Package: cpueTargeting (via r-universe)

November 5, 2026

Title Simulators and Tools for CPUE Targeting and Standardization

Version 0.4.0

Description Multispecies operating models for exploring spatial and vertical targeting and catch-per-unit-effort standardization. Includes the multi-habitat cluster example of Winker et al. (2014), the four-species bathymetric Run 22 model based on the spatial simulator of Thorson et al. (2017) <doi:10.1139/cjfas-2015-0598> and Ono et al. (2015) <doi:10.1016/j.fishres.2015.05.021>, and its extension to a longline tuna fishery with species vertical distributions and hook depth by hooks-between-floats, following the habitat-based approach of Bigelow et al. (2002). Provides a step-by-step toolbox for real catch and effort data: data formatting, cumulative species curves, fleet and catch composition exploration, habitat-based subsetting (Stephens and MacCall 2004), and targeting covariates from the catch composition (principal components with the Optimal Coordinates test, and clustering into fishing tactics).

License GPL-3 + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Depends R (>= 4.1.0)

Imports ggplot2 (>= 3.4.0), grDevices, grid, stats, utils

Suggests cluster, knitr, mgcv, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

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

Date/Publication 2026-10-14 12:05:18 UTC

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

RemoteRef HEAD

RemoteSha 39090fba788bf2ecbd19ea6240b02bc2d4fe61c7

Contents

add_targeting . . . 3
as_set_data . . . 4
bathymetric_settings . . . 5
bind_comparisons . . . 5
cluster_tactics . . . 6
composition_pca . . . 7
compute_oc . . . 8
cpue_data . . . 9
cpue_diagnostics . . . 11
cpue_index . . . 11
cpue_matrix . . . 12
export_index . . . 12
fit_cpue . . . 13
influence_cpue . . . 15
logbook_settings . . . 16
logbook_truth . . . 17
nominal_cpue . . . 17
plot_availability . . . 18
plot_biplot . . . 18
plot_cluster_2014 . . . 19
plot_clusters . . . 19
plot_composition . . . 20
plot_cpue_map . . . 21
plot_diagnostics . . . 21
plot_effects . . . 22
plot_fleet . . . 22
plot_index . . . 23
plot_indices . . . 23
plot_influence . . . 24
plot_logbook_om . . . 24
plot_nominal . . . 25
plot_oc . . . 25
plot_om . . . 26
plot_pcs . . . 26
plot_replicates . . . 27
plot_residuals . . . 27
plot_sections . . . 28
plot_species_curve . . . 29
plot_step . . . 30
plot_stephens_maccall . . . 30
plot_targeting . . . 31
plot_thermocline . . . 31
plot_thermocline_effect . . . 32
plot_tuna_vertical_distributions . . . 32
plot_vessel_effects . . . 33
plot_vessel_profile . . . 34

ppc_cpue . . . 34
read_replicates . . . 37
run_replicates . . . 37
schaefer_update . . . 38
select_models . . . 39
select_species . . . 39
select_tactics . . . 40
select_vessels . . . 42
simulate_bathymetric . . . 43
simulate_cluster . . . 45
simulate_logbook . . . 46
simulate_tuna . . . 47
species_curve . . . 48
standardize_cpue . . . 49
stephens_maccall . . . 51
subset.cpue_data . . . 52
tactic_table . . . 53
tuna_settings . . . 53
vessel_summary . . . 55

Index **56**

`add_targeting` *Add targeting covariates: PC scores and fishing tactics*

Description

One step that runs `composition_pca()`, the Optimal Coordinates test and `cluster_tactics()`, and adds the results to the data:

- `PC1`, `PC2`, ...: the PC scores of the catch composition (all PCs are added; the DPC approach uses the first `n_pc`);
- `FT`: the fishing tactic (cluster) as a factor.

Usage

```
add_targeting(
  x,
  n_pc = NULL,
  n_clusters = NULL,
  method = c("clara", "kmeans", "hclust"),
  on = c("composition", "pcs"),
  transform = "fourth_root",
  species = x$species,
  seed = 1,
  ...
)
```

Arguments

x	A cpue_data object, ideally after select_species().
n_pc	Number of PCs to use (default: Optimal Coordinates test).
n_clusters	Number of clusters (default: n_pc + 1).
method, on, seed	Passed to cluster_tactics().
transform, species	Passed to composition_pca().
...	Further arguments for cluster_tactics().

Details

These are the targeting covariates of the DPC and Cluster models. Check them with plot_oc(), plot_biplot(), tactic_table() and plot_targeting() before using them.

Value

The cpue_data object with the new columns and a targeting element (PCA, settings and cluster composition).

Examples

```
## Not run:
x <- add_targeting(x)              # OC test decides n_pc and n_clusters
plot_oc(x)
plot_biplot(x)
tactic_table(x)
plot_targeting(x)$time

## End(Not run)
```

---

as_set_data *Reshape set-by-species records (long) to one row per set*

Description

Reshape set-by-species records (long) to one row per set.

Usage

```
as_set_data(data, value = "CPUE")
```

Arguments

data	Long data with one row per set and species, e.g. sim$Data.
value	Column to spread across species (default "CPUE").

Value

A data frame with set-level columns and one column per species.

Examples

```
## Not run:
sim <- simulate_tuna(seed = 1)
sets <- as_set_data(sim$Data)
head(sets)

## End(Not run)
```

`bathymetric_settings` *Default settings for the four-species bathymetric scenario (Run 22)*

Description

Settings of the original `OMcpueSIM22.R` example (scenario 2, targeting changes). Species distributions are driven by bathymetry and distance; the model has no vertical component. Prices change in three 10-year blocks: years 1-10 favour species 1, years 11-20 species 2, and years 21-30 species 4.

Usage

```
bathymetric_settings()
```

Value

A named list accepted by `simulate_bathymetric`.

`bind_comparisons` *Combine several standardisation results into one*

Description

Useful to compare fits made with different settings (e.g. spatial resolutions) in one plot. Give each call a different `label` so that the model names are unique.

Usage

```
bind_comparisons(...)
```

Arguments

`...`	Outputs of `standardize_cpue`.

Value

An object like the output of standardize_cpue with the indices, diagnostics and formulas of all inputs (data and pca are taken from the first).

Examples

```
## Not run:
a <- standardize_cpue(om, models = c("Base", "HBF+S"))
b <- standardize_cpue(om, models = "Base+S", spatial_res = 1,
                              label = " (1x1)")
plot_diagnostics(bind_comparisons(a, b))

## End(Not run)
```

---

cluster_tactics *Cluster operations into fishing tactics*

Description

Cluster operations into fishing tactics

Usage

```
cluster_tactics(
  pca,
  n_clusters = pca$oc$n_clusters,
  method = c("clara", "kmeans", "hclust"),
  on = c("composition", "pcs"),
  n_pc = pca$oc$npc,
  seed = 1,
  samples = 100,
  sampsize = 250
)
```

Arguments

pca	Output of composition_pca().
n_clusters	Number of clusters (default: OC result + 1, Winker et al. 2014).
method	"clara" (default, as in the CPUE guidelines; needs the cluster package), "kmeans", or "hclust" (Ward's method; only for up to 20,000 operations).
on	Cluster the transformed composition ("composition", default, as in the guidelines) or the first n_pc PC scores ("pcs").
n_pc	Number of PCs when on = "pcs".
seed	Random seed (CLARA and k-means are stochastic).
samples, sampsize	CLARA settings: number of samples and sample size (default 250, as in the guidelines).

Value

An integer vector of clusters, numbered by decreasing size.

`composition_pca` *Principal components of the catch composition*

Description

The catch of each operation is converted to proportions by weight (to remove the effect of catch volume), transformed (fourth root by default, which down-weights dominant species), and decomposed with a PCA on the correlation matrix. The PC scores describe the catch composition of each operation on a few continuous axes: the basis of both the DPC approach (scores as smooth covariates) and of clustering into fishing tactics (Winker et al. 2013, 2014).

Usage

```
composition_pca(
  x,
  species = x$species,
  transform = c("fourth_root", "sqrt", "none", "presence"),
  scale = TRUE
)
```

Arguments

`x`	A `cpue_data` object, ideally after `select_species()`.
`species`	Species to use (default: all in `x`).
`transform`	Transformation of the proportions: `"fourth_root"` (default, as in the CPUE guidelines), `"sqrt"`, `"none"`, or `"presence"` (binary presence/absence instead of proportions).
`scale`	Logical; PCA on the correlation matrix (default). The Kaiser-Guttman part of the OC test (eigenvalue >= 1) assumes this.

Details

The number of non-trivial PCs is chosen with the Optimal Coordinates test (`compute_oc()`).

Value

An object of class `cpue_pca` with `pca` (the `stats::prcomp()` result), `scores`, `eigen` (eigenvalues and shares of variance), `oc` (result of `compute_oc()`), `composition` (the transformed matrix, used for clustering), `species`, `labels` and `transform`.

`compute_oc` *Optimal Coordinates test for the number of non-trivial components*

Description

Non-graphical solution to Cattell's scree test (Raiche et al. 2013, as in the R package nFactors), combined with the Kaiser-Guttman rule. For each component i, the eigenvalue is predicted by the straight line through eigenvalues $i+1$ and k (the last); component i is retained if its eigenvalue is at least this prediction and at least 1. The number of non-trivial components is the number retained in sequence from the first. Winker et al. (2014) take the number of clusters as this number plus one.

Usage

```
compute_oc(eig)
```

Arguments

`eig`	Eigenvalues in decreasing order.

Details

Same rule as `computeOC()` in the CPUE guidelines, with the last comparisons defined so that the test also works when every component passes (at most $k-1$ components are retained).

Value

A list with `npc`, `n_clusters` (`npc + 1`), `tests` (eigenvalue, OC prediction and whether each component passes) and `lines` (the OC lines for plotting).

References

Raiche, G., Walls, T.A., Magis, D., Riopel, M. and Blais, J.-G. (2013) Non-graphical solutions for Cattell's scree test. Methodology 9: 23-29.

Examples

```
compute_oc(c(2.1, 1.3, 0.4, 0.2))$npc
```

cpue_data *Prepare catch and effort data for analysis*

Description

First step of the real-data workflow. Brings logbook or observer records into one standard object: one row per fishing operation (trip, set or haul) and one column per species with the catch. All later steps (`species_curve()`, `stephens_maccall()`, `add_targeting()`, ...) work on this object, so they know which columns are species and which play the roles of year, vessel, effort and position.

Usage

```
cpue_data(
  data,
  year,
  species,
  catch = NULL,
  effort = NULL,
  id = NULL,
  vessel = NULL,
  lon = NULL,
  lat = NULL,
  covariates = NULL,
  synonyms = NULL,
  drop_species = NULL
)
```

Arguments

`data`	A data frame.
`year`	Name of the year column (numeric).
`species`	Long format: name of the species column. Wide format: names of the species catch columns.
`catch`	Long format only: name of the catch column (weight or numbers).
`effort`	Optional name of the effort column (e.g. hours fished, hooks/1000, days). Operations with missing or non-positive effort are removed. Without effort, the catch per operation is the CPUE.
`id`	Optional name(s) of the column(s) identifying an operation.
`vessel`	Optional name of the vessel column.
`lon, lat`	Optional names of position columns (decimal degrees or grid coordinates).
`covariates`	Optional names of other operation-level columns to keep (e.g. `"month"`, `"port"`, `"HBF"`, `"gear"`).
`synonyms`	Optional named character vector to merge species before anything else, e.g. `c("Trichiurus lepturus" = "Lepidopus caudatus")` (names: as recorded; values: as they should be counted).
`drop_species`	Optional species names to drop entirely (e.g. bait or discarded categories).

Details

Two input formats are accepted:

- **long**: one row per operation and species, e.g. the usual logbook extract with columns `trip`, `year`, `vessel`, `species`, `catch_kg`. Give the name of the species column in `species` and of the catch column in `catch`. Catches of the same species within an operation are summed.
- **wide**: one row per operation with one column per species. Give the names of the species columns in `species` and leave `catch` empty.

Operations are identified by `id` (one or more columns, e.g. `c("vessel", "trip")`). Without `id`, every unique combination of the operation-level columns is treated as one operation, which can merge different operations that happen to share all values; a message reports this case.

Species names are converted to syntactic column names (e.g. `"Pagellus bogaraveo"` becomes `Pagellus.bogaraveo`) so that they can be used in model formulas; the original names are kept in `species_labels` and used in plots.

Value

An object of class `cpue_data`: a list with

- `data`: one row per operation with the role columns, `effort` if given, and one catch column per species;
- `species`: the species column names, ordered by total catch;
- `species_labels`: original species names (named by column name);
- `roles`: which column plays which role;
- `log`: a record of the processing steps and how many operations each removed (printed by `print()`).

See Also

`species_curve()`, `select_species()`, `add_targeting()`

Examples

```
logbook <- data.frame(
  trip = rep(1:3, each = 2), year = rep(c(2020, 2020, 2021), each = 2),
  vessel = rep(c("A", "B", "A"), each = 2), hours = rep(c(5, 8, 6), each = 2),
  species = c("Pagellus bogaraveo", "Conger conger", "Pagellus bogaraveo",
              "Trichiurus lepturus", "Lepidopus caudatus", "Conger conger"),
  kg = c(20, 5, 12, 3, 4, 9))
x <- cpue_data(logbook, year = "year", species = "species", catch = "kg",
               effort = "hours", id = "trip", vessel = "vessel",
               synonyms = c("Trichiurus lepturus" = "Lepidopus caudatus"))
x
head(x$data)
```

cpue_diagnostics *Compare fitted CPUE models*

Description

One row per model (and per component for delta models) with the number of observations, AIC, BIC, deviance explained, effective degrees of freedom, the estimated Tweedie power p and the SD of the random vessel effect.

Usage

```
cpue_diagnostics(fit)
```

Arguments

fit	Output of fit_cpue().

Details

Information criteria are only comparable between models fitted to the same data with the same family. They tend to favour many targeting terms, because the catch composition is not independent of the response (Winker et al. 2014); use them together with the influence plots and the change in the index.

Value

A data frame.

cpue_index *Standardised index table*

Description

Standardised index table

Usage

```
cpue_index(fit, model = NULL)
```

Arguments

fit	Output of fit_cpue().
model	Model name (default: the last model).

Value

A data frame with `year`, `n` (operations), `nominal` (mean CPUE), `index` (standardised CPUE in the units of the data, at the reference covariate values), `relative` (index / its mean), `se_log` (standard error of the log index, the usual input to stock assessments), `cv`, and the 95% CI of the relative index.

`cpue_matrix` *CPUE of each operation*

Description

Catch divided by effort (or the catch itself when no effort column was given).

Usage

```
cpue_matrix(x, species = x$species)
```

Arguments

`x`	A `cpue_data` object.
`species`	Species column names (default: all).

Value

A matrix with one column per species.

`export_index` *Write the standardised index to a CSV file*

Description

Write the standardised index to a CSV file

Usage

```
export_index(fit, file, model = NULL, digits = 4)
```

Arguments

`fit`	Output of `fit_cpue()`.
`file`	Path of the CSV file.
`model`	Model name (default: the last model).
`digits`	Number of significant digits.

Value

The index table, invisibly.

fit_cpue *Standardise CPUE for one species with mgcv*

Description

Fits a set of GAMs or GAMMs (mgcv) to the CPUE of one species and derives a standardised annual index from each. This is the model step of the real-data workflow, after cpue_data(), select_species() and add_targeting().

Usage

```
fit_cpue(
  x,
  species,
  models = NULL,
  terms = NULL,
  vessel = FALSE,
  family = c("tweedie", "delta_lognormal", "delta_gamma"),
  n_pc = NULL,
  pc_k = 4,
  spatial_k = 30,
  offset = FALSE,
  average_over = c("FT", "month"),
  max_ref = 5000,
  add_models = NULL,
  engine = c("gam", "bam"),
  seed = 1,
  ...
)
```

Arguments

x	A cpue_data object (after add_targeting() for the DPC and Cluster models).
species	The species to standardise (column or original name).
models	Model names (see Details). Default: Base, Covariates (if terms is given), and DPC and Cluster (if targeting covariates are available).
terms	Character vector of additional model terms used in all models except Base, e.g. c('s(month, bs = "cc", k = 8)', "port"). They can use any column of x$data.
vessel	Logical; add a random vessel effect to every model.
family	"tweedie", "delta_lognormal" or "delta_gamma".
n_pc	Number of PCs in the DPC models (default: the number used by add_targeting()).
pc_k	Basis dimension of each PC smoother.
spatial_k	Basis dimension of the spatial smoother.

offset	Logical. FALSE (default): the response is CPUE (catch / effort). TRUE: the response is catch with offset(log(effort)) (Tweedie and the positive part of delta models).
average_over	Variables whose observed values are averaged over to compute the index (see Details).
max_ref	Maximum number of reference combinations per year (a random sample is used if there are more).
add_models	Optional named character vector of further right-hand sides; they must contain fyear, e.g. c(Interaction = "fyear + s(PC1, by = FT) + FT").
engine	"gam" (default) or "bam" (with discrete = TRUE, much faster for large data).
seed	Seed for sampling reference combinations.
...	Further arguments for mgcv::gam() or mgcv::bam(), e.g. knots = list(month = c(0.5, 12.5)) for a cyclic month smoother.

Details

Models. Every model contains the year effect (fyear, a factor) and differs in the targeting term:

- Base: year only;
- Covariates: year + terms (e.g. month, port, gear);
- DPC: year + terms + s(PC1) + ... + s(PCn) (direct principal components, Winker et al. 2013, 2014);
- Cluster: year + terms + FT (fishing tactic as a factor).

Append +S to a name to add the spatial smoother s(pos_x, pos_y) of the positions given in cpue_data() (e.g. "DPC+S"). Further models can be given as right-hand sides in add_models. With vessel = TRUE, every model gets a random vessel intercept s(vessel_re, bs = "re").

Families.

- "tweedie" (default): mgcv::tw(), a compound Poisson-gamma distribution with a point mass at zero. The power parameter p is estimated jointly with the model, also with random vessel effects, so the two-step approach (estimate p in a GAM, then fix it in gamm()) is not needed.
- "delta_lognormal", "delta_gamma": a binomial GAM for the probability of a positive catch and a lognormal (Gaussian on log CPUE) or gamma GAM for positive CPUE, with the same right-hand side. The index is the product of both.

Index. For each year, the expected CPUE is predicted for a reference set of covariate values and averaged: positions are averaged over all observed cells, the variables in average_over over their observed values (by default fishing tactic and month), other numeric covariates (e.g. PC scores) are set to their median and other factors to their most common level. Random vessel effects are set to zero (the average vessel). Standard errors of the log index come from the delta method using the Bayesian covariance matrix of the coefficients (Vp).

Value

An object of class `cpue_fit`: a list with `models` (for each model the right-hand side and the fitted GAM(s)), `indices` (annual index of every model: absolute and relative to its mean, log SE and 95% CI), `nominal` (nominal CPUE by year), `data` (the data used, with the model columns), `reference` (the reference set used for the index), `species`, `label` and `settings`. See `cpue_index()`, `cpue_diagnostics()`, `plot_index()`, `plot_influence()`.

References

Bentley, N., Kendrick, T.H., Starr, P.J. and Breen, P.A. (2012) Influence plots and metrics: tools for better understanding fisheries catch-per-unit-effort standardizations. ICES Journal of Marine Science 69: 84-88.

Examples

```
## Not run:
fit <- fit_cpue(x, species = "Pagellus bogaraveo",
                terms = 's(month, bs = "cc", k = 8)', vessel = TRUE,
                knots = list(month = c(0.5, 12.5)))
cpue_diagnostics(fit)
plot_index(fit)
plot_influence(fit, "DPC")
cpue_index(fit, "DPC")

## End(Not run)
```

`influence_cpue` *Influence of each model term on the index*

Description

Bentley et al. (2012): for term t with effect $f_t(x_i)$ on operation i, the influence in year y is $\exp(\overline{f_t}_y - \overline{f_t})$, the ratio between the expected CPUE given the covariate values of that year and given those of all years. A term has influence only if its covariate changes between years: values above 1 mean the term raised the nominal CPUE of that year (and the standardisation lowers it).

Usage

```
influence_cpue(fit, model = NULL, component = NULL)
```

Arguments

`fit`	Output of `fit_cpue()`.
`model`	Model name (default: the last model).
`component`	For delta models, `"positive"` (default) or `"binomial"`.

Value

A data frame with `term`, `year` and `influence`.

`logbook_settings` *Default settings for the logbook simulator*

Description

Ten species in a mixed demersal-pelagic fishery. Demersal species live along a continuous depth gradient and on a patchy substrate (reef patches within sediment), which gives four habitats as in Winker et al. (2014): shallow reef, deep reef, shallow sediment and deep sediment. Pelagic species follow sea surface temperature and distance offshore and migrate seasonally through the area. Species names are illustrative.

Usage

```
logbook_settings()
```

Details

Main elements (all can be changed):

- `species`: one row per species with `group` (demersal/pelagic), depth preference (`depth_mu`, `depth_cv`, lognormal), substrate affinity (`reef`, `sediment`, multipliers of density), SST preference (`sst_mu`, `sst_sd`, normal; `NA` for none), seasonal availability (`season_amp`, 0 = resident, 1 = absent at the low point; `season_peak`, month), `dispersal`, Schaefer `B0` and `r`, `B_start` (initial biomass / B0) and base `price`.
- `tactics`: catchability of each species by tactic (rows `Reef`, `Sediment`, `Pelagic`), i.e. gear selectivity.
- `price_periods`: years at which prices change and multipliers for demersal and pelagic species in each period (default: demersal premium, then a pelagic boom, then neutral prices).
- Fleet: `vessel_types` (numbers of vessels by type), `home` (home tactics of each type), `loyalty` (share of a type's trips restricted to its home tactics; the other trips choose among all tactics by expected value), vessel catchability CV, late entries with efficiency creep (`entry_share`, `creep`), early exits (`exit_share`). The default has demersal and pelagic specialists with loyalty 0.8 and mixed vessels that choose purely by expected value. (`allowed`, from earlier versions, is still accepted and means `home` with loyalty 1.)
- `trips`: trips per year (default: an increase, cut by 20% with the closure); `preference`: how strongly effort concentrates on the most valuable cells and tactics.
- `closure`: year, depth range and tactics of a spatial closure (default: sediment gear banned deeper than 300 m from year 17).
- `logbook`: how trips are reported (see `simulate_logbook()`).

Value

A named list for `simulate_logbook()`.

logbook_truth *True relative biomass of one species from simulate_logbook()*

Description

True relative biomass of one species from simulate_logbook()

Usage

```
logbook_truth(sim, species)
```

Arguments

sim	Output of simulate_logbook().
species	Species name.

Value

A data frame with year and true, e.g. for plot_index(true =).

nominal_cpue *Nominal CPUE by year*

Description

Nominal CPUE by year

Usage

```
nominal_cpue(x, species = x$species)
```

Arguments

x	A cpue_data object.
species	Species (column or original names; default: all).

Value

A data frame with species, year, n (operations), positive (proportion of operations catching the species), mean (mean CPUE), mean_positive (mean CPUE of positive operations), median and relative (mean scaled to its average over years).

plot_availability *Vertical availability by HBF over time*

Description

Mean realised vertical availability (overlap between hooks and fish) of the sets fishing with each HBF level, by year and species. With a fixed thermocline the lines are flat apart from changes in where the fleet fishes; with a moving thermocline the catchability of a given HBF changes from year to year, which an HBF factor cannot absorb.

Usage

```
plot_availability(sim)
```

Arguments

sim	Output of simulate_tuna.

Value

A ggplot object.

plot_biplot *PCA biplot of the catch composition*

Description

Operations as points (coloured by fishing tactic, by default) and species loadings as arrows, on two chosen PCs.

Usage

```
plot_biplot(x, pcs = c(1, 2), colour_by = "FT", max_points = 5000)
```

Arguments

x	A cpue_data object after add_targeting() (or a cpue_pca, then points are not coloured).
pcs	The two PCs to show.
colour_by	Column of x$data used to colour points (default "FT"; NULL for none).
max_points	Plot a random sample of at most this many operations.

Value

A ggplot object.

plot_cluster_2014 *Plot the 2014 cluster simulation*

Description

Creates diagnostic ggplot figures for annual biomass, annual habitat effort allocation, and trip-level catches by species.

Usage

```
plot_cluster_2014(simulation)
```

Arguments

simulation	Output from simulate_cluster().

Value

A named list of three ggplot objects: biomass, effort, and catches.

Examples

```
## Not run:
sim <- simulate_cluster(seed = 1)
plots <- plot_cluster_2014(sim)
plots$biomass

## End(Not run)
```

plot_clusters *What do the catch-composition clusters capture?*

Description

Diagnostic plots for the fishing tactics (clusters) used by the Cluster models, following the CPUE guidelines: the species composition of each cluster, the share of sets in each cluster by year, the most common cluster in each cell, the realised hook depth by cluster (tuna model only) and the sets in PC space coloured by cluster. Clusters are labelled by their number and main species, e.g. "2: ALB".

Usage

```
plot_clusters(comparison, switch_years = NULL, max_points = 5000)
```

Arguments

comparison	Output of standardize_cpue (or of bind_comparisons).
switch_years	Optional years to mark with vertical lines in the time plot, e.g. the years in which prices change (c(10.5, 20.5)).
max_points	Maximum number of sets drawn in the PC plot.

Details

hook_depth is latent truth from the operating model; it is used here only to show what the clusters pick up, not in any standardisation model.

Value

A list of ggplots: composition, time, space, depth (tuna model only) and pca.

Examples

```
## Not run:
sim <- simulate_tuna(seed = 123)
fit <- standardize_cpue(sim, models = c("Base", "Cluster+S"), engine = "bam")
p <- plot_clusters(fit, switch_years = c(10.5, 20.5))
p$composition
p$time

## End(Not run)
```

---

plot_composition *Plot the catch composition over time*

Description

Share of each species in the total catch (type = "weight") or in the positive records (type = "occurrence") by year or by another grouping column. Changes in composition are the first sign of changes in targeting (or in relative abundance).

Usage

```
plot_composition(x, type = c("weight", "occurrence"), by = NULL)
```

Arguments

x	A cpue_data object.
type	"weight" or "occurrence".
by	Grouping column (default: the year column).

Value

A ggplot object.

plot_cpue_map *Map of operations and CPUE*

Description

Map of operations and CPUE

Usage

```
plot_cpue_map(x, species = NULL, res = 1, by_period = NULL)
```

Arguments

x	A cpue_data object with lon and lat.
species	Optional species for the CPUE map (column or original name); without it, the number of operations per cell is mapped.
res	Cell size in units of lon/lat (e.g. 0.5 degrees).
by_period	Optional number of years per panel (e.g. 5) to see shifts in fishing grounds.

Value

A ggplot object.

plot_diagnostics *Compare the accuracy of standardised indices*

Description

Bar chart of the root mean squared error of the log index (relative to the true biomass index) by model and species.

Usage

```
plot_diagnostics(comparison, models = NULL)
```

Arguments

comparison	Output of standardize_cpue, or a named list of such outputs (one per scenario), shown as differently coloured bars.
models	Optional subset of model names to show.

Value

A ggplot object.

plot_effects *Partial effects of the covariates*

Description

Effect of each covariate on the expected CPUE (relative to its mean effect), with the other covariates at their reference values and the random vessel effect at zero.

Usage

```
plot_effects(fit, model = NULL, component = NULL)
```

Arguments

fit	Output of fit_cpue().
model	Model name (default: the last model).
component	For delta models, "positive" (default) or "binomial".

Value

A list of ggplots: numeric (smooth effects, e.g. PCs and month, with 95% CIs), factor (factor effects, e.g. fishing tactic), and space (map of the spatial effect over the observed cells), each only if present.

plot_fleet *Overview of the fleet and the records*

Description

A set of exploratory plots of how the data are distributed over years, vessels and covariates, i.e. the non-random allocation of effort that standardisation has to deal with.

Usage

```
plot_fleet(x, by = NULL, max_vessels = 60)
```

Arguments

x	A cpue_data object.
by	Optional covariate (e.g. "month", "port", "HBF") for the by_covariate plot.
max_vessels	Show at most this many vessels in the vessel plot (the ones with most records).

Value

A list of ggplots: records (operations per year), vessels (vessel x year tile plot of the number of operations, showing fleet turnover), vessel_years (vessels active per year and new entries) and, if by is given, by_covariate (share of operations by covariate level over years).

plot_index *Plot standardised indices*

Description

Plot standardised indices

Usage

```
plot_index(fit, models = NULL, ci = TRUE, nominal = TRUE, true = NULL)
```

Arguments

fit	Output of fit_cpue().
models	Models to show (default: all).
ci	Logical; show 95% confidence intervals.
nominal	Logical; show the nominal CPUE (black dashed line).
true	Optional data frame with columns year and true (e.g. the true biomass of a simulation), scaled to mean 1 and drawn in black.

Value

A ggplot object (relative indices, each scaled to its mean).

plot_indices *Plot standardised indices against the true biomass trend*

Description

Plot standardised indices against the true biomass trend.

Usage

```
plot_indices(comparison, models = NULL, ci = FALSE,
  type = c("lines", "smooth", "points+smooth"), span = 0.5)
```

Arguments

comparison	Output of standardize_cpue, or a named list of such outputs (one per scenario), which are shown as rows.
models	Optional subset of model names to show.
ci	Logical; show 95% confidence intervals.
type	How to draw the indices: "lines" (default) joins the annual values; "smooth" shows only a loess smoother of each index, which makes trends easier to compare; "points+smooth" shows the annual values as points with the smoother on top.
span	Smoothing span of the loess smoother (larger = smoother).

Value

A ggplot object.

`plot_influence` *Plot the influence of each model term*

Description

Plot the influence of each model term

Usage

```
plot_influence(fit, model = NULL, component = NULL)
```

Arguments

`fit`	Output of `fit_cpue()`.
`model`	Model name (default: the last model).
`component`	For delta models, `"positive"` (default) or `"binomial"`.

Value

A ggplot object.

`plot_logbook_om` *Diagnostic plots of the logbook simulator*

Description

Diagnostic plots of the logbook simulator

Usage

```
plot_logbook_om(sim)
```

Arguments

`sim`	Output of `simulate_logbook()`.

Value

A list of ggplots: `env` (depth with reef patches), `sst`, `density` (initial density by species), `tactics` (true share of trips by tactic over years), `effort` (trips by vessel type), `prices`, and `cpue` (nominal CPUE from the logbook against the true biomass).

plot_nominal *Plot nominal CPUE*

Description

Plot nominal CPUE

Usage

```
plot_nominal(x, species = x$species)
```

Arguments

x	A cpue_data object.
species	Species (column or original names; default: all).

Value

A list of ggplots: `index` (relative nominal CPUE by year), `positive` (proportion of positive operations by year), `distribution` (histogram of log CPUE of positive operations) and `boxplot` (CPUE of positive operations by year, log scale).

plot_oc *Plot the scree and the Optimal Coordinates test*

Description

Plot the scree and the Optimal Coordinates test

Usage

```
plot_oc(x)
```

Arguments

x	A cpue_data object after `add_targeting()`, or a cpue_pca.

Value

A ggplot object.

plot_om *Diagnostic plots for an operating-model run*

Description

Diagnostic plots for an operating-model run.

Usage

```
plot_om(sim)
```

Arguments

sim	Output of simulate_bathymetric or simulate_tuna.

Value

A named list of ggplots: Env (bathymetry, or SST for the tuna model), Thermocline (fixed thermocline anomaly; tuna model only), SpDens (initial densities), Hist (CPUE frequencies), CPUE (nominal versus true index) and, for the tuna model, HBF (annual use of each HBF level) and Avail (see plot_availability). Depth is kept as an alias of Env for the bathymetric model.

plot_pcs *What do the principal components of the catch composition capture?*

Description

Plots the PC scores used by the DPC models in four ways: the species loadings (what each PC means), mean scores by cell (where each tactic is used), mean scores against the realised hook depth of the set for each HBF (how the PCs track vertical targeting; tuna model only), and mean scores by year (how targeting changes over time).

Usage

```
plot_pcs(comparison, pcs = c("PC1", "PC2"), depth_bin = 20, min_sets = 30)
```

Arguments

comparison	Output of standardize_cpue (or of bind_comparisons).
pcs	Principal components to show.
depth_bin	Width of the hook-depth bins (m).
min_sets	Minimum number of sets for a depth bin to be shown.

Details

`hook_depth` is latent truth from the operating model; it is used here only to show what the PCs pick up, not in any standardisation model.

Value

A list of ggplots: `loadings`, `space`, `depth` (tuna model only) and `time`.

`plot_replicates` *Accuracy of standardised indices across replicates*

Description

Boxplots of the RMSE of the log index over seeds, by model, scenario and species.

Usage

```
plot_replicates(replicates, models = NULL)
```

Arguments

`replicates`	Output of `run_replicates`.
`models`	Optional subset of model names to show.

Value

A ggplot object.

`plot_residuals` *Residual diagnostics*

Description

For the Tweedie family, randomised quantile residuals (Dunn and Smyth 1996) are computed by simulation from the fitted distribution, so that they are standard normal if the model is correct, also with many zeros. For delta models, deviance residuals of the chosen component are shown.

Usage

```
plot_residuals(
  fit,
  model = NULL,
  component = NULL,
  by = NULL,
  nsim = 100,
  max_points = 5000
)
```

Arguments

fit	Output of fit_cpue().
model	Model name (default: the last model).
component	For delta models, "positive" (default) or "binomial".
by	Optional column of the data to plot residuals against (e.g. "FT", "month", "vessel").
nsim	Number of simulations per observation for quantile residuals.
max_points	Maximum number of points in scatter plots.

Value

A list of ggplots: qq, fitted (residuals against log fitted values with a smoother), year (by year) and, if by is given, by.

References

Dunn, P.K. and Smyth, G.K. (1996) Randomized quantile residuals. Journal of Computational and Graphical Statistics 5: 236-244.

---

plot_sections *Species distributions in three dimensions: sections and depth layers*

Description

Combines the horizontal density of each species with its vertical profile in each cell (including the thermocline anomaly of the chosen year) and shows the result as a latitude x depth section (density averaged over x) and as maps of the density in depth layers, species by layer. Horizontal densities are the initial (unfished) distribution.

Usage

```
plot_sections(sim, year = 1, layers = c(0, 100, 200, 300, 500))
```

Arguments

sim	Output of simulate_tuna.
year	Year whose thermocline anomaly is used (matters only when the thermocline moves between years).
layers	Depth-layer boundaries (m) for the layer maps.

Value

A list of two ggplots, section and layers.

Examples

```
## Not run:
sim <- simulate_tuna(seed = 1)
p <- plot_sections(sim)
p$section
p$layers

## End(Not run)
```

`plot_species_curve` *Plot the cumulative species curve*

Description

Plot the cumulative species curve

Usage

```
plot_species_curve(
  x,
  threshold = 0.95,
  by = c("occurrence", "weight"),
  max_species = 30
)
```

Arguments

`x`	A `cpue_data` object.
`threshold`	Cumulative share used to select species (drawn as a line; the selected species are coloured).
`by`	`"occurrence"` (default) or `"weight"`.
`max_species`	Show at most this many species.

Value

A ggplot object.

plot_step *Step plot of the standardisation*

Description

Bentley et al. (2012): each panel shows the index of a model (in the order fitted) together with the index of the previous model (grey, dashed), so that the effect of adding each term on the index is visible. Fit the models in the order in which terms are added, e.g. `models = c("Base", "Covariates", "DPC")`.

Usage

```
plot_step(fit, models = NULL)
```

Arguments

`fit`	Output of `fit_cpue()`.
`models`	Models in order (default: all, as fitted).

Value

A ggplot object.

plot_stephens_maccall *Diagnostic plots for Stephens and MacCall subsetting*

Description

Diagnostic plots for Stephens and MacCall subsetting

Usage

```
plot_stephens_maccall(sm)
```

Arguments

`sm`	Output of `stephens_maccall()`.

Value

A list of ggplots: `threshold` (difference between observed and predicted presences against the threshold, with the critical value), `coefficients` (species effects on the presence of the target with 95% CIs), `probability` (distribution of predicted probabilities for operations with and without the target) and `kept` (share of operations kept by year; a trend means the subset changes over time).

plot_targeting *What the targeting covariates capture*

Description

What the targeting covariates capture

Usage

```
plot_targeting(x, by = NULL, res = 1)
```

Arguments

x	A cpue_data object after add_targeting().
by	Optional column (e.g. "month", "HBF", "port", "vessel") to compare PC scores and tactics across its levels.
res	Cell size for the maps (if positions are available).

Value

A list of ggplots:

- loadings: species loadings on the PCs used;
- composition: species composition of each tactic;
- time: share of tactics by year;
- pcs_time: median and 10-90% range of the PC scores by year;
- by: PC1 scores and tactic shares by the levels of by (if given);
- map_tactic, map_pc: most frequent tactic and mean PC1 per cell (if positions are available).

plot_thermocline *Plot the thermocline anomaly*

Description

Plot the thermocline anomaly.

Usage

```
plot_thermocline(sim, years = NULL)
```

Arguments

sim	Output of simulate_tuna.
years	Years to map (default: six years spread over the series).

Value

A list of two ggplots: `series` (mean anomaly by year with the 10-90% range across cells) and `maps` (anomaly fields in selected years).

---

plot_thermocline_effect

Show how a thermocline shift moves species relative to the hooks

---

Description

Species depth profiles for a shallow, average and deep thermocline, with the expected hook-depth profiles of each HBF level in grey. When the thermocline is shallow, deep-set hooks fish below the deep species; when it is deep, shallow sets miss them.

Usage

```
plot_thermocline_effect(settings = tuna_settings(), shifts = c(-40, 0, 40))
```

Arguments

`settings`	Settings list, usually from `tuna_settings`.
`shifts`	Thermocline anomalies to show (m, positive = deeper).

Value

A ggplot object.

Examples

```
plot_thermocline_effect()
```

---

plot_tuna_vertical_distributions

Plot species vertical profiles and hook-depth profiles by HBF

Description

One panel per HBF level. Grey bars show where the hooks of a set with that HBF fish (the expected hook-depth distribution, drawn like Fig. 2 of Bigelow et al. 2002); the coloured translucent areas show where each species lives (thermocline anomaly of zero). Where a species' area overlaps the grey bars, that set can catch it. The number next to each species is its vertical availability to that HBF, i.e. the overlap that multiplies catchability in the operating model (see `plot_availability`).

Usage

```
plot_tuna_vertical_distributions(settings = tuna_settings(), overlay = TRUE,
  file = NULL)
```

Arguments

settings	Settings list, usually from tuna_settings.
overlay	Logical; TRUE (default) overlays the species on each HBF panel; FALSE shows the species in a separate first panel.
file	Optional file path; if supplied, a 300-dpi PNG is saved.

Value

A ggplot object.

Examples

```
plot_tuna_vertical_distributions()
```

---

plot_vessel_effects *Plot the random vessel effects*

Description

Plot the random vessel effects

Usage

```
plot_vessel_effects(fit, model = NULL, component = NULL)
```

Arguments

fit	Output of fit_cpue().
model	Model name (default: the last model).
component	For delta models, "positive" (default) or "binomial".

Value

A ggplot of the estimated vessel effects (log scale) ordered by size, with bands of one and 1.96 estimated SDs.

plot_vessel_profile *Catch profile of each vessel*

Description

Share of each vessel's total catch by species or species group, with vessels ordered by the share of the first group. Specialists appear at either end, mixed vessels in between; useful to define indicator vessels and to see how far the fleet is split into specialists.

Usage

```
plot_vessel_profile(x, groups = NULL, by = NULL, max_vessels = 80)
```

Arguments

x	A cpue_data object with a vessel column.
groups	Optional named list of species vectors (column or original names), e.g. list(demersal = ..., pelagic = ...). Species not in any group are pooled as other. Default: each species separately.
by	Optional named vector with one label per vessel (names = vessel ids), e.g. a vessel type, used to facet the plot.
max_vessels	Show at most this many vessels (those with most records).

Value

A ggplot object; the data (one row per vessel and group) are in its data element.

See Also

vessel_summary(), select_vessels()

ppc_cpue *Predictive check of the error distribution*

Description

Compares the observed CPUE with replicate data sets simulated from the fitted models, to check whether the error distribution (Tweedie, delta-lognormal or delta-gamma) reproduces the share of zeros, the upper tail and the shape of the positive catches, overall and by year (or any other grouping).

Usage

```
ppc_cpue(
  fit,
  models = NULL,
  n_sim = 50,
  by = "year",
  param_uncertainty = TRUE,
  probs = c(0.9, 0.99),
  bins = 30,
  scale = c("fourth_root", "log", "raw"),
  seed = 1
)
```

Arguments

`fit`	Output of `fit_cpue()`.
`models`	Models to check (default: all).
`n_sim`	Number of replicate data sets.
`by`	Grouping for the summaries: `"year"` (default), any column of `fit$data` (e.g. `"FT"`, `"month"`) or `NULL` for all operations together.
`param_uncertainty`	Logical; draw the coefficients of each replicate from their approximate posterior.
`probs`	Upper quantiles of positive CPUE to compare.
`bins`	Number of bins of the histogram of positive CPUE (the zeros form an extra bin).
`scale`	Scale of the histogram: `"fourth_root"` (default), `"log"` (log10; zeros shown as a separate bar) or `"raw"`.
`seed`	Random seed.

Details

How it works. For each of `n_sim` replicates, the expected value of every operation is computed from the fitted model and a new observation is drawn from the fitted error distribution (Tweedie with the estimated power and dispersion; for delta models a Bernoulli draw for presence and a lognormal or gamma draw for the positive catch). With `param_uncertainty = TRUE` (default), the model coefficients of each replicate are first drawn from their approximate posterior (multivariate normal with the Bayesian covariance matrix `Vp`), which makes this an approximate posterior predictive check; otherwise it is a plug-in (parametric bootstrap) check at the estimated values. Random vessel effects are kept at their estimated values.

Each replicate is treated as a complete data set: the summaries (share of zeros, mean, upper quantiles of positive CPUE, histogram of CPUE) are computed per replicate, and the spread across replicates gives the envelope (2.5-97.5%) against which the observed value is judged. The `tail_prob` column is the share of replicates with a value at least as large as the observed one (a posterior predictive p-value; values near 0 or 1 indicate misfit).

Reading the plots.

- `zeros`: observed share of zero catches above the envelope means the model predicts too few zeros (e.g. a lognormal or Poisson-like error, or a missing targeting term in years when the fleet fished elsewhere); below means too many.
- `tails`: observed upper quantiles below the envelope mean the distribution's tail is too heavy (e.g. a negative binomial or a lognormal with large variance); above means too light.
- `distribution`: the observed histogram of CPUE, **including the zeros** (bars; the zero bar is the leftmost) against the predictive distribution (points and 95% ranges), as shares of all operations. Systematic departures show a wrong shape (e.g. the wrong Tweedie power) or a wrong balance between zeros and small catches.

Scale of the histogram. Positive CPUE is strongly right-skewed and often spans several orders of magnitude: on the raw scale (`scale = "raw"`) nearly all operations fall in the first bins and the tail is invisible. The fourth root (default) compresses the tail and keeps zero at its natural place next to the small catches; `scale = "log"` shows the shape of the positive catches best (a lognormal is symmetric there), with the zeros as a separate bar at the left. The axis is labelled in CPUE units.

Pooled over all operations, errors in opposite directions can cancel, so the by-year summaries are usually more informative. Models without targeting covariates often get the yearly share of zeros wrong in exactly the years when targeting changes.

Value

An object of class `cpue_ppc`: a list with `summary` (one row per model, group and statistic: observed value, median and 2.5-97.5% range of the replicates, `tail_prob`), `overall` (the same for all operations together), `histogram` (observed and simulated shares per bin) and `plots` (`zeros`, `tails`, `distribution`).

References

Gelman, A., Meng, X.-L. and Stern, H. (1996) Posterior predictive assessment of model fitness via realized discrepancies. Statistica Sinica 6: 733-807.

Kleiber, C. and Zeileis, A. (2016) Visualizing count data regressions using rootograms. The American Statistician 70: 296-303.

See Also

`plot_residuals()` for randomised quantile residuals, which check each observation against its own predicted distribution.

Examples

```
## Not run:
pp <- ppc_cpue(fit, models = c("Base", "DPC+S"))
pp$plots$zeros
pp$plots$tails
pp$plots$distribution
pp$overall

## End(Not run)
```

read_replicates *Read replicate results saved one file per seed*

Description

Combines the `.rds` files written by `run_replicates` (e.g. one file per seed from a parallel run) into one result for `plot_replicates`.

Usage

```
read_replicates(dir, pattern = "\\.rds$", label = NULL)
```

Arguments

`dir`	Folder with the `.rds` files.
`pattern`	Regular expression for the file names.
`label`	Optional label stored in a column `label` (e.g. `"Flat trends"`), to combine runs of different scenarios.

Value

A list with `diagnostics` and `indices`, as from `run_replicates`.

run_replicates *Run scenarios over several seeds (replicates)*

Description

Simulates each scenario for each seed, standardises the data and keeps the indices and accuracy diagnostics. Results are saved to `file` after every fit, and seeds/scenarios already in `file` are skipped, so a long (e.g. overnight) run can be interrupted and resumed.

Usage

```
run_replicates(scenarios, seeds, models, extra = NULL, file = NULL,
  simulator = simulate_tuna, ...)
```

Arguments

`scenarios`	Named list of settings lists (see `tuna_settings`).
`seeds`	Integer vector of seeds.
`models`	Models passed to `standardize_cpue`.
`extra`	Optional list of extra standardisation calls, each a list of arguments for `standardize_cpue` (e.g. a 1 x 1 control with a `label`); their results are added with `bind_comparisons`.

file	Optional .rds file to save and resume from.
simulator	Simulation function, simulate_tuna (default) or simulate_bathymetric; scenarios must hold matching settings.
...	Further arguments for standardize_cpue (e.g. spatial_res, n_pc, engine).

Value

A list with diagnostics and indices, each with columns scenario and seed added.

Examples

```
## Not run:
base <- tuna_settings()
scenarios <- list(Fixed = base,
                  Moving = modifyList(base, list(thermocline_annual_sd = 15,
                                                 thermocline_annual_field_sd = 10)))
reps <- run_replicates(scenarios, seeds = 1:10,
                       models = c("Base+S", "Cluster+S", "DPC+S", "HBF+S"),
                       engine = "bam", file = "replicates.rds")
plot_replicates(reps)

## End(Not run)
```

---

schaefer_update

The growth rate r has one value per species and is applied by column. (The original code multiplied B by r directly, which recycles r over cells rather than species.)

Description

The growth rate r has one value per species and is applied by column. (The original code multiplied B by r directly, which recycles r over cells rather than species.)

Usage

```
schaefer_update(biomass, carrying, r, catch)
```

select_models *Keep only some models of a standardisation result*

Description

Useful to reuse an expensive fit (e.g. a 1 x 1 spatial control) and attach it to new fits with `bind_comparisons` instead of refitting it.

Usage

```
select_models(comparison, models)
```

Arguments

`comparison`	Output of `standardize_cpue` or `bind_comparisons`.
`models`	Model names to keep.

Value

The same structure, restricted to `models`.

Examples

```
## Not run:
control <- select_models(old_fit, "Base+S (1x1)")
plot_indices(bind_comparisons(new_fit, control))

## End(Not run)
```

select_species *Select the main species*

Description

Keeps the species that, ranked by `species_curve()`, are needed to reach the cumulative share `threshold`, plus any species listed in `keep` (e.g. the species of interest if it is rare). The other species are dropped or pooled into one column `Other`. Operations that caught none of the remaining species are removed, because their catch composition is undefined.

Usage

```
select_species(
  x,
  threshold = 0.95,
  by = c("occurrence", "weight"),
  keep = NULL,
  n = NULL,
  pool_other = FALSE
)
```

Arguments

x	A cpue_data object.
threshold	Cumulative share to reach (default 0.95).
by	"occurrence" (default) or "weight".
keep	Species to keep in any case (column names or original names).
n	Alternatively, the number of species to keep (overrides threshold).
pool_other	Logical; pool the dropped species into Other.

Details

Pooling keeps the information that an operation caught "something else", which can help to separate tactics, but a large Other group can also dominate the composition; try both.

Value

The cpue_data object restricted to the selected species.

Examples

```
## Not run:
plot_species_curve(x, threshold = 0.95)
x <- select_species(x, threshold = 0.95, keep = "Pagellus bogaraveo")

## End(Not run)
```

---

select_tactics *Keep or drop fishing tactics and recompute the targeting covariates*

Description

An intermediate cleaning step: remove operations of fishing tactics (clusters) that are irrelevant for the species to be standardised, e.g. the pelagic tactic when analysing a demersal species, and then recompute the PCA and clusters on the remaining operations. Without the other tactics the PCA describes the variation that matters for the target (e.g. reef versus sediment, shallow versus deep) instead of spending its first axes on the contrast with the removed tactics.

Usage

```
select_tactics(
  x,
  keep = NULL,
  drop = NULL,
  species = NULL,
  min_share = 0.5,
  retarget = TRUE,
  n_pc = NULL,
  n_clusters = NULL,
  pca_species = x$species,
  ...
)
```

Arguments

x	A cpue_data object after add_targeting().
keep, drop	Cluster numbers to keep or drop.
species	Species (column or original names) whose combined share of a cluster's catch decides whether it is kept.
min_share	Minimum combined share of species (default 0.5).
retarget	Logical; rerun add_targeting() on the remaining operations (default). If FALSE, the old PC scores and clusters are removed.
n_pc, n_clusters	Passed to add_targeting() (default: the Optimal Coordinates test is run again on the subset).
pca_species	Species used for the new PCA (default: all species of x; species without catches in the subset are dropped automatically).
...	Further arguments for add_targeting(); by default the clustering method, on and transform of the previous call are reused.

Details

Clusters can be chosen by number (keep, drop) or by their species content: with species and min_share, clusters are kept if the given species make up at least min_share of their catch (by weight), e.g. all demersal species with min_share = 0.5.

Caution. The clusters are derived from the catch composition, so this selection is based on the data being standardised. Select on groups of *other* species (or on the tactic that clearly does not catch the target) rather than on the target alone, and compare the index with and without the selection.

Value

The subsetted cpue_data object with new targeting covariates. x$targeting$previous holds the composition of the previous clusters and which were kept.

See Also

tactic_table(), select_vessels(), stephens_maccall()

Examples

```
## Not run:
tactic_table(x)                                  # which cluster is pelagic?
xd <- select_tactics(x, drop = 3)                # by number
demersal <- c("Dentex dentex", "Pagellus bogaraveo", "Conger conger")
xd <- select_tactics(x, species = demersal, min_share = 0.5)
tactic_table(xd)

## End(Not run)
```

---

select_vessels *Select core (indicator) vessels*

Description

Keeps vessels that meet a set of rules, a common way to define an indicator fleet with consistent reporting and a consistent interest in the target species (e.g. Hoyle et al. 2015):

- activity: at least `min_years` years with at least `min_records` operations each;
- target species (optional): caught in at least `min_years_target` years, in at least a share `min_share_target` of the vessel's operations, and/or a total catch of at least `min_catch_target`.

Usage

```
select_vessels(
  x,
  min_years = 3,
  min_records = 10,
  target = NULL,
  min_years_target = NULL,
  min_share_target = NULL,
  min_catch_target = NULL
)
```

Arguments

`x`	A `cpue_data` object with a vessel column.
`min_years`	Minimum number of qualifying years.
`min_records`	Minimum number of operations for a year to qualify.
`target`	Target species for the optional rules (column or original names; several species are combined).
`min_years_target`	Minimum number of years with catches of the target.

`min_share_target`
: Minimum share of the vessel's operations catching the target.

`min_catch_target`
: Minimum total catch of the target.

Details

Rules on the target species select vessels that fish its habitat regularly. They are applied to whole vessels, so the selection does not change with the vessel's targeting from trip to trip; combine them with `select_tactics()` or `stephens_maccall()` for trip-level selection. Check the selection with `vessel_summary()` and `plot_vessel_profile()`.

Value

The subsetted `cpue_data` object; the vessel table is attached as attribute `vessels` (with a column `selected`).

References

Hoyle, S.D., Langley, A.D. and Campbell, R.A. (2014) Recommended approaches for standardizing CPUE data from pelagic fisheries. WCPFC-SC10-2014/SA-IP-10.

Examples

```
## Not run:
vs <- vessel_summary(x, target = "Conger conger")
x_core <- select_vessels(x, min_years = 5, min_records = 10,
                         target = "Conger conger", min_years_target = 5,
                         min_share_target = 0.3)
attr(x_core, "vessels")

## End(Not run)
```

---

simulate_bathymetric *Simulate the four-species bathymetric Run 22 scenario*

Description

Four species are distributed over a simulated bathymetry according to depth preferences and short-distance movement. Fishing effort is allocated in proportion to expected catch value, which changes with species prices, so targeting shifts over time.

This is a re-implementation of the spatial multispecies simulator of Thorson et al. (2017, Appendix A; function `Generate_scenario_data` in the `spatial_DFA` package, https://github.com/James-Thorson/spatial_DFA), which generalises the simulator of Ono et al. (2015), with the "Run 22" settings used for the WKBMSYSPICT CPUE guidelines. A full description is in `system.file("appendix", "cpueTargeting_simulator_appendix.html",package = "cpueTargeting")`.

Usage

```
simulate_bathymetric(settings = bathymetric_settings(), seed = NULL, plots = TRUE)
```

Arguments

settings	A named settings list, usually from bathymetric_settings.
seed	Optional integer seed for reproducibility.
plots	Logical; build the diagnostic ggplots in Plots.

Details

Compared with the original script, the dynamics are identical except for three corrections: species-specific r is now applied by species (the original recycled r over cells), the initial distribution is the exact stable distribution of the movement process, and reported block centres are aligned with the grid.

Value

A list with

- Data: one row per set and species (year, SetID, X, Y, Xobs, Yobs, Species, Sp, CPUE, depth, vessel);
- Biomass: annual nominal and true indices, scaled to mean 1;
- Biomass.XY: initial density by cell and species;
- Env: the bathymetry;
- Plots: diagnostic ggplots (if plots = TRUE);
- settings: the settings used.

References

Thorson, J.T., Fonner, R., Haltuch, M.A., Ono, K. and Winker, H. (2017) Accounting for spatiotemporal variation and fisher targeting when estimating abundance from multispecies fishery data. Canadian Journal of Fisheries and Aquatic Sciences 74, 1794-1807.

Ono, K., Punt, A.E. and Hilborn, R. (2015) Think outside the grids: an objective approach to define spatial strata for catch and effort analysis. Fisheries Research 170, 89-101.

Examples

```
## Not run:
om <- simulate_bathymetric(seed = 42)
head(om$Data)
om$Plots$CPUE

## End(Not run)
```

`simulate_cluster` *Simulate the 2014 multi-habitat cluster scenario*

Description

A callable version of the multi-habitat, multi-species cluster targeting example from Winker et al. (2014), based on the original `SimulationCPUE_3d.R` script.

Usage

```
simulate_cluster(years = 20, trips_per_year = 500,
  scenario = c("targeting", "constant"), habitat_switch = 0.25,
  tweedie_power = 1.3, tweedie_dispersion = 10, seed = NULL)
```

Arguments

`years`	Number of annual time steps.
`trips_per_year`	Number of trips sampled in each year.
`scenario`	Either `"targeting"` for changing habitat allocation or `"constant"` for approximately constant allocation.
`habitat_switch`	Probability that the second habitat uses a separate catch-rate draw.
`tweedie_power`	Tweedie power parameter, between 1 and 2.
`tweedie_dispersion`	Positive Tweedie dispersion parameter.
`seed`	Optional integer seed for reproducibility.

Value

A list with `data`, `biomass`, `effort`, and `settings`.

Examples

```
## Not run:
sim <- simulate_cluster(years = 20, trips_per_year = 100, seed = 1)
head(sim$data)

## End(Not run)
```

`simulate_logbook` *Simulate a realistic multispecies logbook*

Description

A spatial operating model of a mixed demersal-pelagic fishery with discrete tactic choice, combining the spatial multispecies simulator (`simulate_bathymetric()`) with the habitat-based tactics of Winker et al. (2014) (`simulate_cluster()`), and a reporting layer that turns sets into a typical trip-level logbook.

Usage

```
simulate_logbook(settings = logbook_settings(), seed = NULL, plots = TRUE)
```

Arguments

`settings`	Settings list, usually a modified `logbook_settings()`.
`seed`	Optional seed.
`plots`	Logical; build diagnostic plots.

Details

Environment. A shelf-to-slope bathymetry (depth increases offshore, plus a Gaussian random field), a patchy substrate (reef patches from a thresholded random field) and an SST gradient. Demersal densities are the product of a depth preference and a substrate affinity, so the four habitats shallow/deep x reef/sediment arise as continuous density gradients. Pelagic densities follow SST and depth, and their availability in the area changes seasonally (migrants).

Dynamics. Schaefer production in each cell once a year, then movement (exponential distance kernel weighted by habitat). Catches are removed every month.

Fleet. Each month, trips choose a tactic (`Reef`, `Sediment`, `Pelagic`) and a cell with probability proportional to `expected value ^ preference`, where expected value is the sum over species of price x catchability of the tactic x available biomass. A share `loyalty` of a specialist's trips is restricted to its home tactics; its other trips, and all trips of mixed vessels, choose among all tactics. Prices change between periods, which shifts tactics; a closure bans a tactic from part of the area; vessels enter (more efficient) and leave.

Logbook. Some trips make a second set (possibly with another tactic) and report the sum, with one position; positions are reported at a coarse resolution and are sometimes missing; catches are rounded; small catches of low-value species are often not recorded; one species is sometimes recorded under another name. The tactic is not reported.

Value

A list with

- `logbook`: one row per trip and recorded species (`trip`, `year`, `month`, `vessel`, `hours`, `lon`, `lat`, `species`, `kg`), ready for `cpue_data()`;

- `trips`: the truth per trip (vessel type, whether the trip stayed on home tactics (`loyal`), tactic of each set, number of sets, cell, depth, substrate, habitat);
- `truth`: true total biomass by year and species (also scaled to mean 1);
- `nominal`: nominal CPUE (kg/hour) by year and species from the logbook;
- `env`: depth, substrate and SST by cell; `density`: initial density by cell and species; `prices`: prices by year and species;
- `Plots` (if `plots = TRUE`) and `settings`.

See Also

`logbook_truth()`, `plot_logbook_om()`

Examples

```
## Not run:
sim <- simulate_logbook(seed = 1)
head(sim$logbook)
sim$Plots$density
sim$Plots$tactics
x <- cpue_data(sim$logbook, year = "year", species = "species", catch = "kg",
               effort = "hours", id = "trip", vessel = "vessel",
               lon = "lon", lat = "lat", covariates = "month",
               synonyms = c("Blackspot seabream" = "Pagellus bogaraveo"))

## End(Not run)
```

---

simulate_tuna *Simulate a longline fishery for four tuna species with vertical targeting*

Description

Extends the spatial multispecies simulator of Thorson et al. (2017) and Ono et al. (2015), as implemented in `simulate_bathymetric`, with a pelagic horizontal habitat (SST), species vertical distributions, longline hook depth by HBF and vertical targeting, following the habitat-based approach of Bigelow et al. (2002). A full description of both models is in `system.file("appendix", "cpueTargeting_simulator_appendix.html",package = "cpueTargeting")`. The catchability of a set for a species is the vessel's catchability times the overlap between the set's hook-depth distribution and the species' vertical distribution in that cell (Bigelow's effective effort). See `tuna_settings` for the processes and parameters.

Usage

```
simulate_tuna(settings = tuna_settings(), seed = NULL, plots = TRUE)
```

Arguments

`settings`	A named list, usually a modified `tuna_settings`.
`seed`	Optional integer seed for reproducibility.
`plots`	Logical; build the diagnostic ggplots in `Plots`.

Value

A list with

- `Data`: one row per set and species with `year`, `SetID`, `X`, `Y`, `Xobs`, `Yobs`, `vessel`, `HBF`, `sst`, `thermocline` (anomaly in that year, m), `Species`, `Sp`, `CPUE`, and the latent truth `hook_depth` (median hook depth of the set), `vertical_avail` (realised vertical overlap) and `hbs_avail` (overlap predicted by a habitat model from HBF, the expected hook profile and the average thermocline, as in a habitat-based standardisation);
- `Biomass`: annual nominal and true indices, scaled to mean 1;
- `Biomass.XY`: initial density by cell and species;
- `Env`: SST and the fixed thermocline anomaly by cell;
- `Thermocline`: thermocline anomaly by cell and year;
- `Plots`: diagnostic ggplots (if `plots = TRUE`);
- `settings`: the settings used.

References

Bigelow, K.A., Hampton, J. and Miyabe, N. (2002) Application of a habitat-based model to estimate effective longline fishing effort and relative abundance of Pacific bigeye tuna (*Thunnus obesus*). Fisheries Oceanography 11, 143-155.

Thorson, J.T., Fonner, R., Haltuch, M.A., Ono, K. and Winker, H. (2017) Accounting for spatiotemporal variation and fisher targeting when estimating abundance from multispecies fishery data. Canadian Journal of Fisheries and Aquatic Sciences 74, 1794-1807.

Ono, K., Punt, A.E. and Hilborn, R. (2015) Think outside the grids: an objective approach to define spatial strata for catch and effort analysis. Fisheries Research 170, 89-101.

Examples

```
## Not run:
om <- simulate_tuna(seed = 42)
head(om$Data)
om$Plots$CPUE

## End(Not run)
```

`species_curve` *Cumulative species curve*

Description

Ranks species by how often they are caught (`by = "occurrence"`, the share of all positive species records, as in the CPUE guidelines) or by total catch (`by = "weight"`), with the cumulative share. Used to choose the set of species that carries the targeting signal: rare species add noise to the catch composition, so the PCA and clustering use only the species that together make up most of the records (e.g. 95%).

Usage

```
species_curve(x, by = c("occurrence", "weight"))
```

Arguments

x	A cpue_data object.
by	"occurrence" (default) or "weight".

Value

A data frame with species (column name), label, rank, value (number of positive records or total catch), share and cumulative.

See Also

plot_species_curve(), select_species()

---

standardize_cpue *Standardise simulated CPUE with alternative targeting covariates*

Description

Fits Tweedie GAMs (mgcv::gam with tw()) to each species and compares the annual indices with the true biomass trend.

Usage

```
standardize_cpue(simulation, models = NULL, spatial_res = 5,
  spatial_k = NULL, add_models = NULL, vessel_re = FALSE, n_pc = 3,
  n_clusters = n_pc + 1, seed = 1, engine = c("gam", "bam"), label = NULL)

standardize_tuna_indices(...)
```

Arguments

...	Arguments passed to standardize_cpue().
simulation	Output of simulate_tuna or simulate_bathymetric.
models	Models to fit, e.g. c("Base", "Base+S", "DPC+S", "HBF+S"). Default: Base, plus Base, Cluster, DPC, HBF, Cluster_HBF and DPC_HBF with +S (the HBF models only for the tuna model).
spatial_res	Size of the reporting blocks in cells (5 = 5 x 5 blocks as in the CPUE guidelines; 1 = full resolution).
spatial_k	Basis dimension of the spatial smoother. By default it grows with the number of blocks (8 for 100 blocks, at most 60).

add_models	Optional named character vector of extra right-hand sides, e.g. c(HBF_period = "year + s(Xs, Ys, k = 8) + HBF_period"). They can use any column of the returned data, including Xs, Ys, period (10-year block as a factor) and HBF_period (HBF x period).
vessel_re	Logical; add a random vessel effect s(vessel, bs = "re").
n_pc	Number of principal components used by the DPC models (default 3; the 2023 CPUE guidelines used 2; at most the number of species). More PCs capture more of the targeting signal, but also more risk of absorbing real changes in relative abundance.
n_clusters	Number of fishing tactics for the Cluster models. By default n_pc + 1, the rule of Winker et al. (2014).
seed	Seed for the k-means clustering.
engine	"gam" (default) or "bam" with discrete = TRUE, which is much faster for large data sets.
label	Optional text appended to every model name, e.g. " (1x1)", to tell fits at different resolutions apart after bind_comparisons.

Details

Each model is year plus a targeting term:

- Base: no targeting term.
- DPC: s(PC1) + ... + s(PCn) from the catch composition (direct principal components), with n = n_pc.
- Cluster: fishing tactic from k-means clustering of the catch composition (as a factor).
- HBF: HBF as a factor (tuna model only).
- Cluster_HBF: Cluster + HBF (tuna model only).
- DPC_HBF: DPC + HBF (tuna model only).
- HBS: offset(log(hbs_avail)), a specialised habitat-based standardisation: effort is weighted by the overlap between the expected hook profile of the HBF and the species' depth profile under the average thermocline (tuna model only; not fitted by default).

Append +S to any model name to add a spatial smoother s(Xs, Ys), e.g. "DPC+S"; "Base+S" is the spatial-only model. Xs, Ys are the centres of spatial_res x spatial_res blocks of cells. The latent hook_depth and realised vertical_avail are never used as predictors.

Value

A list with indices (species, year, model, index with 95% CI and the true index), diagnostics (root mean squared error of the log index by species and model), formulas (the right-hand side of each model), data (the set-level data used, with PCs, tactics and block centres) and pca (loadings and variances of the catch-composition PCA; see plot_pcs).

Examples

```
## Not run:
om <- simulate_tuna(seed = 42)
comparison <- standardize_cpue(om, models = c("Base", "Base+S",
                                              "DPC+S", "Cluster+S", "HBF+S"))
comparison$diagnostics
plot_indices(comparison)

## End(Not run)
```

`stephens_maccall` *Select operations likely to have fished the habitat of a target species*

Description

Stephens and MacCall (2004): a binomial GLM predicts the presence of the target species from the presence or absence of the other species in each operation. Operations with a predicted probability above a critical value are kept. The critical value is the one at which the number of predicted presences equals the number of observed presences, i.e. it minimises $|\sum obs - \sum(\hat{p} > p_{crit})|$, which balances false positives and false negatives. This is the implementation used in the simulation study of Winker et al. (2014).

Usage

```
stephens_maccall(
  x,
  target,
  species = NULL,
  thresholds = seq(0.01, 0.99, by = 0.01),
  min_keep = 0
)
```

Arguments

`x`	A `cpue_data` object.
`target`	Target species (column or original name).
`species`	Predictor species (default: all others in `x`).
`thresholds`	Candidate critical probabilities.
`min_keep`	Minimum share of operations to keep; if the critical value would keep fewer, the largest threshold that keeps at least this share is used and a message is given.

Details

The species set matters: species that co-occur with the target (positive coefficients) and species of other habitats (negative coefficients) both carry information. Use `select_species()` first, or give `species`.

Value

An object of class `stephens_maccall`: a list with `fit` (the GLM), `coefficients` (species, estimate, SE), `prob` (predicted probability for each operation), `curve` (threshold, difference between observed and predicted presences, and number of operations kept), `critical` (the critical probability), `selected` (logical, one per operation), `target`, and `year` (for the plots).

References

Stephens, A. and MacCall, A. (2004) A multispecies approach to subsetting logbook data for purposes of estimating CPUE. Fisheries Research 70: 299-310.

See Also

`plot_stephens_maccall()`, `subset.cpue_data()`

Examples

```
## Not run:
sm <- stephens_maccall(x, target = "Pagellus bogaraveo")
plot_stephens_maccall(sm)
x_sm <- subset(x, sm$selected, note = "Stephens & MacCall")

## End(Not run)
```

---

subset.cpue_data *Keep a subset of operations*

Description

Works like `base::subset()` on the operation table and records the step in the processing log. `subset` can be a logical expression evaluated within `x$data` (e.g. `year >= 2005 & month %in% 4:9`) or a logical vector, e.g. the `selected` element of `stephens_maccall()`.

Usage

```
## S3 method for class 'cpue_data'
subset(x, subset, note = NULL, ...)
```

Arguments

`x`	A `cpue_data` object.
`subset`	Logical expression or vector.
`note`	Optional text for the processing log.
`...`	Not used.

Value

The subsetted `cpue_data` object.

Examples

```
## Not run:
x2 <- subset(x, year >= 2010, note = "recent years")

## End(Not run)
```

`tactic_table` *Species composition of each fishing tactic*

Description

Species composition of each fishing tactic

Usage

```
tactic_table(x, type = c("weight", "occurrence"))
```

Arguments

`x`	A `cpue_data` object after `add_targeting()`.
`type`	`"weight"` (percentage of the cluster's total catch, as in the CPUE guidelines) or `"occurrence"` (percentage of operations catching each species).

Value

A data frame with one row per cluster: `FT`, number and share of operations, the dominant species, and one column per species.

`tuna_settings` *Default settings for the longline tuna scenario*

Description

Four tuna species in a conceptual Southern Hemisphere ocean (y points north, towards the equator, so the north is warm).

Usage

```
tuna_settings(preset = c("default", "mild", "stress_test"))
```

Arguments

`preset`	`"default"`, `"mild"` or `"stress_test"`.

Details

- Horizontal habitat is a normal preference for sea surface temperature (`sst_opt`, `sst_tol`). Yellowfin (YFT) and bigeye (BET) prefer warm water; southern bluefin (BFT) and albacore (ALB) prefer cooler water. Wide tolerances keep the species mixed rather than separated.
- Vertical habitat is a normal depth profile (`depth_mu`, `depth_sd`, in m). Within each pair one species is shallow (YFT, ALB) and one deep (BET, BFT). A thermocline-depth anomaly shifts all profiles up or down together in each cell, so the catch of a given gear configuration varies in space and, optionally, between years. The anomaly in year t and cell i is `fixed[i] + shift[t] + annual[i, t]`: `fixed` is a spatial field that stays the same every year (`thermocline_sd`; 0 = same vertical profile everywhere); `shift` is a basin-wide AR(1) shift (`thermocline_annual_sd`, `thermocline_annual_rho`), e.g. ENSO-like years; `annual` is a new spatial field each year (`thermocline_annual_field_sd`).
- Hook depth: each HBF (hooks between floats) level has a typical (median) hook depth `HBF_depth`, roughly as in Fig. 2 of Bigelow et al. (2002). Each set's hook profile is shifted by a lognormal set-level error (`set_cv`; e.g. line angle, currents), so HBF is an imperfect proxy for depth. Hooks within a set are lognormally spread around the set's median (`hook_cv`).
- Targeting: species prices change in three 10-year blocks: ALB (shallow, cool water), then BET (deep, warm water), then BET and BFT (deep, warm and cool water), loosely following the historical shift from albacore to bigeye targeting and the later interest in recovering southern bluefin. Skippers choose where to fish and which HBF to use from the expected catch value.
- Effort increases towards 800 sets per year and is cut by 25% in years 21-30, so pressure on some species is relieved.
- True trends differ between species: ALB is fished down from 90% of carrying capacity, BET declines steadily, BFT recovers from 20% and declines again once it is targeted, and YFT dips and recovers.

All values are conceptual and meant to be tuned. With the same seed, scenarios that differ only in the thermocline settings share the same SST field, the same fixed thermocline pattern (scaled by `thermocline_sd`) and the same vessels; the fishery itself then evolves differently.

Value

A named list accepted by `simulate_tuna`.

Presets

- `"default"`: strong targeting (prices differ by a factor of 4 between the target species and the others, `preference = 1.5`, `HBF_choice_temp = 0.15`) and stocks fished to 30-60% of carrying capacity with contrasting trends. Nominal CPUE is biased by about 15-20%.
- `"mild"`: as the default but with moderate targeting (prices differ by a factor of 2, `preference = 1.25`, `HBF_choice_temp = 0.2`). Nominal CPUE is only slightly biased, so the standardisation methods are hard to tell apart in a single run.
- `"stress_test"`: the first settings of the vignette: strong targeting with ALB, BET and BFT blocks, but light fishing from unfished stocks (`q = 0.2`, `B_start = 1`, effort increasing towards 800 sets per year), so the true trends are nearly flat and nominal CPUE changes mainly through targeting.

Examples

```
s <- tuna_settings()
tuna_settings()$price[c(1, 11, 21), ]
mild <- tuna_settings("mild")
```

`vessel_summary` *Summary of each vessel's activity and catches of target species*

Description

Summary of each vessel's activity and catches of target species

Usage

```
vessel_summary(x, target = NULL, min_records = 1)
```

Arguments

`x`	A `cpue_data` object with a vessel column.
`target`	Optional target species (column or original names; several species are combined).
`min_records`	Number of operations for a year to count in `years_qualified`.

Value

A data frame with one row per vessel: `records`, `years` (years with any record), `years_qualified` (years with at least `min_records` records), `first`, `last` and, with `target`, `records_target` (operations catching the target), `share_target` (their share of the vessel's operations), `years_target` (years in which the target was caught) and `catch_target` (total catch of the target).

See Also

`select_vessels()`, `plot_vessel_profile()`

Index

add_targeting, 3
add_targeting(), *9, 10*, *13*, *18*, *25*, *31*, *41*, *53*
as_set_data, 4

base::subset(), *52*
bathymetric_settings, 5, *44*
bind_comparisons, 5, *20*, *26*, *37*, *39*, *50*

cluster_tactics, 6
cluster_tactics(), *3, 4*
composition_pca, 7
composition_pca(), *3, 4*, *6*
compute_oc, 8
compute_oc(), *7*
cpue_data, 9
cpue_data(), *13, 14*, *46*
cpue_diagnostics, 11
cpue_diagnostics(), *15*
cpue_index, 11
cpue_index(), *15*
cpue_matrix, 12

export_index, 12

fit_cpue, 13
fit_cpue(), *11, 12*, *15*, *22–24*, *28*, *30*, *33*, *35*

influence_cpue, 15

logbook_settings, 16
logbook_settings(), *46*
logbook_truth, 17
logbook_truth(), *47*

nominal_cpue, 17

plot_availability, 18, *26*, *32*
plot_biplot, 18
plot_biplot(), *4*
plot_cluster_2014, 19
plot_clusters, 19
plot_composition, 20
plot_cpue_map, 21
plot_diagnostics, 21
plot_effects, 22
plot_fleet, 22
plot_index, 23
plot_index(), *15*
plot_indices, 23
plot_influence, 24
plot_influence(), *15*
plot_logbook_om, 24
plot_logbook_om(), *47*
plot_nominal, 25
plot_oc, 25
plot_oc(), *4*
plot_om, 26
plot_pcs, 26, *50*
plot_replicates, 27, *37*
plot_residuals, 27
plot_residuals(), *36*
plot_sections, 28
plot_species_curve, 29
plot_species_curve(), *49*
plot_step, 30
plot_stephens_maccall, 30
plot_stephens_maccall(), *52*
plot_targeting, 31
plot_targeting(), *4*
plot_thermocline, 31
plot_thermocline_effect, 32
plot_tuna_vertical_distributions, 32
plot_vessel_effects, 33
plot_vessel_profile, 34
plot_vessel_profile(), *43*, *55*
ppc_cpue, 34

read_replicates, 37
run_replicates, *27*, 37, *37*

schaefer_update, 38

select_models, 39
select_species, 39
select_species(), *4*, *7*, *10*, *13*, *49*, *51*
select_tactics, 40
select_tactics(), *43*
select_vessels, 42
select_vessels(), *34*, *42*, *55*
simulate_bathymetric, *5*, *26*, *38*, 43, *47*, *49*
simulate_bathymetric(), *46*
simulate_cluster, 45
simulate_cluster(), *46*
simulate_logbook, 46
simulate_logbook(), *16, 17*, *24*
simulate_tuna, *18*, *26*, *28*, *31*, *38*, 47, *49*, *54*
species_curve, 48
species_curve(), *9, 10*, *39*
standardize_cpue, *5, 6*, *20, 21*, *23*, *26*, *37–39*, 49
standardize_tuna_indices (standardize_cpue), 49
stats::prcomp(), *7*
stephens_maccall, 51
stephens_maccall(), *9*, *30*, *42, 43*, *52*
subset.cpue_data, 52
subset.cpue_data(), *52*

tactic_table, 53
tactic_table(), *4*, *42*
tuna_settings, *32, 33*, *37*, *47*, 53

vessel_summary, 55
vessel_summary(), *34*, *43*