

Package: rsofun (via r-universe)

October 25, 2026

Title The P-Model and BiomeE Modelling Framework

Version 5.1.0.9001

Description Implements the Simulating Optimal FUNctioning framework for site-scale simulations of ecosystem processes, including model calibration. It contains 'Fortran 90' modules for the P-model (Stocker et al. (2020) <[doi:10.5194/gmd-13-1545-2020](https://doi.org/10.5194/gmd-13-1545-2020)>), SPLASH (Davis et al. (2017) <[doi:10.5194/gmd-10-689-2017](https://doi.org/10.5194/gmd-10-689-2017)>) and BiomeE (Weng et al. (2015) <[doi:10.5194/bg-12-2655-2015](https://doi.org/10.5194/bg-12-2655-2015)>).

URL <https://github.com/geco-bern/rsfun>,
<https://geco-bern.github.io/rsfun/>

BugReports <https://github.com/geco-bern/rsfun/issues>

License GPL-3

Depends R (>= 4.1.0)

Imports dplyr (>= 1.2.0), purrr, tidyr, magrittr, GenSA,
BayesianTools, lubridate, multidplyr, stats, utils, doParallel,
msm, foreach, methods

LazyData true

LazyDataCompression xz

ByteCompile true

NeedsCompilation yes

RoxygenNote 7.3.3

Suggests covr, constructive, cowplot, rcmdcheck, testthat (>= 3.0.0),
rmarkdown, ggplot2, knitr, sensitivity, rpmodel, rlang, khroma,
yardstick

VignetteBuilder knitr

Encoding UTF-8

Config/testthat/edition 3

Config/pak/sysreqs cmake make libicu-dev libuv1-dev libssl-dev zlib1g-dev

Repository <https://geco-bern.r-universe.dev>

Date/Publication 2026-09-25 07:35:04 UTC

RemoteUrl <https://github.com/geco-bern/rsfun>

RemoteRef HEAD

RemoteSha 0bf91fc97dc0bbc650aeb04f985972bae7bd4b1d

Contents

biomee_gs_leuning_drivers	3
biomee_gs_leuning_output	8
biomee_p_model_drivers	9
biomee_p_model_luluc_drivers	9
biomee_p_model_luluc_output	10
biomee_p_model_output	10
biomee_validation	11
build_luc_matrix	11
calib_sofun	12
calib_sofun_legacy	15
cost_likelihood_biomee	17
cost_likelihood_pmodel	18
cost_likelihood_pmodel_legacy	20
cost_rmse_biomee	22
cost_rmse_pmodel	23
createMixedPrior	25
get_mod_obs_pmodel	25
init_dates_dataframe	26
is_beta_prior	27
is_lognormal_prior	28
is_normal_prior	29
is_trunclognormal_prior	29
is_truncnormal_prior	30
is_uniform_prior	31
p_model_oldformat_drivers	31
p_model_oldformat_validation	32
pmodel_drivers	32
pmodel_output	34
pmodel_validation	34
run_biomee_f_bysite	36
run_pmodel_f_bysite	41
run_pmodel_onestep_f_bysite	44
runread_biomee_f	46
runread_pmodel_f	47

Index

50

biomee_gs_leuning_drivers

rsofun BiomeE driver data (Leuning photosynthesis model)

Description

Example driver to run the BiomeE-model at the CH-LAE site using the Leuning photosynthesis specification (and half-hourly time step) It can also be used together with leaf trait data from CH-LAE ([biomee_validation](#)) to optimize model parameters.

Usage

```
biomee_gs_leuning_drivers
```

Format

A tibble of driver data.

sitename Site name

params_siml Simulation parameters as a data.frame, including the following data:

spinup Flag indicating whether this simulation does spin-up (deprecated).

spinupyears Number of spin-up years. Set to 0 for no spinup.

recycle Number of first N years of forcing data.frame that are recycled for spin-up.

firstyeartrend Year of first transient year (AD) (optional). Is only used to set years in output data frames. Defaults to 0 if not provided.

nyeartrend Number of transient years (optional). Determines the length of simulation output after spin-up. Defaults to number of years contained in the forcing data. (If longer than forcing data, last year of forcing is repeated until the end (spin-down).)

steps_per_day Time resolution of the forcing (day-1).

do_U_shaped_mortality Flag indicating whether U-shaped mortality is used.

do_closedN_run Flag indicating whether doing N closed runs to recover N balance enforcing 0.2 kg N m⁻² in the inorganic N pool.

method_photosynth String specifying the method of photosynthesis used in the model, either "pmodel" or "gs_leuning".document()

method_mortality String indicating the type of mortality in the model. One of the following: "dbh" is size-dependent mortality, "const_selfthin" is constant self thinning (in development), "cstarvation" is carbon starvation, and "growthrate" is growth rate dependent mortality.

do_daily_diagnostics Whether to output daily diagnostics ('output_daily_tile'). Default: True.

site_info Site meta info in a data.frame. This data structure can be freely used for documenting the dataset, but must include at least the following data:

lon Longitude of the site location in degrees east.

lat Latitude of the site location in degrees north.

elv Elevation of the site location, in meters above sea level.

forcing Forcing data.frame used as input

ppfd Photosynthetic photon flux density ($\text{mol m}^{-2} \text{s}^{-1}$)

tair Air temperature (deg C)

vpd Vapor pressure deficit (Pa)

rain Precipitation ($\text{kgH}_2\text{O m}^{-2} \text{s}^{-1}$ == mm s^{-1})

wind Wind velocity (m s^{-1})

pair Atmospheric pressure (Pa)

co2 Atmospheric CO_2 concentration in ppm.

params_tile Tile-level model parameters, into a single row data.frame, including the following data:

soiltype Integer indicating the type of soil: Sand = 1, LoamySand = 2, SandyLoam = 3, SiltLoam = 4, FrittedClay = 5, Loam = 6, Clay = 7.

FLDCAP Field capacity (vol/vol). Water remaining in a soil after it has been thoroughly saturated and allowed to drain freely.

WILTPT Wilting point (vol/vol). Water content of a soil at which plants wilt and fail to recover.

K1 Fast soil C decomposition rate (yr^{-1}).

K2 Slow soil C decomposition rate (yr^{-1}).

K_nitrogen Mineral Nitrogen turnover rate (yr^{-1}).

MLmixRatio Ratio of C and N returned to litters from microbes.

etan N loss rate through runoff (organic and mineral) (yr^{-1}).

LMamin Minimum LMA, leaf mass per unit area, LMamin/LMA describes fraction of leaf that become fine litter (metabolic parts), kg C m^{-2} . Has to be smaller than LMA. Difference to LMA is structural parts (becoming coarse litter).

fsc_fine Fraction of fast turnover carbon in fine biomass.

fsc_wood Fraction of fast turnover carbon in wood biomass.

l_fract Fraction of the carbon retained after leaf drop.

retransN Retranslocation coefficient of nitrogen.

f_initialBSW Coefficient for setting up initial sapwood.

f_N_add Re-fill of N for sapwood.

tf_base Calibratable scalar for respiration, used to increase LUE levels.

params_species A data.frame containing species-specific model parameters, with one species per row, including the following data:

The following columns pertaining to the plant type: lifeform Integer set to 0 for grasses and 1 for trees.

pt Integer indicating the type of plant according to photosynthesis: 0 for C3; 1 for C4

The following columns pertaining to the root parameters: alpha_FR Fine root turnover rate (yr^{-1}).

rho_FR Material density of fine roots (kg C m^{-3}).

root_r Radius of the fine roots, in m.

root_zeta e-folding parameter of root vertical distribution, in m.

Kw_root Fine root water conductivity ($\text{mol m}^{-2} \text{s}^{-1} \text{MPa}^{-1}$).

The following columns pertaining to the photosynthesis parameters: wet_leaf_dreg
Wet leaf photosynthesis down-regulation.

m_cond Factor of stomatal conductance.

alpha_phot Photosynthesis efficiency.

gamma_LN Leaf respiration coefficient per unit N, in $\text{mol C kg N}^{-1} \text{yr}^{-1}$.

gamma_SW Sapwood respiration rate, in $\text{kg C m}^{-2} \text{yr}^{-1}$.

gamma_FR Fine root respiration rate, $\text{kg C kg C}^{-1} \text{yr}^{-1}$.

tk_crit Critical temperature triggering offset of leaf phenology (unused with evergreen), in Kelvin.

tk_crit_on Critical temperature triggering onset of leaf phenology (unused with evergreen), in Kelvin.

gdd_crit Critical value of GDD5 for turning ON leaf phenology (unused with evergreen).

kphio Quantum yield efficiency φ_0 at optimal temperature, in mol mol^{-1} . (optional, defaults to 0.05)

beta Unit cost ratio of carboxylation to transpiration, unitless. (optional, defaults to 146.0)

rd_to_vcmax Ratio of R_{dark} to V_{cmax25} , unitless. (optional, defaults to 0.014)

kc_jmax J_{max} cost ratio, unitless. (optional, defaults to 0.41)

kphio_par_a Shape parameter of temperature-dependency of quantum yield efficiency, in degC^{-2} . (optional, defaults to 0.0)

kphio_par_b Optimal temperature of quantum yield efficiency, in degC . (optional, defaults to 25.0)

The following columns pertaining to the allometry parameters: alphaHT Coefficient for allometry ($\text{height} = \text{alphaHT} * \text{DBH}_m ** \text{thetaHT}$), in $\text{m m}^{-\text{thetaHT}}$.

thetaHT Coefficient for allometry ($\text{height} = \text{alphaHT} * \text{DBH}_m ** \text{thetaHT}$), unitless.

thetaBM Coefficient for allometry ($\text{biomass} = \text{alphaBM} * \text{DBH} ** \text{thetaBM}$), unitless. Since alphaBM is derived from alphaHT , it makes sense to also ensure $\text{thetaBM} = \text{thetaHT} + 2$.

alphaCA Coefficient for allometry ($\text{projected crown area} = \text{alphaCA} * \text{DBH}_m ** \text{thetaCA}$), in $\text{m}^2 - / \text{thetaCA}$.

thetaCA Coefficient for allometry ($\text{projected crown area} = \text{alphaCA} * \text{DBH}_m ** \text{thetaCA}$), unitless. Dybzinski (eq. G1) showed that $\text{thetaCA} = \text{thetaBM} - 1$. Moreover, Weng et al. (2015, eq A2) uses $\text{thetaBM} = \text{thetaHT} + 2$.

rho_wood Wood density (kg C m^{-3}). (Only used to derive alphaBM .)

taperfactor Multiplicative factor to correct the cylindric volume/mass calculated with DBH, unitless. E.g. for a cone taperfactor is 1/3 ($V = 1/3 \text{ Pi} * r^2 * \text{HT}$). (Only used to derive alphaBM .)

phiCSA Ratio of sapwood area to leaf area.

phiRL Ratio of fine root to leaf area.

extinct TODO: same as kappa below) light extinction coefficient in the canopy (Beer's law), unitless. (optional, defaults to 0.75)

kappa light extinction coefficient in the canopy (Beer's law), unitless. (optional, defaults to 0.5)

The following columns pertaining to the reproduction parameters: **seedlingsize** Initial size of seedlings, in kg C per individual.

maturalage Age at which trees can reproduce (years).

v_seed Fraction of G_SF to G_F.

The following columns pertaining to the mortality parameters: **mortrate_d_c** Canopy tree mortality rate (yr^{-1}). (Differently used depending on method_mortality and do_U_shaped_mortality: CAI_max ($\text{m}^2 \text{m}^{-2}$) if "const_selfthin", multiplicative factor (-) if "cstarvation" (), multiplicative factor ($\text{yr}^{-1} \text{m}^{-1.5}$) (deathrate = par_mort * dbh ** 1.5) if "dbh" and "do_U_shaped_mortality")

mortrate_d_u Understory tree mortality rate (yr^{-1}).

A_mort A coefficient in understory mortality rate correction, (deathrate = mortrate_d_u * (1+A*exp(B*DBH))/(1+exp(B*DBH))), in year^{-1} . (optional, defaults to 9.0)

B_mort B coefficient in understory mortality rate correction, (deathrate = mortrate_d_u * (1+A*exp(B*DBH))/(1+exp(B*DBH))), in m^{-1} . (optional, defaults to -60.0)

The following columns pertaining to the leaf parameters: **LMA** Leaf mass per unit area (kg C m^{-2}).

LNbase Basal leaf N per unit leaf area, in kg N m^{-2} . NOTE: it is used to represent the N content in metabolic parts of the leaf. (TODO: correct?)

CNleafsupport C/N ratio of the structural parts of the leaf (non-metabolic parts).

tauNSC Residence time of C in NSC (to define storage capacity), in years.

f_LFR_max Max fraction of total C growth that is allocated to leaves and fine root (remaining C growth used for seeds and DBH growth), unitless. (optional, defaults to 0.85)#'

fNSmax Multiplier for NSNmax as sum of potential bl and br.

internal_gap_frac Fraction of internal gaps in the canopy

The following columns pertaining to the C/N ratios for plant pools: **CNsw0** C/N ratio of sapwood, in kg C kg N^{-1} .

CNwood0 C/N ratio of heartwood, in kg C kg N^{-1} .

CNroot0 C/N ratio of roots, in kg C kg N^{-1} .

CNseed0 C/N ratio of seeds, in kg C kg N^{-1} .

Nfixrate0 Reference N fixation rate ($\text{kg N kg C}^{-1} \text{root}$).

NfixCost0 Carbon cost of N fixation (kg C kg N^{-1}).

The following columns pertaining to the calibratable parameters: **LAI_light** Maximum LAI limited by light.

init_cohort A data.frame of initial cohort specifications, including the following data:

init_cohort_species Index of a species described in param_species.

init_cohort_age Initial tree/cohort age, in yr. (optional) Default: 0.

init_cohort_nindivs Initial individual density, in individuals per m^2 .

init_cohort_bl Initial biomass of leaf, in kg C per individual.

init_cohort_br Initial biomass of fine root, in kg C per individual.

init_cohort_bsw Initial biomass of sapwood, in kg C per individual.

init_cohort_bHW Initial biomass of heartwood, in kg C per individual.

- init_cohort_seedC** Initial biomass of seed, in kg C per individual.
- init_cohort_nsc** Initial non-structural biomass, in kg C per individual. (optional, defaults to value derived from br_max and bl_max)
- init_cohort_bl_n14** Initial nitrogen in leaf, in kg N per individual. (optional, defaults to C biomass * CN-ratio)
- init_cohort_br_n14** Initial nitrogen in fine root, in kg N per individual. (optional, defaults to C biomass * CN-ratio)
- init_cohort_bsw_n14** Initial nitrogen in sapwood, in kg N per individual. (optional, defaults to C biomass * CN-ratio)
- init_cohort_bHW_n14** Initial nitrogen in heartwood, in kg N per individual. (optional, defaults to C biomass * CN-ratio)
- init_cohort_seedC_n14** Initial nitrogen in seed, in kg N per individual. (optional, defaults to C biomass * CN-ratio)
- init_cohort_nsc_n14** Initial non-structural biomass, in kg N per individual. (optional, defaults to value derived from br_max and bl_max)
- lu_index** Land use type this cohorts belongs to (given as index in init_lu array). (optional) Default: 0 (attach to all LU types except thoses which do not accept vegetation – cf init_lu.vegetated).
- restart_status** Restart phenology status for the cohort ('0' = leaf-off, '1' = leaf-on). Optional; when omitted, a cold-start default is used.
- restart_layer** Restart canopy layer for the cohort. Optional; when omitted, cohorts are relay-ered as in a cold start.
- restart_firstlayer** Restart flag indicating whether the cohort has previously occupied the top layer. Optional.
- restart_gdd** Restart growing degree days stored on the cohort. Optional.
- restart_leaf_age** Restart leaf age, in years. Optional.
- restart_topyear** Restart count of years spent in the top canopy layer. Optional.
- restart_bl_max** Restart target leaf biomass ('bl_max'), in kg C per individual. Optional.
- restart_br_max** Restart target fine-root biomass ('br_max'), in kg C per individual. Optional.
- init_soil** A data.frame of initial soil pools, including the following data:
- init_fast_soil_C** Initial fast soil carbon, in kg C m⁻².
- init_slow_soil_C** Initial slow soil carbon, in kg C m⁻².
- init_Nmineral** Mineral nitrogen pool, in kg N m⁻².
- N_input** Annual nitrogen input to soil N pool, in kg N m⁻² yr⁻¹.
- init_fast_soil_N** Initial fast soil nitrogen, in kg N m⁻². (optional, defaults to C biomass * CN-ratio)
- init_slow_soil_N** Initial slow soil nitrogen, in kg N m⁻². (optional, defaults to C biomass * CN-ratio)
- init_pmier_C** Initial carbon in soil microbial biomass, in kg C m⁻². (optional, defaults to 0.0)
- init_pmier_d13C** Initial d13C signature in soil microbial biomass, in permil. (optional, defaults to -9999)
- init_pmier_N** Initial nitrogen in soil microbial biomass, in kg N m⁻². (optional, defaults to 0.0)

- init_wcl1** Initial volumetric water content of soil layer 1, in m^3m^{-3} . (optional, defaults to field capacity)
- init_wcl2** Initial volumetric water content of soil layer 2, in m^3m^{-3} . (optional, defaults to field capacity)
- init_wcl3** Initial volumetric water content of soil layer 3, in m^3m^{-3} . (optional, defaults to field capacity)
- init_N0_ecosystem** Initial total amount of nitrogen in ecosystem (only used for nitrogen workaround), in kg N m^{-2} . (optional, defaults to sum of soil and plant pools.) Might not be needed.
- restart_tk_pheno** Restart smoothed phenology temperature, in Kelvin. Optional. ###
- restart_vegn_gdd** Restart tile-level growing degree days. Optional.
- init_lu** A data.frame of initial land unit (LU) specifications, including the following data:
- fraction** Initial grid cell fraction occupied by this LU, dimensionless (0 to 1) or m^{-2} LU area per m^{-2} grid cell area. The sum of all fractions is typically equal to 1, but may be less in which case the difference is the fraction of the grid cell occupied by ice/water.
- preset** Predefined land use type (optional). One of: 'unmanaged', 'urban', 'cropland', 'pasture'. See below for meaning of these presets. Leave empty to not use any preset.
- vegetated** Whether this LU accepts vegetation. Default for preset 'urban': False, default for other presets: True.
- extra_N_input** Additional inorg N supply (to account for N fertiliser application), in $\text{kg m}^{-2}\text{yr}^{-1}$. Default for preset 'cropland': 0.01, default other presets: 0.
- extra_turnover_rate** Additional soil turnover rate (to account for soil management such as tillage), dimensionless. Default for preset 'cropland': 0.2, default for other presets: 0.
- oxidized_litter_fraction** Fraction of above-ground turnover that is directly oxidized (crop and grass harvest), dimensionless. Default for preset 'cropland': 0.9, default for preset 'pasture': 0.4, default for other presets: 0.
- luc_forcing** Array of land use change (LUC) used during transient phase. During spinup, the initial land unit fractions are used (i.e. no transition). If there are more transient years than provided LUC data, the last state is maintained until the end of the transient phase (i.e. no transition). The array is a $n \times n$ square matrix, where n is the number of LU (i.e. dimension of `init_lu`). Each entry $f(i, j)$ expresses the grid cell fraction of LU i (row) being transferred to LU j (column). I.e. same units as `init_lu$fraction`. Self transitions are allowed, meaning that a part of the land unit is clear cut, but the area remains in the same land use.

biomee_gs_leuning_output

rsofun BiomeE (gs_leuning) output data

Description

Example output dataset from a BiomeE-model run using divers [biomee_gs_leuning_drivers](#) See [runread_biomee_f](#) and [run_biomee_f_bysite](#) for a detailed description of the outputs.

Usage

biomee_gs_leuning_output

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1 rows and 2 columns.

```
biomee_p_model_drivers
```

rsofun BiomeE driver data (P-model photosynthesis model)

Description

Example driver data to run the BiomeE-model at the CH-LAE site using the P-model photosynthesis specification (and daily time step). It can also be used together with leaf trait data from CH-LAE ([biomee_validation](#)) to optimize model parameters.

Usage

```
biomee_p_model_drivers
```

Format

See [biomee_gs_leuning_drivers](#)

```
biomee_p_model_luluc_drivers
```

rsofun BiomeE driver data (P-model photosynthesis model) with LU-LUC

Description

Example driver data to run the BiomeE-model at the CH-LAE site using the P-model photosynthesis specification (and daily time step). It provides an example of land use change (LUC).

Usage

```
biomee_p_model_luluc_drivers
```

Format

See [biomee_gs_leuning_drivers](#)

```
biomee_p_model_luluc_output
      rsofun BiomeE (P-model) output data
```

Description

Example output dataset from a BiomeE-model run using divers [biomee_p_model_luluc_drivers](#) See [runread_biomee_f](#) and [run_biomee_f_bysite](#) for a detailed description of the outputs.

Usage

```
biomee_p_model_luluc_output
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1 rows and 4 columns.

```
biomee_p_model_output  rsofun BiomeE (P-model) output data
```

Description

Example output dataset from a BiomeE-model run using divers [biomee_p_model_drivers](#) See [runread_biomee_f](#) and [run_biomee_f_bysite](#) for a detailed description of the outputs.

Usage

```
biomee_p_model_output
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1 rows and 2 columns.

biomee_validation	<i>rsofun BiomeE validation data</i>
-------------------	--------------------------------------

Description

Small example dataset of target observations (leaf trait data) at the CH-LAE site to optimize model parameters with the function [calib_sofun_legacy](#)

Usage

```
biomee_validation
```

Format

A tibble of validation data:

sitename site name

data validation data

Source

Lukas Hörtnagl, Werner Eugster, Nina Buchmann, Eugenie Paul-Limoges, Sophia Etzold, Matthias Haeni, Peter Pluess, Thomas Baur (2004-2014) FLUXNET2015 CH-Lae Laegern, Dataset. <https://doi.org/10.18140/FLX/14>

build_luc_matrix	<i>Build LUC matrix</i>
------------------	-------------------------

Description

Build land-use change (LUC) transition matrix from patterns.

Usage

```
build_luc_matrix(patterns, n_lu, n_years, out = vector())
```

Arguments

patterns	A list of patterns. Each pattern must be a sequence of transition values whose size is a multiple of <code>n_luxn_lu</code> . The <code>n_lu</code> first values are the transitions from each LU to the first LU, and so on. If the sequence contains more years than <code>n_years</code> , it will be truncated.
n_lu	Number of land use types (LU).
n_years	Number of years (i.e. length of the 3rd dimension).
out	For internal use only.

Value

An `n_luxn_luxn_years` transition matrix.

Examples

```
# Example of building a 6 year-long transition matrix consisting of 6 times 2x2 matrices

# A one time transfer of 0.5 of the total cell fraction from LU 2 to LU 1
pattern1 <- c(0, 0, 0.5, 0)
# The null pattern (no transition)
null_pattern <- rep(0, 4)
# A repeated self-transition of 0.1 of the total cell fraction from LU 2 to LU 2 every other year
pattern2 <- rep(c(0, 0, 0, 0.1), null_pattern), 3)

# Building the transition matrix
build_luc_matrix(list(pattern1, pattern2), 2, 6)
```

calib_sofun

Calibrates SOFUN model parameters (for parallelized applications)

Description

Runs the requested model calibration and returns results

Usage

```
calib_sofun(
  drivers,
  obs,
  settings_calib,
  optim_out = TRUE,
  logpath = file.path(tempdir(), paste0("out_calib-", "my_calibration_name",
    ".rds.log.txt")),
  ...
)
```

Arguments

<code>drivers</code>	A data frame with driver data. See pmodel_drivers for a description of the data structure. Additional columns can optionally be provided to <code>drivers</code> to control e.g. the processing within a personalized cost function.
<code>obs</code>	A data frame containing observational data used for model calibration. See pmodel_validation for a description of the data structure. Additional columns can optionally be provided to <code>obs</code> to control e.g. the processing within a personalized cost function.
<code>settings_calib</code>	A list containing model calibration settings. See the 'P-model usage' vignette for more information and examples.

	method	A string indicating the optimization method, either 'GenSA' or 'BayesianTools'.
	par	A list of model parameters. For each parameter, an initial value and lower and upper bounds should be provided. The calibratable parameters include model parameters 'kphio', 'kphio_par_a', 'kphio_par_b', 'soilm_thetastar', 'soilm_betao', 'beta_costunitratio', 'rd_to_vcmax', 'tau_acclim', 'kc_jmax' and 'rootzone_whc', and (if doing Bayesian calibration) error parameters for each target variable, named for example 'err_gpp'. This list must match the input parameters of the calibration metric and the parameters should be given in the order above.
	metric	A cost function. See the 'Cost functions for parameter calibration' vignette for examples.
	control	A list of arguments passed on to the optimization function. If method = 'GenSA', see GenSA . If method = 'BayesianTools' the list should include at least <code>sampler_runMCMC</code> and <code>settings_runMCMC</code> , see BayesianTools::runMCMC . For method = 'BayesianTools', <code>n_parallel_nrChains</code> can optionally be used to specify the number of cores to be used for parallel sampling of chains. (If ≥ 2 it overwrites <code>settings_runMCMC\$nrChains</code> , if 0 or 1 parallel sampling is deactivated).
optim_out		A logical indicating whether the function returns and stores the raw output of the optimization functions (defaults to TRUE).
logpath		A path where a log-file is created and progress logged for parallel sampling, defaults to a temporary directory. The path is stored in the output object as simulation 'name' and 'logpath'. (Note that this argument is only used when parallelized sampling is requested.)
...		Optional arguments, simply passed on to the cost function.

Details

This is the main function that handles the calibration of SOFUN model parameters.

Value

A named list containing the calibrated parameter vector 'par', the output object from the optimization 'mod', and additional identifier outputs 'name', 'logpath', and 'walltime'. For more details on the 'mod' output and how to evaluate it, see [runMCMC](#) (also [this post](#)) and [GenSA](#). If parallel (`n_parallel_nrChains >= 2`) also a logfile is created at 'logpath'.

Examples

```
# Calib 1: use Bayesian calibration approach with likelihood:
# Define priors of model parameters that will be calibrated
params_to_estimate <- list(
  kphio          = list(lower = 0.02, upper = 0.15, init = 0.05),
  err_gpp        = list(lower = 0.01, upper = 3, init = 0.8),
  err_bigD13C    = list(lower = 0.5, upper = 4.0, init = 2.0)
)
# Fix model parameters that won't be calibrated
params_fix       <- list(
  kphio_par_a    = 0,
```

```

    kphio_par_b      = 25,
    soilm_thetastar  = 0.6 * 240,
    soilm_betao      = 0.01,
    beta_unitcostratio = 146,
    rd_to_vcmax      = 0.014,
    tau_acclim       = 30,
    kc_jmax          = 0.41
  )
  # Define calibration settings
  settings_calib <- list(
    method = "BayesianTools",
    par     = params_to_estimate,
    metric  = rsfun::cost_likelihood_pmodel,
    control = list(
      sampler_runMCMC = "DEzs",
      settings_runMCMC = list(
        burnin = 0,
        iterations = 10,      # kept artificially low
        nrChains = 1          # 2, # these are the independent chains
      ),
      n_parallel_nrChains = 1 # 2, this can be parallelized
    )
  )
  # Run the calibration for GPP and D13C data
  drivers_to_use <- rsfun::pmodel_drivers |>
    dplyr::filter(sitename %in% c("FR-Pue", "lon_+146.13_lat_-032.97"))
  obs_to_use <- rsfun::pmodel_validation |>
    dplyr::filter(sitename %in% c("FR-Pue", "lon_+146.13_lat_-032.97"))
  calib_output <- rsfun::calib_sofun(
    drivers = drivers_to_use,
    obs     = obs_to_use,
    settings_calib = settings_calib,
    # extra arguments for the cost function
    targets = c("gpp" = "fluxnet", "bigD13C" = "cornwell"),
    par_fixed = params_fix
  )
  calib_output$mod      # BayesianTools::mcmcSamplerList
  calib_output$par      # Named vector
  calib_output$walltime # Benchmarked time
  calib_output$runtime  # unused
  calib_output$name     # optionally used calibration name
  calib_output$fpath    # path of rds output

  # Calib 2: use GenSa optimization of RMSE
  # Calibrate the model and optimize the free parameters using demo datasets
  settings_calib_rmse <- list(
    method = "GenSA",          # minimizes the RMSE
    metric = cost_rmse_pmodel, # our cost function returning the RMSE
    control = list( # control parameters for optimizer GenSA
      maxit = 2),
    par = list( # bounds for the parameter space
      beta_unitcostratio = list(lower = 100, upper = 200, init = 130.0)
    )
  )

```

```

)
drivers_to_use <- rsofun::pmodel_drivers |>
  dplyr::filter(sitename %in% c("lon_+146.13_lat_-032.97"))
obs_to_use <- rsofun::pmodel_validation |>
  dplyr::filter(sitename %in% c("lon_+146.13_lat_-032.97"))
pars_calib_rmse <- calib_sofun(
  # calib_sofun arguments:
  drivers = drivers_to_use,
  obs = obs_to_use,
  settings_calib = settings_calib_rmse,
  # extra arguments passed to the cost function:
  targets = c("bigD13C" = "cornwall"),
  par_fixed = list( # fix all other parameters
    kphio = 0.09, # set to zero to disable temperature-dependence
    kphio_par_a = 0.0, # set to zero to disable temperature-dependence
    # of kphio, setup ORG
    kphio_par_b = 1.0,
    soilm_thetastar = 0.6 * 240, # to recover paper setup with soil moisture stress
    soilm_betao = 0.0,
    rd_to_vcm_max = 0.014, # value from Atkin et al. 2015 for C3 herbaceous
    tau_acclim = 30.0,
    kc_jmax = 0.41
  )
)

```

calib_sofun_legacy	<i>Calibrates SOFUN model parameters</i>
--------------------	--

Description

This is the main function that handles the calibration of SOFUN model parameters.

Usage

```
calib_sofun_legacy(drivers, obs, settings, optim_out = TRUE, ...)
```

Arguments

drivers	A data frame with driver data. See pmodel_drivers for a description of the data structure. Additional columns can optionally be provided to drivers to control e.g. the processing within a personalized cost function.
obs	A data frame containing observational data used for model calibration. See pmodel_validation for a description of the data structure. Additional columns can optionally be provided to obs to control e.g. the processing within a personalized cost function.
settings	A list containing model calibration settings. See the 'P-model usage' vignette for more information and examples.
method	A string indicating the optimization method, either 'GenSA' or 'BayesianTools'.

<code>par</code>	A list of model parameters. For each parameter, an initial value and lower and upper bounds should be provided. The calibratable parameters include model parameters 'kphio', 'kphio_par_a', 'kphio_par_b', 'soilm_thetastar', 'soilm_betao', 'beta_costunitratio', 'rd_to_vcmax', 'tau_acclim', 'kc_jmax' and 'rootzone_whc', and (if doing Bayesian calibration) error parameters for each target variable, named for example 'err_gpp'. This list must match the input parameters of the calibration metric and the parameters should be given in the order above.
<code>metric</code>	A cost function. See the 'Cost functions for parameter calibration' vignette for examples.
<code>control</code>	A list of arguments passed on to the optimization function. If method = 'GenSA', see GenSA . If method = 'BayesianTools' the list should include at least settings and sampler, see BayesianTools::runMCMC .
<code>optim_out</code>	A logical indicating whether the function returns the raw output of the optimization functions (defaults to TRUE).
<code>...</code>	Optional arguments, simply passed on to the cost function.

Value

A named list containing the calibrated parameter vector 'par' and the output object from the optimization 'mod'. For more details on this output and how to evaluate it, see [runMCMC](#) (also [this post](#)) and [GenSA](#).

Examples

```
# Fix model parameters that won't be calibrated
params_fix <- list(
  kphio_par_a      = 0,
  kphio_par_b      = 1.0,
  soilm_thetastar  = 0.6 * 240,
  soilm_betao      = 0.01,
  beta_unitcostratio = 146,
  rd_to_vcmax      = 0.014,
  tau_acclim       = 30,
  kc_jmax          = 0.41
)

# Define calibration settings
settings <- list(
  method = "BayesianTools",
  par = list(
    kphio = list(lower = 0.04, upper = 0.09, init = 0.05),
    err_gpp = list(lower = 0.01, upper = 4, init = 2)
  ),
  metric = rsofun::cost_likelihood_pmodel,
  control = list(
    sampler = "DEzs",
    settings = list(
      nrChains = 1,
      burnin = 0,

```

```

        iterations = 50      # kept artificially low
      )
    )
  )

# Run the calibration for GPP data
calib_output <- rsofun::calib_sofun_legacy(
  drivers = rsofun::pmodel_drivers |> dplyr::filter(sitename == "FR-Pue"),
  obs     = rsofun::pmodel_validation |> dplyr::filter(sitename == "FR-Pue"),
  settings = settings,
  # extra arguments for the cost function
  targets = c("gpp" = "fluxnet"),
  par_fixed = params_fix
)

```

cost_likelihood_biomee

Log-likelihood cost function for BiomeE with different targets

Description

Cost function for parameter calibration, which computes the log-likelihood for the biomee model fitting several target variables for a given set of parameters.

Usage

```
cost_likelihood_biomee(par, obs, drivers, targets)
```

Arguments

par	A named vector containing parameter values for 'phiRL', 'LAI_light', 'tf_base', 'par_mort' in that order, and for the error terms corresponding to the target variables, e.g. 'err_GPP' if GPP is a target. Make sure that the order of the error terms in par coincides with the order provided in the targets argument.
obs	A nested data frame of observations, following the structure of biomee_validation, for example.
drivers	A nested data frame of driver data, for example biomee_gs_leuning_drivers.
targets	A character vector indicating the target variables for which the optimization will be done. This should be a subset of c("GPP", "LAI", "Density", "Biomass").

Details

The cost function performs a BiomeE model run for the value of par given as argument. The likelihood is calculated assuming that the predicted targets are independent, normally distributed and centered on the observations. The optimization should be run using BayesianTools, so the likelihood is maximized.

Value

The log-likelihood of the simulated targets by the biomee model versus the observed targets.

Examples

```
# do not run long-running simulations
# Compute the likelihood for a set of
# BiomeE model parameter values
# and the example data
cost_likelihood_biomee(
  par = c(phiRL = 3.5,
    LAI_light = 3.5,
    tf_base = 1,
    par_mort = 1,      # model params
    err_GPP = 0.5),    # err_GPP
  obs = biomee_validation,
  drivers = biomee_p_model_drivers,
  targets = c("GPP")
)
```

```
cost_likelihood_pmodel
```

Cost function computing a log-likelihood for calibration of P-model parameters

Description

The cost function performs a P-model run for the input drivers and model parameter values, and computes the outcome's normal log-likelihood centered at the input observed values and with standard deviation given as an input parameter (calibratable).

Usage

```
cost_likelihood_pmodel(
  par,
  obs,
  drivers,
  targets,
  par_fixed = NULL,
  parallel = FALSE,
  ncores = 1,
  get_mod_obs = get_mod_obs_pmodel
)
```

Arguments

<code>par</code>	A named list for the parameters to be calibrated, including a subset of model parameters and error terms for each target variable (for example 'gpp_err').
<code>obs</code>	A nested data.frame of observations, with columns 'sitename' and 'data' (see pmodel_validation to check its structure).
<code>drivers</code>	A nested data.frame of driver data. See pmodel_drivers for a description of the data structure.
<code>targets</code>	A named character vector indicating the target variable(s) for which the likelihood is computed. The names of 'target' indicate target variables, while the values indicate the source data set to be used. The values must be present in the 'obs\$source' column. The names must be column name(s) of the 'obs\$data' data.frame and they must have error term(s) in either 'par' or 'par_fixed'.
<code>par_fixed</code>	A named list of model parameter values to keep fixed during the calibration. These should complement the input <code>par</code> such that all model parameters are uniquely defined and passed on to runread_pmodel_f .
<code>parallel</code>	A logical specifying whether simulations are to be parallelised (sending data from a certain number of sites to each core). Defaults to FALSE.
<code>ncores</code>	An integer specifying the number of cores used for parallel computing. Defaults to 2.
<code>get_mod_obs</code>	Function running P-model and combining model outputs and observations (defaults to <code>get_mod_obs_pmodel</code>)

Details

To run the P-model, all model parameters must be given. The cost function uses arguments `par` and `par_fixed` such that, in the calibration routine, `par` can be updated by the optimizer and `par_fixed` are kept unchanged throughout calibration.

If the validation data contains a "date" column (fluxes), the simulated target time series is compared to the observed values on those same dates (e.g. for GPP).

Value

The log-likelihood of the observed target values, assuming that they are independent, normally distributed and centered on the predictions made by the P-model run with standard deviation given as input `par` e.g. `gpp_err`.

Examples

```
# Compute the likelihood for a set of
# model parameter values involved in the
# temperature dependence of kphio
# and example data
cost_likelihood_pmodel(
  par = c(kphio      = 0.05,
          kphio_par_a = -0.01,
          kphio_par_b = 1,
          # error model parameters
```

```

    err_gpp      = 2),
  obs = pmodel_validation |> dplyr::filter(sitename == "FR-Pue"),
  drivers = pmodel_drivers |> dplyr::filter(sitename == "FR-Pue"),
  targets = c("gpp" = "fluxnet"),
  par_fixed = list(
    soilm_thetastar = 0.6 * 240, # old setup with soil moisture stress
    soilm_betao     = 0.0,
    beta_unitcostratio = 146.0,
    rd_to_vcmax      = 0.014,    # from Atkin et al. 2015 for C3 herbaceous
    tau_acclim       = 30.0,
    kc_jmax          = 0.41
  )
)

```

cost_likelihood_pmodel_legacy

Cost function computing a log-likelihood for calibration of P-model parameters

Description

The cost function performs a P-model run for the input drivers and model parameter values, and computes the outcome's normal log-likelihood centered at the input observed values and with standard deviation given as an input parameter (calibratable).

Usage

```

cost_likelihood_pmodel_legacy(
  par,
  obs,
  drivers,
  targets,
  par_fixed = NULL,
  parallel = FALSE,
  ncores = 2
)

```

Arguments

par	A named vector of values for the parameters to be calibrated, including a subset of model parameters (described in runread_pmodel_f), in order, and error terms for each target variable (for example 'gpp_err'), in the same order as the targets appear in targets.
obs	A nested data.frame of observations, with columns 'sitename' and 'data' (see pmodel_validation to check its structure).
drivers	A nested data.frame of driver data. See pmodel_drivers for a description of the data structure.

targets	A character vector indicating the target variables for which the likelihood is computed. The names must be column name(s) of the 'obs\$data' data.frame and they must have error term(s) in either 'par' or 'par_fixed'.
par_fixed	A named list of model parameter values to keep fixed during the calibration. These should complement the input par such that all model parameters are passed on to <code>runread_pmodel_f</code> .
parallel	A logical specifying whether simulations are to be parallelised (sending data from a certain number of sites to each core). Defaults to FALSE.
ncores	An integer specifying the number of cores used for parallel computing. Defaults to 2.

Details

To run the P-model, all model parameters must be given. The cost function uses arguments `par` and `par_fixed` such that, in the calibration routine, `par` can be updated by the optimizer and `par_fixed` are kept unchanged throughout calibration.

If the validation data contains a "date" column (fluxes), the simulated target time series is compared to the observed values on those same dates (e.g. for GPP). Otherwise, there should only be one observed value per site (leaf traits), and the outputs (averaged over the growing season, weighted by predicted GPP) will be compared to this single value representative of the site (e.g. `Vcmax25`). As an exception, when the date of a trait measurement is available, it will be compared to the trait value predicted on that date.

Value

The log-likelihood of the observed target values, assuming that they are independent, normally distributed and centered on the predictions made by the P-model run with standard deviation given as input (via 'par' because the error terms are estimated through the calibration with 'BayesianTools', as shown in the "Parameter calibration and cost functions" vignette).

Examples

```
# Compute the likelihood for a set of
# model parameter values involved in the
# temperature dependence of kphio
# and example data
cost_likelihood_pmodel_legacy(
  par = c(kphio      = 0.05,
          kphio_par_a = -0.01,
          kphio_par_b = 1,      # model parameters
          err_gpp     = 2),     # err_gpp
  obs = pmodel_validation |> dplyr::filter(sitename == "FR-Pue"),
  drivers = pmodel_drivers |> dplyr::filter(sitename == "FR-Pue"),
  targets = c("gpp"), # NOTE: the old legacy format of targets
  par_fixed = list(
    soilm_thetastar = 0.6 * 240, # old setup with soil moisture stress
    soilm_betao     = 0.0,
    beta_unitcostratio = 146.0,
    rd_to_vcmax      = 0.014,    # from Atkin et al. 2015 for C3 herbaceous
```

```

    tau_acclim      = 30.0,
    kc_jmax         = 0.41
  )
)
cost_likelihood_pmodel_legacy(
  par = c(kphio      = 0.05,
    kphio_par_a = -0.01,
    kphio_par_b = 1,      # model parameters
    err_le      = 2,
    err_gpp     = 2,
    err_bigD13C = 2),
  obs = pmodel_validation,
  drivers = pmodel_drivers,
  targets = c("gpp", "le", "bigD13C"), # NOTE: the old legacy format of targets
  par_fixed = list(
    soilm_thetastar = 0.6 * 240, # old setup with soil moisture stress
    soilm_betao     = 0.0,
    beta_unitcostratio = 146.0,
    rd_to_vcmmax    = 0.014,      # from Atkin et al. 2015 for C3 herbaceous
    tau_acclim      = 30.0,
    kc_jmax         = 0.41
  )
)

```

cost_rmse_biomee	<i>RMSE cost function for BiomeE</i>
------------------	--------------------------------------

Description

Cost function for parameter calibration, which computes the root mean squared error (RMSE) between BiomeE simulations (using the input set of parameters) and observed target variables. Cost function for parameter calibration, which computes the RMSE for the biomee model fitting target variables 'GPP', 'LAI', 'Density' and 'Biomass' for a given set of parameters.

Usage

```
cost_rmse_biomee(par, obs, drivers)
```

Arguments

par	A vector containing parameter values for 'phiRL', 'LAI_light', 'tf_base', 'par_mort' in that order.
obs	A nested data frame of observations, following the structure of biomee_validation, for example.
drivers	A nested data frame of driver data, for example biomee_gs_leuning_drivers.

Value

The root mean squared error (RMSE) between the observed and simulated values of 'GPP', 'LAI', 'Density' and 'Biomass' (all variables have the same weight). Relative errors (difference divided by observed values) are used instead of absolute errors. The cost function performs a BiomeE model run for parameter values `par` and model drivers `drivers` given as arguments, producing the simulated values used to compute the RMSE.

Examples

```
# do not run long-running simulations
# Compute RMSE for a set of
# model parameter values
# and example data
cost_rmse_biomee(
  par = c(3.5, 3.5, 1, 1),
  obs = biomee_validation,
  drivers = biomee_p_model_drivers
)
```

code
cost_rmse_pmodel

Cost function computing RMSE for calibration of P-model parameters

Description

The cost function performs a P-model run for the input drivers and parameter values, and compares the output to observations of various targets by computing the root mean squared error (RMSE).

Usage

```
cost_rmse_pmodel(
  par,
  obs,
  drivers,
  targets,
  par_fixed = NULL,
  target_weights = NULL
)
```

Arguments

<code>par</code>	A vector of values for the parameters to be calibrated (a subset of those described in runread_pmodel_f , in order).
<code>obs</code>	A nested data.frame of observations, with columns 'sitename' and 'data' (see pmodel_validation to check its structure).
<code>drivers</code>	A nested data.frame of driver data. See pmodel_drivers for a description of the data structure.

targets	A named character vector indicating the target variable(s) for which the RMSE is computed. The names of 'target' indicate target variables, while the values indicate the source data set to be used. The values must be present in the 'obs\$source' column. The names must be column name(s) of the 'obs\$data' data.frame.
par_fixed	A named list of model parameter values to keep fixed during the calibration. These should complement the input par such that all model parameters are passed on to runread_pmodel_f .
target_weights	A vector of weights to be used in the computation of the RMSE if using several targets. By default (target_weights = NULL) the RMSE is computed separately for each target and then averaged. The provided weights are used to compute a weighted average of RMSE across targets.

Details

To run the P-model, all model parameters must be given. The cost function uses arguments par and par_fixed such that, in the calibration routine, par can be updated by the optimizer and par_fixed are kept unchanged throughout calibration.

If the validation data contains a "date" column (fluxes), the simulated target time series is compared to the observed values on those same dates (e.g. for GPP). Otherwise, there should only be one observed value per site (leaf traits), and the outputs (averaged over the growing season, weighted by predicted GPP) will be compared to this single value representative of the site (e.g. Vcmax25). As an exception, when the date of a trait measurement is available, it will be compared to the trait value predicted on that date.

Value

The root mean squared error (RMSE) between observed values and P-model predictions. The RMSE is computed for each target separately and then aggregated (mean or weighted average).

Examples

```
# Compute RMSE for a set
# of model parameter values
# and example data
cost_rmse_pmodel(
  par = c(0.05, -0.01, 0.5), # kphio related parameters
  obs = pmodel_validation |> dplyr::filter(sitename == "FR-Pue"),
  drivers = pmodel_drivers |> dplyr::filter(sitename == "FR-Pue"),
  targets = c("gpp" = "fluxnet"),
  par_fixed = list(
    soilm_thetastar = 0.6 * 240, # old setup with soil moisture stress
    soilm_betao = 0.0,
    beta_unitcostratio = 146.0,
    rd_to_vcmax = 0.014, # from Atkin et al. 2015 for C3 herbaceous
    tau_acclim = 30.0,
    kc_jmax = 0.41
  )
)
```

createMixedPrior	<i>Internal convenience function to create a mixed prior distribution</i>
------------------	---

Description

Similar to BayesianTools::createUniformPrior()

Usage

```
createMixedPrior(prior_definitions)
```

Arguments

prior_definitions
A named list with parameters for a distribution

Value

TRUE or FALSE, if the provided parameters describe a beta distribution

Examples

```
library(BayesianTools)
prior_definitions_mixed <- list(
  par = list(
    par_uniform      = list(lower = 10, upper = 30, init = 15),
    par_normal       = list(mean = 10, sd = 2),
    par_lognormal    = list(meanlog = -4, sdlog = 1.1),
    par_truncnormal  = list(mean = 10, sd = 2, lower = 9, upper = 14),
    # par_trunclognormal = list(meanlog = -4, sdlog = 1.1, endpoint = 0.5),
    par_beta         = list(shape1 = 5, shape2 = 2)
  )
)
priorMixed <- rsfun::createMixedPrior(prior_definitions_mixed$par)
priorMixed$sampler(2000)
hist(priorMixed$sampler(2000)[, 5])
```

get_mod_obs_pmodel	<i>Function running P-model and combining model and observations</i>
--------------------	--

Description

The get_mod_obs function performs a P-model run for the input drivers and model parameter values, and collects and combines the model output with the observations. It provides output for all possible targets (defined by the error terms provided in params_modl_and_err) and must be subset for requested targets by the calling cost_function.

Usage

```
get_mod_obs_pmodel(
  drivers,
  obs,
  params_modl_and_err,
  parallel,
  ncores,
  return_continuous_timeseries = FALSE
)
```

Arguments

drivers	A nested data.frame of driver data. See pmodel_drivers for a description of the data structure.
obs	A nested data.frame of observations, see pmodel_validation and cost_likelihood_pmodel for details.
params_modl_and_err	A named list for the model parameters and error terms for each target variable, see cost_likelihood_pmodel for details.
parallel	A logical specifying whether simulations are to be parallelised (sending data from a certain number of sites to each core).
ncores	An integer specifying the number of cores used for parallel computing.
return_continuous_timeseries	A logical specifying whether modelled values are returned as daily, continuous time series (including days without observations) or whether modelled values are only returned when a corresponding observation is available.

init_dates_dataframe *Initialises a tibble with dates*

Description

Creates a tibble with rows for each date from 'yrstart' to 'yrend' in 'yyyy-mm-dd' format. Intervals of dates are specified by argument 'freq'. `ddf <- init_dates_dataframe(2000, 2003, startmoy=1, startdoy=1, freq="days", endmoy=12, enddom=31, noleap=FALSE)`

Usage

```
init_dates_dataframe(
  yrstart,
  yrend,
  startmoy = 1,
  startdoy = 1,
  freq = "days",
  endmoy = 12,
  enddom = 31,
  noleap = FALSE
)
```

Arguments

yrstart	An integer defining the start year of dates covered by the dataframe.
yrend	An integer defining the end year of dates covered by the dataframe.
startmoy	An integer defining the start month-of-year of dates covered by the dataframe. Defaults to 1.
startdoy	An integer defining the start day-of-year of dates covered by the dataframe. Defaults to 1.
freq	A character string specifying the time steps of dates (in rows). Defaults to "days". Any of "days", "months", "years". If freq = "months" the 15 th day of the months is used as date, and if freq = "years" the 1 st of January of each year is returned.
endmoy	An integer defining the end month-of-year of dates covered by the dataframe. Defaults to 12.
enddom	An integer defining the end day-of-year of dates covered by the dataframe. Defaults to 31.
noleap	Whether leap years are ignored, that is, whether the 29 th of February is removed. Defaults to FALSE.

Value

A tibble with dates.

is_beta_prior	<i>Checks if provided parameter list describes a beta distribution</i>
---------------	--

Description

Checks if provided parameter list describes a beta distribution

Usage

```
is_beta_prior(distr_pars)
```

Arguments

distr_pars	A named list with parameters for a distribution
------------	---

Value

TRUE or FALSE, if the provided parameters describe a beta distribution

Examples

```
prior_definitions_mixed <- list(
  par = list(
    par_uniform      = list(lower = 10, upper = 30, init = 15),
    par_normal       = list(mean = 10, sd = 2),
    par_lognormal    = list(meanlog = -4, sdlog = 1.1),
    par_truncnormal  = list(mean = 10, sd = 2, lower = 9, upper = 14),
    par_trunclognormal = list(meanlog = -4, sdlog = 1.1, endpoint = 2.2),
    par_beta         = list(shape1 = 5, shape2 = 2)
  )
)
lapply(prior_definitions_mixed$par, rsofun::is_beta_prior)
```

is_lognormal_prior	<i>Checks if provided parameter list describes a lognormal distribution</i>
--------------------	---

Description

Checks if provided parameter list describes a lognormal distribution

Usage

```
is_lognormal_prior(distr_pars)
```

Arguments

distr_pars A named list with parameters for a distribution

Value

TRUE or FALSE, if the provided parameters describe a lognormal distribution

Examples

```
prior_definitions_mixed <- list(
  par = list(
    par_uniform      = list(lower = 10, upper = 30, init = 15),
    par_normal       = list(mean = 10, sd = 2),
    par_lognormal    = list(meanlog = -4, sdlog = 1.1),
    par_truncnormal  = list(mean = 10, sd = 2, lower = 9, upper = 14),
    par_trunclognormal = list(meanlog = -4, sdlog = 1.1, endpoint = 2.2),
    par_beta         = list(shape1 = 5, shape2 = 2)
  )
)
lapply(prior_definitions_mixed$par, rsofun::is_lognormal_prior)
```

is_normal_prior	<i>Checks if provided parameter list describes a normal distribution</i>
-----------------	--

Description

Checks if provided parameter list describes a normal distribution

Usage

```
is_normal_prior(distr_pars)
```

Arguments

distr_pars A named list with parameters for a distribution

Value

TRUE or FALSE, if the provided parameters describe a normal distribution

Examples

```
prior_definitions_mixed <- list(
  par = list(
    par_uniform      = list(lower = 10, upper = 30, init = 15),
    par_normal       = list(mean = 10, sd = 2),
    par_lognormal    = list(meanlog = -4, sdlog = 1.1),
    par_truncnormal  = list(mean = 10, sd = 2, lower = 9, upper = 14),
    par_trunclognormal = list(meanlog = -4, sdlog = 1.1, endpoint = 2.2),
    par_beta         = list(shape1 = 5, shape2 = 2)
  )
)
lapply(prior_definitions_mixed$par, rsofun::is_normal_prior)
```

is_trunclognormal_prior	<i>Checks if provided parameter list describes a truncated lognormal distribution</i>
-------------------------	---

Description

Checks if provided parameter list describes a truncated lognormal distribution

Usage

```
is_trunclognormal_prior(distr_pars)
```

Arguments

distr_pars A named list with parameters for a distribution

Value

TRUE or FALSE, if the provided parameters describe a truncated lognormal distribution

Examples

```
prior_definitions_mixed <- list(
  par = list(
    par_uniform      = list(lower = 10, upper = 30, init = 15),
    par_normal       = list(mean  = 10, sd   = 2),
    par_lognormal    = list(meanlog = -4, sdlog = 1.1),
    par_truncnormal  = list(mean  = 10, sd   = 2, lower = 9, upper = 14),
    par_trunclognormal = list(meanlog = -4, sdlog = 1.1, endpoint = 2.2),
    par_beta         = list(shape1 = 5, shape2 = 2)
  )
)
lapply(prior_definitions_mixed$par, rsofun::is_trunclognormal_prior)
```

is_truncnormal_prior	<i>Checks if provided parameter list describes a truncated normal distribution</i>
----------------------	--

Description

Checks if provided parameter list describes a truncated normal distribution

Usage

```
is_truncnormal_prior(distr_pars)
```

Arguments

distr_pars A named list with parameters for a distribution

Value

TRUE or FALSE, if the provided parameters describe a truncated normal distribution

Examples

```
prior_definitions_mixed <- list(
  par = list(
    par_uniform      = list(lower = 10, upper = 30, init = 15),
    par_normal       = list(mean  = 10, sd   = 2),
    par_lognormal    = list(meanlog = -4, sdlog = 1.1),
    par_truncnormal  = list(mean  = 10, sd   = 2, lower = 9, upper = 14),
    par_trunclognormal = list(meanlog = -4, sdlog = 1.1, endpoint = 2.2),
    par_beta         = list(shape1 = 5, shape2 = 2)
  )
)
lapply(prior_definitions_mixed$par, rsofun::is_truncnormal_prior)
```

is_uniform_prior	<i>Checks if provided parameter list describes a uniform distribution</i>
------------------	---

Description

Checks if provided parameter list describes a uniform distribution

Usage

```
is_uniform_prior(distr_pars)
```

Arguments

distr_pars A named list with parameters for a distribution

Value

TRUE or FALSE, if the provided parameters describe a uniform distribution

Examples

```
prior_definitions_mixed <- list(
  par = list(
    par_uniform      = list(lower = 10, upper = 30, init = 15),
    par_normal       = list(mean  = 10, sd   = 2),
    par_lognormal    = list(meanlog = -4, sdlog = 1.1),
    par_truncnormal  = list(mean    = 10, sd    = 2, lower = 9, upper = 14),
    par_trunclognormal = list(meanlog = -4, sdlog = 1.1, endpoint = 2.2),
    par_beta         = list(shape1 = 5, shape2 = 2)
  )
)
lapply(prior_definitions_mixed$par, rsofun:::is_uniform_prior)
```

p_model_oldformat_drivers	<i>rsofun P-model driver data</i>
---------------------------	-----------------------------------

Description

Small dataset representing the driver to run the P-model at the FR-Pue site. Legacy data set. See details in the documentation of its replacement [pmodel_drivers](#)

Usage

```
p_model_oldformat_drivers
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1 rows and 4 columns.

p_model_oldformat_validation	<i>rsofun P-model GPP validation data</i>
------------------------------	---

Description

Small example dataset of target observations (daily GPP flux data) to optimize model parameters with the function [calib_sofun_legacy](#). Legacy data set. See details in the documentation of its replacement [pmodel_validation](#)

Usage

```
p_model_oldformat_validation
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1 rows and 2 columns.

pmodel_drivers	<i>rsofun P-model driver data</i>
----------------	-----------------------------------

Description

Small dataset representing the driver to run the P-model at the six Fluxnet sites and 5 sites with trait observations. It can be used together with daily GPP or ET flux time series data and trait data to optimize model parameters (see [pmodel_validation](#)).

Usage

```
pmodel_drivers
```

Format

A tibble of driver data:

sitename A character string containing the site name.

forcing A tibble of a time series of forcing climate data. Required parameters differ for daily simulations or for onestep acclimation. For daily simulations these include daily values of:

date Date of the observation in YYYY-MM-DD format.

temp Daytime average air temperature in °C.

vpd Daytime average vapour pressure deficit in Pa.

ppfd Photosynthetic photon flux density (PPFD) in $\text{mol m}^{-2} \text{s}^{-1}$. If all values are NA, it indicates that PPFD should be calculated by the SPLASH model and column `ccov`.

netrad Net radiation in W m^{-2} . WARNING: This is currently ignored as a model forcing.

patm Atmospheric pressure in Pa.
snow Snow in water equivalents mm s^{-1} .
rain Rain as precipitation in liquid form in mm s^{-1} .
tmin Daily minimum air temperature in $^{\circ}\text{C}$.
tmax Daily maximum air temperature in $^{\circ}\text{C}$.
wind Wind velocity (m s^{-1})
fapar Fraction of photosynthetic active radiation (fAPAR), taking values between 0 and 1.
co2 Atmospheric CO_2 concentration in ppm.
ccov Cloud coverage in %. This is only used when either PPFD or net radiation are not prescribed.

For onestep acclimation these include growing season averages (i.e. single values) of:

temp Daytime average air temperature in $^{\circ}\text{C}$.
vpd Daytime average vapour pressure deficit in Pa.
ppfd Photosynthetic photon flux density (PPFD) in $\text{mol m}^{-2} \text{s}^{-1}$.
patm Atmospheric pressure in Pa.
co2 Atmospheric CO_2 concentration in ppm.

params_siml A tibble of simulation parameters. Required parameters differ for daily simulations or for onestep acclimation. For daily simulations these include:

spinup A logical value indicating whether this simulation does spin-up.
spinupyears Number of spin-up years.
recycle Number of first N years of forcing data.frame that are recycled for spin-up.
outdt An integer indicating the output periodicity.
ltre A logical value, TRUE if evergreen tree.
ltne A logical value, TRUE if evergreen tree and N-fixing.
ltrd A logical value, TRUE if deciduous tree.
ltnd A logical value, TRUE if deciduous tree and N-fixing.
lgr3 A logical value, TRUE if grass with C3 photosynthetic pathway.
lgn3 A logical value, TRUE if grass with C3 photosynthetic pathway and N-fixing.
lgr4 A logical value, TRUE if grass with C4 photosynthetic pathway.
onestep A logical value, FALSE to request daily simulations. Defaults to FALSE if missing.

For onestep acclimation these include:

lc4 A logical value indicating whether to use C4 photosynthetic pathway.
onestep A logical value, TRUE to request onestep acclimation. Defaults to FALSE if missing.

site_info A tibble containing site meta information. This data structure can be freely used for documenting the dataset, but must include at least the following data:

lon Longitude of the site location in degrees east.
lat Latitude of the site location in degrees north.
elv Elevation of the site location, in meters above sea level.
whc A numeric value for the rooting zone water holding capacity (in mm). (Not needed if `onestep == "TRUE"`.)

Source

Pastorello, G., Trotta, C., Canfora, E. et al. The FLUXNET2015 dataset and the ONEFlux processing pipeline for eddy covariance data. *Sci Data* 7, 225 (2020). <https://doi.org/10.1038/s41597-020-0534-3>

University of East Anglia Climatic Research Unit; Harris, I.C.; Jones, P.D.; Osborn, T. (2021): CRU TS4.05: Climatic Research Unit (CRU) Time-Series (TS) version 4.05 of high-resolution gridded data of month-by-month variation in climate (Jan. 1901- Dec. 2020). NERC EDS Centre for Environmental Data Analysis, date of citation. <https://catalogue.ceda.ac.uk/uuid/c26a65020a5e4b80b20018f148556681>

Weedon, G. P., G. Balsamo, N. Bellouin, S. Gomes, M. J. Best, and P. Viterbo (2014), The WFDEI meteorological forcing data set: WATCH Forcing Data methodology applied to ERA-Interim reanalysis data, *Water Resour. Res.*, 50, 7505–7514, doi:10.1002/2014WR015638.

Fick, S.E. and R.J. Hijmans, 2017. WorldClim 2: new 1km spatial resolution climate surfaces for global land areas. *International Journal of Climatology* 37 (12): 4302-4315.

pmodel_output	<i>rsofun P-model output data</i>
---------------	-----------------------------------

Description

Example output dataset from a p-model run using [pmodel_drivers](#). Results are stored as nested tibble in the column data. See [run_pmodel_f_bysite](#) and [run_pmodel_onestep_f_bysite](#) for a detailed description of the outputs.

Usage

```
pmodel_output
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 5 rows and 3 columns.

pmodel_validation	<i>rsofun P-model GPP and leaf D13C validation data</i>
-------------------	---

Description

Small example dataset of target observations (daily GPP flux data as well as leaf D13C data) to optimize model parameters with the function [calib_sofun](#)

Usage

```
pmodel_validation
```

Format

A tibble of validation data:

sitename A character string containing the site name (e.g. 'FR-Pue').

source A character string containing the name of the data source. It can be used for specifying which data sets to use for calibration.

data A single tibble or a list of tibbles with target observations. For daily simulations this single tibble can contain daily values for:

date Date vector with format YYYY-MM-DD.

gpp The observed Gross Primary Productivity (GPP) for each time stamp (in $\text{gC m}^{-2} \text{d}^{-1}$).

gpp_qc A quality check indicator.

le Latent heat of exchange ...

... etc.

For onestep acclimation this can fore example include:

bigD13C A tibble containing observations of leaf D13C values.

Vcmax A tibble containing observations of Vcmax values.

Jmax A tibble containing observations of Jmax values.

Source

Pastorello, G., Trotta, C., Canfora, E. et al. The FLUXNET2015 dataset and the ONEFlux processing pipeline for eddy covariance data. *Sci Data* 7, 225 (2020). <https://doi.org/10.1038/s41597-020-0534-3>

Cornwell, W. (2025). *traitecoevo/leaf13C: V0.2.2 (Version v0.2.2)* [Computer software]. Zenodo. <https://doi.org/10.5281/zenodo.15239220>

Cornwell, William K., et al. "Climate and soils together regulate photosynthetic carbon isotope discrimination within C3 plants worldwide." *Global Ecology and Biogeography* 27.9 (2018): 1056-1067.

Examples

```
require(ggplot2)
require(tidyr)
p_model_oldformat_validation %>% tidyr::unnest(data)
pmodel_validation |> dplyr::filter(grepl("[A-Z]", sitename)) |> unnest(data)
pmodel_validation |> dplyr::filter(grepl("lon", sitename)) |> unnest(data)
```

run_biomee_f_bysite	<i>Run BiomeE (R wrapper)</i>
---------------------	-------------------------------

Description

Run BiomeE Fortran model on single site.

Usage

```
run_biomee_f_bysite(
  sitename,
  params_siml,
  site_info,
  forcing,
  params_tile,
  params_species,
  init_cohort,
  init_soil,
  init_lu = NULL,
  luc_forcing = NULL,
  makecheck = TRUE
)
```

Arguments

sitename	Site name.
params_siml	Simulation parameters.
site_info	Site meta info in a data.frame.
forcing	A data.frame of forcing climate data, used as input.
params_tile	Tile-level model parameters, into a single row data.frame.
params_species	A data.frame containing species-specific model parameters, with one species per row. See examples biomee_gs_leuning_drivers or biomee_p_model_drivers
init_cohort	A data.frame of initial cohort specifications.
init_soil	A data.frame of initial soil pools.
init_lu	A data.frame of initial land unit (LU) specifications.
luc_forcing	An array of land use change (LUC) used during transient phase. For further specifications of above inputs and examples see biomee_gs_leuning_drivers , biomee_p_model_drivers , or biomee_p_model_luluc_drivers .
makecheck	A logical specifying whether checks are performed to verify forcings and model parameters. TRUE by default.

Value

A data.frame with columns containing model output for each land unit (LU). See examples [biomee_gs_leuning_output](#), [biomee_p_model_output](#), or [biomee_p_model_luluc_output](#). If only one land unit (LU) is simulated, the column is named 'data'. If multiple land units (LU) are simulated, the columns are named according to the LU names. If multiple land units (LU) are simulated, an additional column 'aggregated' contains output aggregating all tiles as well as product pools. Model output for each land unit (LU) is provided as a list. Each list has elements: output_daily_tile, output_annual_tile, output_annual_cohorts, restart_init_cohort, and restart_init_soil. Model output for the aggregated land units (LU) is provided as a list containing output_daily_cell.

output_daily_tile A data.frame with daily outputs at tile level.

year Year of the simulation.
doy Day of the year.
Tk Air temperature (Kelvin).
Tksoil Dampened soil temperature (Kelvin).
Prcp Precipitation ($\text{mm m}^{-2} \text{ day}^{-1}$).
SoilWater Soil water content in root zone (kg m^{-2}).
Transp Transpiration ($\text{mm m}^{-2} \text{ day}^{-1}$).
Evap Evaporation ($\text{mm m}^{-2} \text{ day}^{-1}$).
Runoff Water runoff ($\text{mm m}^{-2} \text{ day}^{-1}$).
ws1 Soil water content for layer 1 ($\text{mm} = \text{kg H}_2\text{O m}^{-2}$).
ws2 Soil water content for layer 2 ($\text{mm} = \text{kg H}_2\text{O m}^{-2}$).
ws3 Soil water content for layer 3 ($\text{mm} = \text{kg H}_2\text{O m}^{-2}$).
LAI Leaf area index (m^2/m^2).
NPP Net primary productivity ($\text{kg C m}^{-2} \text{ day}^{-1}$).
GPP Gross primary production ($\text{kg C m}^{-2} \text{ day}^{-1}$).
Rauto Plant autotrophic respiration ($\text{kg C m}^{-2} \text{ day}^{-1}$).
Rh Heterotrophic respiration ($\text{kg C m}^{-2} \text{ day}^{-1}$).
NSC Non-structural carbon (kg C m^{-2}).
seedC Biomass of seeds (kg C m^{-2}).
leafC Biomass of leaves (kg C m^{-2}).
rootC Biomass of fine roots (kg C m^{-2}).
sapwoodC Biomass of sapwood (kg C m^{-2}).
heartwoodC Biomass of heartwood (kg C m^{-2}).
NSN Non-structural N pool (kg N m^{-2}).
seedN Nitrogen of seeds (kg N m^{-2}).
leafN Nitrogen of leaves (kg N m^{-2}).
rootN Nitrogen of roots (kg N m^{-2}).
sapwoodN Nitrogen of sapwood (kg N m^{-2}).
heartwoodN Nitrogen of heartwood (kg N m^{-2}).
mcrbC Microbial carbon (kg C m^{-2}).
fastSOM Fast soil carbon pool (kg C m^{-2}).
slowSOM Slow soil carbon pool (kg C m^{-2}).

mcrbN Microbial nitrogen (kg N m^{-2}).
fastSoilN Fast soil nitrogen pool (kg N m^{-2}).
slowSoilN Slow soil nitrogen pool (kg N m^{-2}).
mineralN Mineral nitrogen pool (kg N m^{-2}).
N_uptk Nitrogen uptake ($\text{kg N m}^{-2} \text{ day}^{-1}$).

output_annual_tile A data.frame with annual outputs at tile level.

year Year of the simulation.
CAI Crown area index (m^2/m^2).
LAI Leaf area index (m^2/m^2).
Density Number of trees per area (trees ha^{-1}).
DBH Diameter at tile level (cm).
Density12 Tree density for trees with DBH > 12 cm (individuals ha^{-1}).
DBH12 Diameter at tile level considering trees with DBH > 12 cm (cm).
QMD12 Quadratic mean diameter at tile level considering trees with DBH > 12 cm (cm).
NPP Net primary productivity ($\text{kg C m}^{-2} \text{ yr}^{-1}$).
GPP Gross primary productivity ($\text{kg C m}^{-2} \text{ yr}^{-1}$).
Rauto Plant autotrophic respiration ($\text{kg C m}^{-2} \text{ yr}^{-1}$).
Rh Heterotrophic respiration ($\text{kg C m}^{-2} \text{ yr}^{-1}$).
Prctp Annual precipitation ($\text{mm m}^{-2} \text{ yr}^{-1}$).
SoilWater Soil water content in root zone (kg m^{-2}).
Transp Transpiration ($\text{mm m}^{-2} \text{ yr}^{-1}$).
Evap Evaporation ($\text{mm m}^{-2} \text{ yr}^{-1}$).
Runoff Water runoff ($\text{mm m}^{-2} \text{ yr}^{-1}$).
plantC Plant biomass (kg C m^{-2}).
soilC Soil carbon (kg C m^{-2}).
totC Total carbon in plant and soil (kg C m^{-2}).
plantN Plant nitrogen (kg N m^{-2}).
soilN Soil nitrogen (kg N m^{-2}).
totN Total nitrogen in plant and soil (kg N m^{-2}).
NSC Nonstructural carbohydrates (kg C m^{-2}).
seedC Seed biomass (kg C m^{-2}).
leafC Leaf biomass (kg C m^{-2}).
rootC Fine root biomass (kg C m^{-2}).
sapwoodC Sapwood biomass (kg C m^{-2}).
heartwoodC Heartwood biomass (kg C m^{-2}).
NSN Nonstructural nitrogen (kg N m^{-2}).
seedN Seed nitrogen (kg N m^{-2}).
leafN Leaf nitrogen (kg N m^{-2}).
rootN Fine root nitrogen (kg N m^{-2}).
sapwoodN Sapwood nitrogen (kg N m^{-2}).
heartwoodN Heartwood nitrogen (kg N m^{-2}).
mcrbC Microbial carbon (kg C m^{-2}).

fastSOM Fast soil carbon pool (kg C m^{-2}).
slowSOM Slow soil carbon pool (kg C m^{-2}).
mcrbN Microbial nitrogen (kg N m^{-2}).
fastSoilN Fast soil nitrogen pool (kg N m^{-2}).
slowsoilN Slow soil nitrogen pool (kg N m^{-2}).
mineralN Mineral nitrogen pool (kg N m^{-2}).
N_{fxed} Nitrogen fixation (kg N m^{-2}).
N_{uptk} Nitrogen uptake (kg N m^{-2}).
N_{yrMin} Annual available nitrogen (kg N m^{-2}).
N_{P2S} Annual nitrogen from plants to soil (kg N m^{-2}).
N_{loss} Annual nitrogen loss (kg N m^{-2}).
totseedC Total seed carbon (kg C m^{-2}).
totseedN Total seed nitrogen (kg N m^{-2}).
Seedling_C Total carbon from all compartments but seeds (kg C m^{-2}).
Seedling_N Total nitrogen from all compartments but seeds (kg N m^{-2}).
MaxAge Age of the oldest tree in the tile (years).
MaxVolume Maximum volume of a tree in the tile (m^3).
MaxDBH Maximum DBH of a tree in the tile (m).
NPPL Growth of a tree, including carbon allocated to leaves ($\text{kg C m}^{-2} \text{ yr}^{-1}$).
NPPW Growth of a tree, including carbon allocated to sapwood ($\text{kg C m}^{-2} \text{ yr}^{-1}$).
n_{deadtrees} Number of trees that died ($\text{trees m}^{-2} \text{ yr}^{-1}$).
c_{deadtrees} Carbon biomass of trees that died ($\text{kg C m}^{-2} \text{ yr}^{-1}$).
m_{turnover} Continuous biomass turnover ($\text{kg C m}^{-2} \text{ yr}^{-1}$).
c_{turnover_time} Carbon turnover rate, calculated as the ratio between plant biomass and NPP (yr^{-1}).
lu_fraction Fraction of BiomeE grid cell that is occupied by this land unit (LU tile) tile (unitless, or m^{-2} LU area per m^{-2} grid cell area).

output_annual_cohorts A data.frame of annual outputs at the cohort level.

year Year of the simulation.
cID An integer indicating the cohort identity.
PFT An integer indicating the Plant Functional Type.
layer An integer indicating the crown layer, numbered from top to bottom.
density Number of trees per area (trees ha^{-1}).
flayer Fraction of layer area occupied by this cohort.
DBH Tree diameter (cm).
dDBH Diameter growth of a tree in this cohort (cm yr^{-1}).
height Tree height (m).
age Age of the cohort (years).
BA Basal area a tree in this cohort ($\text{m}^2 \text{ tree}^{-1}$).
dBA Basal area increment of a tree in this cohort ($\text{m}^2 \text{ tree}^{-1} \text{ yr}^{-1}$).
Acrown Crown area of a tree in this cohort ($\text{m}^2 \text{ tree}^{-1}$).
Aleaf Total area of leaves ($\text{m}^2 \text{ tree}^{-1}$).

- wood** Sum of sapwood and heartwood biomass of a tree in this cohort (kg C tree⁻¹).
- NSC** Non-structural carbon of a tree in this cohort (kg C tree⁻¹).
- seedC** Biomass of seeds of a tree in this cohort (kg C tree⁻¹).
- leafC** Biomass of leaves of a tree in this cohort (kg C tree⁻¹).
- rootC** Biomass of fine roots of a tree in this cohort (kg C tree⁻¹).
- sapwoodC** Biomass of sapwood of a tree in this cohort (kg C tree⁻¹).
- heartwoodC** Biomass of heartwood of a tree in this cohort (kg C tree⁻¹).
- seedN** Nitrogen content in seeds of a tree in this cohort (kg N tree⁻¹).
- leafN** Nitrogen content in leaves of a tree in this cohort (kg N tree⁻¹).
- rootN** Nitrogen content in fine roots of a tree in this cohort (kg N tree⁻¹).
- sapwoodN** Nitrogen content in sapwood of a tree in this cohort (kg N tree⁻¹).
- heartwoodN** Nitrogen content in heartwood of a tree in this cohort (kg N tree⁻¹).
- NSN** Non-structural nitrogen of a tree in this cohort (kg N tree⁻¹).
- treeG** Total growth of a tree, including carbon allocated to seeds, leaves, fine roots, and sapwood (kg C tree⁻¹ yr⁻¹).
- fseed** Fraction of carbon allocated to seeds to total growth.
- fleaf** Fraction of carbon allocated to leaves to total growth.
- froot** Fraction of carbon allocated to fine roots to total growth.
- fwood** Fraction of carbon allocated to sapwood to total growth.
- NPP** Net primary productivity of a tree (kg C tree⁻¹ yr⁻¹).
- GPP** Gross primary productivity of a tree (kg C tree⁻¹ yr⁻¹).
- Rauto** Plant autotrophic respiration (kg C tree⁻¹ yr⁻¹).
- N_uptk** Nitrogen uptake (kg N tree⁻¹ yr⁻¹).
- N_fxed** Nitrogen fixation (kg N tree⁻¹ yr⁻¹).
- deathrate** Mortality rate of this cohort, including natural mortality, starvation and any other processes causing a loss of individuals in general (yr⁻¹).
- n_deadtrees** Plant to soil N flux due to mortality, including natural mortality, starvation and any other processes causing a loss of individuals in general (kg N yr⁻¹ m⁻²).
- c_deadtrees** Plant to soil C flux due to mortality, including natural mortality, starvation and any other processes causing a loss of individuals in general (kg C yr⁻¹ m⁻²).
- restart_init_cohort** A data.frame that can be passed back as `init_cohort` for a restart run. It contains the initialized pools together with restart-only cohort state such as phenology status, canopy layer, GDD, leaf age, and the stored `bl_max/br_max` targets.
- restart_init_soil** A one-row data.frame that can be passed back as `init_soil` for a restart run. It contains the soil pools together with the tile-level phenology restart state currently used during initialization.

If there are multiple land units (LU) there will also be a column named 'aggregated' containing a data.frame in the column 'output_annual_cell' with annual outputs aggregating all tiles present in the simulation cell. Note that quantities per m2 refer to m2 of grid cell area, i.e. the full area of the BiomeE simulation. 'lu_fraction' refers to the sum of all the tiles, which must remain constant and which represents the fraction of the cell area that is not water/ice. In most cases, it would be close to 1. It contains columns:

output_annual_cell A data.frame with annual outputs aggregating all tiles present in the simulation cell. Note that quantities per m^2 refer to m^2 of grid cell area, i.e. the full area of the BiomeE simulation. 'lu_fraction' refers to the sum of all the tiles, which must remain constant and which represents the fraction of the cell area that is not water/ice. In most cases, it would be close to 1.

all columns from 'output_yearly_tile' See above for output_yearly_tile, but now expressed per unit area of the BiomeE grid cell.

lu_fraction Fraction of BiomeE grid cell that is occupied by this land unit (LU tile) tile (unitless, or m^2 LU area per m^2 grid cell area).

prod_pool_1_C Carbon in product pool 1 (kg C m^{-2} grid cell).

prod_pool_1_N Nitrogen in product pool 1 (kg N m^{-2} grid cell).

prod_pool_2_C Carbon in product pool 2 (kg C m^{-2} grid cell).

prod_pool_2_N Nitrogen in product pool 2 (kg N m^{-2} grid cell).

Rprod_0_C Carbon loss rate directly from land use change (LUC) (kg C m^{-2} grid cell yr^{-1}).

Rprod_0_N Nitrogen loss rate directly from land use change (LUC) (kg C m^{-2} grid cell yr^{-1}).

Rprod_1_C Carbon loss rate from product pool 1 (kg C m^{-2} grid cell yr^{-1}).

Rprod_1_N Nitrogen loss rate from product pool 1 (kg N m^{-2} grid cell yr^{-1}).

Rprod_2_C Carbon loss rate from product pool 2 (kg C m^{-2} grid cell yr^{-1}).

Rprod_2_N Nitrogen loss rate from product pool 2 (kg N m^{-2} grid cell yr^{-1}).

Examples

```
# do not run long-running simulations
# Example BiomeE model run

# Use example drivers data
drivers <- biomee_p_model_drivers

# Run BiomeE for the first site
mod_output <- run_biomee_f_bysite(
  sitename = drivers$sitename[1],
  params_siml = drivers$params_siml[[1]],
  site_info = drivers$site_info[[1]],
  forcing = drivers$forcing[[1]],
  params_tile = drivers$params_tile[[1]],
  params_species = drivers$params_species[[1]],
  init_cohort = drivers$init_cohort[[1]],
  init_soil = drivers$init_soil[[1]]
)
```

Description

Run P-model on a single site for a forcing time series.

Usage

```
run_pmodel_f_bysite(
  sitename,
  params_siml,
  site_info,
  forcing,
  params_modl,
  makecheck = TRUE,
  verbose = TRUE
)
```

Arguments

sitename	Site name.
params_siml	Simulation parameters. See pmodel_drivers
site_info	Site meta info in a data.frame.
forcing	A data frame of forcing climate data, used as input.
params_modl	A named list of free (calibratable) model parameters. See runread_pmodel_f
makecheck	A logical specifying whether checks are performed to verify forcings and model parameters. TRUE by default.
verbose	A logical specifying whether to print warnings. Defaults to TRUE. For further specifications of above inputs and examples see pmodel_drivers .

Details

Depending on the input model parameters, it's possible to run the different P-model setups presented in Stocker et al. 2020 GMD. The P-model version implemented in this package allows more flexibility than the one presented in the paper, with the following functions:

The temperature dependence of the quantum yield efficiency is given by:

$$\varphi_0(T) = c(1 + a(T - b)^2) \text{ if } 0 < c(1 + a(T - b)^2) < 1,$$

$$\varphi_0(T) = 0 \text{ if } c(1 + a(T - b)^2) \leq 0, \text{ and}$$

$$\varphi_0(T) = 1 \text{ if } c(1 + a(T - b)^2) \geq 1.$$

The ORG setup can be reproduced by setting $kphio_par_a = 0$ and calibrating the $kphio$ parameter only. The BRC setup (which calibrates $c_L = \frac{a_L b_L}{4}$ in Eq. 18) is more difficult to reproduce, since the temperature-dependency has been reformulated and a custom cost function would be necessary for calibration. The new parameters are related to c_L as follows:

$$a = -0.0004919819$$

$$b = 32.35294$$

$$c = 0.6910823c_L$$

The soil moisture stress is implemented as

$$\beta(\theta) = \frac{\beta_0 - 1}{\theta^{*2}}(\theta - \theta^*)^2 + 1 \text{ if } 0 \leq \theta \leq \theta^* \text{ and}$$

$$\beta(\theta) = 1 \text{ if } \theta > \theta^*.$$

In Stocker et al. 2020 GMD, the threshold plant-available soil water is set as $\theta^* = 0.6 * whc$ where whc is the site's water holding capacity. Also, the β reduction at low soil moisture ($\beta_0 = \beta(0)$) was parameterized as a linear function of mean aridity (Eq. 20 in Stocker et al. 2020 GMD) but is considered a constant model parameter in this package. Hence, the FULL calibration setup cannot be exactly replicated.

Value

Model output is provided as a tidy dataframe (tibble), with columns:

`date` Date of the observation in YYYY-MM-DD format.

`year_dec` Decimal representation of year and day of the year (for example, 2007.000 corresponds to 2007-01-01 and 2007.003 to 2007-01-02).

`fapar` Fraction of photosynthetic active radiation (fAPAR), taking values between 0 and 1.

`gpp` Gross Primary Productivity (GPP) for each time stamp (in $\text{gC m}^{-2} \text{d}^{-1}$).

`aet` Actual evapotranspiration (AET), calculated by SPLASH following Priestly-Taylor (in mm d^{-1}).

`le` Latent heat flux (in $\text{J m}^{-2} \text{d}^{-1}$).

`pet` Potential evapotranspiration (PET), calculated by SPLASH following Priestly-Taylor (in mm d^{-1}).

`vcmax` Maximum rate of RuBisCO carboxylation (V_{cmax}) (in $\text{mol C m}^{-2} \text{s}^{-1}$).

`jmax` Maximum rate of electron transport for RuBP regeneration (in $\text{mol CO}_2 \text{m}^{-2} \text{s}^{-1}$).

`vcmax25` Maximum rate of carboxylation (V_{cmax}), normalised to 25°C (in $\text{mol C m}^{-2} \text{s}^{-1}$).

`jmax25` Maximum rate of electron transport, normalised to 25°C (in $\text{mol C m}^{-2} \text{s}^{-1}$).

`gs_accl` Acclimated stomatal conductance (in $\text{mol C (mol photons)}^{-1} \text{Pa}^{-1}$. (Multiply by `ppfd` ($\text{mol photons m}^{-2} \text{d}^{-1}$) and `fapar` to express per unit ground area and time.)

`wscal` Relative soil water content, between 0 (permanent wilting point, PWP) and 1 (field capacity, FC).

`chi` Ratio of leaf-internal to ambient CO_2 , $c_i:c_a$ (unitless).

`iwue` Intrinsic water use efficiency (iWUE) (unitless, multiply with `patm` (Pa) to get iWUE in Pa).

`rd` Dark respiration (R_d) in $\text{gC m}^{-2} \text{s}^{-1}$. (Multiply by $1/12$ (mol C / gC) to convert to $\text{mol C m}^{-2} \text{s}^{-1}$.)

`tsoil` Soil temperature, in $^\circ\text{C}$.

`netrad` Net radiation, in W m^{-2} . WARNING: this is currently ignored as a model forcing. Instead, net radiation is internally calculated by SPLASH.

`wcont` Soil water content, in mm.

`snow` Snow water equivalents, in mm.

`cond` Water input by condensation, in mm d^{-1}

`cleaf` C mass of a virtual leaf carbon pool to keep track of isotopic composition, in gC m^{-2}

`cleafd13c` ^{13}C isotopic signature (delta) of `cleaf`, in permil.

Examples

```
# Define model parameter values from previous work
params_mod1 <- list(
  kphio          = 0.04998,    # setup ORG in Stocker et al. 2020 GMD
  kphio_par_a    = 0.0,        # disable temperature-dependence of kphio
  kphio_par_b    = 1.0,
  soilm_thetastar = 0.6 * 240, # old setup with soil moisture stress
  soilm_betao    = 0.0,
  beta_unitcostratio = 146.0,
  rd_to_vcmax    = 0.014,      # from Atkin et al. 2015 for C3 herbaceous
  tau_acclim     = 30.0,
  kc_jmax        = 0.41
)

# Run the Fortran P-model
mod_output <- run_pmodel_f_bysite(
  # unnest drivers example data
  sitename = pmodel_drivers$sitename[[5]],
  params_siml = pmodel_drivers$params_siml[[5]],
  site_info = pmodel_drivers$site_info[[5]],
  forcing = pmodel_drivers$forcing[[5]],
  params_mod1 = params_mod1
)
```

```
run_pmodel_onestep_f_bysite
      Run P-model (single time step)
```

Description

Run P-model on a single site for a single time step. This does not include the simulation of ecosystem-level quantities, water limitation, nor a simulation of water fluxes. Instead, this corresponds to a leaf-level representation of the acclimation of photosynthesis.

Usage

```
run_pmodel_onestep_f_bysite(lc4, forcing, params_mod1, makecheck = TRUE)
```

Arguments

lc4	Logical specifying whether P-model simulation is for C4 (as opposed to C3). Defaults to FALSE.
forcing	A data frame of forcing climate data, used as input (single row).
params_mod1	A named list of free (calibratable) model parameters. See runread_pmodel_f
makecheck	A logical specifying whether checks are performed to verify forcings and model parameters. TRUE by default. For further specifications of above inputs and examples see pmodel_drivers

Details

TBC

Value

Model output is provided as a tidy dataframe (tibble), with columns:

vcmax Maximum rate of RuBisCO carboxylation (Vcmax) (in mol C m⁻² s⁻¹).

jmax Maximum rate of electron transport for RuBP regeneration (in mol CO₂ m⁻² s⁻¹).

vcmax25 Maximum rate of carboxylation (Vcmax), normalised to 25°C (in mol C m⁻² s⁻¹).

jmax25 Maximum rate of electron transport, normalised to 25°C (in mol C m⁻² s⁻¹).

gs_acc1 Acclimated stomatal conductance (in mol C (mol photons)⁻¹ Pa⁻¹. (Multiply by ppfd (mol photons m⁻² d⁻¹) and fapar to express per unit ground area and time.)

chi Ratio of leaf-internal to ambient CO₂, ci:ca (unitless).

iwue Intrinsic water use efficiency (iWUE) (unitless, multiply with patm (Pa) to get iWUE in Pa).

rd Dark respiration (Rd) in gC m⁻² s⁻¹. (Multiply by 1/12 (mol C / gC) to convert to mol C m⁻² s⁻¹.)

bigdelta 13C isotope discrimination of leaf assimilates against atmospheric signature (permil).

Examples

```
# Define model parameter values from previous work
params_mod1 <- list(
  kphio           = 0.04998,    # setup ORG in Stocker et al. 2020 GMD
  kphio_par_a     = 0.0,        # disable temperature-dependence of kphio
  kphio_par_b     = 1.0,
  beta_unitcostratio = 146.0,
  rd_to_vcmax     = 0.014,      # from Atkin et al. 2015 for C3 herbaceous
  kc_jmax         = 0.41
)

# Run the Fortran P-model
run_pmodel_onestep_f_bysite(
  lc4 = FALSE,
  forcing = data.frame(
    temp = 20,      # temperature, deg C
    vpd  = 1000,    # Pa,
    ppfd = 300 / 10^6, # mol/m2/s
    co2  = 400,     # ppm,
    patm = 101325   # Pa
  ),
  params_mod1 = list(
    kphio           = 0.04998,    # setup ORG in Stocker et al. 2020 GMD
    kphio_par_a     = 0.0,        # disable temperature-dependence of kphio
    kphio_par_b     = 1.0,
    beta_unitcostratio = 146.0,
    rd_to_vcmax     = 0.014,      # from Atkin et al. 2015 for C3 herbaceous
    kc_jmax         = 0.41
  ),
```

```

    makecheck = TRUE
  )

```

runread_biomee_f	<i>Run BiomeE</i>
------------------	-------------------

Description

Runs BiomeE model for multiple sites.

Usage

```
runread_biomee_f(drivers, makecheck = TRUE, parallel = FALSE, ncores = 1)
```

Arguments

drivers	A nested data frame with one row for each site and columns named according to the arguments of function run_biomee_f_bysite . Namely sitename, params_siml, site_info, forcing, params_tile, params_species, init_cohort and init_soil.
makecheck	A logical specifying whether checks are performed to verify forcings and model parameters. TRUE by default.
parallel	Deprecated. Use ncores instead.
ncores	An integer specifying the number of cores used for parallel computing (sites processed in parallel). Default: 1 (no parallel execution).

Value

A data frame (tibble) with one row for each site. The columns are the site information site_info and one column per land unit (LU) in addition to an aggregated output aggregated. By default, the only LU is named data and aggregated is not present since aggregating one LU is not useful. When multiple LU are configured (using init_lu), the columns are named using the LU name provided in init_lu. See [run_biomee_f_bysite](#) for a detailed description of the outputs. Example outputs are provided as [biomee_p_model_output](#) and [biomee_p_model_luluc_output](#).

Examples

```

# Example BiomeE model run
# do not run long-running simulations
runread_biomee_f(
  drivers = biomee_p_model_drivers
)
## Not run: # do not run this long-running example at all, only *show* example
runread_biomee_f(
  drivers = biomee_gs_leuning_drivers
)

## End(Not run)

```

runread_pmodel_f *Run P-model*

Description

Runs P-model for multiple sites.

Usage

```
runread_pmodel_f(drivers, par, makecheck = TRUE, parallel = FALSE, ncores = 1)
```

Arguments

drivers	A nested data frame with one row for each site and columns sitename and further columns params_siml, site_info and forcing. (params_siml contains the information whether or not to run the onestep or daily model through TRUE/FALSE in column onestep. It also contains parameters lc4 in the case of onestep==TRUE and those for function run_pmodel_f_bysite in the case of onestep==FALSE.)
par	A named list of free (calibratable) model parameters. kphio The quantum yield efficiency at optimal temperature φ_0, in mol mol⁻¹. When temperature dependence is used, it corresponds to the multiplicative parameter c (see Details). kphio_par_a The shape parameter a of the temperature-dependency of quantum yield efficiency (see Details). To disable the temperature dependence, set kphio_par_a = 0. kphio_par_b The optimal temperature parameter b of the temperature dependent quantum yield efficiency (see Details), in °C. soilm_thetastar The threshold parameter θ^* in the soil moisture stress function (see Details), given in mm. To turn off the soil moisture stress, set soilm_thetastar = 0. soilm_betao The intercept parameter β_0 in the soil moisture stress function (see Details). This is the parameter calibrated in Stocker et al. 2020 GMD. beta_unitcostratio The unit cost of carboxylation, corresponding to $\beta = b/a'$ in Eq. 3 of Stocker et al. 2020 GMD. rd_to_vcmax Ratio of Rdark (dark respiration) to Vcmax25. tau_acclim Acclimation time scale of photosynthesis, in days. kc_jmax Parameter for Jmax cost ratio (corresponding to c^* in Stocker et al. 2020 GMD).
makecheck	A logical specifying whether checks are performed to verify forcings and model parameters. TRUE by default.
parallel	A logical specifying whether simulations are to be parallelised (sending data from a certain number of sites to each core). Defaults to FALSE. (Not applied to rows where onestep==TRUE.)
ncores	An integer specifying the number of cores used for parallel computing (by default ncores = 2).

Details

Depending on the input model parameters, it's possible to run the different P-model setups presented in Stocker et al. 2020 GMD. The P-model version implemented in this package allows more flexibility than the one presented in the paper, with the following functions:

The temperature dependence of the quantum yield efficiency is given by:

$$\varphi_0(T) = c(1 + a(T - b)^2) \text{ if } 0 < c(1 + a(T - b)^2) < 1,$$

$$\varphi_0(T) = 0 \text{ if } c(1 + a(T - b)^2) \leq 0, \text{ and}$$

$$\varphi_0(T) = 1 \text{ if } c(1 + a(T - b)^2) \geq 1.$$

The ORG setup can be reproduced by setting `kphio_par_a = 0` and calibrating the `kphio` parameter only. The BRC setup (which calibrates $c_L = \frac{a_L b_L}{4}$ in Eq. 18) is more difficult to reproduce, since the temperature-dependency has been reformulated and a custom cost function would be necessary for calibration. The new parameters are related to c_L as follows:

$$a = -0.0004919819$$

$$b = 32.35294$$

$$c = 0.6910823c_L$$

The soil moisture stress is implemented as

$$\beta(\theta) = \frac{\beta_0 - 1}{\theta^* - 2}(\theta - \theta^*)^2 + 1 \text{ if } 0 \leq \theta \leq \theta^* \text{ and}$$

$$\beta(\theta) = 1 \text{ if } \theta > \theta^*.$$

In Stocker et al. 2020 GMD, the threshold plant-available soil water is set as $\theta^* = 0.6 * whc$ where `whc` is the site's water holding capacity. Also, the β reduction at low soil moisture ($\beta_0 = \beta(0)$) was parameterized as a linear function of mean aridity (Eq. 20 in Stocker et al. 2020 GMD) but is considered a constant model parameter in this package. Hence, the FULL calibration setup cannot be exactly replicated.

Value

A data frame (tibble) with one row for each site, site information stored in the nested column `site_info` and outputs stored in the nested column `data`. See [run_pmodel_f_bysite](#) and [run_pmodel_onestep_f_bysite](#) for a detailed description of the outputs. Example outputs are provided as [pmodel_output](#).

Examples

```
# Define model parameter values from previous work
params_modl <- list(
  kphio           = 0.04998,    # setup ORG in Stocker et al. 2020 GMD
  kphio_par_a     = 0.0,       # disable temperature-dependence of kphio
  kphio_par_b     = 1.0,
  soilm_thetastar = 0.6 * 240, # old setup with soil moisture stress
  soilm_betao     = 0.0,
  beta_unitcostratio = 146.0,
  rd_to_vcmax     = 0.014,     # from Atkin et al. 2015 for C3 herbaceous
  tau_acclim      = 30.0,
  kc_jmax         = 0.41
)

# Run the model for these parameters and the example drivers
output <- rsfun::runread_pmodel_f(
  drivers = rsfun::pmodel_drivers |> dplyr::slice(1),
  par = params_modl)
```

```
output <- rsofun::runread_pmodel_f(  
  drivers = rsofun::pmodel_drivers |> dplyr::slice(1),  
  par = params_mod1, parallel = TRUE, ncores = 3)
```

Index

* datasets

- biomee_gs_leuning_drivers, [3](#)
- biomee_gs_leuning_output, [8](#)
- biomee_p_model_drivers, [9](#)
- biomee_p_model_luluc_drivers, [9](#)
- biomee_p_model_luluc_output, [10](#)
- biomee_p_model_output, [10](#)
- biomee_validation, [11](#)
- p_model_oldformat_drivers, [31](#)
- p_model_oldformat_validation, [32](#)
- pmodel_drivers, [32](#)
- pmodel_output, [34](#)
- pmodel_validation, [34](#)

BayesianTools::runMCMC, [13, 16](#)

- biomee_gs_leuning_drivers, [3, 8, 9, 36](#)
- biomee_gs_leuning_output, [8, 37](#)
- biomee_p_model_drivers, [9, 10, 36](#)
- biomee_p_model_luluc_drivers, [9, 10, 36](#)
- biomee_p_model_luluc_output, [10, 37, 46](#)
- biomee_p_model_output, [10, 37, 46](#)
- biomee_validation, [3, 9, 11](#)
- build_luc_matrix, [11](#)

calib_sofun, [12, 34](#)

calib_sofun_legacy, [11, 15, 32](#)

cost_likelihood_biomee, [17](#)

cost_likelihood_pmodel, [18](#)

cost_likelihood_pmodel_legacy, [20](#)

cost_rmse_biomee, [22](#)

cost_rmse_pmodel, [23](#)

createMixedPrior, [25](#)

GenSA, [13, 16](#)

get_mod_obs_pmodel, [25](#)

init_dates_dataframe, [26](#)

is_beta_prior, [27](#)

is_lognormal_prior, [28](#)

is_normal_prior, [29](#)

is_trunclognormal_prior, [29](#)

is_truncnormal_prior, [30](#)

is_uniform_prior, [31](#)

p_model_oldformat_drivers, [31](#)

p_model_oldformat_validation, [32](#)

pmodel_drivers, [12, 15, 19, 20, 23, 26, 31, 32, 34, 42, 44](#)

pmodel_output, [34, 48](#)

pmodel_validation, [12, 15, 19, 20, 23, 26, 32, 34](#)

run_biomee_f_bysite, [8, 10, 36, 46](#)

run_pmodel_f_bysite, [34, 41, 47, 48](#)

run_pmodel_onestep_f_bysite, [34, 44, 48](#)

runMCMC, [13, 16](#)

runread_biomee_f, [8, 10, 46](#)

runread_pmodel_f, [19–21, 23, 24, 42, 44, 47](#)