Package: biospheremetrics (via r-universe)

May 26, 2026

Type Package

Title Biosphere integrity metrics for LPJmL

Version 1.1.0

Maintainer Fabian Stenzel <stenzel@pik-potsdam.de>

Description Functions to compute Biosphere integrity metrics BioCol and EcoRisk based on output from LPJmL.

License AGPL-3

Encoding UTF-8

LazyData true

URL https://github.com/stenzelf/biospheremetrics

RoxygenNote 7.3.2

Imports lpjmlkit, rlang, tidyr, dplyr, scales, terra, withr, readr, magrittr, yaml, stats, zoo, utils, abind, fields, fBasics, ggplot2, rnaturalearth, tidyterra, RColorBrewer, circlize, graphics, grDevices, maps, methods

Suggests testthat (>= 3.0.0), knitr

VignetteBuilder knitr

Date 2025-05-20

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libicu-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libx11-dev

Repository https://pik-piam.r-universe.dev

Date/Publication 2025-05-20 13:37:03 UTC

RemoteUrl https://github.com/stenzelf/biospheremetrics

RemoteRef HEAD

RemoteSha 012b8cc99c673777e9f0dc2817755b4348c7b193

Contents

`average_nyear_window` *Calculate averages (mean) for defined window sizes*

Description

Define window sizes (nyear_window) to be used to calculate averages (mean) for each window (`dim(x)[3] / nyear_window`). Instead of discrete windows, also moving averages can be computed as well as years inbetween interpolated.

Usage

```
average_nyear_window(
  x,
  nyear_window = NULL,
  moving_average = FALSE,
  interpolate = FALSE,
  nyear_reference = NULL
)
```

Arguments

x	LPJmL output array with dim(x)=c(cell, month, year)
nyear_window	integer, if supplied it defines the years for each window to be averaged over in dim(x)[3]. If nyear_window == 1 values are used directly (instead of calculating an average). nyear_window has to be smaller than dim(x)[3] and dim(x)[3] is ideally a multipe of nyear_window. Defaults to NULL
moving_average	logical. If TRUE moving average is computed. start and end are interpolated using spline interpolation.
interpolate	logical. If TRUE and nyear_window is defined (with moving_average == FALSE years are interpolated (spline) to return array with same dimensions as x (mainlydim(x)[3] -> year).
nyear_reference	integer, if supplied (default NULL), it defines a time_span for ideally reference runs to be used as a baseline. E.g. nyear_reference = 30 to be used for preindustrial climate reference.

Value

array with same amount of cells and months as x. 3rd dimension is defined by nyear_window, basically dim(x)[3]/nyear_window or equal to dim(x)[3] if moving_average == TRUE or interpolate == TRUE

---

calc_biocol *Calculate BioCol*

Description

Wrapper function to calculate BioCol

Usage

```
calc_biocol(
  path_lu,
  path_pnv,
  start_year,
  stop_year,
```

```
  reference_npp_time_span = NULL,
  reference_npp_file = NULL,
  gridbased = TRUE,
  read_saved_data = FALSE,
  save_data = FALSE,
  data_file = NULL,
  include_fire = FALSE,
  external_fire = FALSE,
  external_wood_harvest = FALSE,
  grass_scaling = FALSE,
  npp_threshold = 20,
  grass_harvest_file = NULL,
  external_fire_file = NULL,
  external_wood_harvest_file = NULL,
  replace_input_file_names = NULL,
  suppress_warnings = TRUE
)
```

Arguments

path_lu	folder of landuse scenario run
path_pnv	folder of pnv reference run
start_year	first year of simulations
stop_year	last year of simulations
reference_npp_time_span	time span to read reference npp from, using index years 10:39 from potential npp input if set to NULL (default: NULL)
reference_npp_file	file to read reference npp from, using potential npp input if set to NULL (default: NULL)
gridbased	logical are pft outputs gridbased or pft-based?
read_saved_data	flag whether to read previously saved data instead of reading it in from output files (default FALSE)
save_data	whether to save input data to file (default FALSE)
data_file	file to save/read input data to/from (default NULL)
include_fire	boolean include firec in calculation of BioCol? (default TRUE)
external_fire	instead of reading in firec for fire emissions, read in this external firec file from a separate spitfire run with disabled lighning. this will then include only human induced fires (default FALSE)
external_wood_harvest	include external wood harvest from LUH2_v2h (default FALSE)
grass_scaling	whether to scale pasture harvest according to data given via grass_harvest_file (default FALSE)

`npp_threshold` lower threshold for npp (to mask out non-lu areas according to Haberl et al. 2007). Below BioCol will be set to 0. (default: 20 gC/m2)

`grass_harvest_file`
file containing grazing data to rescale the grassland harvests according to Herrero et al. 2013. File contains: grazing_data list object with $name and $id of 29 world regions, and $Herrero_2000_kgDM_by_region containing for each of these regions and mapping_lpj67420_to_grazing_regions array with a mapping between 67420 LPJmL cells and the 29 regions

`external_fire_file`
path to external file with human induced fire fraction c(cell,month,year) since 1500

`external_wood_harvest_file`
path to R-file containing processed timeline of maps for LUH2_v2h woodharvest

`replace_input_file_names`
list with alternative names for output identifiers to replace the ones in inst/ext_files/metric_files.yml. e.g. list(npp="mnpp") would replace the expected output for npp with mnpp followed by the automatically detected file extension (.bin.json)

`suppress_warnings`
suppress warnings when reading files (default: TRUE)

Value

list data object containing BioCol and components as arrays: biocol, biocol_overtime, biocol_overtime_piref, biocol_frac, npp_potential, biocol_overtime_abs_frac_piref, biocol_frac_piref, npp_act_overtime, npp_pot_overtime, npp_eco_overtime, npp_ref, harvest_cft_overtime, npp_luc_overtime, rharvest_cft_overtime, fire_overtime, timber_harvest_overtime, harvest_cft, rharvest_cft, wood_harvest_overtime, biocol_harvest, biocol_luc, lat, lon, cellarea

Examples

```
## Not run:
calc_biocol(
  path_lu = run_folder,
  path_pnv = pnv_folder,
  gridbased = TRUE,
  start_year = 1980,
  stop_year = 2014,
  reference_npp_time_span = 1510:1539,
  read_saved_data = FALSE,
  save_data = FALSE,
  npp_threshold = 20,
)

## End(Not run)
```

calc_delta_v *Calculates changes in vegetation structure (vegetation_structure_change)*

Description

Utility function to calculate changes in vegetation structure (vegetation_structure_change) for calculation of EcoRisk

Usage

```
calc_delta_v(
  fpc_ref,
  fpc_scen,
  bft_ref,
  bft_scen,
  cft_ref,
  cft_scen,
  weighting = "equal"
)
```

Arguments

fpc_ref	reference fpc array (dim: [ncells,npfts+1])
fpc_scen	scenario fpc array (dim: [ncells,npfts+1])
bft_ref	reference bft array (dim: [ncells,nbfts])
bft_scen	scenario bft array (dim: [ncells,nbfts])
cft_ref	reference cft array (dim: [ncells,ncfts])
cft_scen	scenario cft array (dim: [ncells,ncfts])
weighting	apply "old" (Ostberg-like), "new", or "equal" weighting of vegetation_structure_change weights (default "equal")

Value

vegetation_structure_change array of size ncells with the vegetation_structure_change value [0,1] for each cell

Examples

```
## Not run:
vegetation_structure_change <- calc_delta_v(
  fpc_ref = fpc_ref_mean,
  fpc_scen = apply(fpc_scen, c(1, 2), mean),
  bft_ref = bft_ref_mean,
  bft_scen = apply(bft_scen, c(1, 2), mean),
  cft_ref = cft_ref_mean,
```

```
  cft_scen = apply(cft_scen, c(1, 2), mean),
  weighting = "equal"
)

## End(Not run)
```

`calc_ecorisk` *Calculate the ecosystem change metric EcoRisk between 2 sets of states This function is called by the wrapper function (ecorisk_wrapper), unless you know what you are doing, don't use this function directly.*

Description

Function to calculate the ecosystem change metric EcoRisk, based on gamma/vegetation_structure_change work from Sykes (1999), Heyder (2011), and Ostberg (2015,2018). This is a reformulated version in R, not producing 100 than the C/bash version from Ostberg et al. 2018, but similar the methodology

Usage

```
calc_ecorisk(
  fpc_ref,
  fpc_scen,
  bft_ref,
  bft_scen,
  cft_ref,
  cft_scen,
  state_ref,
  state_scen,
  weighting = "equal",
  lat,
  lon,
  cell_area,
  dimensions_only_local = FALSE,
  nitrogen = TRUE,
  external_variability = FALSE,
  c2vr = NULL
)
```

Arguments

`fpc_ref`	reference run data for fpc
`fpc_scen`	scenario run data for fpc
`bft_ref`	reference run data for fpc_bft
`bft_scen`	scenario run data for fpc_bft
`cft_ref`	reference run data for cftfrac

cft_scen	scenario run data for cftfrac
state_ref	reference run data for state variables
state_scen	scenario run data for state variables
weighting	apply "old" (Ostberg-like), "new", or "equal" weighting of vegetation_structure_change weights (default "equal")
lat	latitude array
lon	longitude array
cell_area	cellarea array
dimensions_only_local	flag whether to use only local change component for water/carbon/nitrogen fluxes and pools, or use an average of local change, global change and ecosystem balance (default FALSE)
nitrogen	include nitrogen outputs (default: TRUE)
external_variability	include external change_to_variability_ratio? (default: FALSE)
c2vr	list with external change_to_variability_ratios for each component (default: NULL)

Value

list data object containing arrays of ecorisk_total, vegetation_structure_change, local_change, global_importance, ecosystem_balance, carbon_stocks, carbon_fluxes, water_fluxes (+ nitrogen_stocks and nitrogen_fluxes)

---

calc_roc_data *Calculate ROC curve data*

Description

Calculate data for ROC curve comparison between binary external and continuous internal indicator.

Usage

```
calc_roc_data(
  external_binary,
  internal_continuous,
  external_name,
  cellArea,
  range_internal,
  sampling_res = 0.01
)
```

Arguments

external_binary	array with binary transgression value: 0 - no, 1 - yes
internal_continuous	array with continuous values
external_name	name of external indicator
cellArea	array with area for each cell of internal indicator
range_internal	range of internal data, e.g. c(0,1)
sampling_res	sampling rate for internal indicator. default: 0.01

---

calculate_within_biome_diffs

Calculate ecorisk with each biomes average cell

Description

Function to calculate ecorisk with each biomes average cell as a measure of internal variability

Usage

```
calculate_within_biome_diffs(
  biome_classes,
  data_file_base,
  intra_biome_distrib_file,
  create = FALSE,
  nitrogen = TRUE,
  res = 0.05,
  plotting = FALSE,
  plot_folder,
  time_span_reference,
  ecorisk_components = 13
)
```

Arguments

biome_classes	biome classes object as returned by classify biomes, calculated for data_file_base
data_file_base	base EcoRisk to compute differences with (only ref is relevant)
intra_biome_distrib_file	file to additionally write results to
create	create new modified files, or read already existing ones?
nitrogen	include nitrogen outputs (default: TRUE)
res	how finegrained the distribution should be (resolution)
plotting	whether plots for each biome should be created

`plot_folder` folder to plot into

`time_span_reference`
suitable 30 year reference period (e.g. c(1901,1930), c(1550,1579))

`ecorisk_components`
integer. how many subcomponents does the ecorisk object have?

Value

data object with distibution - dim: c(biomes,ecorisk_variables,bins)

`classify_biomes` *Classify biomes*

Description

Classify biomes based on foliage protected cover (FPC) and temperature LPJmL output plus either vegetation carbon or pft_lai depending on the savanna_proxy option and elevation if montane_arctic_proxy requires this

Usage

```
classify_biomes(
  path_reference = NULL,
  files_reference = NULL,
  time_span_reference,
  savanna_proxy = list(pft_lai = 6),
  montane_arctic_proxy = list(elevation = 1000),
  tree_cover_thresholds = list(),
  avg_nyear_args = list(),
  input_files = list(),
  diff_output_files = list()
)
```

Arguments

`path_reference` path to the reference LPJmL run. If not provided, the path is extracted from the file paths provided in files_reference.

`files_reference`
list with variable names and corresponding file paths (character string) of the reference LPJmL run. All needed files are provided as key value pairs, e.g.: list(leaching = "/temp/leaching.bin.json"). If not needed for the applied method, set to NULL.

`time_span_reference`
time span to be used for the scenario run, defined as an character string, e.g. ‘as.character(1901:1930)‘.

`savanna_proxy` 'list' with either pft_lai or vegc as key and value in m2/m2 for pft_lai (default = 6) and gC/m2 for vegc (default would be 7500), Set to 'NULL' if no proxy should be used.

`montane_arctic_proxy`
'list' with either "elevation" or "latitude" as name/key and value in m for elevation (default 1000) and degree for latitude (default would be 55), Set to 'NULL' if no proxy is used.

`tree_cover_thresholds`
list with minimum tree cover thresholds for definition of forest, woodland, savanna and grassland. Only changes to the default have to be included in the list, for the rest the default is used. Default values, based on the IGBP land cover classification system: "boreal forest" = 0.6 "temperate forest" = 0.6 "temperate woodland" = 0.3 "temperate savanna" = 0.1 "tropical forest" = 0.6 "tropical woodland" = 0.3 "tropical savanna" = 0.1 In the boreal zone, there is no woodland, everything below the boreal forest threshold will be classified as boreal tundra.

`avg_nyear_args` list of arguments to be passed to average_nyear_window (see for more info). To be used for time series analysis

`input_files` 'list' with input file paths to be used instead of outputs default: empty list

`diff_output_files`
'list' with output files, to be used from another location than the output folders - default: empty list

Value

list object containing biome_id (main biome per grid cell [dim=c(ncells)]), and list of respective biome_names[dim=c(nbiomes)]

Examples

```
## Not run:
classify_biomes(
  path_data = "/path/to/lpjml_run/to/classify",
  timespan = c(1982:2011)
)

## End(Not run)
```

`disaggregate_into_biomes`

Averages EcoRisk values across regions

Description

Returns the average value across either 4 regions or all (19) biomes for EcoRisk and each of the subcomponents for each

Usage

```
disaggregate_into_biomes(
  data,
  biome_class,
  type = "quantile",
  classes = "4biomes"
)
```

Arguments

data	List object, of which every item should be disaggregated
biome_class	biome class list object as returned by classify_biomes
type	string controlling whether to return minimum, mean, maximum ("minmeanmax") or Q10,Q50,Q90 ("quantile") - default: "quantile"
classes	string for into how many regions should be disaggregated "4biomes" (tropics/temperate/boreal/arctic) or "allbiomes"

Examples

```
## Not run:
disaggregate_into_biomes(
  ecorisk = ecorisk,
  biome_class = biome_classes,
  type = "quantile", classes = "4biomes"
)

## End(Not run)
```

---

ecorisk_combine_hist_and_scen_data

Create modified EcoRisk data combining two time series

Description

Function to combine two EcoRisk data files (e.g. historic and futures) into one file.

Usage

```
ecorisk_combine_hist_and_scen_data(hist_file, scen_file, combined_file)
```

Arguments

hist_file	path to input file with historic data
scen_file	path to input file with scenario data
combined_file	path to save modified data to

ecorisk_cross_table *Create modified EcoRisk data for crosstable*

Description

Function to create a modified EcoRisk data file where for each biome the average scenario cell is compared to the average scenario cell of all other biomes. This can then be used to compute a crosstable with the average difference between each of them as in the SI of Ostberg et al. 2013 (Critical impacts of global warming on land ecosystems)

Usage

```
ecorisk_cross_table(
  data_file_in,
  data_file_out,
  biome_classes_in,
  pick_cells = NULL,
  baseline_ref = FALSE
)
```

Arguments

data_file_in	path to input data
data_file_out	path to save modified data to
biome_classes_in	biome classes object as returned from classify_biomes
pick_cells	pick one specific cell as representative for the biome instead of computing the average state
baseline_ref	logical, use reference state as baseline? default is FALSE - use scenario state

Value

c2vr array to be used in the ecorisk call

ecorisk_wrapper *Wrapper for calculating the ecosystem change metric EcoRisk*

Description

Function to read in data for ecorisk, and call the calculation function once, if overtime is FALSE, or for each timeslice of length window years, if overtime is TRUE

Usage

```
ecorisk_wrapper(
  path_ref,
  path_scen,
  read_saved_data = FALSE,
  save_data = NULL,
  save_ecorisk = NULL,
  nitrogen = TRUE,
  weighting = "equal",
  time_span_reference,
  time_span_scenario,
  dimensions_only_local = FALSE,
  overtime = FALSE,
  window = 30,
  debug_mode = FALSE,
  replace_input_file_names = NULL,
  external_variability = FALSE,
  c2vr = NULL,
  suppress_warnings = TRUE
)
```

Arguments

path_ref	folder of reference run
path_scen	folder of scenario run
read_saved_data	whether to read in previously saved data (default: FALSE)
save_data	file to save read in data to (default NULL)
save_ecorisk	file to save EcoRisk data to (default NULL)
nitrogen	include nitrogen outputs for pools and fluxes into EcoRisk calculation (default FALSE)
weighting	apply "old" (Ostberg-like), "new", or "equal" weighting of vegetation_structure_change weights (default "equal")
time_span_reference	vector of years to use as scenario period
time_span_scenario	vector of years to use as scenario period
dimensions_only_local	flag whether to use only local change component for water/carbon/nitrogen fluxes and pools, or use an average of local change, global change and ecosystem balance (default FALSE)
overtime	logical: calculate ecorisk as time-series? (default: FALSE)
window	integer, number of years for window length (default: 30)
debug_mode	write out all nitrogen state variables (default FALSE)

replace_input_file_names
: list with alternative names for output identifiers to replace the ones in inst/ext_files/metric_files.yml. e.g. list(npp="mnpp") would replace the expected output for npp with mnpp followed by the automatically detected file extension (.bin.json)

external_variability
: use externally supplied variability for the reference period? experimental! (default: FALSE)

c2vr
: external variability array

suppress_warnings
: suppress warnings - default: TRUE

Value

list data object containing arrays of ecorisk_total, vegetation_structure_change, local_change, global_importance, ecosystem_balance, carbon_stocks, carbon_fluxes, water_fluxes (+ nitrogen_stocks and nitrogen_fluxes)

Examples

```
## Not run:
ecorisk_wrapper(
  path_ref = pnv_folder,
  path_scen = run_folder,
  read_saved_data = FALSE,
  nitrogen = TRUE,
  save_data = NULL,
  save_ecorisk = NULL,
  time_span_reference = c(1550:1579),
  time_span_scenario = c(1987:2016)
)

## End(Not run)
```

---

get_biome_names *Get biome names*

Description

Returns biome names with variable length (abbreviated, short, or full)

Usage

```
get_biome_names(biome_name_length = 2)
```

Arguments

biome_name_length
: integer chose from 1,2,3 for abbreviated, short, or full biome names

list_outputs *List required output files for given metric*

Description

List required output files for given metric based on parameter file `inst/extfiles/metric_files.yml`

Usage

```
list_outputs(metric = "all", only_first_filename = TRUE)
```

Arguments

`metric`	character string/list of strings. metrics to list outputs for can be one of: "all" - list all outputs for all metrics "ecorisk" - list outputs for ecorisk metric without nitrogen "ecorisk_nitrogen" - list outputs for ecorisk metric with nitrogen "biocol" - list outputs for biocol metric "biome" - list outputs for the biome classification
`only_first_filename`	if several legal output names are listed, only output the first of them (default: TRUE)

Value

list object with required outputs, their required temporal resolution and if it is optional

Examples

```
## Not run:
list_outputs(metric = "ecorisk_nitrogen")

## End(Not run)
```

plot_biocol *Plot absolute BioCol, overtime, maps, and npp into given folder*

Description

Wrapper function to plot absolute biocol, overtime, maps, and npp into given folder

Usage

```
plot_biocol(
  biocol_data,
  path_write,
  plotyears,
  min_val,
  max_val,
  legendpos,
  details = FALSE,
  start_year,
  mapyear,
  mapyear_buffer = 5,
  highlightyear,
  eps = FALSE
)
```

Arguments

biocol_data	biocol data list object (returned from calc_biocol) containing biocol_overtime, biocol_overtime_abs, biocol_overtime_abs_frac_piref, biocol_overtime_frac_piref, biocol_overtime_frac, biocol_overtime_abs_frac, npp_harv_overtime, npp_luc_overtime, npp_act_overtime, npp_pot_overtime, npp_eco_overtime, harvest_grasslands_overtime, harvest_bioenergy_overtime, harvest_cft_overtime, rharvest_cft_overtime, fire_overtime, timber_harvest_overtime, wood_harvest_overtime, biocol, biocol_frac, npp, biocol_frac_piref, npp_potential, npp_ref, harvest_cft, rharvest_cft, biocol_harvest, biocol_luc, lat, lon, cellarea
path_write	folder to write outputs into
plotyears	range of years to plot over time
min_val	y-axis minimum value for plot over time
max_val	y-axis maximum value for plot over time
legendpos	position of legend
details	show all harvest components or not
start_year	first year of biocol_data object
mapyear	year to plot biocol map for
mapyear_buffer	+- years around mapyear to average biocol (make sure these years exist in biocol_data) - default: 5
highlightyear	year(s) that should be highlighted in overtime plot
eps	write plots as eps, instead of png (default = FALSE)

Examples

```
## Not run:
plot_biocol(
  biocol_data = biocol,
  path_write = "~/BioCol_plots/",
```

```
  plotyears = c(1980, 2014),
  min_val = 0,
  max_val = 90,
  legendpos = "left",
  start_year = 1980,
  mapyear = 2000,
  highlightyear = 2000,
  eps = FALSE
)

## End(Not run)
```

---

plot_biocol_map *Plot global map of BioCol to file*

Description

Plot global map of BioCol to file with legend colors similar to Haberl et al. 2007

Usage

```
plot_biocol_map(
  data,
  lon,
  lat,
  file = NULL,
  title = "",
  legendtitle = "",
  zero_threshold = 0.001,
  haberl_legend = FALSE,
  eps = FALSE
)
```

Arguments

data	array containing BioCol percentage value for each gridcell
lon	longitude array
lat	latitude array
file	to write into, if not supplied (default is NULL) write to screen
title	character string title for plot (default: "")
legendtitle	character string legend title (default: "")
zero_threshold	smallest value to be distinguished from 0 in legend, both for negative and positive values (default: 0.001)
haberl_legend	use color palette from Haberl et al.? (default: FALSE)
eps	write eps file instead of PNG (boolean) - (default: FALSE)

Examples

```
## Not run:
plot_biocol_map(
  data = biocol$biocol_frac[, "2000"] * 100,
  file = "./BioCol_map_yr2000.png",
)

## End(Not run)
```

`plot_biocol_ts` *Plot absolute BioCol, overtime, maps, and npp into given folder*

Description

Plot to file a comparison over time of global sums of BioCol, NPPpot, NPPeco, and NPPact, with legend similar to Krausmann et al. 2013

Usage

```
plot_biocol_ts(
  biocol_data,
  file = NULL,
  first_year,
  plot_years,
  highlight_years = 2000,
  details = FALSE,
  min_val = 0,
  max_val = 100,
  max_val_right = 0.45,
  legendpos = "topleft",
  eps = FALSE,
  ref = "pi"
)
```

Arguments

`biocol_data`	biocol data list object (returned from calc_biocol) containing biocol, npp_eco_overtime, npp_act_overtime, npp_pot_overtime, npp_bioenergy_overtime, biocol_overtime, npp_harv_overtime, biocol_overtime_perc_piref, biocol_perc, biocol_perc_piref, npp all in GtC
`file`	character string for location/file to save plot to
`first_year`	first year of biocol object
`plot_years`	range of years to plot over time
`highlight_years`	year(s) that should be highlighted in overtime plot (default: 2000)

details	show all harvest components or not, (default: FALSE)
min_val	y-axis minimum value for plot over time (default: 0)
max_val	y-axis maximum value for plot over time (default: 100)
max_val_right	maximum value for the BioCol y-axis labs right (default: 0.45)
legendpos	position of legend (default: "topleft")
eps	write plots as eps, instead of png (default = FALSE)
ref	reference period for biocol ("pi" or "act"), to either use biocol_data$biocol_overtime_perc_piref or biocol_data$biocol_overtime

Examples

```
## Not run:
plot_biocol_ts(
  biocol_data = biocol_data,
  file = "./BioCol_overtime_LPJmL_1550-2015.png",
  first_year = 1550,
  plot_years = c(1550, 2015),
  min_val = 0,
  max_val = 80,
  ref = "pi",
  legendpos = "topleft",
  details = TRUE,
  max_val = max_val,
  highlight_years = c(1900, 2000)
)

## End(Not run)
```

---

plot_biome_averages *Plot radial EcoRisk with 4/16 biomes*

Description

Function to plot to file (or screen) an aggregated radial status of EcoRisk values [0-1] for the different sub-categories to file

Usage

```
plot_biome_averages(
  data,
  file = NULL,
  biome_class_names,
  title = "",
  title_size = 2,
  leg_scale = 1,
  eps = FALSE,
  palette = NULL
)
```

Arguments

data	EcoRisk data array c(4[biomes],[nEcoRiskcomponents], 3[min,median,max])
file	to write into (if not supplied - default NULL - prints to screen)
biome_class_names	to write into
title	character string title for plot, default empty
title_size	character string title for plot
leg_scale	character string title for plot
eps	write as eps, replacing png in filename (default: True)
palette	color palette to plot EcoRisk with, defaults to the Ostberg color scheme white-blue-yellow-red

Examples

```
## Not run:
plot_biome_internal_distribution(
  data = biomes,
  file = "./biomes.png"
)

## End(Not run)
```

---

plot_biome_internal_distribution

Plot distribution of similarity within biomes

Description

Function to plot the distribution of similarity within biomes

Usage

```
plot_biome_internal_distribution(
  data,
  file = NULL,
  biomes_abbrv = NULL,
  scale = 1,
  title = "",
  legendtitle = "",
  eps = FALSE,
  palette = NULL
)
```

Arguments

data	data object with distibution - as returned by calculateWithInBiomeDiffs for each subcategory of ecorisk. dim: c(biomes,bins)
file	to write into, if not supplied (default is NULL) write to screen
biomes_abbrv	character vector. abbreviated names of biomes (defaults to NULL -> extract dimension names from data)
scale	scaling factor for distribution. defaults to 1
title	character string title for plot, default empty
legendtitle	character string legend title, default empty
eps	write as eps or png (default: FALSE -> png)
palette	color palette to plot EcoRisk with, defaults to the Ostberg color scheme white-blue-yellow-red

Examples

```
## Not run:
plot_biome_internal_distribution(
  data = biomes,
  file = "./biomes.png"
)

## End(Not run)
```

---

plot_biomes *Plot global distribution of lpjml simulated biomes*

Description

Plots a map with the biome distribution as derived from a lpjml run based on the "classify_biomes" function

Usage

```
plot_biomes(
  biome_data,
  file_name = NULL,
  display_area = FALSE,
  cellarea = NULL,
 order_legend = c(1, 2, 9, 10, 11, 3, 4, 5, 12, 13, 14, 6, 7, 8, 15, 16, 17, 18, 19),
  projection = "+proj=robin",
  grid_path = NULL
)
```

Arguments

biome_data	output (list) from classify_biomes()
file_name	directory for saving the plot (character string)
display_area	boolean, adding occupied area per biome (default F)
cellarea	array with cellarea in m2 per gridcell (default NULL)
order_legend	in which order the biomes should be displayed default: c(1,2,9,10,11,3,4,5,12,13,14,6,7,8,15,16,17,18,19)
projection	character string defining the projection, default: "+proj=robin", for mercator: "+proj=wgs84"
grid_path	character string providing the path to a grid file

Examples

```
## Not run:

biomes <- classify_biomes(
 config_reference = path_reference,
 time_span_reference = as.character(2008:2017),
 savanna_proxy = list(vegc = 7500)
)

plot_biomes(
  biome_data = biomes,
  file_name = "/p/projects/open/Johanna/R/biomes.pfd"
  grid_path = ".grid.bin.json"
)

## End(Not run)
```

---

plot_ecorisk_cross_table

Plot crosstable showing (dis-)similarity between average biome pixels

---

Description

Function to plot to file (or screen) a crosstable showing (dis-)similarity between average biome pixels based on EcoRisk (former Gamma) metric from LPJmL simulations

Usage

```
plot_ecorisk_cross_table(
  data,
  file = NULL,
  lmar = 3,
  eps = FALSE,
  palette = NULL
)
```

Arguments

data	crosstable data as array with [nbiomes,nbiomes] and row/colnames
file	to write into (if not supplied - default NULL - prints to screen)
lmar	left margin for plot in lines (default: 3)
eps	write as eps or png
palette	color palette to plot EcoRisk with, defaults to the Ostberg color scheme white-blue-yellow-red

Examples

```
## Not run:
plot_ecorisk_cross_table(
  data = crosstable,
  file = "./ecorisk_crosstable.png"
)

## End(Not run)
```

---

plot_ecorisk_map *Plot EcoRisk map to file*

Description

Function to plot a global map of EcoRisk values [0-1] per grid cell to file

Usage

```
plot_ecorisk_map(
  ecorisk_object,
  plot_dimension,
  year = 1,
  file = NULL,
  focus_biome = NULL,
  biome_classes = NULL,
  title = "",
  legendtitle = "",
  eps = FALSE,
  title_size = 1,
  leg_yes = TRUE,
  palette = NULL
)
```

Arguments

ecorisk_object	ecorisk object from which to plot
plot_dimension	which dimension from ecorisk object to plot
year	which year to plot (index or year-string), default: 1
file	to write into, if not supplied (default is NULL) write to screen
focus_biome	highlight the biome with this id and desaturate all other (default NULL – no highlight)
biome_classes	to mask the focus_biome from
title	character string title for plot, default empty
legendtitle	character string legend title
eps	write as eps or png
title_size	size of the title (default: 1)
leg_yes	logical. whether to plot legend or not. defaults to TRUE
palette	color palette to plot EcoRisk with, defaults to the Ostberg color scheme white-blue-yellow-red

Examples

```
## Not run:
plot_biome_internal_distribution(
  data = biomes,
  file = "./biomes.png"
)

## End(Not run)
```

---

plot_ecorisk_maps *Plot EcoRisk maps*

Description

Function to plot with one command maps of all components of EcoRisk to files

Usage

```
plot_ecorisk_maps(ecorisk, out_folder, year = 1)
```

Arguments

ecorisk	EcoRisk object e.g. returned from calc_ecorisk
out_folder	folder to plot the data into
year	which year to plot, supply either as index, or character string of year (default = 1)

Examples

```
## Not run:
plot_ecorisk_maps(
  ecorisk = ecorisk,
  out_folder = "./plots/ecorisk/"
)

## End(Not run)
```

`plot_ecorisk_over_time_panel` *Plot timeline of EcoRisk variables as panel to file with 4/16 biomes*

Description

Function to plot a panel of 4/16 timelines per biome aggregated EcoRisk values `[0-1]` to file

Usage

```
plot_ecorisk_over_time_panel(
  data,
  biome_names,
  file = NULL,
  yrange = c(0, 1),
  timerange,
  eps = FALSE,
  varnames = NULL
)
```

Arguments

`data`	EcoRisk data array c(4/19`[biomes]`,`[nEcoRiskcomponents]`, 3`[min,mean,max]`)
`biome_names`	names of biomes
`file`	to write into (if not supplied - default NULL - prints to screen)
`yrange`	range for y axis (default c(0,1))
`timerange`	of the data input
`eps`	write as eps or png
`varnames`	list vector with variable names

Examples

```
## Not run:
plot_biome_internal_distribution(
  data = biomes,
  file = "./biomes.png"
)

## End(Not run)
```

`plot_ecorisk_radial` *Plot radial EcoRisk plot to file*

Description

Function to plot an aggregated radial status of EcoRisk values `[0-1]` for the different sub-categories to file

Usage

```
plot_ecorisk_radial(
  data,
  file,
  title = "",
  leg_yes = TRUE,
  eps = FALSE,
  use_quantile = TRUE
)
```

Arguments

`data`	EcoRisk data array c(4/19`[biomes]`,`[nEcoRiskcomponents]`, 3`[min,mean,max]`)
`file`	to write into
`title`	character string title for plot, default empty
`leg_yes`	logical. whether to plot legend or not. defaults to TRUE
`eps`	write as eps or png
`use_quantile`	show quantiles or min,mean,max

Examples

```
## Not run:
plot_biome_internal_distribution(
  data = biomes,
  file = "./biomes.png"
)

## End(Not run)
```

plot_ecorisk_radial_panel *Plot radial EcoRisk panel to file with 4/16 biomes*

Description

Function to plot an aggregated radial status of EcoRisk values [0-1] for the different sub-categories to file

Usage

```
plot_ecorisk_radial_panel(
  data,
  biome_names,
  file = NULL,
  use_quantile = TRUE,
  eps = FALSE
)
```

Arguments

data	EcoRisk data array c(4/19[biomes],[nEcoRiskcomponents], 3[min,mean,max])
biome_names	names of biomes
file	to write into (if not supplied - default NULL - prints to screen)
use_quantile	is it quantiles or minmeanmax data? - text for whiskers
eps	write as eps or png

Examples

```
## Not run:
plot_ecorisk_radial_panel(
  data = ecorisk_disaggregated_full[-c(3, 18, 19), c(1:5, 8, 9, 13), ],
  biome_names = biomes,
  file = "./biomes.png"
)

## End(Not run)
```

`plot_ecorisk_radial_to_screen` *Plot radial EcoRisk plot to screen*

Description

Function to plot an aggregated radial status of EcoRisk values `[0-1]` for the different sub-categories to screen

Usage

```
plot_ecorisk_radial_to_screen(
  data,
  title = "",
  zoom = 1,
  type = "regular",
  title_size = 2,
  titleline = -2,
  use_quantile = TRUE
)
```

Arguments

`data`	EcoRisk data array c(`[nEcoRiskcomponents]`, 3`[min,mean,max]`)
`title`	character string title for plot, default empty
`zoom`	scaling factor for circle plot. defaults to 1
`type`	plot type, 'legend1' for variable and color legend, 'legend2' for value legend, or 'regular' (default setting) for the regular EcoRisk plot
`title_size`	scaling factor for tile. defaults to 1
`titleline`	line at which the title will be displayed. defaults to -2
`use_quantile`	use quantiles or min,mean,max. defaults to TRUE

Examples

```
## Not run:
plot_biome_internal_distribution(
  data = biomes,
  file = "./biomes.png"
)

## End(Not run)
```

plot_global *Plot global LPJmL array*

Description

Plot global LPJmL array to file (if file argument is given) or screen (else). type argument controls plot type: (exponential, linear, or manual legend). Depending on this more parameters are required. Data plot ranges: exp: c(-2^pow2max,-2^-pow2min,0,2^-pow2min,2^pow2max) lin: c(min,max) man: brks colors for pos and neg values can be given, default is Blues for the positive and Reds for the negative numbers 0-range (from 2^-pow2min to 2^pow2min) is white. The negatives can be omitted by setting only_pos=TRUE, in case there are only pos values.

Usage

```
plot_global(
  data,
  file = NULL,
  title = "",
  pow2min = NULL,
  pow2max = NULL,
  min = NULL,
  max = NULL,
  brks = NULL,
  palette = NULL,
  col_pos = "GnBu",
  type = "exp",
  col_neg = "YlOrRd",
  lat,
  lon,
  legendtitle = "",
  leg_yes = TRUE,
  only_pos = FALSE,
  n_legend_ticks = 20,
  min_0 = 0.01,
  extent = c(-180, 180, -60, 90),
  country_borders = TRUE,
  eps = FALSE,
  cex = 1
)
```

Arguments

data	array with data to plot in LPJmL specific array c(67420)
file	character string for location/file to save plot to, if not supplied, the plot is displayed to screen (default: NULL)
title	character string title for plot

pow2min	for exponential legend: smallest positive number to be distinguished from 0 (2^-pow2min)
pow2max	for exponential legend: upper (positive) end of data range to plot (2^pow2max)
min	for linear legend: lower end of data range to plot (0 is placed symmetrically between min and max, if onlypos = FALSE)
max	for linear legend: upper end of data range to plot (0 is placed symmetrically between min and max, if onlypos = FALSE)
brks	breaks for manual plotting type (type=man) default: NULL
palette	palette for manual plotting type (type=man) default: NULL
col_pos	color palette for the positives
type	string indicating whether to plot manual (man), exponential (exp) or linear (lin) legend (default: exp). man requires: parameters brks and palette defined, exp requires: parameters pow2min and pow2max given defined, lin requires: parameters min and max defined
col_neg	color palette for the negatives
lat	latitude array
lon	longitude array
legendtitle	character string legend title
leg_yes	boolean whether to show legend (default: TRUE)
only_pos	boolean to show only positive half of legend (default: FALSE)
n_legend_ticks	(default: 20)
min_0	(default: 0.01)
extent	extent for plot, default: c(-180, 180, -60, 90)
country_borders	boolean, whether to plot borders (default: TRUE)
eps	boolean whether to write eps file instead of PNG (default: FALSE)
cex	text scaling factor (default: 1)

Examples

```
## Not run:
plot_global(
  data = biocol_data$biocol[, "2015"],
  file = "BioCol_absolute.png",
  type = "exp",
  pow2min = 0,
  pow2max = 12,
  legendtitle = "GtC",
  leg_yes = TRUE,
  only_pos = FALSE,
  eps = FALSE
)

## End(Not run)
```

plot_overtime_to_screen *Plot timeline of EcoRisk variables to screen*

Description

Function to plot timeline of EcoRisk variables to screen

Usage

```
plot_overtime_to_screen(
  data,
  timerange,
  yrange = c(0, 1),
  leg_yes = TRUE,
  leg_only = FALSE,
  varnames = NULL
)
```

Arguments

data	EcoRisk data array c(4/19[biomes],8/10[nEcoRiskcomponents],3[min,mean,max],timeslices)
timerange	of the data input
yrange	range for y axis default c(0,1)
leg_yes	plot legend (default TRUE)
leg_only	plot only the legend? default: FALSE
varnames	manual override of ecorisk subvariable names - default: NULL

Examples

```
## Not run:
plot_biome_internal_distribution(
  data = biomes,
  file = "./biomes.png"
)

## End(Not run)
```

read_calc_biocol *Calculate BioCol based on file lists from a PNV run and LU run of LPJmL. Do not use this function directly, unless you are instructed to do so, there is a wrapper called calc_biocol() which is for use of endusers.*

Description

Function to calculate BioCol based on a PNV run and LU run of LPJmL

Usage

```
read_calc_biocol(
  files_scenario,
  files_baseline,
  files_reference = NULL,
  time_span_scenario,
  time_span_baseline = NULL,
  time_span_reference = NULL,
  gridbased = TRUE,
  read_saved_data = FALSE,
  save_data = FALSE,
  data_file = NULL,
  include_fire = FALSE,
  external_fire = FALSE,
  external_wood_harvest = FALSE,
  grass_scaling = FALSE,
  npp_threshold = 20,
  epsilon = 0.001,
  grass_harvest_file = NULL,
  external_fire_file = NULL,
  external_wood_harvest_file = NULL,
  suppress_warnings = TRUE
)
```

Arguments

files_scenario	list with variable names and corresponding file paths (character string) of the scenario LPJmL run. All needed files are provided in XXX. E.g.: list(npp = "/temp/npp.bin.json")
files_baseline	list with variable names and corresponding file paths (character string) of the baseline LPJmL run. All needed files are provided in XXX. E.g.: list(npp = "/temp/npp.bin.json"). If not needed for the applied method, set to NULL.
files_reference	list with npp file path (character string) of the reference LPJmL run (usually Holocene/preindustrial). E.g.: list(npp = "/temp/npp.bin.json"). If NULL uses baseline npp.

time_span_scenario
: time span to be used for the scenario run, defined as a character vector, e.g. ‘as.character(1982:2011)‘ (required)

time_span_baseline
: time span to be used for the baseline run, defined as a character vector, e.g. ‘as.character(1901:1930)‘. Can differ in offset and length from ‘time_span_scenario‘! If ‘NULL‘ value of ‘time_span_scenario‘ is used

time_span_reference
: time span to read reference npp from, using index years 10:39 from potential npp input if set to NULL (default: NULL)

gridbased
: logical are pft outputs gridbased or pft-based?

read_saved_data
: flag whether to read previously saved data instead of reading it in from output files (default FALSE)

save_data
: whether to save input data to file (default FALSE)

data_file
: file to save/read input data to/from (default NULL)

include_fire
: boolean include firec in calculation of BioCol? (default TRUE)

external_fire
: instead of reading in firec for fire emissions, read in this external firec file from a separate spitfire run with disabled lighning. this will then include only human induced fires (default FALSE)

external_wood_harvest
: include external wood harvest from LUH2_v2h (default FALSE)

grass_scaling
: whether to scale pasture harvest according to data given via grass_harvest_file (default FALSE)

npp_threshold
: lower threshold for npp (to mask out non-lu areas according to Haberl et al. 2007). Below BioCol will be set to 0. (default: 20 gC/m2)

epsilon
: minimum value for npp, below which it will be set to 0

grass_harvest_file
: file containing grazing data to rescale the grassland harvests according to Herrero et al. 2013. File contains: grazing_data list object with $name and $id of 29 world regions, and $Herrero_2000_kgDM_by_region containing for each of these regions and mapping_lpj67420_to_grazing_regions array with a mapping between 67420 LPJmL cells and the 29 regions

external_fire_file
: path to external file with human induced fire fraction c(cell,month,year) since 1500

external_wood_harvest_file
: path to R-file containing processed timeline of maps for LUH2_v2h woodharvest

suppress_warnings
: suppress warnings when reading files (default: TRUE)

Value

list data object containing BioCol and components as arrays: biocol_overtime, biocol_overtime_abs, biocol_overtime_abs_frac_piref, biocol_overtime_abs_frac, biocol_overtime_pos, biocol_overtime_pos_frac_piref,biocol_o biocol_overtime_frac_piref, biocol_overtime_frac, npp_harv_overtime, npp_luc_overtime,npp_act_overtime, npp_pot_overtime,npp_eco_overtime, harvest_grasslands_overtime, harvest_bioenergy_overtime, harvest_cft_overtime, rharvest_cft_overtime, fire_overtime, timber_harvest_overtime, wood_harvest_overtime, biocol, biocol_frac, npp, biocol_frac_piref, npp_potential, npp_ref, harvest_cft, rharvest_cft, biocol_harvest, biocol_luc, lat, lon, cellarea

---

`read_ecorisk_data` *Read in output data from LPJmL to calculate the ecosystem change metric EcoRisk. This function is called by the wrapper function (ecorisk_wrapper), unless you know what you are doing, don't use this function directly.*

Description

Utility function to read in output data from LPJmL for calculation of EcoRisk

Usage

```
read_ecorisk_data(
  files_reference,
  files_scenario,
  save_file = NULL,
  time_span_reference,
  time_span_scenario,
  nitrogen,
  debug_mode = FALSE,
  suppress_warnings = TRUE
)
```

Arguments

`files_reference`	folder of reference run
`files_scenario`	folder of scenario run
`save_file`	file to save read in data to (default NULL)
`time_span_reference`	vector of years to use as scenario period
`time_span_scenario`	vector of years to use as scenario period
`nitrogen`	include nitrogen outputs for pools and fluxes into EcoRisk calculation (default FALSE)
`debug_mode`	write out all nitrogen state variables (default FALSE)
`suppress_warnings`	suppress writing of Warnings, default: TRUE

Value

list data object containing arrays of state_ref, mean_state_ref, state_scen, mean_state_scen, fpc_ref, fpc_scen, bft_ref, bft_scen, cft_ref, cft_scen, lat, lon, cell_area

---

`replace_ref_data_with_average_ref_biome_cell` *Create modified EcoRisk data file*

Description

Function to create a modified EcoRisk data file where each reference cell is compared to the average reference biome cell. The scenario period is overwritten with the original reference period and all reference cells are set to the average cell of the prescribed reference biome ref_biom

Usage

```
replace_ref_data_with_average_ref_biome_cell(
  data_file_in,
  data_file_out,
  biome_classes_in,
  ref_biom
)
```

Arguments

`data_file_in`	path to input data
`data_file_out`	path to save modified data to
`biome_classes_in`	biome classes object as returned from classify_biomes
`ref_biom`	reference biome from biome classes that all cells should be compared to

---

`roc_plot` *Plot ROC curve*

Description

Plot ROC curve(s)

Usage

```
roc_plot(filename = NULL, values)
```

Arguments

`filename`	path to plot roc curve to. default: NULL -> plot to screen
`values`	roc data object as obtained from calc_roc_data(), can be an array with dimensions = c(thresholds, internal_metric(s), external_indicator(s), 3`["TP","FP","slope"]`)

roc_plot_paper *Plot ROC curve*

Description

Plot ROC curve(s) - paper version

Usage

```
roc_plot_paper(filename = NULL, values)
```

Arguments

filename	path to plot roc curve to. default: NULL -> plot to screen
values	roc data object as obtained from calc_roc_data(), can be an array with dimensions = c(thresholds, internal_metric(s), external_indicator(s), 3["TP","FP","slope"])

state_diff_global *c based on Heyder 2011 eq. 10-13*

Description

c based on Heyder 2011 eq. 10-13

Usage

```
state_diff_global(ref, scen, cell_area, epsilon = 10^-4)
```

Arguments

ref	mean reference state vector of dimension c(ncells,variables)
scen	mean scenario state vector of dimension c(ncells,variables)
cell_area	area of each cell as a vector of dim=c(ncells)
epsilon	threshold for variables to be treated as 0

Value

the length of the difference vector for each cell

`state_diff_local` *based on Heyder 2011 eq. 6-9; epsilon case handling from code by Sebastian Ostberg (not documented in papers)*

Description

based on Heyder 2011 eq. 6-9; epsilon case handling from code by Sebastian Ostberg (not documented in papers)

Usage

```
state_diff_local(ref, scen, epsilon = 10^-4)
```

Arguments

`ref`	mean reference state vector of dimension c(ncells,variables)
`scen`	mean scenario state vector of dimension c(ncells,variables)
`epsilon`	threshold for variables to be treated as 0

Value

the length of the difference vector for each cell

Index