# Package: boundaries (via r-universe)

November 21, 2024

**Title** Planetary Boundary Status based on LPJmL simulations

**Version** 1.0.4

**Description** A systematic approach to quantify the status of the terrestrial planetary boundaries based on the Dynamic Global Vegetation Model (DGVM) Lund-Potsdam-Jena managed Land (LPJmL) hosted at the Potsdam Institute for Climate Impact Research (PIK). The supported planetary boundaries are ``biosphere integrity'', ``land-system change'', ``bluewater'', ``greenwater'' and ``nitrogen flows''.

**License** AGPL-3

**LazyData** true

**RoxygenNote** 7.3.2

**Roxygen** list(markdown = TRUE)

**Encoding** UTF-8

**Depends** R (>= 3.5.0)

**URL** https://github.com/PIK-tess/boundaries, https://doi.org/10.5281/zenodo.11550559

**BugReports** https://github.com/PIK-tessLPJmL/boundaries/issues

**Imports** magrittr, tibble, dplyr, future, sf, terra, reshape2, lpjmlkit, readr, tidyr, rlang, yaml, abind, methods, tidyselect, matrixStats, biospheremetrics

**Suggests** testthat (>= 3.0.0), cowplot, ggh4x, ggnewscale, ggplot2, ggspatial, ggpattern, ggtrace, gridExtra, purrr, rnaturalearth, ggpubr, ggrepel, scales, tidyterra

**Remotes** github::stenzelf/biospheremetrics, github::yjunechoe/ggtrace

**Config/testthat/edition** 3

**Date** 2024-10-28

**Config/pak/sysreqs** 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

**RemoteUrl** https://github.com/PIK-tess/boundaries

**RemoteRef** HEAD

**RemoteSha** 95f2fe0251ebac34853b7c121d3623effee9c50e

# Contents



---

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

---

## Description

Define window sizes (time_series_avg) to be used to calculate moving averages (mean). If time_series_avg is not supplied, the function calculates the mean over all years. If time_repeat is supplied, the function replicates the mean values for the defined amount of years.

## Usage

```
aggregate_time(x, time_series_avg = NULL, time_repeat = NULL)
```

### Arguments

| | |
|---|---|
| `x` | LPJmL output array with `dim(x)=c(cell, month, year)` |
| `time_series_avg` | integer. Number of years to be used for the moving average calculation. If `NULL`, all years are averaged for one status calculation, for `1` the whole time span is used to calculate a status time series. |
| `time_repeat` | integer, if supplied (default NULL), it defines a length of years to be replicated. Only if time_series_avg is not supplied. |

### Value

array with same amount of cells and months as x if time_series_avg is supplied. If time_repeat is supplied, the array has the same amount of cells and months as x but the amount of years is multiplied by time_repeat.

---

## `as_risk_level` *Convert status of control variable to risk level*

---

### Description

Convert status of control variable to planetary boundary risk level (safe, increasing risk, high risk), based on the output from calc_*

### Usage

```
as_risk_level(control_variable, type = "continuous", normalize = "safe")
```

### Arguments

| | |
|---|---|
| `control_variable` | output array from calc_* with the status of the control variable, incl. pb thresholds as attribute |
| `type` | character string to define whether to return risk level as continuous (normalized so that 0 = holocene state and 1 = planetary boundary; >1 = transgressed) or discrete variable (0 = no PB status assessed, 1 = safe, 2 = increasing risk, 3 = high risk) |
| `normalize` | character string to define normalization, either "safe" (normalized from holocene to pb = the safe zone) or "increasing risk" (normalized from pb to high risk level = increasing risk zone if the pb status is > pb, otherwise normalized from holocene to pb). Only used if type set to "continuous" |

### Examples

```
## Not run:
as_risk_level(
  control_variable = biosphere_status,
  type = "discrete"
)

## End(Not run)
```

---

`biosphere_status` *Status calculation of the biosphere integrity boundary.*

---

### Description

Biosphere status calculation based on BioCol (HANPP) from a baseline run (with potential natural vegetation) and a scenario run (actual land use) of LPJmL, both within the `time_span_scenario`. Additionally a separate reference NPP file (e.g. from a Holocene run) can be supplied with `files_reference` = list(npp = "path/to/npp.bin.json"), which will use time_span_reference, or file index years 3:32 if time_span_reference is not supplied.

### Usage

```
biosphere_status(
  files_scenario,
  files_reference,
  spatial_scale = "subglobal",
  time_span_scenario = as.character(1982:2011),
  time_span_reference = NULL,
  approach = "stenzel2023",
  time_series_avg = NULL,
  config_args = list(),
  thresholds = NULL,
  path_baseline,
  time_span_baseline = time_span_scenario,
  npp_threshold = 20,
  biocol_option = "only_above_zero",
  eurasia = TRUE,
  ...
)
```

### Arguments

`files_scenario`

list with variable names and corresponding file paths (character string) of the scenario LPJmL run. All needed files need to be provided. E.g.: list(grid = "/temp/grid.bin.json", npp = "/temp/npp.bin.json"). Handled via `calc_status`.

`files_reference`
: list with variable names and corresponding file paths (character string) of the reference NPP, HANPP should be compared against. In this case only NPP is required. list(npp = "/temp/npp.bin.json").

`spatial_scale`
: character string indicating spatial resolution either "grid", "subglobal" or "global"

`time_span_scenario`
: time span to be used for the scenario run, defined as character string

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

`approach`
: approach (character string) to be used , currently available approach is `"stenzel2023"`

`time_series_avg`
: integer. Number of years to be used for the moving average calculation. If `NULL`, all years are averaged for one status calculation, for `1` the whole time span is used to calculate a status time series.

`config_args`
: list of arguments to be passed on from the model configuration.

`thresholds`
: named character string with thresholds to be used to define the lower end of safe, increasing risk and high risk zone, e.g. c(holocene = 0.0, pb = 0.1, highrisk = 0.2). If set to NULL, default values from metric_files.yml will be used.

`path_baseline`
: character string with path to outputs for the baseline run, file names are taken from files scenario.

`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

`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)

`biocol_option`
: which biocol values to use for aggregation. options: netsum, only_above_zero, abs

`eurasia`
: logical. If `spatial_scale = "subglobal"` merge continents Europe and Asia to avoid arbitrary biome cut at europe/asia border. Defaults to `TRUE`

`...`
: arguments forwarded to `classify_biomes`

## Value

Object of class `control_variable` with the boundary status of the biosphere integrity boundary.

## Examples

```
## Not run:
boundary_status <- calc_status(
  boundary = "biosphere",
  config_scenario = "path/to/config_scenario.json",
```

```
  config_reference = "path/to/config_reference.json",
  spatial_scale = "global",
  time_span_scenario = 1901:2019,
  time_span_reference = 1901:1930,
  approach = "stenzel2023",
  path_baseline = "path/to/baseline_outputs"
)

## End(Not run)
```

---

`bluewater_status` *Status calculation of the bluewater boundary*

---

## Description

Planetary Boundary status calculation of the bluewater boundary (as part of the freshwater boundary) based on a scenario LPJmL run and a reference LPJmL run.

## Usage

```
bluewater_status(
  files_scenario,
  files_reference,
  spatial_scale,
  time_span_scenario = as.character(1982:2011),
  time_span_reference = time_span_scenario,
  approach = "gerten2020",
  time_series_avg = NULL,
  config_args = list(),
  thresholds = NULL,
  cut_min = 0.0864
)
```

## Arguments

`files_scenario`
: list with variable names and corresponding file paths (character string) of the scenario LPJmL run. Handled automatically via `calc_status()`.

`files_reference`
: list with variable names and corresponding file paths (character string) of the files_reference LPJmL run. Handled automatically via `calc_status()`.

`spatial_scale`
: character string indicating spatial resolution options: "global", "subglobal", "grid"; for "grid" the approach "gerten2020" is applicable based on EFR calculations; for "global"/"subglobal" the share (%) of total global/basin area with deviations is calculated

`time_span_scenario`
: time span to use output from the scenario run, e.g. `1982:2011`.

`time_span_reference`
: time span use output from the reference run, e.g. `1901:1930`.

`approach`
: approach (character string) to be used , currently available approach is `"gerten2020"` based on Gerten et al. 2020 for spatial_scale = "grid" and "wang_erlandsson2022" as well as "porkka2024" for spatial_scale = "global" or "subglobal"

`time_series_avg`
: integer. Number of years to be used for the moving average calculation. If `NULL`, all years are averaged for one status calculation, for `1` the whole time span is used to calculate a status time series.

`config_args`
: list of arguments to be passed on from the model configuration.

`thresholds`
: named character string with thresholds to be used to define the safe, increasing risk and high risk zone, the approach and scale specific default thresholds are defined in metric_files.yml are are applied if thresholds are set to NULL.

`cut_min`
: double. Exclude boundary calculations for discharge < cut_min and dismiss EFR transgresssions if < cut_min for "gerten2020" approach, Default: 0.0864 hm3/day (=1 m3/s)

## Value

Object of class `control_variable` with the boundary status of the bluewater boundary.

## Examples

```
## Not run:
boundary_status <- calc_status(
  boundary = "bluewater",
  config_scenario = "path/to/config_scenario.json",
  config_reference = "path/to/config_reference.json",
  spatial_scale = "global",
  time_span_scenario = 1901:2019,
  time_span_reference = 1901:1930,
  approach = "porkka2024"
)

## End(Not run)
```

---

`calc_efrs` *Calculate environmental flow requirements (EFRs)*

---

## Description

Calculate environmental flow requirements (EFRs) based on the number of years of `dim(x)[3]` or specify a nyear_avg calculate the EFRs for each bin in `dim(x)[3]`.

## Usage

```
calc_efrs(x, approach = "vmf")
```

## Arguments

| | |
|---|---|
| `x` | discharge array with `dim(x)=c(cell, month, year)` |
| `approach` | EFR approach to be used , available methods are `c("vmf", "q90q50")` based on Pastor et al. 2014 and c(vmf_min", "vmf_max") as modified by Gerten et al. 2020 as well as `"steffen2015"`, a modified version of vmf by Steffen et al. 2015 |

## Value

EFRs with same unit as `x` (discharge), with `dim(x)=c(ncells, 12)` or `dim(EFRs)=c(ncells, 12, dim(x)[3] / nyear_avg)` if nyear_avg is defined

## Examples

```
## Not run:
# basic example
efrs1 <- calcEFRs(discharge_30y = discharge, approach = "vmf")

dim(efrs1)
# c(67420, 12)

# example for using a 30 year average bin for a 90 year discharge and
#  interpolate between 3 windows afterwards to return 90 years (interpolated)
efrs2 <- calcEFRs(
  discharge_90y = discharge,
  approach="vmf"
)

dim(efrs2)
# c(67420, 12, 90)
# if interpolate == FALSE  dim(efrs2) returns c(67420, 12, 3)

# example for using a 1 year (no average) bin for a 100 year discharge
efrs3 <- calcEFRs(discharge_100y = discharge,
                  approach = "vmfmin")

dim(efrs3)
# c(67420, 12, 100)

## End(Not run)
```

`calc_status` *Calculate the planetary boundary status*

## Description

Calculate the PB status for a defined planetary boundary based on a scenario LPJmL run and a reference LPJmL run. For boundary function specific arguments to be passed (via `...`) see the respective function documentation of `biosphere_status()`, `nitrogen_status()`, `greenwater_status()`, `bluewater_status()` or `lsc_status()`

## Usage

```
calc_status(
  boundary,
  config_scenario,
  config_reference,
  spatial_scale,
  time_span_scenario = 1982:2011,
  time_span_reference = time_span_scenario,
  time_series_avg = NULL,
  approach = list(),
  thresholds = list(),
  in_parallel = TRUE,
  ...
)
```

## Arguments

| | |
|---|---|
| `boundary` | character vector, boundary for which status is calculated. Available terrestrial boundaries are `c("bluewater", "greenwater", "lsc", "nitrogen", "biosphere")`. |
| `config_scenario` | character string. File path to the LPjmL configuration file (json) of the scenario run. The configuration file contains the information about the LPJmL run, e.g. the output directory |
| `config_reference` | character string. See config_scenario. For the reference run |
| `spatial_scale` | character string indicating spatial resolution options: "global", "subglobal", "grid"; |
| `time_span_scenario` | time span to be used for the scenario run, defined as an integer (or character) vector, e.g. `1982:2011` (default) |
| `time_span_reference` | time span to be used for the scenario run, defined as an integer (or character) vector, e.g. `1901:1930`. Can differ in offset and length from `time_span_scenario`! If `NULL` value of `time_span_scenario` is used |

`time_series_avg`
: integer. Number of years to be used for the moving average calculation. If `NULL`, all years are averaged for one status calculation, for `1` the whole time span is used to calculate a status time series.

`approach`
: list of methods to be used for each boundary. If `NULL` the default approach is used

`thresholds`
: list of thresholds to be used for each boundary. If `NULL` the default thresholds are used

`in_parallel`
: logical, if `TRUE` the function uses parallelization (default) based on the `future` package (asynchronous execution). If `FALSE` no parallelization is used

`...`
: further arguments to be passed to each calc_* function

## Value

list with objects of class `control_variable`. To directly get the `boundary_status` use `as_risk_level()`.

## Examples

```
## Not run:
boundary_status <- calc_status(
  boundary = c("biosphere","nitrogen", "greenwater", "bluewater", "lsc")
  config_scenario = "path/to/config_scenario.json",
  config_reference = "path/to/config_reference.json",
  spatial_scale = "global",
  time_span_scenario = 1901:2019,
  time_span_reference = 1901:1930
)

## End(Not run)
```

---

# calc_water_deviations

*Calculate water status based on deviations of a monthly scenario variable from a corresponding monthly reference variable*

---

## Description

Calculate deviations (<q5 / >q95) for a monhtly variable in a scenario LPJmL run as compared to a reference LPJmL run, either referring to global area share with deviations (spatial_scale: global), or to number of months or years with deviations (spatial resolution: cell). From this, calculate a global or gridded PB status

### Usage

```
calc_water_deviations(
  files_scenario,
  files_reference,
  spatial_scale = "subglobal",
  time_span_scenario = NULL,
  time_span_reference,
  approach = "porkka2024",
  thresholds = NULL,
  time_series_avg = NULL,
  config_args = list(),
  variable = "rootmoist"
)
```

### 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(leaching = "/temp/leaching.bin.json")

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

`spatial_scale`
: character string indicating spatial scale; "global" or "subglobal" for calculation of the share (%) of total global/basin area with deviations (either one value per year (wang-erlandsson2022) or one value per year and month (porkka2024)); "grid" not yet defined

`time_span_scenario`
: time span to be used for the scenario run, defined as character string, e.g. `as.character(1982:2011)` (default)

`time_span_reference`
: time span to be used for the reference run, defined as a character string (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

`approach`
: approach (character string) to be used , currently available approach is `c("wang-erlandsson2022")` based on Wang-Erlandsson et al. 2022 (referring only to the driest/wettest month of each year) or `porkka2024` based on Porkka et al. 2023 (referring to each month of a year; default)

`thresholds`
: list with thresholds to be used to define the safe, increasing risk and high risk zone, For spatial_scale = "global" and "subglobal", this refers to the quantiles of the global/basin area with deviations in the reference period. The default is: c(holocene = 50, pb = 95, highrisk = NULL). If set to NULL, the default is taken from metric_files.yml For highrisk, the value is currently hard-coded to 0.5 (following Richardson et al. 2023)

time_series_avg
: integer. Number of years to be used for the moving average calculation. If NULL, all years are averaged for calculation, for 1 the whole time span is used to calculate a time series.

config_args
: list of arguments to be passed on from the model configuration.

variable
: character string with the name of the variable to be used for the calculation of the water deviations. Default is "rootmoist"

---

`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 information.

## Usage

```
classify_biomes(
  config_reference = NULL,
  files_reference = NULL,
  time_span_reference,
  savanna_proxy = list(vegc = 7500),
  montane_arctic_proxy = list(elevation = 1000),
  tree_cover_thresholds = list(),
  approach = "default",
  time_series_avg = NULL,
  config_args = list()
)
```

## Arguments

config_reference
: character string. File path to the LPjmL configuration file (json) of the reference run. The configuration file contains the information about the LPJmL run, e.g. the output directory

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(vegc = "/temp/vegc.bin.json`. If `config_reference` is supplied with all needed files, files reference can be set to `NULL`.

time_span_reference
: time span to be used for the classification of biomes, defined as 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 (current default: 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: 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.

`approach` character string indicating which biome classification approach to use. Currently only one is defined ("default").

`time_series_avg`
integer. Number of years to be used for the moving average calculation. If `NULL`, all years are averaged for one status calculation, for `1` the whole time span is used to calculate a status time series.

`config_args` list of arguments to be passed on from the model configuration.

## 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(
  config_reference = "./outputs/config.json",
  time_span_reference = 1982:2011
)

## End(Not run)
```

---

`greenwater_status` *Status calculation of the greenwater boundary*

---

## Description

Planetary Boundary status calculation of the greenwater boundary based on rootmoisture in a scenario LPJmL run and a reference LPJmL run.

## Usage

```
greenwater_status(
  files_scenario,
  files_reference,
  spatial_scale = "global",
  time_span_scenario = as.character(1982:2011),
  time_span_reference = time_span_scenario,
  approach = "wang-erlandsson2022",
  time_series_avg = NULL,
  config_args = list(),
  thresholds = NULL
)
```

## Arguments

`files_scenario`
: list with variable names and corresponding file paths (character string) of the scenario LPJmL run. Handled automatically via `calc_status()`.

`files_reference`
: list with variable names and corresponding file paths (character string) of the files_reference LPJmL run. Handled automatically via `calc_status()`.

`spatial_scale`
: character string indicating spatial resolution either "grid", "subglobal" or "global" for calculation of the share (%) of total global area with deviations

`time_span_scenario`
: time span to use output from the scenario run, e.g. `1982:2011`.

`time_span_reference`
: time span use output from the reference run, e.g. `1901:1930`.

`approach`
: approach (character string) to be used , currently available approach is `c("wang-erlandsson2022")` based on Wang-Erlandsson et al. 2022 (referring only to the driest/wettest month of each year) or `porkka2024` based on Porkka et al. 2023 (referring to each month of a year)

`time_series_avg`
: integer. Number of years to be used for the moving average calculation. If `NULL`, all years are averaged for one status calculation, for `1` the whole time span is used to calculate a status time series.

`config_args`
: list of arguments to be passed on from the model configuration.

`thresholds`
: named character string with thresholds to be used to define the safe, increasing risk and high risk zone, e.g. c(holocene = 0.5, pb = 0.95, highrisk = 0.99). For spatial resolution = "grid", this refers to the p value (significance level of increases in deviations) with the default: c(holocene = 1, pb = 0.05, highrisk = 0.01). For spatial resolution = "global", this refers to the quantiles of the global area with deviations in the reference period. The dafault for global resolution is: c(holocene = 0.5, pb = 0.95, highrisk = 0.99). If set to NULL, the respective default is taken (see above; matching the spatial_scale, defined in metric_files.yml).

### Value

Object of class `control_variable` with the boundary status of the greenwater boundary.

### Examples

```
## Not run:
boundary_status <- calc_status(
  boundary = "greenwater",
  config_scenario = "path/to/config_scenario.json",
  config_reference = "path/to/config_reference.json",
  spatial_scale = "global",
  time_span_scenario = 1901:2019,
  time_span_reference = 1901:1930,
  approach = "porkka2024"
)

## End(Not run)
```

---

## `list_outputs` *List required LPJmL outputs and temporal resolution*

---

### Description

Function to return a list of output IDs with required resolution and file names for a given metric. The list is based on the metric_files.yml file in the boundaries package (`"./inst/metric_files.yml"`).

### Usage

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

### Arguments

| | |
|---|---|
| `metric` | Character string containing name of metric to get required outputs. Available options are `c("biome", "nitrogen", "lsc", "bluewater", "greenwater", "biosphere")` or just `"all"` or `"benchmark"`. Default is `"all"`. |
| `spatial_scale` | character. Spatial resolution, available options are `"subglobal"` (at the biome level), `"global"` and `"grid"` or `"all"` (default). |
| `approach` | List of character strings containing the approach to calculate the metric. Or `"all"` to get all approaches (default). |
| `only_first_filename` | Logical. If TRUE, only the first file name will be returned for each output. If FALSE, all file names will be returned. |

### Value

List of output IDs with required resolution and file names for a given metric

### Examples

```
## Not run:
list_outputs(
  "biome",
  approach = list("biome" = approach),
  spatial_scale = "subglobal",
  only_first_filename = FALSE
)

## End(Not run)
```

---

## lsc_status *Status calculation of the land-system change boundary*

---

### Description

Planetary Boundary status calculation of the LSC (land-system change) boundary based on a scenario LPJmL run and a reference LPJmL run.

### Usage

```
lsc_status(
  files_scenario,
  files_reference,
  spatial_scale = "subglobal",
  time_span_scenario = as.character(1982:2011),
  time_span_reference = time_span_scenario,
  approach = "steffen2015",
  time_series_avg = NULL,
  config_args = list(),
  thresholds = NULL,
  eurasia = TRUE,
  ...
)
```

### Arguments

`files_scenario`
: list with variable names and corresponding file paths (character string) of the scenario LPJmL run. Handled automatically via `calc_status()`.

`files_reference`
: list with variable names and corresponding file paths (character string) of the reference LPJmL run. Handled automatically via `calc_status()`.

`spatial_scale` character. Spatial resolution, available options are `"subglobal"` (at the biome level, default), `"global"` and `"grid"`

`time_span_scenario`
time span to use output from the scenario run, e.g. `1982:2011`.

`time_span_reference`
time span use output from the reference run, e.g. `1901:1930`.

`approach` approach (character string) to be used , currently available approach is `"steffen2015"`

`time_series_avg`
integer. Number of years to be used for the moving average calculation. If `NULL`, all years are averaged for one status calculation, for `1` the whole time span is used to calculate a status time series.

`config_args` list of arguments to be passed on from the model configuration.

`thresholds` list with deforestation thresholds for defining safe, increasing risk and high risk zone. Default based on Steffen et al. 2015 if thresholds set to NULL (https://doi.org/10.1126/science.1259855): for gridded and biome scale application: 'list(pb = list(temperate = 0.5, tropical = 0.15, boreal = 0.15), highrisk = list(temperate = 0.7, tropical = 0.4, boreal = 0.4)) for global scale application: list(holocene = 0, pb = 0.25, highrisk = 0.46) pb = threshold between safe zone and increasing risk zone (e.g. 50% for boreal forest with default value) highrisk = threshold between increasing risk and high risk zone

`eurasia` logical. If `spatial_scale = "subglobal"` merge continents Europe and Asia to avoid arbitrary biome cut at europe/asia border. Defaults to `TRUE`

`...` arguments forwarded to `classify_biomes`

## Value

Object of class `control_variable` with the boundary status of the lsc boundary.

## Examples

```
## Not run:
boundary_status <- calc_status(
  boundary = "lsc",
  config_scenario = "path/to/config_scenario.json",
  config_reference = "path/to/config_reference.json",
  spatial_scale = "global",
  time_span_scenario = 1901:2019,
  time_span_reference = 1901:1930
)

## End(Not run)
```

`nitrogen_status` *Status calculation of the nitrogen boundary*

## Description

Planetary Boundary status calculation of the the nitrogen boundary based on a scenario LPJmL run and if `approach == "braun2022_minusref"` a reference LPJmL run.

## Usage

```
nitrogen_status(
  files_scenario,
  files_reference,
  spatial_scale = "grid",
  time_span_scenario = 1982:2011,
  time_span_reference = time_span_scenario,
  approach = "braun2022",
  time_series_avg = NULL,
  config_args = list(),
  thresholds = NULL,
  cut_arid = 0.2,
  cut_runoff = 0,
  with_groundwater_denit = TRUE
)
```

## Arguments

`files_scenario`
: list with variable names and corresponding file paths (character string) of the scenario LPJmL run. Handled automatically via `calc_status()`.

`files_reference`
: list with variable names and corresponding file paths (character string) of the files_reference LPJmL run. Handled automatically via `calc_status()`.

`spatial_scale`
: character. Spatial resolution, available options are `"global"` and `"grid"`

`time_span_scenario`
: time span to use output from the scenario run, e.g. `1982:2011`.

`time_span_reference`
: time span use output from the reference run, e.g. `1901:1930`.

`approach`
: (character string) to be used , currently available approach is `"braun2022"` based on unpublished suggestion by Johanna Braun. Second approach option is `"braun2022_minusref"` to subtract reference run output

`time_series_avg`
: integer. Number of years to be used for the moving average calculation. If `NULL`, all years are averaged for one status calculation, for `1` the whole time span is used to calculate a status time series.

`config_args`
: list of arguments to be passed on from the model configuration.

`thresholds` list with highrisk and pb threshold for N concentration (mg N/l) in runoff to surface water Default: highrisk = 5, pb = 2 (based on Schulte-Uebbing et al. 2022, https://doi.org/10.1038/s41586-022-05158-2: "we used a threshold for N concentration in run-off to surface water. This threshold was set to 5.0 mgN/l, based on the assumption that on average 50% of N entering surface water is removed through retention and sedimentation"))

`cut_arid` double. Exclude boundary calculations below the defined threshold for aridity (annual precipitation / annual potential evapotranspiration); Default: 0.2

`cut_runoff` double. Exclude boundary calculations below the defined runoff threshold; Default: 0 mm per year (no treshold)

`with_groundwater_denit` logical. Include global assumptions made on groundwater denitrification losses. Defaults to TRUE ( = simulated leaching is multiplied with 0.71 based on simulated denitrification losses in ground water from Bouwman et al 2013)

## Value

Object of class `control_variable` with the boundary status of the nitrogen boundary.

## Examples

```
## Not run:
boundary_status <- calc_status(
  boundary = "nitrogen",
  config_scenario = "path/to/config_scenario.json",
  config_reference = "path/to/config_reference.json",
  spatial_scale = "global",
  time_span_scenario = 1901:2019,
  time_span_reference = 1901:1930
)

## 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(x, filename = NULL, projection = "+proj=robin", grid_path = NULL)
```

### Arguments

| | |
|---|---|
| x | output (list) from classify_biomes() |
| filename | directory for saving the plot (character string) |
| projection | character string defining the projection, default set to "+proj=robin" |
| 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(
  x = biomes,
  filename = "/p/projects/open/Johanna/R/biomes.pfd"
  grid_path = ".grid.bin.json"
)

## End(Not run)
```

---

## plot_status *Plot the status of planetary boundaries*

---

### Description

Plot the status of planetary boundaries. The function takes the output from calc_status and plots the status of the planetary boundaries depending on the spatial scale applying different forms of plotting.

### Usage

```
plot_status(x, filename = NULL, add_legend = TRUE, stylized = FALSE, ...)
```

### Arguments

| | |
|---|---|
| x | output object from calc_* with the status of the control variable for one point in time, incl. pb thresholds as attribute |
| filename | character string providing file name (including directory and file extension). Defaults to NULL (plotting to screen) |
| add_legend | logical, specify whether a legend should be plotted |
| stylized | Logical. If `spatial_scale == "global"`, the function will plot the status of the planetary boundaries using a stylized plot. |
| ... | additional arguments passed to the plotting functions, see also `status_global`, `status_map` and `status_stylized` |

### Examples

```
## Not run:
pb_status <- calc_status(
  boundary = c("lsc", "biosphere", "bluewater", "greenwater", "nitrogen"),
  config_scenario = "./config_lu_1500_2016.json",
  config_reference = "./config_pnv_1500_2016.json",
  time_span_scenario = as.character(1986:2016),
  time_span_reference = as.character(1986:2016),
  spatial_scale = "global",
  approach = list(
    bluewater = "porkka2024",
    nitrogen = "schulte_uebbing2022"
  ),
  savanna_proxy = list(vegc = 7500),
  time_series_avg = 1,
  path_baseline = "./pnv_1500_2016/",
)

plot_status(
  x = pb_status,
  filename = "status.png",
  add_legend = TRUE,
  stylized = TRUE
)

## End(Not run)
```

---

## status_global *Plot the global status of planetary boundaries*

---

### Description

Plot line plots with the PB status over time for a scenario LPJmL run and derived planetary boundary statuses. Legend can be plotted seperately based on the status_legend() function

### Usage

```
status_global(
  x,
  filename = NULL,
  all_in_one = FALSE,
  ncol = 2,
  normalize = "increasing risk"
)
```

### Arguments

| | |
|---|---|
| x | list with global output from calc_status |
| filename | character string providing file name (including directory and file extension). Defaults to NULL (plotting to screen and returning plot object for further customization) |
| all_in_one | boolean, if TRUE, all PB stati will be normalized and plotted in one panel |
| ncol | number of plot columns (only relevant if more than one pb is plotted and all_in_one = FALSE) |
| normalize | see as_risk_level() for details. Default set to "increasing risk". Only relevant if all_in_one = TRUE |

### Examples

```
## Not run:
status_global(
  filename = "./my_boundary_status.png",
  x = status_output,
  all_in_one = FALSE,
  ncol = 2
)

## End(Not run)
```

---

## status_legend *Plot the legend for the normalized colors of PB statuses*

---

### Description

Plot a legend for the colors of PB statuses, normalized based on the size of the increasing risk, for globally aggregated plots, or spatially distributed maps

### Usage

```
status_legend(filename = NULL, fontsize = 3)
```

### Arguments

| | |
|---|---|
| filename | character string providing file name (including directory and file extension). Defaults to NULL (return plot object for further adaptation) |
| fontsize | numeric specifying the size of the font to be used for legend labels. Default set to 3. |

### Examples

```
## Not run:
status_legend(
  filename = "./mylegend.png",
)

## End(Not run)
```

---

`status_map` *Plot the global status of planetary boundaries*

---

### Description

Plot global map(s) with the status of planetary boundaries for a scenario LPJmL run and derived planetary boundary statuses. Legend can be plotted seperately based on the status_legend() function

### Usage

```
status_map(
  x,
  filename = NULL,
  risk_level = TRUE,
  projection = "+proj=robin",
  ncol = 2,
  grid_path = NULL
)
```

### Arguments

| | |
|---|---|
| `x` | output object from calc_* with the status of the control variable for one point in time, incl. pb thresholds as attribute |
| `filename` | character string providing file name (including directory and file extension). Defaults to NULL (plotting to screen and returning the ggplot object) |
| `risk_level` | logical, specify whether the status should be plotted as risk level. Default set to TRUE. |
| `projection` | character string defining the projection, default set to "+proj=robin" |
| `ncol` | integer, number of columns in the plot, default set to 2 |
| `grid_path` | character string providing the path to a grid file |

## Examples

```
## Not run:
 status_map(
  filename = "./my_boundary_status.png",
  x = calc_output
  grid_path = "/path/to/gridfile.bin.json"
)

## End(Not run)
```

---

`status_stylized` *Plot polar boundaries plot including time series of boundaries*

---

## Description

Plot time series of boundaries into iconic polar boundaries plot only focussing on the terrestrial boundaries (half-circle). Wedges are scaled and normalized based on the "increasing risk" method according to each boundary (see `as_risk_level()` for details).

## Usage

```
status_stylized(x, filename = NULL, add_legend = TRUE, background_alpha = 1)
```

## Arguments

| | |
|---|---|
| `x` | list with global output from calc_status |
| `filename` | character string providing file name (including directory and file extension). Defaults to NULL (plotting to screen and returning ´ plot object) |
| `add_legend` | logical, specify whether a legend should be plotted |
| `background_alpha` | numeric, specify the alpha value for the background (default 1 - transparent) |

## Examples

```
## Not run:
pb_status <- calc_status(
  boundary = c("lsc", "biosphere", "bluewater", "greenwater", "nitrogen"),
  config_scenario = "./config_lu_1500_2016.json",
  config_reference = "./config_pnv_1500_2016.json",
  time_span_scenario = as.character(1986:2016),
  time_span_reference = as.character(1986:2016),
  spatial_scale = "global",
  approach = list(
    bluewater = "porkka2024",
    nitrogen = "schulte_uebbing2022"
  ),
```

```
  savanna_proxy = list(vegc = 7500),
  time_series_avg = 1,
  path_baseline = "./pnv_1500_2016/",
)

status_stylized(pb_status, "status_stylized.png")

## End(Not run)
```

---

`validate_simulation` *Validate simulated global PB-relevant variables against literature Calculate a table with global modelled vs literature values for key variables relevant to planetary boundaries*

---

## Description

Validate simulated global PB-relevant variables against literature Calculate a table with global modelled vs literature values for key variables relevant to planetary boundaries

## Usage

```
validate_simulation(
  config_scenario,
  config_reference,
  time_span_scenario,
  time_span_reference,
  path_baseline,
  filename,
  ...
)
```

## Arguments

`config_scenario`
: character string. File path to the LPjmL configuration file (json) of the scenario run. The configuration file contains the information about the LPJmL run, e.g. the output directory

`config_reference`
: character string. See config_scenario. For the reference run

`time_span_scenario`
: time span to be used for the scenario run and parallel PNV run, defined as a character string, e.g. `as.character(1982:2011)`

`time_span_reference`
: time span to be used for the reference run, defined as an integer vector, e.g. `1901:1930`. Can differ in offset and length from `time_span_scenario`! If `NULL` value of `time_span_scenario` is used

| | |
|---|---|
| path_baseline | character string for path to outputs for the baseline run, file names are taken from files scenario |
| filename | character string for file path to save the output, (.csv file) |
| ... | arguments to be passed to calc_status |

## Value

table with comparison between lpjml values and literature ranges

## Examples

```
## Not run:
validation <- validate_simulation(
  config_scenario = "./my_path/config_scenario.json",
  config_reference = "./my_path/config_reference.json",
  time_span_scenario = as.character(2010:2017),
  time_span_reference = as.character(1500:1699),
  path_baseline = "./my_path/outputs_baseline/",
  filename = "./my_path/table.csv"
)

## End(Not run)
```

# Index