Package: FLOSS (via r-universe)

November 5, 2026

Type Package

Title FL Operating Model and Sampling Simulator

Version 0.5.8

Description Builds multi-fleet, multi-species age-structured operating models from life-history parameters, fleet selectivity, relative catchability and a target MSY, driven by shared fleet effort (with optional effort creep), and generates the data such fisheries produce (catch, effort, CPUE, survey indices, length frequencies and biological samples). Designed for training and for trainers who need realistic example datasets with a known truth. Built on FLR ('FLCore', 'FLBRP', 'FLasher', 'FLife') and 'FLRef'.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.1), FLCore

Imports methods, stats, utils, FLBRP, FLasher, FLife, FLRef, ggplot2, yaml

Suggests testthat (>= 3.0.0), knitr, rmarkdown, writexl, shiny, shinydashboard, commonmark

Additional_repositories https://flr.r-universe.dev

URL https://henning-winker.github.io/FLOSS/, https://github.com/Henning-Winker/FLOSS

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

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/testthat/edition 3

VignetteBuilder knitr

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

Date/Publication 2026-11-04 09:23:13 UTC

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

RemoteRef HEAD

RemoteSha c2c92c17abb8920f9166a918bdedfab54c74e285

Contents

alk_obs . . . 3
as_flr . . . 4
build_om . . . 5
build_oms . . . 6
caa_obs . . . 7
calibrate_msy . . . 8
catch_curve . . . 8
catch_curve_length . . . 9
catch_fleet . . . 10
catch_table . . . 11
cpue_check . . . 12
effort_sim . . . 12
effort_trend . . . 13
export_data . . . 14
export_truth . . . 15
fishery . . . 15
fit_growth . . . 17
fit_lw . . . 17
fit_maturity . . . 18
floss.r4col . . . 19
floss_gears . . . 19
floss_guide . . . 20
floss_tip . . . 21
length_indicators . . . 21
lfd_aggregate . . . 22
lfd_annual . . . 23
load_fishery . . . 24
M_empirical . . . 24
oem_settings . . . 25
one_gear_fishery . . . 27
one_gear_species . . . 28
planted_artefacts . . . 28
plot_aggregate . . . 29
plot_availability . . . 30
plot_bio . . . 30
plot_caa . . . 31
plot_effort . . . 32
plot_eq . . . 32
plot_eq_selectivity . . . 33
plot_fit . . . 34
plot_fleets . . . 34

plot_indicators . . . 35
plot_kobe . . . 35
plot_length_indicators . . . 36
plot_lfd . . . 37
plot_obs . . . 38
plot_om . . . 39
plot_seasons . . . 39
plot_srr . . . 40
plot_ssb_vb . . . 40
plot_treatment . . . 41
plot_truth . . . 41
read_dataset . . . 42
reef_fishery . . . 42
reef_oem . . . 43
run_app . . . 43
run_treatment . . . 44
sample_data . . . 45
season_design . . . 46
selectivity . . . 47
simulate_fishery . . . 48
species . . . 49
species_lhpar . . . 50
species_library . . . 50
species_sel . . . 51
status_table . . . 52
survey_settings . . . 52
treatment . . . 53
trend_values . . . 54
write_spec . . . 55

Index **56**

`alk_obs` *Observed age-length key*

Description

Counts aged fish by length bin and age, from the biological samples.

Usage

```
alk_obs(
  data,
  species,
  iter = 1,
  source = NULL,
  years = NULL,
  bin_mm = NULL,
```

```
  prop = FALSE
)
```

Arguments

data	a floss_data (or a list with a bio and lfd data frame, e.g. from read_dataset()).
species	species name.
iter	iteration(s); several are pooled.
source	sources (fleets / survey) to pool; default all.
years	years to pool; default all.
bin_mm	length bin (mm); default the bin of the length samples.
prop	return $P(a \mid l)$ (rows sum to one) instead of counts.

Value

a matrix, length bins (lower limit, mm) x ages.

---

as_flr *FLR objects for JABBA3 and FLicc*

Description

Converts the observations of one species and iteration into FLR objects:

Usage

```
as_flr(data, species = NULL, iter = 1, all = FALSE)
```

Arguments

data	a floss_data.
species	species name; default the first. NULL with all = TRUE returns a list over species.
iter	one iteration.
all	return a named list with one element per species.

Details

- catch: FLQuants, observed catch (t) by fleet (age "all"), and catch_total (FLQuant, all fleets), as jb3_data() of JABBA3 takes them;
- indices: FLIndices of FLIndexBiomass, one per fleet CPUE and the survey, with the log-scale standard error (from the sampling CV) in index.var, as JABBA3 expects;
- effort: FLQuants of effort by fleet (fishing days), for JABBA3's auxiliary data;
- lfd: FLQuants, one per gear (fleets and survey), length (lower bin limit, cm) x year, counts summed over seasons, as FLicc's fiticc() and stocklen() take them; lfd_raised the same raised to the catch (lfd_aggregate());

- `lhpar`: `FLPar` with `linf`, `k`, `t0`, `M`, `L50`, `a`, `b`, `tmax` and `s` (steepness), with `a` converted so that weight is in kg (FLicc convention; FLOSS uses g);
- `catch_by_gear`: share of the catch by fleet over the years with length samples (FLicc's `catch_by_gear`).

Value

a list of class `floss_flr` (or a list of them).

Examples

```
x <- simulate_fishery("lutjanus_bohar", nsim = 2, iters = 1)
f <- as_flr(x$data)
f$indices
# JABBA3: jb3_data(f$catch, f$indices, auxiliary = f$effort)
# FLicc:  stocklen(f$lfd[c("gillnet", "linefish")], f$lhpar)
```

`build_om` *Build a stochastic operating model for one species*

Description

Apical fishing mortality of fleet f is $F_{f,y} = q_f \, \eta_{f,y} \, E_{f,y} \, \delta_{f,y}$ with $q_f = relq_f \times F_{MSY}$, nominal effort E and efficiency η from `effort_sim()` (shared by all species), and optional species-specific catchability noise δ (`sigmaQ`). Fleet F-at-age is apical F times the fleet's selectivity-at-age; total F-at-age is the sum over fleets. The stock is projected with `ffwd()` under Fbar control and lognormal (optionally AR1) recruitment deviations. The first two years are dropped as burn-in, as in the reef example.

Usage

```
build_om(
  sp,
  fishery,
  nsim = 100,
  effort = NULL,
  sigmaQ = 0,
  ar1 = FALSE,
  seed = fishery$seed,
  treatment = NULL
)
```

Arguments

sp	a species().
fishery	a fishery().
nsim	number of iterations.
effort	optional result of effort_sim() for this fishery (pass it when building several species so they share one effort realisation; by default it is regenerated from the fishery seed, which gives the same).
sigmaQ	lognormal SD of species-specific catchability noise.
ar1	use the species' rho for AR1 recruitment deviations.
seed	base seed for recruitment (and sigmaQ) deviations; combined with the species id so species differ but are reproducible.
treatment	optional treatment() (changes to effort, recruitment, selectivity, creep or M); random numbers are unchanged, so a treated and an untreated operating model differ only by the treatment.

Details

Reference points are recomputed for the realised exploitation pattern (median F-at-age over iterations, last years) with FLRef::computeFbrp() and stored in the returned FLStockR.

Value

an object of class floss_om: a list with stock (FLStockR), sr (FLSR), brp (calibrated equilibrium), sel (fleet selectivities), fapic and f_fleet (FLQuants of apical F and F-at-age by fleet), effort, efficiency, species, fishery and settings.

---

build_oms *Build operating models for several species sharing one fishery*

Description

All species see the same effort realisation; recruitment deviations differ by species. Pass oms to reuse operating models already built for the same fishery (matched by species id and fishery hash), so adding a species does not rebuild the others.

Usage

```
build_oms(species_list, fishery, nsim = 100, oms = NULL, ...)
```

Arguments

species_list	a list of species() (or species ids from the library).
fishery	a fishery().
nsim	number of iterations.
oms	optional named list of existing floss_om objects to reuse.
...	passed to build_om().

Value

a named list of class `floss_oms`.

`caa_obs` *Catch-at-age from length samples and an age-length key*

Description

Applies an observed age-length key to the length samples of each year, $\hat{n}_{a,y} = \sum_l n_{l,y} P(a \mid l)$. Length bins without aged fish cannot be assigned an age and are dropped (their share is reported). With `raise = "catch"` the numbers are raised so that their weight ($W = aL^b$, g) equals the observed catch of the source (t).

Usage

```
caa_obs(
  data,
  species,
  source,
  iter = 1,
  years = NULL,
  alk = c("pooled", "annual"),
  raise = c("none", "catch")
)
```

Arguments

`data`	a `floss_data` (or a list with a `bio` and `lfd` data frame, e.g. from `read_dataset()`).
`species`	species name.
`source`	one fleet (or `"survey"`).
`iter`	iteration(s); several are pooled.
`years`	years to pool; default all.
`alk`	`"pooled"` (one key over all years of the selection) or `"annual"` (a key per year).
`raise`	`"none"` (sample numbers) or `"catch"`.

Value

a data frame with `species`, `iter`, `year`, `source`, `age`, `n`, and attribute `unassigned` (share of measured fish per year in length bins without aged fish).

calibrate_msy *Equilibrium calibrated to a target MSY*

Description

Builds the FLife equilibrium (`lhEql`, ages `0:tmax`, plus group `tmax`, constant M), applies the joint selectivity-at-age (fleet selectivities weighted by relative catchability, `FLRef::combine_sel_age()`) and rescales virgin biomass so that equilibrium MSY equals `sp$msy` (two passes of `FLRef::updsr()`).

Usage

```
calibrate_msy(sp, sels = NULL)
```

Arguments

`sp`	a `species()`.
`sels`	optional result of `species_sel()`.

Value

an `FLBRP` with reference points.

catch_curve *Age-based catch curve*

Description

Regression of $\log n_a$ on age over the descending limb of the catch-at-age (pooled over the selected years); total mortality $Z = -$slope. By default the fit starts one age after the mode and drops the last (plus-group) age.

Usage

```
catch_curve(
  x,
  species = NULL,
  source = NULL,
  iter = 1,
  years = NULL,
  ages = NULL,
  om = NULL
)
```

Arguments

x	catch-at-age from caa_obs(), or a floss_data (then caa_obs() is called with species, source, iter, years).
species	species name.
source	fleet for caa_obs().
iter	iteration(s) (pooled).
years	years to use; default all.
ages	ages to include in the regression (default: descending limb).
om	optional floss_om of the species: adds the true mean Z over the fitted ages and years.

Value

a floss_fit with pars (Z) and, when the true M is known, F = Z - M.

---

catch_curve_length *Length-converted catch curve and selectivity*

Description

Pauly's length-converted catch curve on the pooled length samples of one source, using growth parameters (e.g. from fit_growth()): relative age $t(L) = t_0 - \log(1 - L/L_\infty)/k$, time to grow through a bin Δt, and $\log(n/\Delta t)$ against t. Z is minus the slope of the descending limb. Selectivity on the ascending limb is the ratio of observed to expected numbers (the descending line extrapolated back), fitted with a logistic curve.

Usage

```
catch_curve_length(
  data,
  species,
  source,
  linf,
  k,
  t0 = 0,
  iter = 1,
  years = NULL,
  lmax_prop = 0.95
)
```

Arguments

`data`	a `floss_data` or a `read_dataset()` list.
`species`	species name.
`source`	one fleet (or `"survey"`).
`linf`, `k`, `t0`	growth parameters (cm, per year).
`iter`	iteration(s) (pooled).
`years`	years to use; default all.
`lmax_prop`	bins above `lmax_prop` x Linf are excluded.

Value

a `floss_fit` with `pars` (`Z`, `L50`, `L95`).

`catch_fleet` *True catch by fleet*

Description

Partial Baranov catch of each fleet from fleet F-at-age, total Z and numbers-at-age of the operating model: $C_{f,a,y} = N_{a,y}\, F_{f,a,y}/Z_{a,y}\,(1 - e^{-Z_{a,y}})$.

Usage

```
catch_fleet(om, type = c("wt", "n"))
```

Arguments

`om`	a `floss_om`.
`type`	`"wt"` (catch weight, t) or `"n"` (catch-at-age, numbers).

Value

`FLQuants` by fleet.

`catch_table` *Catch, effort and nominal CPUE: by fleet or aggregated*

Description

- `catch_table()`: observed catch (t) by fleet, summed over fleets (`"aggregate"`), or summed over fleets and species (`"combined"`).
- `effort_table()`: observed effort (fishing days) by fleet or summed over fleets.
- `nominal_cpue()`: catch / effort (kg per fishing day) at the chosen level, and as an index relative to its mean. `"aggregate"` divides the catch of all fleets by the effort of all fleets, ignoring that a day of trap fishing and a day of linefishing catch differently; `"combined"` also sums the catch of all species.

Usage

```
catch_table(data, species = NULL, by = c("fleet", "aggregate", "combined"))

effort_table(data, by = c("fleet", "aggregate"))

nominal_cpue(data, species = NULL, by = c("fleet", "aggregate", "combined"))
```

Arguments

`data`	a `floss_data`.
`species`	species names; default all (for `"combined"`, the species summed).
`by`	`"fleet"`, `"aggregate"` or `"combined"` (catch, CPUE).

Details

Nominal CPUE uses all years with catch and effort, unlike the CPUE index of `sample_data()` (which is a standardised index for the sampled years). With changing fleet mix or effort creep, nominal CPUE drifts away from the stock: compare it with `cpue_true` (true catch / true effort) and the true biomass.

Value

a data frame with `species`, `iter`, `year`, `source` (fleet or `"all fleets"`) and the values (observed and true).

Examples

```
x <- simulate_fishery(nsim = 2, iters = 1)
head(nominal_cpue(x$data, by = "aggregate"))
plot_aggregate(x$data, "Lutjanus bohar", what = "cpue", by = "fleet")
```

cpue_check *Check the CPUE assumptions against the truth (trainers)*

Description

CPUE is assumed proportional to vulnerable biomass, $C/E = qB$. Fits $\log I = \log q + \beta \log B$ ($\beta = 1$: proportional; < 1: hyperstable; > 1: hyperdepletion) and the trend in $\log(I/B^{\beta})$ over years (catchability drift, e.g. effort creep). Needs the answer key (`index_true`), so it works on a `floss_data`.

Usage

```
cpue_check(data, species, source, iter = 1)
```

Arguments

`data`	a `floss_data` or a `read_dataset()` list.
`species`	species name.
`source`	fleet.
`iter`	iteration(s) (pooled).

Value

a `floss_fit` with `pars` (`beta`, `q_trend_pct` per year, `r`).

effort_sim *Simulate fleet effort*

Description

Nominal effort $E_{f,y}$ is the trend times multiplicative lognormal noise with mean one, $\exp(\epsilon - \sigma_E^2/2)$, $\epsilon \sim N(0, \sigma_E^2)$. The noise is drawn once per fishery and iteration, so all species share it. Efficiency (effort creep) is $\eta_{f,y} = (1 + c_f)^{\max(0, y - y_f^c)}$; it scales fishing mortality but not the nominal effort index.

Usage

```
effort_sim(fishery, nsim = 100, seed = fishery$seed)
```

Arguments

`fishery`	a `fishery()`.
`nsim`	number of iterations.
`seed`	random seed; default the fishery's seed, so repeated calls give the same effort.

Value

a list with FLQuants effort (nominal) and efficiency, one element per fleet, each FLQuant(age = "all", year, iter = nsim).

Examples

```
eff <- effort_sim(reef_fishery(), nsim = 5)
eff$effort$traps
```

---

effort_trend *Effort trend for one fleet*

Description

Describes the shape of a fleet's nominal effort over time. "fudc" is the up-down-constant shape of FLRef::fudc(): effort starts at e0 in the first year, rises linearly to fhi x fref by the first break, moves linearly to flo x fref by the second break and stays there. "values" takes a user-supplied vector, one value per year.

Usage

```
effort_trend(
  type = c("fudc", "values"),
  e0 = 0.05,
  fhi = 1,
  flo = 0.8,
  fref = 1,
  breaks = c(0.5, 0.75),
  values = NULL
)
```

Arguments

type	"fudc" or "values".
e0	effort in the first year (relative units; near-unfished start).
fhi, flo, fref	levels for "fudc" (relative units).
breaks	two proportions of the time series (0-1) where the trend reaches fhi and then flo.
values	numeric vector of relative effort, one per year ("values").

Value

a list of class floss_trend.

Examples

```
effort_trend(fhi = 1, flo = 0.8, breaks = c(0.4, 0.6))
```

`export_data` *Export datasets for trainees*

Description

Writes one folder (or workbook) per iteration with the observed data, the settings needed to re-create them, and, separately, the answer key:

Usage

```
export_data(
  data,
  dir,
  give = c("data_lh", "data"),
  format = "csv",
  answer_key = TRUE,
  aggregate = FALSE,
  zip = FALSE
)
```

Arguments

`data`	a `floss_data`.
`dir`	output folder (created).
`give`	what goes into the trainee folders: `"data"` or `"data_lh"` (adds the life-history parameters).
`format`	any of `"csv"`, `"xlsx"` (needs 'writexl'; one workbook per iteration) and `"rds"` (one `floss_flr.rds` per iteration: a list by species of `as_flr()` objects, i.e. FLQuants of catch, effort and length frequencies, FLIndices and the life-history FLPar, ready for JABBA3 and FLicc).
`answer_key`	write the `answer_key` folder.
`aggregate`	also write the aggregated tables (for teaching): `catch_aggregate` (by fleet, all fleets, all species), `effort_aggregate` (by fleet, all fleets), `cpue_nominal` (catch / effort at the three levels) and `lfd_aggregate` (by fleet and all fleets, as sampled and raised to the catch); see `catch_table()` and `lfd_aggregate()`.
`zip`	also write `<dir>.zip`.

Details

```
<dir>/README.txt, settings.yaml
<dir>/iter_01/catch.csv effort.csv cpue.csv survey.csv lfd.csv bio.csv [lhpars.csv]
<dir>/answer_key/README_answer_key.txt truth.csv status.csv refpts.csv
  catch_true.csv bio_true_age.csv planted_artefacts.csv
```

Value

the folder, invisibly.

export_truth *Download the truth (trainers)*

Description

Writes the true time series (SSB, recruitment, Fbar, catch, B/Bmsy, F/Fmsy, catch/MSY), the reference points and the status table, without any observed data.

Usage

```
export_truth(x, dir, format = "csv", year = NULL, iters = NULL)
```

Arguments

x	a floss_data (uses its answer key) or a floss_om / floss_oms.
dir	output folder (created).
format	"csv" and/or "xlsx" (one workbook truth.xlsx with three sheets; needs 'writexl').
year	assessment year; default the last year.
iters	iterations (for operating models; default all).

Value

the folder, invisibly.

fishery *Define a fishery*

Description

A fishery is a set of fleets that share a time horizon. Each fleet has a nominal effort trend, process noise on effort that is **common to all species** (so species caught by the same fleet share effort anomalies) and an optional annual efficiency increase (effort creep) that raises fishing mortality but is not visible in the nominal effort index.

Usage

```
fishery(
  fleets,
  years,
  trends,
  sigmaE = 0.15,
  creep = 0,
  creep_start = NULL,
  name = "fishery",
```

```
  seed = 123
)

validate_fishery(x)

fishery_years(x)
```

Arguments

fleets	character vector of fleet names.
years	integer vector of years (first year = unfished start).
trends	named list of effort_trend() objects, one per fleet.
sigmaE	lognormal SD of the effort process noise (shared across species); scalar or one value per fleet.
creep	annual efficiency increase per fleet (0.02 = 2% per year).
creep_start	first year of creep per fleet (default: first year).
name	a label.
seed	seed for the effort realisation; every species built with this fishery sees the same effort.
x	a floss_fishery.

Details

Fishing mortality of species s by fleet f in year y is

$$F_{s,f,y} = q_{s,f}\,\eta_{f,y}\,E_{f,y}, \quad \eta_{f,y} = (1+c_f)^{\max(0,\,y-y^{c}_{f})},$$

where E is nominal effort (trend x lognormal noise), c_f the creep rate and $q_{s,f}$ the species catchability, set in species() as relative catchability x F_{MSY}.

Value

a list of class floss_fishery.

Examples

```
fishery(fleets = c("traps", "linefish"), years = 1980:2025,
        trends = list(traps = effort_trend(fhi = 1, flo = 0.8),
                      linefish = effort_trend(fhi = 2, flo = 1)))
```

fit_growth — *Fit von Bertalanffy growth to observed length-at-age*

Description

Nonlinear least squares, $L = L_\infty(1-e^{-k(a+\delta-t_0)})$, on the aged fish of the biological samples. Fish are caught during the year, so the observed (integer) age is shifted by `age_offset` (0.5 = mid-year) to make `t0` comparable with the true value.

Usage

```
fit_growth(
  data,
  species,
  iter = 1,
  source = NULL,
  years = NULL,
  t0 = NULL,
  age_offset = 0.5
)
```

Arguments

`data`	a `floss_data` or a `read_dataset()` list.
`species`	species name.
`iter`	iteration(s) (pooled).
`source`	sources to use; default all.
`years`	years to use; default all.
`t0`	fix t0 at this value (`NULL` = estimate).
`age_offset`	fraction of a year added to observed ages.

Value

a `floss_fit` with `pars` (`linf`, `k`, `t0`).

fit_lw — *Fit the length-weight relationship*

Description

Linear regression of $\log W$ on $\log L$ (W in g, L in cm), with the lognormal bias correction $a = e^{\alpha+\sigma^2/2}$.

Usage

```
fit_lw(data, species, iter = 1, source = NULL, years = NULL)
```

Arguments

data	a floss_data or a read_dataset() list.
species	species name.
iter	iteration(s) (pooled).
source	sources to use; default all.
years	years to use; default all.

Value

a floss_fit with pars (a, b).

---

fit_maturity *Fit a logistic maturity ogive*

Description

Binomial GLM (logit link) of maturity (0/1) on length: $L_{50} = -\beta_0/\beta_1$, $L_{95} = (\log 19 - \beta_0)/\beta_1$.

Usage

```
fit_maturity(
  data,
  species,
  iter = 1,
  source = NULL,
  years = NULL,
  bin_cm = NULL
)
```

Arguments

data	a floss_data or a read_dataset() list.
species	species name.
iter	iteration(s) (pooled).
source	sources to use; default all.
years	years to use; default all.
bin_cm	length bin for the plotted observed proportions.

Value

a floss_fit with pars (L50, L95).

floss.r4col *FLOSS colour palette (r4ss colours, as in JABBA)*

Description

The "rich colours" palette of r4ss and JABBA (`rc4()`, `ss3col()`, after `rich.colors` by Arni Magnusson): dark blue through green and yellow to red. For three colours it returns blue, red and green3, as in JABBA. Available for users who prefer the r4ss look; FLOSS plots use their own muted palette by default.

Usage

```
floss.r4col(n, alpha = 1)
```

Arguments

`n`	number of colours.
`alpha`	transparency (0-1).

Value

a character vector of colours.

Examples

```
floss.r4col(4)
barplot(rep(1, 6), col = floss.r4col(6))
```

floss_gears *FLRef gear objects for an operating model*

Description

Builds `FLRef::build_gear()` objects (length selectivity, capture- conditioned length key, sample sizes) for every fleet and the survey, on the length grid of the sampling settings.

Usage

```
floss_gears(om, oem)
```

Arguments

`om`	a `floss_om`.
`oem`	`oem_settings()`.

Value

named list of gears.

`floss_guide` *Copy, and optionally render, a FLOSS guide*

Description

The guides are installed with the package (folder `guide`), whether or not the vignettes were built. `floss_guide()` writes a guide to `dir` as a stand-alone R Markdown file that you can edit and knit, plus an R script with its code, and with `render = TRUE` renders it to HTML.

Usage

```
floss_guide(
 guide = c("user-guide", "teaching", "cpue-aggregation", "methods", "how-it-works",
    "reef-om", "all"),
  dir = ".",
  render = FALSE,
  script = TRUE,
  open = interactive(),
  overwrite = FALSE
)
```

Arguments

`guide`	`"user-guide"`, `"teaching"`, `"cpue-aggregation"`, `"methods"` (appendix with the equations), `"how-it-works"`, `"reef-om"`, or `"all"`.
`dir`	folder to write to (created).
`render`	also render to HTML (needs 'rmarkdown' and pandoc; runs all the code, which takes a few minutes for the user guide and the teaching scenarios).
`script`	also write the R script: the code with the guide text as `#'` comments (`knitr::purl()` with `documentation = 2`). Fully commented scripts of the user guide and the teaching scenarios are also installed: `system.file("guide", "FLOSS_user_guide.R", package = "FLOSS")`.
`open`	open the HTML after rendering.
`overwrite`	replace existing files.

Value

the paths of the files written, invisibly.

Examples

```
## Not run:
floss_guide("user-guide")                 # floss-02-user-guide.Rmd and .R
floss_guide("all", dir = "guides", render = TRUE)

## End(Not run)
```

floss_tip *Short explanations of FLOSS settings*

Description

The texts used for the information tooltips of the app.

Usage

```
floss_tip(id = NULL)
```

Arguments

id	setting name; NULL returns all tips.

Value

a character string (or a named list of all tips).

Examples

```
floss_tip("cpue_rho")
```

length_indicators *Length-based indicators by year and source*

Description

From the length samples of each year and source:

Usage

```
length_indicators(
  x,
  species,
  iter = 1,
  source = NULL,
  Lc = NULL,
  lh = NULL,
  season_weight = c("none", "equal")
)
```

Arguments

x	a floss_data, a read_dataset() list, or a floss_treatments (then one set of indicators per run, column run).
species	species name.
iter	iteration(s).
source	sources; default all.
Lc	length of first capture (cm) per source (named) or one value; default the modal length.
lh	optional list with linf, k, M, lm (cm, per year); default from the species settings or lhpars.
season_weight	how seasonal length samples are pooled into a year (lfd_annual()): "none" sums the counts (imbalanced sampling then biases the indicators), "equal" gives every season the same weight.

Details

- Lc: length of first full capture, here the mid-point of the modal length bin of the pooled samples of the source (or given).
- Lmean: mean length of fish at or above Lc; L95: 95th percentile; Lmax5: mean length of the largest 5% of fish.
- Pmat, Popt, Pmega (Froese 2004): proportion of fish above the length at maturity, within 10% of the optimum length $L_{opt} = 3L_\infty/(3 + M/k)$, and above 1.1 L_{opt}.
- Z_lbar: total mortality from mean length (Beverton and Holt 1956), $Z = k(L_\infty - \bar{L})/(\bar{L} - L_c)$. It assumes equilibrium: constant recruitment and mortality, and flat-topped selectivity.
- Z_true (when the answer key is present): true Fbar + M in that year, to show when and why Z_lbar departs from it.

Value

a data frame, one row per species, iteration, year and source.

---

lfd_aggregate *Length samples: by fleet or pooled, as sampled or raised to the catch*

Description

- raise = "sample": the counts as measured. Pooled over fleets (by = "aggregate"), each fleet weighs by its sample size, not by its catch: a fleet that is sampled heavily but catches little dominates the pooled distribution.
- raise = "catch": each fleet's (season-weighted) sample is raised to the fleet's observed catch: numbers = sample numbers x catch (t) / sample weight (t), with weights from the length-weight relationship. Pooled over fleets, each fleet then weighs by its catch.

Usage

```
lfd_aggregate(
  data,
  species = NULL,
  by = c("fleet", "aggregate"),
  raise = c("sample", "catch")
)
```

Arguments

data	a floss_data.
species	species names; default all (for "combined", the species summed).
by	"fleet" or "aggregate" (all fleets pooled, source "all fleets").
raise	"sample" or "catch".

Details

Seasons are pooled for the year: counts summed for "sample", equal season weights for "catch" (lfd_annual()). The survey is not included.

Value

a data frame like data$lfd without season (n is numeric when raised).

---

lfd_annual *Pool seasonal length samples into annual ones*

Description

Sums the length samples of all seasons of a year. With weight = "equal" each season gets the same weight whatever its sample size (each season's distribution is rescaled to the mean seasonal sample size before summing), which removes the bias of imbalanced sampling when the catch is spread evenly over the year.

Usage

```
lfd_annual(lfd, weight = c("none", "equal"))
```

Arguments

lfd	the lfd table of a floss_data (with or without season).
weight	"none" (sum the counts) or "equal" (equal season weights).

Value

the table without the season column.

load_fishery *Example fishery shipped with the package*

Description

Example fishery shipped with the package

Usage

```
load_fishery(name = "reef_3fleets")
```

Arguments

name	fishery name (file in inst/extdata/fishery).

M_empirical *Empirical natural mortality estimators*

Description

Empirical natural mortality estimators

Usage

```
M_empirical(linf = NULL, k = NULL, tmax = NULL, temp = NULL)
```

Arguments

linf, k	growth parameters (cm, per year).
tmax	maximum observed age (years).
temp	mean water temperature (deg C) for Pauly's (1980) estimator; omitted when NULL.

Value

a data frame with method, M and inputs.

Examples

```
M_empirical(linf = 71.4, k = 0.21, tmax = 16)
```

`oem_settings` *Sampling settings*

Description

Collects every setting of the observation model. Arguments that can differ by fleet take one value or a vector named by fleet. Catch and effort cover all years of the operating model; CPUE, length frequencies and biological samples cover the last `tail` years of each fleet (or `years`).

Usage

```
oem_settings(
  fishery,
  tail = 10,
  catch_cv = 0.1,
  catch_bias = 0,
  effort_cv = 0.1,
  effort_days = 1000,
  cpue_cv = 0.2,
  cpue_rho = 0.3,
  cpue_creep = TRUE,
  cpue_beta = 1,
  lfd_ess_age = 30,
  lfd_ess_len = 300,
  lfd_bin = NULL,
  lfd_cv = 0.1,
  bio_n = 100,
  bio_p_aged = 0.5,
  bio_age_scheme = c("random", "stratified"),
  bio_n_per_bin = 5,
  age_cv = 0.1,
  wt_cv = 0.1,
  mat_L95 = 1.15,
  catch_bias_from = NULL,
  cpue_q_step = NULL,
  bio_age_bias = 0,
  survey = survey_settings(),
  years = NULL,
  seasons = season_design(4)
)
```

Arguments

`fishery`	a `fishery()` (or a `floss_om` / `floss_oms`).
`tail`	number of most recent years with CPUE, length and biological samples, per fleet.

`catch_cv`, `catch_bias`
: catch CV and under-reporting (0.1 = 10% under).

`effort_cv`
: effort-index CV.

`effort_days`
: fishing days per year of a fleet at relative effort 1 (per fleet): observed effort is reported in fishing days.

`cpue_cv`, `cpue_rho`
: CPUE CV and residual autocorrelation.

`cpue_creep`
: does CPUE carry effort creep (rising catchability)?

`cpue_beta`
: proportionality exponent, $I = qB^{\beta}$: 1 = proportional, < 1 hyperstable (e.g. range contraction), > 1 hyperdepletion.

`lfd_ess_age`, `lfd_ess_len`
: stage-1 (clusters) and stage-2 (fish measured) sample sizes.

`lfd_bin`
: length bin (cm); `NULL` picks about Linf/40 per species.

`lfd_cv`
: CV of length-at-age.

`bio_n`, `bio_p_aged`
: fish per year and share aged (random ageing).

`bio_age_scheme`
: `"random"` (a share `bio_p_aged` of fish aged) or `"stratified"` (length-stratified subsample: at most `bio_n_per_bin` fish aged per length bin, source and year, as `FLRef::alk.sample()`).

`bio_n_per_bin`
: otoliths per length bin for the stratified scheme.

`age_cv`, `wt_cv`
: ageing-error CV and weight CV.

`mat_L95`
: length at 95% maturity relative to `lm` (L50).

`catch_bias_from`
: year from which `catch_bias` applies, per fleet (default: all years); an artefact such as under-reporting that starts with a new quota.

`cpue_q_step`
: artefact: named list by fleet of `c(year, mult)`, the fleet's catchability changes by `mult` from `year` (e.g. new electronics), or `NULL`.

`bio_age_bias`
: proportional ageing bias (default 0 = unbiased); a planted data problem when not 0, e.g. -0.1 = fish are aged 10% too young (typical of old fish read from whole otoliths).

`survey`
: `survey_settings()` or `NULL` for no survey.

`years`
: optional named list by fleet of `c(first, last)` years, overriding `tail`.

`seasons`
: seasonal sampling of the fleets' length and biological samples: a `season_design()` for all fleets, or a named list of designs by fleet. Default: four seasons, balanced. `season_design(1)` gives one annual sample at mid-year.

Details

- **Catch** by fleet: true catch x (1 - bias) x lognormal error.
- **Effort** index by fleet: nominal effort x lognormal error (effort creep is not visible).
- **CPUE** by fleet: vulnerable biomass x efficiency (if `cpue_creep`) with lognormal AR1 residuals, $\epsilon_y = \rho\,\epsilon_{y-1} + \sqrt{1-\rho^2}\,\delta_y$. Use $\rho > 0$ for non-random (clustered) sampling of the fleet.

- **Length frequencies**: the two-stage FLRef protocol (`FLRef::lfd.sim()`): `lfd_ess_age` clusters drawn from the fleet's catch-at-age, then `lfd_ess_len` fish measured through the gear's length key.
- **Biological samples**: `bio_n` fish per fleet and year with length (mm), age (with ageing error; aged at random or length-stratified), weight (g, with error) and maturity (0/1).

Value

a list of class `floss_oem`.

Examples

```
oem_settings(reef_fishery(), tail = c(traps = 5, gillnet = 10, linefish = 15))
```

---

one_gear_fishery *One-gear fishery with constant (or stepped) F/Fmsy*

Description

A fishery with a single fleet whose effort is directly F/Fmsy when used with `one_gear_species()`. Use for light vs hard fishing comparisons.

Usage

```
one_gear_fishery(
  F_rel = 1,
  years = 1969:2025,
  fleet = "handline",
  ramp = 10,
  sigmaE = 0.1,
  seed = 123
)
```

Arguments

`F_rel`	F/Fmsy: one value (constant after a short ramp), or a vector with one value per year.
`years`	years.
`fleet`	fleet name.
`ramp`	years to ramp up from near zero to `F_rel` (0 = immediately).
`sigmaE`	effort noise.
`seed`	seed.

Value

a `floss_fishery`.

Examples

```
light <- one_gear_fishery(0.5)
hard  <- one_gear_fishery(2)
```

`one_gear_species` *Adapt a species to a one-gear fishery*

Description

Copies a species with a single fleet (default a logistic handline with L50 at the length at maturity) and scales its catchability so that the fishery's effort equals F/Fmsy.

Usage

```
one_gear_species(sp, fleet = "handline", sel = NULL)
```

Arguments

`sp`	a `species()`.
`fleet`	fleet name (as in `one_gear_fishery()`).
`sel`	a `selectivity()`; default logistic with L50 = `lm`, L95 = 1.2 `lm`.

Value

a `floss_species`.

`planted_artefacts` *Artefacts planted in the sampling settings and the fishery*

Description

Lists every setting that makes the data depart from the stock in a way a careful analyst should detect (the "warning flags"): effort creep carried by CPUE, non-proportional CPUE, strongly auto-correlated residuals, catch bias, catchability steps, ageing bias and very small length samples, plus treatments applied to the operating model. Written to the answer key by `export_data()`.

Usage

```
planted_artefacts(oem, fishery = NULL, treatment = NULL)
```

Arguments

`oem`	`oem_settings()`.
`fishery`	the `fishery()`.
`treatment`	optional `treatment()` of the operating model.

Value

a data frame with `artefact`, `where`, `detail` (empty if none).

`plot_aggregate` *Plot catch, effort or nominal CPUE by fleet or aggregated*

Description

One panel per fleet (or one for all fleets); with several iterations the lines are coloured by iteration. With `truth = TRUE`, true values are dashed: for nominal CPUE, true catch / true effort, and in grey the true spawning biomass relative to its mean, to show how far nominal CPUE is from the stock.

Usage

```
plot_aggregate(
  data,
  species = NULL,
  iter = 1,
  what = c("cpue", "catch", "effort"),
  by = c("fleet", "aggregate", "combined"),
  truth = FALSE
)
```

Arguments

`data`	a `floss_data`.
`species`	species name (ignored for `by = "combined"`).
`iter`	iteration(s).
`what`	`"catch"`, `"effort"` or `"cpue"` (nominal).
`by`	`"fleet"`, `"aggregate"` or `"combined"` (catch, CPUE).
`truth`	show true values.

Value

a `ggplot`.

plot_availability *Plot which data exist for which years*

Description

One row per data stream and source, one tile per year with data.

Usage

```
plot_availability(data, species = NULL, iter = 1)
```

Arguments

data	a floss_data (or read_dataset() list).
species	species name; default the first.
iter	iteration(s); several iterations are overlaid (lines) or shown in rows (length samples).

Value

a ggplot.

plot_bio *Plot biological samples (observations only)*

Description

- "age_length": length against observed age.
- "length_weight": weight against length.
- "maturity": observed proportion mature by length bin (point size = number of fish).
- "alk": observed age-length key (counts; bubble area).
- "length": length composition of the biological samples.

Usage

```
plot_bio(
  data,
  species = NULL,
  iter = 1,
  what = c("age_length", "length_weight", "maturity", "alk", "length"),
  source = NULL,
  by = c("fleet", "aggregate", "season"),
  truth = FALSE
)
```

Arguments

data	a floss_data (or read_dataset() list).
species	species name; default the first.
iter	iteration(s); several iterations are overlaid (lines) or shown in rows (length samples).
what	one of the options above.
source	sources to include; default all.
by	panels: "fleet" (one panel per fleet), "aggregate" (all fleets in one panel, as a naive analyst would pool them) or "season" (one panel per sampling season: fish grow through the year, so length at age rises from season to season).
truth	draw the true growth, length-weight and maturity curves.

Details

Points are always coloured by fleet (or survey); by sets the panels. With truth = TRUE the true relationship of the operating model is drawn over the observations: the von Bertalanffy growth curve (at the middle of the sampling season, or mid-year), the length-weight curve and the maturity ogive.

Value

a ggplot.

---

plot_caa *Plot catch-at-age from* caa_obs()

Description

Plot catch-at-age from caa_obs()

Usage

```
plot_caa(caa, type = c("bubble", "bar"))
```

Arguments

caa	result of caa_obs().
type	"bubble" (year x age) or "bar" (age composition by year).

Value

a ggplot.

plot_effort *Plot fleet effort*

Description

Median and 90% interval of nominal effort by fleet, with a few individual iterations. With `efficiency = TRUE` the effective effort (nominal x efficiency) is shown as well, to visualise effort creep.

Usage

```
plot_effort(x, nsim = 100, iters = 1:3, efficiency = FALSE)
```

Arguments

`x`	a `fishery()` (effort is simulated) or a `floss_om`.
`nsim`	iterations to simulate when `x` is a fishery.
`iters`	iterations drawn as lines.
`efficiency`	also show effective effort.

Value

a `ggplot`.

Examples

```
plot_effort(reef_fishery(creep = 0.02), efficiency = TRUE)
```

plot_eq *Equilibrium curves with the realised trajectory*

Description

Yield, SSB and recruitment against F and SSB at equilibrium (`FLRef::ploteq()`), with the stock's yearly values as points (`obs = TRUE`). During a fishing-down phase catches lie above the equilibrium yield curve: high catches are possible only while biomass is being removed. The curves are for the exploitation pattern of the last years; with a changing fleet mix, earlier years belonged to a different equilibrium.

Usage

```
plot_eq(om, iter = 1, obs = TRUE, rel = FALSE)
```

Arguments

om	a floss_om.
iter	iteration whose trajectory is shown.
obs	show the realised trajectory.
rel	plot relative to the MSY reference points.

Value

a ggplot.

---

plot_eq_selectivity *Equilibrium curves under alternative selectivities*

Description

The same stock (life history, recruitment, unfished biomass) fished with different selectivities: equilibrium yield and SSB against F (Fbar). Catching fish before they have grown lowers the maximum yield and the F at which it is reached (growth overfishing); protecting small fish raises both. Points mark MSY for each selectivity.

Usage

```
plot_eq_selectivity(sp, sels, fmax = 10)
```

Arguments

sp	a species().
sels	named list of selectivity() objects.
fmax	largest F shown, as a multiple of Fmsy of the species' own (joint) selectivity.

Value

a ggplot.

Examples

```
sp <- load_species("lutjanus_bohar")
plot_eq_selectivity(sp, list(
  small = selectivity("logistic", L50 = 25, L95 = 30),
  mature = selectivity("logistic", L50 = 35, L95 = 42),
  large = selectivity("logistic", L50 = 50, L95 = 58)))
```

plot_fit *Plot a basic fit*

Description

Draws the observations, the fitted curve and the estimates; with `truth = TRUE` (trainers) the true curve or values are added as dashed lines.

Usage

```
plot_fit(fit, truth = FALSE, what = NULL)
```

Arguments

`fit`	a `floss_fit`.
`truth`	add the truth where known.
`what`	for `catch_curve_length()`: `"curve"` or `"sel"` (selectivity); for `cpue_check()`: `"scatter"` or `"time"`.

Value

a `ggplot`.

plot_fleets *Plot apical fishing mortality or catch by fleet*

Description

Plot apical fishing mortality or catch by fleet

Usage

```
plot_fleets(om, what = c("fapic", "catch"), iters = 1:3)
```

Arguments

`om`	a `floss_om`.
`what`	`"fapic"` (apical F by fleet, relative to Fmsy) or `"catch"` (true catch by fleet, t).
`iters`	iterations drawn as lines.

Value

a `ggplot`.

plot_indicators *Indicator dashboard*

Description

All series of one species on one page, each scaled to its own mean: abundance indices (CPUE by fleet, survey), total catch, and mean length and Pmega by source. Independent indicators of the same stock should broadly agree; where they diverge, one of them is probably not tracking the stock (e.g. effort creep, a gear change, a cohort passing through).

Usage

```
plot_indicators(data, species = NULL, iter = 1, truth = FALSE)
```

Arguments

data	a floss_data (or read_dataset() list).
species	species name; default the first.
iter	iteration(s); several iterations are overlaid (lines) or shown in rows (length samples).
truth	add true SSB (scaled to its mean) to the abundance panel.

Value

a ggplot.

plot_kobe *Kobe plot of true status*

Description

True B/Bmsy against F/Fmsy, coloured by species. With several iterations, one point per species and iteration in the assessment year. With one iteration (or trajectory = TRUE), the whole history of each species is drawn as a path from the first year (open circle) to the assessment year (filled circle), as in JABBA's Kobe plot.

Usage

```
plot_kobe(x, year = NULL, iters = NULL, species = NULL, trajectory = NULL)
```

Arguments

x	a floss_data (uses its answer key) or a floss_om / floss_oms.
year	assessment year; default the last year.
iters	iterations (for operating models; default all).
species	species to show; default all.
trajectory	draw the history as a path; default when one iteration is shown.

Value

a ggplot.

---

plot_length_indicators

Plot length-based indicators

Description

Mean length (with the optimum length and Lc), the share of mature, optimum-sized and mega-spawning fish, and Z from mean length against the true Z. With several runs (from a floss_treatments), runs are coloured and sources are panels.

Usage

```
plot_length_indicators(li, source = NULL, stat = c("median", "all"))
```

Arguments

li	result of length_indicators().
source	sources to show; default all.
stat	with several iterations: "median" (median over iterations, the systematic response) or "all" (one line per iteration, the noise).

Value

a ggplot.

`plot_lfd` *Plot length samples*

Description

- `type = "bars"` (default): the classic length-frequency histogram, one panel per year (rows) and source (columns), see below.
- `type = "ridge"`: one length distribution per year, drawn as a ridge along the time axis (as FLRef's `plot_lfd_ridgeline()`), one panel per source; with `cohorts = TRUE`, von Bertalanffy lines follow each year class, so modes can be followed along them.
- `type = "bubble"`: one row per year, bubble area = proportion of the year's sample at each length. With `cohorts = TRUE`, von Bertalanffy growth lines follow each year class through the samples: real modes move along these lines; patterns that jump or vanish are sampling noise or selectivity, not dynamics.
- `type = "bars"`: the classic length-frequency histogram, one panel per year (rows) and source (columns), as proportions of each year's sample. With `cohorts = TRUE`, the mean length of each age class in that year is marked at the top, so modes can be matched to year classes.

Usage

```
plot_lfd(
  data,
  species = NULL,
  iter = 1,
  source = NULL,
  type = c("bars", "ridge", "bubble"),
  years = NULL,
  cohorts = FALSE,
  season = NULL,
  lfd = NULL,
  by_season = FALSE
)
```

Arguments

`data`	a `floss_data` (or `read_dataset()` list).
`species`	species name; default the first.
`iter`	iteration(s); several iterations are overlaid (lines) or shown in rows (length samples).
`source`	fleets / survey to show; default all.
`type`	`"bars"`, `"ridge"` or `"bubble"`.
`years`	years to show; default the last six for `"bars"`, all for `"ridge"` and `"bubble"`.
`cohorts`	add cohort growth lines (bubble) or age-class marks (bars).
`season`	seasons to include; default all.

lfd	optional length table to plot instead of data$lfd, e.g. from lfd_aggregate() (pooled over fleets or raised to the catch).
by_season	show the seasons separately instead of pooling them.

Details

Seasonal samples are pooled for the year (counts summed), unless by_season = TRUE: then each season is shown separately (histogram: a panel per year and season; ridge: one ridge per season within the year, coloured by season, as FLRef's plot_lfd_ridgeline() with seasonal data; bubble: seasons stacked within the year). Fish grow through the year, so modes move to the right from season to season; with imbalanced sampling (season_design()) some seasons have few or no samples.

Value

a ggplot.

Examples

```
x <- simulate_fishery("lutjanus_bohar", nsim = 2, iters = 1)
plot_lfd(x$data, cohorts = TRUE)
plot_lfd(x$data, type = "bars", source = c("gillnet", "linefish"))
```

---

plot_obs *Plot observed catch, CPUE and survey indices*

Description

Plot observed catch, CPUE and survey indices

Usage

```
plot_obs(
  data,
  species = NULL,
  iter = 1,
  what = c("index", "catch"),
  truth = FALSE
)
```

Arguments

data	a floss_data (or read_dataset() list).
species	species name; default the first.
iter	iteration(s); several iterations are overlaid (lines) or shown in rows (length samples).
what	"catch" or "index" (CPUE and survey).
truth	add the true values (dashed; trainer view).

Value

a ggplot.

`plot_om` *Plot stock status of an operating model*

Description

Thin wrapper around `FLRef::plotAdvice()`. Several species are shown relative to their own reference points (B/Bmsy, F/Fmsy, C/MSY).

Usage

```
plot_om(om)
```

Arguments

`om`	a `floss_om` or `floss_oms`.

Value

a ggplot.

`plot_seasons` *Seasonal length sampling: sample sizes and mean length by season*

Description

Left: fish measured per season and year (how balanced the sampling is). Right: mean length by season and year, with the pooled annual mean (black) and the mean with equal season weights (dashed). Fish grow through the year, so later seasons have larger fish; when most samples come from one part of the year, the pooled mean follows that season, and a shift in the sampling season looks like a change in the stock.

Usage

```
plot_seasons(data, species = NULL, iter = 1, source = NULL)
```

Arguments

`data`	a `floss_data` (or `read_dataset()` list).
`species`	species name; default the first.
`iter`	iteration(s); several iterations are overlaid (lines) or shown in rows (length samples).
`source`	fleets / survey to show; default all.

Value

a ggplot.

---

plot_srr *Stock-recruitment curve with simulated pairs*

Description

The Beverton-Holt curve of the operating model with the SSB-recruitment pairs of the selected iterations (recruitment at age 0 in the same year). The scatter is the recruitment variability (sigmaR); it is why steepness is hard to estimate from data.

Usage

```
plot_srr(om, iters = 1:3)
```

Arguments

om	a floss_om.
iters	iterations to show.

Value

a ggplot.

---

plot_ssb_vb *Spawning biomass versus vulnerable biomass by fleet*

Description

Who reproduces and who is caught are not the same fish. The left panel compares each fleet's selectivity-at-length with the maturity ogive; the right panel compares the trend in spawning biomass (SSB, reproductive potential) with the trend in each fleet's vulnerable biomass (the biomass its catches and CPUE come from), both relative to the first year. With dome-shaped selectivity, large spawners escape: a fleet's CPUE can fall faster or slower than SSB, so an index can be a distorted picture of the reproductive stock.

Usage

```
plot_ssb_vb(om, iters = NULL)
```

Arguments

om	a floss_om.
iters	iterations (medians are shown).

Value

a ggplot (two panels, via facets).

---

plot_treatment *Plot the response to treatments*

Description

True B/Bmsy, F/Fmsy, catch/MSY and recruitment (relative to the baseline mean) for the baseline and each treatment: median and 90% interval over iterations; the treatment start is marked.

Usage

```
plot_treatment(x, species = NULL)
```

Arguments

x	a floss_treatments.
species	species id; default the first.

Value

a ggplot.

---

plot_truth *Plot true status trajectories*

Description

B/Bmsy, F/Fmsy and catch/MSY over time by species, one line per dataset (iteration), with the reference level 1 dashed (as FLRef::plotAdvice() on ratios).

Usage

```
plot_truth(x, species = NULL, iters = NULL)
```

Arguments

x	a floss_data (uses its answer key) or a floss_om / floss_oms.
species	species to show; default all.
iters	iterations (for operating models; default all).

Value

a ggplot.

read_dataset *Read an exported dataset*

Description

Reads the csv files of one exported iteration back into a list of data frames, so that trainees (or the plotting functions) can use them.

Usage

```
read_dataset(dir)
```

Arguments

dir	the iteration folder (e.g. "…/iter_01").

Value

a list of data frames (catch, effort, cpue, survey, lfd, bio, lhpars where present), each with an iter column set to 1.

reef_fishery *The three-fleet reef fishery used in the examples*

Description

Traps, gillnet and linefish, 1969-2025, with the effort trends of the multi-gear reef operating model (Winker, FLsim multigear reef example).

Usage

```
reef_fishery(sigmaE = 0.15, creep = 0, seed = 123)
```

Arguments

sigmaE	lognormal SD of the effort process noise (shared across species); scalar or one value per fleet.
creep	annual efficiency increase per fleet (0.02 = 2% per year).
seed	seed for the effort realisation; every species built with this fishery sees the same effort.

Value

a floss_fishery.

Examples

```
reef_fishery()
```

reef_oem *Sampling settings of the reef example*

Description

Length, CPUE and biological samples cover the most recent 5 years for traps, 10 for gillnet and 15 for linefish; linefish CPUE residuals are more autocorrelated (clustered fishing).

Usage

```
reef_oem(...)
```

Arguments

`...`	further arguments to `oem_settings()`.

Value

a `floss_oem`.

run_app *Start the FLOSS app*

Description

Opens the FLOSS Shiny app: build a fishery and species, the operating model, treatments, sampling and export, step by step in the left panel. The app uses the same functions as the package, so everything done in the app can be repeated in a script.

Usage

```
run_app(...)
```

Arguments

`...`	passed to `shiny::runApp()`, e.g. `launch.browser = TRUE` or `port = 3838`.

Value

called for its side effect (runs the app).

Examples

```
if (interactive()) run_app()
```

`run_treatment` *Run a baseline and treatments with common random numbers*

Description

Builds the baseline operating model(s) and one per treatment, all with the same effort noise, recruitment deviations and (when `oem` is given) sampling random numbers.

Usage

```
run_treatment(
  species,
  fishery,
  treatments,
  nsim = 20,
  oem = NULL,
  iters = 1,
  seed = fishery$seed,
  ...
)
```

Arguments

`species`	species ids or `species()` objects.
`fishery`	a `fishery()`.
`treatments`	a `treatment()` or a named list of treatments.
`nsim`	iterations.
`oem`	optional `oem_settings()`: also sample data for `iters`.
`iters`	iterations to sample.
`seed`	base seed (operating models and sampling).
`...`	passed to `build_om()`.

Value

a list of class `floss_treatments` with `oms` and (if `oem`) `data`, each a named list (`baseline`, then the treatments).

`sample_data` *Generate observed data*

Description

Samples catch, effort, CPUE, survey index, length frequencies and biological samples from one or several operating models, for the chosen iterations, and returns them with the truth (answer key).

Usage

```
sample_data(om, oem, iters = 1:10, seed = 123)

sample_catch(om, oem, iters = 1:10)

sample_effort(om, oem, iters = 1:10)

sample_cpue(om, oem, iters = 1:10)

sample_survey(om, oem, iters = 1:10)

sample_lfd(om, oem, iters = 1:10)

sample_bio(om, oem, iters = 1:10)
```

Arguments

`om`	a `floss_om` or `floss_oms` (several species).
`oem`	`oem_settings()`.
`iters`	iterations to sample (one dataset per iteration).
`seed`	base seed.

Details

Each data stream of each species uses its own reproducible random stream, so changing e.g. the biological-sample settings does not change the CPUE.

Value

an object of class `floss_data`: a list of data frames `catch`, `effort`, `cpue`, `survey`, `lfd`, `bio`, `lhpars`, `truth`, `refpts`, plus `settings` (fishery, species and sampling specs).

Functions

- `sample_catch()`: observed catch by fleet (t).
- `sample_effort()`: observed nominal effort by fleet, in fishing days (relative effort x `effort_days` of the fleet).

- `sample_cpue()`: CPUE by fleet with AR1 lognormal residuals, scaled to mean 1 over the sampled years; `index_true` is the true vulnerable biomass (without effort creep).
- `sample_survey()`: survey biomass index (relative to its mean).
- `sample_lfd()`: two-stage length samples by fleet and survey, by season (see `season_design()`); `len_mm` is the lower bin limit.
- `sample_bio()`: biological samples: one row per fish, by season.

`season_design` *Seasonal sampling design*

Description

Describes how the annual length and biological samples of a fleet are spread over seasons. The default, four seasons with equal shares, is a balanced design. Imbalanced designs:

Usage

```
season_design(
  n = 4,
  share = NULL,
  shift = NULL,
  kappa = Inf,
  n_mult = NULL,
  p_missing = 0,
  labels = NULL
)
```

Arguments

`n`	number of seasons (1 = one annual sample at mid-year, as in FLOSS <= 0.4).
`share`	sample share by season (rescaled to sum to 1); default equal.
`shift`	optional list of `from` (year), `share` and optionally `over` (years): the shares change from `from` on, linearly over `over` years (default 0, a step). Several shifts as a list of such lists.
`kappa`	Dirichlet concentration for random year-to-year shares; `Inf` (default) = no random variation.
`n_mult`	optional list of `from`, `mult` and optionally `over`: the annual sample size (clusters, fish measured, biological samples) is multiplied by `mult` from `from` on (ramped over `over` years).
`p_missing`	probability that a year is not sampled at all.
`labels`	season names; default `"S1"`, `"S2"`, ...

Details

- **fixed imbalance**: unequal `share`, e.g. `c(0.5, 0.3, 0.2, 0)` (most samples early in the year, none in season 4, e.g. a monsoon closure of the landing site);
- **shift**: the shares change from a given year (`shift`), e.g. a new observer programme that samples in the second half of the year only;
- **opportunistic**: the shares vary at random from year to year around `share` (`kappa`, the concentration of a Dirichlet distribution: 50 = small departures, 5 = large ones), as when samples are taken whenever an observer happens to be available.

Sampling intensity over the years:

- **intensity change**: the annual sample size is multiplied by `n_mult` from a year on (`list(from = 2020, mult = 0.3)`: a funding cut), with an optional ramp over `over` years;
- **missing years**: each year is not sampled at all with probability `p_missing`.

Within a season, the age composition of the catch is the Baranov catch between the start and end of the season (fishing effort and natural mortality constant through the year), and lengths are those at the middle of the season (von Bertalanffy growth with age + season mid-point). Because effort is constant within the year, the catch of each season is known from the dynamics, and equal season weights (`lfd_annual()`) are the correct way to pool unevenly spread samples.

Value

a list of class `floss_seasons`.

Examples

```
season_design()                                    # balanced, 4 seasons
season_design(share = c(0.6, 0.3, 0.1, 0))         # fixed imbalance
season_design(shift = list(from = 2015, share = c(0, 0, 0.5, 0.5)))
season_design(kappa = 5)                           # opportunistic
```

`selectivity` *Selectivity-at-length for one fleet*

Description

Arguments follow `FLRef::sel_la()`: `"logistic"` uses `L50`, `L95`; `"normal"` uses `Lpeak`, `sd`; `"dnormal"` uses `Lpeak`, `sd_left`, `sd_right`. Lengths in cm.

Usage

```
selectivity(
  type = c("logistic", "normal", "dnormal"),
  L50 = NULL,
  L95 = NULL,
  Lpeak = NULL,
  sd = NULL,
```

```
  sd_left = NULL,
  sd_right = NULL
)
```

Arguments

type	"logistic", "normal" or "dnormal".
L50, L95, Lpeak, sd, sd_left, sd_right	shape parameters (cm).

Value

a list of class floss_sel.

Examples

```
selectivity("dnormal", Lpeak = 20, sd_left = 5, sd_right = 15)
```

---

simulate_fishery *Simulate a fishery and its data in one call*

Description

Wrapper around build_oms() and sample_data().

Usage

```
simulate_fishery(
  species = species_library()$id,
  fishery = load_fishery(),
  oem = NULL,
  nsim = 20,
  iters = 1:min(10, nsim),
  seed = 123,
  ...
)
```

Arguments

species	species ids from the library or species() objects.
fishery	a fishery().
oem	oem_settings(); default oem_settings(fishery).
nsim	iterations of the operating models.
iters	iterations to sample as datasets.
seed	base seed for sampling.
...	passed to build_oms() (e.g. sigmaQ, ar1).

Value

a list with oms (floss_oms) and data (floss_data).

Examples

```
sim <- simulate_fishery("siganus_sutor", reef_fishery(), nsim = 5, iters = 1)
sim$data
```

---

species *Define a species*

Description

Define a species

Usage

```
species(
  species,
  common_name = "",
  lh,
  msy,
  sel,
  relq,
  productivity = "",
  id = NULL
)

validate_species(x, fishery = NULL)
```

Arguments

species	scientific name.
common_name	common name.
lh	named list or vector of life-history parameters: linf, k, t0 (von Bertalanffy, cm and years), a, b (W = aL^b, g and cm), lm (length at 50% maturity, cm), tm (age at 50% maturity), tmax (maximum age, also the plus group), M (natural mortality), s (steepness), sigR (recruitment SD) and rho (recruitment autocorrelation, used when build_om(ar1 = TRUE)).
msy	target MSY (t) used to scale the stock.
sel	named list of selectivity() objects, one per fleet.
relq	named numeric vector of relative catchability by fleet: apical F of fleet f is relq[f] x Fmsy x effort[f].
productivity	optional label (e.g. "High").

id	file-safe identifier; default from species.
x	a floss_species.
fishery	optional floss_fishery: checks that the species has selectivity and catchability for every fleet.

Value

a list of class floss_species.

---

species_lhpar *Life-history FLPar for FLife*

Description

Life-history FLPar for FLife

Usage

```
species_lhpar(sp)
```

Arguments

sp	a species().

Value

FLPar from FLife::lhPar().

---

species_library *Species library*

Description

The library combines the read-only example species shipped with the package and the species saved by the user with save_species(), which are kept in tools::R_user_dir("FLOSS", "data") (override with options(FLOSS.user_dir = ...)). A user species with the same id as an example takes precedence.

Usage

```
species_library()

load_species(id)

save_species(x, overwrite = FALSE)

delete_species(id)
```

Arguments

id	species id (see species_library()$id).
x	a floss_species.
overwrite	overwrite an existing user species with the same id.

Value

species_library(): a data frame with one row per species (id, species, common_name, msy, source, file).

Examples

```
species_library()
load_species("siganus_sutor")
```

---

species_sel *Selectivity-at-length and -at-age by fleet*

Description

Wraps FLRef::sel_la() for every fleet of the species, over ages 0:tmax.

Usage

```
species_sel(sp, lhpar = species_lhpar(sp))
```

Arguments

sp	a species().
lhpar	optional FLPar from species_lhpar().

Value

named list (one per fleet) of sel_la() results.

status_table *True stock status*

Description

Summarises the true status of each species in each dataset (iteration): B/Bmsy, F/Fmsy and catch/MSY in a given year, and the Kobe status (`"Healthy"`: B >= Bmsy and F <= Fmsy; `"Overfishing"`: F > Fmsy with B >= Bmsy; `"Overfished"`: B < Bmsy with F <= Fmsy; `"Overfished and overfishing"`: both).

Usage

```
status_table(x, year = NULL, iters = NULL, summary = FALSE)
```

Arguments

`x`	a `floss_data` (uses its answer key) or a `floss_om` / `floss_oms`.
`year`	assessment year; default the last year.
`iters`	iterations (for operating models; default all).
`summary`	return the median over iterations per species instead.

Value

a data frame.

survey_settings *Survey settings*

Description

A fishery-independent survey that samples every species. Because one survey gear meets species of very different size, its selectivity is given **relative to each species' L-infinity** (e.g. `L50 = 0.2` means 50% selected at 20% of Linf).

Usage

```
survey_settings(
  years = NULL,
  timing = 0,
  cv = 0.25,
  rho = 0,
  sel = list(type = "logistic", L50 = 0.2, L95 = 0.3),
  lfd = TRUE,
  ess_age = 50,
  ess_len = 500,
```

```
  bio_n = 0,
  q_step = NULL
)
```

Arguments

`years`	first and last survey year (default: set by `oem_settings()` to the last 25 years).
`timing`	fraction of the year when the survey takes place (0 = 1 Jan).
`cv`, `rho`	observation CV and lag-1 autocorrelation of the index residuals.
`sel`	selectivity relative to Linf, as for `selectivity()`.
`lfd`, `ess_age`, `ess_len`	take two-stage length samples from the survey (number of stations and of fish measured).
`bio_n`	fish per year for biological samples from the survey (0 = none).
`q_step`	artefact: `c(year, mult)`, the survey catchability changes by `mult` from `year` (e.g. a new vessel or gear), or `NULL`.

Value

a list of class `floss_survey`.

`treatment` *Define a treatment*

Description

A treatment is one or more controlled changes applied to an operating model from a given year. Run with `run_treatment()`, the baseline and the treated operating models share every random number (effort noise, recruitment deviations, sampling), so all differences come from the treatment.

Usage

```
treatment(
  name = "treatment",
  effort = NULL,
  recruitment = NULL,
  selectivity = NULL,
  creep = NULL,
  M = NULL
)
```

Arguments

`name`	label.
`effort`, `recruitment`, `selectivity`, `creep`, `M`	treatment components.

Details

Components (each optional, NULL = no change):

- effort: list of fleet, from, mult: nominal effort of the fleet is multiplied by mult from year from (mult = 0: fleet leaves; a value0 for a fleet with no effort before has no effect, so use the fisherytrend to add a fleet).
- recruitment: list with type = "pulse" (years, mult), "trend" (from, rate per year, e.g. -0.05) or "regime" (from, mult): multiplies the recruitment deviations.
- selectivity: list of fleet, from, sel (a selectivity()): the fleet's selectivity changes from year from (e.g. a gear switch or a minimum size).
- creep: list of fleet, from, rate: efficiency increases by rate per year from from (in addition to the fishery's own creep).
- M: list of from, mult: natural mortality is multiplied from from.

Several entries of one component are given as a list of lists.

Value

a list of class floss_treatment.

Examples

```
treatment("halve linefish effort",
          effort = list(fleet = "linefish", from = 2010, mult = 0.5))
treatment("strong 2015 year class",
          recruitment = list(type = "pulse", years = 2015, mult = 5))
```

---

trend_values *Deterministic effort trend values*

Description

Deterministic effort trend values

Usage

```
trend_values(trend, years)
```

Arguments

trend	a effort_trend().
years	integer vector of years.

Value

numeric vector, one value per year.

Examples

```
trend_values(effort_trend(fhi = 2, flo = 1, breaks = c(0.5, 0.8)), 1969:2025)
```

`write_spec` *Write and read FLOSS specs as YAML*

Description

Species and fishery specs are stored as plain YAML so that they can be edited by hand, shared between trainers and kept under version control.

Usage

```
write_spec(x, file)

read_spec(file)
```

Arguments

`x`	a `floss_species` or `floss_fishery`.
`file`	path to a `.yaml` file.

Value

`write_spec()` returns `file` invisibly; `read_spec()` returns the spec, re-validated through its constructor.

Examples

```
f <- tempfile(fileext = ".yaml")
write_spec(reef_fishery(), f)
read_spec(f)
```

Index

alk_obs, 3
as_flr, 4
as_flr(), *14*

build_om, 5
build_om(), *6*, *44*
build_oms, 6
build_oms(), *48*

caa_obs, 7
caa_obs(), *9*, *31*
calibrate_msy, 8
catch_curve, 8
catch_curve_length, 9
catch_fleet, 10
catch_table, 11
catch_table(), *14*
cpue_check, 12

delete_species *(*species_library*)*, 50

effort_sim, 12
effort_sim(), *5*, *6*
effort_table *(*catch_table*)*, 11
effort_trend, 13
effort_trend(), *16*, *54*
export_data, 14
export_data(), *28*
export_truth, 15

ffwd(), *5*
fishery, 15
fishery(), *6*, *12*, *25*, *28*, *32*, *44*, *48*
fishery_years *(*fishery*)*, 15
fit_growth, 17
fit_growth(), *9*
fit_lw, 17
fit_maturity, 18
FLife::lhPar(), *50*
floss.r4col, 19
floss_gears, 19
floss_guide, 20
floss_tip, 21
FLRef::alk.sample(), *26*
FLRef::build_gear(), *19*
FLRef::combine_sel_age(), *8*
FLRef::computeFbrp(), *6*
FLRef::fudc(), *13*
FLRef::lfd.sim(), *27*
FLRef::plotAdvice(), *39*
FLRef::ploteq(), *32*
FLRef::sel_la(), *47*, *51*
FLRef::updsr(), *8*

knitr::purl(), *20*

length_indicators, 21
length_indicators(), *36*
lfd_aggregate, 22
lfd_aggregate(), *4*, *14*, *38*
lfd_annual, 23
lfd_annual(), *22*, *23*, *47*
load_fishery, 24
load_species *(*species_library*)*, 50

M_empirical, 24

nominal_cpue *(*catch_table*)*, 11

oem_settings, 25
oem_settings(), *19*, *28*, *43–45*, *48*, *53*
one_gear_fishery, 27
one_gear_fishery(), *28*
one_gear_species, 28
one_gear_species(), *27*

planted_artefacts, 28
plot_aggregate, 29
plot_availability, 30
plot_bio, 30
plot_caa, 31
plot_effort, 32

plot_eq, 32
plot_eq_selectivity, 33
plot_fit, 34
plot_fleets, 34
plot_indicators, 35
plot_kobe, 35
plot_length_indicators, 36
plot_lfd, 37
plot_obs, 38
plot_om, 39
plot_seasons, 39
plot_srr, 40
plot_ssb_vb, 40
plot_treatment, 41
plot_truth, 41

read_dataset, 42
read_dataset(), *4*, *7*, *10*, *12*, *17, 18*, *22*, *30, 31*, *35*, *37–39*
read_spec *(*write_spec*)*, 55
reef_fishery, 42
reef_oem, 43
run_app, 43
run_treatment, 44
run_treatment(), *53*

sample_bio *(*sample_data*)*, 45
sample_catch *(*sample_data*)*, 45
sample_cpue *(*sample_data*)*, 45
sample_data, 45
sample_data(), *11*, *48*
sample_effort *(*sample_data*)*, 45
sample_lfd *(*sample_data*)*, 45
sample_survey *(*sample_data*)*, 45
save_species *(*species_library*)*, 50
save_species(), *50*
season_design, 46
season_design(), *26*, *38*, *46*
selectivity, 47
selectivity(), *28*, *33*, *49*, *53, 54*
shiny::runApp(), *43*
simulate_fishery, 48
species, 49
species(), *6*, *8*, *16*, *28*, *33*, *44*, *48*, *50, 51*
species_lhpar, 50
species_lhpar(), *51*
species_library, 50
species_sel, 51
species_sel(), *8*

status_table, 52
survey_settings, 52
survey_settings(), *26*

treatment, 53
treatment(), *6*, *28*, *44*
trend_values, 54

validate_fishery *(*fishery*)*, 15
validate_species *(*species*)*, 49

write_spec, 55