA)`, `fscale=c(NA,1,1,1)`. If only NA's are specified in a year, the F model is used for forecasting. The length of the vector specifies how many years forward the scenarios run. Unlike the forecast function, no value should be given for the base year. Internally, the old specification is translated such that "fval=x" becomes "F=x", "fscale=x" becomes "F=x*", "catchval=x" becomes "C=x", "nextssb=x" becomes "SSB=x", and "landval=x" becomes "L=x".

Forecasts using Laplace approximation or simulations

NA

Warnings

Long term forecasts with random walk recruitment can lead to unstable behaviour and difficulties finding suitable F values for the constraints. If no suitable F value can be found, an error message will be shown, and F values will be NA or NaN. Likewise, forecasts leading to high F values in some years may cause problems for the optimization as they will be used as starting values for the next years. Since the model works on log space, all target values should be strictly positive. Values too close to zero may cause problems.

See Also

`forecast`

modeltable

model table

Description

model table

Usage

```
modeltable(fits, ...)
```

```
## S3 method for class 'sam'
modeltable(fits, ...)
```

```
## S3 method for class 'samset'
modeltable(fits, ...)
```

Arguments

`fits` A sam fit as returned from the `sam.fit` function, or a collection `c(fit1, fit2, ...)` of such fits

`...` extra arguments not currently used

Details

...

modelVersionInfo	<i>Description of model</i>
------------------	-----------------------------

Description

Description of model

Usage

```
modelVersionInfo(fit, ...)
```

Arguments

`fit` returned object from `sam.fit`

`...` Additional parameters to be passed to ...

Details

Writes a string to install the version of the package which was used to run the model.

mohn	<i>Mohn's rho calculation</i>
------	-------------------------------

Description

Mohn's rho calculation

Usage

```
mohn(fits, what = NULL, lag = 0, ...)
```

```
## S3 method for class 'samset'
mohn(fits, what = NULL, lag = 0, ...)
```

Arguments

fits	a samset object as returned from the retro function.
what	a function computing the quantity to calculate Mohn's rho for (default NULL computes Fbar, SSB, and R).
lag	lag applied to fits and reference fit.
...	extra arguments

Details

...

nobs.sam	<i>Extract number of observations from sam object</i>
----------	---

Description

Extract number of observations from sam object

Usage

```
## S3 method for class 'sam'
nobs(object, ...)
```

Arguments

object	sam fitted object as returned from the sam.fit function
...	extra arguments

Details

...

nscodConf	<i>nscodConf</i>
-----------	------------------

Description

nscodConf

Usage

```
data("nscodConf")
```


Format

The format is: \$ minAge \$ maxAge \$ maxAgePlusGroup \$ keyLogFsta \$ corFlag \$ keyLogFpar \$ keyQpow \$ keyVarF \$ keyVarLogN \$ keyVarObs \$ stockRecruitmentModelCode \$ noScaledYears \$ keyScaledYears \$ keyParScaledYA \$ fbarRange

Details

...

Source

...

References

...

Examples

```
data(nscodConf)
## maybe str(nscodConf) ; plot(nscodConf) ...
```

nscodData

nscodData

Description

nscodData

Usage

```
data("nscodData")
```

Format

The format is: \$ noFleets \$ fleetTypes \$ sampleTimes \$ noYears \$ years \$ nobs \$ idx1 \$ idx2 \$ aux \$ logobs \$ propMat \$ stockMeanWeight \$ catchMeanWeight \$ natMor \$ landFrac \$ disMeanWeight \$ landMeanWeight \$ propF \$ propM

Details

...

Source

...

References

...

Examples

```
data(nscodData)
## maybe str(nscodData) ; plot(nscodData) ...
```

nscodParameters	<i>nscodParameters</i>
-----------------	------------------------

Description

nscodParameters

Usage

```
data("nscodParameters")
```

Format

The format is: List of 14 \$ logFpar \$ logQpow \$ logSdLogFsta \$ logSdLogN \$ logSdLogObs \$ rec_loga \$ rec_logb \$ itrans_rho \$ logScale \$ logScaleSSB \$ logPowSSB \$ logSdSSB \$ logF \$ logN

Details

...

Source

...

References

...

Examples

```
data(nscodParameters)
## maybe str(nscodParameters) ; plot(nscodParameters) ...
```

`obscov`*Extract observation covariance matrices from a SAM fit*

Description

Extract observation covariance matrices from a SAM fit

Usage

```
obscov(fit, corr = FALSE, ...)
```

```
## S3 method for class 'sam'  
obscov(fit, corr = FALSE, ...)
```

```
## S3 method for class 'samset'  
obscov(fit, corr = FALSE, ...)
```

Arguments

<code>fit</code>	the object returned from <code>sam.fit</code>
<code>corr</code>	if TRUE return correlation matrices rather than covariances
<code>...</code>	extra arguments not currently used

Value

a list of matrices

`parplot`*SAM parameter plot*

Description

SAM parameter plot

Usage

```
parplot(fit, cor.report.limit = 0.95, ...)
```

```
## S3 method for class 'sam'  
parplot(fit, cor.report.limit = 0.95, ...)
```

```
## S3 method for class 'samset'  
parplot(fit, cor.report.limit = 0.95, ...)
```

Arguments

`fit` the object returned from `sam.fit`
`cor.report.limit` correlations with absolute value > this number is reported in the plot
`...` extra arguments transferred to plot

Details

Plot of all estimated model parameters (fixed effects). Shown with confidence interval.

partable	<i>parameter table</i>
----------	------------------------

Description

parameter table

Usage

```
partable(fit, ...)

## S3 method for class 'sam'
partable(fit, ...)
```

Arguments

`fit` ...
`...` extra arguments not currently used

Details

...

plot.hcr	<i>Plot hcr object</i>
----------	------------------------

Description

Plot hcr object

Usage

```
## S3 method for class 'hcr'
plot(x, ...)
```

Arguments

x output from the [hcr](#) function
 ... extra arguments

Details

...

plot.sam	<i>Plot sam object</i>
----------	------------------------

Description

Plot sam object

Usage

```
## S3 method for class 'sam'
plot(x, ...)
```

Arguments

x fitted object as returned from the [sam.fit](#) function.
 ... extra arguments (not possible to use add=TRUE — instead collect to a list of fits using e.g the [c\(...\)](#), and then plot that collected object).

Details

gives a 3 plot overview plot of ssb, fbar, and recruits. These plots are available individually via the functions [ssbplot](#), [fbarplot](#), and [recplot](#).

plot.samforecast	<i>Plot samforecast object</i>
------------------	--------------------------------

Description

Plot samforecast object

Usage

```
## S3 method for class 'samforecast'
plot(x, ...)
```

Arguments

x fitted object as returned from the `sam.fit` function
 ... extra arguments

Details

...

plot.samres *Plot sam residuals*

Description

Plot sam residuals

Usage

```
## S3 method for class 'samres'
plot(x, type = "bubble", ...)
```

Arguments

x an object of type 'samres' as returned from functions `residuals.sam` or `progres`.
 type either "bubble" (default) or "summary"
 ... extra arguments

Details

In the "bubble" type red indicate negative residuals and blue positive. The area of the circles scales with the absolute size of the residuals.

Examples

```
## Not run:
data(nscodData)
data(nscodConf)
data(nscodParameters)
fit <- sam.fit(nscodData, nscodConf, nscodParameters)
par(ask=FALSE)
plot(residuals(fit))

## End(Not run)
```

plot.samset

Plot sam object

Description

Plot sam object

Usage

```
## S3 method for class 'samset'  
plot(x, ...)
```

Arguments

x fitted object as returned from the `sam.fit` function.
... extra arguments

Details...

plot.samypr

Plot sam object

Description

Plot sam object

Usage

```
## S3 method for class 'samypr'  
plot(x, ...)
```

Arguments

x ...
... extra arguments

Details

...

plotby *Plot by one or two*

Description

Plot by one or two

Usage

```
plotby(  
  x = NULL,  
  y = NULL,  
  z = NULL,  
  x.line = NULL,  
  y.line = NULL,  
  z.line = NULL,  
  by = NULL,  
  bubblescale = 1,  
  x.common = !is.null(x),  
  y.common = !is.null(y),  
  z.common = !is.null(z),  
  xlab = NULL,  
  ylab = NULL,  
  xlim = NULL,  
  ylim = NULL,  
  zmax = NULL,  
  axes = TRUE,  
  ...  
)
```

Arguments

x	numeric vector of points to be plotted
y	numeric vector of points to be plotted
z	numeric vector of points to be plotted
x.line	numeric vector of points of line to be added
y.line	numeric vector of points of line to be added
z.line	numeric vector of points of line to be added
by	vector or two column matrix to create sub sets from
bubblescale	scaling of bubble size
x.common	logical: use same x-axis for all plots
y.common	logical: use same y-axis for all plots
z.common	logical: use same z-axis for all plots
xlab	normal graphical parameter

ylab	normal graphical parameter
xlim	normal graphical parameter
ylim	normal graphical parameter
zmax	internally used to scale bubbles similarly
axes	normal graphical parameter
...	additional graphical parameters

Details

Function used for splitting plots e.g. used to plot residuals

Examples

```
exdat<-expand.grid(age=1:5, year=1950:2016, fleet=1:3)
exdat$perfectres<-rnorm(nrow(exdat))
attach(exdat)
par(ask=FALSE)
plotby(year,age,perfectres, by=fleet)
detach(exdat)
```

plotit

Plot helper

Description

Plot helper

Usage

```
plotit(fit, what, ...)

## S3 method for class 'sam'
plotit(
  fit,
  what,
  x = fit$data$years,
  ylab = what,
  xlab = "Years",
  ex = numeric(0),
  trans = function(x) x,
  add = FALSE,
  ci = TRUE,
  cicol = gray(0.5, alpha = 0.5),
  addCI = NA,
  drop = 0,
  unnamed.basename = "current",
```

```
xlim = NULL,  
ylim = NULL,  
ylimAdd = NA,  
...  
)  
  
## S3 method for class 'samset'  
plotit(  
  fit,  
  what,  
  x = fit$data$years,  
  ylab = what,  
  xlab = "Years",  
  ex = numeric(0),  
  trans = function(x) x,  
  add = FALSE,  
  ci = TRUE,  
  cicol = gray(0.5, alpha = 0.5),  
  addCI = rep(FALSE, length(fit)),  
  drop = 0,  
  unname.basename = "current",  
  xlim = NULL,  
  ...  
)  
  
## S3 method for class 'samforecast'  
plotit(  
  fit,  
  what,  
  x = fit$data$years,  
  ylab = what,  
  xlab = "Years",  
  ex = numeric(0),  
  trans = function(x) x,  
  add = FALSE,  
  ci = TRUE,  
  cicol = gray(0.5, alpha = 0.5),  
  addCI = NA,  
  drop = 0,  
  unname.basename = "current",  
  xlim = NULL,  
  ylim = NULL,  
  ...  
)  
  
## S3 method for class 'hcr'  
plotit(  
  fit,
```

```

what,
x = fit$data$years,
ylab = what,
xlab = "Years",
ex = numeric(0),
trans = function(x) x,
add = FALSE,
ci = TRUE,
cicol = gray(0.5, alpha = 0.5),
addCI = NA,
drop = 0,
unnamed.basename = "current",
xlim = NULL,
...
)

```

Arguments

<code>fit</code>	the fitted object from <code>sam.fit</code> of a set of such fits <code>c(fit1,fit2)</code>
<code>what</code>	quoted name of object to extract
<code>...</code>	extra arguments transferred to <code>plot</code>
<code>x</code>	x-values
<code>ylab</code>	label on y-axis
<code>xlab</code>	label on x-axis
<code>ex</code>	extra y's to make room for
<code>trans</code>	function to transform values by
<code>add</code>	logical, plotting is to be added on existing plot
<code>ci</code>	logical, confidence intervals should be plotted
<code>cicol</code>	color to plot the confidence polygon
<code>addCI</code>	A logical vector indicating if confidence intervals should be plotted for the added fits.
<code>drop</code>	number of years to be left unplotted at the end.
<code>unnamed.basename</code>	the name to assign an unnamed basefit
<code>xlim</code>	xlim for the plot
<code>ylim</code>	ylim for the plot
<code>ylimAdd</code>	values to add when calculating ylim for the plot

Details

The basic plotting used by many of the plotting functions (e.g. `ssbplot`, `fbarplot` ...)

predstdplot *Prediction-standard deviation plot*

Description

Prediction-standard deviation plot

Usage

```
predstdplot(  
  fit,  
  fleet,  
  age = NULL,  
  type = "log",  
  ylim = NULL,  
  ylab = "Standard deviation",  
  xlab = "Prediction",  
  main = "Pred-std relation",  
  ...  
)
```

Arguments

fit	A sam fit object returned from sam.fit.
fleet	Fleet number to plot relation between prediction and standard deviation.
age	Relation at age. Only used in cases with more than one relation within the same fleet.
type	Either 'log' or 'natural': relation for observations on a log or natural scale.
ylim	Optional, sent to plot
ylab	Optional, sent to plot
xlab	Optional, sent to plot
main	Optional, sent to plot
...	Sent to plot

Details

Plot the relation between observation prediction and standard deviation.

print.hcr *Print hcr object*

Description

Print hcr object

Usage

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

Arguments

x a sam hcr object as returned by [hcr](#)
... extra arguments

Details

prints the HCR forecast

print.sam *Print sam object*

Description

Print sam object

Usage

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

Arguments

x the fitted object as returned from the [sam.fit](#) function
... extra arguments

Details

prints the log-likelihood and the main convergence criteria

print.samcoef	<i>Print samcoef object</i>
---------------	-----------------------------

Description

Print samcoef object

Usage

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

Arguments

x	samcoef object as returned from the coef.sam function
...	extra arguments

print.samforecast	<i>Print samforecast object</i>
-------------------	---------------------------------

Description

Print samforecast object

Usage

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

Arguments

x	an object as returned from the forecast function
...	extra arguments

Details

...

print.samres	<i>Print samres object</i>
--------------	----------------------------

Description

Print samres object

Usage

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

Arguments

x	a sam residual object as returned from either residuals.sam or progres
...	extra arguments

Details

prints the residuals as a data.frame

print.samset	<i>Print samset object</i>
--------------	----------------------------

Description

Print samset object

Usage

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

Arguments

x	a list of sam models
...	extra arguments

Details

...

```
print.samypr          Print samypr object
```

Description

Print samypr object

Usage

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

Arguments

```
x          an object as returned from the ypr function
...        extra arguments
```

Details

...

```
print.sam_referencepoints
          Print referencepoint object
```

Description

Print referencepoint object

Usage

```
## S3 method for class 'sam_referencepoints'
print(x, tables = c("F", "Biomass", "Yield"), digits = 4, format = "f", ...)
```

Arguments

```
x          a sam referencepoint object as returned by referencepoints
tables     tables to print
digits     number of digits to print
format     printing format for numbers
...        extra arguments
```

Details

prints the F reference point table

procrs	<i>Compute process residuals (single joint sample)</i>
--------	--

Description

Compute process residuals (single joint sample)

Usage

```
procrs(fit, map = fit$obj$env$map, ...)
```

Arguments

fit	the fitted object as returned from the <code>sam.fit</code> function
map	map from original fit
...	extra arguments (not currently used)

Details

Single joint sample residuals of $\log(F)$ and $\log(N)$

Value

an object of class `samres`

qtable	<i>table of survey catchabilities</i>
--------	---------------------------------------

Description

table of survey catchabilities

Usage

```
qtable(fit, ...)
```

Arguments

fit	...
...	extra arguments not currently used

Details

...

qtable.sam	<i>table of survey catchabilities</i>
------------	---------------------------------------

Description

table of survey catchabilities

Usage

```
## S3 method for class 'sam'
qtable(fit, ...)
```

Arguments

fit	A sam fit as returned from the sam.fit function
...	extra arguments not currently used

qtableplot	<i>plot survey catchabilities</i>
------------	-----------------------------------

Description

plot survey catchabilities

plot survey catchabilities

Usage

```
qtableplot(qt, exp = FALSE)
```

```
## S3 method for class 'samqtable'
qtableplot(qt, exp = FALSE)
```

Arguments

qt	An object of class 'samqtable' as returned from qtable
exp	if true return on natural scale rather than log

read.data.files	<i>Read all standard data SAM files and return a list as created by 'setup.sam.data'</i>
-----------------	--

Description

Read all standard data SAM files and return a list as created by 'setup.sam.data'

Usage

```
read.data.files(dir = ".")
```

Arguments

dir	Directory to read from
-----	------------------------

Details

Read all standard SAM data files

Value

list (as created by 'setup.sam.data')

read.ices	<i>Function to read ICES/CEFAS data files and validate if input makes sense</i>
-----------	---

Description

Function to read ICES/CEFAS data files and validate if input makes sense

Usage

```
read.ices(filen)
```

Arguments

filen	The filename
-------	--------------

Details

First two lines are ignored and can be used for comments. Can read formats 1 full, 2 row, 3 scalar, and 5 column

Tests: Formatcode is valid, years and ages are pos. integers minimum <= maximum for years and ages number of rows and coulms match year and age ranges data contains only numbers.

Returns: A validated data matrix.

read.surveys *Function to read ices survey format*

Description

Function to read ices survey format

Usage

read.surveys(filename)

Arguments

filename the file

Details

...

read.table.nowarn *Function to suppress incomplete final line warning*

Description

Function to suppress incomplete final line warning

Usage

read.table.nowarn(...)

Arguments

... arguments

Details

...

recplot	<i>SAM Recruits plot</i>
---------	--------------------------

Description

SAM Recruits plot

Usage

```
recplot(fit, lagR = FALSE, ...)

## S3 method for class 'sam'
recplot(fit, lagR = FALSE, ...)

## S3 method for class 'samset'
recplot(fit, lagR = FALSE, ...)

## S3 method for class 'samforecast'
recplot(fit, lagR = FALSE, ...)

## S3 method for class 'hcr'
recplot(fit, lagR = FALSE, ...)
```

Arguments

fit	the object returned from sam.fit
lagR	use the age after the youngest as R
...	extra arguments transferred to plot including the following: add logical, plotting is to be added on existing plot ci logical, confidence intervals should be plotted cicol color to plot the confidence polygon

Details

Plot of numbers of recruits (youngest age class)

rectable	<i>Recruit table</i>
----------	----------------------

Description

Recruit table

Usage

```
rectable(fit, lagR = FALSE, ...)
```

```
## Default S3 method:
```

```
rectable(fit, lagR = FALSE, ...)
```

Arguments

fit	...
lagR	use the age after the youngest as R
...	extra arguments not currently used

Details

...

reduce	<i>reduce helper function to reduce data</i>
--------	--

Description

reduce helper function to reduce data

Usage

```
reduce(data, year = NULL, fleet = NULL, age = NULL, conf = NULL)
```

Arguments

data	a data object as returned by the function <code>setup.sam.data</code>
year	a vector of years to be excluded.
fleet	a vector of fleets to be excluded.
age	a vector of ages fleets to be excluded.
conf	an optional corresponding configuration to be modified along with the data change. Modified is returned as attribute "conf"

Details

When more than one vector is supplied they need to be of same length, as only the pairs are excluded

referencepoints	<i>Estimate reference points</i>
-----------------	----------------------------------

Description

Estimate reference points

Usage

```
referencepoints(
  fit,
  nYears,
  Fsequence,
  aveYears,
  selYears,
  SPRpercent,
  catchType,
  MSYreduction,
  newtonSteps = 3,
  optN = 100,
  jacobianHScale = 0.5,
  ...
)

## S3 method for class 'sam'
referencepoints(
  fit,
  nYears = 100,
  Fsequence = seq(0, 4, len = 200),
  aveYears = max(fit$data$years) + (-9:0),
  selYears = max(fit$data$years),
  SPRpercent = c(0.35),
  dYPRpercent = c(0.1),
  B0percent = c(0.2),
  catchType = "catch",
  MSYreduction = c(0.05),
  newtonSteps = 3,
  optN = 20,
  jacobianHScale = 0.5,
  fixRP = c(),
  RecCorrection = "median",
  biasCorrect = FALSE,
  nlminb.control = list(eval.max = 1000, iter.max = 1000),
  ...
)
```


Arguments

<code>fit</code>	an object to calculate reference points for
<code>nYears</code>	Number of years to use in per-recruit calculations
<code>Fsequence</code>	Sequence of F values used to report per-recruit and equilibrium values
<code>aveYears</code>	Vector of year indices used to calculate average natural mortality, weights, etc. Following ICES guidelines, the default is the last 10 years (starting at 0)
<code>selYears</code>	Vector of year indices used to calculate selectivity (starting at 0)
<code>SPRpercent</code>	Vector of x values for $F[x * 100\%]$ reference points. Default is 0.35.
<code>catchType</code>	Catch type used: (total) catch, landings, discard.
<code>MSYreduction</code>	Vector of proportions for MSY ranges. Default is 0.05 giving an MSY range corresponding to no more than a 5% yield reduction.
<code>newtonSteps</code>	Number of additional Newton steps at the end of the reference point optimization.
<code>optN</code>	N used for numerical optimizers to find equilibrium biomass
<code>jacobianHScale</code>	Scale step size in jacobian calculation
<code>...</code>	not used
<code>dYPRpercent</code>	Defunct
<code>B0percent</code>	Defunct
<code>fixRP</code>	Defunct
<code>RecCorrection</code>	Defunct
<code>biasCorrect</code>	Defunct
<code>nlminb.control</code>	Defunct

Value

a `sam_referencepoints` fit

References

Albertsen, C. M. and Trijoulet, V. (2020) Model-based estimates of reference points in an age-based state-space stock assessment model. *Fisheries Research*, 230, 105618. doi: 10.1016/j.fishres.2020.105618

See Also

[forecastMSY](#)

refit *Re-fit a model from stockassessment.org*

Description

Re-fit a model from stockassessment.org

Usage

```
refit(fit, newConf, startingValues, ...)
```

Arguments

fit	a sam fit or the name of a fit from stockassessment.org
newConf	list changes to the configuration
startingValues	list of parameter values to use as starting values
...	Arguments passed to sam.fit

Value

A new sam fit

residplot *Compute and plot residual diagnostic tests for a SAM model fit*

Description

Compute and plot residual diagnostic tests for a SAM model fit

Usage

```
residplot(  
  fit,  
  resid = NULL,  
  bubbles = TRUE,  
  palette = hcl.colors(4, palette = "Blues"),  
  p.breaks = c(0, 0.001, 0.05, 0.1, 1),  
  reverseFleetOrder = FALSE,  
  plot = TRUE  
)
```

Arguments

fit	an object of class 'sam'
resid	residuals (object of class "samres") from 'fit'. Calculated from 'fit' if not provided here.
bubbles	show bubble plots (default: TRUE)
palette	vector of colors used for p-value groupings
p.breaks	breaks for p-value groupings
reverseFleetOrder	reverse order of fleets within year for OSA residuals (default: FALSE)
plot	plot pvalues (default: TRUE)

Details

Student's t-test for bias, Chi-square test for variance 1, Ljung-Box test on lag 1 for auto-correlation in time and age directions, F for equal variances for obs < or > median, and Shapiro-Wilk test for normality.

Value

list of matrices with p-values corresponding to the plots

residuals.sam	<i>Extract residuals from sam object</i>
---------------	--

Description

Extract residuals from sam object

Usage

```
## S3 method for class 'sam'
residuals(object, discrete = FALSE, subset = 1:nobs(object), ...)
```

Arguments

object	sam fitted object as returned from the sam.fit function
discrete	logical if model contain discrete observations
...	extra arguments for TMB's oneStepPredict

Details

one-observation-ahead quantile residuals are calculated

...

retro	<i>retro run</i>
-------	------------------

Description

retro run

Usage

```
retro(fit, year = NULL, ncores = detectCores(), ...)
```

```
## S3 method for class 'sam'
```

```
retro(fit, year = NULL, ncores = detectCores(), ...)
```

Arguments

fit	a fitted model object as returned from <code>sam.fit</code>
year	either 1) a single integer <code>n</code> in which case runs where all fleets are reduced by 1, 2, ..., <code>n</code> are returned, 2) a vector of years in which case runs where years from and later are excluded for all fleets, and 3) a matrix of years where each column is a fleet and each column corresponds to a run where the years and later are excluded.
ncores	the number of cores to attempt to use
...	extra arguments to <code>sam.fit</code>

Details

...

rmaxplot	<i>SAM rmax plot</i>
----------	----------------------

Description

SAM rmax plot

Usage

```
rmaxplot(fit, ...)
```

```
## Default S3 method:
```

```
rmaxplot(fit, ...)
```

```
## S3 method for class 'samforecast'
```

```
rmaxplot(fit, ...)
```

```
## S3 method for class 'hcr'
rmaxplot(fit, ...)
```

Arguments

<code>fit</code>	the object returned from <code>sam.fit</code>
<code>...</code>	extra arguments transferred to plot including the following: <code>add</code> logical, plotting is to be added on existing plot <code>ci</code> logical, confidence intervals should be plotted <code>cicol</code> color to plot the confidence polygon

Details

Plot of life expectancy

rmaxtable

rmax table

Description

rmax table

Usage

```
rmaxtable(fit, ...)
```

```
## Default S3 method:
rmaxtable(fit, ...)
```

Arguments

<code>fit</code>	...
<code>...</code>	extra arguments not currently used

Details

...

rmvnorm *rmvnorm helper function to draw multivariate normal samples*

Description

rmvnorm helper function to draw multivariate normal samples

Usage

```
rmvnorm(n = 1, mu, Sigma)
```

Arguments

n the number of samples.
mu the mean vector.
Sigma a positive-definite symmetric matrix specifying the covariance matrix.

Details

Generates samples via the Cholesky decomposition, which is less platform dependent than eigenvalue decomposition.

Value

If $n = 1$ a vector of the same length as μ , otherwise an n by $\text{length}(\mu)$ matrix with one sample in each row.

runwithout *runwithout helper function*

Description

runwithout helper function

Usage

```
runwithout(fit, year, fleet, ...)  
  
## S3 method for class 'sam'  
runwithout(fit, year = NULL, fleet = NULL, map = fit$obj$env$map, ...)
```

Arguments

fit	a fitted model object as returned from sam.fit
year	a vector of years to be excluded. When both fleet and year are supplied they need to be of same length, as only the pairs are excluded
fleet	a vector of fleets to be excluded. When both fleet and year are supplied they need to be of same length, as only the pairs are excluded
...	extra arguments to sam.fit
map	map to use

Details

...

sam.fit

Fit SAM model

Description

Fit SAM model

Usage

```
sam.fit(
  data,
  conf,
  parameters,
  newtonsteps = 3,
  rm.unidentified = FALSE,
  run = TRUE,
  lower = getLowerBounds(parameters, conf),
  upper = getUpperBounds(parameters, conf),
  sim.condRE = TRUE,
  ignore.parm.uncertainty = FALSE,
  rel.tol = 1e-10,
  penalizeSpline = FALSE,
  fullDerived = FALSE,
  ...
)
```

Arguments

data	data for the sam model as returned from the setup.sam.data function
conf	model configuration which can be set up using the defcon function and then modified either directly in R or by saving it to a text file using the function saveConf , modifying the text file, and then reading the configuration from the textfile using the function loadConf . For more details about the configuration see details.

parameters	initial values which can be set up using the <code>defpar</code> function and then modified.
newtonsteps	optional extra true newton steps
rm.unidentified	option to eliminate unidentified model parameters based on gradient in initial value (somewhat experimental)
run	if FALSE return AD object without running the optimization
lower	named list with lower bounds for optimization (only met before extra newton steps)
upper	named list with upper bounds for optimization (only met before extra newton steps)
sim.condRE	logical with default TRUE. Simulated observations will be conditional on estimated values of F and N, rather than also simulating F and N forward from their initial values.
ignore.parm.uncertainty	option passed to <code>TMB:::sdreport</code> reported uncertainties will not include fixed effect parameter uncertainties
rel.tol	option passed to <code>stats:::nlminb</code> sets the convergence criteria
penalizeSpline	Add penalization to spline recruitment?
fullDerived	Report all derived values?
...	extra arguments to <code>MakeADFun</code>

Details

The model configuration object `conf` is a list of different objects defining different parts of the model. The different elements of the list are:

\$minAge: A single integer defining the the lowest age class in the assessment.

\$maxAge: A single integer defining the the highest age class in the assessment.

\$maxAgePlusGroup: Is last age group considered a plus group (1 yes, or 0 no).

\$keyLogFsta: A matrix of integers. The number of rows is equal to the number of fleets and the number of columns is equal to the number of age classes. The matrix describes the coupling of the fishing mortality states (the first rows are the catch fleet without effort). '-1' is used for entries where no fishing mortality applies (e.g. age groups in survey fleets, or unobserved age groups). For the valid entries consecutive integers starting at zero must be used, because they are used as indices in the corresponding state vector. If the same number is used for two fleet-age combinations, then the fishing mortality for those are assumed equal (linked to the same state).

\$corFlag: An integer vector to specify the correlation structure of log-scale of fishing mortality increments (0 independent, 1 compound symmetry, or 2 AR(1)). The length of the vector is equal to the number of catch fleets without effort information.

\$keyLogFpar: A matrix of integers. The number of rows is equal to the number of fleets and the number of columns is equal to the number of age classes. The matrix describes the coupling of survey catchability parameters (so only used for survey fleets). '-1' is used for entries where catchability should not be specified (e.g. fleet - age groups combinations where fishing

mortality is specified above, or unobserved fleet - age group combinations). For the valid entries consecutive integers starting at zero must be used, because they are used as indices in the corresponding parameter vector. If the same number is used for two age classes, then the catchability for those age classes are assumed equal (linked to the same parameter).

\$keyQpow: A matrix of integers. The number of rows is equal to the number of fleets and the number of columns is equal to the number of age classes. The matrix describes the coupling of density dependent catchability power parameters. This can only be applied to fleets - age combinations where a catchability is defined. '-1' is used for entries where this cannot be applied (e.g. fleet - age groups combinations where fishing mortality is specified above, or unobserved fleet - age group combinations). '-1' is also used to specify that density dependent catchability power parameters is turned off (the most common setup). For entries where density dependent catchability power parameter is to be estimates entries consecutive integers starting at zero must be used. If the same number is used for two age classes, then the density dependent catchability power parameter for those age classes are assumed equal (linked to the same parameter).

\$keyVarF: A matrix of integers. The number of rows is equal to the number of fleets and the number of columns is equal to the number of age classes. The matrix describes the coupling of variance parameters for the different states in the log-scale fishing mortality random walk process. '-1' should be used for entries where no fishing mortality state is defined in keyLogFsta above. For the valid entries consecutive integers starting at zero must be used, because they are used as indices in the corresponding parameter vector. If the same number is used for two age classes, then the catchability for those age classes are assumed equal (linked to the same parameter). ((a curiosity of this setup is that it is possible to set different variance parameter indices for F-states that are coupled in keyLogFsta. This is ignored and the index corresponding to the lowest F-state number is used)).

\$keyVarLogN: A vector of integers. The length of the vector is equal to the number of age classes. The vector describes the coupling of variance parameters for the log(N)-process. Consecutive integers starting at zero must be used, because they are used as indices in the corresponding parameter vector. If the same number is used for two age classes, then the catchability for those age classes are assumed equal. A typical setup is to use a unique index for the first age group, because that corresponds to the variance in the (stock-)recruitment, which is often not similar to the variance in the survival process from year to year.

\$keyVarObs: A matrix of integers. The number of rows is equal to the number of fleets and the number of columns is equal to the number of age classes. The matrix describes the coupling of observation variance parameters. '-1' should be used for entries where no observations are available. For the valid entries consecutive integers starting at zero must be used, because they are used as indices in the corresponding parameter vector. If the same number is used for two age classes, then the observation variance for those age classes are assumed equal (linked to the same parameter).

\$obsCorStruct: A factor specifying the covariance structure used across ages for each fleet. The length of the factor is equal to the number of fleets. The possible options are: ("ID" independent, "AR" AR(1), or "US" for unstructured).

\$keyCorObs: A matrix of integers. The number of rows is equal to the number of fleets and the number of columns is equal to the number of age classes `_minus_ _one_`. The matrix describes the coupling AR correlations between age classes, and hence is only meaningful for fleets where the "AR" observation correlation structure is chosen. '-1' should be used for entries where no observations are available. Notice that the matrix has one column less than

the number of age classes, which is because the correlation between age classes is described. Consecutive integers starting at zero must be used. If the same number is used for a given fleet it means that a normal AR(1) structure is used. If different numbers are used for a fleet it means that the correlation parameter changes where the numbers differ. If the "AR" structure is specified above, then the corresponding row in this matrix must have valid non-negative entries.

\$stockRecruitmentModelCode: A single integer to specify the stock recruitment connection to use:

Code	Model
0	plain random walk on log recruitment
1	Ricker
2	Beverton-Holt
3	piece-wise constant
61	segmented regression (hockey stick)
62	AR(1) on log-recruitment
63	bent hyperbola (smooth hockey stick)
64	power function with degree < 1
65	power function with degree > 1
66	Shepherd
67	Deriso/Hassel
68	Saila-Lorda
69	sigmoidal Beverton-Holt
90	CMP spline (Non-increasing spline on log(R/S))
91	Integrated spline on log(R/S)
92	Spline on log(R/S)

See Albertsen & Trijoulet (2020) for details.

\$constRecBreaks: A vector of years to determine piece-wise constant recruitment periods for recruitment model 3. A vector of knot placements on log-SSB for spline recruitment models (90, 91, 92).

\$hockeyStickCurve Determines the smoothness of recruitment model 63. The smoothness is estimated if set to NA.

\$noScaledYears: A single integer specifying the number of years where catch scaling is to be estimated (most often 0, as this is a somewhat exotic option).

\$keyScaledYears: A vector of the years where catch scaling is applied (length should match noScaledYears) (most often empty, as this is a somewhat exotic option).

\$keyParScaledYA: A matrix of integers specifying the couplings of scale parameters (nrow = noScaledYears, ncol = no ages) (most often empty, as this is a somewhat exotic option).

\$fbarRange: An integer vector of length 2 specifying lowest and highest age included in Fbar (average fishing mortality summary).

\$keyBiomassTreat: A vector of integers with length equal to the number of fleets. '-1' should be used for entries where the corresponding fleet is not a mass index. A the corresponding fleet is a mass index, then three options are available (0 SSB index, 1 catch index, and 2 FSB index).

\$obsLikelihoodFlag: A factor specifying the type of likelihood to use for each fleet. The length of the factor is equal to the number of fleets. The possible options are: ("LN" for log-normal and "ALN" Additive logistic normal).

\$fixVarToWeight: A single integer. If weight attribute is supplied for observations this option defines how it is treated (0 as relative weight, 1 as a fixed variance = weight).

Value

an object of class sam

References

Albertsen, C. M. and Trijoulet, V. (2020) Model-based estimates of reference points in an age-based state-space stock assessment model. Fisheries Research, 230, 105618. doi:[10.1016/j.fishres.2020.105618](https://doi.org/10.1016/j.fishres.2020.105618)

Examples

```
data(nscodData)
data(nscodConf)
data(nscodParameters)
fit <- sam.fit(nscodData, nscodConf, nscodParameters, silent = TRUE)
```

saveConf	<i>Saves a model configuration list to a file</i>
----------	---

Description

Saves a model configuration list to a file

Usage

```
saveConf(x, file = "", overwrite = FALSE)
```

Arguments

x	sam configuration list as returned from defcon or loadConf
file	the file to save the configuration to
overwrite	logical if an existing file should be overwritten (FALSE by default)

Details

function useful for saving a model configuration. A saved configuration can be read back in via the loadConf function

sdplot	<i>Plots the sd of the log observations as estimated in SAM in increasing order</i>
--------	---

Description

Plots the sd of the log observations as estimated in SAM in increasing order

Usage

```
sdplot(fit, barcol = NULL, marg = NULL, ylim = NULL, ...)
```

```
## S3 method for class 'sam'
```

```
sdplot(fit, barcol = NULL, marg = NULL, ylim = NULL, show.rel.w = FALSE, ...)
```

Arguments

fit	the object returned from sam.fit
barcol	color for each fleet and age
marg	margin for plot (mar in par())
ylim	bounds for y-axis
...	extra arguments to plot
show.rel.w	plots the relative weight of each observation rather than the sd, estimated as $(1/sd^2)/\max(1/sd^2)$

setS	<i>small helper function</i>
------	------------------------------

Description

small helper function

Usage

```
setS(x)
```

Arguments

x	vector of indices
---	-------------------

Details

internal

setSeq	<i>small helper function</i>
--------	------------------------------

Description

small helper function

Usage

```
setSeq(min, max)
```

Arguments

min	from
max	to

Details

internal

setup.sam.data	<i>Combine the data sources to SAM readable object</i>
----------------	--

Description

Combine the data sources to SAM readable object

Usage

```
setup.sam.data(  
  fleets = NULL,  
  surveys = NULL,  
  residual.fleets = NULL,  
  prop.mature = NULL,  
  stock.mean.weight = NULL,  
  catch.mean.weight = NULL,  
  dis.mean.weight = NULL,  
  land.mean.weight = NULL,  
  natural.mortality = NULL,  
  prop.f = NULL,  
  prop.m = NULL,  
  land.frac = NULL,  
  recapture = NULL,  
  sum.residual.fleets = NULL,  
  keep.all.ages = FALSE  
)
```

Arguments

fleets	comm fleets vith effort (currently unimplemented)
surveys	surveys
residual.fleets	fleet, or list of fleets without effort information
prop.mature	pm
stock.mean.weight	sw
catch.mean.weight	cw
dis.mean.weight	dw
land.mean.weight	lw
natural.mortality	nm
prop.f	...
prop.m	...
land.frac	...
recapture	...
sum.residual.fleets	...
keep.all.ages	...

Details

...

simstudy

*Simulate data from fitted model and re-estimate from each run***Description**

Simulate data from fitted model and re-estimate from each run

Usage

simstudy(fit, nsim, ncores = detectCores())

Arguments

fit	a fitted model returned from sam.fit
nsim	number of simulations
ncores	number of cores to be used

simulate.sam	<i>Simulate from a sam object</i>
--------------	-----------------------------------

Description

Simulate from a sam object

Usage

```
## S3 method for class 'sam'  
simulate(  
  object,  
  nsim = 1,  
  seed = NULL,  
  full.data = TRUE,  
  keep.process = FALSE,  
  ...  
)
```

Arguments

object	sam fitted object as returned from the sam.fit function
nsim	number of response lists to simulate. Defaults to 1.
seed	random number seed
full.data	logical, should each inner list contain a full list of data. Defaults to TRUE
keep.process	Keep logN and logF processes when full.data = TRUE?
...	extra arguments

Details

simulates data sets from the model fitted and conditioned on the random effects estimated

Value

returns a list of lists. The outer list has length nsim. Each inner list contains simulated values of logF, logN, and obs with dimensions equal to those parameters.

sprplot	<i>SAM SPR plot</i>
---------	---------------------

Description

SAM SPR plot

Usage

```
sprplot(fit, ...)

## Default S3 method:
sprplot(fit, ...)

## S3 method for class 'samforecast'
sprplot(fit, ...)

## S3 method for class 'hcr'
sprplot(fit, ...)
```

Arguments

fit	the object returned from sam.fit
...	extra arguments transferred to plot including the following: add logical, plotting is to be added on existing plot ci logical, confidence intervals should be plotted cicol color to plot the confidence polygon

Details

Plot of deterministic equilibrium spawners per recruit assuming biological parameters and selectivity for that year remains unchanged in the future.

sprtable	<i>SPR table</i>
----------	------------------

Description

SPR table

Usage

```
sprtable(fit, ...)

## Default S3 method:
sprtable(fit, ...)
```


Arguments

fit	...
...	extra arguments not currently used

Details

...

srplot	<i>Plots the stock recruitment</i>
--------	------------------------------------

Description

Plots the stock recruitment

Usage

```
srplot(fit, ...)

## S3 method for class 'sam'
srplot(
  fit,
  textcol = "red",
  years = TRUE,
  linetype = "l",
  linecol = "black",
  polycol = do.call("rgb", c(as.list(col2rgb("black"))[, 1]), list(alpha = 0.1)),
  polyborder = do.call("rgb", c(as.list(col2rgb("black"))[, 1]), list(alpha = 0.3)),
  polylty = 3,
  polylwd = 1,
  xlim,
  ylim,
  add = FALSE,
  CIlevel = 0.95,
  addCurve = TRUE,
  ...
)
```

Arguments

fit	the object returned from sam.fit
...	extra arguments to plot
textcol	color of years on plot
years	the plotting symbols are the years
linetype	type for the plot (default line)

linecol	color of lines between points
polycol	Inner color of error ellipses
polyborder	Border color of error ellipses
polylty	Border line type of error ellipses
polylwd	Border line width of error ellipses
xlim	bounds for x-axis
ylim	bounds for y-axis
add	false if a new plot should be created
CIlevel	Confidence level for error ellipses on stock-recruitment pairs
addCurve	Call addRecruitmentCurve?

ssbplot

SAM SSB plot

Description

SAM SSB plot

Usage

```
ssbplot(fit, ...)

## Default S3 method:
ssbplot(fit, ...)

## S3 method for class 'samforecast'
ssbplot(fit, ...)

## S3 method for class 'hcr'
ssbplot(fit, ...)
```

Arguments

fit	the object returned from sam.fit
...	extra arguments transferred to plot including the following: add logical, plotting is to be added on existing plot ci logical, confidence intervals should be plotted cicol color to plot the confidence polygon

Details

Plot of spawning stock biomass

ssbtable	<i>SSB table</i>
----------	------------------

Description

SSB table

Usage

```
ssbtable(fit, ...)
```

```
## Default S3 method:
ssbtable(fit, ...)
```

Arguments

fit	...
...	extra arguments not currently used

Details

...

stochasticReferencepoints	<i>Estimate stochastic reference points</i>
---------------------------	---

Description

The function estimates reference points based on stochastic model forecasts. The following reference points are implemented:

F=x F fixed to x, e.g., "F=0.3" (NOT IMPLEMENTED YET)

StatusQuo F in the last year of the assessment (NOT IMPLEMENTED YET)

StatusQuo-y F in the y years before the last in the assessment, e.g., "StatusQuo-1" (NOT IMPLEMENTED YET)

MSY F that maximizes yield

0.xMSY Fs that gives 0.x*100% of MSY, e.g., "0.95MSY"

Max F that maximizes yield per recruit

0.xdYPR F such that the derivative of yield per recruit is 0.x times the derivative at F=0, e.g., "0.1dYPR"

0.xSPR F such that spawners per recruit is 0.x times spawners per recruit at F=0, e.g., "0.35SPR"

0.xB0 F such that biomass is 0.x times the biomass at F=0, e.g., "0.2B0"

Usage

```

stochasticReferencepoints(fit, referencepoints, ...)

## S3 method for class 'sam'
stochasticReferencepoints(
  fit,
  referencepoints,
  method = "Q0.5",
  catchType = "catch",
  nYears = 300,
  Frange = c(0, 2),
  nosim = 1000,
  aveYears = max(fit$data$years) + (-9:0),
  selYears = max(fit$data$years),
  newton.control = list(),
  seed = .timeToSeed(),
  formula = ~ibc(F, 5),
  nosim_ci = 200,
  derivedSummarizer = median,
  ...
)

```

Arguments

<code>fit</code>	a sam fit
<code>referencepoints</code>	a character vector of reference points to estimate (see Details)
<code>...</code>	additional parameters that can be passed on
<code>method</code>	estimation method (See Details)
<code>catchType</code>	catch type: catch, landing, discard
<code>nYears</code>	Number of years to forecast
<code>Frange</code>	Range of F values to consider
<code>nosim</code>	Number of simulations for estimation
<code>aveYears</code>	Years to average over for biological input
<code>selYears</code>	Years to average over for selectivity
<code>newton.control</code>	List of control parameters for optimization
<code>seed</code>	Seed for simulations
<code>formula</code>	Formula to estimate optimization criteria as a function of F
<code>nosim_ci</code>	Number of simulations for bootstrap confidence intervals
<code>derivedSummarizer</code>	Function to summarize derived per-recruit values

Details

Reference points can be estimated using these methods:

Mean Use least squares to estimate mean equilibrium values

Q0.x Use quantile regression to estimate the 0.x quantile of equilibrium values

Median Identical to Q0.5

Mode (NOT IMPLEMENTED YET)

To estimate median equilibrium yield, as required by ICES, the method "Q0.5" should be used.

Value

sam reference point object

reference point object

Examples

```
## Not run:  
  stochasticReferencepoints(fit, c("MSY", "0.95MSY", "Max", "0.35SPR", "0.1dYPR"))  
  
## End(Not run)
```

stockassessment-deprecated

Deprecated and defunct functions

Description

Deprecated and defunct functions

referencepoints

For referencepoints, use [deterministicReferencepoints](#).

summary.sam	<i>Summary of sam object</i>
-------------	------------------------------

Description

Summary of sam object

Usage

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

Arguments

object	sam fitted object as returned from the <code>sam.fit</code> function
...	extra arguments

Details

summary table containing recruits, SSB, and Fbar

tableit	<i>Table helper</i>
---------	---------------------

Description

Table helper

Usage

```
tableit(fit, what, x = fit$data$years, trans = function(x) x, ...)
```

```
## S3 method for class 'sam'
tableit(fit, what, x = fit$data$years, trans = function(x) x, ...)
```

```
## S3 method for class 'samforecast'
tableit(fit, what, x = fit$data$years, trans = function(x) x, ...)
```

Arguments

fit	returned object from <code>sam.fit</code>
what	quoted name of what to extract
x	rownames of table
trans	function to be applied
...	extra arguments not currently used

Details

...

tsbplot	<i>SAM TSB plot</i>
---------	---------------------

Description

SAM TSB plot

Usage

tsbplot(fit, ...)

Default S3 method:

tsbplot(fit, ...)

Arguments

fit	the object returned from sam.fit
...	extra arguments transferred to plot including the following: add logical, plotting is to be added on existing plot ci logical, confidence intervals should be plotted cicol color to plot the confidence polygon

Details

Plot of total stock biomass

tsbtable	<i>TSB table</i>
----------	------------------

Description

TSB table

Usage

tsbtable(fit, ...)

Default S3 method:

tsbtable(fit, ...)

Arguments

fit ...
 ... extra arguments not currently used

Details

...

write.data.files *Write all data files from a list as created by 'setup.sam.data'*

Description

Write all data files from a list as created by 'setup.sam.data'

Usage

```
write.data.files(dat, dir = ".", writeToOne = TRUE, ...)
```

Arguments

dat A list as created by 'setup.sam.data'
 dir Directory where the files are written
 writeToOne Write multi fleet data to one file if data is equal for all fleets

Details

Write all data files from a list as created by 'setup.sam.data'

write.ices *Write ICES/CEFAS data file from matrix*

Description

Write ICES/CEFAS data file from matrix

Usage

```
write.ices(x, fileout, writeToOne = TRUE, ...)
```

Arguments

x a matrix where rownames are taken as years and colnames are taken as ages
 fileout file name or connection
 writeToOne Write multi fleet data to one file if data is equal for all fleets
 ... Arguments to be passed to write

Details

Takes the data and writes them in the ICES/CEFAS format. It is assumed that rows represent consecutive years and cols consecutive ages

write.surveys	<i>Write surveys in ICES/CEFAS data file from a model object</i>
---------------	--

Description

Write surveys in ICES/CEFAS data file from a model object

Usage

```
write.surveys(fit, fileout, ...)
```

Arguments

fit	A fitted object as returned from sam.fit
fileout	file name or connection
...	Arguments to be passed to write

Details

Takes the survey data from the fitted object and writes them in the ICES/CEFAS format.

yearslostplot	<i>SAM years lost to fishing plot</i>
---------------	---------------------------------------

Description

SAM years lost to fishing plot

Usage

```
yearslostplot(fit, cause, ...)

## Default S3 method:
yearslostplot(fit, cause = c("Fishing", "Other", "LifeExpectancy"), ...)

## S3 method for class 'samforecast'
yearslostplot(fit, cause = c("Fishing", "Other", "LifeExpectancy"), ...)

## S3 method for class 'hcr'
yearslostplot(fit, cause = c("Fishing", "Other", "LifeExpectancy"), ...)
```

Arguments

fit	the object returned from sam.fit
cause	Fisning, Other, or LifeExpectancy
...	extra arguments transferred to plot including the following: add logical, plotting is to be added on existing plot ci logical, confidence intervals should be plotted cicol color to plot the confidence polygon

Details

Plot of years lost to fishing

yearslosttable	<i>Years Lost table</i>
----------------	-------------------------

Description

Years Lost table

Usage

```
yearslosttable(fit, cause, ...)

## Default S3 method:
yearslosttable(fit, cause = c("Fishing", "Other", "LifeExpectancy"), ...)
```

Arguments

fit	...
cause	Fisning, Other, or LifeExpectancy
...	extra arguments not currently used

Details

...

ypr *Yield per recruit calculation*

Description

Yield per recruit calculation

Usage

```
ypr(
  fit,
  Flimit = 2,
  Fdelta = 0.01,
  aveYears = min(15, length(fit$data$years)),
  sprProp = 0.35,
  ...
)
```

```
## S3 method for class 'sam'
ypr(
  fit,
  Flimit = 2,
  Fdelta = 0.01,
  aveYears = min(15, length(fit$data$years)),
  sprProp = 0.35,
  ...
)
```

Arguments

fit	the object returned from sam.fit
Flimit	Upper limit for Fbar
Fdelta	increments on the Fbar axis
aveYears	Number of years back to use when calculating averages (selection, weights, ...)
sprProp	Proportion of SPR at F=0, for example 0.35 if F0.35SPR
...	extra arguments not currently used

yprplot	<i>SAM YPR plot</i>
---------	---------------------

Description

SAM YPR plot

Usage

```
yprplot(fit, ...)

## Default S3 method:
yprplot(fit, ...)

## S3 method for class 'samforecast'
yprplot(fit, ...)

## S3 method for class 'hcr'
yprplot(fit, ...)
```

Arguments

fit	the object returned from sam.fit
...	extra arguments transferred to plot including the following: add logical, plotting is to be added on existing plot ci logical, confidence intervals should be plotted cicol color to plot the confidence polygon

Details

Plot of deterministic equilibrium yield per recruit assuming biological parameters and selectivity for that year remains unchanged in the future.

yprtable	<i>YPR table</i>
----------	------------------

Description

YPR table

Usage

```
yprtable(fit, ...)

## Default S3 method:
yprtable(fit, ...)
```

Arguments

fit ...
... extra arguments not currently used

Details

...

Index

* datasets

- nscodConf, 48
 - nscodData, 49
 - nscodParameters, 50
- addforecast, 4
- addRecruitmentCurve, 5
- b0plot, 7
- b0table, 8
- bc, 8
- c. sam, 9
- catchbyfleetplot, 9
- catchbyfleetable, 10
- catchplot, 10
- catchtable, 11
- caytable, 12
- clean.void.catches, 12
- coef.sam, 13, 63
- componentplot, 13
- corplot, 14
- corplotcommon, 15
- dataplot, 15
- defcon, 12, 16, 79
- defpar, 16, 80
- deterministicReferencepoints, 17, 93
- deterministicReferencepoints.sam, 18
- empirobscorrplot, 19
- equilibriumbiomassplot, 20
- equilibriumbiomassstable, 20
- faytable, 21
- fbarplot, 22
- fbartable, 23
- fitfromweb, 24
- fitplot, 24
- forecast, 25, 28, 45, 46
- forecastMSY, 27, 43, 44, 73
- fselectivityplot, 28
- generationlengthplot, 29
- generationlengthtable, 29
- getAllDerivedValues, 30
- getFleet, 30
- getLowerBounds, 31
- getResidualFleets, 31
- getUpperBounds, 32
- grad, 32
- hcr, 33, 36, 43, 44, 54, 62
- ibc, 34
- icesAdviceRule, 35
- iibc, 36
- is.whole.positive.number, 36
- jacobian, 37
- jit, 38
- leaveout, 39
- lifeexpectancyplot, 39
- lifeexpectancytable, 40
- loadConf, 41, 79
- logLik.sam, 41
- modelDescription, 42
- modelforecast, 33, 42
- modeltable, 46
- modelVersionInfo, 47
- mohn, 47
- newton, 19
- nobs.sam, 48
- nscodConf, 48
- nscodData, 49
- nscodParameters, 50
- ntable, 51
- obscorrplot, 51

obsconv, 52

parplot, 52

partable, 53

plot.hcr, 53

plot.sam, 54

plot.samforecast, 54

plot.samres, 55

plot.samset, 56

plot.samypr, 56

plotby, 57

plotit, 58

predstdplot, 61

print.hcr, 62

print.sam, 62

print.sam_referencepoints, 65

print.samcoef, 63

print.samforecast, 63

print.samres, 64

print.samset, 64

print.samypr, 65

procres, 55, 64, 66

qtable, 66

qtable.sam, 67

qtableplot, 67

read.data.files, 68

read.ices, 68

read.surveys, 69

read.table.nowarn, 69

recplot, 54, 70

rectable, 70

reduce, 71

referencepoints, 28, 65, 72

refit, 74

residplot, 74

residuals.sam, 55, 64, 75

retro, 76

rmaxplot, 76

rmaxtable, 77

rmvnorm, 78

runwithout, 78

sam.fit, 9, 13, 39, 41, 48, 54–56, 62, 66, 75, 76, 79, 87, 94

saveConf, 79, 83

sdplot, 84

sdreport, 19

setS, 84

setSeq, 85

setup.sam.data, 85

simstudy, 86

simulate.sam, 87

sprplot, 88

sprtable, 88

srplot, 89

ssbplot, 54, 90

ssbtable, 91

stochasticReferencepoints, 91

stockassessment-deprecated, 93

summary.sam, 94

tableit, 94

tsbplot, 95

tsbtable, 95

write.data.files, 96

write.ices, 96

write.surveys, 97

yearslostplot, 97

yearslosttable, 98

ypr, 99

yprplot, 100

yprtable, 100