Package: JABBA3 (via r-universe)

November 5, 2026

Type Package

Title Bayesian State-Space Surplus Production Models in 'RTMB'

Version 3.0.0.9000

Description The next generation of 'JABBA' (Just Another Bayesian Biomass Assessment), now with hyperdrive. Fits Bayesian state-space surplus production models in 'RTMB' in well under a second, so you can spend less time waiting for chains to converge and more time arguing about priors. Full posteriors are available through 'tmbstan' (NUTS) for those who still like their MCMC. Inputs and outputs follow 'FLR' conventions for use with 'ggplotFL', 'FLRef' and 'mse', and helpers convert classic 'JABBA' inputs so old scripts survive the jump to hyperspace.

License EUPL (>= 1.2)

Encoding UTF-8

Depends R (>= 4.1)

Imports RTMB, Matrix, ggplot2, methods, stats, utils, grDevices, graphics

Suggests tmbstan, rstan, patchwork, MASS, FLCore, ggplotFL, FLRef, testthat (>= 3.0.0)

URL https://github.com/Henning-Winker/JABBA3

BugReports https://github.com/Henning-Winker/JABBA3/issues

LazyData true

RoxygenNote 7.3.1

Repository https://henning-winker.r-universe.dev

Date/Publication 2026-11-05 11:12:05 UTC

RemoteUrl https://github.com/Henning-Winker/JABBA3

RemoteRef HEAD

RemoteSha dbc9d7057b720fe4bf6ff496b91fc64e39f54b87

Contents

flr2jbdat . . . 4
iccat . . . 4
jabba3 . . . 5
jabba3_info . . . 6
jabbaplot3 . . . 7
jara_examples . . . 8
jb3_align . . . 9
jb3_bprior . . . 9
jb3_catchfill . . . 10
jb3_contraction . . . 11
jb3_convergence . . . 13
jb3_data . . . 13
jb3_data_jabba . . . 15
jb3_diags . . . 16
jb3_draws . . . 17
jb3_ensemble . . . 17
jb3_fill_table . . . 18
jb3_flqs . . . 18
jb3_FLStockR . . . 19
jb3_fmsy2r . . . 20
jb3_fwd . . . 21
jb3_fwdtable . . . 22
jb3_hcr . . . 23
jb3_hindcast . . . 24
jb3_jointindex . . . 25
jb3_kobematrix . . . 26
jb3_kobematrix_gfcm . . . 27
jb3_kobematrix_iotc . . . 29
jb3_leslie . . . 30
jb3_model . . . 32
jb3_noise . . . 35
jb3_nuts_diags . . . 36
jb3_nuts_ess . . . 36
jb3_osa . . . 37
jb3_partable . . . 38
jb3_priors . . . 38
jb3_project . . . 40
jb3_projtable . . . 42
jb3_qtable . . . 43
jb3_retro . . . 43
jb3_rho_gt . . . 44
jb3_runs . . . 45
jb3_sel5 . . . 45
jb3_selex . . . 46
jb3_splitindex . . . 47
jb3_trend . . . 48

jb3_tstable . . . 50
jb3_weights . . . 51
jb3_workflow . . . 52
jb3cols . . . 53
jbdat2flr . . . 53
jbinput2jabba3 . . . 54
jbplot_advice . . . 55
jbplot_all . . . 56
jbplot_auxfits . . . 57
jbplot_biplot . . . 57
jbplot_bprior . . . 58
jbplot_catch . . . 58
jbplot_catcherror . . . 59
jbplot_contraction . . . 59
jbplot_crisk . . . 60
jbplot_data . . . 61
jbplot_ebsb_status . . . 61
jbplot_ensemble . . . 62
jbplot_fits . . . 62
jbplot_hcr . . . 63
jbplot_hcxval . . . 64
jbplot_indexcor . . . 64
jbplot_indices . . . 65
jbplot_kobe . . . 65
jbplot_kobematrix . . . 66
jbplot_noise . . . 67
jbplot_nuts . . . 68
jbplot_ppc . . . 69
jbplot_priorpost . . . 70
jbplot_priorpost_trj . . . 70
jbplot_procdev . . . 71
jbplot_projection . . . 72
jbplot_q . . . 73
jbplot_residuals . . . 74
jbplot_retro . . . 74
jbplot_runs . . . 75
jbplot_selex . . . 75
jbplot_spdiags . . . 76
jbplot_spphase . . . 76
jbplot_stdresiduals . . . 77
jbplot_steps . . . 77
jbplot_summary . . . 78
jbplot_trend . . . 78
jbplot_trj . . . 79
scale_x_year . . . 80
spictinp2jabba3 . . . 80
theme_jabba3 . . . 81

Index **82**

flr2jbdat *Convert FLR inputs back to classic JABBA data frames*

Description

The inverse of `jbdat2flr()`: an `FLQuant`/`FLQuants` catch and `FLIndices` to the JABBA data frames `catch`, `cpue` and `se`, and auxiliary `FLQuants` (with SEs in attribute "se", as returned by `jbdat2flr()`) to `auxiliary` and `auxiliary.se`.

Usage

```
flr2jbdat(catch, indices = NULL, auxiliary = NULL)
```

Arguments

`catch`	FLQuant or FLQuants
`indices`	FLIndices (or a single FLIndexBiomass); NULL for catch-only
`auxiliary`	optional FLQuants (or FLQuant) of auxiliary data

Value

list(catch, cpue, se, auxiliary, auxiliary.se) of data frames (NULL where not given)

iccat *Iccat example stocks (from JABBA)*

Description

Catch, CPUE and SE for Atlantic bigeye tuna (bet), white marlin (whm), South Atlantic swordfish (swos) and blue marlin (bum), as in the JABBA package.

Usage

```
iccat
```

Format

list of stocks, each a list with catch, cpue, se (data frames) and info

Details

Bigeye also carries two **hypothetical** auxiliary series for the guide chapter on catch-only assessment and auxiliary data. They are **not real data**: both are simulated from the F trajectory of the JABBA3 bet example fit (Fox, both JPN longline indices; script `data-raw/bet_hypothetical.R`):

`bet$effort_hyp` hypothetical effort, data.frame(year, effort), 1960-2017: E_t = F_t / (q0 x 1.02^(t - 1960)) x lognormal noise (SD 0.15), i.e. with 2% annual effort creep

`bet$ffmsy_hyp` hypothetical F/Fmsy index, data.frame(year, ffmsy, se), 1990-2009 only (a period of high exploitation): F/Fmsy_t x lognormal noise (SD 0.3), se = 0.3

---

jabba3 *Fit a JABBA3 model*

Description

Fits the Pella-Tomlinson state-space surplus production model in RTMB (posterior mode of the fixed effects with the log-depletion states Laplace-integrated) or with NUTS (`tmbstan`, all states sampled). Optionally runs one of the BioProcDyn workflows (Tier 1 lag, Tier 2 tuned dynamic process error, Selex) automatically.

Usage

```
jabba3(
  data,
  priors = jb3_priors(data),
  model = jb3_model(),
  engine = c("rtmb", "nuts"),
  ndraws = 0,
  nuts = list(),
  osa = TRUE,
  workflow = c("none", "tier1", "tier2", "selex", "selex_tier1", "selex_tier2"),
  amat = NULL,
  run = NULL,
  se = TRUE,
  peels = 0,
  index.peels = 0,
  nstart = 4,
  silent = TRUE,
  prior.only = FALSE
)
```

Arguments

`data`	`jb3_data()` object
`priors`	`jb3_priors()` object (default: JABBA defaults with the FishLife sdb prior)
`model`	`jb3_model()` object

engine	"rtmb" (default) or "nuts"
ndraws	number of pseudo-posterior draws from the joint precision at the mode (rtmb; 0 = none). Draws feed the Kobe cloud and projections; intervals come from the delta method otherwise.
nuts	list of tmbstan settings: chains (3), iter (4000), warmup (iter/2), seed, cores, thin, control (adapt_delta 0.95, max_treedepth 12). NUTS is meant for final runs: convergence is checked after sampling and a warning is issued if R-hat > 1.01, bulk/tail ESS < 100 per chain, divergences occur or E-BFMI < 0.3 (table in fit$nuts$diags)
osa	compute one-step-ahead residuals (particle filter at the mode)
workflow	"none" (default, manual), "tier1", "tier2", "selex", "selex_tier1", "selex_tier2" (see jb3_workflow())
amat	age at 50% maturity (production lag for the Tier 1 / Tier 2 workflows)
run	name of the run (scenario)
se	compute standard errors (sdreport). FALSE gives point estimates only, about twice as fast, for simulation testing; the convergence verdict then rests on the gradient (no Hessian check)
peels, index.peels	internal: retrospective (all data) and hindcast (index only) peels
nstart	maximum number of optimisation starts
silent	suppress RTMB tracing

Value

list of class jabba3

Examples

```
data(iccat)
d <- jb3_data(iccat$bet$catch, iccat$bet$cpue, iccat$bet$se, stock = "bet")
fit <- jabba3(d, jb3_priors(d, r = c(0.2, 0.5)), jb3_model(shape = "fox"))
fit
```

---

jabba3_info *Engines and status of this JABBA3 installation*

Description

Reports the package version and which estimation engines are available.

Usage

```
jabba3_info()
```

Value

invisibly, a list with the version and engine availability

jabbaplot3 *JABBA3 summary plot with selectable panels*

Description

A grid of panels (3 x 3 by default, as `plot()` for a SPiCT fit: B, F, Catch / B/Bmsy, F/Fmsy, Kobe / process deviations, surplus production dynamics, surplus production phase on the Kobe colours) built from the individual JABBA3 plots, for a single fit or a named list of fits (runs coloured). Panels are chosen by name, in order, as in `FLRef::ploteq(panels = )`:

"B", "F", "Catch", "B/Bmsy", "F/Fmsy", "B/K", "Process deviation" trajectories with intervals (Bthr/Blim on B/Bmsy)

"kobe" Kobe phase plot (single fit: with the terminal-year uncertainty; several fits: one trajectory per run)

"spdyn" realised surplus production against biomass with the production curve (`jbplot_spdyn()`); several fits: the production curves

"spphase" yield against biomass on the Kobe colours (`jbplot_spphase()`)

"production" production curves with Bmsy/MSY and the catch (yield) path

"fits" index fits

"residuals" OSA residuals (first run)

"procdev" process deviations

"priorpost" prior and posterior of r

`plot(fit)` calls `jabbaplot3(fit)`.

Usage

```
jabbaplot3(
  x,
  type = c("full", "simple"),
  panels = NULL,
  ncol = NULL,
  nested = FALSE,
  obs = TRUE,
  bthr = 0.5,
  blim = 0.3,
  base_size = 9
)

## S3 method for class 'jabba3'
plot(x, type = c("full", "simple"), ...)
```

Arguments

x	a jabba3 fit, a named list of fits or a workflow fit
type	"full" (3 x 3, as SPiCT plot()) or "simple" (2 x 2: B/Bmsy, F/Fmsy, Catch, Kobe, as SPiCT plot2()); ignored when panels is given
panels	panel names or their positions in the full set. Besides the full set: "Catch/MSY", "B/K", "residuals", "production", "fits", "priorpost". With several runs, "Catch" is shown as catch relative to each run's MSY ("Catch/MSY"); a single run shows the catch with its MSY line
ncol	number of columns
nested	nested 90/60% bands on the trajectories (single fit)
obs	single fit: add the observed indices scaled to biomass (I/q) and to B/Bmsy (I/(q Bmsy)) on the "B" and "B/Bmsy" panels, coloured by index (as SPiCT)
bthr, blim	as in jbplot_trj()
base_size	font size
...	passed to jabbaplot3()

Value

a patchwork (or a list of ggplots if patchwork is not installed)

Examples

```
## Not run:
jabbaplot3(fit)
plot(fit)
jabbaplot3(list(Fox = fit, Lag.AR1 = rec), panels = c("B/Bmsy", "F/Fmsy", "kobe", "production"), ncol = 2)

## End(Not run)
```

---

jara_examples *Example data for the data-exploration (JARA) models*

Description

Anonymised example data: generic series names, multiplicative lognormal noise and random rescaling (absolute catches and biomass by one common factor, so that fleets and areas still add up); years and missing values as in the source data.

areas a list with catch: catch history from six fleets/areas (fleet1-fleet6) with missing early years for two of them; idx and idx.se: relative survey indices for four areas (area1-area4) with log-scale SEs; bio and bio.se: swept-area survey biomass for the same four areas (thousands) with log-scale SEs.

regions survey and CPUE indices from three regions: areaA and areaB, surveys of two adjacent areas; C.survey1, C.survey2 and C.cpue1-C.cpue5, surveys and CPUE series of a third region.

Usage

```
jara_examples
```

Format

list

---

jb3_align *Alignment of the input series*

Description

Year ranges of the catch and index inputs as given, and the part used after aligning all series to the catch years.

Usage

```
jb3_align(x)
```

Arguments

x	`jb3_data()` object

Value

data.frame(series, type, first, last, n, used_first, used_last, n_used)

---

jb3_bprior *Status priors (JABBA b.prior)*

Description

Prior on stock status in a given year: B/Bmsy, B/K or F/Fmsy, lognormal with the given median and CV (as JABBA's `b.prior = c(mean, cv, year, type)`). Several priors are combined with `rbind()` and passed to `jb3_priors(status = )`.

Usage

```
jb3_bprior(
  median,
  cv = 0.2,
  year = NULL,
  type = c("bbmsy", "bk", "ffmsy"),
  log.sd = NULL
)
```

Arguments

median	median of the prior (e.g. 0.5 for B/Bmsy)
cv	coefficient of variation (converted to a log-scale SD, sqrt(log(1 + cv^2)))
year	year(s) of the prior; NULL or NA = terminal year of the fit
type	"bbmsy" (default), "bk" or "ffmsy"
log.sd	log-scale SD, instead of cv

Value

data.frame(year, type, median, log.sd)

Examples

```
jb3_bprior(0.7, 0.2, type = "bbmsy")                # terminal-year B/Bmsy prior
rbind(jb3_bprior(0.5, 0.2, 2010), jb3_bprior(1.2, 0.3, 2010, "ffmsy"))
```

---

jb3_catchfill *Reconstruct catch series through missing years*

Description

Census-type state-space smoothing of one or several catch series (e.g. landings by country or fleet), as JARA's census catch reconstruction: a random walk (with optional drift and time blocks) on log catch per series, observed with a near-zero error where the catch is known, so known values are reproduced and missing years are filled with the series-specific (and block-specific) trend; the series are summed to a total with uncertainty.

Usage

```
jb3_catchfill(
  catch,
  se = 0.01,
  drift = !is.null(timeblock),
  timeblock = NULL,
  na.below = NULL,
  years = NULL,
  sdp.pr = c(0.2, 1)
)
```

Arguments

catch	data.frame(year, catch[, more series]); NA for missing values (years can also be absent)
se	observation SD (log scale) for known catches: a single value (JARA: fixed.obsE = 0.001) or a data.frame with the same layout as catch (year, then one column per series; as JARA's catch SE matrix), e.g. to give assumed start values a larger SE than reported catches; missing entries take 0.01

drift	random walk with drift (default TRUE with a time block, otherwise FALSE)
timeblock	optional year(s) after which the drift changes (JARA timeblock)
na.below	optional threshold: catches below it are treated as unobserved
years	optional full year range (e.g. to extend backwards)
sdp.pr	lognormal prior on the process SD

Details

The output goes straight into jb3_data() as catch: the total catch (observed where all series are known, filled otherwise) with its year-specific log-scale SE (se.log: se in fully observed years, the SE of the estimated total where any series was filled), used by the catch error of the model in place of the constant sdc. jb3_fill_table() gives the original series with the NAs replaced by predictions and the total.

Value

data.frame(year, obs (sum of observed series, NA if any missing), catch, se.log, est, lci, uci, filled) of class jb3catchfill, with the census fit in attribute "fit"

Examples

```
## Not run:
cc <- iccat$bet$catch; cc[c(5:8, 30:31), 2] <- NA
cf <- jb3_catchfill(cc)
d <- jb3_data(cf, iccat$bet$cpue, iccat$bet$se)  # filled catch with year-specific catch SE
fit <- jabba3(d, jb3_priors(d, r = c(0.2, 0.5)), jb3_model(shape = "fox"))

## End(Not run)
```

---

jb3_contraction *Prior-posterior contraction and shift*

Description

Compares the posterior of a fit with its prior predictive run (jabba3(..., prior.only = TRUE), the same model, priors and catch with the index and auxiliary likelihoods switched off) for parameters, reference points and status. For each quantity (log scale for positive quantities):

- contraction = 1 - var(posterior)/var(prior predictive): the share of the prior uncertainty removed by the data (0 = the data add nothing, 1 = fully determined by the data);
- shift = (mean(posterior) - mean(prior predictive))/sd(prior predictive): how far the data move the estimate, in prior SDs.

The diagnostic is adapted from the posterior shrinkage versus posterior z-score plot of Betancourt (2018), used in the Bayesian workflow of Schad et al. (2021). There, shrinkage = 1 - var(posterior)/var(prior) and z = (mean(posterior) - true value)/sd(posterior), computed over simulated data sets with known truth, as a check of the model design. A real assessment has no true

value, so here the z-score is replaced by the shift from the prior predictive mean, which measures prior-data conflict rather than calibration, and the reference distribution is the prior predictive run rather than the prior density (`contraction.pdf` is the version closer to Betancourt's shrinkage). The class names and thresholds are JABBA3 choices. A quantity with low contraction is prior-driven; one with a large shift is in conflict with its prior (the data pull it into the tail of the prior); high contraction with a moderate shift means it is informed by the data. A clearly negative contraction (posterior wider than the prior predictive) means the data move the quantity into a region the priors and catch constrain less, and is flagged as "posterior wider". The prior predictive run keeps the catch and the status priors, so contraction measures the information in the indices and auxiliary data beyond the priors and the catch; a catch-only fit therefore has a contraction of zero by construction. For r, K and the initial depletion `contraction.pdf` compares the posterior with the stated prior density instead, which also shows how much the catch history (stock survival) trims the prior.

Usage

```
jb3_contraction(
  fit,
  prior = NULL,
  quantities = c("K", "r", "MSY", "bkinit", "B/Bmsy", "F/Fmsy"),
  years = NULL,
  contraction.min = 0.25,
  shift.max = 2,
  nuts = list(),
  digits = 3
)
```

Arguments

`fit`	a `jabba3` fit, or a named list of fits
`prior`	the prior predictive fit(s) (same layout as `fit`); NULL runs them with NUTS
`quantities`	parameters (columns of the draws: K, r, MSY, Bmsy, Fmsy, sdb, bkinit, m) and status quantities (B/Bmsy, F/Fmsy, B/K)
`years`	years for the status quantities (default the last year)
`contraction.min`	below this a quantity is classed as prior-driven
`shift.max`	above this (absolute) a quantity is classed as in conflict with its prior
`nuts`	NUTS settings for the prior predictive runs
`digits`	rounding (NULL for none)

Value

data.frame(run, quantity, prior.median, post.median, contraction, shift, contraction.pdf, class); the prior predictive fits are attached as attribute "prior"

References

Betancourt, M. (2018). Calibrating model-based inferences and decisions. arXiv:1803.08393. Schad, D.J., Betancourt, M. and Vasishth, S. (2021). Toward a principled Bayesian workflow in cognitive science. Psychological Methods 26, 103-126.

Examples

```
## Not run:
ct <- jb3_contraction(fit)
jbplot_contraction(ct)

## End(Not run)
```

`jb3_convergence` *Convergence checks*

Description

FLicc-style convergence table: optimiser code, maximum gradient, positive-definite Hessian, standard errors and the Newton decrement 0.5 g' H^-1 g, with an overall verdict: "ok" (decrement < 0.001), "warning" (<= 0.1) or "fail" (above 0.1, or Hessian not positive definite).

Usage

```
jb3_convergence(fit)
```

Arguments

`fit`	a `jabba3` fit

Value

data.frame of checks with a plain-language interpretation; verdict in attribute "tier"

`jb3_data` *Assessment data for JABBA3*

Description

Collects catch and abundance indices into the internal data object used by `jabba3()`. Native inputs follow FLR: `catch` as an `FLQuant` (or `FLQuants` by fleet, summed) and `indices` as `FLIndices` of `FLIndexBiomass`, with the log-scale SE of each observation in `index.var` (NA or empty = not given). Plain data frames in the JABBA layout are accepted as well (see also `jb3_data_jabba()`).

Usage

```
jb3_data(
  catch,
  indices = NULL,
  se = NULL,
  stock = "stock",
  catch.se = NULL,
  auxiliary = NULL,
  auxiliary.se = NULL,
  auxiliary.type = "effort",
  catch.gaps = c("stop", "estimate"),
  nseason = NULL,
  index.time = 0
)
```

Arguments

catch	FLQuant, FLQuants, or data.frame(year, catch[, more fleets])
indices	FLIndices, FLIndexBiomass, or data.frame(year, index1, index2, ...)
se	optional data.frame(year, se1, se2, ...) of log-scale standard errors (data.frame input only)
stock	stock name
catch.se	optional year-specific log-scale SE of the catch: a vector along the catch years or data.frame(year, se) (e.g. from jb3_catchfill()); used by the catch error (jb3_model(catch.error = TRUE)) in place of the constant sdc; NA = sdc
auxiliary	optional auxiliary data (JABBA auxiliary): data.frame(year, series1, ...), FLQuants (e.g. from jbdat2flr(), SEs in attribute "se"), or a named list of series, each a data.frame(year, value[, se]) or a list(data = data.frame(year, value), se = , type =). Types (see auxiliary.type): "effort" (fishing effort, modelled as F / (q_E x creep)), "f" (fishing mortality, harvest rate), "ffmsy", "bbmsy", "bk" (status estimates, e.g. from another assessment or a length-based method) and "z" (total mortality, as JABBA). See jb3_model() for the catchability, additional SD and effort creep of the auxiliary series.
auxiliary.se	optional log-scale SEs of the auxiliary data, same layout as auxiliary (data.frame input)
auxiliary.type	type of each auxiliary series (recycled), one of "effort", "f", "ffmsy", "bbmsy", "bk", "z"
catch.gaps	"stop" (default): the model years are the catch years and catch must be given in all of them; "estimate": the model years span the catch and the indices, and catch in years without catch (gaps, years before the first or after the last catch year) is estimated with a random walk on log harvest rate (see jb3_model(), sdf), as in SPiCT
nseason	number of seasons of seasonal inputs (needed with a time column; with a season column the default is the largest season). Whether the model runs on seasons is set by jb3_model(nseason =).

index.time — timing of annual indices within the year (fraction of the year, recycled over indices; 0 = start of the year, as JABBA). Used when the model runs on seasons: the index is matched to the season starting nearest to it.

Details

Alignment. The series need not cover the same years. The model years are the catch years (leading and trailing years without catch are trimmed; gaps inside the catch series are an error, see jb3_catchfill()). Each index keeps its own years and is matched by year; index observations outside the catch years are not used (message). Rows may be in any order; the year column is the column named year (or the first column). Standard errors are matched to the indices by year and by column name (or by position when the names differ but the number of columns agrees); SEs may be given for some years or some indices only. Catch by fleet is summed with missing values counted as no catch, unless all fleets are missing in a year. FLIndices may hold FLIndexBiomass objects of different lengths. The years used are listed by jb3_align().

Seasonal data. Catch and indices may carry a season column (1, ..., nseason) or a time column (decimal year as in SPiCT, e.g. 2001.25 for the start of the second quarter) after year (data frames: year, season, catch or time, catch). Catch in a season covers that season; years without seasonal values may be given annually (rows with season NA). Seasonal index values are observations of the biomass at the start of their season (time: matched to the nearest season start). The annual catch and the annual (geometric mean) indices are kept in catch, I for the annual model; the seasonal values in catch.season (years x seasons) and iobs (long table: year, frac, index, obs, se).

Catch-only. indices = NULL gives a catch-only data set: the model is then informed by the catch, the priors (including status priors, jb3_priors(status =)) and any auxiliary data; jabba3() runs it with NUTS by default.

Value

object of class jb3data: years, catch, index matrix I, SE matrix SE (years x indices, zero columns for catch-only), auxiliary matrices A, A.SE and types A.type (NULL without auxiliary data), stock

Examples

```
data(iccat)
d <- jb3_data(iccat$bet$catch, iccat$bet$cpue, iccat$bet$se, stock = "bet")
d
# catch-only with hypothetical effort (simulated, not real data)
d0 <- jb3_data(iccat$bet$catch, NULL, auxiliary = iccat$bet$effort_hyp[, c("year", "effort")],
               auxiliary.type = "effort", stock = "bet")
```

---

jb3_data_jabba *Convert classic JABBA inputs*

Description

The JABBA data frames (catch, cpue, se as used by build_jabba()) to a jb3data object.

Usage

```
jb3_data_jabba(catch, cpue, se = NULL, assessment = "stock")
```

Arguments

catch	data.frame(year, catch...)
cpue	data.frame(year, index...)
se	optional data.frame(year, se...)
assessment	stock name

---

jb3_diags *Residual diagnostics: JABBA set + SPiCT set*

Description

One table per index, from the OSA residuals by default:

- JABBA set: n, RMSE of the log residuals, SDNR of the standardised residuals, runs test (p) with the share of residuals outside the three-sigma limits.
- SPiCT set (as `plotspict.diagnostic`): bias t-test, Ljung-Box test (lag 4) and Shapiro-Wilk normality test.

Under a correct model OSA residuals are iid N(0, 1); with `type = "ordinary"` the tests run on the classic standardised residuals (smoothed states: shrunk and autocorrelated by construction, so the tests are biased). Residual tests check the error assumptions, they have little power to detect structural misspecification (e.g. an omitted production lag).

Usage

```
jb3_diags(fit, type = c("osa", "ordinary"))
```

Arguments

fit	a `jabba3` fit
type	"osa" (default) or "ordinary"

Value

data.frame with one row per index and a `flag` column (any test with $p < 0.05$); the convergence verdict is in attribute "convergence"

jb3_draws *Pseudo-posterior draws at the mode*

Description

Draws all fixed and random effects from the multivariate normal with the joint precision at the mode (Laplace approximation), and returns derived trajectories and parameters.

Usage

```
jb3_draws(fit, ndraws = 1000, seed = 1)
```

Arguments

fit	a jabba3 fit (rtmb)
ndraws	number of draws
seed	random seed

jb3_ensemble *Weighted ensemble of fits*

Description

Combines several fits (e.g. alternative production functions, priors, lag and AR(1) settings) into one weighted ensemble, as the GFCM/ICCAT JABBA ensembles: each fit contributes a number of draws proportional to its weight (posterior draws for NUTS fits, MVN draws at the mode otherwise), and the joint draws give the ensemble trajectories, reference points and status. The result behaves like a fit for `jbplot_advice()`, `jbplot_trj()`, `jbplot_kobe()`, `jb3_FLStockR()`, `jb3_partable()`-style summaries and projections (`jb3_project()`, `jb3_fwd()`, `jb3_hcr()`), which run each member's own model on its share of the draws.

Usage

```
jb3_ensemble(fits, weights = NULL, ndraws = 3000, seed = 123)
```

Arguments

fits	named list of jabba3 fits (same years)
weights	weights (default equal), e.g. from `jb3_weights()`
ndraws	total number of draws
seed	random seed

Value

object of class `c("jb3ensemble", "jabba3")`

Examples

```
## Not run:
runs <- list(Schaefer = f1, Fox = f2, Pella = f3)
ens <- jb3_ensemble(runs, weights = jb3_weights(runs)$weight)
jbplot_ensemble(ens); jbplot_advice(ens); jbplot_kobe(ens)

## End(Not run)
```

---

jb3_fill_table *Original series with missing values filled*

Description

For a census trend fit (or `jb3_catchfill()` output): the original series with NAs (and values treated as unobserved) replaced by the fitted states, and their total (as JARA's reconstructed catch table).

Usage

```
jb3_fill_table(x, digits = NULL)
```

Arguments

`x`	a census `jb3trend` fit or `jb3catchfill` output
`digits`	rounding

Value

data.frame(year, series..., Total)

---

jb3_flqs *FLR outputs*

Description

Trajectories as `FLQuants` (iterations = draws when the fit has them, otherwise the mode) and reference points as `FLPar`, for use with ggplotFL, FLRef and mse.

Usage

```
jb3_flqs(fit, draws = !is.null(fit$draws))

jb3_flpar(fit)
```

Arguments

fit	a jabba3 fit
draws	use the draws (TRUE when available)

Value

jb3_flqs(): FLQuants stock (B/Bmsy), harvest (F/Fmsy), biomass, F, catch; jb3_flpar(): FLPar

---

jb3_FLStockR *Convert a JABBA3 fit to an FLStockR*

Description

Biomass-dynamic FLStockR (one "age"), as FLRef::jabba2FLStockR(): stock and ssb() are the biomass, catch the (estimated) catch and harvest the harvest rate F = C/B; iterations are the posterior (NUTS) or MVN draws, or the point estimates. The refpts slot holds Fmsy, Bmsy, MSY, Bthr, Blim, B0 (= K); by default everything is relative to Bmsy and Fmsy (Fmsy = Bmsy = 1). Use with FLRef::plotAdvice(), ggplotFL and mse.

Usage

```
jb3_FLStockR(
  fit,
  bthr = 0.5,
  blim = 0.3,
  rel = TRUE,
  draws = !is.null(fit$draws),
  ndraws = 0
)
```

Arguments

fit	a jabba3 fit
bthr, blim	biomass threshold and limit as fractions of Bmsy (defaults 0.5 and 0.3, SPiCT benchmark defaults)
rel	relative units, B/Bmsy and F/Fmsy (default; advice from surplus production models is given in relative terms); FALSE for absolute biomass and harvest rate
draws	use the fit's draws as iterations (default if present); ndraws > 0 generates MVN draws otherwise
ndraws	number of MVN draws to generate when the fit has none (0: point estimates)

Value

FLStockR

Examples

```
## Not run:
stk <- jb3_FLStockR(fit, ndraws = 500)
FLRef::plotAdvice(stk, type = "spm") + ggplot2::facet_wrap(~qname, ncol = 1, scales = "free_y")  # B/Bmsy, F/Fmsy,
FLRef::plotAdvice(jb3_FLStockR(fit, rel = FALSE, ndraws = 500), type = "spm")            # absolute

## End(Not run)
```

`jb3_fmsy2r` *Convert between Fmsy and the surplus production r for a given shape*

Description

In the Pella-Tomlinson form used by JABBA3, $F_{MSY} = r/m$ with m the shape implied by B_{MSY}/K, so the r prior that reproduces a given harvest rate at MSY is $r = mF_{MSY}$ (as `JABBA::r_from_fmsy_bmsyk()`). Works for $B_{MSY}/K < 1/e$ (m < 1). A Leslie r is the per-capita growth at low biomass, which equals $r/(m-1)$ only for the Schaefer model; matching F_{MSY} (from an age-structured equilibrium, or $r_{Leslie}/2$ as the Schaefer equivalent) keeps the r prior consistent across shapes.

Usage

```
jb3_fmsy2r(Fmsy, BmsyK = 0.4, shape = c("pella", "schaefer", "fox"))

jb3_r2fmsy(r, BmsyK = 0.4, shape = c("pella", "schaefer", "fox"))
```

Arguments

`Fmsy`	harvest rate at MSY
`BmsyK`	Bmsy/K (default 0.4); ignored when shape is "schaefer" or "fox"
`shape`	"pella" (from `BmsyK`), "schaefer" or "fox"
`r`	intrinsic rate of the production function

Value

r (`jb3_fmsy2r`) or Fmsy (`jb3_r2fmsy`)

Examples

```
jb3_fmsy2r(0.2, BmsyK = 0.4)       # r = 1.19 x 0.2
jb3_fmsy2r(0.2, shape = "schaefer")
jb3_r2fmsy(0.3, BmsyK = 0.36)
```

`jb3_fwd` *Forward projections with a forecast control (fwdctrl)*

Description

The JABBA3 equivalent of JABBA's `fw_jabba_ctrl()`: stochastic projections of a fit or a weighted ensemble under one or more forecast controls. A control is a `data.frame(year, value, quant)` with one row per projection year, starting in the year after the last assessment year; `quant` is one of

"Catch" catch (tonnes)

"Fmsy" F relative to Fmsy (alias "F/Fmsy")

"F" harvest rate

"Csq", "Fsq" multiplier of the status quo catch or F/Fmsy (mean of the last `nsq` years)

"HCR" the hockey-stick rule (`jb3_hcr()`) applied draw by draw to the start-of-year B/Bmsy, with `value` = Ftgt/Fmsy and the rule settings in `hcr`

The first transition (last assessment year to the next) uses the estimated catch of the last assessment year, so the biomass in the first projection year is a one-step-ahead prediction. Process error follows the fitted model (`procerr`, see `jb3_project()`).

Usage

```
jb3_fwd(
  x,
  fwdctrl,
  ndraws = 1000,
  procerr = c("ar1", "mean", "none"),
  nsq = 3,
  hcr = list(),
  seed = 123
)
```

Arguments

`x`	a `jabba3` fit or a `jb3_ensemble()`
`fwdctrl`	a control data.frame or a (named) list of them, one per scenario
`ndraws`	draws when the fit has none
`procerr`	"ar1" (default), "mean" or "none"
`nsq`	years for the status quo (Csq, Fsq)
`hcr`	list of hockey-stick settings for quant = "HCR": btrigger, bclose, fmin (as fractions of Bmsy and Fmsy)
`seed`	random seed

Value

a `jb3proj` object (as `jb3_project()`), with the controls in `ctrl`

Examples

```
## Not run:
csq <- mean(tail(bet$catch$catch, 3))
fc <- list(Fmsy = data.frame(year = 2018:2021, value = c(csq, csq, 1, 1), quant = c("Catch", "Catch", "Fmsy", "Fmsy"
           Csq  = data.frame(year = 2018:2021, value = 1, quant = "Csq"))
stf <- jb3_fwd(fit, fc)
jbplot_projection(stf); jb3_fwdtable(stf)

## End(Not run)
```

`jb3_fwdtable` *Fishing opportunities table*

Description

The GFCM/ICES-style summary of fishing opportunities by forecast scenario (as `FLRef::fwd2type1()` used in the GFCM JABBA ensemble guides): the catch and F in the evaluation (TAC) year, the biomass at the start of the following year (`osa = TRUE`, one-step-ahead response to the TAC), its change relative to the year before the TAC year, and the probabilities (%) of the biomass being above Blim, Bpa (= Bthr) and Btgt (= Bmsy).

Usage

```
jb3_fwdtable(
  proj,
  eval.yr = NULL,
  osa = TRUE,
  rel = TRUE,
  bthr = 0.5,
  blim = 0.3,
  digits = 3
)
```

Arguments

`proj`	a `jb3proj` from `jb3_fwd()` or `jb3_project()`
`eval.yr`	evaluation (TAC) year; default the TAC year of the forecast control (or the first projection year)
`osa`	evaluate the biomass one step ahead (year after the TAC year)
`rel`	relative units (F/Ftgt, B/Btgt) as in the GFCM tables; FALSE gives catch, F and biomass in absolute units
`bthr, blim`	Bpa (Bthr) and Blim as fractions of Bmsy
`digits`	decimals

Value

data.frame with columns Basis, C y, F/Ftgt y, B/Btgt y+1, B change, P(B>Blim), P(B>Bpa), P(B>Btgt)

`jb3_hcr` *Hockey-stick harvest control rule*

Description

The GFCM/ICES-type hockey-stick rule (JABBA's `jb_hockeystick()`): F/Fmsy = `ftgt` above `btrigger` (B/Bmsy), decreasing linearly to `fmin` at `bclose` (0 by default, as JABBA; set `bclose = blim` to close the fishery below Blim). The rule is evaluated on the biomass in the first implementation (TAC) year, after the intermediate years of a forecast control: first an `ftgt` forecast is run (or `x` is such a forecast), then

- `uncertainty = FALSE`: F_adv from the median B/Bmsy;
- `uncertainty = TRUE` (precautionary, the default): F_adv = mean over all draws of the rule applied to each draw's B/Bmsy.

The returned F_adv (F/Fmsy) can be used in a new control (`quant = "Fmsy"`); `quant = "HCR"` in `jb3_fwd()` applies the rule draw by draw in every projection year instead.

Usage

```
jb3_hcr(
  x,
  ftgt = 1,
  btrigger = 0.5,
  blim = 0.3,
  bclose = 0,
  fmin = 0.001,
  year = NULL,
  uncertainty = TRUE,
  imp.years = 2,
  imp.catch = NULL,
  scenario = 1,
  ...
)
```

Arguments

`x`	a fit or `jb3ensemble` (an Ftgt forecast is run with `imp.years` intermediate years at `imp.catch`, default status quo catch), or a `jb3proj` from `jb3_fwd()`
`ftgt`	target F/Fmsy
`btrigger`	B/Bmsy below which F is reduced (GFCM: Bthr = 0.5 Bmsy)
`blim`	limit B/Bmsy (shown in the plot; default 0.3)

bclose	B/Bmsy at which F reaches fmin
fmin	minimum F/Fmsy
year	evaluation year (default: the TAC year of the forecast)
uncertainty	precautionary F_adv across draws
imp.years, imp.catch	intermediate years and their catch when x is a fit
scenario	scenario of x to use when x is a forecast (default the first)
...	passed to jb3_fwd() (e.g. ndraws, procerr)

Value

object of class jb3hcr: Fadv (F/Fmsy), Fadv.det, Fadv.pa, B/Bmsy draws in the evaluation year, the rule settings and the forecast

---

jb3_hindcast *Hindcast cross-validation*

Description

Refits with the index observations of the last 1..n years removed (catch kept) and predicts each removed observation one step ahead; MASE as in ss3diags (Winker and Kell), with the naive prediction = last observed value.

Usage

```
jb3_hindcast(fit, n = 5, MAE.base.adj = 0.1)
```

Arguments

fit	a jabba3 fit
n	number of peels
MAE.base.adj	minimum naive error for MASE.adj

Value

list of class jb3hindcast: pred (index, year, obs, hat, peel), mase

jb3_jointindex *Joint index from a trend fit*

Description

As JARA's `jointindex()`: for each year the precision-weighted mean of the q-scaled observed indices ($\log I_{it} - \log q_i$, weights 1/total observation variance) with its log-scale SE, and the smoothed state-space trend. Years without observations are NA, as in JARA, or take the state estimate and its SE with `fill = TRUE` (column `source`). The joint index is scaled to a mean of 1 and can be passed to `jb3_data()` as a single index with `log.se`. For a census fit the table holds the total abundance (sum of the area states). Either format goes straight into `jb3_data()`: the data.frame (class `jb3jointindex`) as `indices` (its `log.se` is used as the SE), or the `FLIndexBiomass` with `index.var` = log.se.

Usage

```
jb3_jointindex(
  fit,
  format = c("data.frame", "FLIndexBiomass"),
  name = "joint",
  fill = FALSE
)
```

Arguments

`fit`	a `jb3trend` fit
`format`	"data.frame" or "FLIndexBiomass"
`name`	index name
`fill`	fill years without observations with the state estimate (default FALSE: NA, as JARA)

Value

data.frame(year, trend, index, log.se, lci, uci, source) of class `jb3jointindex`, or an FLIndexBiomass

Examples

```
## Not run:
ji <- jb3_jointindex(tr)
d <- jb3_data(catch, ji)                        # joint index as the single index, log.se as SE
d  <- jb3_data(catch, FLCore::FLIndices(joint = jb3_jointindex(tr, "FLIndexBiomass")))

## End(Not run)
```

`jb3_kobematrix` *Kobe II strategy matrix*

Description

Kobe II strategy matrix (K2SM) in the ICCAT format: the probability, by scenario (rows, e.g. TAC levels) and projection year (columns), that F < Fmsy, that B > Bmsy, and of both (the green Kobe quadrant). Several projections, e.g. from alternative models or a JABBA and an SS3 base case, can be combined with equal or given weights, as in the ICCAT combined matrices.

Usage

```
jb3_kobematrix(
  proj,
  what = c("green", "B", "F", "all"),
  years = NULL,
  percent = FALSE,
  weights = NULL,
  bref = 1,
  fref = 1,
  risk = FALSE,
  rows = c("auto", "label", "value"),
  digits = 2
)
```

Arguments

`proj`	output of `jb3_project()` or `jb3_fwd()`, or a (named) list of them with the same scenarios and years, to be combined
`what`	`"green"` (all conditions jointly: F < Fref and B > Bref), `"B"` (B > Bref), `"F"` (F < Fref), or `"all"` for the single and joint matrices (a, b, c of the ICCAT tables) as a list
`years`	projection years to show (default all)
`percent`	if TRUE, probabilities in percent, rounded to integers (ICCAT style)
`weights`	weights for combining a list of projections (default equal)
`bref`	biomass reference point as a multiple of Bmsy (default 1, Bmsy), optionally named, e.g. `c(Btrigger = 0.5)` or `c(Blim = 0.3)`; NULL to drop the biomass condition
`fref`	fishing mortality reference point as a multiple of Fmsy (default 1, Fmsy), optionally named, e.g. `c(Flim = 1.3)`; NULL to drop the F condition
`risk`	if TRUE, report the probability of not meeting the conditions, e.g. P(B < Blim) or P(F > Fmsy or B < Btrigger)
`rows`	row labels: `"label"` the scenario names in the order of the projection (e.g. Fmsy, Fcur, 0.8Fmsy, from named `scenarios`, `fmsy` or `catch` in `jb3_project()`, or a named `fwdctrl` list), `"value"` the TAC or F/Fmsy value; `"auto"` (default) uses the names when they were given

digits rounding of probabilities when percent = FALSE

Details

The reference points are general: bref and fref set the biomass and fishing mortality thresholds as multiples of Bmsy and Fmsy, optionally named, so the same matrix gives, for example, the ICES conditions B > Btrigger and F < Fmsy (bref = c(Btrigger = 0.5), fref = c(Fmsy = 1)), or B > Blim alone (bref = c(Blim = 0.3), fref = NULL); with risk = TRUE the complement is reported, e.g. the ICES risk P(B < Blim).

Probabilities are the proportions of projection draws in each condition, evaluated as B/Bmsy >= bref and F/Fmsy <= fref (with a small tolerance, so a scenario at exactly the F reference counts as below it), which for continuous draws is the same as the strict inequalities of the ICCAT tables.

Value

a data.frame with the scenario in the first column (TAC for constant catch scenarios, F/Fmsy for F scenarios, otherwise scenario) and one column per year; for what = "all" a named list of such data.frames, of class jb3k2sm

See Also

jbplot_kobematrix(), jb3_kobematrix_iotc(), jb3_projtable()

Examples

```
## Not run:
pc <- jb3_project(fit, catch = seq(1000, 3500, 250), nyears = 10)
jb3_kobematrix(pc, percent = TRUE)                        # P(green)
k2 <- jb3_kobematrix(pc, what = "all", percent = TRUE)   # ICCAT a), b), c)
jbplot_kobematrix(pc)
# ICES: B > Btrigger and F < Fmsy; risk P(B < Blim)
jb3_kobematrix(pc, what = "all", percent = TRUE, bref = c(Btrigger = 0.5))
jb3_kobematrix(pc, percent = TRUE, bref = c(Blim = 0.3), fref = NULL, risk = TRUE)
# named runs, in the order given
pr <- jb3_project(fit, scenarios = list(Fmsy = c(Fmsy = 1), Fcur = c(Fsq = 1), "0.7Fmsy" = c(Fmsy = 0.7),
                                        "0.8Fmsy" = c(Fmsy = 0.8)))
jb3_kobematrix(pr, what = "all", percent = TRUE)
# combined matrix of two models, equal weight
jb3_kobematrix(list(base = pc1, alt = pc2), what = "all", percent = TRUE)

## End(Not run)
```

---

jb3_kobematrix_gfcm *Kobe II strategy matrix, GFCM format*

Description

Three matrices for the GFCM advice summary sheet, by scenario (rows) and projection year (columns): (a) the probability of being in the green Kobe quadrant, F < Ftgt and B > Btgt, (b) the probability that B > Bthr, and (c) the risk P(B < Blim). Reference points are multiples of Bmsy and Fmsy; the defaults are the GFCM and ICES defaults for surplus production models (Btgt = Bmsy, Ftgt = Fmsy, Bthr = 0.5 Bmsy, Blim = 0.3 Bmsy).

`jb3_kobematrix_ices()`: two matrices for ICES-type advice, (a) the probability that F < Fmsy and B > Bpa, and (b) the risk P(B < Blim), with Bpa = 0.5 Bmsy and Blim = 0.3 Bmsy by default (SPM proxies).

Usage

```
jb3_kobematrix_gfcm(
  proj,
  years = NULL,
  btgt = 1,
  ftgt = 1,
  bthr = 0.5,
  blim = 0.3,
  percent = TRUE,
  weights = NULL,
  rows = c("auto", "label", "value"),
  digits = 2
)

jb3_kobematrix_ices(
  proj,
  years = NULL,
  fmsy = 1,
  bpa = 0.5,
  blim = 0.3,
  percent = TRUE,
  weights = NULL,
  rows = c("auto", "label", "value"),
  digits = 2
)
```

Arguments

`proj`	output of `jb3_project()` or `jb3_fwd()`, or a (named) list of them with the same scenarios and years, to be combined
`years`	projection years to show (default all)
`btgt`, `ftgt`, `bthr`, `blim`	reference points relative to Bmsy and Fmsy
`percent`	if TRUE, probabilities in percent, rounded to integers (ICCAT style)
`weights`	weights for combining a list of projections (default equal)

rows	row labels: "label" the scenario names in the order of the projection (e.g. Fmsy, Fcur, 0.8Fmsy, from named scenarios, fmsy or catch in jb3_project(), or a named fwdctrl list), "value" the TAC or F/Fmsy value; "auto" (default) uses the names when they were given
digits	rounding of probabilities when percent = FALSE
fmsy, bpa	F and biomass reference points of the ICES matrix (relative to Fmsy and Bmsy)

Value

a list of class jb3k2sm with three data.frames, for print() and jbplot_kobematrix()

Examples

```
## Not run:
k3 <- jb3_kobematrix_gfcm(pf)
k3
jbplot_kobematrix(k3, colours = "kobe")

## End(Not run)
```

---

jb3_kobematrix_iotc *Kobe II strategy matrix, IOTC format*

Description

Kobe II strategy matrix in the format of the IOTC stock assessment summaries: for each catch scenario (columns, labelled as a percentage of a reference catch and in tonnes) the probability (%) of violating the MSY-based reference points, P(B < Bmsy) and P(F > Fmsy), and the limit reference points, P(B < Blim) and P(F > Flim), in selected projection years (by default 3 and 10 years after the last assessment year).

Usage

```
jb3_kobematrix_iotc(
  proj,
  years = NULL,
  ref.catch = NULL,
  blim = 0.5,
  flim = 1.3,
  weights = NULL,
  digits = 1
)
```

Arguments

proj	output of jb3_project() (constant catch scenarios), or a list of them to combine
years	projection years (default the last assessment year + 3 and + 10, or the first and last projection year)
ref.catch	reference catch for the column labels (default the catch in the last assessment year)
blim, flim	limit reference points relative to Bmsy and Fmsy (IOTC defaults for tropical tunas differ by stock, e.g. Blim = 0.4 or 0.5 Bmsy, Flim = 1.3 or 1.4 Fmsy)
weights	weights for combining a list of projections (the "weighted probability" of the IOTC tables)
digits	decimals of the percentages

Value

a data.frame of class jb3k2iotc: columns block ("MSY" or "Limit"), refpt, year and one column per scenario

See Also

jb3_kobematrix() (ICCAT format), jbplot_kobematrix()

Examples

```
## Not run:
cref <- tail(fit$data$catch, 1)
pc <- jb3_project(fit, catch = cref*c(0.6, 0.7, 0.8, 0.9, 1, 1.1, 1.2), nyears = 10)
k2 <- jb3_kobematrix_iotc(pc, ref.catch = cref)
k2
jbplot_kobematrix(k2)                       # grey
jbplot_kobematrix(k2, colours = "kobe")     # colour

## End(Not run)
```

---

jb3_leslie *Intrinsic rate of increase and generation time from a Leslie matrix*

Description

The maximum intrinsic rate of increase $r = \log \lambda$ (dominant eigenvalue of the Leslie matrix with recruitment at the slope at the origin of the Beverton-Holt stock-recruitment relationship, $4h/(\phi_0(1-h))$) and the generation time GT (mean age of spawners in the unfished population), as JABBA::jbleslie() and FLSRTMB::productivity().

Usage

```
jb3_leslie(
  x,
  h = NULL,
  sex = "F",
  lhpar = NULL,
  amin = 0,
  amax = NULL,
  plus_group = TRUE
)
```

Arguments

x	lhpar, FLStock, FLQuants, or list/data.frame of biology at age
h	steepness (default from lhpar$s, else 0.7)
sex	sex that determines reproduction (default "F")
lhpar	growth parameters for length-based FLQuants
amin, amax	age range for lhpar (default 0 to the age at which 1% survive)
plus_group	treat the last age as a plus group

Details

Inputs:

FLicc lhpar an FLPar (or named vector or list) with linf, k, M (or Mk), L50, L95, a, b and optionally t0 (default -0.5), s (steepness, default 0.7) and amax; biology at age from von Bertalanffy growth, $W = aL^b$ and logistic maturity at length. Sex-structured: an FLPar with a second dimension (e.g. sex = c("F", "M")), or list(F = , M =); reproduction from sex.

FLStock m, mat, stock.wt by age (median over years and iterations; sex from the unit dimension if it has "F").

FLQuants m, mat, wt (or stock.wt) by age with unit = c("F", "M") for sexes, or by length (first dimension len) together with lhpar for growth (length at age).

list or data.frame age, M, mat, wt (as JABBA::jbio()).

Value

named numeric: r, gt, lambda, spr0 (unfished spawning biomass per recruit)

Examples

```
lh <- c(linf = 55.7, k = 0.08, M = 0.162, L50 = 31.2, a = 0.0047, b = 3.15, s = 0.75)
jb3_leslie(lh)
```

jb3_model *Model structure for JABBA3*

Description

Structural settings, separate from the priors (as `settings` in FLicc).

Usage

```
jb3_model(
  shape = c("pella", "schaefer", "fox"),
  BmsyK = 0.4,
  lag = 0,
  sdb.form = c("const", "dyn"),
  sdb.fixed = NULL,
  xlim = c(0.2, 3),
  rho = 0,
  rw = FALSE,
  rw.scale = 1,
  biascor = FALSE,
  start = c("all", "index"),
  obs.error = c("additive", "spict"),
  se.min = NULL,
  se.na = 0.3,
  sets.q = NULL,
  sets.var = NULL,
  catch.error = TRUE,
  sdc = 0.1,
  ebsb = NULL,
  ebsb.index = NULL,
  ebsb.catch = 1,
  ebsb.removal = TRUE,
  q.uniform = FALSE,
  P.bound = c(0.02, 1.3),
  K.bound = c(0.01, 1e+10),
  sdi.bound = 1,
  sdi.est = TRUE,
  param = c("centred", "noncentred"),
  effort.creep = 0,
  aux.q = NULL,
  aux.sd = NULL,
  aux.se.min = NULL,
  frw.all = FALSE,
  nseason = 1,
  catch.alloc = "F",
  season.w = NULL,
  sdb.max = NULL
```

)

Arguments

shape	"pella" (shape from BmsyK), "schaefer" (Bmsy/K = 0.5) or "fox" (m = 1.001)
BmsyK	Bmsy/K for the Pella-Tomlinson form (0.4 default, as JABBA)
lag	production lag in years (Tier 1: age at maturity); production in year t uses the depletion of t - lag. With nseason > 1 the lag may be fractional and is rounded to whole seasons (e.g. 0.75 years = 3 quarters)
sdb.form	"const" (default) or "dyn": process error SD sdb_t = sdb x (B/Bmsy)^-k (Tier 2)
sdb.fixed	fixed process error SD at Bmsy instead of estimating it (Tier 2 sets the tuned level here)
xlim	bounds of B/Bmsy in the dynamic SD
rho	AR(1) coefficient of the process deviations (0 = independent, default). sdb stays the marginal SD: d_t ~ N(rho d_t-1, sdb_t^2 (1 - rho^2)). Estimated instead when priors$rho is switched on.
rw	random walk in the process deviations (the conditional rho = 1 limit): d_t ~ N(d_t-1, (rw.scale sdb_t)^2)
rw.scale	innovation SD of the random walk relative to sdb (e.g. sqrt(1 - 0.7^2) to match AR(1) rho = 0.7)
biascor	lognormal process error bias correction (log mean - var/2)
start	"all" (default): process error in all years; "index": deterministic before the first index year (JABBA proc.dev.all = FALSE)
obs.error	"additive" (default, JABBA): total SD = sqrt(se^2 + sdi^2); "spict": SD = sdi x relative se weights
se.min	fixed minimum observation SD; NULL = JABBA default (0.1 without SEs, 0.001 with SEs)
se.na	SE used where an index has SEs but a year has none (JABBA: 0.3)
sets.q, sets.var	catchability and observation-variance groups (one entry per index; default one each)
catch.error	estimate catch with lognormal error (random effects with SD sdc); on by default, as JABBA (add.catch.CV = TRUE)
sdc	catch observation SD (log scale) when catch.error = TRUE: a constant or a vector along the years (year-specific catch uncertainty; see also jb3_data(catch.se =), which takes precedence where given)
ebsb	Selex: EB/SB ratio curve c(u1, u2, u3, P1, P2), a named list of curves (one per selectivity) or the output of jb3_selex(); the state is then spawning biomass, indices are modelled as q x SB x (EB/SB)(SB/SB0)
ebsb.index	curve per index (number or name; 0 = spawning-biomass index); default curve 1 for all
ebsb.catch	curve for the removals (default 1)
ebsb.removal	catch removes C / (EB/SB) from spawning biomass (default TRUE)

`q.uniform` legacy JABBA uniform prior on q (density proportional to q on log q)

`P.bound`, `K.bound`
soft bounds on B/K and K (JABBA defaults)

`sdi.bound` soft upper bound on sdi (JABBA: 1)

`sdi.est` estimate the additional observation SD of the indices (default); FALSE uses the input SEs (or `se.min`) only, as JABBA with `fixed.obsE` and `sigma.est = FALSE`

`param` parameterisation of the process deviations for NUTS: `"centred"` (default; the states log(B/K) are sampled directly) or `"noncentred"` (standard normal innovations z_t ~ N(0, 1) are sampled, and d_t = rho d_t-1 + sdb_t sqrt(1 - rho^2) z_t, with lag, bias correction, random walk and dynamic SD as in the centred form). The model is the same; only the coordinates of the sampler change. The posterior mode (`engine = "rtmb"`) always uses the centred form. For bigeye tuna the centred form mixed better, with and without an index (see the user guide), because each state depends on all earlier innovations through the nonlinear production function; the non-centred form is kept as an option for weakly informed states.

`effort.creep` auxiliary effort only: annual effort creep (increase of catchability per unit effort), fixed (default 0), or `"estimate"` with the normal prior `jb3_priors(creep = )`. Effort is modelled as E_t = F_t / (q_E (1 + creep)^(t - t0)), t0 the first effort year.

`aux.q` catchability of the auxiliary series: `NULL` (default) estimates q for "effort" and "z" (flat on log q) and fixes q = 1 for "f", "ffmsy", "bbmsy" and "bk" (absolute measurements); or one entry per series, a number (fixed q) or NA (estimated). For effort, q is the catchability F = q E in the first effort year; for the other types the observation is q times the model quantity.

`aux.sd` additional observation SD of the auxiliary series: `NULL` (default) estimates one per series (JABBA `auxiliary.sigma = TRUE`, diffuse inverse-gamma prior as for the indices); or fixed value(s) per series

`aux.se.min` minimum observation SD of the auxiliary series, NULL = JABBA default (0.1 without SEs, 0.001 with)

`frw.all` only with missing catch (`jb3_data(catch.gaps = "estimate")`): FALSE (default) applies the random walk on log harvest rate only to increments involving a year without catch, so the years with catch stay catch-conditioned as before (sdf from its prior, `jb3_priors(sdf = )`); TRUE applies it to all years (SPiCT-like; sdf then informed by the years with catch, which tends to shrink it and smooth F through the gaps)

`nseason` number of seasonal time steps per year (1 = annual model, as JABBA). The state of each step is the depletion at the start of the season; production per step is r/nseason (times `season.w`), catch of a step is removed during that step (an Euler scheme with dt = 1/nseason). Process variance per step is set so that the deviations of a year add up to the annual `sdb` (with AR(1): step coefficient rho^(1/nseason)); the lag is given in years. Annual outputs: biomass at the start of the year, harvest rate = annual catch / mean exploitable biomass over the year; seasonal series in `fit$ts.season`.

`catch.alloc` how annual catch is split over the seasons in years without seasonal catch: "F" (default; constant harvest fraction within the year, i.e. catch proportional to the

	exploitable biomass of each season, as SPiCT's F x B), "equal" (catch / nseason), or a vector of shares (one per season, e.g. from quarterly landings)
season.w	relative production weights of the seasons (rescaled to mean 1; NULL = equal), e.g. a recruitment pulse c(0, 3, 1, 0)
sdb.max	optional upper cap on the process error SD sdb (soft bound, as sdi.bound); NULL = none

Value

list of class jb3model

---

jb3_noise *Observation and process noise of indices*

Description

Three measures of how noisy each index is: the mean input CV (sampling or observation error, from the log-SEs), the process-error CV of Francis (2011), from the residuals of a loess smoother fitted to the log index over time (the year-to-year noise the index carries beyond a smooth trend), and, for a trend fit, the estimated additional SD $\sigma_{p,i}$ of each index against the common trend (Conn 2010) with the RMSE of its residuals. A loess CV much larger than the input CV, or a large $\sigma_{p,i}$, points to variation in catchability or availability that the sampling error does not capture; a survey index estimated with a design-based estimator typically carries more such noise than one standardised with a (geostatistical) model (Winker et al. 2017).

Usage

```
jb3_noise(x, se = NULL, span = 0.6, digits = 3)
```

Arguments

x	a jb3trend fit, or a data.frame of indices (year in the first column)
se	data.frame of log-SEs in the layout of x (when x is a data.frame)
span	loess span (Francis 2011 smoother)
digits	rounding (NULL for none)

Value

data.frame(index, n, years, obs.cv, loess.cv, and for a trend fit sd.extra, sd.total, rmse)

References

Francis, R.I.C.C. (2011) Data weighting in statistical fisheries stock assessment models. CJFAS 68, 1124-1138.

Examples

```
## Not run:
tr <- jb3_trend(iccat$bum$cpue, iccat$bum$se)
jb3_noise(tr); jbplot_noise(tr)

## End(Not run)
```

`jb3_nuts_diags` *NUTS convergence summary*

Description

Rank-normalised split R-hat, bulk and tail effective sample sizes (Vehtari et al. 2021) for every sampled parameter, and the sampler diagnostics: divergent transitions, maximum tree depth hits and E-BFMI by chain.

Usage

```
jb3_nuts_diags(fit, states = TRUE)
```

Arguments

`fit`	a `jabba3` fit with `engine = "nuts"`
`states`	include the log-depletion states and catch deviations

Value

list(pars = data.frame(par, mean, sd, q2.5, q50, q97.5, Rhat, ess_bulk, ess_tail), sampler = data.frame by chain, ok = logical verdict: R-hat < 1.01, bulk and tail ESS > 100 per chain, no divergences, E-BFMI > 0.3)

`jb3_nuts_ess` *Effective sample size of the derived states*

Description

Bulk and tail ESS (rank-normalised, Vehtari et al. 2021) and R-hat of the derived trajectories (e.g. B/Bmsy and F/Fmsy by year) of a NUTS fit, also per 1000 post-warmup draws. Useful to compare parameterisations (`jb3_model(param = "centred")` against `"noncentred"`), because the sampled states differ between them.

Usage

```
jb3_nuts_ess(fit, quantities = c("B/Bmsy", "F/Fmsy"))
```

Arguments

`fit`	a `jabba3` fit with `engine = "nuts"`
`quantities`	derived quantities, any of "B/Bmsy", "F/Fmsy", "B/K", "B", "F"

Value

data.frame(quantity, year, Rhat, ess_bulk, ess_tail, bulk.per1000, tail.per1000); attribute "summary" with minimum and median by quantity and the run time and parameterisation

---

`jb3_osa` *One-step-ahead residuals*

Description

OSA (prediction) residuals of the abundance indices, as in SPiCT: residual = qnorm(P(I_t <= obs | I_1..I_t-1)) with the fixed effects at the mode and the states integrated sequentially by a bootstrap particle filter (exact up to Monte Carlo error for the nonlinear production model, including lag, dynamic process error and Selex). Under a correct model they are iid N(0, 1). Ordinary residuals from the smoothed states are shrunk and autocorrelated by construction, so residual tests use the OSA residuals by default.

Usage

```
jb3_osa(fit, np = 5000, seed = 1)
```

Arguments

`fit`	a `jabba3` fit
`np`	number of particles
`seed`	random seed

Value

numeric vector, in the order of the observations in `fit$index`

jb3_partable *Parameter and reference point table*

Description

Estimates with 95% intervals (posterior median and credible interval for NUTS fits, mode and delta-method interval otherwise) for the model parameters, reference points and the terminal status.

Usage

```
jb3_partable(x, digits = 3, format = c("long", "wide"))
```

Arguments

x	a jabba3 fit or a (named) list of fits
digits	significant digits for the formatted table
format	"long" (one row per run and parameter, numeric columns) or "wide" (one column per run, "est (lci-uci)")

Value

data.frame

Examples

```
## Not run:
jb3_partable(fit)
jb3_partable(list(Fox = fit, Lag.AR1 = rec), format = "wide")

## End(Not run)
```

jb3_priors *Priors for JABBA3*

Description

One named list of priors, each c(median, log.sd, on): lognormal with the given median and log-scale sd, and an on/off switch (on = 0 gives a flat prior on the log scale). The defaults follow JABBA (build_jabba()), except the process error prior.

Usage

```
jb3_priors(
  data = NULL,
  r = c(0.2, 0.5, 1),
  K = NULL,
  bkinit = c(0.9, 0.246, 1),
  sdb = "fishlife",
  sigR = 0.5,
  q = NULL,
  m = c(NA, 0.3, 0),
  kdyn = NULL,
  rho = c(0.7, 0.15, 0),
  sdi = c(0.2, 0.5, 0),
  status = NULL,
  psi = NULL,
  creep = c(0.02, 0.02),
  sdf = c(0.3, 0.5, 1),
  K.max = NULL
)
```

Arguments

data	optional jb3data, used for the default K prior (8 x max catch, CV 1, as JABBA)
r	intrinsic growth rate, on by default
K	carrying capacity; NULL = 8 x max catch with log.sd 0.833 (JABBA: CV 1)
bkinit	initial depletion B1/K (JABBA: psi); default c(0.9, 0.246) (JABBA: CV 0.25)
sdb	process error SD: "fishlife" (default) = LN(0.80 x sigR x r^0.45, 0.3) with r the median of the r prior (FishLife relation for the SD at Bmsy); or c(median, log.sd, on); or "igamma" for JABBA's diffuse inverse-gamma(0.001, 0.001) on the variance (not recommended for the posterior mode).
sigR	recruitment SD used by sdb = "fishlife" (0.5 when unknown)
q	catchability: NULL = flat on log q for every q (default); otherwise a matrix (or vector) with one row c(median, log.sd, on) per catchability group
m	Pella-Tomlinson shape: off by default (m fixed from BmsyK in jb3_model()); c(median, log.sd, 1) to estimate m
kdyn	depletion dependence of the process error SD (sdb.form = "dyn"), c(mean, sd, on) normal on k, bounded to (-0.5, 2); NULL = life-history centre max(0, 0.064 - 0.155 log r), sd 0.15
rho	AR(1) coefficient of the process deviations, c(mean, sd, on) normal on rho in (0, 0.99); off by default (rho fixed by jb3_model(rho =))
sdi	index observation SD prior (only with obs.error = "spict"), off by default
status	optional status priors: a data.frame(year, type, median, log.sd) with type one of "BK", "BBmsy", "FFmsy" (lower case and JABBA's "bk", "bbmsy", "ffmsy" accepted; year NA = terminal year of the fit), the output of jb3_bprior(), or a JABBA b.prior vector c(mean, cv, year, type) (the cv used as log-scale SD, as in JABBA; year NA = terminal year)

psi	alias of bkinit (JABBA name)
creep	normal prior c(mean, sd) on the annual effort creep, used with jb3_model(effort.creep = "estimate")
sdf	SD of the random walk in log harvest rate, used for years without catch (jb3_data(catch.gaps = "estimate")): lognormal prior c(median, log.sd, on), default median 0.3 (log.sd 0.5). SPiCT benchmarks often use a loose prior with the catch SD fixed, e.g. c(4, 0.5, 1), together with jb3_model(frw.all = TRUE).
K.max	optional soft upper prior on K, to avoid implausibly large biomass when the data say little about the scale: no penalty below the cap, and a half-normal on log K above it. "r" sets the cap from r (the median of the r prior) at the 95th percentile of K / max catch across 72 ICES stocks (K = unfished exploitable biomass): cap = 7.6 x max catch x r^-0.587 (about 19 x max catch at r = 0.2, 12 x at 0.4, 7.6 x at 1); a number gives the cap as a multiple of the maximum catch; c(multiple, log.sd) also sets the SD of the half-normal (default 0.3). NULL (default) = no cap

Value

list of class jb3priors

Examples

```
jb3_priors(r = c(0.3, 0.3))
```

---

jb3_project *Projections*

Description

Stochastic projections under constant F/Fmsy (default) or constant catch scenarios, from posterior draws (NUTS) or pseudo-posterior draws at the mode (generated if the fit has none), with process error and the production lag. Works for a single fit and for a weighted jb3_ensemble(). Optional intermediate years (imp.years) with a fixed catch (default the status quo catch) or F/Fmsy precede the scenarios, as the intermediate years of short-term forecasts. A wrapper around jb3_fwd(), which takes general forecast controls.

Usage

```
jb3_project(
  fit,
  fmsy = NULL,
  catch = NULL,
  nyears = 10,
  ndraws = 1000,
  procerr = c("ar1", "mean", "none"),
  imp.years = 0,
```

```
  imp.catch = NULL,
  imp.fmsy = NULL,
  nsq = 3,
  scenarios = NULL
)
```

Arguments

fit	a jabba3 fit or a jb3ensemble
fmsy	vector of F/Fmsy multipliers (one scenario each); default c(0, 0.5, 0.75, 1, 1.25)
catch	alternatively, vector of constant catches
nyears	number of projection years (including the intermediate years)
ndraws	draws when the fit has none
procerr	process error in the projection years, consistent with the fitted model: "ar1" (default) continues each draw's AR(1) process from its last estimated deviation, d_t+1 = rho d_t + e, e ~ N(0, sdb^2 (1 - rho^2)), so persistence carries into the forecast and the marginal SD stays sdb (with rho = 0 this is iid sampling as in JABBA's fw_jabba()); "mean" uses the conditional mean only, d_t+h = rho^h d_T (no new innovations; uncertainty from the parameters and states only); "none" sets the deviations to zero (deterministic production; the bias correction is dropped too)
imp.years	number of intermediate years before the scenarios start (default 0)
imp.catch	catch in the intermediate years (default: status quo catch, mean of the last nsq years)
imp.fmsy	alternatively, F/Fmsy in the intermediate years
nsq	years for the status quo
scenarios	alternatively, a named list of scenarios in the order to report them, each one named value c(quant = value) with quant "Fmsy" (F/Fmsy), "Catch", "F", "Fsq" or "Csq" (multipliers of the status quo F or catch), e.g. list(Fmsy = c(Fmsy = 1), Fcur = c(Fsq = 1), "0.8Fmsy" = c(Fmsy = 0.8), "TAC 2000" = c(Catch = 2000)); the names label the scenarios in plots and tables. Named fmsy or catch vectors are labelled by their names too.

Details

The first transition uses the estimated catch of the last assessment year, so the biomass in the first projection year is a one-step-ahead prediction; catches are capped at 95% of the exploitable biomass.

Value

list of class jb3proj: ts (scenario, year, quantity, quantiles), iters, scenarios, status, ctrl, tac.year

`jb3_projtable` *Projection tables*

Description

`jb3_projtable()`: catch options table by scenario, in the style of ICES advice sheets: F/Fmsy and F, the catch in the first projection year, median B/Bmsy in selected years with the change from the terminal assessment year, and the probabilities of B >= Bmsy, F <= Fmsy and of the Kobe green quadrant.

Usage

```
jb3_projtable(proj, years = NULL, digits = 3, blim = 0.3)
```

Arguments

`proj`	output of `jb3_project()`
`years`	projection years to report (calendar years); default the first, fifth and last projection year
`digits`	rounding
`blim`	limit as a fraction of Bmsy for the risk P(B < Blim) (default 0.3, SPiCT benchmark default; NULL to omit)

Details

`jb3_kobematrix()`: Kobe II strategy matrix (ICCAT): probability by scenario (rows) and projection year (columns) of being in the green quadrant (B >= Bmsy and F <= Fmsy), or of B >= Bmsy or F <= Fmsy alone.

Value

data.frame

Examples

```
## Not run:
pj <- jb3_project(fit, fmsy = c(0, 0.5, 0.75, 1, 1.25))
jb3_projtable(pj)
jb3_kobematrix(pj)

## End(Not run)
```

jb3_qtable *Catchability relative to the reference index*

Description

For a relative trend fit (`jb3_trend()`), the catchability of each index relative to the first (reference) index, whose q is 1 by definition: the scaling that aligns each index with the common trend. Estimates and 95% intervals come from the sdreport of log q (lognormal), or from the NUTS draws for `engine = "nuts"`.

Usage

```
jb3_qtable(fit, digits = NULL)
```

Arguments

`fit`	a relative `jb3trend` fit
`digits`	rounding (NULL for none)

Value

data.frame(index, q, lci, uci, log.se, n, years, reference)

Examples

```
## Not run:
tr <- jb3_trend(iccat$bet$cpue, iccat$bet$se)
jb3_qtable(tr); jbplot_q(tr)

## End(Not run)
```

jb3_retro *Retrospective analysis*

Description

Refits the model with the last 1..n years of all data removed and computes Mohn's rho for B, F, B/Bmsy and F/Fmsy.

Usage

```
jb3_retro(fit, n = 5)
```

Arguments

`fit`	a `jabba3` fit
`n`	number of peels

Value

list of class `jb3retro`: `fits` (reference first), `ts` (long), `rho`

`jb3_rho_gt` *Life-history guidance for the AR(1) persistence of process error*

Description

The persistence of process deviations in exploitable biomass increases with generation time (GT). Across age-structured simulations for 486 bony-fish species of the FAO SOFIA list (FishLife life histories, F-contrast trajectories) the lag-1 autocorrelation followed $\rho \approx \exp(-c/GT)$ with c = 2.25 (R^2 = 0.81; c = 2.14 with size-dependent selectivity, c = 2.35 in nine ICES case-study operating models). The default `rho = 0.75` corresponds to GT of about 6-10 years; use this function to adjust it for shorter- or longer-lived stocks. GT can be computed from the same Leslie matrix as the r prior (e.g. `JABBA::jbleslie()` or `FLSRTMB::productivity()`).

Usage

```
jb3_rho_gt(gt, c = 2.2, digits = 2)

jb3_rho_table()
```

Arguments

`gt`	generation time (years)
`c`	decay constant (default 2.2)
`digits`	rounding

Value

suggested rho

`jb3_rho_table()`: guidance table by generation-time class with the FishLife calibration medians

Examples

```
jb3_rho_gt(c(2, 4, 8, 15))
jb3_rho_table()
```

jb3_runs *Runs test with three-sigma control limits*

Description

Wald-Wolfowitz runs test on the signs of residuals (normal approximation) and three-sigma limits from the average moving range (Montgomery), as in JABBA `jbruns_sig3()`, ss3diags and JAGG.

Usage

```
jb3_runs(x, mixing = c("less", "two.sided", "greater"))
```

Arguments

`x`	residuals
`mixing`	"less" (default, one-sided: too few runs), "two.sided" or "greater"

Value

list with `sig3lim` (lcl, ucl) and `p.runs`

jb3_sel5 *JABBA 5-parameter selectivity*

Description

The JABBA-Select selectivity (parameters as `JABBA::jselex()`): logistic ascending limb with 50% and 95% selection at `SL50` and `SL95`; beyond `SLdesc` a normal descending limb (SD = `SLdesc` x `CVdesc`) declining from 1 towards a minimum `Smin` (`Smin` = 1, or a large `SLdesc`, gives logistic selectivity). Note: `JABBA::jselex()` as coded inverts the descending limb (selectivity drops to `Smin` just beyond `SLdesc` and rises again); here the descending limb declines continuously. Usable on length or age (`x` and the parameters on the same scale).

Usage

```
jb3_sel5(x, pars, type = c("length", "age"))
```

Arguments

`x`	lengths or ages
`pars`	c(SL50, SL95, SLdesc, CVdesc, Smin), named or in this order
`type`	"length" or "age" (documentation of the scale; the function is the same)

Value

selectivity in [0, 1]

Examples

```
jb3_sel5(seq(10, 100, 10), c(SL50 = 35, SL95 = 45, SLdesc = 70, CVdesc = 0.15, Smin = 0.05))
jb3_sel5(0:15, c(2, 3, 8, 0.2, 0.1), type = "age")
```

`jb3_selex` *Selex inputs from length-based selectivity: EB/SB curves for the spawning-biomass state*

Description

Computes, for each fleet (selectivity), the equilibrium ratio of exploitable to spawning biomass EB/SB as a function of spawning-biomass depletion P = SB/SB0 over a grid of F, and fits the saturating curve $EB/SB(P) = u_1 + (u_2 - u_1)(1 - e^{-u_3(P-P_1)})/(1 - e^{-u_3(P_2-P_1)})$ used by `jb3_model(ebsb = )` (as `JABBA::make_ebsb()` and `fit_ebsb()`). Biology at age comes from an FLicc `lhpar` (von Bertalanffy growth, weight and maturity at length, M, steepness `s`); selectivity is defined at length with the FLicc functions and parameters (`selpar` on the length scale, e.g. from `FLicc::selpars_flicc(fit)`) and converted to age by integrating over the length-at-age distribution (coefficient of variation `CVL`). Sex-structured `lhpar` (dimension `sex = c("F", "M")`): both sexes contribute to EB and SB with `sex_ratio`.

Usage

```
jb3_selex(
  lhpar,
  selpar,
  sel.type = "length",
  h = NULL,
  CVL = 0.1,
  sex_ratio = NULL,
  amin = 0,
  amax = NULL,
  Fgrid = NULL
)
```

Arguments

`lhpar`	FLicc life-history parameters (FLPar, named vector or `list(F = , M = )`)
`selpar`	FLicc selectivity parameters on the length scale: an FLPar (one fleet), FLPars or named list (one per fleet); the selectivity function is identified from the parameter names (logistic: SL50, SL95; dsnormal: mode, lsd, rsd; normal: mode, sd; binormal; bidsnormal)
`sel.type`	"length" (default) or "age", per fleet; age-based selectivity uses the JABBA 5-parameter form (`jb3_sel5()`), which can also be used at length (parameters SL50, SL95, SLdesc, CVdesc, Smin)
`h`	steepness (default `lhpar$s`, else 0.7)
`CVL`	coefficient of variation of length at age (0: selectivity at the mean length)

sex_ratio	weights of the sexes (default equal)
amin, amax	age range
Fgrid	grid of F (default: from 0 until SB/SB0 < 0.02)

Value

object of class jb3selex: ebsb (named list of c(u1, u2, u3, P1, P2), ready for jb3_model(ebsb =)), curves (equilibrium EB/SB by fleet), at.len and at.age (selectivity and maturity), leslie (r and GT)

Examples

```
lh <- c(linf = 55.7, k = 0.08, M = 0.162, L50 = 31.2, a = 0.0047, b = 3.15, s = 0.75)
sx <- jb3_selex(lh, list(Gillnet = c(mode = 32, lsd = 4, rsd = 15), Trawl = c(SL50 = 22, SL95 = 28)))
sx$ebsb
```

---

jb3_splitindex *Split an index at a change in vessel, gear or method*

Description

Splits one index into two series at a known change in catchability (a new survey vessel, a new trawl gear, a change in the fishery), so that a trend or production model estimates a separate q for each period. Years fished with both set-ups (calibration or paired tows) can be given to both series; they identify the q ratio directly, while without overlap the ratio rests on the continuity of the trend across the change.

Usage

```
jb3_splitindex(
  I,
  se = NULL,
  index,
  at,
  overlap = NULL,
  new = NULL,
  labels = paste0(index, c(".old", ".new"))
)
```

Arguments

I	data.frame of indices (year in the first column)
se	data.frame of log-SEs in the layout of I (optional)
index	name of the index to split
at	first year of the new set-up

overlap	years in which the index is measured with both set-ups: the values go to both series (for calibration years with separate values, supply them in new)
new	optional values of the new set-up in the overlap years (same length)
labels	names of the two series

Value

list(I, se) with the split index replaced by two columns

---

jb3_trend *State-space trend models for abundance indices (JARA)*

Description

Fits the JARA models in RTMB: a random walk with drift on log abundance observed through one or several indices. The relative joint index follows the hierarchical analysis of multiple noisy abundance indices of Conn (2010), as used in SEDAR assessments; JARA's random-walk form gives practically the same joint index, and state = "iid" fits Conn's independent-year-effects form directly.

"relative" joint index of relative abundance indices (CPUE, surveys): each index has its own catchability q (q of the first index fixed at 1) and observation variance $se^2 + fixed.obsE^2 + sdi^2$; the common state is the joint trend.

"census" indices are absolute abundance (e.g. swept-area or acoustic biomass) of different areas: one state per area (q = 1), smoothed through missing years, summed to a total abundance with uncertainty.

variance.weighting: "model" estimates an additional observation SD per index, "equal" one common additional SD, "fixed" uses only the input SEs plus fixed.obsE. Weak lognormal priors are placed on the process SD (sdp.pr) and the additional observation SD (sdi.pr). The fit carries ordinary and OSA residuals (RTMB::oneStepPredict, exact for this Gaussian model) in the same format as a jabba3 fit, so jbplot_fits(), jbplot_residuals(), jbplot_runs() and jbplot_ppc() apply directly. As in JABBA3 the residual plots default to OSA residuals; type = "ordinary" gives the log residuals used by JARA.

Usage

```
jb3_trend(
  I,
  se = NULL,
  type = c("relative", "census"),
  state = c("rw", "iid"),
  fixed.obsE = 0,
  variance.weighting = c("model", "fixed", "equal"),
  drift = TRUE,
  timeblock = NULL,
  na.below = NULL,
```

```
  sdp.pr = c(0.1, 1),
  sdi.pr = c(0.1, 1),
  se.na = 0.2,
  engine = c("rtmb", "nuts"),
  nuts = list(),
  osa = TRUE,
  run = NULL
)
```

Arguments

I	data.frame(year, index1, ...) or FLIndices / FLIndexBiomass
se	optional data.frame of log-scale SEs with the same columns (ignored for FLIndices, which carry index.var)
type	"relative" or "census"
state	relative model only: "rw" (default), the JARA random walk with drift on the log joint state; or "iid", independent year effects $n_t \sim N(\bar{n}, 1)$ without time-series structure, i.e. the hierarchical model of Conn (2010) used in SEDAR assessments ($\log U_{it} \sim N(n_t + c_i, \sigma^2_{p,i} + \sigma^2_{s,it})$, with variance.weighting = "model" for the index-specific $\sigma_{p,i}$)
fixed.obsE	additional fixed observation SD added to all indices (as in JARA)
variance.weighting	"model", "equal" or "fixed"
drift	estimate the drift r (random walk with drift); FALSE for a plain random walk
timeblock	optional year(s): the drift (mean rate of change) differs before and after each block year (as JARA's timeblock); missing values are interpolated with the block-specific trend
na.below	optional threshold: values below it are treated as unobserved (NA), e.g. incomplete low reports
sdp.pr, sdi.pr	lognormal priors c(median, log.sd) on the process SD and the additional observation SD
se.na	SE used where an index value has no SE
engine	"rtmb" (mode, Laplace) or "nuts" (tmbstan)
nuts	list of tmbstan settings (chains, iter, warmup, seed, cores)
osa	compute OSA residuals
run	label

Value

object of class jb3trend

References

Conn, P.B. (2010). Hierarchical analysis of multiple noisy abundance indices. Canadian Journal of Fisheries and Aquatic Sciences, 67, 108-120. doi:10.1139/F09-175

Examples

```
## Not run:
data(iccat)
tr <- jb3_trend(iccat$bum$cpue, iccat$bum$se, fixed.obsE = 0.1)
jbplot_jointindex(tr); jbplot_fits(tr); jbplot_runs(tr, type = "ordinary"); jbplot_ppc(tr)
ji <- jb3_jointindex(tr)

## End(Not run)
```

---

jb3_tstable *Time series table of estimated quantities*

Description

Time series table of estimated quantities

Usage

```
jb3_tstable(
  fit,
  quantities = c("Catch", "B", "F", "B/Bmsy", "F/Fmsy", "B/K"),
  ci = TRUE,
  years = NULL
)
```

Arguments

fit	a jabba3 fit
quantities	any of "Catch", "B", "F", "B/Bmsy", "F/Fmsy", "B/K", "Surplus production", "Process deviation", "Process SD"
ci	include the 95% interval columns
years	optional subset of years

Value

data.frame, one row per year and columns <quantity>, <quantity>.lci, <quantity>.uci

jb3_weights *Ensemble weights from model diagnostics*

Description

Scores each fit on diagnostic criteria, as used to weight JABBA ensembles: convergence (a fit that did not converge gets weight 0), runs tests of the residuals (all indices pass), the RMSE of the residuals (below `rmse.max`), and optionally the hindcast prediction skill (MASE < 1 for all indices, from `jb3_hindcast()`) and the retrospective pattern (|Mohn's rho| of B/Bmsy within `rho.lim`, from `jb3_retro()`). The weight is the proportion of criteria passed (normalised to sum to 1); `weight = "equal"` gives equal weights to all converged runs.

Usage

```
jb3_weights(
  fits,
  hindcast = NULL,
  retro = NULL,
  rmse.max = 0.3,
  rho.lim = c(-0.15, 0.2),
  weight = c("score", "equal")
)
```

Arguments

`fits`	named list of fits
`hindcast`	optional list of `jb3_hindcast()` outputs (same order)
`retro`	optional list of `jb3_retro()` outputs (same order)
`rmse.max`	RMSE threshold (log scale)
`rho.lim`	Mohn's rho bounds for B/Bmsy
`weight`	"score" or "equal"

Value

data.frame(run, converged, runs, rmse, mase, mohn, score, weight)

jb3_workflow *BioProcDyn workflows: Tier 1, Tier 2 and Selex*

Description

Runs the workflows of the BioProcDyn protocol, automatically (`jabba3(..., workflow = )`) or step by step (`jb3_tier1()`, `jb3_tier2()` on any fit).

- `"tier1"`: production lag at the age at maturity (`amat`).
- `"tier2"`: step 1 = constant sdb (FishLife sdb prior); step 2 = dynamic sdb_t = sdb x (B/Bmsy)^-k with the level tuned on step 1, sdb = sdb1 / sqrt(mean(x^-2k)) over the step-1 B/Bmsy (bounded to 0.2-3), k estimated with the life-history prior, plus the lag.
- `"selex"`, `"selex_tier1"`, `"selex_tier2"`: the same with the spawning-biomass state and EB/SB-mapped indices and removals; needs `model$ebsb` and priors on the SSB scale.

Usage

```
jb3_workflow(data, priors, model, workflow, amat = NULL, ...)

jb3_tier1(fit, amat, ...)

jb3_tier2(fit, amat = NULL, ...)
```

Arguments

`data`, `priors`, `model`	as `jabba3()`
`workflow`	one of "tier1", "tier2", "selex", "selex_tier1", "selex_tier2"
`amat`	age at 50% maturity (lag)
`...`	passed to `jabba3()`
`fit`	a `jabba3` fit (step 1)

Value

the final fit, with all steps in `$steps`

jb3cols *Colours for runs, scenarios or peels (r4ss rule, as JAGG and ss3diags)*

Description

Colours for runs, scenarios or peels (r4ss rule, as JAGG and ss3diags)

Usage

```
jb3cols(n, alpha = 1)
```

Arguments

n	number of colours
alpha	transparency

jbdat2flr *Convert classic JABBA data frames to FLR objects*

Description

Converts the JABBA input data frames (as used by `JABBA::build_jabba()`) to FLR objects: catch to an `FLQuant` (or `FLQuants`, one per fleet), abundance indices to `FLIndices` of `FLIndexBiomass` with the log-scale standard errors in `index.var`, and auxiliary data (e.g. effort) to `FLQuants`. The result can be passed straight to `jb3_data()`.

Usage

```
jbdat2flr(
  catch,
  index = NULL,
  index.se = NULL,
  auxiliary = NULL,
  auxiliary.se = NULL,
  units.catch = "t",
  units.index = "NA",
  fleets = NULL
)

jbcatch2flq(catch, units = "t", fleets = NULL)

jbidx2FLIndices(index, index.se = NULL, units = "NA")
```

Arguments

catch	data.frame(year, catch[, catch fleet 2, ...])
index	data.frame(year, index1, index2, ...) (JABBA cpue); NULL for catch-only
index.se	optional data.frame(year, se1, se2, ...) of log-scale standard errors (JABBA se); same columns as index
auxiliary	optional data.frame(year, aux1, ...) (JABBA auxiliary, e.g. effort)
auxiliary.se	optional data.frame of log-scale standard errors for auxiliary
units.catch, units.index	units recorded in the FLR objects
fleets	return catch by fleet as FLQuants (default: TRUE if more than one catch column)
units	units of the FLQuant(s)

Value

list with catch (FLQuant or FLQuants), indices (FLIndices) and auxiliary (FLQuants with the SEs in attribute "se"; NULL when not given)

Examples

```
## Not run:
data(iccat)
x <- jbdat2flr(iccat$bet$catch, iccat$bet$cpue, iccat$bet$se)
x$indices
d <- jb3_data(x$catch, x$indices, stock = "bet")

## End(Not run)
```

---

jbinput2jabba3 *Convert a JABBA input object to JABBA3*

Description

Translates a JABBA::build_jabba() object into the three JABBA3 inputs: data, priors and model. Priors are carried over on the JABBA scale (r, K, psi -> bkinit, shape, q1, status prior), as are the lag, catch error, catchability and variance groups, the process-error start, the BioProcDyn options (lognormal sigma.proc prior, fixed sigma.proc, depletion-dependent floor, bias correction) and the Selex EB/SB curves.

Usage

```
jbinput2jabba3(jbinput, q.uniform = FALSE, keep.igamma = FALSE, verbose = TRUE)
```

Arguments

jbinput	output of JABBA::build_jabba()
q.uniform	reproduce JABBA's uniform prior on q
keep.igamma	use the diffuse inverse-gamma prior on the process variance instead of the FishLife prior
verbose	print the conversion messages

Details

Differences that cannot be carried over one to one are reported as messages:

- JABBA's inverse-gamma prior on the process variance is replaced by the JABBA3 default (FishLife lognormal prior on sdb), because the inverse-gamma mode collapses towards zero (set keep.igamma = TRUE for the diffuse inverse-gamma(0.001, 0.001) instead).
- JAGS used a uniform prior on q; JABBA3 uses a flat prior on log q (set q.uniform = TRUE to reproduce JABBA exactly).
- Plim (hockey-stick production) is not supported and is dropped.
- Auxiliary data are carried over with their type, SEs and the JABBA observation error settings (auxiliary.sigma -> estimated additional SD, auxiliary.obsE -> aux.se.min); JABBA's auxiliary.lag (mean exploitation over the previous years) has no JABBA3 equivalent and is reported (JABBA3 relates each observation to the same year). JABBA's uniform prior on qA becomes a flat prior on log qA.
- Catch-only models (cpue = NULL) become jb3_data(catch, NULL); the status prior (b.prior) carries over.

Value

list(data, priors, model), e.g. x <- jbinput2jabba3(jb); fit <- jabba3(x$data, x$priors, x$model)

---

jbplot_advice *Advice plot*

Description

Stock status (B/Bmsy, F/Fmsy) and catch with MSY reference points, as FLRef::plotAdvice() and JAGG's jagg_advice(); rel = FALSE for B, F and catch on absolute scales. Nested 90/60% bands by default (nested = FALSE for the 95% interval).

Usage

```
jbplot_advice(x, rel = TRUE, panels = NULL, nested = TRUE, ncol = 1, ...)
```

Arguments

x	a jabba3 fit, a list of fits, or a workflow fit
rel	relative (default) or absolute quantities
panels	quantities to show instead of the default set
nested	nested 90% and 60% bands (JAGG style) instead of the 95% interval
ncol	facet columns
...	passed to jbplot_trj() (e.g. extra, proj, drop.last, probs)

---

jbplot_all *Save the standard plot set*

Description

Writes the plots of a fit to PNG files (data, indices, advice, trajectories, fits, residuals, standardised residuals, runs test, SPiCT diagnostics, Kobe, biplot, prior-posterior, production phase and dynamics, process error; catch error, status priors, PPC (with draws), Selex status, workflow steps and NUTS diagnostics when they apply; retrospective, hindcast, projections and catch risk when given).

Usage

```
jbplot_all(
  fit,
  dir = ".",
  retro = NULL,
  hc = NULL,
  proj = NULL,
  width = 8,
  height = 5.5,
  dpi = 150
)
```

Arguments

fit	a jabba3 fit
dir	output folder
retro, hc, proj	optional outputs of jb3_retro(), jb3_hindcast(), jb3_project()
width, height, dpi	passed to ggsave()

Value

invisibly, the file names

jbplot_auxfits *Fits to the auxiliary data*

Description

Observed auxiliary series (effort, F, F/Fmsy, B/Bmsy, B/K, Z) with the fitted values, their 95% interval and the 95% predictive interval, one facet per series; several fits are coloured by run.

Usage

```
jbplot_auxfits(
  x,
  log = FALSE,
  ppd = TRUE,
  colour = "darkorange3",
  title = NULL
)
```

Arguments

x	a jabba3 fit, a list of fits, or a workflow fit
log	log scale
ppd	draw the predictive interval
colour	interval colour for a single run (grey; projections are coloured by scenario)
title	optional plot title (e.g. to flag hypothetical data)

jbplot_biplot *Stock status biplot*

Description

The 'post-modern' biplot of Quinn and Collie (2005), as JABBA's jbplot_biplot(): B/Bmsy against F/Fmsy with the zones of a hockey-stick harvest control rule (F target ftarget x Fmsy above Bmsy, decreasing to zero at bthreshold x Bmsy; green below the target line, red above F = Fmsy line scaled the same way, yellow in between), the trajectory, and 50/80/95% highest-density contours of the terminal status with the share in each zone.

Usage

```
jbplot_biplot(fit, ftarget = 0.8, bthreshold = 0.2, npoints = 2000)
```

Arguments

fit	a jabba3 fit
ftarget	target F/Fmsy above Bmsy
bthreshold	B/Bmsy at which the target F reaches zero
npoints	draws for the terminal status when the fit has none

---

jbplot_bprior *Status priors: prior against posterior*

Description

As JABBA's jbplot_bprior(), for each status prior given in jb3_priors(status =): the log-normal prior on B/K, B/Bmsy or F/Fmsy in its year against the posterior (draws, or the lognormal approximation at the mode), with the posterior-to-prior mean (PPMR) and variance ratios (PPVR, CV based as in JABBA).

Usage

```
jbplot_bprior(fit)
```

Arguments

fit	a jabba3 fit

Value

a ggplot, or NULL (with a message) when the fit has no status priors

---

jbplot_catch *Catch: observed and estimated*

Description

Catch: observed and estimated

Usage

```
jbplot_catch(fit)
```

Arguments

fit	a jabba3 fit

jbplot_catcherror *Catch error: observed and estimated catch*

Description

As JABBA's `jbplot_catcherror()`: observed catch (points) and estimated catch with its 95% interval, and the log ratio of estimated to observed catch. Without catch error the catch is assumed known and the plot says so.

Usage

```
jbplot_catcherror(fit)
```

Arguments

`fit`	a `jabba3` fit

jbplot_contraction *Plot prior-posterior contraction against shift*

Description

One point per quantity (and run) from `jb3_contraction()`: contraction on the x-axis, absolute shift (in prior SDs) on the y-axis. Shaded regions mark prior-driven quantities (low contraction) and prior-data conflict (large shift).

Usage

```
jbplot_contraction(x, ...)
```

Arguments

`x`	a `jb3contraction` table, or a fit / list of fits passed to `jb3_contraction()`
`...`	passed to `jb3_contraction()` when `x` is a fit

Value

a ggplot

`jbplot_crisk` *Catch risk curves*

Description

As JABBA's `jbplot_Crisk()` (NOAA-style risk plots): the probability that B < Bmsy, that F > Fmsy and that B < `bfrac` x Bmsy (e.g. MSST = (1 - M) Bmsy) against the projected catch (or F/Fmsy) of each scenario, one line per projection year, from `jb3_project()`.

Usage

```
jbplot_crisk(
  proj,
  years = NULL,
  bfrac = 0.5,
  x = c("value", "ratio"),
  ref = NULL
)
```

Arguments

`proj`	output of `jb3_project()` with two or more constant catch (or F) scenarios
`years`	projection years to show (default: up to 4 spread over the horizon)
`bfrac`	also show P(B < bfrac x Bmsy), e.g. 0.5 for a limit or `c(MSST = 1 - M)` (a name labels the panel); NULL to omit
`x`	x-axis: `"value"` (catch or F/Fmsy of the scenario) or `"ratio"`, relative to the status quo catch (or F/Fmsy), i.e. the mean of the last `nsq` assessment years used in the projection, as `type = "ratio"` in JABBA's `fw_jabba()`
`ref`	reference for `x = "ratio"` (default the status quo of the projection)

Examples

```
## Not run:
M <- 0.231
pr <- jb3_project(fit, scenarios = setNames(lapply(seq(0, 3, 0.25), function(m) c(Csq = m)), seq(0, 3, 0.25)),
                  nyears = 31, imp.years = 3)
jbplot_crisk(pr, years = seq(2025, 2050, 5), bfrac = c(MSST = 1 - M), x = "ratio")

## End(Not run)
```

jbplot_data *Data overview*

Description

Data overview

Usage

```
jbplot_data(data, indices = NULL, ...)
```

Arguments

data	a jb3data object, a fit, or catch (FLQuant, FLQuants or data frame) with indices
indices	FLIndices or index data frame when data is catch
...	passed to jb3_data() (e.g. se, stock)

jbplot_ebsb_status *Selex: spawning and exploitable biomass status*

Description

As JABBA's jbplot_ebsb_status() for a Selex fit (jb3_model(ebsb =)): SB/SBmsy (the state, with its 95% interval) and, for each EB/SB curve, EB/EBmsy = (SB/SBmsy) x rho(SB/K) / rho(SBmsy/K).

Usage

```
jbplot_ebsb_status(fit)
```

Arguments

fit	a jabba3 Selex fit

jbplot_ensemble *Ensemble of fits*

Description

Compares several fits (scenarios, sensitivity runs), as JABBA's `jbplot_ensemble()`: trajectories coloured by run with their 95% intervals and, on the right, the Kobe plot with each run's terminal-year uncertainty cloud and its median. With `joint = TRUE` the runs are pooled (equal weight) into one ensemble trajectory with its interval and one Kobe cloud.

Usage

```
jbplot_ensemble(
  x,
  type = c("B/Bmsy", "F/Fmsy", "B/K", "Catch"),
  joint = FALSE,
  kobe = TRUE,
  ndraws = 1000
)
```

Arguments

`x`	a list of `jabba3` fits (named), or a workflow fit
`type`	quantities for the trajectory panels
`joint`	pool the runs into one ensemble
`kobe`	add the Kobe panel (needs patchwork); FALSE returns the trajectory panels only
`ndraws`	draws per run (from the fit's draws, or lognormal at the mode)

jbplot_fits *Fits to the abundance indices*

Description

Observations, fitted index with its 95% interval and the 95% predictive interval (grey), one facet per index. With `years = "all"` the fitted index (q x biomass) is shown over the full assessment period, also in years without observations, as in JABBA's `jbplot_cpuefits()`; intervals then come from posterior draws (the NUTS draws of the fit, or multivariate normal draws at the mode), and the predictive interval uses the median observation SD of each index.

Usage

```
jbplot_fits(
  x,
  log = FALSE,
  ppd = TRUE,
  steps = FALSE,
  colour = "dodgerblue",
  years = c("observed", "all"),
  ndraws = 1000
)
```

Arguments

x	a jabba3 fit, a list of fits, or a workflow fit
log	log scale
ppd	draw the predictive interval
steps	for a workflow fit, plot all steps
colour	interval colour for a single run (grey; projections are coloured by scenario)
years	"observed" (default): years with observations of each index; "all": the full assessment period
ndraws	number of draws for years = "all" when the fit has no draws

---

jbplot_hcr *Harvest control rule plot*

Description

The hockey-stick rule in F/Fmsy against B/Bmsy (as FLRef::plotGFCM()), with Blim, Btrigger and Bmsy, the distribution of B/Bmsy in the evaluation year (the TAC year after the intermediate years) and the advised F.

Usage

```
jbplot_hcr(
  x = NULL,
  ftgt = 1,
  btrigger = 0.5,
  blim = 0.3,
  bclose = 0,
  fmin = 0.001,
  bmax = 2
)
```

Arguments

x	a jb3hcr (from jb3_hcr()); or NULL to draw the rule only
ftgt, btrigger, blim, bclose, fmin	rule settings when x is NULL
bmax	maximum B/Bmsy shown

---

jbplot_hcxval *Hindcast cross-validation plot with MASE*

Description

Hindcast cross-validation plot with MASE

Usage

```
jbplot_hcxval(hc, nyears = 5, adj = FALSE, ncol = NULL, indices = NULL)
```

Arguments

hc	output of jb3_hindcast()
nyears	years before the hindcast horizon to show
adj	also show MASE.adj in the facet labels
ncol	facet columns
indices	optional subset of indices to show (e.g. the dominant ones)

Value

a ggplot; the MASE table in attribute "mase"

---

jbplot_indexcor *Pairwise correlation of abundance indices*

Description

Correlation matrix of the log indices over the years in which each pair overlaps, with p-values (Pearson test) and the number of shared years: a quick check for conflicting indices before fitting a joint index. Pairs with fewer than min.n shared years are left blank.

Usage

```
jbplot_indexcor(x, min.n = 5, alpha = 0.05)
```

Arguments

x	a jb3trend fit, or data.frame(year, index1, ...)
min.n	minimum number of shared years
alpha	significance level marked in the plot

Value

a ggplot; the table (index1, index2, r, p, n) is in attribute "cor"

---

jbplot_indices *Abundance indices with observation error*

Description

As JABBA's jbplot_indices(): each index scaled to its mean, with the 95% interval from the input log-scale standard errors (dodged by index).

Usage

```
jbplot_indices(data)
```

Arguments

data	a jb3data object or a fit

---

jbplot_kobe *Kobe plot*

Description

B/Bmsy against F/Fmsy: trajectory and the joint uncertainty in the last year, from the draws (NUTS or MVN) or, without draws, from the bivariate normal of the terminal log B/Bmsy and log F/Fmsy (delta method), with the percentage in each quadrant.

Usage

```
jbplot_kobe(
  fit,
  npoints = 2000,
  year = NULL,
  contours = c(0.5, 0.8, 0.95),
  style = c("points", "contours")
)
```

Arguments

fit	a jabba3 fit
npoints	number of points in the cloud
year	year of the uncertainty cloud (default: last assessment year), as JAGG's jagg_kobe()
contours	contour lines on the uncertainty points: TRUE/probability levels (default 50/80/95%) or FALSE for points only; highest-density contours of the terminal-year uncertainty (needs MASS; NULL to omit)
style	"points" (draws as points with contour lines) or "contours" (filled 50/80/95% highest-density regions, as jbplot_biplot())

---

jbplot_kobematrix *Plot a Kobe II strategy matrix*

Description

Shaded Kobe II strategy matrices (ICCAT style): probabilities by scenario (rows) and year (columns) for F < Fmsy, B > Bmsy and both, with the probabilities printed in the cells.

Usage

```
jbplot_kobematrix(
  x,
  what = c("F", "B", "green"),
  years = NULL,
  weights = NULL,
  colours = "grey",
  breaks = NULL,
  size = 3,
  pct = TRUE,
  ylab = NULL,
  bref = 1,
  fref = 1,
  risk = FALSE,
  rows = c("auto", "label", "value"),
  risk.breaks = c(0, 5, 10, 20, 30, 40, 50, 100)
)
```

Arguments

x	output of jb3_project() (or a list of them, combined as in jb3_kobematrix()), or of jb3_kobematrix(..., what = "all"), jb3_kobematrix_gfcm() or jb3_kobematrix_iotc(); probability and risk matrices together are shaded separately (needs patchwork)
what	which matrices: any of "F", "B", "green" (default all available)
years	projection years (default all)

weights	weights for combining a list of projections
colours	"grey" (ICCAT report tables), "kobe" (red to green), or a vector of colours, one per probability class
breaks	probability classes in percent (default 0, 50, 60, 70, 80, 90, 100)
size	text size of the cell labels
pct	if TRUE (default) cell labels with a percent sign, as in the ICCAT tables
ylab	row header (default "Catch (t)" for TAC scenarios)
bref, fref, risk, rows	reference points, risk and row labels, passed to jb3_kobematrix() when x is a projection
risk.breaks	risk classes in percent for risk matrices (e.g. P(B < Blim)), shaded in reverse

Value

a ggplot

Examples

```
## Not run:
pc <- jb3_project(fit, catch = seq(1000, 3500, 250), nyears = 10)
jbplot_kobematrix(pc)
jbplot_kobematrix(pc, what = "green", colours = "kobe")
jbplot_kobematrix(pc, bref = c(Btrigger = 0.5))                        # ICES conditions
jbplot_kobematrix(pc, bref = c(Blim = 0.3), fref = NULL, risk = TRUE)  # ICES risk P(B < Blim)

## End(Not run)
```

---

jbplot_noise *Plot observation and process noise of indices*

Description

Bars of the measures of jb3_noise() by index: input observation CV, Francis (2011) loess process CV and, for a trend fit, the estimated additional SD against the common trend.

Usage

```
jbplot_noise(x, se = NULL, span = 0.6)
```

Arguments

x	a jb3trend fit, or a data.frame of indices (year in the first column)
se	data.frame of log-SEs in the layout of x (when x is a data.frame)
span	loess span (Francis 2011 smoother)

Value

a ggplot

jbplot_nuts *NUTS diagnostic plots*

Description

Convergence graphics for a fit with `engine = "nuts"`, on the sampled (unconstrained, mostly log) scale.

trace traces by chain; divergent transitions as rug ticks

rank rank histograms by chain (Vehtari et al. 2021): flat and overlapping when chains mix

dens posterior densities by chain

acf autocorrelation by chain

pairs pairwise scatter with divergent transitions in red (funnels, ridges, non-identifiability)

rhat R-hat and bulk ESS for every sampled parameter, states included

energy marginal energy against energy transitions with E-BFMI by chain

`jbplot_nutsdiags()` combines trace, rank, R-hat/ESS and energy on one page (needs patchwork).

Usage

```
jbplot_nuts(
  fit,
  type = c("trace", "rank", "dens", "acf", "pairs", "rhat", "energy"),
  pars = NULL,
  inc_warmup = FALSE,
  lags = 20,
  npoints = 1500
)

jbplot_trace(fit, pars = NULL, inc_warmup = FALSE)

jbplot_rank(fit, pars = NULL, bins = 20)

jbplot_mcmcdens(fit, pars = NULL)

jbplot_mcmcacf(fit, pars = NULL, lags = 20)

jbplot_pairs(fit, pars = NULL, npoints = 1500)

jbplot_rhat(fit)

jbplot_energy(fit)

jbplot_nutsdiags(fit, pars = NULL)
```

Arguments

fit	a jabba3 fit with engine = "nuts"
type	plot type, see Details
pars	parameter names (e.g. c("logK", "logr"); "logq" matches all logq[i]). Default: all fixed effects
inc_warmup	include warmup iterations (trace only)
lags	maximum lag for acf
npoints	maximum points per panel for pairs
bins	number of rank bins

Value

a ggplot

Examples

```
## Not run:
fn <- jabba3(d, pr, jb3_model(shape = "fox"), engine = "nuts", nuts = list(chains = 3, iter = 2000, cores = 3))
jb3_nuts_diags(fn)$sampler
jbplot_nuts(fn, "trace"); jbplot_nuts(fn, "pairs", pars = c("logK", "logr", "logsdb"))
jbplot_nutsdiags(fn)

## End(Not run)
```

---

jbplot_ppc *Posterior predictive check*

Description

As JABBA's jbplot_PPC(): for each draw, the discrepancy D = sum(((log I - log I.hat)/sd)^2) of the observed indices against that of indices replicated from the observation model; points on either side of the 1:1 line, and the Bayesian p-value P(D.rep > D.obs) (near 0.5 under a well-specified model, extreme values < 0.05 or > 0.95 flag misfit). Uses the NUTS draws, or MVN draws at the mode (generated if the fit has none).

Usage

```
jbplot_ppc(fit, joint = FALSE, ndraws = 1000, npoints = 1000)
```

Arguments

fit	a jabba3 fit
joint	pool all indices into one check
ndraws	number of MVN draws when the fit has none
npoints	maximum points per panel

Value

a ggplot; the Bayesian p-values in attribute "ppc"

`jbplot_priorpost` *Prior and posterior distributions*

Description

Prior densities (lognormal; flat priors not drawn) against the posterior (draws, or the lognormal approximation at the mode), with the posterior-to-prior variance ratio (PPVR, log scale) after JABBA's `jbplot_ppdist()`. K is shown on a log axis.

Usage

```
jbplot_priorpost(fit, pars = c("r", "K", "bkinit", "sdb", "m", "kdyn"))
```

Arguments

`fit`	a `jabba3` fit
`pars`	parameters to show

`jbplot_priorpost_trj` *Prior predictive against posterior*

Description

Compares a prior predictive run (`jabba3(..., prior.only = TRUE)`: catch and priors only, the data likelihoods switched off) with a fit to data: the trajectories (median and 95% interval) and the marginal densities of selected parameters and of terminal-year status. Shows how much the data (indices, auxiliary series) move the assessment away from what the priors and the catch alone imply (the prior predictive model of Kell et al. 2025). Both runs are best fitted with NUTS; fits without draws use multivariate normal draws at the mode.

Usage

```
jbplot_priorpost_trj(
  prior,
  posterior = NULL,
  type = c("B/Bmsy", "F/Fmsy"),
  pars = c("r", "K", "B/Bmsy"),
  labels = NULL,
  ndraws = 1000
)
```

Arguments

prior	the prior predictive run (or any reference run), or a named list of two runs (e.g. `list(Prior = fp, Posterior = fd)`), in which case `posterior` is ignored
posterior	the fit to data
type	trajectories to show, any of "B/Bmsy", "F/Fmsy", "B/K", "B", "F"
pars	marginal densities: any of "r", "K", "MSY", "Bmsy", "Fmsy", "bkinit", "sdb" and the terminal-year "B/Bmsy", "F/Fmsy", "B/K"; `NULL` for trajectories only
labels	names of the two runs (default from the list names, or "Prior" and "Posterior")
ndraws	draws used for fits without NUTS draws

Value

a patchwork (trajectories above, densities below)

Examples

```
## Not run:
fp <- jabba3(d0, pr0, jb3_model(shape = "fox"), prior.only = TRUE)
fd <- jabba3(d0, pr0, jb3_model(shape = "fox"))
jbplot_priorpost_trj(fp, fd)
jbplot_trj(list(Prior = fp, Posterior = fd), type = c("B/Bmsy", "F/Fmsy"))  # trajectories only

## End(Not run)
```

---

jbplot_procdev *Process error deviations and process SD*

Description

Process error deviations and process SD

Usage

```
jbplot_procdev(x, steps = FALSE)
```

Arguments

x	a `jabba3` fit, a list of fits, or a workflow fit
steps	for a workflow fit, plot all steps

`jbplot_projection` *Projection plot*

Description

Trajectories of the last assessment years (grey: median and 95% interval) and of each projection scenario (coloured: median and 95% interval), from `jb3_project()` or `jb3_fwd()`. Dotted vertical lines mark the last assessment year and, with intermediate years, the last intermediate year before the TAC year.

Usage

```
jbplot_projection(
  proj,
  type = c("B/Bmsy", "F/Fmsy", "Catch"),
  nyears = 15,
  refpts = TRUE,
  bthr = 0.5,
  blim = 0.3,
  extra = NULL,
  label.size = 2.5
)
```

Arguments

`proj`	output of `jb3_project()` or `jb3_fwd()`
`type`	panels: any of "B/Bmsy", "F/Fmsy", "Catch", "B", "F"
`nyears`	number of assessment years shown before the projection
`refpts`	draw the reference points (FALSE for none)
`bthr`	biomass threshold as a fraction of Bmsy (default 0.5, orange), optionally named (Bthr, Bpa, MSST, ...); `NULL` to omit
`blim`	biomass limit as a fraction of Bmsy (default 0.3, red), optionally named; `NULL` to omit
`extra`	further reference points as a named vector relative to Bmsy (names starting with "B") or Fmsy (names starting with "F"), e.g. `c(Flim = 1.3)`; names Blim/Flim are drawn red, Bpa/Fpa/Bthr/MSST orange, others green
`label.size`	size of the reference point labels (0 for lines without labels)

Details

Reference points are drawn as labelled dashed lines: Bmsy and Fmsy (dark green) on the B/Bmsy and F/Fmsy panels, a biomass threshold `bthr` (orange) and limit `blim` (red) on the B/Bmsy panel, and MSY on the catch panel. A name sets the label, e.g. `bthr = c(Bpa = 0.5)` (ICES), `bthr = c(Bthr = 0.5)` (GFCM) or `bthr = c(MSST = 1 - M)` (U.S. minimum stock size threshold); further lines can be added with `extra`.

Value

a ggplot

Examples

```
## Not run:
pj <- jb3_project(fit, fmsy = c(0, 0.5, 0.75, 1, 1.25))
jbplot_projection(pj)                                    # Bthr = 0.5, Blim = 0.3 Bmsy
jbplot_projection(pj, bthr = c(Bpa = 0.5), blim = c(Blim = 0.3))
jbplot_projection(pj, bthr = c(MSST = 1 - 0.2), blim = NULL)  # U.S.: MSST = (1 - M) Bmsy
jbplot_projection(pj, extra = c(Flim = 1.3))

## End(Not run)
```

---

jbplot_q *Distribution of catchability relative to the reference index*

Description

Density of q relative to the first (reference) index for each other index of a relative trend fit, with the reference as a vertical line at 1: the lognormal approximation from the sdreport, or the NUTS draws for `engine = "nuts"`. Narrow distributions mean the indices overlap enough to be scaled against each other; wide distributions mean the scaling relies on the continuity of the trend.

Usage

```
jbplot_q(fit, log = TRUE)
```

Arguments

`fit`	a relative `jb3trend` fit
`log`	plot q on a log axis

Value

a ggplot

jbplot_residuals *Index residuals by year*

Description

As JABBA's `jbplot_residuals()`: residuals of all indices coloured by index, a boxplot per year, a loess smoother and the RMSE; OSA residuals (N(0,1) under the model) by default, or ordinary log residuals.

Usage

```
jbplot_residuals(fit, type = c("osa", "ordinary"))
```

Arguments

`fit`	a `jabba3` fit
`type`	"osa" (default) or "ordinary"

jbplot_retro *Retrospective plot with Mohn's rho*

Description

Retrospective plot with Mohn's rho

Usage

```
jbplot_retro(retro, type = c("B/Bmsy", "F/Fmsy", "B", "F"), ncol = NULL)
```

Arguments

`retro`	output of `jb3_retro()`
`type`	quantities
`ncol`	facet columns

Value

a ggplot; Mohn's rho in attribute "rho"

jbplot_runs *Runs test*

Description

Residuals as segments from zero, three-sigma limits (green: runs test passed, p >= 0.05; red: failed), outliers in red, after `ggplotFL::plotRunstest()` and JABBA's `jbplot_runstest()`. OSA residuals by default.

Usage

```
jbplot_runs(
  fit,
  type = c("osa", "ordinary"),
  mixing = c("less", "two.sided", "greater")
)
```

Arguments

`fit`	a `jabba3` fit
`type`	"osa" (default) or "ordinary"
`mixing`	alternative of the runs test: "less" (default, too few runs), "two.sided" or "greater"

Value

a ggplot; the runs test table is in attribute "runs" (as JAGG's `jagg_runstest()`)

jbplot_selex *Selectivity and EB/SB curves*

Description

Left: selectivity by fleet with the maturity ogive (dashed), at length (default) or at age. Right: the equilibrium ratio of exploitable to spawning biomass against spawning-biomass depletion (points) with the fitted Selex curve (lines), as used by `jb3_model(ebsb = )`.

Usage

```
jbplot_selex(x, at = c("length", "age"))
```

Arguments

`x`	output of `jb3_selex()`
`at`	"length" or "age"

Value

a patchwork (or list of ggplots)

---

jbplot_spdiags *SPiCT-style residual diagnostics*

Description

Per index, as `spict::plotspict.diagnostic()`: OSA residuals over time (bias t-test), autocorrelation function (Ljung-Box test) and normal QQ plot (Shapiro-Wilk test), p-values in the panel titles, in red when < 0.05.

Usage

```
jbplot_spdiags(fit)
```

Arguments

`fit`	a `jabba3` fit (with OSA residuals)

Value

a patchwork of ggplots (or a list without the patchwork package)

---

jbplot_spphase *Surplus production phase and dynamics plots*

Description

`jbplot_spphase()`: the surplus production curve with the catch trajectory (yield against biomass) on the Kobe colours mapped to production space (as `JABBA::jbplot_spphase()`): green B > Bmsy and F < Fmsy (below the line C = Fmsy B), orange B > Bmsy and F > Fmsy, yellow B < Bmsy and F < Fmsy, red B < Bmsy and F > Fmsy. The hatched band below Bmsy marks catches between Fmsy B and the production curve, where F < Fmsy and the stock rebuilds. Blue band: MSY 95% interval.

Usage

```
jbplot_spphase(fit, rel = FALSE)

jbplot_spdyn(fit, rel = FALSE)

jbplot_production(fit, type = c("phase", "dyn"), rel = FALSE)
```

Arguments

fit	a jabba3 fit
rel	relative axes: B/K and catch (or SP) / MSY
type	"phase" (jbplot_spphase) or "dyn" (jbplot_spdyn)

Details

jbplot_spdyn(): the realised annual surplus production SP_t = B_t+1 - B_t + C_t against B_t, with the expected production curve (as JABBA::jbplot_spdyn()); departures from the curve are the process deviations (and the lag) in production units.

Value

a ggplot

---

jbplot_stdresiduals *Standardised residuals*

Description

As JABBA's jbplot_stdresiduals(): standardised log residuals (residual / total observation SD) of all indices as segments from zero, a boxplot per year, a loess smoother through the yearly means, and SDNR with its 95% critical value (chi-square; SDNR above it suggests the observation error is underestimated).

Usage

```
jbplot_stdresiduals(fit)
```

Arguments

fit	a jabba3 fit

---

jbplot_steps *Workflow comparison*

Description

Compares the steps of a workflow fit (or any list of fits): B/Bmsy and F/Fmsy trajectories, and a table of the step settings and terminal status in the subtitle.

Usage

```
jbplot_steps(x)
```

Arguments

x	workflow fit or list of fits

jbplot_summary *Summary of one or several fits*

Description

As JABBA's `jbplot_summary()`: B, F, B/Bmsy, F/Fmsy and B/K trajectories with 95% intervals, and the surplus production curves with Bmsy and MSY marked, coloured by run.

Usage

```
jbplot_summary(x, steps = FALSE, kobe = TRUE, ndraws = 1000)
```

Arguments

`x`	a `jabba3` fit, a list of fits, or a workflow fit
`steps`	for a workflow fit, show all steps
`kobe`	add the Kobe panel with the terminal-year status (FALSE: trajectories and production curves only)
`ndraws`	points per run in the Kobe cloud

jbplot_trend *Trend plots*

Description

`jbplot_trend()`: relative fits, the q-scaled observed indices (by index) with the state-space trend and its 95% interval; census fits, the area states with the total abundance. `jbplot_jointindex()`: the joint index with 95% intervals and the state-space trend (red dashed), as JARA's `jrplot_jointindex()`. `jbplot_catchfill()`: observed and reconstructed catch.

Usage

```
jbplot_trend(fit)

jbplot_jointindex(fit)

jbplot_catchfill(fit)
```

Arguments

`fit`	a `jb3trend` fit (or `jb3_catchfill()` output for `jbplot_catchfill`)

Value

a ggplot

jbplot_trj *Trajectories with credible intervals*

Description

One facet per quantity, the estimate (mode, or posterior median with NUTS) with its 95% interval, reference points as labelled lines (FLRef::plotAdvice style). Several fits (a list, or the steps of a workflow fit) are coloured by run. With `nested = TRUE` the plot follows JAGG's `jagg_advice()`: a 90% (alpha 0.2) and a 60% band (alpha 0.4) around the median, from the draws when the fit has them, otherwise from the lognormal (delta-method) approximation. Projections from `jb3_project()` are appended after the last year (one colour per scenario, with intervals).

Usage

```
jbplot_trj(
  x,
  type = c("B", "F", "B/Bmsy", "F/Fmsy", "Catch", "Process deviation"),
  steps = FALSE,
  refpts = TRUE,
  extra = NULL,
  ncol = NULL,
  colour = "grey45",
  nested = FALSE,
  probs = c(0.05, 0.2, 0.5, 0.8, 0.95),
  drop.last = FALSE,
  proj = NULL,
  label.size = 2.5,
  bthr = 0.5,
  blim = 0.3
)
```

Arguments

`x`	a `jabba3` fit, a list of fits, or a workflow fit
`type`	quantities: any of "B", "F", "B/Bmsy", "F/Fmsy", "B/K" (alias "B/B0"), "Catch", "Catch/MSY", "Process deviation", "Process SD", "Surplus production", "EB"
`steps`	for a workflow fit, plot all steps
`refpts`	draw reference points (with several runs: relative quantities only)
`extra`	named vector of extra reference points (names starting with B go to the biomass panel, F to the F panel, on the scale of the panels plotted, e.g. `c(Blim = 0.5, Bpa = 0.8)` with B/Bmsy)
`ncol`	facet columns
`colour`	interval colour for a single run (grey; projections are coloured by scenario)
`nested`	nested 90% and 60% bands (JAGG style) instead of the 95% interval
`probs`	five probabilities for `nested`: outer and inner bands and the median

drop.last	drop the last year of F, F/Fmsy and catch (as JAGG's osa = TRUE)
proj	optional output of jb3_project(), appended after the last year
label.size	size of the reference point labels
bthr, blim	biomass threshold and limit as fractions of Bmsy (defaults 0.5 and 0.3, the SPiCT benchmark defaults of WKBMSYSPiCT); drawn on the B/Bmsy panel (and the B panel for a single run); NULL to omit. With several runs, reference points are drawn only on relative quantities (B/Bmsy, F/Fmsy, and Bmsy/K on B/K when it is the same in all runs): absolute levels (K, Bmsy, Fmsy, MSY) differ between runs.

---

scale_x_year *Integer year axis*

Description

Integer year axis

Usage

```
scale_x_year(n = 5, ...)
```

Arguments

n	approximate number of breaks
...	passed to scale_x_continuous()

---

spictinp2jabba3 *Convert a SPiCT input list to JABBA3*

Description

Translates a SPiCT input list (inp, as passed to spict::fit.spict()) into JABBA3 data, priors and model. SPiCT works in continuous time; JABBA3 is annual, so catches are summed and index observations within a year averaged (geometric mean) by calendar year (floor(time)). The SPiCT observation-error form is used (obs.error = "spict"), with stdevfacI as relative year weights.

Usage

```
spictinp2jabba3(inp, stock = "stock", verbose = TRUE)
```

Arguments

inp	SPiCT input list
stock	stock name
verbose	print conversion messages

Details

Priors: `logr`, `logK`, `logbkfrac` (-> bkinit), `logsdb` (-> sdb), `logn` (-> shape m), `logsdi` (-> sdi) are carried over with their on/off switches (prior vectors `c(mean, sd[, on])` on the log scale). A fixed n (`phases$logn = -1`) fixes the shape. SPiCT's r and JABBA's r coincide for the Pella-Tomlinson form (MSY = rK / n^(n/(n-1))). The ratio priors `logalpha`, `logbeta` and the F process have no JABBA3 equivalent and are reported. Catch error is on with `sdc` from the `logsdc` prior or initial value.

Value

list(data, priors, model)

`theme_jabba3` *JABBA3 ggplot theme*

Description

`theme_bw()` with the compact legend of `FLRef::plotAdvice()` (as JAGG's `theme_jagg()`).

Usage

```
theme_jabba3(base_size = 11)
```

Arguments

`base_size`	base font size

Index

∗ **datasets**
    iccat, 4
    jara_examples, 8

flr2jbdat, 4

iccat, 4

jabba3, 5, *13*, *52*
jabba3_info, 6
jabbaplot3, 7
jara_examples, 8
jb3_align, 9, *15*
jb3_bprior, 9, *39*
jb3_catchfill, 10, *14, 15*
jb3_contraction, 11, *59*
jb3_convergence, 13
jb3_data, *5*, *11*, 13, *25*, *53*
jb3_data_jabba, *13*, 15
jb3_diags, 16
jb3_draws, 17
jb3_ensemble, 17, *21*, *40*
jb3_fill_table, 18
jb3_flpar *(*jb3_flqs*)*, 18
jb3_flqs, 18
jb3_FLStockR, 19
jb3_fmsy2r, 20
jb3_fwd, 21, *23*, *26*, *28*, *40*, *72*
jb3_fwdtable, 22
jb3_hcr, *21*, 23
jb3_hindcast, 24, *51*, *64*
jb3_jointindex, 25
jb3_kobematrix, 26, *30*, *66, 67*
jb3_kobematrix_gfcm, 27, *66*
jb3_kobematrix_ices *(*jb3_kobematrix_gfcm*)*, 27
jb3_kobematrix_iotc, *27*, 29, *66*
jb3_leslie, 30
jb3_model, *5*, *14*, 32, *39*, *46*
jb3_noise, 35, *67*
jb3_nuts_diags, 36
jb3_nuts_ess, 36
jb3_osa, 37
jb3_partable, 38
jb3_priors, *5*, 38
jb3_project, *21*, *26*, *28*, *30*, 40, *42*, *60*, *66*, *72*, *79, 80*
jb3_projtable, *27*, 42
jb3_qtable, 43
jb3_r2fmsy *(*jb3_fmsy2r*)*, 20
jb3_retro, 43, *51*, *74*
jb3_rho_gt, 44
jb3_rho_table *(*jb3_rho_gt*)*, 44
jb3_runs, 45
jb3_sel5, 45, *46*
jb3_selex, *33*, 46, *75*
jb3_splitindex, 47
jb3_tier1 *(*jb3_workflow*)*, 52
jb3_tier2 *(*jb3_workflow*)*, 52
jb3_trend, *43*, 48
jb3_tstable, 50
jb3_weights, *17*, 51
jb3_workflow, *6*, 52
jb3cols, 53
jbcatch2flq *(*jbdat2flr*)*, 53
jbdat2flr, *4*, 53
jbidx2FLIndices *(*jbdat2flr*)*, 53
jbinput2jabba3, 54
jbplot_advice, 55
jbplot_all, 56
jbplot_auxfits, 57
jbplot_biplot, 57
jbplot_bprior, 58
jbplot_catch, 58
jbplot_catcherror, 59
jbplot_catchfill *(*jbplot_trend*)*, 78
jbplot_contraction, 59
jbplot_crisk, 60
jbplot_data, 61

jbplot_ebsb_status, 61
jbplot_energy *(jbplot_nuts)*, 68
jbplot_ensemble, 62
jbplot_fits, 62
jbplot_hcr, 63
jbplot_hcxval, 64
jbplot_indexcor, 64
jbplot_indices, 65
jbplot_jointindex *(jbplot_trend)*, 78
jbplot_kobe, 65
jbplot_kobematrix, *27*, *29*, *30*, 66
jbplot_mcmcacf *(jbplot_nuts)*, 68
jbplot_mcmcdens *(jbplot_nuts)*, 68
jbplot_noise, 67
jbplot_nuts, 68
jbplot_nutsdiags *(jbplot_nuts)*, 68
jbplot_pairs *(jbplot_nuts)*, 68
jbplot_ppc, 69
jbplot_priorpost, 70
jbplot_priorpost_trj, 70
jbplot_procdev, 71
jbplot_production *(jbplot_spphase)*, 76
jbplot_projection, 72
jbplot_q, 73
jbplot_rank *(jbplot_nuts)*, 68
jbplot_residuals, 74
jbplot_retro, 74
jbplot_rhat *(jbplot_nuts)*, 68
jbplot_runs, 75
jbplot_selex, 75
jbplot_spdiags, 76
jbplot_spdyn *(jbplot_spphase)*, 76
jbplot_spphase, 76
jbplot_stdresiduals, 77
jbplot_steps, 77
jbplot_summary, 78
jbplot_trace *(jbplot_nuts)*, 68
jbplot_trend, 78
jbplot_trj, *8*, 79

plot.jabba3 *(jabbaplot3)*, 7

scale_x_year, 80
spictinp2jabba3, 80

theme_jabba3, 81