Package 'atlantistools'

August 16, 2017

Type Package

Title Process and Visualise Output from Atlantis Models

Version 0.4.3

Description Atlantis is an end-to-end marine ecosystem modelling framework. It was originally developed in Australia by E.A. Fulton, A.D.M. Smith and D.C. Smith (2007) and has since been adopted in many marine ecosystems around the world (<http://atlantis.cmar.csiro.au>). The output of an Atlantis simulation is stored in various file formats like .netcdf and .txt and different output structures are used for the output variables like e.g. productivity or biomass. This package is used to convert the different output types to a unified format according to the ``tidy-data'' approach by H. Wickham (2014) <DOI:10.18637/jss.v059.i10>. Additionally, ecological metrics like for example spatial overlap of predator and prey or consumption can be calculated and visualised with this package. Due to the unified data structure it is very easy to share model output with each other and perform model comparisons.

URL https://github.com/alketh/atlantistools

BugReports https://github.com/alketh/atlantistools/issues

Depends R (>= 3.2.3)

License GPL-3

LazyData true

RoxygenNote 6.0.1

Suggests testthat, knitr, rmarkdown, vdiffr, rfishbase

Imports dplyr, magrittr, RNetCDF, stringr, lazyeval, tidyr, ggplot2, proj4, gridExtra, scales, circlize, RColorBrewer, purrr, rvest, xml2, tibble

VignetteBuilder knitr

NeedsCompilation no

Author Alexander Keth [aut, cre]

Maintainer Alexander Keth <alexander.keth@uni-hamburg.de>

Repository CRAN

Date/Publication 2017-08-16 08:39:06 UTC

R topics documented:

`agg_data` *Aggregate data using dplyr functionality.*

Description

This function is a 'wrapper' for the group_by and summarize procedure used in dplyr to aggregate dataframes.

Usage

```
agg_data(data, col = "atoutput", groups, out = "atoutput", fun)

agg_perc(data, col = "atoutput", groups, out = "atoutput")

group_data(data, groups)
```

Arguments

data	Dataframe the aggregation is applied to.
col	Column of the dataframe the summarise function is applied. Default is atoutput.
groups	Vector of character strings giving the grouping variables.
out	Character string specifying the name of the output column. Default is atoutput.
fun	Aggregation function to apply.

Value

grouped datarame with the aggregated data.

Examples

```
agg_ref_nums <- agg_data(data = ref_nums, groups = c("species", "agecl"), fun = mean)
```

---

calculate_biomass_spatial

Calculate spatially explicit biomass (in [t]) for each group and age-class per timestep.

Description

Calculate spatially explicit biomass time series for each group and ageclass within our model. Data is read in from 'output[...].nc'. Biomass for age-based groups is calculated as (StructN [mgN] + ResN [mgN]) * Numbers [individuals]. Biomass for non age-based groups is calculated as N [mgN] * volume [m^3] (sediment-dz [m] / volume [m^3] for epibenthic groups). mgN is converted to t based on the stettings in the biol.prm file. Simulation time steps are converted to time in years based on output timesteps given in run.prm.

Usage

```
calculate_biomass_spatial(nums, sn, rn, n, vol_dz, bio_conv, bps)
```

Arguments

nums	Dataframe with information about numbers for age-based groups. Should be generated with load_nc.
sn	Dataframe with information about structural nitrogen for age-based groups. Should be generated with load_nc.
rn	Dataframe with information about reserve nitrogen for age-based groups. Should be generated with load_nc.
n	Dataframe with information about nitrogen for non-ge-based groups. Should be generated with load_nc.
vol_dz	Dataframe with information about volume and layer height per polygon. Should be generated with load_nc_physics.
bio_conv	Numeric value to transform weight in mg N to tonnes. Should be generated with get_conv_mgnbiot.
bps	Vector of character strings giving the complete list of epibenthic functional groups (Only present in the sediment layer). The names have to match the column 'Name' in the 'functionalGroups.csv' file. Should be generated with load_bps.

Value

Dataframe with columns 'species', 'agecl', 'polygon', 'layer', 'time'. Biomass in [t] is stored in column 'atoutput'.

Examples

```
# 1. Using built in datasets.
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")
init <- file.path(d, "INIT_VMPA_Jan2015.nc")
bps <- load_bps(fgs = fgs, init = init)

bio_conv <- get_conv_mgnbiot(prm_biol = prm_biol)

df <- calculate_biomass_spatial(nums = ref_nums, sn = ref_structn, rn = ref_resn, n = ref_n,
                                vol_dz = ref_vol_dz, bio_conv = bio_conv, bps = bps)

# 2. Read in dataframes from existing Atlantis simulation.
bboxes <- get_boundary(boxinfo = load_box(file.path(d, "VMPA_setas.bgm")))
nc_gen <- file.path(d, "outputSETAS.nc")
prm_run <- file.path(d, "VMPA_setas_run_fishing_F_Trunk.prm")

groups_age <- c("Planktiv_S_Fish", "Pisciv_S_Fish")
groups_rest <- c("Cephalopod", "Megazoobenthos", "Diatom", "Lab_Det", "Ref_Det")

nums <- load_nc(nc = nc_gen, bps = bps, fgs = fgs,
                select_groups = groups_age, select_variable = "Nums",
                prm_run = prm_run, bboxes = bboxes)
sn <- load_nc(nc = nc_gen, bps = bps, fgs = fgs,
```

```
              select_groups = groups_age, select_variable = "StructN",
              prm_run = prm_run, bboxes = bboxes)
rn <- load_nc(nc = nc_gen, bps = bps, fgs = fgs,
              select_groups = groups_age, select_variable = "ResN",
              prm_run = prm_run, bboxes = bboxes)
n <- load_nc(nc = nc_gen, bps = bps, fgs = fgs,
             select_groups = groups_rest, select_variable = "N",
             prm_run = prm_run, bboxes = bboxes)
vol <- load_nc_physics(nc = nc_gen, select_physics = c("volume", "dz"),
                       prm_run = prm_run, bboxes = bboxes, aggregate_layers = FALSE)

df <- calculate_biomass_spatial(nums = nums, sn = sn, rn = rn, n = n, vol_dz = vol,
                                bio_conv = bio_conv, bps = bps)

# 3. Read in dataframes from existing Atlantis simulation with Map().
vars <- list("Nums", "StructN", "ResN", "N")
grps <- list(groups_age, groups_age, groups_age, groups_rest)
dfs <- Map(load_nc, select_variable = vars, select_groups = grps,
           MoreArgs = list(nc = nc_gen, bps = bps, fgs = fgs,
                           prm_run = prm_run, bboxes = bboxes))

df <- calculate_biomass_spatial(nums = dfs[[1]], sn = dfs[[2]], rn = dfs[[3]], n = dfs[[4]],
                                vol_dz = vol, bio_conv = bio_conv, bps = bps)
```

`calculate_consumed_biomass` *Calculate the consumed biomass in [t] of prey j by predator i.*

Description

Consumption data is extracted from output[...]PROD.nc. Age based groups are stored as "Eat_" non age based groups as "Grazing_". Units are mg N m^-3 d^-1. Factors are species, time, box and agecl (if present). We will refer to species as pred from here on to indicate the predator perspective. Diet contribution data is extracted from DietCheck.txt. Currently this only works for models based on the trunk code. Units are The consumed biomass is calculated as follows: - Calculate consumed biomass as Eat (or Grazing) * boxvolume per time, pred, agecl, box. - Convert to biomass in [t]. - Combine with diet contributions and calculate consumed biomass of prey species.

Usage

```
calculate_consumed_biomass(eat, grazing, dm, vol, bio_conv)
```

Arguments

`eat`	Dataframe with information about consumption for age-based groups. Should be generated with `load_nc`.
`grazing`	Dataframe with information about consumption for non-age-based groups. Should be generated with `load_nc`.

dm	Dataframe with information about diet contributions per predator. Should be generated with load_dietcheck using convert_names = TRUE.
vol	Dataframe with information about volume per polygon and layer. Should be generated with load_nc_physics.
bio_conv	Numeric value to transform weight in mg N to tonnes. Should be generated with get_conv_mgnbiot.

Value

Dataframe with columns 'pred', 'agecl', 'polygon', 'time', 'prey'. Consumed biomass in [t] is stored in column 'atoutput'.

Examples

```
# 1. Using built-in datasets.
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")

bio_conv <- get_conv_mgnbiot(prm_biol = prm_biol)

df <- calculate_consumed_biomass(eat = ref_eat, grazing = ref_grazing, dm = ref_dm,
                                 vol = ref_vol, bio_conv = bio_conv)

# 2. Read in dataframes from existing Atlantis simulation.
bboxes <- get_boundary(boxinfo = load_box(file.path(d, "VMPA_setas.bgm")))
nc_gen <- file.path(d, "outputSETAS.nc")
nc_prod <- file.path(d, "outputSETASPROD.nc")
prm_run <- file.path(d, "VMPA_setas_run_fishing_F_Trunk.prm")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")
init <- file.path(d, "INIT_VMPA_Jan2015.nc")

bps <- load_bps(fgs = fgs, init = init)

groups_age <- c("Planktiv_S_Fish", "Pisciv_S_Fish")
groups_rest <- c("Cephalopod", "Megazoobenthos", "Diatom", "Lab_Det", "Ref_Det")

df_eat <- load_nc(nc = nc_prod, bps = bps, fgs = fgs,
              select_groups = groups_age, select_variable = "Eat",
              prm_run = prm_run, bboxes = bboxes)
df_grz <- load_nc(nc = nc_prod, bps = bps, fgs = fgs,
              select_groups = groups_rest, select_variable = "Grazing",
              prm_run = prm_run, bboxes = bboxes)
df_dm <- load_dietcheck(dietcheck = file.path(d, "outputSETASDietCheck.txt"),
                    fgs = fgs, prm_run = prm_run, version_flag = 2, convert_names = TRUE)
vol <- load_nc_physics(nc = nc_gen, select_physics = "volume",
                       prm_run = prm_run, bboxes = bboxes, aggregate_layers = FALSE)

df <- calculate_consumed_biomass(eat = df_eat, grazing = df_grz, dm = df_dm,
                                 vol = vol, bio_conv = bio_conv)
```

`calculate_spatial_overlap` *Calculate 3d overlap of predator groups with their prey over time using Schoener Index.*

Description

Calculate 3d overlap of predator groups with their prey over time using Schoener Index.

Usage

```
calculate_spatial_overlap(biomass_spatial, dietmatrix, agemat)
```

Arguments

`biomass_spatial`	Biomass timeseries of each group and ageclass per polygon and layer. This dataframe should be generated with `calculate_biomass_spatial`.
`dietmatrix`	Availability matrix given in the biological parameter file. This dataframe should be generated with `load_dietmatrix`. Please make sure to use `convert_names = TRUE` in `load_dietmatrix`.
`agemat`	First mature age class for age structured groups. This dataframe should be generated with `prm_to_df` using "age_mat" as parameter.

Value

List of Schoener's similarity indices ranging from 1 (perfect overlap) to 0 (zero overlap). Dataframe in first listentry gives pred, pred agecl, prey, preyageclass specific index. Dataframe in second listentry gives index per pred, pred agecl. Si is calculated as weighted mean with the availabilities as weights.

Examples

```
# Using built in datasets.
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")

dietmatrix <- load_dietmatrix(prm_biol, fgs, convert_names = TRUE)
agemat <- prm_to_df(prm_biol = prm_biol, fgs = fgs,
                    group = get_age_acronyms(fgs = fgs),
                    parameter = "age_mat")

sp_overlap <- calculate_spatial_overlap(biomass_spatial = ref_bio_sp,
                                        dietmatrix = dietmatrix,
                                        agemat = agemat)
```

change_avail *Change the availability matrix to simplify automated ATLANTIS calibrations.*

Description

Change the availability of predator XXX on specific preygroups.

Usage

```
change_avail(dietmatrix, pred = NULL, pred_stanza = NULL, prey = NULL,
  roc, relative = TRUE, consecutive = FALSE)
```

Arguments

dietmatrix	Dataframe in 'long' format containing information about availabilities with columns 'pred', 'prey', 'pred_stanza', 'prey_stanza', 'code', 'prey_id' and 'avail'. The dataframe should be generated with load_dietmatrix().
pred	Character vector of predator Acronyms (see get_acronyms()). Selecting NULL as pred results in all predators being selected. This can be helpful if you want to increase the feeding pressure on a specific prey item by all groups. Default is NULL.
pred_stanza	Integer vector indicating if the predator is juvenile (= 1) or adult (= 2). pred and pred_stanza need to be of the same length. In rare instances, e.g. pred_stanza being NULL or one single integer either all pred_stanzas are selected or the single pred_stanza is applied to all predators. Default is NULL.
prey	List of character vectors of prey Acronyms (see get_acronyms()). pred and prey need to be of the same length. Selecting NULL as prey results in all prey groups being selected. This can be helpful if you want to increase the available prey for a specific predator overall. Default is NULL.
roc	Vector of multiplication factors which shall be applied to the old set of parameters. Please supply one value per selected group. In case relative is FALSE the new absolute values can be passed as roc.
relative	Logical if TRUE values are changed relative to base values. If FALSE new values can be passed directly. Default is NULL.
consecutive	Boolean indicating if multiple calls to change_avail are performed one after another TRUE. Default is FALSE.

Value

parameterfile *.prm file with the new parameter values.

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
dm <- load_dietmatrix(prm_biol = file.path(d, "VMPA_setas_biol_fishing_Trunk.prm"),
                      fgs = file.path(d, "SETasGroupsDem_NoCep.csv"))

dm1 <- change_avail(dietmatrix = dm,
                    pred = "FPS",
                    pred_stanza = 1,
                    prey = "CEP",
                    roc = 0.1234,
                    relative = FALSE)
# Show only rows with availability of 0.1234
dm1[apply(apply(dm1[, 5:ncol(dm1)], MARGIN = 2, function(x) x == 0.1234), MARGIN = 1, any), ]

dm2 <- change_avail(dietmatrix = dm,
                    pred = c("FPS", "FVS"),
                    pred_stanza = c(1, 2),
                    prey = list(c("FPS", "FVS"), c("FPS", "FVS")),
                    roc = list(c(0.1111, 0.2222), c(0.3333, 0.4444)),
                    relative = FALSE)

# Show only rows with availability of 0.1111, 0.2222, 0.3333 or 0.4444
dm2[apply(apply(dm2[, 5:ncol(dm2)], MARGIN = 2,
function(x) is.element(x,  c(0.1111, 0.2222, 0.3333, 0.4444))), MARGIN = 1, any), ]
```

---

change_prm *Change biological parameterfile to simplify automated ATLANTIS calibrations.*

Description

This function is used to help automate the calibration routine for ATLANTIS models.

Usage

```
change_prm(prm_biol, select_acronyms, roc, parameter, relative = TRUE,
  save_to_disc = TRUE, version_flag = 2)
```

Arguments

prm_biol	Character string giving the connection to the biological parameterfile. The filename usually contains `biol_fishing` and does end in `.prm`.
select_acronyms	Character vector of functional groups which shall be read in. Names have to match the ones used in the *.prm file. Check column "Code" in "functionalGroups.csv" for clarification.
roc	Vector of multiplication factors which shall be applied to the old set of parameters. Please supply one value per selected group. In case relative is FALSE the new absolute values can be passed as roc.

parameter	Character value of the model parameter which shall be changed. Only one parameter can be selected per function call.
relative	Logical if TRUE values are changed relative to base values. If FALSE new values can be passed directly. Default is TRUE.
save_to_disc	Logical indicating if the resulting prm file should be overwritten (TRUE) or not (FALSE). Defaults to TRUE.
version_flag	The version of ATLANTIS model. 1 for bec_dev, 2 for trunk. default is 2..

Value

parameterfile *.prm file with the new parameter values.

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")

new_prm <- change_prm(prm_biol,
                      select_acronyms = c("FPS", "FVS"),
                      roc = c(2,3),
                      parameter = "KWRR",
                      save_to_disc = FALSE)
```

---

change_prm_cohort *Change biological parameterfile for parameters which expect multiple values.*

Description

This function is used to help automate the calibration routine for ATLANTIS models.

Usage

```
change_prm_cohort(prm_biol, select_acronyms, roc, parameter, relative = TRUE,
  save_to_disc = TRUE)
```

Arguments

prm_biol	Character string giving the connection to the biological parameterfile. The filename usually contains biol_fishing and does end in .prm.
select_acronyms	Character vector of functional groups which shall be read in. Names have to match the ones used in the *.prm file. Check column "Code" in "functionalGroups.csv" for clarification.

roc	Matrix of multiplication factors which shall be applied to the old set of parameters. Please supply one row per selected group. Each row should have as many entries as the parameter itself. E.g. if you want to change the clearance rate for two fish groups you need to supply a matrix with 2 rows and 10 columns. In case you use different cohort numbers for age-structured groups supply a list of multiplication factors. Each list entry should be group specific.
parameter	Character value of the model parameter which shall be changed. Only one parameter can be selected per function call.
relative	Logical if TRUE values are changed relative to base values. If FALSE new values can be passed directly. Default is TRUE.
save_to_disc	Logical indicating if the resulting prm file should be overwritten (TRUE) or not (FALSE). Defaults to TRUE.

Value

parameterfile *.prm file with the new parameter values.

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")

new_prm <- change_prm_cohort(prm_biol = file.path(d, "VMPA_setas_biol_fishing_Trunk.prm"),
                             select_acronyms = c("FPS", "FVS"),
                             roc = matrix(rep(2, times = 20), nrow = 2, ncol = 10),
                             parameter = "C",
                             save_to_disc = FALSE)
# C_FPS is in line 640. Old values are 0.0002 0.3 0.6 0.6 0.6 0.6 0.5 0.5 0.4 and 0.4.
new_prm[640:641]
# C_FVS is in line 652. Old values are 40.0 40.0 40.0 120.0 150.0 250.0 250.0 300.0 300.0 and 300.0.
new_prm[652:653]

# Also works for lists as argument
new_prm <- change_prm_cohort(prm_biol = file.path(d, "VMPA_setas_biol_fishing_Trunk.prm"),
                             select_acronyms = c("FPS", "FVS"),
                             roc = list(rep(3, times = 10), rep(2, times = 10)),
                             parameter = "C",
                             save_to_disc = FALSE)
```

---

check_df_names *Function to check the names of a dataframe.*

Description

This function is used in most plotting routines to check if the correct dataframe is passed.

Usage

```
check_df_names(data, expect, optional = NULL)
```

Arguments

data	Dataframe to check.
expect	Character vector giving the names of the expected columns
optional	Character vector giving the names of optional columns. Default is NULL.

Examples

```
## Not run:
check_df_names(preprocess$biomass_age, expect = c("time", "species", "atoutput", "ages"))

## End(Not run)
```

---

check_growth *This function is used to check the individual growth per group over time.*

Description

This function is used to check the individual growth per group over time.

Usage

```
check_growth(data, yearly = FALSE)
```

Arguments

data	Dataframe with information about individual age based growth over time. This should be generated with preprocess(). You can test either structural or reserve weight.
yearly	Logical specifying if relative change in individual weight shall be calculated on a yearly basis (TRUE) ot not (FALSE). Default is FALSE.

Value

Dataframe showing the output of the linear model fit (slope & F-statistic) per group and age.

Examples

```
check_growth(preprocess$structn_age)
check_growth(preprocess$resn_age)
check_growth(preprocess$resn_age, yearly = TRUE)
```

combine_ages — *Combine ageclasses to juvenile and adult stanza according to age at maturity.*

Description

Combine ageclasses to juvenile and adult stanza according to age at maturity.

Usage

```
combine_ages(data, grp_col, agemat, value_col = "atoutput")
```

Arguments

data	Dataframe with ageclass specific information.
grp_col	Character string giving the name of the group column in data. E.g. 'species', 'pred', 'prey' etc.
agemat	First mature age class for age structured groups. This dataframe should be generated with prm_to_df using "age_mat" as parameter.
value_col	Character string giving the name of the column to sum. Default is "atoutput".

Value

Dataframe with ageclasses combined to stanzas.

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")

agemat <- prm_to_df(prm_biol = prm_biol, fgs = fgs,
                    group = get_age_acronyms(fgs = fgs),
                    parameter = "age_mat")

combine_ages(ref_nums, grp_col = "species", agemat = agemat)
```

combine_groups — *Combine values from different groups if specific groups only have a low contribution to the overall value.*

Description

Combine values from different groups if specific groups only have a low contribution to the overall value.

Usage

```
combine_groups(data, group_col, groups = names(data)[!is.element(names(data),
  c("atoutput", "time", group_col))], combine_thresh = 15)
```

Arguments

data	Dataframe whose groups shall be combined.
group_col	Character string giving the name of the group column in ‘data‘.
groups	Vector of character strings giving the grouping variables.
combine_thresh	Integer indicating the number of groups to display. Default is 15.

Value

dataframe with groups combined to "Rest" if contribution is low.

See Also

Other combine functions: combine_runs

Examples

```
df <- combine_groups(ref_dm, group_col = "prey")
df <- combine_groups(ref_dm, group_col = "prey", combine_thresh = 2)
```

---

combine_runs *This function is used to combine model output from different simulations!*

Description

This function is used to combine model output from different simulations!

Usage

```
combine_runs(outs, runs)
```

Arguments

outs	List of preprocessed Atlantis simulations. Each entry is a list generated with ‘preprocess()‘.
runs	Vector of character strings giving the name of each simulation settings.

Value

Named list with the the same format as in ‘preprocess()‘. Each dataframe has an additional column run.

See Also

Other combine functions: combine_groups

Examples

```
outs <- list(preprocess, preprocess)
runs <- c("run1", "run2")
test <- combine_runs(outs, runs)
names(test[[1]])
head(test[[1]])
```

---

convert_bgm *Transform data from bgm-file to map dataframe.*

Description

Transform data from bgm-file to map dataframe.

Usage

```
convert_bgm(bgm)
```

Arguments

bgm	Character string giving the connection to the atlantis bgm file. The filename ends in .bgm.

See Also

Other convert functions: convert_factor, convert_time

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
bgm <- file.path(d, "VMPA_setas.bgm")

bgm <- convert_bgm(bgm)
head(bgm)
```

convert_factor *Function to convert any column with information about functional groups to a factor whose levels use the LongName of the functional groups file.*

Description

This function is used to match the labels of the plots!

Usage

```
convert_factor(data_fgs, col)
```

Arguments

data_fgs	Dataframe with information about functionalGroups. Usually loaded with load_fgs().
col	Column of the dataframe which is converted to a factor.

Value

Column of a dataframe in factor format.

See Also

Other convert functions: convert_bgm, convert_time

convert_relative_initial *Calculate relative timeseries using the initial value as benchmark.*

Description

Calculate relative timeseries using the initial value as benchmark.

Usage

```
convert_relative_initial(data, col = "atoutput")
```

Arguments

data	Dataframe to apply the transformation to.
col	Character value giving the name of the column to transform. Default is "atoutput".

Value

Dataframe with transformed column 'col'.

Examples

```
df <- convert_relative_initial(ref_structn)
head(df[df$layer == 1, ], n = 15)
```

`convert_time` *Convert timestep to actual time!*

Description

Convert timestep to actual time!

Usage

```
convert_time(prm_run, col)
```

Arguments

`prm_run`	Character string giving the connection of the run parameterfile. The filename usually contains `run_fishing` and ends in `.prm`".
`col`	Numeric vector. Usually a column in a dataframe with information about time. Either given as timesteps or days.

Value

Numeric vector with the time in years.

See Also

Other convert functions: `convert_bgm`, `convert_factor`

`custom_grid` *Utility functions used for various plotting routines within atlantistools.*

Description

Utility functions used for various plotting routines within atlantistools.

Usage

```
custom_grid(plot, grid_x, grid_y)
```

Arguments

`plot`	ggplot2 plot.
`grid_x`	character vector used in facet_grid in horizontal direction.
`grid_y`	character vector used in facet_grid in vertical direction.

Value

ggplot2 plot.

`extract_prm` *Extract values for Atlantis parameters from the biological parameter file.*

Description

Extract values for Atlantis parameters from the biological parameter file.

Usage

```
extract_prm(prm_biol, variables)

extract_prm_cohort(prm_biol, variables)
```

Arguments

`prm_biol`	Character string giving the connection to the biological parameterfile. The filename usually contains `biol_fishing` and does end in `.prm`.
`variables`	Character string giving the flag to search for. This should be a combination of the parameter name and the group-Code.

Value

numeric vector.

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")

# You can pass a single variable
extract_prm(prm_biol, variables = "KWRR_FVS")

# Or multiple variables
extract_prm(prm_biol, variables = paste("KWRR", c("FVS", "FPS"), sep = "_"))

# Use extract_prm_cohort do extract data for age specific parameters.
# They are usually stored in the next line following the parameter tag.
extract_prm_cohort(prm_biol, variables = "C_FVS")
extract_prm_cohort(prm_biol, variables = paste("C", c("FVS", "FPS"), sep = "_"))
```

fishbase_data *fishbase_data*

Description

A table of all the the species found in Fihsbase, including taxonomic classification.

Usage

```
fishbase_data
```

Format

A data frame with 33104 rows and 12 variables:

`SpecCode` integer. Species code.

`Genus` character.

`Species` character.

`SpeciesRefNo` integer. Reference number.

`FBname` character. Fishbase name.

`SubFamily` character.

`FamCode` integer. Family Code.

`GenCode` integer. Genetic Code.

`SubGenCode` integer. Sub Genetic Code.

`Family` character.

`Order` character.

`Class` character.

Source

http://www.fishbase.org/ rfishbase::fishbase

flip_layers *Flip layers for visualization.*

Description

Within Atlantis the water column id 0 is the water column closest to the sediment. In order to simplify graphical interpretation of vertical plots this order is reversed. The surface layer is 1 by default. The sediment layer id is equal to the number of total layers. Please note that this is only used for graphical display.

Usage

```
flip_layers(data)
```

Arguments

data	dataframe with columns polygon and layer. layer id is based on atlantis output (0 = layer closest to the sediment)

Value

dataframe with flipped layerids. 1 = surface.

Examples

```
data <- rbind(expand.grid(species = "sp1", polygon = 0, layer = 0:7),
              expand.grid(species = "sp1", polygon = 1, layer = 0:4),
              expand.grid(species = "sp1", polygon = 2, layer = 0:2),
              expand.grid(species = "sp1", polygon = 3, layer = c(0:3, 7)))
data$atoutput <- runif(nrow(data), min = 0, max = 2)
flip_layers(data)
```

get_boundary *Get boundary boxes from Atlantis box information.*

Description

Use the output from load_box and obtain a vector specifying which boxes are along the boundary.

Usage

```
get_boundary(boxinfo)
```

Arguments

boxinfo	A list as returned from load_box.

Value

A vector specifying which boxes are on the boundary.

Author(s)

Kelli Faye Johnson

See Also

load_box

Other get functions: get_colpal, get_conv_mgnbiot, get_groups

Other get functions: get_colpal, get_conv_mgnbiot, get_groups

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
boxes <- load_box(bgm = file.path(d, "VMPA_setas.bgm"))
get_boundary(boxinfo = boxes)
```

---

get_colpal *Create discrete color palette used in plots.*

Description

The colors are derived from http://colorbrewer2.org/ using the palette 12-class Paired. In addition 9 different gray tones were added in case more than 12 categories are needed (This happens quite often with feeding data).

Usage

```
get_colpal()
```

Value

Vector of colors in hexadecimal code.

See Also

Other get functions: get_boundary, get_conv_mgnbiot, get_groups

get_conv_mgnbiot *Extract conversion factor used to transform data from nitrogen in mg to biomass in tonnes.*

Description

Extract conversion factor used to transform data from nitrogen in mg to biomass in tonnes.

Usage

```
get_conv_mgnbiot(prm_biol)
```

Arguments

prm_biol	Character string giving the connection to the biological parameterfile. The filename usually contains biol_fishing and does end in .prm.

Value

Conversion factor as numeric value.

See Also

Other get functions: get_boundary, get_colpal, get_groups

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")

get_conv_mgnbiot(prm_biol)
```

get_diet_fishbase *Extract reference for diet information from http:://www.fishbase.se*

Description

This function extracts reference for diet information from http://www.fishbase.se

Usage

```
get_diet_fishbase(fish, mirror = "se")
```

Arguments

fish	Vector of fish species with genus and species information.
mirror	Character string defining the url mirror to use. Defaults to se. In case data extraction is slow use a different mirror. Try to avoid frequently used mirrors like uk or com.

Value

Dataframe with species, country, locality, linf and k.

Examples

```
## Not run:
# For some reason the examples break with appveyor.
fish <- c("Gadus morhua", "Merlangius merlangus", "Maurolicus muelleri")
diet <- get_diet_fishbase(fish)

fish <- c("Gadus morhua", "Merlangius merlangus", "Ammodytes marinus")
diet <- get_diet_fishbase(fish)

## End(Not run)
```

---

get_groups *Collection of similar functions which get specific columns from the Atlantis* functionalGroups.csv

---

Description

This collection of functions uses the dataframe of functional groups created with load_fgs and creates various character strings of group names or acronym names.

Usage

```
get_groups(fgs)

get_age_groups(fgs)

get_acronyms(fgs)

get_age_acronyms(fgs)

get_nonage_acronyms(fgs)

get_fish_acronyms(fgs)
```

Arguments

fgs	Character string giving the connection to the functional groups file. The filename usually contains Groups and does end in .csv.

Details

Currently, the following character strings can be created - get_groups: Extract column "Name" - get_age_groups: Extract column "Name". Selects groups with 10 ageclasses. - get_acronym: Extract column "Code" - get_age_acronym: Extract column "Code". Selects groups with 10 ageclasses. - get_nonage_acronym: Extracts columns "Code". Only groups with ageclasses different from 10 are selected. - get_fish_acronyms: Extract column "Code". Only groups with InvertType equal to "FISH" or "SHARK" are selected.

Value

Character string.

See Also

Other get functions: get_boundary, get_colpal, get_conv_mgnbiot

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")

get_age_groups(fgs)
get_nonage_acronyms(fgs)
```

---

get_growth_fishbase *Extract growth parameters from http:://www.fishbase.se.*

Description

This function extracts values for Linf, k and t0 from http://www.fishbase.se

Usage

```
get_growth_fishbase(fish, mirror = "se")
```

Arguments

fish	Vector of fish species with genus and species information.
mirror	Character string defining the url mirror to use. Defaults to se. In case data extraction is slow use a different mirror. Try to avoid frequently used mirrors like uk or com.

Details

Before the actual extraction takes place fishbase IDs for every species are extracted using `get_ids_fishbase`. The IDs are needed to generate the urls later on.

Value

Dataframe with species, country, locality, linf and k.

Examples

```
## Not run:
# For some reason the examples break with appveyor.
fish <- c("Gadus morhua", "Merlangius merlangus")
df <- get_growth_fishbase(fish)
head(df)

df <- get_growth_fishbase(fish, mirror = "de")
head(df)

fish <- c("Sprattus sprattus")
df <- get_growth_fishbase(fish)
head(df)
# Only use for debugging purposes.
fish <- read.csv("Z:/my_data_alex/fish_species_names_from_ibts.csv", stringsAsFactors = FALSE)[, 1]
url <- get_growth_fishbase(fish)
url <- urls$ref_url

## End(Not run)
```

`get_ids_fishbase` *Extract fishbase IDs using the package "rfishbase" to generate species specific fishbase URLs*

Description

This function extracts fishbase IDs using the database provided by the `rfishbase` package.

Usage

```
get_ids_fishbase(fish)
```

Arguments

`fish`	Vector of fish species with genus and species information.

Details

The function depends on the package "rfishbase" which creates a local copy of the fishbase database. The IDs are needed to generate URLs to scan www.fishbase.se for detailed informations about fish growth for example.

Value

named vector with species names and fishbase IDs.

Examples

```
fish <- c("Gadus morhua", "Merlangius merlangus", "Clupea harengus")
get_ids_fishbase(fish)
```

`get_maturity_fishbase` *Extract maturity parameters from http:://www.fishbase.se.*

Description

This function extracts values for maturity from http:://www.fishbase.se

Usage

```
get_maturity_fishbase(fish, mirror = "se")
```

Arguments

`fish`	Vector of fish species with genus and species information.
`mirror`	Character string defining the url mirror to use. Defaults to `se`. In case data extraction is slow use a different mirror. Try to avoid frequently used mirrors like `uk` or `com`.

Details

Before the actual extraction takes place fishbase IDs for every species are extracted using `get_ids_fishbase`. The IDs are needed to generate the urls later on.

Value

Dataframe with species, country, locality, linf and k.

Examples

```
## Not run:
# For some reason the examples break with appveyor.
fish <- c("Gadus morhua", "Squalus acanthias")
df <- get_maturity_fishbase(fish)
head(df)

## End(Not run)
```

get_ref_fishbase *Extract the bibliographic info from www.fishbase.org.*

Description

Extract bibliographic information for growth parameters (linf, k, t0) from www.fishbase.org

Usage

```
get_ref_fishbase(ref_id, mirror = "se")
```

Arguments

ref_id	vector of reference ids.
mirror	Character string defining the url mirror to use. Defaults to se. In case data extraction is slow use a different mirror. Try to avoid frequently used mirrors like uk or com.

Value

Dataframe

Examples

```
## Not run:
df <- get_growth_fishbase("Scyliorhinus canicula")

df$data_ref[df$data_ref == df$main_ref] <- NA
df <- tidyr::gather_(data = df,
                     key_col = "ref_type",
                     value_col = "ref_id",
                     gather_cols = c("main_ref", "data_ref"), na.rm = TRUE)
ref_id <- unique(df$ref_id)
get_ref_fishbase(ref_id)

## End(Not run)
```

load_box *Load the box specification file for an Atlantis scenario*

Description

Read in the .bgm file that specifies the the box coordinates for an Atlantis scenario.

Usage

```
load_box(bgm)
```

Arguments

bgm	Character string giving the connection to the atlantis bgm file. The filename ends in .bgm.

Value

A list of information regarding boxes for an Atlantis scenario.

Author(s)

Kelli Faye Johnson

See Also

Other load functions: load_bps, load_dietcheck, load_fgs, load_init_age, load_init, load_nc_physics, load_nc, load_rec, load_spec_mort, load_txt

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
bgm <- file.path(d, "VMPA_setas.bgm")

boxes <- load_box(bgm)
```

---

load_bps *Extracts the names of the epibenthic biomasspools from the initial conditions file.*

Description

Use fgs data.frame as read in by load_fgs to get the biomass pool information.

Usage

```
load_bps(fgs, init)
```

Arguments

fgs	Character string giving the connection to the functional groups file. The filename usually contains Groups and does end in .csv.
init	Character string giving the connection of the initial conditions netcdf file. The filename usually contains init and ends in .nc.

Value

Character vector of epibenthic biomass pools.

See Also

load_fgs

Other load functions: load_box, load_dietcheck, load_fgs, load_init_age, load_init, load_nc_physics, load_nc, load_rec, load_spec_mort, load_txt

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")

fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")
init <- file.path(d, "INIT_VMPA_Jan2015.nc")

load_bps(fgs, init)
```

---

load_dietcheck *Read in the atlantis dietcheck.txt file and perform some basic data transformations.*

Description

Read in the atlantis dietcheck.txt file and perform some basic data transformations.

Usage

```
load_dietcheck(dietcheck, fgs, prm_run, convert_names = FALSE,
  report = FALSE, version_flag = 2)
```

Arguments

dietcheck	Character string giving the connection of the dietcheck file. The filename usually contains Dietcheck and ends in .txt".
fgs	Character string giving the connection to the functional groups file. The filename usually contains Groups and does end in .csv.
prm_run	Character string giving the connection of the run parameterfile. The filename usually contains run_fishing and ends in .prm".
convert_names	Logical indicating if group codes are transformed to LongNames (TRUE) or not (default = FALSE).
report	Logical indicating if incomplete DietCheck information shall be printed TRUE or not FALSE.
version_flag	The version of ATLANTIS model. 1 for bec_dev, 2 for trunk. default is 2..

Value

A data.frame in long format with the following column names: time, pred, habitat, prey and atoutput (i.e., variable).

See Also

Other load functions: load_box, load_bps, load_fgs, load_init_age, load_init, load_nc_physics, load_nc, load_rec, load_spec_mort, load_txt

Examples

```
# Apply to bec-dev models.
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
dietcheck <- file.path(d, "outputSETASDietCheck.txt")
fgs <- file.path(d, "SETasGroups.csv")
prm_run <- file.path(d, "VMPA_setas_run_fishing_F_New.prm")

diet <- load_dietcheck(dietcheck, fgs, prm_run, version_flag = 1)
head(diet, n = 10)

# Apply to trunk models.
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
dietcheck <- file.path(d, "outputSETASDietCheck.txt")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")
prm_run <- file.path(d, "VMPA_setas_run_fishing_F_Trunk.prm")
diet <- load_dietcheck(dietcheck, fgs, prm_run)
head(diet, n = 10)
```

---

load_dietmatrix *Extract the dietmatrix from the biological parameterfile*

Description

Extracts the diet matrix as long dataframe from the biological paremeter file of any ATLANTIS simulation.

Usage

```
load_dietmatrix(prm_biol, fgs, transform = TRUE, convert_names = FALSE,
  version_flag = 2)

write_diet(dietmatrix, prm_biol, save_to_disc = TRUE)
```

Arguments

prm_biol	Character string giving the connection to the biological parameterfile. The filename usually contains biol_fishing and does end in .prm.
fgs	Character string giving the connection to the functional groups file. The filename usually contains Groups and does end in .csv.
transform	Boolean indicating if the returned dataframe is displayed in "long" (transform = TRUE, default) or "wide" (transform = FALSE) format. You should use the "wide" format in case you aim to change your diet matrix entries.

convert_names	Logical indicating if group codes are transformed to LongNames (TRUE) or not (default = FALSE).
version_flag	The version of ATLANTIS model. 1 for bec_dev, 2 for trunk. default is 2..
dietmatrix	Dataframe of the ATLANTIS dietmatrix generated with load_dietmatrix using transform = FALSE.
save_to_disc	Logical indicating if the resulting prm file should be overwritten (TRUE) or not (FALSE). Defaults to TRUE.

Value

dataframe of the availability matrix in long format with columns pred, pred_stanza (1 = juvenile, 2 = adult), prey_stanza, prey, avail, code.

Examples

```
# Can be applied to trunk models.
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")

dm <- load_dietmatrix(prm_biol, fgs)
head(dm, n = 10)

# Use write_diet to update your existing parameterfile.
dietmatrix <- load_dietmatrix(prm_biol, fgs, transform = FALSE)

# Write is set to FALSE here for technical reasons. Make sure to set it to TRUE in case you
# want to update your file.
new_diet <- write_diet(dietmatrix, prm_biol, save_to_disc = FALSE)

# And to bec-dev models.
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_New.prm")
fgs <- file.path(d, "SETasGroups.csv")

dm <- load_dietmatrix(prm_biol, fgs, version_flag = 1)
head(dm, n = 10)
```

---

load_fgs *Load the functional group file*

Description

Read in the functional group file as dataframe.

Usage

```
load_fgs(fgs)
```

Arguments

fgs	Character string giving the connection to the functional groups file. The filename usually contains Groups and does end in .csv.

Value

A data.frame of functional group information.

See Also

Other load functions: load_box, load_bps, load_dietcheck, load_init_age, load_init, load_nc_physics, load_nc, load_rec, load_spec_mort, load_txt

Examples

```
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
file <- "SETasGroups.csv"
fgs <- load_fgs(file.path(d, file))
head(fgs)

d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
file <- "SETasGroupsDem_NoCep.csv"
fgs <- load_fgs(file.path(d, file))
head(fgs)
```

---

load_init *This function is used to read in data from the initial conditions file.*

Description

This function is used to read in data from the initial conditions file.

Usage

```
load_init(init, vars)
```

Arguments

init	Character string giving the connection of the initial conditions netcdf file. The filename usually contains init and ends in .nc.
vars	Vector of character strings giving the variables to extract from the netcdf file.

Value

A list of dataframes with columns atoutput, polygon and layer (if present).

Author(s)

Alexander Keth

See Also

Other load functions: load_box, load_bps, load_dietcheck, load_fgs, load_init_age, load_nc_physics, load_nc, load_rec, load_spec_mort, load_txt

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
init <- file.path(d, "INIT_VMPA_Jan2015.nc")

load_init(init, vars = "Planktiv_S_Fish1_Nums")
load_init(init, vars = c("Planktiv_S_Fish2_ResN", "Planktiv_S_Fish3_ResN"))
load_init(init, vars = "Megazoobenthos_N")

## Not run:
dir <- "C:/Users/siebo/Documents/Atlantis/BalticAtlantis/run_files_73days_Nej"
init <- file.path(dir, "new_init_Baltic_05Dec2015_v2.nc")
vars <- "Sprat1_ResN"
load_init(init = init, vars = vars)

## End(Not run)
```

---

load_init_age *This function loads weight at age data (in mgN) from the initial conditions file.*

Description

This function loads weight at age data (in mgN) from the initial conditions file.

Usage

```
load_init_age(init, fgs, select_variable, select_groups = NULL, bboxes)

load_init_nonage(init, fgs, select_variable = "N", select_groups = NULL,
  bboxes, bps)

load_init_stanza(init, fgs, select_variable = "N", select_groups = NULL,
  bboxes)

load_init_physics(init, select_variable, bboxes)

load_init_weight(init, fgs, bboxes)
```

Arguments

init	Character string giving the connection of the initial conditions netcdf file. The filename usually contains init and ends in .nc.

fgs	Character string giving the connection to the functional groups file. The filename usually contains Groups and does end in .csv.
select_variable	Character value specifying which variable to load. For load_init_age this can be "Nums", "ResN", "StructN", For load_init_nonage please select "N" (default) For load_init_physics simply pass the names of the physical variables.
select_groups	Character vector of functional groups which shall be read in. Names have to match the ones used in the ncdf file. Check column "Name" in "functionalGroups.csv" for clarification. Default is NULL resulting in all available groups.
bboxes	Integer vector giving the box-id of the boundary boxes. Can be created with get_boundary.
bps	Vector of character strings giving the complete list of epibenthic functional groups (Only present in the sediment layer). The names have to match the column 'Name' in the 'functionalGroups.csv' file. Can be created with load_bps.#'

Value

A dataframes with columns atoutput, polygon, layer (if present), species (if present).

Author(s)

Alexander Keth

See Also

Other load functions: load_box, load_bps, load_dietcheck, load_fgs, load_init, load_nc_physics, load_nc, load_rec, load_spec_mort, load_txt

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
init <- file.path(d, "INIT_VMPA_Jan2015.nc")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")
bboxes <- get_boundary(load_box(bgm = file.path(d, "VMPA_setas.bgm")))

bps <- load_bps(fgs = fgs, init = init)

# There are no values in the initial conditions file. Therefore atoutput is NA.
load_init_age(init = init, fgs = fgs, bboxes = bboxes,
              select_variable = "ResN",
              select_groups = "Planktiv_S_Fish")

load_init_age(init = init, fgs = fgs, bboxes = bboxes, select_variable = "ResN")
load_init_nonage(init = init, fgs = fgs, bboxes = bboxes, bps = bps,
   select_groups = "Megazoobenthos")
load_init_nonage(init = init, fgs = fgs, bboxes = bboxes, bps = bps)
load_init_stanza(init = init, fgs = fgs, bboxes = bboxes)
load_init_weight(init = init, fgs = fgs, bboxes = bboxes)
```

load_nc *Load Atlantis outputfiles (netcdf)*

Description

This function loads Atlantis outputfiles (any netcdf file) and converts them to a data.frame.

Usage

```
load_nc(nc, fgs, bps, select_groups, select_variable, prm_run, bboxes,
  check_acronyms = TRUE, warn_zeros = FALSE, report = TRUE)
```

Arguments

nc	Character string giving the connection of the netcdf file to read in. The filename usually contains output and ends in .nc".
fgs	Character string giving the connection to the functional groups file. The filename usually contains Groups and does end in .csv.
bps	Vector of character strings giving the complete list of epibenthic functional groups (Only present in the sediment layer). The names have to match the column 'Name' in the 'functionalGroups.csv' file. Can be created with load_bps.#'
select_groups	Character vector of funtional groups which shall be read in. Names have to match the ones used in the netcdf file. Check column "Name" in "functionalGroups.csv" for clarification.
select_variable	Character value specifying which variable to load. Only one variable of the options available (i.e., c("N", "Nums", "ResN", "StructN", "Eat", "Growth", "Prodn", "Grazing') can be loaded at a time.
prm_run	Character string giving the connection of the run parameterfile. The filename usually contains run_fishing and ends in .prm".
bboxes	Integer vector giving the box-id of the boundary boxes. Can be created with get_boundary.
check_acronyms	Logical testing if functional-groups in select_groups are inactive in the current model run. They will be omitted in the output.
warn_zeros	Logical indicating if check for actual zeros in the data shall be printed or not. Default is FALSE.
report	Logical indicating if progress bars shall be printed (TRUE) or not (FALSE). Default is TRUE.

Value

A data.frame in long format with the following column names: species, timestep, polygon, agecl, and atoutput (i.e., variable).

Author(s)

Alexander Keth

See Also

Other load functions: load_box, load_bps, load_dietcheck, load_fgs, load_init_age, load_init, load_nc_physics, load_rec, load_spec_mort, load_txt

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")

nc <- file.path(d, "outputSETAS.nc")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")
bps <- load_bps(init = file.path(d, "INIT_VMPA_Jan2015.nc"), fgs = fgs)
bboxes <- get_boundary(boxinfo = load_box(bgm = file.path(d, "VMPA_setas.bgm")))
prm_run <- file.path(d, "VMPA_setas_run_fishing_F_Trunk.prm")

test <- load_nc(nc = nc, bps = bps, fgs = fgs, prm_run = prm_run, bboxes = bboxes,
  select_groups = c("Planktiv_S_Fish", "Cephalopod", "Diatom"),
  select_variable = "ResN")

test <- load_nc(nc = nc, bps = bps, fgs = fgs, prm_run = prm_run, bboxes = bboxes,
  select_groups = c("Planktiv_S_Fish", "Cephalopod", "Diatom"),
  select_variable = "Nums")
```

---

load_nc_physics *Load Atlantis outputfiles (netcdf)*

Description

This function loads Atlantis outputfiles (netcdf) and converts them to a dataframe.

Usage

```
load_nc_physics(nc, select_physics, prm_run, bboxes, aggregate_layers = FALSE,
  warn_zeros = FALSE)
```

Arguments

nc	Character string giving the connection of the netcdf file to read in. The filename usually contains output and ends in .nc".
select_physics	Character vector of physical variables which shall be read in. Names have to match the ones used in the ncdf file.
prm_run	Character string giving the connection of the run parameterfile. The filename usually contains run_fishing and ends in .prm".
bboxes	Integer vector giving the box-id of the boundary boxes. Can be created with get_boundary.

aggregate_layers
: Logical indicating if values for layers should be aggregated (TRUE) or not (FALSE). Default is FALSE.

warn_zeros
: Logical indicating if check for actual zeros in the data shall be printed or not. Default is FALSE.

Details

This functions converts the ATLANTIS output to a dataframe which can be processed in R.

Value

A data.frame in long format with the following column names: variable, time, polygon, layer, and atoutput (i.e., variable).

See Also

Other load functions: load_box, load_bps, load_dietcheck, load_fgs, load_init_age, load_init, load_nc, load_rec, load_spec_mort, load_txt

Examples

```
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
nc <- file.path(d, "outputSETAS.nc")
prm_run <- file.path(d, "VMPA_setas_run_fishing_F_New.prm")
bboxes <- get_boundary(boxinfo = load_box(file.path(d, bgm = "VMPA_setas.bgm")))
select_physics <- c("salt", "NO3", "volume")

test <- load_nc_physics(nc, select_physics, prm_run, bboxes)
str(test)

d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
nc <- file.path(d, "outputSETAS.nc")
prm_run <- file.path(d, "VMPA_setas_run_fishing_F_Trunk.prm")
bboxes <- get_boundary(boxinfo = load_box(file.path(d, bgm = "VMPA_setas.bgm")))

test <- load_nc_physics(nc, select_physics, prm_run, bboxes)
str(test)

test <- load_nc_physics(nc, select_physics = "nominal_dz", prm_run, bboxes)
```

---

load_rec *Load information for SSB and Recruits from an Atlantis model run.*

Description

Load information for SSB and Recruits from an Atlantis model run.

Usage

```
load_rec(yoy, ssb, prm_biol)
```

Arguments

yoy	Character string giving the connection of the YOY file. The filename usually contains outputYOY and ends in .txt".
ssb	Character string giving the connection of the YOY file. The filename usually contains outputSSB and ends in .txt".
prm_biol	Character string giving the connection to the biological parameterfile. The filename usually contains biol_fishing and does end in .prm.

Value

Dataframe with information about ssb in tonnes and recruits in thousands.

See Also

Other load functions: load_box, load_bps, load_dietcheck, load_fgs, load_init_age, load_init, load_nc_physics, load_nc, load_spec_mort, load_txt

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
yoy <- file.path(d, "outputSETASYOY.txt")
ssb <- file.path(d, "outputSETASSSB.txt")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")

load_rec(yoy, ssb, prm_biol)
```

---

load_spec_mort *Load mortality information from outputSpecificPredMort.txt*

Description

Load mortality information from outputSpecificPredMort.txt

Usage

```
load_spec_mort(specmort, prm_run, fgs, convert_names = FALSE,
  version_flag = 2)
```

Arguments

specmort	Character string giving the connection of the specific mortality file. The filename usually contains SpecificPredMort and ends in .txt".
prm_run	Character string giving the connection of the run parameterfile. The filename usually contains run_fishing and ends in .prm".
fgs	Character string giving the connection to the functional groups file. The filename usually contains Groups and does end in .csv.
convert_names	Logical indicating if group codes are transformed to LongNames (TRUE) or not (default = FALSE).
version_flag	The version of ATLANTIS model. 1 for bec_dev, 2 for trunk. default is 2..

Value

Dataframe with information about ssb in tonnes and recruits in thousands.

See Also

Other load functions: load_box, load_bps, load_dietcheck, load_fgs, load_init_age, load_init, load_nc_physics, load_nc, load_rec, load_txt

Examples

```
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
specmort <- file.path(d, "outputSETASSpecificPredMort.txt")
prm_run <- file.path(d, "VMPA_setas_run_fishing_F_New.prm")
fgs <- file.path(d, "SETasGroups.csv")

df <- load_spec_mort(specmort, prm_run, fgs, version_flag = 1)
head(df)

d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
specmort <- file.path(d, "outputSETASSpecificPredMort.txt")
prm_run <- file.path(d, "VMPA_setas_run_fishing_F_Trunk.prm")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")

df <- load_spec_mort(specmort, prm_run, fgs)
head(df)
```

---

load_txt *Function to load various txt files from Atlantis simulations*

Description

Function to load various txt files from Atlantis simulations

Usage

```
load_txt(file, id_col = "Time")
```

Arguments

file	Character string giving the connection of the output file. The filename usually contains output and ends in .txt".
id_col	Character strings giving the names of the columns which are not variables. Data from all other columns will be gathered with tidyr. Default is "Time".

Value

Dataframe in tidy format!

See Also

Other load functions: load_box, load_bps, load_dietcheck, load_fgs, load_init_age, load_init, load_nc_physics, load_nc, load_rec, load_spec_mort

Examples

```
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
file <- file.path(d, "outputSETASSSB.txt")
load_txt(file)

file <- file.path(d, "outputSETASYOY.txt")
load_txt(file)
```

---

plot_add_box *Low level plotting function to add sudo confidence range to calibration plots.*

Description

Low level plotting function to add sudo confidence range to calibration plots.

Usage

```
plot_add_box(plot, range = c(0.5, 0.2))
```

Arguments

plot	ggplot2 object.
range	max and min relative change of data. Default is c(0.5, 0.2).

Value

ggplot2 plot.

See Also

Other low-level-plot functions: plot_add_range

Examples

```
# Make sure to use a relative timeseries generated with \code{\link{convert_relative_initial}}.
df <- convert_relative_initial(preprocess$structn_age)

# Create the plot with \code{\link{plot_line}}.
plot <- plot_line(df, col = "agecl")

# Add lower and upper range.
plot_add_box(plot)

# You can set the upper and lower range as you like!
plot_add_box(plot, range = c(0.8, 0.4))
```

`plot_add_polygon_overview` *Add spatial representation of polygon layout to a ggplot2 object.*

Description

Add spatial representation of polygon layout to a ggplot2 object.

Usage

```
plot_add_polygon_overview(plot, bgm_as_df, polygon_overview = 0.2)
```

Arguments

`plot`	ggplot2 object. Can be a ggplot grob.
`bgm_as_df`	*.bgm file converted to a dataframe. Please use `convert_bgm` to convert your bgm-file to a dataframe with columns 'lat', 'long', 'inside_lat', 'inside_long' and 'polygon'.
`polygon_overview`	numeric value between 0 and 1 indicating the size used to plot the polygon overview in the upper right corner of the plot. Default is `0.2`.

Value

ggplot grob

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
bgm_as_df <- convert_bgm(bgm = file.path(d, "VMPA_setas.bgm"))

p <- plot_line(preprocess$physics, wrap = NULL)
p <- custom_grid(p, grid_x = "polygon", grid_y = "variable")

grob <- plot_add_polygon_overview(p, bgm_as_df)
gridExtra::grid.arrange(grob)
```

plot_add_range *Low level plotting function to add range of observed values to time series plots.*

Description

This function can be used to add the range of observed data to a timeseries plot generated with plot_line. The density of the color gives an indication of the likelihood of the value.

Usage

```
plot_add_range(plot, ex_data)
```

Arguments

plot	ggplot2 object.
ex_data	Dataframe with observed values for the specific timeseries.

Value

ggplot2 plot.

See Also

Other low-level-plot functions: plot_add_box

Examples

```
# There is no external data so we need to add some noise first!
ex_data <- preprocess$biomass
ex_data$atoutput <- ex_data$atoutput * runif(n = nrow(ex_data), 0, 1)
ex_data$model <- "test"

# Create the timeseries with \code{\link{plot_line}}
plot <- plot_line(preprocess$biomass)

# Add the external data as geom_rug with \code{\link{plot_add_range}}
plot_add_range(plot, ex_data)
```

plot_bar *Function to plot relative contribution of biomass and numbers per cohort.*

Description

Function to plot relative contribution of biomass and numbers per cohort.

Usage

```
plot_bar(data, x = "time", y = "atoutput", fill = "species",
  wrap = NULL, ncol = NULL)
```

Arguments

data	Dataframe to be plotted.
x	x-variable. Default is 'time'.
y	y-variable. Default is 'atoutput'.
fill	Column to use as filling colour. Default is "species".
wrap	Wraping column. Default is 'species'
ncol	Number of columns in multipanel plot. Default is 7.

Value

ggplot2 object

See Also

Other plot functions: plot_boxes, plot_diet_bec_dev, plot_diet, plot_line, plot_rec, plot_species

Examples

```
plot_bar(preprocess$biomass)

# Most models have a large number of groups. Please make sure to combine groups with a low
# contribution prior to plotting with \code{\link{combine_groups}}.
df <- combine_groups(preprocess$biomass, group_col = "species", combine_thresh = 3)
plot_bar(df)

# This function can also be used to plot age-specific data.
plot_bar(preprocess$nums_age, fill = "agecl", wrap = "species")

# Please use \code{\link{agg_perc}} to visualize the relative cohort structure over time.
df <- agg_perc(preprocess$nums_age, groups = c("time", "species"))
plot_bar(df, fill = "agecl", wrap = "species")
```

plot_boxes *Plot layout of boxes!*

Description

Plot layout of boxes!

Usage

```
plot_boxes(data, color_boxes = TRUE)
```

Arguments

data	Dataframe to be plotted.
color_boxes	logical indicating if polygons should be color coded or not. Default is TRUE.

Value

ggplot2 object

See Also

Other plot functions: plot_bar, plot_diet_bec_dev, plot_diet, plot_line, plot_rec, plot_species

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
bgm_data <- convert_bgm(file.path(d, "VMPA_setas.bgm"))

# Use color coding for polygons.
plot_boxes(bgm_data)

# Only use text to indicate polygons.
plot_boxes(bgm_data, color_boxes = FALSE)
```

plot_consumed_biomass *Circle diagram to visualize the consumed biomass for the whole system.*

Description

Circle diagram to visualize the consumed biomass for the whole system.

Usage

```
plot_consumed_biomass(bio_consumed, select_time = NULL, show = 0.95)
```

Arguments

bio_consumed	Consumed biomass of prey groups by predatorgroup and agecl in tonnes for each timestep and polygon. Dataframe with columns 'pred', 'agecl', 'polygon', 'time', 'prey'. Consumed biomass in [t] is stored in column 'atoutput'. Should be generated with link{calculate_consumed_biomass}.
select_time	Numeric value to control the simulation time in years to visualize. By default the start of the simulation is shown. Default is NULL.
show	Numeric value between 0 - 1 to control the amount of links shown. Default is 0.95. Thus, the most important 95 are grouped together as 'Rest'.

Value

plot

Examples

```
## Not run:
# Need to fix NAs in init cdf at some point.
plot_consumed_biomass(ref_bio_cons)
plot_consumed_biomass(ref_bio_cons, select_time = 1.8)
plot_consumed_biomass(ref_bio_cons, select_time = 1.8, show = 0.99)

# Add some gns testing.
load("z:/Atlantis_models/Runs/dummy_01_ATLANTIS_NS/preprocess-north-sea.rda", verbose = T)
plot_consumed_biomass(result$biomass_consumed, show = 0.95, select_time = 2.2)

times <- c(0.2, seq(10, 90, 10), 99.8)
par(mfcol = c(3, 4))
for (i in seq_along(times)) {
  plot_consumed_biomass(result$biomass_consumed, show = 0.95, select_time = times[i])
}

times <- seq(0.2, 2.2, 0.2)
# times <- times[times != 1]
par(mfcol = c(3, 4))
for (i in seq_along(times)) {
  plot_consumed_biomass(result$biomass_consumed, show = 0.95, select_time = times[i])
}

## End(Not run)
```

---

plot_diet *Plot contribution of diet contents for each functional group.*

Description

Visualize diet proportions form predator and prey perspective. The upper panel plot shows the predator perspective while the lower panel plot shows the prey perspective for a given group. Please note that this function only works with models based on the trunk code. Bec_dev models should use plot_diet_bec_dev to get an indication of the feeding interactions.

Usage

```
plot_diet(bio_consumed, species = NULL, wrap_col = "agecl",
  combine_thresh = 7)
```

Arguments

bio_consumed	Consumed biomass of prey groups by predatorgroup and agecl in tonnes for each timestep and polygon. Dataframe with columns 'pred', 'agecl', 'polygon', 'time', 'prey'. Consumed biomass in [t] is stored in column 'atoutput'. Should be generated with link{calculate_consumed_biomass}.
species	Character string giving the acronyms of the species you aim to plot. Default is NULL resulting in all available species being ploted.
wrap_col	Character specifying the column of the dataframe to be used as multipanel plot. Default is "agecl".
combine_thresh	Number of different categories to plot. Lets say predator X has eaten 20 different prey items. If you only want to show the 3 most important prey items set combine_thresh to 3. As rule of thumb values < 10 are useful otherwise to many colors are used in the plots. Default is 7.

Value

List of grobs composed of ggplot2 objects.

See Also

Other plot functions: plot_bar, plot_boxes, plot_diet_bec_dev, plot_line, plot_rec, plot_species

Examples

```
## Not run:
plots <- plot_diet(ref_bio_cons, wrap_col = "agecl")
gridExtra::grid.arrange(plots[[1]])
gridExtra::grid.arrange(plots[[7]])

# Use names() to get the species names!
names(plots)

## End(Not run)

plot <- plot_diet(ref_bio_cons, species = "Small planktivorous fish", wrap_col = "agecl")
gridExtra::grid.arrange(plot[[1]])
```

plot_diet_bec_dev *Plot contribution of diet contents for each functional group.*

Description

Visualize diet proportions form predator and prey perspective. The upper panel plot shows the predator perspective while the lower panel plot shows the prey perspective for a given group. Please note that this function uses the SpecPredMort.txt file to visualize feeding interactions and therefore is only an indication of the realized true diet within the model. Please use `plot_diet` instead.

Usage

```
plot_diet_bec_dev(data, species = NULL, wrap_col, combine_thresh = 15)
```

Arguments

`data`	SpecPredMort.txt read in with `load_spec_mort`.
`species`	Character string giving the acronyms of the species you aim to plot. Default is `NULL` resulting in all available species being ploted.
`wrap_col`	Character specifying the column of the dataframe to be used as multipanel plot. In case you aim to plot SpecMort.txt data use either "agecl" or "stanza".
`combine_thresh`	Number of different categories to plot. Lets say predator X has eaten 20 different prey items. If you only want to show the 3 most important prey items set `combine_thresh` to 3. As rule of thumb values < 10 are useful otherwise to many colors are used in the plots. Default is 15.

Value

List of ggplot2 objects.

See Also

Other plot functions: `plot_bar`, `plot_boxes`, `plot_diet`, `plot_line`, `plot_rec`, `plot_species`

Examples

```
## Not run:
# Plot SpecMort.txt per ageclass.
plots <- plot_diet_bec_dev(preprocess_setas$diet_specmort, wrap_col = "agecl")
gridExtra::grid.arrange(plots[[1]])

# Only plot specific species
plots <- plot_diet_bec_dev(preprocess_setas$diet_specmort, species = "CEP", wrap_col = "agecl")
gridExtra::grid.arrange(plots[[1]])

# Plot SpecMort.txt per stanza First we need to transform the ageclasses to stanzas.
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
diet_stanza <- combine_ages(dir = d,
```

```
                              data = preprocess_setas$diet_specmort,
                              col = "pred",
                              prm_biol = "VMPA_setas_biol_fishing_New.prm")
plots <- plot_diet_bec_dev(diet_stanza, wrap_col = "stanza")
gridExtra::grid.arrange(plots[[1]])

## End(Not run)
```

`plot_line` *Function to plot time series of atlantis ncdf output.*

Description

Function to plot time series of atlantis ncdf output.

Usage

```
plot_line(data, x = "time", y = "atoutput", wrap = "species",
  col = NULL, ncol = 7, yexpand = FALSE)
```

Arguments

`data`	Dataframe to be plotted.
`x`	x-variable. Default is `'time'`.
`y`	y-variable. Default is `'atoutput'`.
`wrap`	Wraping column. Default is `'species'`
`col`	Column to use as colour. Default is `NULL`.
`ncol`	Number of columns in multipanel plot. Default is 7.
`yexpand`	Expands the y axis so it always includes 0. Default is `FALSE`.

Value

ggplot2 object

See Also

Other plot functions: `plot_bar`, `plot_boxes`, `plot_diet_bec_dev`, `plot_diet`, `plot_rec`, `plot_species`

Examples

```
plot_line(preprocess$biomass)
plot_line(preprocess$biomass, col = "species")
plot_line(preprocess$biomass_age, col = "agecl")
plot_line(preprocess$biomass_age, wrap = "agecl", col = "species")

# The function can also be used to compare model outoput with observed data.
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
```

```
ex_data <- read.csv(file.path(d, "setas-bench.csv"), stringsAsFactors = FALSE)
names(ex_data)[names(ex_data) == "biomass"] <- "atoutput"

data <- preprocess$biomass
data$model <- "atlantis"
comp <- rbind(ex_data, data, stringsAsFactors = FALSE)

# Show atlantis as first factor!
lev_ord <- c("atlantis", sort(unique(comp$model))[sort(unique(comp$model)) != "atlantis"])
comp$model <- factor(comp$model, levels = lev_ord)

# Create plot
plot_line(comp, col = "model")

## Not run:
# Use \code{\link{convert_relative_initial}} and \code{\link{plot_add_box}}
# with \code{\link{plot_line}}. Use \code{\link{convert_relative_initial}} to
# generate a relative time series first.
df <- convert_relative_initial(preprocess$structn_age)

# Create the base plot with \code{\link{plot_line}}.
plot <- plot_line(df, col = "agecl")

# Add lower and upper range.
plot_add_box(plot)

# Create spatial timeseries plots in conjuction with \code{\link{custom_grid}}.
plot <- plot_line(preprocess$physics, wrap = NULL)
custom_grid(plot, grid_x = "polygon", grid_y = "variable")

plot <- plot_line(preprocess$flux, wrap = NULL, col = "variable")
custom_grid(plot, grid_x = "polygon", grid_y = "layer")

## End(Not run)
```

`plot_rec` *Plot recruitment.*

Description

Plot recruitment.

Usage

```
plot_rec(data, ex_data, ncol = 7)
```

Arguments

`data`	Dataframe with information about ssb and recruits. This is created from atlantis output files YOY.txt and SSB.txt (Usually output[...]YOY.txt' & 'output[...]SSB.txt') using `load_rec`.

ex_data	Dataframe to compare the atlantis run with.
ncol	Number of columns in multipanel plot. Default is 7.

Value

ggplot2 object

See Also

Other plot functions: plot_bar, plot_boxes, plot_diet_bec_dev, plot_diet, plot_line, plot_species

Examples

```
## Not run:
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
ex_data <- read.csv(file.path(d, "setas-ssb-rec.csv"), stringsAsFactors = FALSE)
plot_rec(preprocess_setas$ssb_rec, ex_data)

## End(Not run)
```

---

plot_spatial_box *Visualize the spatial distribution per species and stanza combination.*

Description

Visualize the spatial distribution per species and stanza combination.

Usage

```
plot_spatial_box(bio_spatial, bgm_as_df, select_species = NULL,
  timesteps = 2, polygon_overview = 0.2)
```

Arguments

bio_spatial	Biomass per group and stanza in tonnes for each timestep, layer and polygon. This dataframe should be generated with calculate_biomass_spatial. The columns of the dataframe have to be 'species', 'species_stanza', 'polygon', 'layer', 'time' and 'atoutput'. Column 'atoutput' is the biomass in tonnes. Please use combine_ages to transform an age-based dataframe to a stanza based dataframe.
bgm_as_df	*.bgm file converted to a dataframe. Please use convert_bgm to convert your bgm-file to a dataframe with columns 'lat', 'long', 'inside_lat', 'inside_long' and 'polygon'.
select_species	Character vector listing the species to plot. If no species are selected NULL (default) all available species are plotted.
timesteps	Integer giving the number of timesteps to visualize. Default is 2. By default the start and end of the simulation is shown. In case timesteps > 2 equally spaced timesteps are added.

polygon_overview
: numeric value between 0 and 1 indicating the size used to plot the polygon overview in the upper right corner of the plot. Default is `0.2`.

Value

grob of 3 ggplot2 plots.

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")

bgm_as_df <- convert_bgm(file.path(d, "VMPA_setas.bgm"))

# Spatial distribution in Atlantis is based on adu- and juv stanzas.
# Therefore, we need to aggregate the age-based biomass to
# stanzas with \code{\link{combine_ages}}.
bio_spatial <- combine_ages(ref_bio_sp, grp_col = "species", agemat = ref_agemat)

## Not run:
# Apply \code{\link{plot_spatial_box}}
grobs <- plot_spatial_box(bio_spatial, bgm_as_df, timesteps = 3)
gridExtra::grid.arrange(grobs[[1]])
gridExtra::grid.arrange(grobs[[9]])

# use names() to select specific plots
names(grobs)

## End(Not run)

# Plot specific species
grobs <- plot_spatial_box(bio_spatial, bgm_as_df,
                          select_species = "Shallow piscivorous fish", timesteps = 3)
gridExtra::grid.arrange(grobs[[1]])
```

---

plot_spatial_overlap *Plot spatial overlap.*

Description

Plot spatial overlap.

Usage

```
plot_spatial_overlap(df_list)
```

Arguments

df_list
: List of dataframes generated with calculate_spatial_overlap

Value

ggplot2 plot.

Examples

```
sp_overlap <- calculate_spatial_overlap(ref_bio_sp, ref_dietmatrix, ref_agemat)
## Not run:
plot_spatial_overlap(sp_overlap)

## End(Not run)
```

`plot_spatial_ts` *Visualize the spatial distribution per species and stanza combination.*

Description

Visualize the spatial distribution per species and stanza combination.

Usage

```
plot_spatial_ts(bio_spatial, bgm_as_df, vol, select_species = NULL,
  ncol = 7, polygon_overview = 0.2)
```

Arguments

`bio_spatial`	Biomass per group and stanza in tonnes for each timestep, layer and polygon. This dataframe should be generated with `calculate_biomass_spatial`. The columns of the dataframe have to be 'species', 'species_stanza', 'polygon', 'layer', 'time' and 'atoutput'. Column 'atoutput' is the biomass in tonnes. Please use `combine_ages` to transform an age-based dataframe to a stanza based dataframe.
`bgm_as_df`	*.bgm file converted to a dataframe. Please use `convert_bgm` to convert your bgm-file to a dataframe with columns 'lat', 'long', 'inside_lat', 'inside_long' and 'polygon'.
`vol`	Volume per polygon and timestep. See model-preprocess.Rmd for details.
`select_species`	Character vector listing the species to plot. If no species are selected `NULL` (default) all available species are plotted.
`ncol`	Number of columns in final plot. Default is `7`.
`polygon_overview`	numeric value between 0 and 1 indicating the size used to plot the polygon overview in the upper right corner of the plot. Default is `0.2`.

Value

grob of 3 ggplot2 plots.

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")

bgm_as_df <- convert_bgm(file.path(d, "VMPA_setas.bgm"))
vol <- agg_data(ref_vol, groups = c("time", "polygon"), fun = sum, out = "volume")

# Spatial distribution in Atlantis is based on adu- and juv stanzas.
# Therefore, we need to aggregate the age-based biomass to
# stanzas with \code{\link{combine_ages}}.
bio_spatial <- combine_ages(ref_bio_sp, grp_col = "species", agemat = ref_agemat)

# Apply \code{\link{plot_spatial_ts}}
grobs <- plot_spatial_ts(bio_spatial, bgm_as_df, vol)
gridExtra::grid.arrange(grobs[[1]])
gridExtra::grid.arrange(grobs[[7]])
```

---

plot_species *Create species specific overview plot.*

Description

This plotting routine is based on Raphael's (Ifremer) plotting routine used during model calibration. Currently 6 plots are created by default: - Biomass over time - Biomass over time per age - StructN over time per age - ResN over time per age - Condition over time per age - Numbers over time per age

Usage

```
plot_species(data_pre, species)
```

Arguments

data_pre	List of preprocessed Atlantis simulation. The list of dataframes should be created with `model-preprocess.Rmd`.
species	Character srtring giving the name of the species to plot. Only age based species are supported.

Value

ggplot2 object of class grob

See Also

Other plot functions: plot_bar, plot_boxes, plot_diet_bec_dev, plot_diet, plot_line, plot_rec

Examples

```
plot <- plot_species(preprocess, species = "Shallow piscivorous fish")
# Use grid.arrange to draw the plot on the current device
gridExtra::grid.arrange(plot)
```

`preprocess` *preprocess*

Description

See "data-vignette-model-preprocess.R" and "model-preprocess.Rmd" for further information.

Usage

```
preprocess
```

Format

List with 17 dataframes.

1. biomass
2. biomass_age
3. biomass_consumed
4. biomass_spatial_stanza
5. diet
6. eat_age
7. flux
8. grazing
9. growth_age
10. growth_rel_init
11. nums
12. nums_age
13. nums_box
14. physics
15. resn_age
16. sink
17. spatial_overlap
18. ssb_rec
19. structn_age
20. vol

preprocess_txt *Preprocess dataframes loaded in with* load_txt()

Description

sep_col is split into multiple columns given by into. If column ageclass is present and values start with 0 one is added to align with agestructure in other functions. Columns without any informations (length(unique()) == 1) are dropped. If the first time step only has zeros as values remove these values. remove zeros overall!

Usage

```
preprocess_txt(df_txt, sep_col = "code", into)
```

Arguments

df_txt	Dataframe read in with load_txt().
sep_col	Column to separate into multiple columns. Default is "code".
into	Character vector given the columns to split sep_col in.

Value

Tidy dataframe.

Examples

```
d <- system.file("extdata", "setas-model-new-becdev", package = "atlantistools")
df <- load_txt(file = file.path(d, "outputSETASSpecificPredMort.txt"))
df <- preprocess_txt(df_txt = df, into = c("pred", "agecl", "empty_col1", "prey", "empty_col2"))
head(df)

df <- load_txt(file = file.path(d, "outputSETASSpecificMort.txt"))
df <- preprocess_txt(df_txt = df, into = c("species", "agecl", "empty_col", "mort"))
head(df)
```

prm_to_df *Extract parameters from the biological parameter file and transform them to a dataframe.*

Description

Extract parameters from the biological parameter file and transform them to a dataframe.

Usage

```
prm_to_df(prm_biol, fgs, group, parameter)

prm_to_df_ages(prm_biol, fgs, group, parameter)
```

Arguments

prm_biol	Character string giving the connection to the biological parameterfile. The filename usually contains biol_fishing and does end in .prm.
fgs	Character string giving the connection to the functional groups file. The filename usually contains Groups and does end in .csv.
group	Character vector giving the functional Groups to extract.
parameter	Character vector giving the parameters to extract.

Value

Dataframe with columns 'species' and as many columns as parameters.

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")
group <- c("FPS", "FVS")
parameter <- c("mum", "C")

prm_to_df_ages(prm_biol, fgs, group, parameter)
prm_to_df(prm_biol, fgs, group, parameter)
```

---

ref_agemat *agemat.*

Description

agemat.

Usage

```
ref_agemat
```

Format

species Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

age_mat First mature age class.

ref_bio_cons *Consumed biomass.*

Description

Consumed biomass.

Usage

```
ref_bio_cons
```

Format

pred Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

agecl Ageclass given as integer from 1 to NumCohorts.

polygon Boxid starting from 0 to numboxes - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

prey Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

atoutput Obseravtion column storing the actual output value. Consumed biomass in tonnes.

ref_bio_sp *Spatial biomass.*

Description

Spatial biomass.

Usage

```
ref_bio_sp
```

Format

species Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

agecl Ageclass given as integer from 1 to NumCohorts.

polygon Boxid starting from 0 to numboxes - 1.

layer Layerid starting from 0 to numlayers - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. Biomass in tonnes.

ref_dietmatrix *Dietmatrix.*

Description

See data-raw/data-create-reference-dfs.R for further information.

Usage

```
ref_dietmatrix
```

Format

pred Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

pred_stanza Predator stanza. 1 = juvenile; 2 = adult.

prey_stanza Prey stanza. 1 = juvenile; 2 = adult.

code Flag from the biological parameter file.

prey Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

avail Obseravtion column storing the actual output value. Availability ranging from 0 to 1.

prey_id Preyid index based on the functional groups file.

ref_dm *Dietcheck.*

Description

Dietcheck.

Usage

```
ref_dm
```

Format

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

pred Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

agecl Ageclass given as integer from 1 to NumCohorts.

prey Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

atoutput Obseravtion column storing the actual output value. Diet contribution in percentage.

ref_eat *Eat.*

Description

Eat.

Usage

```
ref_eat
```

Format

species Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

agecl Ageclass given as integer from 1 to NumCohorts.

polygon Boxid starting from 0 to numboxes - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. Consumption in mg N m-3 d-1.

ref_grazing *Grazing.*

Description

Grazing.

Usage

```
ref_grazing
```

Format

species Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

agecl Ageclass given as integer from 1 to NumCohorts.

polygon Boxid starting from 0 to numboxes - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. Grazing in mg N m-3 d-1.

ref_growth *Growth.*

Description

Growth.

Usage

```
ref_growth
```

Format

species Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

agecl Ageclass given as integer from 1 to NumCohorts.

polygon Boxid starting from 0 to numboxes - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. Growth in mg N d-1.

ref_n *Nitrogen.*

Description

Nitrogen.

Usage

```
ref_n
```

Format

species Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

polygon Boxid starting from 0 to numboxes - 1.

layer Layerid starting from 0 to numlayers - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. Nitrogen in mg N m-3.

ref_nums *Numbers at age data.*

Description

Numbers at age data.

Usage

```
ref_nums
```

Format

species Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

agecl Ageclass given as integer from 1 to NumCohorts.

polygon Boxid starting from 0 to numboxes - 1.

layer Layerid starting from 0 to numlayers - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. Numbers.

ref_physics *Physical variables.*

Description

Physical variables.

Usage

```
ref_physics
```

Format

variable Name of the physical variable: "salt", "NO3", "NH3", "Temp", "Chl_a" and "Denitrification".

polygon Boxid starting from 0 to numboxes - 1.

layer Layerid starting from 0 to numlayers - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. units are salt = PSU; NO3, NH3 = mg N m-3; Temp = degrees Celcius; Chl_a, Denitrifiction = ?

ref_resn *Reserve nitrogen.*

Description

Reserve nitrogen.

Usage

```
ref_resn
```

Format

species Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

agecl Ageclass given as integer from 1 to NumCohorts.

polygon Boxid starting from 0 to numboxes - 1.

layer Layerid starting from 0 to numlayers - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. Reserve weight in mg N.

ref_structn *Structural nitrogen.*

Description

Structural nitrogen.

Usage

```
ref_structn
```

Format

species Name of the functional groups given as character string. The names match with the column 'LongName' in the functionalGroups.csv file.

agecl Ageclass given as integer from 1 to NumCohorts.

polygon Boxid starting from 0 to numboxes - 1.

layer Layerid starting from 0 to numlayers - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. Structural weight in mg N.

ref_to_bibkey *Convert reference to bib-tex-key.*

Description

Convert reference to bib-tex-key.

Usage

```
ref_to_bibkey(ref_df, bib)
```

Arguments

ref_df	dataframe with columns author, year, title in case title is missing match is performed based on author and year only.
bib	character string giving the name of the .bib file.

Value

Character vector.

ref_vol *Volume.*

Description

Volume.

Usage

```
ref_vol
```

Format

variable Name of the physical variable: "volume".

polygon Boxid starting from 0 to numboxes - 1.

layer Layerid starting from 0 to numlayers - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. Volume in m^3

ref_vol_dz *Volume and dz.*

Description

Volume and dz.

Usage

```
ref_vol_dz
```

Format

variable Name of the physical variable: "volume" and "dz".

polygon Boxid starting from 0 to numboxes - 1.

layer Layerid starting from 0 to numlayers - 1.

time Simulation time in years. Modeltimestep was converted to actual time based on the settings in the 'run.prm' file.

atoutput Obseravtion column storing the actual output value. volume in m^3 dz in m

scan_prm *Scan character vector for specific flag!*

Description

Scan character vector for specific flag!

Usage

```
scan_prm(chars, variable)
```

Arguments

chars	Vector of character strings
variable	Character string giving the flag to search for.

scan_reference_fishbase *Scan list of references for character string for fish species*

Description

Scan list of references for character string for fish species

Usage

```
scan_reference_fishbase(fish, chr, mirror = "se")
```

Arguments

fish	Vector of fish species with genus and species information.
chr	Character string to search.
mirror	Character string defining the url mirror to use. Defaults to se. In case data extraction is slow use a different mirror. Try to avoid frequently used mirrors like uk or com.

Value

Dataframe of potentially relevant references.

Examples

```
## Not run:
# For some reason the examples break with appveyor.
fish <- c("Gadus morhua", "Merlangius merlangus")
df <- scan_reference_fishbase(fish, chr = "diet")
df <- scan_reference_fishbase(fish, chr = "xxx")
df <- scan_reference_fishbase(fish, chr = "feed")

## End(Not run)
```

sc_init *Sanity check initial conditions file*

Description

Sanity check initial conditions file

Usage

```
sc_init(init, prm_biol, fgs, bboxes, pred = NULL, set_avail = NULL,
  version_flag = 2)

plot_sc_init(df, mult_mum, mult_c, pred = NULL)
```

Arguments

init	Character string giving the connection of the initial conditions netcdf file. The filename usually contains `init` and ends in `.nc`.
prm_biol	Character string giving the connection to the biological parameterfile. The filename usually contains `biol_fishing` and does end in `.prm`.
fgs	Character string giving the connection to the functional groups file. The filename usually contains `Groups` and does end in `.csv`.
bboxes	Integer vector giving the box-id of the boundary boxes. Can be created with `get_boundary`.
pred	Vector of predator acronyms to check. If `NULL` (default) all age based predators are selected.
set_avail	Numeric value. All present availabilities can be set to a specific value. Default value is `NULL` which results in no changes to the present availability matrix.
version_flag	The version of ATLANTIS model. 1 for bec_dev, 2 for trunk. `default is 2.`.
df	Dataframe to pass to `plot_sc_init()`. df should be generated with sc_init or read in from *.rda (also generated with sc_init()).
mult_mum	Numeric vector of multiplication factors applied to the initial mum values.
mult_c	Numeric vector of multiplication factors applied to the initial C values.

Value

Dataframe/ Plot.

Examples

```
d <- system.file("extdata", "setas-model-new-trunk", package = "atlantistools")
init <- file.path(d, "INIT_VMPA_Jan2015.nc")
prm_biol <- file.path(d, "VMPA_setas_biol_fishing_Trunk.prm")
fgs <- file.path(d, "SETasGroupsDem_NoCep.csv")
bboxes <- get_boundary(load_box(bgm = file.path(d, "VMPA_setas.bgm")))

data1 <- sc_init(init, prm_biol, fgs, bboxes)
## Not run:
dir <- system.file("extdata", "gns", package = "atlantistools")
fgs <- "functionalGroups.csv"
init <- "init_simple_NorthSea.nc"
prm_biol <- "NorthSea_biol_fishing.prm"
bboxes <- get_boundary(load_box(dir = dir, bgm = "NorthSea.bgm"))
mult_mum <- seq(0.5, 10, by = 1)
mult_c <- seq(0.5, 10, by = 1)
```

```
no_avail <- FALSE
save_to_disc <- FALSE
data1 <- sc_init(dir, init, prm_biol, fgs, bboxes, save_to_disc = FALSE)
plot_sc_init(df = data1, mult_mum, mult_c)
plot_sc_init(df = data1, mult_mum, mult_c, pred = "Cod")

data2 <- sc_init(dir, init, prm_biol, fgs, bboxes, pred = "Cod", save_to_disc = FALSE)
plot_sc_init(df = data2, mult_mum, mult_c)

## End(Not run)
```

`str_split_twice` *Extract numeric values from string.*

Description

The function splits any character string at each tab and space and returns all (min_only = FALSE) or only the first (min_only = T) numeric value found in the string.

Usage

```
str_split_twice(char, min_only = TRUE)
```

Arguments

`char`	Character string.
`min_only`	Logical specifying if only the first numeric value (`TRUE`) or all numeric values (`FALSE`) should be returned. Default is `TRUE`.

Value

numeric values inside `char` string.

Examples

```
str_split_twice(char = "Hello   15")
str_split_twice(char = "flag1 15  16\t15", min_only = FALSE)
```

theme_atlantis *Customized theme used in all plots.*

Description

This function is a customized theme for ggplot2 plots. It's applied by default to all plots created within atlantistools.

Usage

```
theme_atlantis(large = 22, medium = 18, small = 14, scale_font = 1,
  rot_xaxis_text = FALSE, rot_strips_y = TRUE)
```

Arguments

large	Integer giving the size of the font for the main parts of the plot. Default is 22.
medium	Integer giving the size of the font used in the legend and facet labels. Default is 18.
small	Integer giving the size of the font used in the rest of the plot. Default is 14.
scale_font	Numeric used to scale all font sizes. Default is 1.
rot_xaxis_text	Logical indicating if x-axis text should be rotated by 45 degree. Default is FALSE.
rot_strips_y	Logical indicating if facet labels should be rotated by 90 degree. Default is TRUE.

Examples

```
## Not run: nums_agg <- agg_data(data = ref_nums, groups = c("species", "time"), fun = sum)
ggplot2::ggplot(data = nums_agg, ggplot2::aes(x = time, y = atoutput)) +
   ggplot2::facet_wrap(~species) +
   theme_atlantis()
## End(Not run)
```

%>% *Pipeoperator*

Description

Atlantistools makes heavy use of dplyr data transformations. Therefore it is advisable to import the pipeoperator %>% from magrittr.

Index