Package ‘medfate’

December 16, 2021

Type Package

Title Mediterranean Forest Simulation

Version 2.3.7

Date 2021-12-15

Description
Functions to simulate Mediterranean forest functioning and dynamics using cohort-based description of vegetation [De Caceres et al. (2015) <doi:10.1016/j.agrformet.2015.06.012>; De Caceres et al. (2021) <doi:10.1016/j.agrformet.2020.108233>].

License GPL (>= 2)

URL https://emf-creaf.github.io/medfate/

LazyLoad yes

Depends R (>= 3.5.0)

Imports ggplot2, meteoland (>= 0.8.1), Rcpp (>= 1.0.6), shiny

Suggests knitr, rmarkdown

LinkingTo Rcpp, meteoland

Encoding UTF-8

NeedsCompilation yes

VignetteBuilder utils, knitr

RoxygenNote 6.1.1

BugReports https://github.com/emf-creaf/medfate/issues

Author Miquel De Cáceres [aut, cre],
Shengli Huang [aut],
Víctor Granda [aut],
Antoine Cabon [aut],
Jordi Martínez-Vilalta [ctb],
Maurizio Mencuccini [ctb],
Nicolas Martin-StPaul [ctb]

Maintainer Miquel De Cáceres <miquelcaceres@gmail.com>

Repository CRAN

Date/Publication 2021-12-16 22:00:02 UTC

R topics documented:

`biophysics` *Physical and biophysical utility functions*

Description

Utility functions for the calculation of biophysical variables.

Usage

```
biophysics_irradianceToPhotonFlux(I, lambda)
biophysics_leafTemperature(absRad, airTemperature, u, E, leafWidth = 1.0)
biophysics_leafTemperature2(SWRabs, LWRnet, airTemperature, u, E, leafWidth = 1.0)
biophysics_leafVapourPressure(leafTemp, leafPsi)
biophysics_radiationDiurnalPattern(t, daylength)
biophysics_temperatureDiurnalPattern(t, tmin, tmax,
                                     tminPrev, tmaxPrev, tminNext, daylength)
biophysics_waterDynamicViscosity(temp)
```

Arguments

`I`	Irradiance (in W*m-2).
`lambda`	Wavelength (in nm).
`u`	Wind speed above the leaf boundary layer (in m/s).
`airTemperature`	Air temperature (in ºC).
`tmin, tmax`	Minimum and maximum daily temperature (ºC).
`tminPrev, tmaxPrev, tminNext`	Maximum and minimum daily temperatures of the previous and following day (ºC).
`absRad`	Absorbed long- and short-wave radiation (in W·m-2).
`SWRabs`	Absorbed short-wave radiation (in W·m-2).
`LWRnet`	Net long-wave radiation balance (in W·m-2).

E	Transpiration flow (in mmol H20·m-2·s-1) per one sided leaf area basis.
leafWidth	Leaf width (in cm).
t	Time of the day (in seconds).
daylength	Day length (in seconds).
temp	Temperature (ºC).
leafTemp	Leaf temperature (ºC).
leafPsi	Leaf water potential (MPa).

Details

Function `biophysics_leafTemperature` calculates leaf temperature according to energy balance equation given in Campbell and Norman (1988). Function `biophysics_radiationDiurnalPattern` follows the equations given in Liu and Jordan (1960). Function `biophysics_temperatureDiurnalPattern` determines diurnal temperature pattern assuming a sinusoidal pattern with T = Tmin at sunrise and T = (Tmin+Tmax)/2 at sunset and a linear change in temperature between sunset and Tmin of the day after (McMurtrie et al. 1990). Function `biophysics_waterDynamicViscosity` calculates water dynamic viscosity following the Vogel (1921) equation.

Value

Values returned for each function are:

- `biophysics_leafTemperature`: leaf temperature (in ºC)
- `biophysics_leafVapourPressure`: leaf vapour pressure (in kPa)
- `biophysics_radiationDiurnalPattern`: the proportion of daily radiation corresponding to the input time in seconds after sunrise.
- `biophysics_temperatureDiurnalPattern`: diurnal pattern of temperature.
- `biophysics_waterDynamicViscosity`: Water dynamic viscosity relative to 20ºC.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Campbell, G. S., and J. M. Norman. 1998. An introduction to environmental biophysics: 2nd edition. (eqns. 14.1 & 14.3)

B. Y. H. Liu and R. C. Jordan, “The interrelationship and characteristic distribution of direct, diffuse and total solar radiation,” Solar Energy, vol. 4, no. 3, pp. 1–19, 1960.

McMurtrie, R. E., D. A. Rook, and F. M. Kelliher. 1990. Modelling the yield of Pinus radiata on a site limited by water and nitrogen. Forest Ecology and Management 30:381–413.

H. Vogel, "Das Temperaturabhangigkeitsgesetz der Viskositat von Flussigkeiten", Physikalische Zeitschrift, vol. 22, pp. 645–646, 1921.

See Also

spwb

carbon *Carbon-related functions*

Description

Set of functions used in the calculation of carbon balance.

Usage

```
carbon_leafStarchCapacity(LAI, N, SLA, leafDensity)
carbon_leafStructuralBiomass(LAI, N, SLA)
carbon_sapwoodStarchCapacity(SA, H, L, V, woodDensity, conduit2sapwood)
carbon_sapwoodStructuralBiomass(SA, H, L, V, woodDensity)
carbon_sapwoodStructuralLivingBiomass(SA, H, L, V, woodDensity, conduit2sapwood)
carbon_sugarConcentration(osmoticWP, temp, nonSugarConc)
carbon_osmoticWaterPotential(sugarConc, temp, nonSugarConc)
carbon_relativeSapViscosity(sugarConc, temp)
carbon_sugarStarchDynamicsLeaf(sugarConc, starchConc, eqSugarConc)
```

Arguments

`LAI`	Leaf area index.
`N`	Density (ind·ha-1).
`SLA`	Specific leaf area (mm2/mg = m2/kg).
`leafDensity`	Density of leaf tissue (dry weight over volume).
`SA`	Sapwood area (cm2).
`H`	Plant height (cm).
`L`	Coarse root length (mm) for each soil layer.
`V`	Proportion of fine roots in each soil layer.
`woodDensity`	Wood density (dry weight over volume).
`conduit2sapwood`	Proportion of sapwood corresponding to conducive elements (vessels or tracheids) as opposed to parenchymatic tissue.
`osmoticWP`	Osmotic water potential (MPa).
`temp`	Temperature (degrees Celsius).
`nonSugarConc`	Concentration of inorganic solutes (mol/l).
`sugarConc`	Concentration of soluble sugars (mol/l).
`starchConc`	Concentration of starch (mol/l)
`eqSugarConc`	Equilibrium concentration of soluble sugars (mol/l).

Value

Values returned for each function are:

- carbon_leafStarchCapacity: Capacity of storing starch in the leaf compartment (mol glucose).
- carbon_sapwoodStarchCapacity: Capacity of storing starch in the sapwood compartment (mol glucose).

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

growth

---

conductancefunctions *Hydraulic confuctance functions*

Description

Set of functions used in the calculation of soil and plant hydraulic conductance.

Usage

```
hydraulics_psi2K(psi, Psi_extract, ws = 3.0)
hydraulics_K2Psi(K, Psi_extract, ws = 3.0)
hydraulics_averagePsi(psi, v, c, d)
hydraulics_vulnerabilityCurvePlot(x, soil = NULL, type="leaf",
                                  psiVec =  seq(-0.1, -8.0, by =-0.01),
                                  relative = FALSE, speciesNames = FALSE,
                               draw = TRUE, ylim = NULL, xlab = NULL, ylab=NULL)
hydraulics_psiCrit(c, d, pCrit = 0.001)
hydraulics_vanGenuchtenConductance(psi, krhizomax, n, alpha)
hydraulics_xylemConductance(psi, kxylemmax, c, d)
hydraulics_xylemPsi(kxylem, kxylemmax, c, d)
hydraulics_correctConductanceForViscosity(kxylem, temp)
hydraulics_psi2Weibull(psi50, psi88 = NA, psi12 = NA)
```

Arguments

psi	A scalar (or a vector, depending on the function) with water potential (in MPa).
K	Whole-plant relative conductance (0-1).
Psi_extract	Soil water potential (in MPa) corresponding to 50% whole-plant relative conductance.
ws	Exponent of the whole-plant relative conductance Weibull function.

v	Proportion of fine roots within each soil layer.
krhizomax	Maximum rhizosphere hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
kxylemmax	Maximum xylem hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
c, d	Parameters of the Weibull function (generic xylem vulnerability curve).
n, alpha	Parameters of the Van Genuchten function (rhizosphere vulnerability curve).
kxylem	Xylem hydraulic conductance (defined as flow per surface unit and per pressure drop).
x	An object of class spwbInput.
soil	A list containing the description of the soil (see soil).
type	Plot type for hydraulics_vulnerabilityCurvePlot, either "leaf", "stem", "root", "rootlayer" or "rhizosphere").
psiVec	Vector of water potential values to evaluate for the vulnerability curve.
relative	A flag to relativize vulnerability curves to the [0-1] interval.
speciesNames	A flag to indicate the use of species names instead of cohort names in plots.
draw	A flag to indicate whether the vulnerability curve should be drawn or just returned.
ylim, xlab, ylab	Graphical parameters to override function defaults.
pCrit	Proportion of maximum conductance considered critical for hydraulic functioning.
psi50, psi88, psi12	Water potentials (in MPa) corresponding to 50%, 88% and 12% percent conductance loss.
temp	Temperature (in degrees Celsius).

Details

Details of the hydraulic model are given in a vignette. Function `hydraulics_vulnerabilityCurvePlot` draws a plot of the vulnerability curves for the given `soil` object and network properties of each plant cohort in `x`.

Value

Values returned for each function are:

- `hydraulics_psi2K`: Whole-plant relative conductance (0-1).
- `hydraulics_K2Psi`: Soil water potential (in MPa) corresponding to the given whole-plant relative conductance value (inverse of `hydraulics_psi2K()`).
- `hydraulics_averagePsi`: The average water potential (in MPa) across soil layers.
- `hydraulics_vanGenuchtenConductance`: Rhizosphere conductance corresponding to an input water potential (soil vulnerability curve).

- hydraulics_xylemConductance: Xylem conductance (flow rate per pressure drop) corresponding to an input water potential (plant vulnerability curve).
- hydraulics_xylemPsi: Xylem water potential (in MPa) corresponding to an input xylem conductance (flow rate per pressure drop).
- hydraulics_psi2Weibull: Parameters of the Weibull vulnerability curve that goes through the supplied psi50 and psi88 values.

Author(s)

Miquel De Cáceres Ainsa, CREAF.

References

Sperry, J. S., F. R. Adler, G. S. Campbell, and J. P. Comstock. 1998. Limitation of plant water use by rhizosphere and xylem conductance: results from a model. Plant, Cell \& Environment 21:347–359.

Sperry, J. S., and D. M. Love. 2015. What plant hydraulics can tell us about responses to climate-change droughts. New Phytologist 207:14–27.

See Also

hydraulics_supplyFunctionPlot, hydraulics_maximumStemHydraulicConductance, spwb, soil

Examples

```
#Manual display of vulnerability curve
kstemmax = 4 # in mmol·m-2·s-1·MPa-1
stemc = 3
stemd = -4 # in MPa
psiVec = seq(-0.1, -7.0, by =-0.01)
kstem = unlist(lapply(psiVec, hydraulics_xylemConductance, kstemmax, stemc, stemd))
plot(-psiVec, kstem, type="l",ylab="Xylem conductance (mmol·m-2·s-1·MPa-1)",
xlab="Canopy pressure (-MPa)", lwd=1.5,ylim=c(0,kstemmax))


#Load example dataset
data(exampleforestMED)
#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params (2 layers)
examplesoil = soil(defaultSoilParams(2))

#Initialize control parameters
control = defaultControl("Granier")

#Switch to 'Sperry' transpiration mode
control = defaultControl("Sperry")

#Initialize input
```

```
x = forest2spwbInput(exampleforestMED,examplesoil, SpParamsMED, control)

#Leaf vulnerability curves
hydraulics_vulnerabilityCurvePlot(x, type="leaf")

#Stem vulnerability curves
hydraulics_vulnerabilityCurvePlot(x, type="stem")
```

`defaultControl` *Default simulation control parameters for models*

Description

Creates a list with global default parameters for simulation models.

Usage

```
defaultControl(transpirationMode = "Granier")
```

Arguments

`transpirationMode`
: Transpiration model (either 'Granier' or 'Sperry'). See `spwbInput`.

Details

The function returns a list with default parameters. Users can change those defaults that need to be set to other values and use the list as input for model functions. The relevant parameters are different for each model function.

Value

A list, with the following options (default values in parentheses):

General:

- `verbose (=TRUE)`: Boolean flag to indicate console output during calculations. In function `fordyn` `verbose` is always set to FALSE.
- `modifyInput (=TRUE)`: Boolean flag to indicate that simulations will modify input object. If set to FALSE, simulations will not modify the input R object but return the current (modified) state variables within the output. In function `fordyn` `modifyInput` is always set to FALSE.
- `fillMissingSpParams (=TRUE)`: Boolean flag to indicate that functions `spwbInput` and `growthInput` should provide estimates for functional parameters if these are lacking in the species parameter table `SpParams`. Note that if `fillMissingSpParams` is set to `FALSE` then simulations may fail if the user does not provide values for required parameters.
- `subdailyResults (=FALSE)`: Boolean flag to force subdaily results to be stored (as a list called 'subdaily' of `spwb_day` objects, one by simulated date) in calls to `spwb`. In function `fordyn` `subdailyResults` is always set to FALSE.

Water balance:

- transpirationMode (="Granier"): Transpiration model (either 'Granier' or 'Sperry'). See spwbInput.
- soilFunctions (="SX"): Soil water retention curve and conductivity functions, either 'SX' (for Saxton) or 'VG' (for Van Genuchten). If transpirationMode = "Sperry" then soilFunctions is set by default to 'VG'.
- defaultWindSpeed (= 2.5): Default wind speed value (in m/s) to be used when missing from data.
- snowpack (=TRUE): Boolean flag to indicate the simulation of snow accumulation and melting.
- leafPhenology (=TRUE): Boolean flag to indicate the simulation of leaf phenology for winter-deciduous species.
- rockyLayerDrainage (=TRUE): Boolean flag to indicate the simulation of drainage from rocky layers (> 95% of rocks).
- unlimitedSoilWater (=FALSE): Boolean flag to indicate the simulation of plant transpiration assuming that soil water is always at field capacity.
- unfoldingDD (=300): Degree-days for complete leaf unfolding after budburst has occurred.
- plantWaterPools (=FALSE): Boolean flag to indicate the simulation of water balance assuming that each cohort has its own water pools.
- verticalLayerSize (= 100): Size of vertical layers (in cm) for the calculation of light extinction (and photosynthesis).
- windMeasurementHeight (=200): Height (in cm) over the canopy corresponding to wind measurements.
- cavitationRefill (= "total"): A string indicating how refilling of embolized conduits is done:
  * "none" - no refilling.
  * "annual" - every first day of the year.
  * "rate" - following a rate of new sapwood formation (only available if transpirationMode = "Sperry")
  * "total" - instantaneous complete refilling.

Water balance with 'Sperry':

- ndailysteps (= 24): Number of steps into which each day is divided for determination of stomatal conductance, transpiration and photosynthesis (24 equals 1-hour intervals).
- nsubsteps (= 3600): Number of substeps into which each step is divided for multi-layer canopy energy balance solving.
- capacitance (=FALSE): Whether the effect of plant water compartments is considered in simulations.
- multiLayerBalance (=FALSE): Flag to indicate multiple canopy energy balance. If FALSE, canopy is considered a single layer for energy balance.
- cochard (=FALSE): Boolean flag to indicate the stomatal closure when leaf water potential is below turgor loss point so that transpiration depends on cuticular properties (Martin-StPaul et al. 2017).
- taper (= TRUE): Whether taper of xylem conduits is accounted for when calculating aboveground stem conductance from xylem conductivity.

- `gainModifier,costModifier` (= `1`): Modifiers (exponents) of the gain and cost functions defined in Sperry et al. (2016).
- `klatstem` (= `0.01`): Stem symplastic-apoplastic lateral conductance (in mmol·s-1·m-2·MPa-1). Only relevant when `capacitance` = `TRUE`.
- `klatleaf` (= `0.01`): Leaf symplastic-apoplastic lateral conductance (in mmol·s-1·m-2·MPa-1). Only relevant when `capacitance` = `TRUE`.
- `numericParams`: A list with the following elements:
  * `maxNsteps` (= `400`): Maximum number of steps in supply function.
  * `ntrial` (= `200`): Number of iteration trials when finding root of equation system.
  * `psiTol` (= `0.0001`): Tolerance value for water potential.
  * `ETol` (= `0.0001`): Tolerance value for flow.
- `thermalCapacityLAI` (=`1000000`): Thermal canopy capacitance per LAI unit.
- `Catm` (=`386`): Default atmospheric (abovecanopy) CO2 concentration (in micromol·mol-1 = ppm). This value will be used whenever CO2 concentration is not specified in the weather input.
- `fracLeafResistance` (= `NA`): Fraction of plant total resistance (leaf+stem+root) that corresponds to leaves. This fraction is used if `VCleaf_kmax` = `NA`.
- `fracRootResistance` (=`0.40`): Fraction of plant total resistance (leaf+stem+root) that corresponds to root system.
- `averageFracRhizosphereResistance` (=`0.15`): Fraction to total continuum (leaf+stem+root+rhizosphere) resistance that corresponds to rhizosphere (averaged across soil water potential values).
- `boundaryLayerSize` (= `2000`): Size of the boundary layer (in cm) over the canopy (relevant for multi-layer canopy energy balance).
- `refillMaximumRate` (= `0.05`): Maximum rate of daily refilling of embolized conduits as sapwood area per leaf area (in cm2·m-2·day-1).

Growth/mortality:

- `allowDessication` (=`TRUE`): Boolean flag to indicate that mortality by dessication is allowed.
- `allowStarvation` (=`TRUE`): Boolean flag to indicate that mortality by starvation is allowed.
- `allowDefoliation` (=`TRUE`): Boolean flag to indicate that complete drought-driven defoliation is allowed.
- `sinkLimitation (=TRUE): Boolean flag to indicate that temperature and turgor limitations to growth are applied.`
- `shrubDynamics` [= `FALSE`]: Boolean flag to allow the application of demographic processes to shrubs.
- `allocationStrategy` (=`"Plant_kmax"`): Strategy for allocation (either "Plant_kmax", for constant maximum plant conductance, or "Al2As" for constant Huber value).
- `nonStomatalPhotosynthesisLimitation` (=`TRUE`): A flag to apply limitations of photosynthesis when leaf sugar levels become too high.
- `phloemConductanceFactor` (= `0.2`)): Factor to transform stem xylem conductance to stem phloem conductance (only for transpirationMode = "Sperry").
- `nonSugarConcentration` (= `0.25`): Non-sugar (inorganic) solute concentration (mol·l-1) in cells.

- equilibriumOsmoticConcentration [= c(leaf = 0.8,sapwood = 0.6)]: Equilibrium osmotic concentrations (mol·l-1) for leaf and sapwood cells. The difference between leaf and sapwood values helps maintaining phloem transport. The equilibrium sugar concentration is equilibriumOsmoticConcentration -nonSugarConcentration defaults to [= c(leaf = 0.55,sapwood = 0.35)].
- minimumRelativeSugarForGrowth [= 0.5]: Minimum concentration of metabolic sugar relative to equilibrium sugar concentration for growth to occur in different tissues. This value (by default 50%) should always be larger than mortalityRelativeSugarThreshold [= 0.3], so that growth stops before starvation, and smaller than 1 (i.e. 100%), so that growth occurs under normal metabolic status.
- respirationRates [= c(leaf = 0.00260274,sapwood = 6.849315e-05,fineroot = 0.002054795)]: Maintenance respiration rates for different tissues (g gluc · g dry-1 · day-1). Sapwood respiration rates apply to parenchymatic tissue only. Values for sapwood and fine roots are used for all species, whereas the value for leaves is the default used when leaf respiration is not specified via SpParams (RERleaf).
- senescenceRates [= c(sapwood = 0.0001261398,fineroot = 0.001897231)]: Senescence rates (day-1) for sapwood and fineroots. Default are equivalent to 4.5% and 50% annual turnover for sapwood and fine roots, respectively.
- constructionCosts [= c(leaf = 1.5,sapwood = 1.47,fineroot = 1.30)]: Construction costs, including respiration and structural carbon, per dry weight of new tissue (g gluc · g dry -1).
- maximumRelativeGrowthRates [= c(leaf = 0.01,sapwood = 0.002,fineroot = 0.1)]: Maximum relative growth rates for leaves (m2 leaf ·cm-2 sapwood· day-1), sapwood (cm2 sapwood ·cm-2 sapwood· day-1) and fine roots (g dw · g dw -1 · day -1). Values for leaves and fine roots are used for all species, whereas the value for sapwood is the default used when relative growth rate is not specified via SpParams (RGRsapwoodmax).
- mortalityMode [= "density/deterministic"]: String describing how mortality is applied. Current accepted values are combinations of "cohort" vs "density" (for whole-cohort mortality vs reduction of stem density) and "deterministic" vs. "stochastic".
- mortalityBaselineRate [= 0.01]: Deterministic proportion or probability specifying the baseline reduction of cohort's density occurring in a year (for mortalityMode = "density/deterministic" or "density/stochastic").
- mortalityRelativeSugarThreshold [= 0.3]: Threshold of stem sugar concentration relative to equilibrium values (by default 30%), resulting in an increased mortality rate/probability whenever levels are below.
- mortalityRWCThreshold [= 0.3]: Threshold of stem relative water content resulting in increased mortality rate/probability whenever levels are below.

Forest dynamics:

- recruitmentMode [= "deterministic"]: String describing how recruitment is applied. Current accepted values are "deterministic" or "stochastic".
- removeDeadCohorts [= TRUE]: Boolean flag to indicate the removal of cohorts whose density is too low.
- minimumCohortDensity [= 1]: Threshold of density resulting in cohort removal.
- seedRain [= NULL]: Vector of species codes whose seed rain is to be simulated. If NULL the species identity of seed rain is taken from species currently present in the forest stand and with minimum size (see below).

- seedProductionTreeHeight [= 300]: Default minimum tree height for producing seeds (when species parameter SeedProductionHeight is missing).
- seedProductionShrubHeight [= 30]: Default minimum shrub height for producing seeds (when species parameter SeedProductionHeight is missing).
- minTempRecr [= 0]: Default threshold of minimum average temperature of the coldest month necessary for recruiting (when species parameter MinTempRecr is missing).
- minMoistureRecr [= 0.3]: Default threshold of minimum moisture index (annual precipitation over annual ETP) necessary for recruiting (when species parameter MinMoistureRecr is missing).
- minFPARRecr [= 10]: Default threshold of minimum fraction of PAR (in %) reaching the ground necessary for recruiting (when species parameter MinFPARRecr is missing).
- recrTreeDBH [= 1]: Default DBH (cm) for recruited trees (when species parameter RecrTreeDBH is missing).
- recrTreeDensity [= 100]: Default density (ind·ha-1) for recruited trees (when species parameter RecrTreeDensity is missing).
- recrTreeHeight [= 100]: Default height (cm) for recruited trees (when species parameter RecrTreeHeight is missing).
- recrShrubCover [= 1]: Default cover (%) for recruited shrubs (when species parameter RecrShrubCover is missing).
- recrShrubHeight [= 100]: Default height (cm) for recruited shrubs (when species parameter RecrShrubHeight is missing).
- recrTreeZ50 [= 100]: Default value for Z50 (mm) in recruited trees (when species parameter RecrZ50 is missing).
- recrShrubZ50 [= 50]: Default value for Z50 (mm) in recruited shrubs (when species parameter RecrZ50 is missing).
- recrTreeZ95 [= 1000]: Default value for Z95 (mm) in recruited trees (when species parameter RecrZ50 is missing).
- recrShrubZ50 [= 500]: Default value for Z95 (mm) in recruited shrubs (when species parameter RecrZ50 is missing).

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwbInput, spwb

---

defaultSoilParams *Default soil parameters*

Description

Creates a data frame with default soil physical description for model functions.

Usage

```
defaultSoilParams(n = 4)
```

Arguments

n   An integer with the number of soil layers (between two and five).

Details

The function returns a data frame with default physical soil description, with soil layers in rows. Users can change those that need to be set to other values and use the list as input for function `soil`.

Value

A data frame with layers in rows and the following columns (and default values):

- `widths` (= `c(300,700,1000,2000)`: Width of soil layers (in mm).
- `clay` (= `25`): Clay percentage for each layer (in %).
- `sand` (= `25`): Sand percentage for each layer (in %).
- `om` (= `NA`): Organic matter percentage for each layer (in %).
- `bd` (= `1.5`): Bulk density for each layer (in g/cm3).
- `rfc` (= `c(20,40,60,85)`): Percentage of rock fragment content (volume basis) for each layer.

Note

While this function is limited to five soil layers, user defined data frames can discretize soils using an unlimited number of soil layers.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

`soil`, `defaultControl`, `SpParamsMED`

Examples

```
defaultSoilParams(4)
```

`evaluation` *Evaluation of simulations results*

Description

Functions to compare model predictions against observed values.

Usage

```
evaluation_table(out, measuredData, type = "SWC", cohort = NULL,
                 temporalResolution = "day", SpParams = NULL)
evaluation_stats(out, measuredData, type="SWC", cohort = NULL,
                 temporalResolution = "day", SpParams = NULL)
evaluation_plot(out, measuredData, type="SWC", cohort = NULL,
                temporalResolution = "day", SpParams = NULL,
                plotType = "dynamics")
evaluation_metric(out, measuredData, type="SWC", cohort=NULL,
                  temporalResolution = "day", SpParams = NULL,
                  metric = "loglikelihood")
```

Arguments

`out` An object of class `spwb`, `growth` or `pwb`.

`measuredData` A data frame with observed/measured values. Dates should be in row names, whereas columns should be named according to the type of output to be evaluated (see details).

`type` A string with the kind of model output to be evaluated. Accepted values are `"SWC"` (soil moisture content), `"REW"` relative extractable water, `"ETR"` (total evapotranspiration), `"SE+TR"` (modelled soil evaporation + transpiration against observed total evapotranspiration), `"E"` (transpiration per leaf area), `"FMC"` (fuel moisture content), `"WP"` (plant water potentials) and `"BAI"` (basal area increment).

`cohort` A string of the cohort to be compared (e.g. "T1_68"). If `NULL` results for the first cohort will be evaluated.

`temporalResolution` A string to indicate the temporal resolution of the model evaluation, which can be "day", "week", "month" or "year". Observed and modelled values are aggregated temporally (using either means or sums) before comparison.

`SpParams` A data frame with species parameters (see `SpParamsMED`), only needed if `type = "FMC"`.

`plotType` Plot type to draw, either `"dynamics"` or `"scatter"`.

`metric` An evaluation metric:

- `"MAE"`: Mean absolute error.
- `"r"`: Pearson's linear correlation coefficient.

- "NSE": Nash-Sutcliffe model efficiency coefficient.
- "NSEabs": Modified Nash-Sutcliffe model efficiency coefficient (L1 norm) (Legates & McCabe 1999).
- "loglikelihood": Logarithm of the likelihood of observing the data given the model predictions, assuming independent Gaussian errors.

Details

Users should provide the appropriate columns in measuredData, depending on the type of output to be evaluated:

- "SWC" or "REW": A column named "SWC" should be present, containing soil moisture content in percent volume. When type="REW", observed values are divided by the 90% quantile, which is assumed to be the moisture content at field capacity.
- "ETR" or "SE+TR": A column named "ETR" should be present, containing stand's evapotranspiration in mm/day (or mm/week, mm/month, etc, depending on the temporal resolution). If type="ETR" observed values will be compared against modelled evapotranspiration (i.e. sum of transpiration, soil evaporation and interception loss), whereas if type= "SE+TR" observed values will be compared against the sum of transpiration and soil evaporation only.
- "E": For each plant cohort whose transpiration is to be evaluated, a column starting with "E_" and continuing with a cohort name (e.g. "E_T1_68") with transpiration in L/m2/day on a leaf area basis (or L/m2/week, L/m2/month, etc, depending on the temporal resolution).
- "FMC": For each plant cohort whose transpiration is to be evaluated, a column starting with "FCM_" and continuing with a cohort name (e.g. "FMC_T1_68") with fuel moisture content as percent of dry weight.
- "WP": For each plant cohort whose transpiration is to be evaluated, two columns, one starting with "PD_" (for pre-dawn) and the other with "MD_" (for midday), and continuing with a cohort name (e.g. "PD_T1_68"). They should contain leaf water potential values in MPa. These are compared against sunlit water potentials.
- "BAI": For each plant cohort whose growth is to be evaluated, a column starting with "BAI_" and continuing with a cohort name (e.g. "BAI_T1_68") with basal area increment in cm2/day, cm2/week, cm2/month or cm2/year, depending on the temporal resolution.

Additional columns may exist with the standard error of measured quantities. These should be named as the referred quantity, followed by "_err" (e.g. "PD_T1_68_err"), and are used to draw confidence intervals around observations.

Row names in measuredData indicate the date of measurement (in the case of days). If measurements refer to months or years, row names should also be in a "year-month-day" format, although with "01" for days and/or months (e.g. "2001-02-01" for february 2001, or "2001-01-01" for year 2001).

Value

- Function evaluation_table returns a data frame with dates, observed and predicted values.
- Function evaluation_stats returns evaluation statistics (a vector or a data frame depending on type):
  - MAE: Mean absolute error.

  - Bias: Mean deviation (positive values correspond to model overestimations).
  - r: Pearson's linear correlation coefficient.
  - NSE: Nash-Sutcliffe model efficiency coefficient.
  - NSEabs: Modified Nash-Sutcliffe model efficiency coefficient (L1 norm) (Legates & McCabe 1999).

- Function evaluation_plot returns a ggplot object.
- Function evaluation_metric returns a scalar with the desired metric.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Legates, D.R., McCabe, G.J., 1999. Evaluating the use of "goodness-of-fit" measures in hydrologic and hydroclimatic model validation. Water Resour. Res. 35, 233–241.

See Also

spwb, growth, optimization, exampleobs

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params (4 layers)
examplesoil1 = soil(defaultSoilParams(4))

#Initialize control parameters
control = defaultControl("Granier")

#Initialize input
x1 = forest2spwbInput(exampleforestMED,examplesoil1, SpParamsMED, control)

#Call simulation function
S1<-spwb(x1, examplemeteo, latitude = 41.82592, elevation = 100)

#Load observed data (in this case the same simulation results with some added error)
data(exampleobs)

#Evaluation statistics for soil water content
evaluation_stats(S1, exampleobs)

#NSE only
```

```
evaluation_metric(S1, exampleobs, metric="NSE")


#Comparison of temporal dynamics
evaluation_plot(S1, exampleobs)

#Loglikelihood value
evaluation_metric(S1, exampleobs)
```

`exampleforest` *Example forest stands*

Description

Data set for illustration of model behaviour. Includes a description of the plant cohorts of a forest stand.

Usage

```
data(exampleforestMED)
data(exampleforestUS)
```

Format

An object of class `forest` containing the description of the tree, sapling and shrub cohorts of a forest patch.

Source

DGCN (2005). Tercer Inventario Forestal Nacional (1997-2007): Catalunya. Dirección General de Conservación de la Naturaleza, Ministerio de Medio Ambiente, Madrid.

See Also

`forest`, `spwb`, `forest2spwbInput`

Examples

```
data(exampleforestMED)
```

`examplemeteo` *Example daily meteorology data*

Description

Example data set of meteorological input.

Usage

```
data(examplemeteo)
```

Format

A data frame containing daily meteorology of a location in Catalonia (Spain) for year 2001.

`MeanTemperature` Mean daily temperature (in degrees Celsius).

`MinTemperature` Minimum daily temperature (in degrees Celsius).

`MaxTemperature` Maximum daily temperature (in degrees Celsius).

`Precipitation` Daily precipitation (in mm of water).

`MeanRelativeHumidity` Mean daily relative humidity (in percent).

`MinRelativeHumidity` Minimum daily relative humidity (in percent).

`MaxRelativeHumidity` Maximum daily relative humidity (in percent).

`Radiation` Incoming radiation (in MJ/m2).

`WindSpeed` Wind speed (in m/s).

`WindDirection` Wind direction (in degrees from North).

`PET` Potential evapo-transpiration (in mm of water).

Source

Interpolated from weather station data (Spanish and Catalan meteorology agencies) using package 'meteoland'.

See Also

`spwb`

Examples

```
data(examplemeteo)
```

exampleobs *Example observed data*

Description

Example (fake) data set of variables measured in a plot.

Usage

```
data(exampleobs)
```

Format

A data frame containing daily 'observed' values for year 2001.

- `SWC` Soil moisture content (in m3/m3).
- `ETR` Total evapotranspiration (mm).
- `E_T1_148` Transpiration of Pinus halepensis cohort 'T1_148' (L/m2 of leaf area).
- `E_T2_168` Transpiration of Quercus ilex cohort 'T2_168' (L/m2 of leaf area).
- `FMC_T1_148` Fuel moisture content of Pinus halepensis cohort 'T1_148' (in percent).
- `FMC_T2_168` Fuel moisture content of Quercus ilex cohort 'T2_168' (in percent).
- `BAI_T1_148` Basal area increment for Pinus halepensis cohort 'T1_148' (in cm2).
- `BAI_T2_168` Basal area increment for Quercus ilex cohort 'T2_168' (in cm2).

Source

This data set was actually created by running a simulation and adding some gaussian error to the outputs.

See Also

evaluation

Examples

```
data(exampleobs)
```

extractSubdaily *Extracts subdaily output*

Description

Given the result of simulations, this function extracts subdaily output corresponding to each simulated day and returns it as a data frame.

Usage

```
extractSubdaily(x, output = "E", dates = NULL)
```

Arguments

`x`	An object returned by simulation functions `spwb`, `pwb` or `growth`.
`output`	See options in section details.
`dates`	A date vector indicating the subset of simulated days for which subdaily output is desired.

Details

This function only works when simulations have been carried using control option 'subdailyResults = TRUE' (see `defaultControl`). Subdaily simulation results will then be stored as elements of the a list called 'subdaily' in the simulation output. Function `extractSubdaily` will assemble subdaily results from this list and return them as a data frame. Options for parameter 'output' are the following:

- Functions pwb() and spwb(): "E","Ag","An","dEdP","RootPsi","StemPsi","LeafPsi","StemPLC","StemRWC","LeafRV "Temperature", "ExtractionInst".
- Additional options for shade and sunlit leaves in pwb() and spbw(): Either "SunlitLeaves$x" or "ShadeLeaves$x" where 'x' is one of the following: "Abs_SWR","Net_LWR","E","Ag","An","Ci","GW","VPD","Tem
- Additional options for function growth(): "GrossPhotosynthesis", "MaintenanceRespiration", "GrowthCosts", "CarbonBalance","SugarLeaf", "SugarSapwood", "StarchLeaf", "StarchSapwood","SugarTransport".

Value

A data frame with a column 'datetime' and as many columns as plant cohorts.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

`spwb`, `pwb`, `defaultControl`

fire_behaviour *Fire behaviour functions*

Description

Function `fire_FCCS()` implements a modification of the fire behavior models described for the Fuel Characteristics Classification System (FCCS) in Prichard et al. (2013). Function `fire_Rothermel()` implements Rothermel's (1972) fire behaviour model (modified from package 'Rothermel' (Giorgio Vacchiano, Davide Ascoli)).

Usage

```
fire_FCCS(FCCSpropsSI, MliveSI = as.numeric(c(90, 90, 60)),
          MdeadSI = as.numeric(c(6, 6, 6, 6, 6)),
          slope = 0, windSpeedSI = 11)
fire_Rothermel(modeltype, wSI, sSI, delta, mx_dead,
               hSI, mSI, u, windDir, slope, aspect)
```

Arguments

`FCCSpropsSI`	A data frame describing the properties of five fuel strata (canopy, shrub, herbs, dead woody and litter) returned by `fuel_FCCS`.
`MliveSI`	Moisture of live fuels (in percent of dry weight) for canopy, shrub, and herb strata.
`MdeadSI`	Moisture of dead fuels (in percent of dry weight) for canopy, shrub, herb, woody and litter strata.
`slope`	Slope (in degrees).
`windSpeedSI`	Wind speed (in m/s) at 20 ft (6 m) over vegetation (default 11 m/s = 40 km/h)
`modeltype`	'S'(tatic) or 'D'(ynamic)
`wSI`	A vector of fuel load (t/ha) for five fuel classes.
`sSI`	A vector of surface-to-volume ratio (m2/m3) for five fuel classes.
`delta`	A value of fuel bed depth (cm).
`mx_dead`	A value of dead fuel moisture of extinction (percent).
`hSI`	A vector of heat content (kJ/kg) for five fuel classes.
`mSI`	A vector of percent moisture on a dry weight basis (percent) for five fuel classes.
`u`	A value of windspeed (m/s) at midflame height.
`windDir`	Wind direction (in degrees from north). North means blowing from north to south.
`aspect`	Aspect (in degrees from north).

Details

Default moisture, slope and windspeed values are benchmark conditions used to calculate fire potentials (Sandberg et al. 2007) and map vulnerability to fire.

Value

Both functions return list with fire behavior variables. In the case of `fire_FCCS`, the function returns the variables in three blocks (lists `SurfaceFire`, `CrownFire` and `FirePotentials`), and the values are:

- `` SurfaceFire$`midflame_WindSpeed [m/s]` ``: Midflame wind speed in the surface fire.
- `SurfaceFire$phi_wind`: Spread rate modifier due to wind.
- `SurfaceFire$phi_slope`: Spread rate modifier due to slope.
- `` SurfaceFire$`I_R_surf [kJ/m2/min]` ``: Intensity of the surface fire reaction.
- `` SurfaceFire$`I_R_litter [kJ/m2/min]` ``: Intensity of the litter fire reaction.
- `` SurfaceFire$`q_surf [kJ/m2]` ``: Heat sink of the surface fire.
- `` SurfaceFire$`q_litter [kJ/m2]` ``: Heat sink of the litter fire.
- `SurfaceFire$xi_surf`: Propagating flux ratio of the surface fire.
- `SurfaceFire$xi_litter`: Propagating flux ratio of the litter fire.
- `` SurfaceFire$`ROS_surf [m/min]` ``: Spread rate of the surface fire(without accounting for faster spread in the litter layer).
- `` SurfaceFire$`ROS_litter [m/min]` ``: Spread rate of the litter fire.
- `` SurfaceFire$`ROS_windslopecap [m/min]` ``: Maximum surface fire spread rate according to wind speed.
- `` SurfaceFire$`ROS [m/min]` ``: Final spread rate of the surface fire.
- `` SurfaceFire$`I_b [kW/m]` ``: Fireline intensity of the surface fire.
- `` SurfaceFire$`FL [m]` ``: Flame length of the surface fire.
- `` CrownFire$`I_R_canopy [kJ/m2/min]` ``: Intensity of the canopy fire reaction.
- `` CrownFire$`I_R_crown [kJ/m2/min]` ``: Intensity of the crown fire reaction (adding surface and canopy reactions).
- `` CrownFire$`q_canopy [kJ/m2]` ``: Heat sink of the canopy fire.
- `` CrownFire$`q_crown [kJ/m2]` ``: Heat sink of the crown fire (adding surface and canopy heat sinks).
- `CrownFire$xi_surf`: Propagating flux ratio of the crown fire.
- `` CrownFire$`canopy_WindSpeed [m/s]` ``: Wind speed in the canopy fire (canopy top wind speed).
- `CrownFire$WAF`: Wind speed adjustment factor for crown fires.
- `` CrownFire$`ROS [m/min]` ``: Spread rate of the crown fire.
- `CrownFire$Ic_ratio`: Crown initiation ratio.
- `` CrownFire$`I_b [kW/m]` ``: Fireline intensity of the crown fire.
- `` CrownFire$`FL [m]` ``: Flame length of the crown fire.

- `FirePotentials$RP`: Surface fire reaction potential ([0-9]).
- `FirePotentials$SP`: Surface fire spread rate potential ([0-9]).
- `FirePotentials$FP`: Surface fire flame length potential ([0-9]).
- `FirePotentials$SFP`: Surface fire potential ([0-9]).
- `FirePotentials$IC`: Crown initiation potential ([0-9]).
- `FirePotentials$TC`: Crown-to-crown transmission potential ([0-9]).
- `FirePotentials$RC`: Crown fire spread rate potential ([0-9]).
- `FirePotentials$CFC`: Crown fire potential ([0-9]).

Note

Default moisture, slope and windspeed values are benchmark conditions used to calculate fire potentials (Sandberg et al. 2007) and map vulnerability to fire.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Albini, F. A. (1976). Computer-based models of wildland fire behavior: A users' manual. Ogden, UT: US Department of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station.

Rothermel, R. C. 1972. A mathematical model for predicting fire spread in wildland fuels. USDA Forest Service Research Paper INT USA.

Prichard, S. J., D. V Sandberg, R. D. Ottmar, E. Eberhardt, A. Andreu, P. Eagle, and K. Swedin. 2013. Classification System Version 3.0: Technical Documentation.

See Also

`fuel_FCCS`

Examples

```
#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Calculate fuel properties according to FCCS
fccs = fuel_FCCS(exampleforestMED, 50,100, SpParamsMED)

#Calculate fire behavior according to FCCS
fire_FCCS(fccs)

#Load fuel model parameter data
data(SFM_metric)
```

```
#Fuel stratification (returns heights in cm)
fs = fuel_stratification(exampleforestMED, SpParamsMED)

#Correct windspeed (transform heights to m)
u = 11 #m/s
umf = u*fuel_windAdjustmentFactor(fs$surfaceLayerTopHeight/100, fs$canopyBaseHeight/100,
                                  fs$canopyTopHeight/100, 60)

#Call Rothermel function using fuel model 'A6'
fire_Rothermel(modeltype="D", wSI = as.numeric(SFM_metric["A6",2:6]),
                    sSI = as.numeric(SFM_metric["A6",7:11]),
                    delta = as.numeric(SFM_metric["A6",12]),
                    mx_dead = as.numeric(SFM_metric["A6",13]),
                    hSI = as.numeric(SFM_metric["A6",14:18]),
                    mSI = c(10,10,10,30,60),
                    u=umf, windDir=0, slope=0, aspect=0)
```

---

fordyn *Forest dynamics*

Description

Function `fordyn` implements a forest dynamics model that simulates growth, mortality and recruitment for plant cohorts in a given forest stand during a period specified in the input climatic data.

Usage

```
fordyn(forest, soil, SpParams, meteo, control,
       latitude , elevation = NA, slope = NA, aspect = NA)
```

Arguments

`forest`	An object of class `forest`.
`soil`	An object of class `soil`.
`SpParams`	A data frame with species parameters (see `SpParamsMED` and `SpParamsMED`).
`meteo`	A data frame with daily meteorological data series. Row names of the data frame should correspond to date strings with format "yyyy-mm-dd" (see `Date`).
`control`	A list with default control parameters (see `defaultControl`).
`latitude`	Latitude (in degrees). Required when `x$TranspirationMode = "Sperry"`.
`elevation, slope, aspect`	Elevation above sea level (in m), slope (in degrees) and aspect (in degrees from North). Required when `x$TranspirationMode = "Sperry"`. Elevation is also required for 'Granier' if snowpack dynamics are simulated.

Details

Function `fordyn` simulates forest dynamics for annual time steps, building on other simulation functions. For each simulated year, the function performs the following steps:

1. Calls function `growth` to simulate daily water/carbon balance, growth and mortality processes.
2. Simulate recruitment (for species present in the stand).
3. Prepares the input of function `growth` for the next annual time step.
4. Store forest status and summaries.

Value

A list of class 'fordyn' with the following elements:

- `"StandSummary"`: A data frame with stand-level summaries (leaf area index, tree basal area, tree density, shrub cover, etc.) at the beginning of the simulation and after each simulated year.
- `"SpeciesSummary"`: A data frame with species-level summaries (leaf area index, tree basal area, tree density, shrub cover, etc.) at the beginning of the simulation and after each simulated year.
- `"CohortSummary"`: A data frame with cohort-level summaries (leaf area index, tree basal area, tree density, shrub cover, etc.) at the beginning of the simulation and after each simulated year.
- `"TreeTable"`: A data frame with tree-cohort data (species, density, diameter, height, etc.) at the beginning of the simulation (if any) and after each simulated year.
- `"DeadTreeTable"`: A data frame with dead tree-cohort data (species, density, diameter, height, etc.) at the beginning of the simulation and after each simulated year.
- `"ShrubTable"`: A data frame with shrub-cohort data (species, density, cover, height, etc.) at the beginning of the simulation and after each simulated year.
- `"DeadShrubTable"`: A data frame with dead shrub-cohort data (species, density, cover, height, etc.) at the beginning of the simulation (if any) and after each simulated year.
- `"ForestStructures"`: A list with the `forest` object of the stand at the beginning of the simulation and after each simulated year.
- `"GrowthResults"`: A list with the results of calling function `growth` for each simulated year.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

`growth`, `plot.growth`

Examples

```
#Load example daily meteorological data
data(examplemeteo)
#Prepare a two-year meteorological data with half precipitation during
#the second year
meteo2001 = examplemeteo
meteo2002 = examplemeteo
meteo2002$Precipitation = meteo2002$Precipitation/2
row.names(meteo2002) = seq(as.Date("2002-01-01"),
                           as.Date("2002-12-31"), by="day")
meteo_01_02 = rbind(meteo2001, meteo2002)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize control parameters
control = defaultControl("Granier")

#Initialize soil with default soil params (4 layers)
examplesoil = soil(defaultSoilParams(4))

#Call simulation function
fd<-fordyn(exampleforestMED, examplesoil,
           SpParamsMED, meteo_01_02, control,
           latitude = 41.82592, elevation = 100)

#Stand-level summaries
fd$StandSummary

#Tree table by annual steps
fd$TreeTable

#Dead tree table by annual steps
fd$DeadTreeTable
```

---

forest *Forest description*

Description

Description of a forest stand.

Usage

```
## S3 method for class 'forest'
summary(object, SpParams, mode = "MED", detailed=FALSE, ...)
```

```
## S3 method for class 'summary.forest'
print(x, digits = getOption("digits"), ...)
emptyforest(ID="", patchsize=10000, ntree = 0, nshrub = 0)
```

Arguments

`object` An object of class `forest` has the following structure:

- `ID`: An identifier of the forest stand (a string).
- `patchsize`: The area that forest stand description represents, in square meters.
- `treeData`: A data frame of tree cohorts (in rows) and the following columns:
  - `Species`: Non-negative integer for tree species identity (i.e., 0,1,2,...).
  - `Height`: Total height (in cm).
  - `DBH`: Diameter at breast height (in cm).
  - `N`: Density (number of individuals/cell).
  - `Z50`: Depth (in mm) corresponding to 50% of fine roots.
  - `Z95`: Depth (in mm) corresponding to 95% of fine roots.
- `shrubData`: A data frame of shrub cohorts (in rows) and the following columns:
  - `Species`: Non-negative integer for shrub species identity (i.e., 0,1,2,...).
  - `Height`: Total height (in cm).
  - `Cover`: Percent cover.
  - `Z50`: Depth (in mm) corresponding to 50% of fine roots.
  - `Z95`: Depth (in mm) corresponding to 95% of fine roots.
- `seedBank`: A data frame containing the abundance of seeds for each species (in rows) and the following columns:
  - `Species`: Non-negative integer for shrub species identity (i.e., 0,1,2,...).
  - `Abundance`: Abundance class (0 - none; 1 - low; 2 - medium; 3 - high; 4 - very high).
- `herbCover`: Percent cover of the herb layer.
- `herbHeight`: Mean height (in cm) of the herb layer.

`SpParams` A data frame with species parameters (see `SpParamsMED`).

`mode` Calculation mode, either "MED" or "US".

`detailed` A boolean flag to indicate that a detailed summary is desired.

`x` The object returned by `summary.forest`.

`digits` Minimal number of significant digits.

`...` Additional parameters for functions `summary` and `print`.

`ID` An identifier of the forest stand (a string).

`patchsize` The area of the forest stand, in square meters.

`ntree, nshrub` Number of tree and shrub cohorts, respectively.

Details

Function `summary.forest` can be used to summarize a `forest` object in the console. Function `emptyforest` creates an empty `forest` object.

Value

Function `summary.forest` returns a list with the basal area and LAI of the forest, either expressed as totals or divided among life stages and species. Function `emptyforest` returns an empty `forest` object.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

`exampleforestMED`, `forest_mergeTrees`

Examples

```
data(exampleforestMED)
data(SpParamsMED)

summary(exampleforestMED, SpParamsMED)
```

`Forest manipulation` *Forest utility functions*

Description

Functions to manipulate a `forest` object.

Usage

```
forest_mergeTrees(x, byDBHclass = TRUE)
forest_mergeShrubs(x, byHeightclass = TRUE)
```

Arguments

`x`	An object of class `forest`.
`byDBHclass`	Boolean flag to indicate that 5-cm tree DBH classes should be kept separated.
`byHeightclass`	Boolean flag to indicate that 10-cm shrub height classes should be kept separated.

Value

Another `forest` object with merged trees or shrubs, depending on the function.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwb, forest, fordyn, summary.forest

---

fuel_properties *Fuel stratification and fuel characteristics*

Description

Function fuel_stratification provides a stratification of the stand into understory and canopy strata. Function fuel_FCCS calculates fuel characteristics from a forest object following an adaptation of the protocols described for the Fuel Characteristics Classification System (Prichard et al. 2013). Function fuel_windAdjustmentFactor determines the adjustment factor of wind for surface fires, according to Andrews (2012).

Usage

```
fuel_stratification(object, SpParams, gdd = NA, mode = "MED",
                      heightProfileStep = 10.0, maxHeightProfile = 5000.0,
                      bulkDensityThreshold = 0.05)
fuel_FCCS(object, ShrubCover, CanopyCover, SpParams, cohortFMC = as.numeric(c()),
        gdd = NA, mode = "MED", heightProfileStep = 10, maxHeightProfile = 5000,
         bulkDensityThreshold = 0.05, depthMode = "crownaverage")
fuel_windAdjustmentFactor(topShrubHeight, bottomCanopyHeight, topCanopyHeight,
                          canopyCover)
```

Arguments

- `object` An object of class forest
- `ShrubCover` Total shrub cover (in percent) of the stand.
- `CanopyCover` Total canopy cover (in percent) of the stand.
- `SpParams` A data frame with species parameters (see SpParamsMED).
- `cohortFMC` A numeric vector of (actual) fuel moisture content by cohort (e.g. can be taken as a row of the matrix returned by moisture_cohortFMC).
- `gdd` Growth degree-days.
- `mode` Calculation mode, either "MED" or "US".
- `heightProfileStep` Precision for the fuel bulk density profile.
- `maxHeightProfile` Maximum height for the fuel bulk density profile.
- `bulkDensityThreshold` Minimum fuel bulk density to delimit fuel strata.
- `depthMode` Specifies how fuel depth (and therefore canopy and understory bulk density) should be estimated:

- `"crownaverage"`: As weighed average of crown lengths using loadings as weights.
- `"profile"`: As the difference of base and top heights in bulk density profiles.
- `"absoluteprofile"`: As the difference of absolute base and absolute top heights in bulk density profiles.

`topShrubHeight` Shrub stratum top height (in m).

`bottomCanopyHeight` Canopy base height (in m).

`topCanopyHeight` Canopy top height (in m).

`canopyCover` Canopy percent cover.

Details

Details are described in a vignette.

Value

Function `fuel_FCCS` returns a data frame with five rows corresponding to fuel layers: `canopy`, `shrub`, `herb`, `woody` and `litter`. Columns correspond fuel properties:

- `w`: Fine fuel loading (in kg/m2).
- `cover`: Percent cover.
- `hbc`: Height to base of crowns (in m).
- `htc`: Height to top of crowns (in m).
- `delta`: Fuel depth (in m).
- `rhob`: Fuel bulk density (in kg/m3).
- `rhop`: Fuel particle density (in kg/m3).
- `PV`: Particle volume (in m3/m2).
- `beta`: Packing ratio (unitless).
- `betarel`: Relative packing ratio (unitless).
- `etabetarel`: Reaction efficiency (unitless).
- `sigma`: Surface area-to-volume ratio (m2/m3).
- `pDead`: Proportion of dead fuels.
- `FAI`: Fuel area index (unitless).
- `h`: High heat content (in kJ/kg).
- `RV`: Reactive volume (in m3/m2).
- `MinFMC`: Minimum fuel moisture content (as percent over dry weight).
- `MaxFMC`: Maximum fuel moisture content (as percent over dry weight).

Function `fuel_stratification` returns a list with the following items:

- `surfaceLayerBaseHeight`: Base height of crowns of shrubs in the surface layer (in cm).
- `surfaceLayerTopHeight`: Top height of crowns of shrubs in the surface layer (in cm).
- `understoryLAI`: Cumulated LAI of the understory layer (i.e. leaf area comprised between surface layer base and top heights).
- `canopyBaseHeight`: Base height of tree crowns in the canopy (in cm).
- `canopyTopHeight`: Top height of tree crowns in the canopy (in cm).
- `canopyLAI`: Cumulated LAI of the canopy (i.e. leaf area comprised between canopy base and top heights).

Function `fuel_cohortFineFMC` returns a list with three matrices (for leaves, twigs and fine fuels). Each of them contains live moisture content values for each day (in rows) and plant cohort (in columns).

Function `fuel_windAdjustmentFactor` returns a value between 0 and 1.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Andrews, P. L. 2012. Modeling wind adjustment factor and midflame wind speed for Rothermel's surface fire spread model. USDA Forest Service - General Technical Report RMRS-GTR:1–39.

Prichard, S. J., D. V Sandberg, R. D. Ottmar, E. Eberhardt, A. Andreu, P. Eagle, and K. Swedin. 2013. Classification System Version 3.0: Technical Documentation.

Reinhardt, E., D. Lutes, and J. Scott. 2006. FuelCalc: A method for estimating fuel characteristics. Pages 273–282.

See Also

fire_FCCS, spwb

Examples

```
#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Show stratification of fuels
fuel_stratification(exampleforestMED, SpParamsMED)

#Calculate fuel properties according to FCCS
fccs = fuel_FCCS(exampleforestMED, 50,100, SpParamsMED)
fccs

fuel_windAdjustmentFactor(fccs$htc[2], fccs$hbc[1], fccs$htc[1], fccs$cover[1])
```

growth *Forest growth*

Description

Function growth is a process-based model that performs energy, water and carbon balances; and determines changes in water/carbon pools, functional variables (leaf area, sapwood area, root area) and structural ones (tree diameter, tree height, shrub cover) for woody plant cohorts in a given forest stand during a period specified in the input climatic data.

Usage

```
growth(x, meteo,  latitude, elevation = NA, slope = NA, aspect = NA)
```

Arguments

x
: An object of class growthInput.

meteo
: A data frame with daily meteorological data series. Row names of the data frame should correspond to date strings with format "yyyy-mm-dd" (see Date). When x$TranspirationMode = "Granier" the following columns are required:
  - MeanTemperature: Mean temperature (in degrees Celsius).
  - Precipitation: Precipitation (in mm).
  - Radiation: Solar radiation (in MJ/m2/day), required only if snowpack = TRUE.
  - PET: Potential evapotranspiration (in mm).
  - WindSpeed: Wind speed (in m/s). If not available, this column can be left with NA values.

  When x$TranspirationMode = "Sperry" the following columns are required:
  - MeanTemperature: Mean temperature(in degrees Celsius).
  - MinTemperature: Minimum temperature (in degrees Celsius).
  - MaxTemperature: Maximum temperature (in degrees Celsius).
  - MinRelativeHumidity: Minimum relative humidity (in percent).
  - MaxRelativeHumidity: Maximum relative humidity (in percent).
  - Precipitation: Precipitation (in mm).
  - Radiation: Solar radiation (in MJ/m2/day).
  - WindSpeed: Wind speed (in m/s). If not available, this column can be left with NA values.
  - CO2: Atmospheric (abovecanopy) CO2 concentration (in ppm). This column may not exist, or can be left with NA values. In both cases simulations will assume a constant value specified in defaultControl.

latitude
: Latitude (in degrees). Required when x$TranspirationMode = "Sperry".

elevation, slope, aspect
: Elevation above sea level (in m), slope (in degrees) and aspect (in degrees from North). Required when x$TranspirationMode = "Sperry". Elevation is also required for 'Granier' if snowpack dynamics are simulated.

Details

Detailed model description is available in the vignettes section. Simulations using the 'Sperry' transpiration mode are computationally much more expensive than those using the simple transpiration mode.

Value

A list of class 'growth' with the following elements:

- `"latitude"`: Latitude (in degrees) given as input.
- `"topography"`: Vector with elevation, slope and aspect given as input.
- `"growthInput"`: A copy of the object `x` of class `growthInput` given as input.
- `"WaterBalance"`: A data frame where different water balance variables (see `spwb`).
- `"EnergyBalance"`: A data frame with the daily values of energy balance components for the soil and the canopy (only for `transpirationMode = "Sperry"`; see `spwb`).
- `"Temperature"`: A data frame with the daily values of minimum/mean/maximum temperatures for the atmosphere (input), canopy and soil (only for `transpirationMode = "Sperry"`; see `spwb`).
- `"Soil"`: A data frame where different soil variables (see `spwb`).
- `"Stand"`: A data frame where different stand-level variables (see `spwb`).
- `"Plants"`: A list of daily results for plant cohorts (see `spwb`).
- `"SunlitLeaves"` and `"ShadeLeaves"`: A list with daily results for sunlit and shade leaves (only for `transpirationMode = "Sperry"`; see `spwb`).
- `"PlantCarbonBalance"`: A list of daily carbon balance results for plant cohorts, with elements:
  - `"GrossPhotosynthesis"`: Daily gross photosynthesis per dry weight of living biomass (g gluc · g dry-1).
  - `"MaintentanceRespiration"`: Daily maintenance respiration per dry weight of living biomass (g gluc · g dry-1).
  - `"GrowthCosts"`: Daily growth costs per dry weight of living biomass (g gluc · g dry-1).
  - `"RootExudation"`: Root exudation per dry weight of living biomass (g gluc · g dry-1).
  - `"CarbonBalance"`: Daily plant carbon balance (photosynthesis - maintenance respiration - growth costs - root exudation) per dry weight of living biomass (g gluc · g dry-1).
  - `"SugarLeaf"`: Sugar concentration (mol·l-1) in leaves.
  - `"StarchLeaf"`: Starch concentration (mol·l-1) in leaves.
  - `"SugarSapwood"`: Sugar concentration (mol·l-1) in sapwood.
  - `"StarchSapwood"`: Starch concentration (mol·l-1) in sapwood.
  - `"SugarTransport"`: Average instantaneous rate of carbon transferred between leaves and stem compartments via floem (mol gluc·s-1).
  - `"LeafPI0"`: Osmotic potential at full turgor of symplastic leaf tissue (MPa).
  - `"StemPI0"`: Osmotic potential at full turgor of symplastic stem tissue (MPa).
- `"PlantStructure"`: A list of daily area and biomass values for compartments of plant cohorts, with elements:

  - "LeafArea": Daily amount of leaf area (in m2) for an average individual of each plant cohort.
  - "SapwoodArea": Daily amount of sapwood area (in cm2) for an average individual of each plant cohort.
  - "FineRootArea": Daily amount of fine root area (in m2) for an average individual of each plant cohort (only for transpirationMode = "Sperry").
  - "SapwoodBiomass": Daily amount of sapwood biomass (in g dry) for an average individual of each plant cohort.
  - "LeafBiomass": Daily amount of leaf biomass (in g dry) for an average individual of each plant cohort.
  - "FineRootBiomass": Daily amount of fine root biomass (in g dry) for an average individual of each plant cohort.
  - "LabileBiomass": Daily amount of labile C biomass, i.e. starch and sugars (in g dry) for an average individual of each plant cohort.
  - "TotalLivingBiomass": Daily amount of total living biomass, i.e. excluding heartwood, (in g dry) for an average individual of each plant cohort.
- "PlantGrowth": A list of daily growth results for plant cohorts, with elements:
  - "LAgrowth": Leaf area growth relative to sapwood area (in m2·cm-2·day-1) for an average individual of each plant cohort.
  - "SAgrowth": Sapwood area relative growth rate (in cm2·cm-2·day-1) for an average individual of each plant cohort.
  - "FRAgrowth": Fine root area growth relative to sapwood area (in m2·cm-2·day-1) for an average individual of each plant cohort (only for transpirationMode = "Sperry").
- "subdaily": A list of objects of class growth_day, one per day simulated (only if required in control parameters, see defaultControl).

Note

Objects x and soil are modified during the simulation.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

growthInput, growth_day, plot.growth

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)
```

```
#Initialize control parameters
control = defaultControl("Granier")

#Initialize soil with default soil params (4 layers)
examplesoil1 = soil(defaultSoilParams(4))

#Initialize vegetation input
x1 = forest2growthInput(exampleforestMED, examplesoil1, SpParamsMED, control)

#Call simulation function
G1<-growth(x1, examplemeteo, latitude = 41.82592, elevation = 100)

## Not run:
#Switch to 'Sperry' transpiration mode
control = defaultControl("Sperry")

#Initialize soil with default soil params (4 layers)
examplesoil2 = soil(defaultSoilParams(4))

#Initialize vegetation input
x2 = forest2growthInput(exampleforestMED,examplesoil2, SpParamsMED, control)

#Call simulation function (11 days)
d = 100:110
G2<-growth(x2, examplemeteo[d,], latitude = 41.82592, elevation = 100)

## End(Not run)
```

---

hydrology_rainInterception
Rainfall interception

Description

Function hydrology_rainInterception calculates the amount of rainfall intercepted daily by the canopy, given a rainfall and canopy characteristics. Two canopy interception models are currently available: the sparse Gash (1995) model and the Liu (2001) model. In both cases the current implementation assumes no trunk interception.

Usage

```
hydrology_rainInterception(Rainfall, Cm, p, ER=0.05, method="Gash1995")
hydrology_erFactor(doy, pet, prec, Rconv = 5.6, Rsyn = 1.5)
hydrology_interceptionPlot(x, SpParams, ER = 0.05, gdd = NA, throughfall = FALSE)
```

Arguments

Rainfall	A numeric vector of (daily) rainfall.

Cm	Canopy water storage capacity.
p	Proportion of throughfall (normally 1 - c, where c is the canopy cover).
ER	The ratio of evaporation rate to rainfall rate.
method	Rainfall interception method (either "Gash1995" or "Liu2001").
doy	Day of the year.
pet	Potential evapotranspiration for a given day (mm).
prec	Precipitation for a given day (mm).
Rconv, Rsyn	Rainfall rate for convective storms and synoptic storms, respectively, in mm/h.
x	An object of class spwbInput.
SpParams	A data frame with species parameters (see SpParamsMED and SpParamsMED).
gdd	Growth degree days (in Celsius).
throughfall	Boolean flag to plot relative throughfall instead of percentage of intercepted rainfall.

Details

Function hydrology_rainInterception can accept either vectors or scalars as parameters Cm, p and ER. If they are supplied as vectors they should be of the same length as Rainfall.

Function hydrology_erFactor calculates the evaporation-to-rainfall ratio for input values of potential evapotranspiration and rainfall, while accounting for seasonal variation in rainfall intensity (mm/h). Default values Rconv = 5.6 and Rsyn = 1.5 come from Miralles et al. (2010).

Value

Function hydrology_rainInterception returns a vector of the same length as Rainfall containing intercepted rain values. Function hydrology_erFactor returns a scalar with the evaporation-to-rainfall ratio.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Liu (2001). Evaluation of the Liu model for predicting rainfall interception in forests world-wide. - Hydrol. Process. 15: 2341-2360.

Gash (1979). An analytical model of rainfall interception by forests. - Quarterly Journal of the Royal Meteorological Society.

Gash et al. (1995). Estimating sparse forest rainfall interception with an analytical model. - Journal of Hydrology.

Miralles DG, Gash JH, Holmes TRH, et al (2010) Global canopy interception from satellite observations. J Geophys Res 115:D16122. doi: 10.1029/2009JD013530.

See Also

spwb

Examples

```
#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Draw rainfall interception for two values of the E/R ratio
hydrology_interceptionPlot(exampleforestMED, SpParamsMED, ER = c(0.05, 0.2))
```

---

hydrology_soilWaterInputs
Soil water processes

Description

High-level functions for hydrological processes. Function hydrology_soilWaterInputs performs canopy water interception and snow accumulation/melt. Function hydrology_soilInfiltrationPercolation performs soil infiltration and percolation from the input given by the previous function.

Usage

```
hydrology_soilWaterInputs(soil, soilFunctions, prec, er, tday, rad,
                          elevation, Cm, LgroundPAR, LgroundSWR, runon = 0,
                          snowpack = TRUE, modifySoil = TRUE)
hydrology_soilInfiltrationPercolation(soil, soilFunctions, waterInput,
                                      rockyLayerDrainage = TRUE, modifySoil = TRUE)
```

Arguments

soil	A list containing the description of the soil (see soil).
soilFunctions	Soil water retention curve and conductivity functions, either 'SX' (for Saxton) or 'VG' (for Van Genuchten).
prec	Precipitation for a given day (mm)
waterInput	Soil water input for a given day (mm).
er	The ratio of evaporation rate to rainfall rate.
tday	Average day temperature (ºC).
rad	Solar radiation (in MJ/m2/day).
elevation	Altitude above sea level (m).
Cm	Canopy water storage capacity.
LgroundPAR	Proportion of photosynthetically-acvive radiation (PAR) reaching the ground.
LgroundSWR	Proportion of short-wave radiation (SWR) reaching the ground.
runon	Surface water amount running on the target area from upslope (in mm).

`snowpack` Boolean flag to indicate the simulation of snow accumulation and melting.

`rockyLayerDrainage` Boolean flag to indicate the simulation of drainage from rocky layers (> 95% of rocks).

`modifySoil` Boolean flag to indicate that the input `soil` object should be modified during the simulation.

Details

The function simulates different vertical hydrological processes, which are described separately in other functions. If `modifySoil = TRUE` the function will modify the `soil` object (including both soil moisture and the snowpack on its surface) as a result of simulating hydrological processes.

Value

Function `hydrology_soilWaterInputs` returns a named vector with the following elements, all in mm:

`Rain` Precipitation as rainfall.

`Snow` Precipitation as snow.

`Interception` Rainfall water intercepted by the canopy and evaporated.

`NetRain` Rainfall reaching the ground.

`Snowmelt` Snow melted during the day, and added to the water infiltrated.

`Runon` Surface water amount running on the target area from upslope.

`Input` Total soil input, including runon, snowmelt and net rain.

Function `hydrology_soilInfiltrationPercolation` returns a named vector with the following elements, all in mm:

`Infiltration` Water infiltrated into the soil (i.e. throughfall + runon + snowmelt - runoff).

`Runoff` Surface water leaving the target area.

`DeepDrainage` Water leaving the target soil towards the water table.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

`spwb_day`, `hydrology_rainInterception`, `hydrology_soilEvaporation` light *Radiation transfer functions*

Description

Functions light_layerIrradianceFraction and light_layerIrradianceFractionBottomUp calculate the fraction of above-canopy irradiance (and the soil irradiance, respectively) reaching each vegetation layer. Function light_layerSunlitFraction calculates the proportion of sunlit leaves in each vegetation layer. Function light_cohortSunlitShadeAbsorbedRadiation calculates the amount of radiation absorbed by cohort and vegetation layers, while differentiating between sunlit and shade leaves.

Usage

```
light_layerIrradianceFraction(LAIme, LAImd, LAImx, k, alpha,
                              trunkExtinctionFraction = 0.1)
light_layerIrradianceFractionBottomUp(LAIme, LAImd, LAImx, k, alpha,
                              trunkExtinctionFraction = 0.1)
light_layerSunlitFraction(LAIme, LAImd, kb)
light_cohortSunlitShadeAbsorbedRadiation(Ib0, Id0, Ibf, Idf, beta,
                                         LAIme, LAImd,
                                         kb, kd, alpha, gamma)
light_instantaneousLightExtinctionAbsortion(LAIme, LAImd, LAImx,
                                            kPAR, alphaSWR, gammaSWR,
                                            ddd, ntimesteps = 24,
                                            trunkExtinctionFraction = 0.1)
light_longwaveRadiationSHAW(LAIme, LAImd, LAImx,
                            LWRatm, Tsoil, Tair, trunkExtinctionFraction = 0.1)
light_cohortAbsorbedSWRFraction(z, x, SpParams, gdd = NA)
```

Arguments

LAIme	A numeric matrix of live expanded LAI values per vegetation layer (row) and cohort (column).
LAImd	A numeric matrix of dead LAI values per vegetation layer (row) and cohort (column).
LAImx	A numeric matrix of maximum LAI values per vegetation layer (row) and cohort (column).
k	A vector of light extinction coefficients.
kb	A vector of direct light extinction coefficients.
kd	A vector of diffuse light extinction coefficients.
Ib0	Above-canopy direct incident radiation.
Id0	Above-canopy diffuse incident radiation.
Ibf	Fraction of above-canopy direct radiation reaching each vegetation layer.

Idf	Fraction of above-canopy diffuse radiation reaching each vegetation layer.
alpha	A vecfor of leaf absorbance by species.
beta	Solar elevation (in radians).
gamma	Vector of canopy reflectance values.
kPAR	A vector of visible light extinction coefficients for each cohort.
alphaSWR	A vecfor of hort-wave absorbance coefficients for each cohort.
gammaSWR	A vector of short-wave reflectance coefficients (albedo) for each cohort.
ddd	A dataframe with direct and diffuse radiation for different subdaily time steps (see function radiation_directDiffuseDay in package meteoland).
ntimesteps	Number of subdaily time steps.
trunkExtinctionFraction	Fraction of extinction due to trunks (for winter deciduous forests).
LWRatm	Atmospheric downward long-wave radiation (W/m2).
Tsoil	Soil temperature (Celsius).
Tair	Canopy layer air temperature vector (Celsius).
x	An object of class forest
SpParams	A data frame with species parameters (see SpParamsMED).
z	A numeric vector with height values.
gdd	Growth degree days.

Details

Functions for short-wave radiation are adapted from Anten & Bastiaans (2016), whereas long-wave radiation balance follows Flerchinger et al. (2009). Vegetation layers are assumed to be ordered from bottom to top.

Value

Functions light_layerIrradianceFraction, light_layerIrradianceFractionBottomUp and light_layerSunlitFraction return a numeric vector of length equal to the number of vegetation layers. Function light_cohortSunlitShadeAbsorbedRadiation returns a list with two elements (matrices): I_sunlit and I_shade.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Anten, N.P.R., Bastiaans, L., 2016. The use of canopy models to analyze light competition among plants, in: Hikosaka, K., Niinemets, U., Anten, N.P.R. (Eds.), Canopy Photosynthesis: From Basics to Application. Springer, pp. 379–398.

Flerchinger, G. N., Xiao, W., Sauer, T. J., Yu, Q. 2009. Simulation of within-canopy radiation exchange. NJAS - Wageningen Journal of Life Sciences 57 (1): 5–15. https://doi.org/10.1016/j.njas.2009.07.004.

See Also

spwb

Examples

```
LAI = 2
nlayer = 10
LAIlayerlive = matrix(rep(LAI/nlayer,nlayer),nlayer,1)
LAIlayerdead = matrix(0,nlayer,1)
kb = 0.8
kd_PAR = 0.5
kd_SWR = kd_PAR/1.35
alpha_PAR = 0.9
gamma_PAR = 0.04
gamma_SWR = 0.05
alpha_SWR = 0.7

Ibfpar = light_layerIrradianceFraction(LAIlayerlive,LAIlayerdead,LAIlayerlive,kb, alpha_PAR)
Idfpar = light_layerIrradianceFraction(LAIlayerlive,LAIlayerdead,LAIlayerlive,kd_PAR, alpha_PAR)
Ibfswr = light_layerIrradianceFraction(LAIlayerlive,LAIlayerdead,LAIlayerlive,kb, alpha_SWR)
Idfswr = light_layerIrradianceFraction(LAIlayerlive,LAIlayerdead,LAIlayerlive,kd_SWR, alpha_SWR)
fsunlit = light_layerSunlitFraction(LAIlayerlive, LAIlayerdead, kb)
SHarea = (1-fsunlit)*LAIlayerlive[,1]
SLarea = fsunlit*LAIlayerlive[,1]

par(mar=c(4,4,1,1), mfrow=c(1,2))
plot(Ibfpar*100, 1:nlayer,type="l", ylab="Layer",
     xlab="Percentage of irradiance", xlim=c(0,100), ylim=c(1,nlayer), col="dark green")
lines(Idfpar*100, 1:nlayer, col="dark green", lty=2)
lines(Ibfswr*100, 1:nlayer, col="red")
lines(Idfswr*100, 1:nlayer, col="red", lty=2)

plot(fsunlit*100, 1:nlayer,type="l", ylab="Layer",
     xlab="Percentage of leaves", xlim=c(0,100), ylim=c(1,nlayer))
lines((1-fsunlit)*100, 1:nlayer, lty=2)


solarElevation = 0.67
SWR_direct = 1100
SWR_diffuse = 300
PAR_direct = 550
PAR_diffuse = 150


abs_PAR = light_cohortSunlitShadeAbsorbedRadiation(PAR_direct, PAR_diffuse,
                        Ibfpar, Idfpar, beta = solarElevation,
                        LAIlayerlive, LAIlayerdead, kb, kd_PAR, alpha_PAR, gamma_PAR)
abs_SWR = light_cohortSunlitShadeAbsorbedRadiation(SWR_direct, SWR_diffuse,
                        Ibfswr, Idfswr, beta = solarElevation,
                        LAIlayerlive, LAIlayerdead, kb, kd_SWR, alpha_SWR, gamma_SWR)
par(mar=c(4,4,1,1), mfrow=c(1,2))
absRadSL = abs_SWR$I_sunlit[,1]
```

```
absRadSH = abs_SWR$I_shade[,1]
lambda = 546.6507
QSL = abs_PAR$I_sunlit[,1]*lambda*0.836*0.01
QSH = abs_PAR$I_shade[,1]*lambda*0.836*0.01
plot(QSL, 1:nlayer,type="l", ylab="Layer",
     xlab="Absorbed PAR quantum flux per leaf area", ylim=c(1,nlayer), col="dark green",
     xlim=c(0,max(QSL)))
lines(QSH, 1:nlayer, col="dark green", lty=2)
plot(absRadSL, 1:nlayer,type="l", ylab="Layer",
     xlab="Absorbed SWR per leaf area (W/m2)", ylim=c(1,nlayer), col="red",
     xlim=c(0,max(absRadSL)))
lines(absRadSH, 1:nlayer, col="red", lty=2)
```

---

modelInput *Input for simulation models*

Description

Functions forest2spwbInput and forest2growthInput take an object of class forest and calculate input data for functions spwb, pwb and growth, respectively. Functions spwbInput and growthInput do the same but starting from different input data. Function forest2aboveground calculates aboveground variables that may be used in spwbInput and growthInput functions. Function forest2belowground calculates belowground fine root distribution.

Usage

```
forest2aboveground(x, SpParams, gdd = NA, mode = "MED")
forest2belowground(x, soil)
forest2growthInput(x, soil, SpParams, control)
forest2spwbInput(x, soil, SpParams, control, mode = "MED")
growthInput(above,  Z50, Z95, soil, SpParams, control)
spwbInput(above,  Z50, Z95, soil, SpParams, control)
```

Arguments

x	An object of class forest.
SpParams	A data frame with species parameters (see SpParamsMED and SpParamsMED).
gdd	Growth degree days to account for leaf phenology effects (in Celsius). This should be left NA in most applications.
mode	Calculation mode, either "MED" or "US".
soil	An object of class soil.
control	A list with default control parameters (see defaultControl).
above	A data frame with aboveground plant information (see the return value of forest2aboveground below). In the case of spwbInput the variables should include SP, N, LAI_live, LAI_dead, H and CR. In the case of growthInput variables should include DBH and Cover.

Z50, Z95 Numeric vectors with cohort depths (in mm) corresponding to 50% and 95% of fine roots.

Details

Functions forest2spwbInput and forest2abovegroundInput extracts height and species identity from plant cohorts of x, and calculate leaf area index and crown ratio.forest2spwbInput also calculates the distribution of fine roots across soil. Both forest2spwbInput and spwbInput find parameter values for each plant cohort according to the parameters of its species as specified in SpParams. If control$transpirationMode = "Sperry" the functions also estimate the maximum conductance of rhizosphere, root xylem and stem xylem elements.

Value

Function forest2aboveground() returns a data frame with the following columns (rows are identified as specified by function plant_ID):

- SP: Species identity (an integer) (first species is 0).
- N: Cohort density (ind/ha) (see function plant_density).
- DBH: Tree diameter at breast height (cm).
- H: Plant total height (cm).
- CR: Crown ratio (crown length to total height) (between 0 and 1).
- LAI_live: Live leaf area index (m2/m2) (one-side leaf area relative to plot area), includes leaves in winter dormant buds.
- LAI_expanded: Leaf area index of expanded leaves (m2/m2) (one-side leaf area relative to plot area).
- LAI_dead: Dead leaf area index (m2/m2) (one-side leaf area relative to plot area).

Functions forest2spwbInput() and spwbInput() return a list of class spwbInput with the following elements (rows of data frames are identified as specified by function plant_ID):

- control: List with control parameters (see defaultControl).
- canopy: A list of stand-level state variables.
- cohorts: A data frame with cohort information, with columns SP and Name.
- above: A data frame with columns H, CR and LAI (see function forest2aboveground).
- below: A data frame with columns Z50, Z95. If control$transpirationMode = "Sperry" additional columns are fineRootBiomass and coarseRootSoilVolume.
- belowLayers: A list. If control$transpirationMode = "Granier" it contains elements:
  - V: A matrix with the proportion of fine roots of each cohort (in rows) in each soil layer (in columns).
  - L: A matrix with the length of coarse roots of each cohort (in rows) in each soil layer (in columns).
  - Wpool: A matrix with the soil moisture relative to field capacity around the rhizosphere of each cohort (in rows) in each soil layer (in columns).

  If control$transpirationMode = "Sperry" there are the following additional elements:

  - VGrhizo_kmax: A matrix with maximum rhizosphere conductance values of each cohort (in rows) in each soil layer (in columns).
  - VGroot_kmax: A matrix with maximum root xylem conductance values of each cohort (in rows) in each soil layer (in columns).
  - RhizoPsi: A matrix with the water potential around the rhizosphere of each cohort (in rows) in each soil layer (in columns).

- paramsPhenology: A data frame with leaf phenology parameters:
  - PhenologyType: Leaf phenology type.
  - LeafDuration: Leaf duration (in years).
  - Sgdd: Degree days needed for leaf budburst (for winter decideous species).
  - Tbgdd: Base temperature for the calculation of degree days to leaf budburst.
  - Ssen: Degree days corresponding to leaf senescence.
  - Phsen: Photoperiod corresponding to start counting senescence degree-days.
  - Tbsen: Base temperature for the calculation of degree days to leaf senescence.

- paramsAnatomy: A data frame with plant anatomy parameters for each cohort (only if control$transpirationMode = "Sperry"):
  - Hmax: Maximum plant height (cm).
  - Hmed: Median plant height (cm).
  - Al2As: Leaf area to sapwood area ratio (in m2·m-2).
  - SLA: Specific leaf area (mm2/mg = m2/kg).
  - LeafWidth: Leaf width (in cm).
  - LeafDensity: Density of leaf tissue (dry weight over volume).
  - WoodDensity: Density of wood tissue (dry weight over volume).
  - FineRootDensity: Density of fine root tissue (dry weight over volume).
  - SRL: Specific Root length (cm·g-1).
  - RLD: Root length density (cm·cm-3).
  - r635: Ratio between the weight of leaves plus branches and the weight of leaves alone for branches of 6.35 mm.

- paramsInterception: A data frame with rain interception and light extinction parameters for each cohort:
  - kPAR: PAR extinction coefficient.
  - g: Canopy water retention capacity per LAI unit (mm/LAI).

  If control$transpirationMode = "Sperry" additional columns are:
  - gammaSWR: Reflectance (albedo) coefficient for SWR .
  - alphaSWR: Absorbance coefficient for SWR .

- paramsTranspiration: A data frame with parameters for transpiration and photosynthesis. If control$transpirationMode = "Granier", columns are:
  - Tmax_LAI: Coefficient relating LAI with the ratio of maximum transpiration over potential evapotranspiration.
  - Tmax_LAIsq: Coefficient relating squared LAI with the ratio of maximum transpiration over potential evapotranspiration.
  - Psi_Extract: Water potential corresponding to 50% relative conductance (in MPa).

- Psi_Critic: Water potential corresponding to 50% of stem cavitation (in MPa).
- WUE: Water use efficiency for carbon assimilation (g C/mm water).
- pRootDisc: Proportion of whole-plant conductance leading to disconnection from soil.

If control$transpirationMode = "Sperry" columns are:

- Gswmax: Maximum stomatal conductance to water vapor (in mol H2O·m-2·s-1).
- Vmax298: Maximum Rubisco carboxilation rate at 25ºC (in micromol CO2·s-1·m-2).
- Jmax298: Maximum rate of electron transport at 25ºC (in micromol photons·s-1·m-2).
- Kmax_stemxylem: Sapwood-specific hydraulic conductivity of stem xylem (in kg H2O·s-1·m-2).
- Kmax_rootxylem: Sapwood-specific hydraulic conductivity of root xylem (in kg H2O·s-1·m-2).
- VCleaf_kmax: Maximum leaf hydraulic conductance.
- VCleaf_c, VCleaf_d: Parameters of the leaf vulnerability curve.
- VCstem_kmax: Maximum stem xylem conductance.
- VCstem_c, VCstem_d: Parameters of the stem xylem vulnerability curve.
- VCroot_c, VCroot_d: Parameters of the root xylem vulnerability curve.
- Plant_kmax: Maximum whole-plant conductance.

- paramsWaterStorage: A data frame with plant water storage parameters for each cohort (only if control$transpirationMode = "Sperry"):
  - LeafPI0: Osmotic potential at full turgor of leaves (MPa).
  - LeafEPS: Modulus of elasticity (capacity of the cell wall to resist changes in volume in response to changes in turgor) of leaves (MPa).
  - LeafAF: Apoplastic fraction (proportion of water outside the living cells) in leaves.
  - Vleaf: Storage water capacity in leaves, per leaf area (L/m2).
  - StemPI0: Osmotic potential at full turgor of symplastic xylem tissue (MPa).
  - StemEPS: Modulus of elasticity (capacity of the cell wall to resist changes in volume in response to changes in turgor) of symplastic xylem tissue (Mpa).
  - StemAF: Apoplastic fraction (proportion of water outside the living cells) in stem xylem.
  - Vstem: Storage water capacity in sapwood, per leaf area (L/m2).
- internalPhenology and internalWater: data frames to store internal state variables.

Functions forest2growthInput and growthInput return a list of class growthInput with the same elements as spwbInput, but with additional information.

- Element above includes the following additional columns:
  - LA_live: Live leaf area per individual (m2/ind).
  - LA_dead: Dead leaf area per individual (m2/ind).
  - SA: Live sapwood area per individual (cm2/ind).
- paramsGrowth: A data frame with growth parameters for each cohort:
  - WoodC: Wood carbon content per dry weight (g C /g dry).
  - RGRsapwoodmax: Maximum relative growth rate (in basal area or sapwood area) (in cm2·cm-2).
  - fHDmin: Minimum value of the height-to-diameter ratio (dimensionless).

    - fHDmax: Maximum value of the height-to-diameter ratio (dimensionless).
- paramsAllometry: A data frame with allometric parameters for each cohort:
    - Aash: Regression coefficient relating the square of shrub height with shrub area.
    - Absh, Bbsh: Allometric coefficients relating phytovolume with dry weight of shrub individuals.
    - Acr, B1cr, B2cr, B3cr, C1cr, C2cr: Regression coefficients used to calculate crown ratio of trees.
    - Acw, Bcw: Regression coefficients used to calculated crown width of trees.
- internalAllocation: A data frame with internal allocation variables for each cohort:
    - allocationTarget: Value of the allocation target variable.
    - leafAreaTarget: Target leaf area (m2) per individual.
    - fineRootBiomassTarget: Target fine root biomass (g dry) per individual (only if transpirationMode = "Sperry").
- internalCarbon and internalRings: data structures to store other internal state variables.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

resetInputs, spwb, soil, forest, SpParamsMED, defaultSoilParams, plant_ID

Examples

```
#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

# Aboveground parameters
above = forest2aboveground(exampleforestMED, SpParamsMED)
above

# Initialize soil with default soil params
examplesoil = soil(defaultSoilParams())

# Rooting depths
Z50 = c(exampleforestMED$treeData$Z50, exampleforestMED$shrubData$Z50)
Z95 = c(exampleforestMED$treeData$Z95, exampleforestMED$shrubData$Z95)

# Initialize control parameters
control = defaultControl("Granier")

# Prepare spwb input
spwbInput(above, Z50, Z95, examplesoil,SpParamsMED, control)

# When starting from an object of class 'forest' the whole process
```

```
# can be simplified:
forest2spwbInput(exampleforestMED, examplesoil, SpParamsMED, control)


# Prepare input for Sperry transpiration mode
control = defaultControl("Sperry")
forest2spwbInput(exampleforestMED,examplesoil,SpParamsMED, control)
```

---

modifyParams *Modify parameters*

Description

Routines to modify species parameter table or model input objects

Usage

```
modifySpParams(SpParams, customParams, subsetSpecies = TRUE)
modifyCohortParams(x, customParams, verbose = TRUE)
modifyInputParams(x, customParams, verbose = TRUE)
```

Arguments

x	A model input object of class `spwbInput` or `growthInput`.
SpParams	A species parameter data frame, typically `SpParamsMED`.
customParams	A data frame or a named vector with new parameter values (see details).
subsetSpecies	A flag to indicate that the output data frame should include only those species mentioned in `customParams`.
verbose	A flag to indicate that messages should be printed on the console.

Details

When calling function `modifySpParams`, `customParams` should be a data frame with as many rows as species and as many columns as parameters to modify, plus a column called 'SpIndex' to match species between the two tables.

When calling `modifyCohortParams`, `customParams` can be a data frame with as many rows as cohorts and as many columns as parameters to modify, plus a column called 'Cohort' which will be matched with the cohort names given by `spwbInput` or `growthInput`. Alternatively, `customParams` can be a named list or named numeric vector as for `modifyInputParams`.

When calling `modifyInputParams`, `customParams` must be either a named list or a named numeric vector. Cohort parameters are specified using the syntax "<cohortName>/<paramName>" for names (e.g. "T2_176/Z50" to modify parameter 'Z50' of cohort 'T2_176'). Soil layer parameters are specified using the syntax "<paramName>@#layer" for names, where #layer is the layer index (e.g. "rfc@1" will modify the rock fragment content of soil layer 1). Control parameters are specified using either "<paramName>" (e.g "phloemConductanceFactor") or "<paramName>$<subParamName>" (e.g "maximumRelativeGrowthRates$leaf"). It may seem unnecessary to modify soil or control parameters via a function, but modifyInputParams is called from optimization functions (see optimization).

Value

Function modifySpParams returns a modified species parameter data frame.

Functions modifyCohortParams and modifyInputParams return a modified spwbInput or growthInput object. Note that modifications may affect other parameters beyond those indicated in customParams, as a result of parameter dependencies (see examples).

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwbInput, SpParamsMED, optimization

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params (4 layers)
examplesoil1 = soil(defaultSoilParams(4))

#Initialize control parameters
control = defaultControl("Granier")

#Initialize input
x1 = forest2spwbInput(exampleforestMED,examplesoil1, SpParamsMED, control)

# Cohort name for Pinus halepensis
PH_coh = paste0("T1_", SpParamsMED$SpIndex[SpParamsMED$Name=="Pinus halepensis"])
PH_coh

# Modify Z50 and Z95 of Pinus halepensis cohort
customParams <- c(200,2000)
names(customParams) <- paste0(PH_coh,c("/Z50", "/Z95"))
x1m <- modifyInputParams(x1, customParams)

# Inspect original and modified objects
x1$below
x1m$below

# Inspect dependencies: fine root distribution across soil layers
```

```
x1$belowLayers$V
x1m$belowLayers$V

# Modify rock fragment content and sand proportion of soil layer 1
x1s <- modifyInputParams(x1, c("rfc@1" = 5, "sand@1" = 10))

# Inspect original and modified soils
x1$soil
x1s$soil

# When modifying growth input objects dependencies increase
x1 = forest2growthInput(exampleforestMED,examplesoil1, SpParamsMED, control)
customParams <- c(2000,2)
names(customParams) <- paste0(PH_coh,c("/Al2As", "/LAI_live"))
x1m <- modifyInputParams(x1, customParams)
```

Mortality *Mortality*

Description

A simple function to determine a daily mortality likelihood according to the value of a stress variable.

Usage

```
mortality_dailyProbability(mortalityBaselineRate,
                           stressValue, stressThreshold,
                           allowStress = TRUE,
                           minValue = 0.0, slope = 1.0)
```

Arguments

mortalityBaselineRate	Baseline mortality rate at the annual scale.
stressValue	Current value of the stress variable (lower values indicate stronger stress).
stressThreshold	Threshold to indicate that lower values increase mortality likelihood.
allowStress	A boolean function to activate stress effects.
minValue	Minimum value of the stress variable (i.e. maximum stress), corresponding to probability of mortality equal to 1.
slope	Slope coefficient modulating how fast probability increases after the stress threshold has been reached.

Value

Returns a probability (between 0 and 1).

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

growth

---

optimization *Multiple model runs and function factories for optimization routines*

Description

Function factories to generate functions to be used in model calibration, uncertainty or sensitivity analysis.

Usage

```
multiple_runs(parMatrix, x,
              meteo, latitude,
              elevation = NA, slope = NA, aspect = NA,
              summary_function = NULL, args = NULL,
              verbose = TRUE)
optimization_function(parNames, x,
                      meteo, latitude,
                      elevation = NA, slope = NA, aspect = NA,
                      summary_function, args= NULL)
optimization_evaluation_function(parNames, x,
                      meteo, latitude,
                      elevation = NA, slope = NA, aspect = NA,
                      measuredData, type = "SWC", cohorts = NULL,
                      temporalResolution = "day", SpParams = NULL,
                      metric = "loglikelihood")
optimization_multicohort_function(cohortParNames, cohortNames, x,
                      meteo, latitude,
                      otherParNames = NULL,
                      elevation = NA, slope = NA, aspect = NA,
                      summary_function, args= NULL)
optimization_evaluation_multicohort_function(cohortParNames, cohortNames, x,
                      meteo, latitude,
                      otherParNames = NULL,
                      elevation = NA, slope = NA, aspect = NA,
                      measuredData, type = "SWC", cohorts = cohortNames,
                      temporalResolution = "day", SpParams = NULL,
                      metric = "loglikelihood")
```

Arguments

parMatrix
: A matrix of parameter values with runs in rows and parameters in columns. Column names should follow parameter modification naming rules (see examples and naming rules in `modifyInputParams`).

parNames
: A string vector of parameter names (see examples and naming rules in `modifyInputParams`).

x
: An object of class `spwbInput` or `growthInput`.

meteo, latitude, elevation, slope, aspect
: Additional parameters to simulation functions `spwb` or `growth`.

verbose
: A flag to indicate extra console output.

measuredData
: A data frame with observed/measured values. Dates should be in row names, whereas columns should be named according to the type of output to be evaluated (see details).

type
: A string with the kind of model output to be evaluated. Accepted values are `"SWC"` (soil moisture content), `"REW"` relative extractable water, `"ETR"` (total evapotranspiration), `"E"` (transpiration per leaf area), `"FMC"` (fuel moisture content) and `"WP"` (plant water potentials).

cohorts
: A string or a vector of strings with the cohorts to be compared (e.g. "T1_68"). If several cohort names are provided, the function `optimization_cohorts_function` evaluates the performance for each one and provides the mean value. If `NULL` results for the first cohort will be evaluated.

temporalResolution
: A string to indicate the temporal resolution of the model evaluation, which can be "day", "week", "month" or "year". Observed and modelled values are aggregated temporally (using either means or sums) before comparison.

SpParams
: A data frame with species parameters (see `SpParamsMED`), only needed if `type = "FMC"`.

metric
: An evaluation metric (see `evaluation_metric`).

summary_function
: A function whose input is the result of `spwb` or `growth`. The function must return a numeric scalar in the case of `optimization_function`, but is not restricted in the case of `multiple_runs`.

args
: A list of additional arguments of `optimization_function`.

cohortParNames
: A string vector of vegetation parameter names for cohorts (e.g. 'Z95' or 'psiExtract').

cohortNames
: A string vector of cohort names. All cohorts will be given the same parameter values for each parameter in 'cohortParNames'.

otherParNames
: A string vector of parameter names (see examples and naming rules in `modifyInputParams`) for non-vegetation parameters (i.e. control parameters and soil parameters).

Details

See `evaluation` for details regarding how to specify measured data.

Functions produced by these function factories should be useful for sensitivity analyses using package 'sensitivity'.

Parameter naming (i.e. parNames) should follow the rules specified in section details of modifyInputParams. The exception to the naming rules applies when multiple cohorts are to be modified to the same values with functions optimization_multicohort_function and optimization_evaluation_multicohort_function. Then, only a vector of parameter names is supplied for cohortParNames.

Value

Function multiple_runs returns a list, whose elements are either the result of calling simulation models or the result of calling summary_function afterwards.

Function optimization_function returns a function whose parameters are parameter values and whose return is a prediction scalar (e.g. total transpiration).

Function optimization_evaluation_function returns a function whose parameters are parameter values and whose return is an evaluation metric (e.g. loglikelihood of the data observations given model predictions). If evaluation data contains information for different cohorts (e.g. plant water potentials or transpiration rates) then the evaluation is performed for each cohort and the metrics are averaged.

Function optimization_multicohorts_function returns a function whose parameters are parameter values and whose return is a prediction scalar (e.g. total transpiration). The difference with optimization_function is that multiple cohorts are set to the same parameter values.

Function optimization_evaluation_multicohort_function returns a function whose parameters are parameter values and whose return is an evaluation metric (e.g. loglikelihood of the data observations given model predictions). If evaluation data contains information for different cohorts (e.g. plant water potentials or transpiration rates) then the evaluation is performed for each cohort and the metrics are averaged. The difference with optimization_evaluation_function is that multiple cohorts are set to the same parameter values.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

evaluation_metric, modifyInputParams, spwb, growth

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params (4 layers)
examplesoil1 = soil(defaultSoilParams(4))

#Initialize control parameters
```

```
control = defaultControl("Granier")

#Initialize input
x1 = forest2spwbInput(exampleforestMED,examplesoil1, SpParamsMED, control)

# Cohort name for Pinus halepensis
PH_coh = paste0("T1_", SpParamsMED$SpIndex[SpParamsMED$Name=="Pinus halepensis"])
PH_coh

#Parameter names of interest
parNames = c(paste0(PH_coh,"/Z50"), paste0(PH_coh,"/Z95"))

#Specify parameter matrix
parMatrix <- cbind(c(200,300), c(500,1000))
colnames(parMatrix) <- parNames

#Define a summary function as the total transpiration over the simulated period
sf<-function(x) {sum(x$WaterBalance$Transpiration, na.rm=TRUE)}

#Perform two runs and evaluate the summary function
multiple_runs(parMatrix,
              x1, examplemeteo, latitude = 42, elevation = 100,
              summary_function = sf)

#Load observed data (in this case the same simulation results with some added error)
# Generate a prediction function for total transpiration over the simulated period
# as a function of parameters "Z50" and "Z95" for Pinus halepensis cohort
of<-optimization_function(parNames = parNames,
                          x = x1,
                          meteo = examplemeteo,
                          latitude = 41.82592, elevation = 100,
                          summary_function = sf)

# Evaluate for the values of the parameter matrix
of(parMatrix[1, ])
of(parMatrix)


# Generate a loglikelihood function for soil water content
# as a function of parameters "Z50" and "Z95" for Pinus halepensis cohort
data(exampleobs)
oef<-optimization_evaluation_function(parNames = parNames,
                x = x1,
                meteo = examplemeteo, latitude = 41.82592, elevation = 100,
                measuredData = exampleobs, type = "SWC",
                metric = "loglikelihood")

# Loglikelihood for the values of the parameter matrix
oef(parMatrix[1, ])
oef(parMatrix)
```

`Parameter means` *Parameter average values*

Description

Internal data set with parameter averages for taxonomic families. This is used by input initialization functions to provide suitable parameter values when missing from species parameter tables.

Format

Data frame `trait_family_means` has taxonomic families in rows and parameter names as columns.

Source

Same sources as `SpParamsMED`

See Also

`SpParamsMED`, `spwbInput`

Examples

```
medfate:::trait_family_means
```

`pheno_updateLeaves` *Leaf phenology*

Description

Function `pheno_leafDevelopmentStatus` returns the expanded status (0 to 1) of leaves according to the growth degree days required to start bud burst and leaf unfolding, as dictated by a simple ecodormancy (one-phase) model (Chuine et al. 2013). Function `pheno_leafSenescenceStatus` returns the 0/1 senescence status of leaves according to the one-phase senescence model of Delpierre et al. (2009) on the basis of photoperiod and temperature. Function `pheno_updateLeaves` updates the status of expanded leaves and dead leaves of object `x` given the photoperiod, temperature and wind of a given day. It applies the development model for 1 < doy < 180 and the senescence model for 181 > doy > 365.

Usage

```
pheno_leafDevelopmentStatus(Sgdd, gdd, unfoldingDD = 300)
pheno_leafSenescenceStatus(Ssen, sen)
pheno_updatePhenology(x, doy, photoperiod, tmean)
pheno_updateLeaves(x, wind, fromGrowthModel)
```

Arguments

Sgdd	Degree days required for leaf budburst (in Celsius).
gdd	Cumulative degree days (in Celsius)
unfoldingDD	Degree-days for complete leaf unfolding after budburst has occurred.
Ssen	Threshold to start leaf senescence.
sen	Cumulative senescence variable.
x	An object of class spwbInput.
doy	Day of the year.
photoperiod	Day length (in hours).
tmean	Average day temperature (in Celsius).
wind	Average day wind speed (in m/s).
fromGrowthModel	Boolean flag to indicate that routine is called from growth simulation function.

Value

Function pheno_leafDevelopmentStatus returns a vector of values between 0 and 1, whereas function pheno_leafSenescenceStatus returns a vector of 0 (senescent) and 1 (expanded) values. The other two functions do not return any value (see details).

Note

Functions pheno_updatePhenology and pheno_updateLeaves modify the input object x. The first modifies phenological state and the second modifies the leaf area accordingly.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Chuine, I., De Cortazar-Atauri, I.G., Kramer, K., Hänninen, H., 2013. Plant development models. Phenology: An Integrative Environmental Science. Springer, pp. 275–293.

Delpierre N, Dufrêne E, Soudani K et al (2009) Modelling interannual and spatial variability of leaf senescence for three deciduous tree species in France. Agric For Meteorol 149:938–948. doi:10.1016/j.agrformet.2008.11.014

See Also

spwb, spwbInput

photo *Photosynthesis submodel functions*

Description

Set of functions used in the calculation of photosynthesis.

Usage

```
photo_GammaTemp(Tleaf)
photo_KmTemp(Tleaf, Oi = 209)
photo_VmaxTemp(Vmax298, Tleaf)
photo_JmaxTemp(Jmax298, Tleaf)
photo_electronLimitedPhotosynthesis(Q, Ci, GT, Jmax)
photo_rubiscoLimitedPhotosynthesis(Ci, GT, Km, Vmax)
photo_photosynthesis(Q, Catm, Gc, Tleaf, Vmax298, Jmax298, verbose = FALSE)
photo_leafPhotosynthesisFunction(E, psiLeaf, Catm, Patm, Tair, vpa, u,
                             absRad, Q, Vmax298, Jmax298,
                             leafWidth = 1.0, refLeafArea = 1, verbose = FALSE)
photo_leafPhotosynthesisFunction2(E, psiLeaf, Catm, Patm, Tair, vpa, u,
                             SWRabs, LWRnet, Q, Vmax298, Jmax298,
                             leafWidth = 1.0, refLeafArea = 1, verbose = FALSE)
photo_sunshadePhotosynthesisFunction(E, psiLeaf, Catm, Patm, Tair, vpa,
                                     SLarea, SHarea, u,
                                     absRadSL, absRadSH, QSL, QSH,
                                    Vmax298SL, Vmax298SH, Jmax298SL, Jmax298SH,
                                     leafWidth = 1.0, verbose = FALSE)
photo_multilayerPhotosynthesisFunction(E, psiLeaf, Catm, Patm, Tair, vpa,
                                      SLarea, SHarea, u,
                                      absRadSL, absRadSH, QSL, QSH,
                                      Vmax298, Jmax298, leafWidth = 1.0,
                                      verbose = FALSE)
```

Arguments

`Tleaf` Leaf temperature (in ºC).

`Oi` Oxigen concentration (mmol*mol-1).

`Vmax298, Vmax298SL, Vmax298SH`
: Maximum Rubisco carboxylation rate per leaf area at 298ºK (i.e. 25 ºC) (micromol*s-1*m-2) (for each canopy layer in the case of `photo_multilayerPhotosynthesisFunction`). 'SH' stands for shade leaves, whereas 'SL' stands for sunlit leaves.

`Jmax298, Jmax298SL, Jmax298SH`
: Maximum electron transport rate per leaf area at 298ºK (i.e. 25 ºC) (micromol*s-1*m-2) (for each canopy layer in the case of `photo_multilayerPhotosynthesisFunction`). 'SH' stands for shade leaves, whereas 'SL' stands for sunlit leaves.

`Q` Active photon flux density (micromol * s-1 * m-2).

Ci	CO2 internal concentration (micromol * mol-1).
GT	CO2 saturation point corrected by temperature (micromol * mol-1).
Jmax	Maximum electron transport rate per leaf area (micromol*s-1*m-2).
Km	Km = Kc*(1.0+(Oi/Ko)) - Michaelis-Menten term corrected by temperature (in micromol * mol-1).
Vmax	Maximum Rubisco carboxylation rate per leaf area (micromol*s-1*m-2).
Catm	CO2 air concentration (micromol * mol-1).
Gc	CO2 leaf (stomatal) conductance (mol * s-1 * m-2).
E	Transpiration flow rate per leaf area (mmol*s-1*m-2).
psiLeaf	Leaf water potential (MPa).
Patm	Atmospheric air pressure (in kPa).
Tair	Air temperature (in ºC).
vpa	Vapour pressure deficit (in kPa).
u	Wind speed above the leaf boundary (in m/s) (for each canopy layer in the case of `photo_multilayerPhotosynthesisFunction`).
absRad	Absorbed long- and short-wave radiation (in W*m^-2).
SWRabs	Absorbed short-wave radiation (in W·m-2).
LWRnet	Net long-wave radiation balance (in W·m-2).
leafWidth	Leaf width (in cm).
refLeafArea	Leaf reference area.
verbose	Boolean flag to indicate console output.
SLarea, SHarea	Leaf area index of sunlit/shade leaves (for each canopy layer in the case of `photo_multilayerPhotosynthesisFunction`).
absRadSL, absRadSH	Instantaneous absorbed radiation (W·m-2) per unit of sunlit/shade leaf area (for each canopy layer in the case of `photo_multilayerPhotosynthesisFunction`).
QSL, QSH	Active photon flux density (micromol * s-1 * m-2) per unit of sunlit/shade leaf area (for each canopy layer in the case of `photo_multilayerPhotosynthesisFunction`).

Details

Details of the photosynthesis submodel are given in a vignette.

Value

Values returned for each function are:

- `photo_GammaTemp`: CO2 compensation concentration (micromol * mol-1).
- `photo_KmTemp`: Michaelis-Menten coefficients of Rubisco for Carbon (micromol * mol-1) and Oxigen (mmol * mol-1).
- `photo_VmaxTemp`: Temperature correction of Vmax298.
- `photo_JmaxTemp`: Temperature correction of Jmax298.

- `photo_electronLimitedPhotosynthesis`: Electron-limited photosynthesis (micromol*s-1*m-2) following Farquhar et al. (1980).
- `photo_rubiscoLimitedPhotosynthesis`: Rubisco-limited photosynthesis (micromol*s-1*m-2) following Farquhar et al. (1980).
- `photo_photosynthesis`: Calculates gross photosynthesis (micromol*s-1*m-2) following (Farquhar et al. (1980) and Collatz et al (1991).
- `photo_leafPhotosynthesisFunction`: Returns a data frame with the following columns:
  - `LeafTemperature`: Leaf temperature (ºC).
  - `LeafVPD`: Leaf vapor pressure deficit (kPa).
  - `LeafCi`: Internal CO2 concentration (micromol * mol-1).
  - `Gsw`: Leaf stomatal conductance to water vapor (mol * s-1 * m-2).
  - `GrossPhotosynthesis`: Gross photosynthesis (micromol*s-1*m-2).
  - `NetPhotosynthesis`: Net photosynthesis, after discounting autotrophic respiration (micromol*s-1*m-2).
- `photo_sunshadePhotosynthesisFunction`: Returns a data frame with the following columns:
  - `GrossPhotosynthesis`: Gross photosynthesis (micromol*s-1*m-2).
  - `NetPhotosynthesis`: Net photosynthesis, after discounting autotrophic respiration (micromol*s-1*m-2).
  - `LeafCiSL`: Sunlit leaf internal CO2 concentration (micromol * mol-1).
  - `LeafCiSH`: Shade leaf internal CO2 concentration (micromol * mol-1).
  - `LeafTempSL`: Sunlit leaf temperature (ºC).
  - `LeafTempSH`: Shade leaf temperature (ºC).
  - `LeafVPDSL`: Sunlit leaf vapor pressure deficit (kPa).
  - `LeafVPDSH`: Shade leaf vapor pressure deficit (kPa).
- `photo_multilayerPhotosynthesisFunction`: Return a data frame with the following columns:
  - `GrossPhotosynthesis`: Gross photosynthesis (micromol*s-1*m-2).
  - `NetPhotosynthesis`: Net photosynthesis, after discounting autotrophic respiration (micromol*s-1*m-2).

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Bernacchi, C. J., E. L. Singsaas, C. Pimentel, A. R. Portis, and S. P. Long. 2001. Improved temperature response functions for models of Rubisco-limited photosynthesis. Plant, Cell and Environment 24:253–259.

Collatz, G. J., J. T. Ball, C. Grivet, and J. A. Berry. 1991. Physiological and environmental regulation of stomatal conductance, photosynthesis and transpiration: a model that includes a laminar boundary layer. Agricultural and Forest Meteorology 54:107–136.

Farquhar, G. D., S. von Caemmerer, and J. A. Berry. 1980. A biochemical model of photosynthetic CO2 assimilation in leaves of C3 species. Planta 149:78–90.

Leuning, R. 2002. Temperature dependence of two parameters in a photosynthesis model. Plant, Cell and Environment 25:1205–1210.

Sperry, J. S., M. D. Venturas, W. R. L. Anderegg, M. Mencuccini, D. S. Mackay, Y. Wang, and D. M. Love. 2016. Predicting stomatal responses to the environment from the optimization of photosynthetic gain and hydraulic cost. Plant Cell and Environment.

See Also

hydraulics_supplyFunctionNetwork, biophysics_leafTemperature, spwb

---

Plant values *Plant description functions*

Description

Functions to calculate attributes of plants in a forest object.

Usage

```
plant_basalArea(x)
plant_largerTreeBasalArea(x)
plant_characterParameter(x, SpParams, parName)
plant_cover(x)
plant_crownBaseHeight(x, SpParams, mode = "MED")
plant_crownLength(x, SpParams, mode = "MED")
plant_crownRatio(x, SpParams, mode = "MED")
plant_density(x, SpParams, mode = "MED")
plant_equilibriumLeafLitter(x, SpParams, AET = 800, mode = "MED")
plant_equilibriumSmallBranchLitter(x, SpParams,
                              smallBranchDecompositionRate = 0.81, mode = "MED")
plant_foliarBiomass(x, SpParams, gdd = NA, mode = "MED")
plant_fuel(x, SpParams, gdd = NA, includeDead = TRUE, mode = "MED")
plant_height(x)
plant_ID(x, treeOffset = 0, shrubOffset = 0)
plant_LAI(x, SpParams, gdd = NA, mode = "MED")
plant_parameter(x, SpParams, parName, fillMissing = TRUE)
plant_phytovolume(x, SpParams)
plant_species(x)
plant_speciesName(x, SpParams)
```

Arguments

x	An object of class forest.
SpParams	A data frame with species parameters (see SpParamsMED).
parName	A string with a parameter name.
mode	Calculation mode, either "MED" or "US".

`gdd`	Growth degree days (to account for leaf phenology effects).
`AET`	Actual annual evapotranspiration (in mm).
`smallBranchDecompositionRate`	Decomposition rate of small branches.
`includeDead`	A flag to indicate that standing dead fuels (dead branches) are included.
`treeOffset, shrubOffset`	Integers to offset cohort IDs.
`fillMissing`	A boolean flag to try imputation on missing values.

Value

A vector with values for each plant of the input `forest` object:

- `plant_basalArea`: Tree basal area (m2/ha).
- `plant_largerTreeBasalArea`: Basal area (m2/ha) of trees larger (in diameter) than the tree. Half of the trees of the same record are included.
- `plant_characterParameter`: The parameter values of each plant, as strings.
- `plant_cover`: Shrub cover (in percent).
- `plant_crownBaseHeight`: The height corresponding to the start of the crown (in cm).
- `plant_crownLength`: The difference between crown base height and total height (in cm).
- `plant_crownRatio`: The ratio between crown length and total height (between 0 and 1).
- `plant_density`: Plant density (ind/ha). Tree density is directly taken from the forest object, while the shrub density is estimated from cover and height by calculating the area of a single individual.
- `plant_equilibriumLeafLitter`: Litter biomass of leaves at equilibrium (in kg/m2).
- `plant_equilibriumSmallBranchLitter`: Litter biomass of small branches (< 6.35 mm diameter) at equilibrium (in kg/m2).
- `plant_foliarBiomass`: Standing biomass of leaves (in kg/m2).
- `plant_fuel`: Fine fuel load (in kg/m2).
- `plant_height`: Total height (in cm).
- `plant_ID`: Cohort coding for simulation functions (concatenation of 'T' (Trees) or 'S' (Shrub), cohort index and species index).
- `plant_LAI`: Leaf area index (m2/m2).
- `plant_parameter`: The parameter values of each plant, as numeric.
- `plant_phytovolume`: Shrub phytovolume (m3/m2).
- `plant_species`: Species identity integer (indices start with 0).
- `plant_speciesName`: String with species taxonomic name (or a functional group).

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwb, forest, summary.forest

Examples

```
#Default species parameterization
data(SpParamsMED)

#Load example plot
data(exampleforestMED)

#A short way to obtain total basal area
sum(plant_basalArea(exampleforestMED), na.rm=TRUE)

#The same forest level function for LAI
sum(plant_LAI(exampleforestMED, SpParamsMED))

#The same forest level function for fuel loading
sum(plant_fuel(exampleforestMED, SpParamsMED))

#Summary function for 'forest' objects can be also used
summary(exampleforestMED, SpParamsMED)

plant_speciesName(exampleforestMED, SpParamsMED)

plant_ID(exampleforestMED)
```

---

plot.spwb *Plots simulation results*

Description

Function plot produces temporal graphs of the results of the soil plant water balance model (see spwb), plant water balance model (see pwb), the forest growth model (see growth) or the forest dynamics model (see fordyn).

Usage

```
## S3 method for class 'spwb'
plot(x, type="PET_Precipitation",
                    cohorts = NULL, bySpecies = FALSE,
                    dates = NULL, subdaily = FALSE,
                    xlim = NULL, ylim=NULL, xlab=NULL, ylab=NULL,
                    summary.freq = NULL, ...)
## S3 method for class 'pwb'
plot(x, type="PlantTranspiration",
                   cohorts = NULL, bySpecies = FALSE,
                   dates = NULL, subdaily = FALSE,
```

```
                      xlim = NULL, ylim=NULL, xlab=NULL, ylab=NULL,
                      summary.freq = NULL, ...)
## S3 method for class 'growth'
plot(x, type="PET_Precipitation",
                         cohorts = NULL, bySpecies = FALSE,
                         dates = NULL, subdaily = FALSE,
                         xlim = NULL, ylim=NULL, xlab=NULL, ylab=NULL,
                         summary.freq = NULL, ...)
## S3 method for class 'fordyn'
plot(x, type="StandBasalArea",
                         ylim=NULL, xlab=NULL, ylab=NULL, ...)
```

Arguments

`x`	An object of class `spwb`, `pwb`, `growth` or `fordyn`.
`type`	The information to be plotted (see details)
`cohorts`	An integer, boolean or character vector to select the plant cohorts to be plotted.
`bySpecies`	Allows aggregating output by species, before drawing plots (only has an effect with some values of `type`). Aggregation can involve a sum (as for plant lai or transpiration) or a LAI-weighted mean (as for plant stress or plant water potential).
`dates`	A Date vector with a subset of dates to be plotted.
`subdaily`	Whether subdaily results should be shown, only for simulations using `transpirationMode = "Sperry"` and having set `subdailyResults = TRUE` in the simulation control object. If `subdaily = TRUE`, then the valid strings for `type` are listed in `plot.spwb_day`.
`xlim`	Range of values for x.
`ylim`	Range of values for y.
`xlab`	x-axis label.
`ylab`	y-axis label.
`summary.freq`	Frequency of summary statistics (see `cut.Date`).
`...`	Additional parameters for function `plot` (not used).

Details

The following plots are currently available for `spwb` (most of them also for `pwb`):

- `"PET_Precipitation"`: Potential evapotranspiration and Precipitation.
- `"PET_NetRain"`: Potential evapotranspiration and Net rainfall.
- `"Snow"`: Snow precipitation and snowpack dynamics.
- `"Export"`: Water exported through deep drainage and surface runoff.
- `"Evapotranspiration"`: Plant transpiration and soil evaporation.
- `"SoilPsi"`: Soil water potential.
- `"SoilRWC"`: Soil relative water content (in percent of field capacity).

- "SoilTheta": Soil moisture water content (in percent volume).
- "SoilVol": Soil water volumetric content (in mm).
- "PlantExtraction": Water extracted by plants from each soil layer.
- "HydraulicRedistribution":Water added to each soil layer coming from other soil layers, transported through the plant hydraulic network (only for transpirationMode = "Sperry").
- "WTD": Water table depth.
- "LAI": Expanded and dead leaf area index of the whole stand.
- "PlantLAI": Plant cohort leaf area index (expanded leaves).
- "SoilPlantConductance":Average instantaneous overall soil plant conductance (calculated as the derivative of the supply function).
- "PlantStress": Plant cohort average daily drought stress.
- "PlantPsi": Plant cohort water potential (only for transpirationMode = "Granier").
- "LeafPsi": Midday leaf water potential (only for transpirationMode = "Sperry").
- "StemPsi": Midday (upper) stem water potential (only for transpirationMode = "Sperry").
- "RootPsi": Midday root crown water potential (only for transpirationMode = "Sperry").
- "PlantTranspiration": Plant cohort transpiration.
- "TranspirationPerLeaf": Plant cohort transpiration per leaf area.
- "PlantGrossPhotosynthesis": Plant cohort photosynthesis.
- "GrossPhotosynthesisPerLeaf": Plant cohort photosynthesis per leaf area.
- "PlantNetPhotosynthesis": Plant cohort net photosynthesis (only for transpirationMode = "Sperry").
- "NetPhotosynthesisPerLeaf": Plant cohort net photosynthesis per leaf area (only for transpirationMode = "Sperry").
- "PlantWUE": Plant cohort daily water use efficiency (gross photosynthesis over transpiration; only for transpirationMode = "Sperry").
- "PlantAbsorbedSWR": Plant cohort absorbed short wave radiation (only for transpirationMode = "Sperry").
- "AbsorbedSWRPerLeaf": Plant cohort absorbed short wave radiation per leaf area (only for transpirationMode = "Sperry").
- "PlantNetLWR": Plant cohort net long wave radiation (only for transpirationMode = "Sperry").
- "NetLWRPerLeaf": Plant cohort net long wave radiation per leaf area (only for transpirationMode = "Sperry").
- "AirTemperature": Minimum/maximum/mean daily temperatures above canopy (only for transpirationMode = "Sperry").
- "CanopyTemperature": Minimum/maximum/mean daily temperatures inside canopy (only for transpirationMode = "Sperry").
- "SoilTemperature": Minimum/maximum/mean daily temperatures inside the first soil layer (only for transpirationMode = "Sperry").
- "CanopyEnergyBalance": Canopy energy balance components (only for transpirationMode = "Sperry").

- `"SoilEnergyBalance"`: Soil energy balance components (only for `transpirationMode = "Sperry"`).

The following are only available for `growth`:

- `"GrossPhotosynthesis"`: Gross photosynthesis rate per dry weight.
- `"MaintenanceRespiration"`: Maintenance respiration cost per dry weight.
- `"CarbonBalance"`: Carbon balance per dry weight.
- `"SugarLeaf"`: Sugar concentration in leaves.
- `"StarchLeaf"`: Starch concentration in leaves.
- `"SugarSapwood"`: Sugar concentration in sapwood.
- `"StarchSapwood"`: Starch concentration in sapwood.
- `"SugarTransport"`: Phloem sugar transport rate.
- `"RootExudation"`: Root exudation rate per dry weight.
- `"LeafPI0"`: Leaf osmotic concentration at full turgor.
- `"StemPI0"`: Sapwood osmotic concentration at full turgor.
- `"SapwoodArea"`: Sapwood area per individual.
- `"LeafArea"`: Leaf area per individual.
- `"FineRootArea"`: Fine root area per individual.
- `"SapwoodBiomass"`: Sapwood dry biomass per individual.
- `"LeafBiomass"`: Leaf dry biomass per individual.
- `"FineRootBiomass"`: Fine root dry biomass per individual.
- `"LabileBiomass"`: Labile C biomass per individual.
- `"TotalLivingBiomass"`: Total dry biomass per individual.
- `"SAgrowth"`: Sapwood area growth rate.
- `"LAgrowth"`: Leaf area growth rate.
- `"FRAgrowth"`: Fine root area growth rate.
- `"HuberValue"`: Ratio of leaf area to sapwood area.
- `"RootAreaLeafArea"`: Ratio of fine root area to leaf area.

The following are only available for `fordyn`:

- `"StandBasalArea"`:Stand basal area of living trees.
- `"StandLAI"`:Stand leaf area index.
- `"StandDensity"`:Stand density of living trees.
- `"SpeciesBasalArea"`:Basal area of living trees by species.
- `"SpeciesLAI"`:Leaf area index by species.
- `"SpeciesDensity"`:Density of living trees by species.
- `"CohortBasalArea"`:Basal area of living trees by plant cohort.
- `"CohortLAI"`:Leaf area index by plant cohort.
- `"CohortDensity"`:Density of living trees by plant cohort.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwb

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params (2 layers)
examplesoil = soil(defaultSoilParams(2))

#Initialize control parameters
control = defaultControl("Granier")

#Initialize input
x = forest2spwbInput(exampleforestMED,examplesoil, SpParamsMED, control)

#Call simulation function
S1<-spwb(x, examplemeteo, latitude = 41.82592, elevation = 100)

#Plot results
plot(S1)
```

---

plot.spwb_day *Plots simulation results for one day*

Description

Functions to plot the subdaily simulation results of spwb_day, growth_day or the transpiration calculations of transp_transpirationSperry.

Usage

```
## S3 method for class 'growth_day'
plot(x, type="PlantTranspiration", bySpecies = FALSE,
                          xlim = NULL, ylim=NULL, xlab = NULL, ylab = NULL, ...)
## S3 method for class 'spwb_day'
plot(x, type="PlantTranspiration", bySpecies = FALSE,
```

```
                        xlim = NULL, ylim=NULL, xlab = NULL, ylab = NULL, ...)
## S3 method for class 'pwb_day'
plot(x, type="PlantTranspiration", bySpecies = FALSE,
                       xlim = NULL, ylim=NULL, xlab = NULL, ylab = NULL, ...)
```

Arguments

x	An object of class spwb_day, growth_day or pwb_day.
type	The information to be plotted (see details).
bySpecies	Allows aggregating output by species, before drawing plots. Aggregation can involve a sum (as for plant lai or transpiration) or a LAI-weighted mean (as for plant stress or plant water potential).
xlim	Range of values for x.
ylim	Range of values for y.
xlab	x-axis label.
ylab	y-axis label.
...	Additional parameters for function plot.

Details

The following plots are currently available for spwb_day and pwb_day:

- "LeafPsi":Leaf water potential (for shade and sunlit leaves).
- "LeafPsiAverage":Average leaf water potential.
- "RootPsi":Root crown water potential.
- "StemPsi":(Upper) stem water potential.
- "StemPLC":(Average) percentage of loss conductance in the stem conduits.
- "StemRWC":(Average) relative water content in the stem.
- "LeafRWC":Relative water content in the leaf.
- "StemSympRWC":(Average) relative water content in the stem symplasm.
- "LeafSympRWC":Relative water content in the leaf symplasm.
- "SoilPlantConductance":Overall soil plant conductance (calculated as the derivative of the supply function).
- "PlantExtraction": Water extracted from each soil layer.
- "PlantTranspiration": Plant cohort transpiration per ground area.
- "TranspirationPerLeaf": Plant cohort transpiration per leaf area.
- "PlantGrossPhotosynthesis": Plant cohort gross photosynthesis per ground area.
- "GrossPhotosynthesisPerLeaf": Plant cohort gross photosynthesis per leaf area.
- "PlantNetPhotosynthesis": Plant cohort net photosynthesis per ground area.
- "NetPhotosynthesisPerLeaf": Plant cohort net photosynthesis per leaf area.
- "PlantAbsorbedSWR": Absorbed short wave radiation per ground area (differentiates sunlit and shade leaves).

- "LeafTranspiration": Instantaneous transpiration per leaf area (differentiates sunlit and shade leaves).
- "LeafGrossPhotosynthesis": Instantaneous gross photosynthesis per leaf area (differentiates sunlit and shade leaves).
- "LeafNetPhotosynthesis": Instantaneous net photosynthesis per leaf area (differentiates sunlit and shade leaves).
- "LeafAbsorbedSWR": Absorbed short wave radiation per leaf area (differentiates sunlit and shade leaves).
- "LeafNetLWR": Net long wave radiation per leaf area (differentiates sunlit and shade leaves).
- "LeafCi": Leaf intercellular CO2 concentration (differentiates sunlit and shade leaves).
- "LeafIntrinsicWUE": Leaf intrinsic water use efficiency, i.e. the ratio between instantaneous photosynthesis and stomatal conductance (differentiates sunlit and shade leaves).
- "LeafVPD": Leaf vapour pressure deficit (differentiates sunlit and shade leaves).
- "LeafStomatalConductance": Leaf stomatal conductance to water vapour (differentiates sunlit and shade leaves).
- "LeafTemperature": Leaf temperature (differentiates sunlit and shade leaves).
- "Temperature": Above-canopy, inside-canopy and soil temperature.
- "CanopyEnergyBalance": Canopy energy balance components.
- "SoilEnergyBalance": Soil energy balance components.
- "PlantWaterBalance": Difference between water extraction from the soil and transpired water per ground area.
- "WaterBalancePerLeaf": Difference between water extraction from the soil and transpired water per leaf area.

Note

Only for soil plant water balance simulations using transpirationMode = "Sperry". This function can be used to display subdaily dynamics of corresponding to single days on spwb runs, if control option subdailyResults is set to TRUE. See also option subdaily in plot.spwb.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwb_day, plot.spwb

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)
```

```
#Default species parameterization
data(SpParamsMED)

#Initialize control parameters
control = defaultControl("Granier")
control$ndailysteps = 24

#Initialize soil with default soil params (2 layers)
examplesoil = soil(defaultSoilParams(2), W=c(0.5,0.5))


#Switch to 'Sperry' transpiration mode
control = defaultControl("Sperry")

#Simulate one day only
x2 = forest2spwbInput(exampleforestMED,examplesoil, SpParamsMED, control)
d = 100
sd2<-spwb_day(x2, rownames(examplemeteo)[d],
             examplemeteo$MinTemperature[d], examplemeteo$MaxTemperature[d],
             examplemeteo$MinRelativeHumidity[d], examplemeteo$MaxRelativeHumidity[d],
             examplemeteo$Radiation[d], examplemeteo$WindSpeed[d],
             latitude = 41.82592, elevation = 100,
             slope= 0, aspect = 0, prec = examplemeteo$Precipitation[d])

#Display transpiration for subdaily steps
plot(sd2, "PlantTranspiration")
```

`resetInputs` *Reset simulation inputs*

Description

Function `resetInputs()` allows resetting state variables in `x` to their defaults.

Usage

```
resetInputs(x)
```

Arguments

x	An object of class `spwbInput` or `growthInput`.

Value

Does not return any value. Instead, it modifies input object `x`.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwbInput, growthInput, spwb

---

root *Root functions*

Description

Functions to calculate properties of fine/coarse roots within the soil, given root system parameters and soil layer definition.

Usage

```
root_conicDistribution(Zcone, d)
root_ldrDistribution(Z50, Z95, d)
root_coarseRootLengths(v, d, depthWidthRatio = 1.0)
root_coarseRootLengthsFromVolume(VolInd, v, d, rfc)
root_coarseRootSoilVolume(v, d, depthWidthRatio = 1.0)
root_coarseRootSoilVolumeFromConductance(Kmax_rootxylem, VCroot_kmax, Al2As,
                                         v, d, rfc)
root_fineRootRadius(specificRootLength, rootTissueDensity)
root_fineRootHalfDistance(rootLengthDensity)
root_fineRootAreaIndex(Ksoil, krhizo, lai,
                       specificRootLength, rootTissueDensity,
                       rootLengthDensity)
root_fineRootBiomass(Ksoil, krhizo, lai, N,
                     specificRootLength, rootTissueDensity, rootLengthDensity)
root_rhizosphereMaximumConductance(Ksoil, fineRootBiomass, lai, N,
                                   specificRootLength, rootTissueDensity,
                                   rootLengthDensity)
root_fineRootSoilVolume(fineRootBiomass, specificRootLength, rootLengthDensity)
root_specificRootSurfaceArea(specificRootLength, rootTissueDensity)
root_individualRootedGroundArea(VolInd, V, d, rfc)
root_horizontalProportions(poolProportions, VolInd, N, V, d, rfc)
```

Arguments

Z50	A vector of depths (in mm) corresponding to 50% of roots.
Z95	A vector of depths (in mm) corresponding to 95% of roots.
Zcone	A vector of depths (in mm) corresponding to the root cone tip.
d	The width (in mm) corresponding to each soil layer.
v	Vector of proportions of fine roots in each soil layer.
depthWidthRatio	Ratio between radius of the soil layer with the largest radius and maximum rooting depth.

`rfc` Percentage of rock fragment content (volume basis) for each layer.

`Kmax_rootxylem` Sapwood-specific hydraulic conductivity of root xylem (in kg H2O·s-1·m-1·MPa-1).

`VCroot_kmax` Root xylem maximum conductance per leaf area (mmol·m-2·s-1·MPa-1).

`Al2As` Leaf area to sapwood area ratio (in m2·m-2).

`specificRootLength` Specific fine root length (length of fine roots over weight).

`rootTissueDensity` Fine root tissue density (weight over volume at turgidity).

`Ksoil` Soil saturated conductivity (mmol·m-1·s-1·MPa-1).

`krhizo` Rhizosphere maximum conductance per leaf area (mmol·m-2·s-1·MPa-1).

`lai` Leaf area index.

`rootLengthDensity` Fine root length density (length of fine roots over soil volume; cm/cm3)

`fineRootBiomass` Biomass of fine roots (g).

`V` Matrix of proportions of fine roots (cohorts x soil layers).

`VolInd` Volume of soil (in m3) occupied by coarse roots per individual.

`N` Density of individuals per hectare.

`poolProportions` Division of the stand area among plant cohorts (proportions).

Details

- `root_conicDistribution()` assumes a (vertical) conic distribution of fine roots, whereas `root_ldrDistribution()` distributes fine roots according to the linear dose response model of Schenck & Jackson (2002). Return a matrix of fine root proportions in each layer with as many rows as elements in `Z` (or `Z50`) and as many columns as soil layers.
- `root_coarseRootLengths()` and `root_coarseRootLengthsFromVolume()` estimate the length of coarse roots (mm) for each soil layer, including axial and radial lengths.
- `root_coarseRootSoilVolume` estimates the soil volume (m3) occupied by coarse roots of an individual.
- `root_coarseRootSoilVolumeFromConductance` estimates the soil volume (m3) occupied by coarse roots of an individual from root xylem conductance.
- `root_fineRootHalfDistance()` calculates the half distance (cm) between neighbouring fine roots.
- `root_fineRootRadius()` calculates the radius of fine roots (cm).
- `root_fineRootAreaIndex()` estimates the fine root area index for a given soil conductivity and maximum rhizosphere conductance.
- `root_fineRootBiomass()` estimates the biomass of fine roots (g dry/individual) for a given soil conductivity and maximum rhizosphere conductance.
- `root_rhizosphereMaximumConductance()` is the inverse of the preceeding function, i.e. it estimates rhizosphere conductance from soil conductivity and fine root biomass.

- `root_fineRootSoilVolume()` calculates the soil volume (m3) occupied with fine roots.
- `root_specificRootSurfaceArea()` returns the specific fine root area (cm2/g).
- `root_individualRootedGroundArea()` calculates the area (m2) covered by roots of an individual, for each soil layer.
- `root_horizontalProportions()` calculates the (horizontal) proportion of roots of each cohort in the water pool corresponding to itself and that of other cohorts, for each soil layer. Returns a list (with as many elements as cohorts) with each element being a matrix.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Schenk, H., Jackson, R., 2002. The global biogeography of roots. Ecol. Monogr. 72, 311–328.

Sperry, J. S., Y. Wang, B. T. Wolfe, D. S. Mackay, W. R. L. Anderegg, N. G. Mcdowell, and W. T. Pockman. 2016. Pragmatic hydraulic theory predicts stomatal responses to climatic water deficits. New Phytologist 212, 577–589.

See Also

spwb, spwb_ldrOptimization, forest2spwbInput, soil

Examples

```
#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

ntree = nrow(exampleforestMED$treeData)

#Initialize soil with default soil params
S = soil(defaultSoilParams())

#Calculate conic root system for trees
V1 = root_conicDistribution(Z=rep(2000,ntree), S$dVec)
print(V1)

#Calculate LDR root system for trees (Schenck & Jackson 2002)
V2 = root_ldrDistribution(Z50 = rep(200,ntree),
          Z95 = rep(1000,ntree), S$dVec)
print(V2)
```

scalingconductance *Scaling from conductivity to conductance*

Description

Functions used to scale from tissue conductivity to conductance of different elements of the continuum.

Usage

```
hydraulics_maximumSoilPlantConductance(krhizomax, krootmax,
                                       kstemmax, kleafmax)
hydraulics_soilPlantResistances(psiSoil, psiRhizo,
                                psiStem, PLCstem, psiLeaf,
                                krhizomax, n, alpha,
                                krootmax, rootc, rootd,
                                kstemmax, stemc, stemd,
                                kleafmax, leafc, leafd)
hydraulics_averageRhizosphereResistancePercent(krhizomax, n, alpha,
                                               krootmax, rootc, rootd,
                                               kstemmax, stemc, stemd,
                                        kleafmax, leafc, leafd, psiStep = -0.01)
hydraulics_findRhizosphereMaximumConductance(averageResistancePercent, n, alpha,
                                             krootmax, rootc, rootd,
                                             kstemmax, stemc, stemd,
                                       kleafmax, leafc, leafd, initialValue = 0)
hydraulics_maximumStemHydraulicConductance(xylemConductivity, refheight, Al2As, height,
                                           taper = FALSE)
hydraulics_rootxylemConductanceProportions(L, V)
hydraulics_referenceConductivityHeightFactor(refheight, height)
hydraulics_terminalConduitRadius(height)
hydraulics_taperFactorSavage(height)
hydraulics_stemWaterCapacity(Al2As, height, wd)
hydraulics_leafWaterCapacity(SLA, ld)
```

Arguments

psiSoil	Soil water potential (in MPa). A scalar or a vector depending on the function.
psiRhizo	Water potential (in MPa) in the rhizosphere (root surface).
psiStem	Water potential (in MPa) in the stem.
psiLeaf	Water potential (in MPa) in the leaf.
PLCstem	Percent loss of conductance (in %) in the stem.
L	Vector with the length of coarse roots (mm) for each soil layer.
V	Vector with the proportion [0-1] of fine roots within each soil layer.

krhizomax	Maximum rhizosphere hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
kleafmax	Maximum leaf hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
kstemmax	Maximum stem xylem hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
krootmax	Maximum root xylem hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
psiStep	Water potential precision (in MPa).
rootc, rootd	Parameters of the Weibull function for roots (root xylem vulnerability curve).
stemc, stemd	Parameters of the Weibull function for stems (stem xylem vulnerability curve).
leafc, leafd	Parameters of the Weibull function for leaves (leaf vulnerability curve).
n, alpha	Parameters of the Van Genuchten function (rhizosphere vulnerability curve).
averageResistancePercent	Average (across water potential values) resistance percent of the rhizosphere, with respect to total resistance (rhizosphere + root xylem + stem xylem).
initialValue	Initial value of rhizosphere conductance.
xylemConductivity	Xylem conductivity as flow per length of conduit and pressure drop (in kg·m-1·s-1·MPa-1).
Al2As	Leaf area to sapwood area (in m2·m-2).
height	Plant height (in cm).
refheight	Reference plant height of measurement of xylem conductivity (in cm).
taper	A boolean flag to indicate correction by taper of xylem conduits (Christoffersen et al. 2017).
SLA	Specific leaf area (mm2·mg-1).
wd	Wood density (g·cm-3).
ld	Leaf tissue density (g·cm-3).

Details

Details of the hydraulic model are given in a vignette.

Value

Values returned for each function are:

- `hydraulics_maximumSoilPlantConductance`: The maximum soil-plant conductance, in the same units as the input segment conductances.
- `hydraulics_averageRhizosphereResistancePercent`: The average percentage of resistance due to the rhizosphere, calculated across water potential values.
- `hydraulics_findRhizosphereMaximumConductance`: The maximum rhizosphere conductance value given an average rhizosphere resistance and the vulnerability curves of rhizosphere, root and stem elements.
- `hydraulics_taperFactorSavage`: Taper factor according to Savage et al. (2010).

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Christoffersen, B. O., M. Gloor, S. Fauset, N. M. Fyllas, D. R. Galbraith, T. R. Baker, L. Rowland, R. A. Fisher, O. J. Binks, S. A. Sevanto, C. Xu, S. Jansen, B. Choat, M. Mencuccini, N. G. McDowell, and P. Meir. 2016. Linking hydraulic traits to tropical forest function in a size-structured and trait-driven model (TFS v.1-Hydro). Geoscientific Model Development Discussions 9: 4227–4255.

Savage, V. M., L. P. Bentley, B. J. Enquist, J. S. Sperry, D. D. Smith, P. B. Reich, and E. I. von Allmen. 2010. Hydraulic trade-offs and space filling enable better predictions of vascular structure and function in plants. Proceedings of the National Academy of Sciences of the United States of America 107:22722–7.

Olson, M.E., Anfodillo, T., Rosell, J.A., Petit, G., Crivellaro, A., Isnard, S., León-Gómez, C., Alvarado-Cárdenas, L.O., & Castorena, M. 2014. Universal hydraulics of the flowering plants: Vessel diameter scales with stem length across angiosperm lineages, habits and climates. Ecology Letters 17: 988–997.

See Also

hydraulics_psi2K, hydraulics_supplyFunctionPlot, spwb, soil

Examples

```
kstemmax = 4 # in mmol·m-2·s-1·MPa-1
stemc = 3
stemd = -4 # in MPa
```

SFM_metric *Standard fuel models (Albini 1976, Scott & Burgan 2005)*

Description

Standard fuel models converted to metric system. Copied from package 'Rothermel' (Giorgio Vacchiano, Davide Ascoli).

Usage

```
data("SFM_metric")
```

Format

A data frame including standard fuel models as in Albini (1976) and Scott and Burgan (2005), to be used as input of `fire_Rothermel` function. All values converted to metric format.

`Fuel_Model_Type` A factor with levels `D` (for dynamic) or `S` (for static).

`Load_1h` Loading of 1h fuel class [t/ha].

`Load_10h` Loading of 10h fuel class [t/ha].

`Load_100h` Loading of 100h fuel class [t/ha]

`Load_Live_Herb` Loading of herbaceous fuels [t/ha]

`Load_Live_Woody` Loading of woody fuels [t/ha]

'`SA/V_1h`' Surface area to volume ratio of 1h fuel class [m2/m3]

'`SA/V_10h`' Surface area to volume ratio of 10h fuel class [m2/m3]

'`SA/V_100h`' Surface area to volume ratio of 100h fuel class [m2/m3]

'`SA/V_Live_Herb`' Surface area to volume ratio of herbaceous fuels [m2/m3]

'`SA/V_Live_Woody`' Surface area to volume ratio of woody fuels [m2/m3]

`Fuel_Bed_Depth` Fuel bed depth [cm]

`Mx_dead` Dead fuel moisture of extinction [percent]

`Heat_1h` Heat content of 1h fuel class [kJ/kg]

`Heat_10h` Heat content of 10h fuel class [kJ/kg]

`Heat_100h` Heat content of 100h fuel class [kJ/kg]

`Heat_Live_Herb` Heat content of herbaceous fuels [kJ/kg]

`Heat_Live_Woody` Heat content of woody fuels [kJ/kg]

Source

Albini, F. A. (1976). Computer-based models of wildland fire behavior: A users' manual. Ogden, UT: US Department of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station.

Scott, J., & Burgan, R. E. (2005). A new set of standard fire behavior fuel models for use with Rothermel's surface fire spread model. Gen. Tech. Rep. RMRSGTR-153. Fort Collins, CO: US Department of Agriculture, Forest Service, Rocky Mountain Research Station.

See Also

`fire_Rothermel`

Examples

```
data(SFM_metric)
```

shinyplot — *Shiny app with interactive plots*

Description

Creates a shiny app with interactive plots for simulation results and evaluation

Usage

```
shinyplot(out, measuredData = NULL, SpParams = NULL)
```

Arguments

out	An object of class spwb or growth
measuredData	A data frame with observed/measured values (see evaluation_plot).
SpParams	A data frame with species parameters (see SpParamsMED).

Details

Only run this function in interactive mode. When `measuredData` is not `NULL`, an additional panel is shown for evaluation plots.

Value

An object that represents the shiny app

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

plot.spwb, evaluation_plot

Examples

```
## Only run this example in interactive R sessions
if (interactive()) {
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params (4 layers)
```

```
examplesoil1 = soil(defaultSoilParams(4))

#Initialize control parameters
control = defaultControl("Granier")

#Initialize input
x1 = forest2spwbInput(exampleforestMED,examplesoil1, SpParamsMED, control)

#Call simulation function
S1<-spwb(x1, examplemeteo, latitude = 41.82592, elevation = 100)

#Load observed data (in this case the same simulation results with some added error)
data(exampleobs)

#Call interactive plot shiny app
shinyplot(S1, exampleobs, SpParamsMED)
}
```

---

soil *Soil initialization*

Description

Initializes soil parameters and state variables for its use in simulations.

Usage

```
soil(SoilParams, VG_PTF = "Toth", W = as.numeric(c(1)), SWE = 0)
## S3 method for class 'soil'
print(x, model="SX", ...)
redefineSoilLayers(SoilParams, widths = c(300, 700, 1000, 2000))
```

Arguments

SoilParams	A data frame of soil parameters (see an example in defaultSoilParams).
VG_PTF	Pedotransfer functions to obtain parameters for the van Genuchten-Mualem equations. Either "Carsel" (Carsel & Parrish 1988) or "Toth" (Toth et al. 2015).
W	A numerical vector with the initial relative water content of each soil layer.
SWE	Initial snow water equivalent of the snow pack on the soil surface (mm).
x	An object of class soil.
model	Either 'SX' or 'VG' for Saxton or Van Genuchten pedotransfer models.
widths	A numeric vector indicating the desired layer widths, in *mm*.
...	Additional parameters to print.

Details

Function print prompts a description of soil characteristics and state variables (water content and temperature) according to a water retention curve (either Saxton's or Van Genuchten's). Volume at field capacity is calculated assuming a soil water potential equal to -0.033 MPa. Parameter Temp is initialized as missing for all soil layers. Function redefineSoilLayers allows redefining soil layer widths of an input data frame of soil parameters.

Value

Function soil returns a list of class soil with the following elements:

- SoilDepth: Soil depth (in mm).
- W: State variable with relative water content of each layer (in as proportion relative to FC).
- Temp: State variable with temperature (in ºC) of each layer.
- Ksoil: Kappa parameter for infiltration.
- Gsoil: Gamma parameter for infiltration.
- dVec: Width of soil layers (in mm).
- sand: Sand percentage for each layer (in percent volume).
- clay: Clay percentage for each layer (in percent volume).
- om: Organic matter percentage for each layer (in percent volume).
- VG_alpha, VG_n, VG_theta_res, VG_theta_sat: Parameters for van Genuchten's pedotransfer functions, for each layer, corresponding to the USDA texture type.
- Ksat: Saturated soil conductivity for each layer (estimated using function soil_saturatedConductivitySX.
- macro: Macroporosity for each layer (estimated using Stolf et al. 2011).
- rfc: Percentage of rock fragment content for each layer.
- Kdrain: Saturated vertical hydraulic conductivity (mm/day) (i.e. how easy is deep drainage towards groundwater). Function soil estimates it as a function of soil saturated hydraulic conductivity, but should be parametrized as a function of bedrock material.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Carsel, R.F., & Parrish, R.S. 1988. Developing joint probability distributions of soil water retention characteristics. Water Resources Research 24: 755–769.

Tóth, B., Weynants, M., Nemes, A., Makó, A., Bilas, G., & Tóth, G. 2015. New generation of hydraulic pedotransfer functions for Europe. European Journal of Soil Science 66: 226–238.

Stolf, R., Thurler, A., Oliveira, O., Bacchi, S., Reichardt, K., 2011. Method to estimate soil macroporosity and microporosity based on sand content and bulk density. Rev. Bras. Ciencias do Solo 35, 447–459.

See Also

soil_psi2thetaSX, soil_psi2thetaVG, spwb, defaultSoilParams

Examples

```
# Initializes soil
s = soil(defaultSoilParams())

# Prints soil characteristics according to Saxton's water retention curve
print(s, model="SX")

# Prints soil characteristics according to Van Genuchten's water retention curve
print(s, model="VG")
```

---

soil hydrology *Soil infiltration and bare soil evaporation*

Description

Function hydrology_infiltrationAmount calculates the amount of water that infiltrates into the topsoil, according to the USDA SCS curve number method (Boughton 1989). The remaining is assumed to be lost as surface runoff. Function hydrology_soilEvaporationAmount calculates the amount of evaporation from bare soil, following Ritchie (1972). Function hydrology_snowMelt calculates the maximum amount of snowmelt according to Kergoat (1998).

Usage

```
hydrology_infiltrationAmount(input, Ssoil)
hydrology_infiltrationRepartition(I, dVec, macro, a = -0.005, b = 3)
hydrology_soilEvaporationAmount(DEF,PETs, Gsoil)
hydrology_soilEvaporation(soil, soilFunctions, pet, LgroundSWR, modifySoil = TRUE)
hydrology_snowMelt(tday, rad, LgroundSWR, elevation)
```

Arguments

input	A numeric vector of (daily) water input (in mm of water).
Ssoil	Soil water storage capacity (can be referred to topsoil) (in mm of water).
DEF	Water deficit in the (topsoil) layer.
PETs	Potential evapotranspiration at the soil surface.
Gsoil	Gamma parameter (maximum daily evaporation).
I	Soil infiltration (in mm of water).
dVec	Width of soil layers (in mm).
macro	Macroporosity of soil layers (in %).
soil	An object of class soil.

soilFunctions	Soil water retention curve and conductivity functions, either 'SX' (for Saxton) or 'VG' (for Van Genuchten).
pet	Potential evapotranspiration for a given day (mm)
LgroundSWR	Proportion of short-wave radiation (SWR) reaching the ground.
modifySoil	Boolean flag to indicate that the input soil object should be modified during the simulation.
tday	Average day temperature (ºC).
rad	Solar radiation (in MJ/m2/day).
elevation	Altitude above sea level (m).
a,b	Parameters of the extinction function used for water infitration.

Details

See description of infiltration and soil evaporation processes in De Caceres et al. (2015).

Value

Function hydrology_infiltrationAmount a vector of the same length as input containing the daily amount of water that infiltrates into the soil (in mm of water). Function hydrology_infiltrationRepartition estimates the amount of infiltrated water that reaches each soil layer. Function hydrology_soilEvaporationAmount returns the amount of water evaporated from the soil. Function hydrology_soilEvaporation returns a vector of water evaporated from each soil layer.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Boughton (1989). A review of the USDA SCS curve number method. - Australian Journal of Soil Research 27: 511-523.

De Cáceres M, Martínez-Vilalta J, Coll L, Llorens P, Casals P, Poyatos R, Pausas JG, Brotons L. (2015) Coupling a water balance model with forest inventory data to evaluate plant drought stress at the regional level. Agricultural and Forest Meteorology.

Kergoat L. (1998). A model for hydrological equilibrium of leaf area index on a global scale. Journal of Hydrology 212–213: 268–286.

Ritchie (1972). Model for predicting evaporation from a row crop with incomplete cover. - Water resources research.

See Also

spwb, hydrology_soilWaterInputs

Examples

```
SoilDepth = c(200,400,800,1200,1500)

#TOPSOIL LAYERS
d1 = pmin(SoilDepth, 300) #<300
#SUBSOIL LAYERS
d2 = pmax(0, pmin(SoilDepth-300,1200)) #300-1500 mm
#ROCK LAYER
d3 = 4000-(d1+d2) #From SoilDepth down to 4.0 m

TS_clay = 15
TS_sand = 25
SS_clay = 15
SS_sand = 25
RL_clay = 15
RL_sand = 25
TS_gravel = 20
SS_gravel = 40
RL_gravel = 95

Theta_FC1=soil_psi2thetaSX(TS_clay, TS_sand, -33) #in m3/m3
Theta_FC2=soil_psi2thetaSX(SS_clay, SS_sand, -33) #in m3/m3
Theta_FC3=soil_psi2thetaSX(RL_clay, RL_sand, -33) #in m3/m3
pcTS_gravel = 1-(TS_gravel/100)
pcSS_gravel = 1-(SS_gravel/100)
pcRL_gravel = 1-(RL_gravel/100)
MaxVol1 = (d1*Theta_FC1*pcTS_gravel)
MaxVol2 = (d2*Theta_FC2*pcSS_gravel)
MaxVol3 = (d3*Theta_FC3*pcRL_gravel)
V = MaxVol1+MaxVol2+MaxVol3

par(mar=c(5,5,1,1), mfrow=c(1,2))
NP = seq(0,60, by=1)
plot(NP,hydrology_infiltrationAmount(NP, V[1]), type="l", xlim=c(0,60), ylim=c(0,60),
     ylab="Infiltration (mm)", xlab="Net rainfall (mm)", frame=FALSE)
lines(NP,hydrology_infiltrationAmount(NP, V[2]), lty=2)
lines(NP,hydrology_infiltrationAmount(NP, V[3]), lty=3)
lines(NP,hydrology_infiltrationAmount(NP, V[4]), lty=4)
lines(NP,hydrology_infiltrationAmount(NP, V[5]), lty=5)
legend("topleft", bty="n", lty=1:5,
       legend=c(paste("d =", SoilDepth, "Vsoil =",round(V),"mm")))
plot(NP,NP-hydrology_infiltrationAmount(NP, V[1]), type="l", xlim=c(0,60), ylim=c(0,60),
     ylab="Runoff (mm)", xlab="Net rainfall (mm)", frame=FALSE)
lines(NP,NP-hydrology_infiltrationAmount(NP, V[2]), lty=2)
lines(NP,NP-hydrology_infiltrationAmount(NP, V[3]), lty=3)
lines(NP,NP-hydrology_infiltrationAmount(NP, V[4]), lty=4)
lines(NP,NP-hydrology_infiltrationAmount(NP, V[5]), lty=5)
legend("topleft", bty="n", lty=1:5,
       legend=c(paste("d =", SoilDepth,"Vsoil =",round(V),"mm")))
```

`soil texture and hydraulics`

Soil texture and hydraulics

Description

Collection of functions relating soil texture with soil hydraulics and soil water content (see description of each function in details).

Usage

```
soil_psi2thetaSX(clay, sand, psi, om = NA)
soil_psi2thetaVG(n, alpha, theta_res, theta_sat, psi)
soil_theta2psiSX(clay, sand, theta, om = NA)
soil_theta2psiVG(n, alpha, theta_res, theta_sat, theta)
soil_saturatedConductivitySX(clay, sand, om = NA, mmol = TRUE)
soil_USDAType(clay, sand)
soil_vanGenuchtenParamsCarsel(soilType)
soil_vanGenuchtenParamsToth(clay, sand, om, bd, topsoil)
soil_psi(soil, model="SX")
soil_theta(soil, model="SX")
soil_water(soil, model="SX")
soil_waterFC(soil, model="SX")
soil_waterWP(soil, model="SX")
soil_waterSAT(soil, model="SX")
soil_waterExtractable(soil, model="SX", minPsi = -5.0)
soil_thetaFC(soil, model="SX")
soil_thetaWP(soil, model="SX")
soil_thetaSAT(soil, model="SX")
soil_thetaSATSX(clay, sand, om = NA)
soil_waterTableDepth(soil, model="SX")
soil_rockWeight2Volume(pWeight, bulkDensity, rockDensity = 2.3)
soil_retentionCurvePlot(soil, model="SX", layer = 1,
                        psi = seq(0, -6.0, by=-0.01),
                        relative = TRUE, to = "SAT")
```

Arguments

`clay`	Percentage of clay (in percent weight).
`sand`	Percentage of sand (in percent weight).
`n, alpha, theta_res, theta_sat`	Parameters of the Van Genuchten-Mualem model (m = 1 - 1/n).
`psi`	Water potential (in MPa).
`theta`	Relative water content (in percent volume).
`om`	Percentage of organic matter (optional, in percent weight).

`mmol`	Boolean flag to indicate that saturated conductivity units should be returned in mmol/m/s/MPa. If `mmol = FALSE` then units are cm/day.
`bd`	Bulk density (in g/cm3).
`topsoil`	A boolean flag to indicate topsoil layer.
`soilType`	A string indicating the soil type.
`soil`	Soil object (returned by function `soil`).
`model`	Either 'SX' or 'VG' for Saxton's or Van Genuchten's water retention models; or 'both' to plot both retention models.
`layer`	Soil layer to be plotted.
`relative`	Boolean flag to indicate that retention curve should be relative to field capacity or saturation.
`to`	Either 'SAT' (saturation) or 'FC' (field capacity).
`minPsi`	Minimum water potential (in MPa) to calculate the amount of extractable water.
`pWeight`	Percentage of corresponding to rocks, in weight.
`bulkDensity`	Bulk density of the soil fraction (g/cm3).
`rockDensity`	Rock density (g/cm3).

Details

- `soil_psi2thetaSX()` and `soil_theta2psiSX()` calculate water potentials (MPa) and water contents (theta) using texture data the formulae of Saxton et al. (1986) or Saxton & Rawls (2006) depending on whether organic matter is available.
- `soil_psi2thetaVG()` and `soil_theta2psiVG()` to the same calculations as before, but using the Van Genuchten - Mualem equations (Wösten & van Genuchten 1988).
- `soil_saturatedConductivitySX()` returns the saturated conductivity of the soil (in cm/day or mmol/m/s/MPa), estimated from formulae of Saxton et al. (1986) or Saxton & Rawls (2006) depending on whether organic matter is available.
- `soil_USDAType()` returns the USDA type (a string) for a given texture.
- `soil_vanGenuchtenParamsCarsel()` gives parameters for van Genuchten-Mualem equations (alpha, n, theta_res and theta_sat, where alpha is in MPa-1) for a given texture type (Leij et al. 1996)
- `soil_vanGenuchtenParamsToth()` gives parameters for van Genuchten-Mualem equations (alpha, n, theta_res and theta_sat, where alpha is in MPa-1) for a given texture, organic matter and bulk density (Toth et al. 2015).
- `soil_psi()` returns the water potential (MPa) of each soil layer, according to its water retention model.
- `soil_theta()` returns the moisture content (as percent of soil volume) of each soil layer, according to its water retention model.
- `soil_water()` returns the water volume (mm) of each soil layer, according to its water retention model.
- `soil_waterExtractable()` returns the water volume (mm) extractable from the soil according to its water retention curves and up to a given soil water potential.

- soil_waterFC() and soil_thetaFC() calculate the water volume (in mm) and moisture content (as percent of soil volume) of each soil layer at field capacity, respectively.
- soil_waterWP() and soil_thetaWP() calculate the water volume (in mm) and moisture content (as percent of soil volume) of each soil layer at wilting point (-1.5 MPa), respectively.
- soil_waterSAT(), soil_thetaSATSX() and soil_thetaSAT() calculate the saturated water volume (in mm) and moisture content (as percent of soil volume) of each soil layer.
- soil_waterTableDepth() returns water table depth in mm from surface.
- soil_rockWeight2Volume() transforms rock percentage from weight to volume basis.
- soil_retentionCurvePlot() allows ploting the water retention curve of a given soil layer.

Value

Depends on the function (see details).

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Leij, F.J., Alves, W.J., Genuchten, M.T. Van, Williams, J.R., 1996. The UNSODA Unsaturated Soil Hydraulic Database User's Manual Version 1.0.

Saxton, K.E., Rawls, W.J., Romberger, J.S., Papendick, R.I., 1986. Estimating generalized soil-water characteristics from texture. Soil Sci. Soc. Am. J. 50, 1031–1036.

Saxton, K.E., Rawls, W.J., 2006. Soil water characteristic estimates by texture and organic matter for hydrologic solutions. Soil Sci. Soc. Am. J. 70, 1569. doi:10.2136/sssaj2005.0117

Wösten, J.H.M., & van Genuchten, M.T. 1988. Using texture and other soil properties to predict the unsaturated soil hydraulic functions. Soil Science Society of America Journal 52: 1762–1770.

Tóth, B., Weynants, M., Nemes, A., Makó, A., Bilas, G., & Tóth, G. 2015. New generation of hydraulic pedotransfer functions for Europe. European Journal of Soil Science 66: 226–238.

See Also

soil

Examples

```
#Determine USDA soil texture type
type = soil_USDAType(clay=40, sand=10)
type

#Van Genuchten's params (bulk density = 1.3 g/cm)
vg = soil_vanGenuchtenParamsToth(40,10,1,1.3,TRUE)
vg

# Initialize soil object with default params
s = soil(defaultSoilParams())
```

```
# Plot Saxton's and Van Genuchten's water retention curves
soil_retentionCurvePlot(s, model="both")
```

---

soil thermodynamics *Soil thermodynamic functions*

Description

Functions soil_thermalConductivity and soil_thermalCapacity calculate thermal conductivity and thermal capacity for each soil layer, given its texture and water content. Functions soil_temperatureGradient and soil_temperatureChange are used to calculate soil temperature gradients (in ºC/m) and temporal temperature change (in ºC/s) given soil layer texture and water content (and possibly including heat flux from above).

Usage

```
soil_thermalConductivity(soil, model = "SX")
soil_thermalCapacity(soil, model = "SX")
soil_temperatureChange(dVec, Temp, sand, clay, W, Theta_FC, Gdown)
soil_temperatureGradient(dVec, Temp)
```

Arguments

soil	Soil object (returned by function soil).
model	Either 'SX' or 'VG' for Saxton's or Van Genuchten's pedotransfer models.
dVec	Width of soil layers (in mm).
Temp	Temperature (in ºC) for each soil layer.
clay	Percentage of clay (in percent weight) for each layer.
sand	Percentage of sand (in percent weight) for each layer.
W	Soil moisture (in percent of field capacity) for each layer.
Theta_FC	Relative water content (in percent volume) at field capacity for each layer.
Gdown	Downward heat flux from canopy to soil (in W·m-2).

Value

Function soil_thermalConductivity returns a vector with values of thermal conductivity (W/m/ºK) for each soil layer. Function soil_thermalCapacity returns a vector with values of heat storage capacity (J/m3/ºK) for each soil layer. Function soil_temperatureGradient returns a vector with values of temperature gradient between consecutive soil layers. Function soil_temperatureChange returns a vector with values of instantaneous temperature change (ºC/s) for each soil layer.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Cox, P.M., Betts, R.A., Bunton, C.B., Essery, R.L.H., Rowntree, P.R., & Smith, J. 1999. The impact of new land surface physics on the GCM simulation of climate and climate sensitivity. Climate Dynamics 15: 183–203.

Dharssi, I., Vidale, P.L., Verhoef, A., MacPherson, B., Jones, C., & Best, M. 2009. New soil physical properties implemented in the Unified Model at PS18. 9–12.

See Also

soil

Examples

```
examplesoil = soil(defaultSoilParams())
soil_thermalConductivity(examplesoil)
soil_thermalCapacity(examplesoil)

#Values change when altering water content (drier layers have lower conductivity and capacity)
examplesoil$W = c(0.1, 0.4, 0.7, 1.0)
soil_thermalConductivity(examplesoil)
soil_thermalCapacity(examplesoil)
```

Species values *Species description functions*

Description

Functions to calculate attributes of a forest object by species or to extract species parameters from a species parameter table (SpParamsMED).

Usage

```
species_basalArea(x, SpParams)
species_cover(x, SpParams)
species_density(x, SpParams, mode = "MED")
species_foliarBiomass(x, SpParams, gdd = NA, mode = "MED")
species_fuel(x, SpParams, gdd = NA, includeDead = TRUE, mode = "MED")
species_LAI(x, SpParams, gdd = NA, mode = "MED")
species_phytovolume(x, SpParams)
species_parameter(SP, SpParams, parName, fillMissing = TRUE)
species_characterParameter(SP, SpParams, parName)
```

Arguments

x	An object of class forest.
SpParams	A data frame with species parameters (see SpParamsMED).
gdd	Growth degree days (to account for leaf phenology effects).

includeDead	A flag to indicate that standing dead fuels (dead branches) are included.
mode	Calculation mode, either "MED" or "US".
SP	An integer vector of species codes.
parName	A string with a parameter name.
fillMissing	A boolean flag to try imputation on missing values.

Value

A vector with values for each species in SpParams:

- species_basalArea: Species basal area (m2/ha).
- species_cover: Shrub cover (in percent).
- species_density: Plant density (ind/ha). Tree density is directly taken from the forest object, while the shrub density is estimated from cover and height by calculating the area of a single individual.
- species_foliarBiomass: Standing biomass of leaves (in kg/m2).
- species_fuel: Fine fuel load (in kg/m2).
- species_LAI: Leaf area index (m2/m2).
- species_phytovolume: Shrub phytovolume (m3/m2).
- species_parameter: A numeric vector with the parameter values of each input species.
- species_characterParameter: A character vector with the parameter values of each input species.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwb, forest, plant_basalArea, summary.forest

Examples

```
# Default species parameterization
data(SpParamsMED)

# Load example plot
data(exampleforestMED)

# Species basal area in the forest plot
species_basalArea(exampleforestMED, SpParamsMED)

# Value of parameter `Psi_Extract` for species 157 (Pinus halepensis)
# and 176 (Quercus ilex)
species_parameter(c(157,176), SpParamsMED, "Psi_Extract")
```

SpParams *Data tables with species parameter definition and values for Mediterranean and US*

Description

A data sets of species parameter definition and values, the latter resulting from existing databases, fit to empirical data or expert-based guesses.

Usage

```
data("SpParamsDefinition")
data("SpParamsMED")
data("SpParamsUS")
```

Format

Data frame `SpParamsDefinition` has parameters in rows and columns 'ParameterName', 'ParameterGroup', 'Definition', 'Type' and 'Units'. Data frames `SpParamsMED` and `SpParamsUS` have species or genus as rows and column names equal to parameter names in `SpParamsDefinition`.

Details

Plant trait parameter sources are listed in the bibliography section. Details of the procedures used to obtain the species parameter tables can be found in the article 'Species parameterization for Spain' at https://emf-creaf.github.io/medfate/.

Source

Asse, D., Randin, C. F., Bonhomme, M., Delestrade, A., & Chuine, I. (2020). Process-based models outcompete correlative models in projecting spring phenology of trees in a future warmer climate. Agricultural and Forest Meteorology, 285–286(January), 107931. https://doi.org/10.1016/j.agrformet.2020.107931

Bartlett, Megan K, Christine Scoffoni, and Lawren Sack. 2012. The determinants of leaf turgor loss point and prediction of drought tolerance of species and biomes: a global meta-analysis. Ecology Letters 15 (5): 393–405. https://doi.org/10.1111/j.1461-0248.2012.01751.x.

Choat, Brendan, Steven Jansen, Tim J Brodribb, Hervé Cochard, Sylvain Delzon, Radika Bhaskar, Sandra J Bucci, et al. 2012. Global convergence in the vulnerability of forests to drought. Nature 491 (7426): 752–55. https://doi.org/10.1038/nature11688.

De Cáceres M, Casals P, Gabriel E, Castro X (2019) Scaling-up individual-level allometric equations to predict stand-level fuel loading in Mediterranean shrublands. Ann For Sci 76:87 (doi:10.1007/s13595-019-0873-4).

Delpierre, N., Dufrêne, E., Soudani, K., Ulrich, E., Cecchini, S., Boé, J., & François, C. (2009). Modelling interannual and spatial variability of leaf senescence for three deciduous tree species in France. Agricultural and Forest Meteorology, 149(6–7), 938–948. https://doi.org/10.1016/j.agrformet.2008.11.014

DGCN (2005). Tercer Inventario Forestal Nacional (1997-2007): Catalunya. Dirección General de Conservación de la Naturaleza, Ministerio de Medio Ambiente, Madrid.

Duursma, Remko A., Christopher J. Blackman, Rosana López, Nicolas K. Martin-StPaul, Hervé Cochard, and Belinda E. Medlyn. 2018. On the minimum leaf conductance: its role in models of plant water use, and ecological and environmental controls. New Phytologist, no. September. https://doi.org/10.1111/nph.15395.

Hoshika, Yasutomo, Yutaka Osada, Alessandra de Marco, Josep Peñuelas, and Elena Paoletti. 2018. Global diurnal and nocturnal parameters of stomatal conductance in woody plants and major crops. Global Ecology and Biogeography 27 (2): 257–75. https://doi.org/10.1111/geb.12681.

Kattge, J., S. Díaz, S. Lavorel, I. C. Prentice, P. Leadley, G. Bonisch, E. Garnier, et al. 2011. TRY - a global database of plant traits. Global Change Biology 17 (9): 2905–35. https://doi.org/10.1111/j.1365-2486.2011.02451.x.

Manzoni, Stefano, Giulia Vico, Amilcare Porporato, and Gabriel Katul. 2013. Biological constraints on water transport in the soil-plant-atmosphere system. Advances in Water Resources 51: 292–304. https://doi.org/10.1016/j.advwatres.2012.03.016.

Martin-StPaul, Nicolas, Sylvain Delzon, and Hervé Cochard. 2017. Plant resistance to drought depends on timely stomatal closure. Ecology Letters 20 (11): 1437–47. https://doi.org/10.1111/ele.12851.

Mencuccini, Maurizio, Teresa Rosas, Lucy Rowland, Brendan Choat, Hans Cornelissen, Steven Jansen, Koen Kramer, et al. 2019. Leaf economics and plant hydraulics drive leaf : wood area ratios. New Phytologist. https://doi.org/10.1111/nph.15998.

Pausas, Juli G., R. Brandon Pratt, Jon E. Keeley, Anna L. Jacobsen, Aaron R. Ramirez, Alberto Vilagrosa, Susana Paula, Iolana N. Kaneakua-Pia, and Stephen D. Davis. 2016. Towards understanding resprouting at the global scale. New Phytologist 209 (3): 945–54. https://doi.org/10.1111/nph.13644.

Sanchez-Martinez, Pablo, Jordi Martinez-Vilalta, Kyle G. Dexter, Ricardo A. Segovia, and Maurizio Mencuccini. 2020. Adaptation and coordinated evolution of plant hydraulic traits. Ecology Letters, August, ele.13584. https://doi.org/10.1111/ele.13584.

Tavşanoğlu, Çağatay, and Juli G. Pausas. 2018. A functional trait database for Mediterranean Basin plants. Scientific Data 5: 1–18. https://doi.org/10.1038/sdata.2018.135.

Vitasse, Y., François, C., Delpierre, N., Dufrêne, E., Kremer, A., Chuine, I., & Delzon, S. (2011). Assessing the effects of climate change on the phenology of European temperate trees. Agricultural and Forest Meteorology, 151(7), 969–980. https://doi.org/10.1016/j.agrformet.2011.03.003

Yebra, Marta, Gianluca Scortechini, Abdulbaset Badi, María Eugenia Beget, Matthias M. Boer, Ross Bradstock, Emilio Chuvieco, et al. 2019. Globe-LFMC, a global plant water status database for vegetation ecophysiology and wildfire applications. Scientific Data 6 (1): 155. https://doi.org/10.1038/s41597-019-0164-9.

Zanne, A. E., G. Lopez-Gonzalez, D. A. Coomes, J. Ilic, S. Jansen, S. L. Lewis, R. B. Miller, N. G. Swenson, M. C. Wiemann, and J. Chave. 2009. Global wood density database. http://datadryad.org/handle/10255/dryad.235.

See Also

spwb

Examples

```
data(SpParamsMED)
```

spwb *Soil-plant water balance*

Description

Function spwb() is a water balance model that determines changes in soil moisture, soil water potentials, plant transpiration and drought stress at daily steps for a given forest stand during a period specified in the input climatic data. Function pwb() performs plant water balance only (i.e. soil moisture dynamics is an input) at daily steps for a given forest stand during a period specified in the input climatic data. On both simulation functions plant transpiration and photosynthesis processes are conducted with different level of detail depending on the transpiration mode.

Usage

```
spwb(x, meteo,  latitude , elevation = NA, slope = NA, aspect = NA)
pwb(x, meteo, W, latitude, elevation = NA, slope = NA, aspect = NA,
    canopyEvaporation = numeric(0),
    snowMelt = numeric(0),
    soilEvaporation = numeric(0))
```

Arguments

x
: An object of class spwbInput.

meteo
: A data frame with daily meteorological data series. Row names of the data frame should correspond to date strings with format "yyyy-mm-dd" (see Date). When using the 'Granier' transpiration mode the following columns are required:

- MeanTemperature: Mean temperature (in degrees Celsius).
- Precipitation: Precipitation (in mm).
- Radiation: Solar radiation (in MJ/m2/day), required only if snowpack = TRUE.
- PET: Potential evapotranspiration (in mm).

When using the 'Sperry' transpiration mode the following columns are required:

- MeanTemperature: Mean temperature (in degrees Celsius).
- MinTemperature: Minimum temperature (in degrees Celsius).
- MaxTemperature: Maximum temperature (in degrees Celsius).
- MinRelativeHumidity: Minimum relative humidity (in percent).
- MaxRelativeHumidity: Maximum relative humidity (in percent).
- Precipitation: Precipitation (in mm).
- Radiation: Solar radiation (in MJ/m2/day).
- WindSpeed: Wind speed (in m/s). If not available, this column can be left with NA values.
- CO2: Atmospheric (abovecanopy) CO2 concentration (in ppm). This column may not exist, or can be left with NA values. In both cases simulations will assume a constant value specified in defaultControl.

W
: A matrix with the same number of rows as meteo and as many columns as soil layers, containing the soil moisture of each layer as proportion of field capacity.

latitude
: Latitude (in degrees).

elevation, slope, aspect
: Elevation above sea level (in m), slope (in degrees) and aspect (in degrees from North). Required when using the 'Sperry' transpiration mode. Elevation is also required for 'Granier' if snowpack dynamics are simulated.

canopyEvaporation
: A vector of daily canopy evaporation (from interception) values (mm). The length should match the number of rows in meteo.

soilEvaporation
: A vector of daily bare soil evaporation values (mm). The length should match the number of rows in meteo.

snowMelt
: A vector of daily snow melt values (mm). The length should match the number of rows in meteo.

Details

The model using 'Granier' transpiration mode is illustrated by function transp_transpirationGranier and described in De Caceres et al. (2015). Simulations using the 'Sperry' transpiration mode are computationally much more expensive, are described in De Cáceres et al. (2021) and are illustrated by function transp_transpirationSperry.

Value

Function spwb returns a list of class 'spwb' whereas Function pwb returns a list of class 'pwb'. There are many elements in common in these lists, so they are listed here together:

- "latitude": Latitude (in degrees) given as input.
- "topography": Vector with elevation, slope and aspect given as input.
- "spwbInput": An (unmodified) copy of the object x of class spwbInput given as input (note that x is modified by the simulation function).
- "WaterBalance": A data frame where different variables (in columns) are given for each simulated day (in rows):
  – "PET": Potential evapotranspiration (in mm).
  – "Precipitation": Input precipitation (in mm).
  – "Rain": Precipitation as rain (in mm).
  – "Snow": Precipitation as snow (in mm).
  – "NetRain": Net rain, after accounting for interception (in mm).
  – "Infiltration": The amount of water infiltrating into the soil (in mm).
  – "Runoff": The amount of water exported via surface runoff (in mm).
  – "DeepDrainage": The amount of water exported via deep drainage (in mm).
  – "Evapotranspiration": Evapotranspiration (in mm).
  – "SoilEvaporation": Bare soil evaporation (in mm).

  - "PlantExtraction": Amount of water extracted from soil by plants (in mm) (can only be different from transpiration for transpirationMode = "Sperry" when capacitance is considered).
  - "Transpiration": Plant transpiration (considering all soil layers) (in mm).
  - "HydraulicRedistribution": Water redistributed among soil layers, transported through the plant hydraulic network (only for transpirationMode = "Sperry").

- "EnergyBalance": A data frame with the daily values of energy balance components for the soil and the canopy (only for transpirationMode = "Sperry").
- "Temperature": A data frame with the daily values of minimum/mean/maximum temperatures for the atmosphere (input), canopy and soil (only for transpirationMode = "Sperry").
- "Soil": A data frame where different variables (in columns) are given for each simulated day (in rows):
  - "W.1", ..., "W.k": Relative soil moisture content (relative to field capacity) in each soil layer.
  - "ML.1", ..., "ML.k": Soil water volume in each soil layer (in L/m2).
  - "MLTot": Total soil water volume (in L/m2).
  - "SWE": Snow water equivalent (mm) of the snow pack.
  - "PlantExt.1", ..., "PlantExt.k": Plant extraction from each soil layer (in mm).
  - "HydraulicInput.1", ..., "HydraulicInput.k": Water that entered the layer coming from other layers and transported via the plant hydraulic network (in mm) (only for transpirationMode = "Sperry").
  - "psi.1", ..., "psi.k": Soil water potential in each soil layer (in MPa).
- "Stand": A data frame where different variables (in columns) are given for each simulated day (in rows):
  - "LAI": LAI of the stand (including live and dead leaves) (in m2/m2).
  - "LAIlive": LAI of the stand assuming all leaves are unfolded (in m2/m2).
  - "LAIexpanded": LAI of the stand of leaves actually unfolded (in m2/m2).
  - "LAIdead": LAI of the stand corresponding to dead leaves (in m2/m2).
  - "Cm": Water retention capacity of the canopy (in mm) (accounting for leaf phenology).
  - "LgroundPAR": The proportion of PAR that reaches the ground (accounting for leaf phenology).
  - "LgroundSWR": The proportion of SWR that reaches the ground (accounting for leaf phenology).
- "Plants": A list of daily results for plant cohorts (see below).
- "subdaily": A list of objects of class spwb_day, one per day simulated (only if required in control parameters, see defaultControl).

When transpirationMode = "Granier", element "Plants" is a list with the following subelements:

- "LAI": A data frame with the daily leaf area index for each plant cohort.
- "AbsorbedSWRFraction": A data frame with the fraction of SWR absorbed by each plant cohort.

- `"Transpiration"`: A data frame with the amount of daily transpiration (in mm) for each plant cohort.
- `"GrossPhotosynthesis"`: A data frame with the amount of daily gross photosynthesis (in g C·m-2) for each plant cohort.
- `"PlantPsi"`: A data frame with the average daily water potential of each plant (in MPa).
- `"PlantStress"`: A data frame with the amount of daily stress [0-1] suffered by each plant cohort (relative whole-plant conductance).

If `transpirationMode="Sperry"`, element `"Plants"` is a list with the following subelements:

- `"LAI"`: A data frame with the daily leaf area index for each plant cohort.
- `"AbsorbedSWR"`: A data frame with the daily SWR absorbed by each plant cohort.
- `"NetLWR"`: A data frame with the daily net LWR by each plant cohort.
- `"Transpiration"`: A data frame with the amount of daily transpiration (in mm) for each plant cohorts.
- `"GrossPhotosynthesis"`: A data frame with the amount of daily gross photosynthesis (in g C·m-2) for each plant cohort.
- `"NetPhotosynthesis"`: A data frame with the amount of daily net photosynthesis (in g C·m-2) for each plant cohort.
- `"dEdP"`: A data frame with mean daily values of soil-plant conductance (derivative of the supply function) for each plant cohort.
- `"PlantWaterBalance"`: A data frame with the daily balance between transpiration and soil water extraction for each plant cohort.
- `"SunlitLeaves"` and `"ShadeLeaves"`: A list with daily results for sunlit and shade leaves:
  - `"PsiMin"`: A data frame with the minimum (midday) daily sunlit or shade leaf water potential (in MPa).
  - `"PsiMax"`: A data frame with the maximum (predawn) daily sunlit or shade leaf water potential (in MPa).
- `"LeafPsiMin"`: A data frame with the minimum (midday) daily (average) leaf water potential of each plant (in MPa).
- `"LeafPsiMax"`: A data frame with the maximum (predawn) daily (average) leaf water potential of each plant (in MPa).
- `"LeafRWC"`: A data frame with the average daily leaf relative water content of each plant (in percent).
- `"StemRWC"`: A data frame with the average daily stem relative water content of each plant (in percent).
- `"LeafSympRWC"`: A data frame with the average daily leaf symplasm relative water content of each plant (in percent).
- `"StemSympRWC"`: A data frame with the average daily stem symplasm relative water content of each plant (in percent).
- `"StemPsi"`: A data frame with the minimum daily stem water potential of each plant (in MPa).
- `"StemPLC"`: A data frame with the average daily proportion of stem conductance loss of each plant ([0-1]).

- "RootPsi": A data frame with the minimum daily root water potential of each plant (in MPa).
- "RhizoPsi": A list of data frames (one per plant cohort) with the minimum daily root water potential of each plant (in MPa).
- "PlantStress": A data frame with the amount of daily stress [0-1] suffered by each plant cohort (relative whole-plant conductance).

Note

State variables stored in objects x are modified during the simulation, unless control parameter modifyInput = FALSE.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

De Cáceres M, Martínez-Vilalta J, Coll L, Llorens P, Casals P, Poyatos R, Pausas JG, Brotons L. (2015) Coupling a water balance model with forest inventory data to predict drought stress: the role of forest structural changes vs. climate changes. Agricultural and Forest Meteorology 213: 77-90 (doi:10.1016/j.agrformet.2015.06.012).

De Cáceres M, Mencuccini M, Martin-StPaul N, Limousin JM, Coll L, Poyatos R, Cabon A, Granda V, Forner A, Valladares F, Martínez-Vilalta J (2021) Unravelling the effect of species mixing on water use and drought stress in holm oak forests: a modelling approach. Agricultural and Forest Meteorology 296 (doi:10.1016/j.agrformet.2020.108233).

See Also

spwbInput, spwb_day, plot.spwb, spwb_ldrOptimization, forest

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params (4 layers)
examplesoil1 = soil(defaultSoilParams(4))

#Initialize control parameters
control = defaultControl("Granier")

#Initialize input
x1 = forest2spwbInput(exampleforestMED,examplesoil1, SpParamsMED, control)
```

```
#Call simulation function
S1<-spwb(x1, examplemeteo, latitude = 41.82592, elevation = 100)

#Plot results
plot(S1)

#Monthly summary (averages) of soil water balance
summary(S1, freq="months",FUN=mean, output="Soil")


## Not run:
examplesoil2 = soil(defaultSoilParams(4))

#Switch to 'Sperry' transpiration mode
control = defaultControl("Sperry")

#Initialize input
x2 = forest2spwbInput(exampleforestMED,examplesoil2, SpParamsMED, control)

#Call simulation function (11 days)
d = 100:110
S2<-spwb(x2, examplemeteo[d,], latitude = 41.82592, elevation = 100)

# Run the model with 'Sperry' transpiration mode using the water balance of
# simulated with the 'Granier' model
WS = as.matrix(S1$Soil[, c("W.1", "W.2", "W.3", "W.4")])
P2<-pwb(x2, examplemeteo[d,], latitude = 41.82592, elevation = 100,
        W = WS[d,],
        canopyEvaporation = S1$WaterBalance$Interception[d],
        snowMelt = S1$WaterBalance$Snowmelt[d],
        soilEvaporation = S1$WaterBalance$SoilEvaporation[d])

## End(Not run)
```

---

spwb_day *Single day simulation*

Description

Function spwb_day performs water balance for a single day and growth_day performs water and carbon balance for a single day.

Usage

```
spwb_day(x, date, tmin, tmax, rhmin, rhmax, rad, wind,
         latitude, elevation, slope, aspect, prec, runon = 0.0)
growth_day(x, date, tmin, tmax, rhmin, rhmax, rad, wind,
           latitude, elevation, slope, aspect, prec, runon = 0.0)
```

Arguments

x	An object of class spwbInput or growthInput.
date	Date as string "yyyy-mm-dd".
tmin, tmax	Minimum and maximum temperature (in degrees Celsius).
rhmin, rhmax	Minimum and maximum relative humidity (in percent).
rad	Solar radiation (in MJ/m2/day).
wind	Wind speed (in m/s).
prec	Precipitation (in mm).
latitude	Latitude (in degrees). Required when using the 'Sperry' transpiration mode.
elevation, slope, aspect	Elevation above sea level (in m), slope (in degrees) and aspect (in degrees from North). Required when using the 'Sperry' transpiration mode.
runon	Surface water amount running on the target area from upslope (in mm).

Details

Detailed model description is available in the vignettes section. The model using 'Granier' transpiration mode is described in De Caceres et al. (2015). Simulations using the 'Sperry' transpiration mode are computationally much more expensive, are described in De Cáceres et al. (2021) and are illustrated by function transp_transpirationSperry.

Value

Function `spwb_day()` returns a list of class `spwb_day` with the following elements:

- `"cohorts"`: A data frame with cohort information, copied from spwbInput.
- `"WaterBalance"`: A vector of water balance components (rain, snow, net rain, infiltration, ...) for the simulated day, equivalent to one row of 'WaterBalance' object given in spwb.
- `"Soil"`: A data frame with results for each soil layer:
  - `"SoilEvaporation"`: Water evaporated from the soil surface (in mm).
  - `"HydraulicInput"`: Water entering each soil layer from other layers, transported via plant hydraulic network (in mm) (only for `transpirationMode = "Sperry"`).
  - `"HydraulicOutput"`: Water leaving each soil layer (going to other layers or the transpiration stream) (in mm) (only for `transpirationMode = "Sperry"`).
  - `"PlantExtraction"`: Water extracted by plants from each soil layer (in mm).
  - `"psi"`: Soil water potential (in MPa).
- `"Stand"`: A named vector with with stand values for the simulated day, equivalent to one row of 'Stand' object returned by spwb.
- `"Plants"`: A data frame of results for each plant cohort (see transp_transpirationGranier or transp_transpirationSperry).

The following items are only returned when `transpirationMode = "Sperry"`:

- `"EnergyBalance"`: Energy balance of the stand (see transp_transpirationSperry).

- "RhizoPsi": Minimum water potential (in MPa) inside roots, after crossing rhizosphere, per cohort and soil layer.
- "SunlitLeaves" and "ShadeLeaves": For each leaf type, a data frame with values of LAI, Vmax298 and Jmax298 for leaves of this type in each plant cohort.
- "ExtractionInst": Water extracted by each plant cohort during each time step.
- "PlantsInst": A list with instantaneous (per time step) results for each plant cohort (see transp_transpirationSperry).
- "LightExtinction": A list of information regarding radiation balance through the canopy, as returned by function light_instantaneousLightExtinctionAbsortion.
- "CanopyTurbulence": Canopy turbulence (see wind_canopyTurbulence).

Note

Object x is modified during the simulation, unless control parameter modifyInput = FALSE.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

De Cáceres M, Martínez-Vilalta J, Coll L, Llorens P, Casals P, Poyatos R, Pausas JG, Brotons L. (2015) Coupling a water balance model with forest inventory data to predict drought stress: the role of forest structural changes vs. climate changes. Agricultural and Forest Meteorology (doi:10.1016/j.agrformet.2015.06.012).

De Cáceres M, Mencuccini M, Martin-StPaul N, Limousin JM, Coll L, Poyatos R, Cabon A, Granda V, Forner A, Valladares F, Martínez-Vilalta J (2021) Unravelling the effect of species mixing on water use and drought stress in holm oak forests: a modelling approach. Agricultural and Forest Meteorology 296 (doi: 10.1016/j.agrformet.2020.108233).

See Also

spwbInput, spwb, plot.spwb_day, growthInput, growth, plot.growth_day

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize control parameters
control = defaultControl("Granier")

# Day to be simulated
```

```
d = 100

#Simulate water balance one day only (Granier)
examplesoil = soil(defaultSoilParams(2))
x1 = forest2spwbInput(exampleforestMED,examplesoil, SpParamsMED, control)
sd1<-spwb_day(x1, rownames(examplemeteo)[d],
             examplemeteo$MinTemperature[d], examplemeteo$MaxTemperature[d],
             examplemeteo$MinRelativeHumidity[d], examplemeteo$MaxRelativeHumidity[d],
             examplemeteo$Radiation[d], examplemeteo$WindSpeed[d],
             latitude = 41.82592, elevation = 100, slope=0, aspect=0,
             prec = examplemeteo$Precipitation[d])


#Simulate water balance for one day only (Sperry's mode)
control = defaultControl("Sperry")
examplesoil2 = soil(defaultSoilParams(2))
x2 = forest2spwbInput(exampleforestMED,examplesoil2, SpParamsMED, control)
sd2<-spwb_day(x2, rownames(examplemeteo)[d],
             examplemeteo$MinTemperature[d], examplemeteo$MaxTemperature[d],
             examplemeteo$MinRelativeHumidity[d], examplemeteo$MaxRelativeHumidity[d],
             examplemeteo$Radiation[d], examplemeteo$WindSpeed[d],
             latitude = 41.82592, elevation = 100, slope=0, aspect=0,
             prec = examplemeteo$Precipitation[d])

#Plot plant transpiration (see function 'plot.swb.day()')
plot(sd2)


#Simulate water and carbon balance for one day only
examplesoil3 = soil(defaultSoilParams(2))
x3 = forest2growthInput(exampleforestMED,examplesoil3, SpParamsMED, control)
sd3<-growth_day(x3, rownames(examplemeteo)[d],
             examplemeteo$MinTemperature[d], examplemeteo$MaxTemperature[d],
             examplemeteo$MinRelativeHumidity[d], examplemeteo$MaxRelativeHumidity[d],
             examplemeteo$Radiation[d], examplemeteo$WindSpeed[d],
             latitude = 41.82592, elevation = 100, slope=0, aspect=0,
             prec = examplemeteo$Precipitation[d])
```

`spwb_ldrCalibration` *Calibration of root distribution*

Description

The function `spwb_ldrCalibration` calibrates the species root distribution within `spwb`, given the arguments `x`, `meteo`, `psi_crit`, `obs` and `calibVar`. This calibration is based on reference measured values. These reference measured values can be `Soil water content`, `Total tranpiration` or `Transpiration by cohort`. Return the calibrated root distribution for each tree species (no shrub calibration is done), expressed as parameters of the function `root_ldrDistribution`.

Usage

```
spwb_ldrCalibration(x, meteo, calibVar, obs,
                    RZmin = 301, RZmax = 4000,
                    V1min = 0.01, V1max = 0.94, resolution = 20, heat_stop = 0,
                    transformation = "identity", verbose = FALSE)
```

Arguments

- `x` An object of class `spwbInput`.
- `meteo` A data frame with daily meteorological data series. When using the 'Granier' transpiration mode the following columns are required:
  - `DOY`: Day of the year (Julian day).
  - `Precipitation`: Precipitation (in mm).
  - `MeanTemperature`: Mean temperature (in degrees Celsius).
  - `PET`: Potential evapotranspiration (in mm).
- `calibVar` A character string indicating the calibration variable to be used. It can be one of the following: `SWC`, `Eplanttot` or `Cohorts`.
- `obs` Measured calibration variable. Depending on the value of `calibVar` it can be a numeric vector with the measured SWC values (if `calibVar = "SWC"`), or a data frame with the first column containing the measured total transpiration (named `Eplanttot`) and the following columns containing the cohorts transpiration.
- `RZmin` The minimum value of RZ (the rooting depth) to be explored (in mm)
- `RZmax` The maximum value of RZ (the rooting depth) to be explored (in mm)
- `V1min` The minimum value of V1 (the root proportion in the first soil layer) to be explored
- `V1max` The maximum value of V1 (the root proportion in the first soil layer) to be explored
- `resolution` An integer defining the number of values to obtain by discretization of the root parameters RZ and V1. The number of parameter combinations and therefore the computation cost increases increase with the square of resolution
- `transformation` Function to modify the size of Z intervals to be explored (by default, bins are equal).
- `heat_stop` An integer defining the number of days during to discard from the calculation of the optimal root distribution. Usefull if the soil water content initialization is not certain
- `verbose` A logical value. Print the internal messages of the function?

Details

This function performs three different kinds of calibration, selecting those root distribution parameters that minimize the MAE between the predicted values and the measured values provided in `obs` argument. If `calibVar = "SWC"` different V1 values are tested running `spwb` maintaining the total soil depth provided in `x` and assuming that value is also the depth containing 95 percent of the roots. If `calibVar = "Eplanttot"` or `calibVar = 'Cohorts'` different combinations of RZ and V1 values are tested for each tree cohort and the root paramters are selected based on the MAE between the total transpiration or the cohort transpiration.

Value

THe function returns a data frame containing the species index used in medfate, calibrated values for Z50, Z95 and V1 and the MAE value for that combination.

Author(s)

Víctor Granda, CREAF

Antoine Cabon, CTFC-CREAF

Miquel De Cáceres Ainsa, CREAF

See Also

`spwb_ldrOptimization` for when no measured data is available, `spwb`, `soil`, `root_ldrDistribution`

---

spwb_ldrOptimization — *Optimization of root distribution*

Description

Functions `spwb_ldrExploration` and `spwb_ldrOptimization` are used to find optimum the species root distribution within `spwb`, given the arguments `x`, `meteo` and `psi_crit`.

Usage

```
spwb_ldrExploration(x, meteo, cohorts = NULL,
                     RZmin = 301, RZmax = 4000,
                    V1min = 0.01, V1max = 0.94, resolution = 10, heat_stop = 0,
                     transformation = "identity",
                     verbose = FALSE, ...)
spwb_ldrOptimization(y, psi_crit, opt_mode = 1)
```

Arguments

`x`	An object of class `spwbInput`.
`meteo`	A data frame with daily meteorological data series (see `spwb`).
`cohorts`	A character string with the names of cohorts to be explored. If `NULL` then all cohorts are explored.
`RZmin`	The minimum value of RZ (the rooting depth) to be explored (in mm)
`RZmax`	The maximum value of RZ (the rooting depth) to be explored (in mm)
`V1min`	The minimum value of V1 (the root proportion in the first soil layer) to be explored
`V1max`	The maximum value of V1 (the root proportion in the first soil layer) to be explored

resolution — An integer defining the number of values to obtain by discretization of the root parameters RZ and V1. The number of parameter combinations and therefore the computation cost increases increase with the square of resolution

transformation — Function to modify the size of Z intervals to be explored (by default, bins are equal).

heat_stop — An integer defining the number of days during to discard from the calculation of the optimal root distribution. Usefull if the soil water content initialization is not certain

... — Additional parameters to function `spwb`.

y — The result of calling `spwb_ldrExploration`.

psi_crit — A numerical vector of length iqual to the number of species in the plot containing the species values of water potential inducing hydraulic failure (in MPa). Use `NA` values to skip optimization for particular plant cohorts.

opt_mode — Optimization mode:

- `opt_mode = 1` maximizes transpiration along the line of stress equal to `psi_crit` (Cabon et al. 2018). The optimization is based on the eco-hydrological equilibrium hypothesis (Eagleson, 1982), which is formulated here as the root distribution for which plant transpiration is maximized while the plant water potential is close to the species-defined critical value `psi_crit` (Cabon et al.,2018).
- `opt_mode = 2` maximizes transpiration among combinations with stress according to `psi_crit`).
- `opt_mode = 3` maximizes photosynthesis among combinations with stress according to `psi_crit`).
- `opt_mode = 4` maximizes transpiration, subject to root construction constrains, among combinations with stress according to `psi_crit`).
- `opt_mode = 5` maximizes photosynthesis, subject to root construction constrains, among combinations with stress according to `psi_crit`).

verbose — A logical value. Print the internal messages of the function?

Details

For each combination of the parameters RZ and V1 the function `spwb_ldrExploration` runs `spwb`, setting the total soil depth equal to RZ. The root proportion in each soil layer is derived from V1, the depth of the first soil layer and RZ using the LDR root distribution model (Schenk and Jackson, 2002) and assuming that the depth containing 95 percent of the roots is equal to RZ. Function `spwb_ldrOptimization` takes the result of the exploration and tries to find optimum root distribution parameters. `psi_crit`, the species specific water potential inducing hydraulic failure, can be approached by the water potential inducing 50 percent of loss of conductance for the and gymnosperms and 88 percent for the angiosperms (Urli et al., 2013, Brodribb et al., 2010). Details of the hypothesis and limitations of the optimization method are given in Cabon et al. (2019).

Value

Function `spwb_ldrExploration` returns a list containing a list containing the explored RZ and V1 combinations as well as arrays with the values of average daily plant transpiration, average daily net photosynthesis and the minimum plant water potential for each cohort and parameter combination.

Function `spwb_ldrOptimization` returns a data frame with containing the species index used in medfate, `psi_crit` and the optimized values of V1 and the LDR parameters Z50 and Z95 (see `root_ldrDistribution`) and as many rows as the number of species.

Author(s)

Antoine Cabon, CREAF

Miquel De Cáceres Ainsa, CREAF

References

Brodribb, T.J., Bowman, D.J.M.S., Nichols, S., Delzon, S., Burlett, R., 2010. Xylem function and growth rate interact to determine recovery rates after exposure to extreme water deficit. New Phytol. 188, 533–542. doi:10.1111/j.1469-8137.2010.03393.x

Cabon, A., Martínez-Vilalta, J., Poyatos, R., Martínez de Aragón, J., De Cáceres, M. (2018) Applying the eco-hydrological equilibrium hypothesis to estimate root ditribution in water-limited forests. Ecohydrology 11: e2015.

Eagleson, P.S., 1982. Ecological optimality in water-limited natural soil-vegetation systems: 1. Theory and hypothesis. Water Resour. Res. 18, 325–340. doi:10.1029/WR018i002p00325

Schenk, H.J., Jackson, R.B., 2002. The Global Biogeography of Roots. Ecol. Monogr. 72, 311. doi:10.2307/3100092

Urli, M., Porte, A.J., Cochard, H., Guengant, Y., Burlett, R., Delzon, S., 2013. Xylem embolism threshold for catastrophic hydraulic failure in angiosperm trees. Tree Physiol. 33, 672–683. doi:10.1093/treephys/tpt030

See Also

`spwb`, `soil`, `root_ldrDistribution`

Examples

```
## Not run:
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params
examplesoil = soil(defaultSoilParams(2))

#Initialize control parameters
control = defaultControl("Granier")

#Initialize input
x = forest2spwbInput(exampleforestMED,examplesoil, SpParamsMED, control)
```

```
#Run exploration
y = spwb_ldrExploration(x = x, meteo = examplemeteo,
                        elevation = 100, latitude = 41.82592)

#Optimization under different modes
spwb_ldrOptimization(y = y, psi_crit = c(-2,-3,-4), opt_mode = 1)
spwb_ldrOptimization(y = y, psi_crit = c(-2,-3,-4), opt_mode = 2)
spwb_ldrOptimization(y = y, psi_crit = c(-2,-3,-4), opt_mode = 3)

## End(Not run)
```

`spwb_resistances` *Soil-plant resistances*

Description

Calculates and draws rhizosphere, root, stem and leaf resistances for simulation time steps

Usage

```
spwb_resistances(x, cohort = 1, relative = FALSE, draw = FALSE,
                 cumulative = FALSE, xlab = NULL, ylab = NULL)
```

Arguments

`x`	An object of class `spwb`. The function only works with the result of simulations with `transpirationMode = "Sperry"`.
`cohort`	An integer index indicating the cohort for which resistances are desired (by default the first cohort).
`relative`	A boolean flag to indicate that relative percentages are desired as output
`draw`	A boolean flag to indicate that a plot should be drawn.
`cumulative`	A flag to indicate that drawn series should be cumulative.
`xlab`	x-axis label.
`ylab`	y-axis label.

Details

The function makes internal calls to `hydraulics_soilPlantResistances`.

Value

A data frame with dates in rows and resistance segments in columns (Rhizosphere, Root, Stem and Leaf). Values depend on whether `relative = TRUE` (percentages) or `relative = FALSE` (absolute resistance values). If `draw = TRUE` then a plot object is returned.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwb, plot.spwb

---

spwb_sensitivity *Sensitivity analysis for soil plant water balance simulations*

Description

Performs a set of calls to spwb with the aim to determine the sensitivity to particular parameters.

Usage

```
spwb_sensitivity(x, soil, meteo,
                 paramType = "above", paramName = "LAI_live", cohort = 1,
                 p_change = c(-80,-40,-20,0,20,40,80),
                 summary.fun = NULL, simplify=TRUE,...)
```

Arguments

x	An object of class spwbInput.
soil	A list containing the description of the soil (see soil).
meteo	A data frame with daily meteorological data series (see spwb).
paramType	Data frame of x to modify.
paramName	Name of the parameter to modify.
cohort	Integer with the cohort to modify.
p_change	Numerical vector with percentages of change.
summary.fun	Summary function to be applied to the results of each simulation.
simplify	Whether the result of summary.fun should be simplified (see sapply).
...	Additional parameters to function spwb.

Details

Due to parameter dependence, modifying some parameters affects others:

- Setting paramName = "Z50/Z95" affects belowLayers$V, belowLayers$VCroot_kmax and belowLayers$VGrhizo_kmax.
- Modifying LAI_live also affects LAI_expanded.
- Modifying VCroot_kmax from paramsTranspiration affects both VCroot_kmax and belowLayers$VCroot_kmax.
- Modifying WaterStorage affects simultaneously Vleaf and Vsapwood from paramsWaterStorage.

- Modifying c from paramsTranspiration affects simultaneously VCleaf_c, VCstem_c and VCroot_c.
- Modifying d from paramsTranspiration affects simultaneously VCleaf_d, VCstem_d and VCroot_d.
- Modifying Plant_kmax from paramsTranspiration affects VCleaf_kmax, VCstem_kmax, VCroot_kmax and belowLayers$VCroot_kmax.
- Modifying Al2As from paramsAnatomy affects Vsapwood in paramsWaterStorage, VCstem_kmax and VCroot_kmax of paramsTranspiration and belowLayers$VCroot_kmax.
- Setting paramName = "Vmax298/Jmax298" affects both Vmax298 and Jmax298 from paramsTranspiration.

Value

If summary.fun = NULL the function returns a list whose elements are the result of calling spwb. Otherwise, the function applies summary.fun to each simulation result and returns these summaries (actually, a call to sapply is done).

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

spwb, summary.spwb

Examples

```
## Not run:
#Load example data and species parameters
data(examplemeteo)
data(exampleforestMED)
data(SpParamsMED)

#Initialize input
examplesoil = soil(defaultSoilParams(2))
control = defaultControl("Granier")
x = forest2spwbInput(exampleforestMED,examplesoil, SpParamsMED, control)

#Perform sensitivity analysis
res = spwb_sensitivity(x, examplesoil, examplemeteo, latitude = 41, elevation = 100)

## End(Not run)
```

spwb_stress *Drought stress indices*

Description

Allows calculating annual-based or monthly-based drought stress indices from spwb objects.

Usage

```
spwb_stress(x, index = "NDD", freq = "years", bySpecies = FALSE, draw = TRUE)
```

Arguments

x	An object of class spwb.
index	A string with the index to be calculated, either "DI", "NDD", "ADS", "MDS" or "WSI" (see details).
freq	Frequency of stress statistics (see cut.Date). Normally, either "years" or "months" for yearly-based or monthly-based indices.
bySpecies	Allows aggregating output by species.
draw	A boolean flag to indicate that a plot should be returned.

Details

The currently available drought stress indices are:

- "ADS": Average of daily drought stress values for the period considered.
- "MDS": Maximum daily drought stress during the period considered.
- "DI": Drought intensity, as defined in De Cáceres et al. (2015).
- "NDD": Number of drought days, as defined in De Cáceres et al. (2015).
- "WSI": Water stress integral, as defined in Myers (1988).

Value

A data frame with periods (e.g., years or months) in rows and plant cohorts (or species) in columns. Values are the calculated stress index. If draw=TRUE a ggplot is returned instead.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

De Cáceres M, Martínez-Vilalta J, Coll L, Llorens P, Casals P, Poyatos R, Pausas JG, Brotons L. (2015) Coupling a water balance model with forest inventory data to predict drought stress: the role of forest structural changes vs. climate changes. Agricultural and Forest Meteorology 213: 77-90 (doi:10.1016/j.agrformet.2015.06.012).

Myers BJ (1988) Water stress integral - a link between short-term stress and long-term growth. Tree Physiology 4: 315–323 (doi: 10.1093/treephys/4.4.315)

See Also

spwb, summary.spwb

---

spwb_waterUseEfficiency
Water use efficiency

Description

Calculates water use efficiency (WUE), at different temporal scales, from soil plant water balance calculations

Usage

```
spwb_waterUseEfficiency(x, type = "Plant Ag/E", leaves = "average", freq = "days",
                        draw = TRUE, ylim = NULL)
```

Arguments

`x`	An object of class `spwb`, `pwb` or `growth`.
`type`	A string to indicate the scale of WUE calculation. Either: • `"Leaf iWUE"`: Leaf intrinsic WUE, i.e. instantaneous ratio between photosynthesis and stomatal conductance (only for simulations with `transpirationMode = "Sperry"` and `subdailyResults = TRUE`). • `"Leaf Ci"`: Leaf intercellular CO2 concentration (only for simulations with `transpirationMode = "Sperry"` and `subdailyResults = TRUE`). • `"Plant An/E"`: Plant (cohort) net photosynthesis over plant transpiration (only for simulations with `transpirationMode = "Sperry"`) • `"Stand An/E"`: Stand net photosynthesis over stand transpiration (only for simulations with `transpirationMode = "Sperry"`) • `"Plant Ag/E"`: Plant (cohort) gross photosynthesis over plant transpiration • `"Stand Ag/E"`: Stand gross photosynthesis over stand transpiration
`leaves`	Either `"sunlit"`, `"shade"` or `"average"`. Refers to the WUE of different leaf types or the average (with weights according to the LAI of sunlit and shade leaves). Only relevant for `type = "iWUE"`.
`freq`	Frequency of summary statistics (see `cut.Date`).
`draw`	A boolean flag to indicate that a plot should be returned.
`ylim`	Range of values for y.

Details

Temporal aggregation of WUE values is done differently depending on the value of `type`. For `type = "Plant Ag/E"`, `type = "Stand Ag/E"`, `type = "Plant An/E"` and `type = "Stand An/E"` sums or daily photosynthesis and transpiration are first calculated at the desired temporal scale and the ratio is calculated afterwards. For `type = "Leaf iWUE"` intrinsic WUE values are first calculated at the daily scale (as averages of instantaneous An/gs ratios weighted by An) and then they are aggregated to the desired scale by calculating weighted averages, where weights are given by daily photosynthesis.

Value

If `draw=TRUE` a plot is returned. Otherwise, the function returns a matrix with WUE values, where rows are dates (at the desired temporal scale), and columns are plant cohorts. In the case of `type = "Plant Ag/E"`, `type = "Stand Ag/E"`, `type = "Plant An/E"` and `type = "Stand An/E"` values are in gC/L. In the case of `type = "Leaf iWUE"` values are in micromol of carbon per mmol of water.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

`spwb`

`Stand values` *Stand values*

Description

Functions to calculate stand attributes of a `forest` object.

Usage

```
stand_basalArea(x)
stand_foliarBiomass(x, SpParams, gdd = NA, mode = "MED")
stand_fuel(x, SpParams, gdd = NA, includeDead = TRUE, mode = "MED")
stand_LAI(x, SpParams, gdd = NA, mode = "MED")
stand_phytovolume(x, SpParams)
```

Arguments

`x`	An object of class `forest`.
`SpParams`	A data frame with species parameters (see `SpParamsMED`).
`gdd`	Growth degree days (to account for leaf phenology effects).
`includeDead`	A flag to indicate that standing dead fuels (dead branches) are included.
`mode`	Calculation mode, either "MED" or "US".

Value

- `stand_basalArea`: Stand basal area (m2/ha).
- `stand_foliarBiomass`: Standing biomass of leaves (in kg/m2).
- `stand_fuel`: Stand fine fuel load (in kg/m2).
- `stand_LAI`: Stand leaf area index (m2/m2).
- `stand_phytovolume`: Stand shrub phytovolume (m3/m2).

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

`forest`, `plant_basalArea`, `summary.forest`

Examples

```
#Default species parameterization
data(SpParamsMED)

#Load example plot
data(exampleforestMED)

#A short way to obtain total basal area
stand_basalArea(exampleforestMED)
```

---

summary.spwb *Summarises simulation results*

Description

Function summary summarizes the model's output in different temporal steps (i.e. weekly, annual, ...).

Usage

```
## S3 method for class 'pwb'
summary(object, freq="years", output="WaterBalance", FUN=sum, bySpecies = FALSE, ...)
## S3 method for class 'spwb'
summary(object, freq="years", output="WaterBalance", FUN=sum, bySpecies = FALSE, ...)
## S3 method for class 'growth'
summary(object, freq="years", output="WaterBalance", FUN=sum, bySpecies = FALSE, ...)
```

Arguments

- `object` — An object of class `spwb`, `pwb`, `growth` or `fordyn`.
- `freq` — Frequency of summary statistics (see `cut.Date`).
- `output` — The data table to be summarized. Accepted values are `"DailyBalance"`, `"PlantStress"`, `"PlantPsi"`, `"PlantTranspiration"`, `"PlantPhotosynthesis"` and `"SoilWaterBalance"`, `"Temperature"` and `"EnergyBalance"`.
- `FUN` — The function to summarize results (e.g., `sum`, `mean`, ...)
- `bySpecies` — Allows aggregating output by species before calculating summaries (only has an effect with some values of `output`). Aggregation can involve a sum (as for plant lai or transpiration) or a LAI-weighted mean (as for plant stress or plant water potential).
- `...` — Additional parameters for function `summary`.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

`spwb`

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params (2 layers)
examplesoil = soil(defaultSoilParams(2))

#Initialize control parameters
control = defaultControl("Granier")

#Initialize input
x = forest2spwbInput(exampleforestMED,examplesoil, SpParamsMED, control)

#Call simulation function
S1<-spwb(x, examplemeteo, latitude = 41.82592, elevation = 100)

#Monthly summary (averages) of soil status
summary(S1, freq="months",FUN=mean, output="Soil")
```

supplyfunctions *Hydraulic supply functions*

Description

Set of functions used in the implementation of hydraulic supply functions (Sperry \& Love 2015).

Usage

```
hydraulics_EXylem(psiPlant, psiUpstream,
                  kxylemmax, c, d, allowNegativeFlux = TRUE,
                  psiCav = 0)
hydraulics_EVanGenuchten(psiRhizo, psiSoil, krhizomax,
                      n, alpha, l = 0.5)
hydraulics_ECapacitance(psi, psiPrev, PLCprev,
                        V, fapo, c, d,
                        pi0, eps,
                        timestep)
hydraulics_ECrit(psiUpstream, kxylemmax, c, d, pCrit = 0.001)
hydraulics_E2psiXylem(E, psiUpstream,
                      kxylemmax, c, d, psiCav = 0)
hydraulics_E2psiXylemUp(E, psiDownstream,
                      kxylemmax, c, d, psiCav = 0)
hydraulics_E2psiVanGenuchten(E, psiSoil, krhizomax, n, alpha,
                      psiStep = -0.0001, psiMax = -10.0)
hydraulics_E2psiTwoElements(E, psiSoil, krhizomax, kxylemmax, n, alpha, c, d,
                      psiCav = 0, psiStep = -1e-04, psiMax = -10.0)
hydraulics_E2psiFineRootLeaf(E, psiFineRoot,
                      krootmax, rootc, rootd,
                      kstemmax, stemc, stemd,
                      kleafmax, leafc, leafd,
                      PLCstem)
hydraulics_E2psiBelowground(E, psiSoil,
                      krhizomax, nsoil, alphasoil,
                      krootmax, rootc, rootd,
                      psiIni = as.numeric(c(0)),
                      ntrial = 10, psiTol = 0.0001, ETol = 0.0001)
hydraulics_E2psiAboveground(E, psiRootCrown,
                      kstemmax, stemc, stemd,
                      kleafmax, leafc, leafd,
                      PLCstem)
hydraulics_E2psiNetwork(E, psiSoil,
                  krhizomax, nsoil, alphasoil,
                  krootmax, rootc, rootd,
                  kstemmax, stemc, stemd,
                  kleafmax, leafc, leafd,
                  PLCstem,
```

```
                  psiIni = as.numeric(c(0)),
                  ntrial = 10,
                  psiTol = 0.0001, ETol = 0.0001)
hydraulics_E2psiNetworkStem1(E, psiSoil,
                  krhizomax, nsoil, alphasoil,
                  krootmax, rootc, rootd,
                  kstemmax, stemc, stemd,
                  PLCstem,
                  psiIni = as.numeric(c(0)),
                  ntrial = 10,
                  psiTol = 0.0001, ETol = 0.0001)
hydraulics_supplyFunctionOneXylem(psiSoil, v,
                  kstemmax, stemc, stemd, psiCav = 0,
                  maxNsteps=200, dE=0.01)
hydraulics_supplyFunctionTwoElements(Emax, psiSoil,
                  krhizomax, kxylemmax, n, alpha, c, d,
                  psiCav = 0, dE = 0.1, psiMax = -10.0)
hydraulics_supplyFunctionThreeElements(Emax, psiSoil,
                  krhizomax, kxylemmax, kleafmax,
                  n, alpha, stemc, stemd, leafc, leafd,
                  psiCav = 0, dE = 0.1, psiMax = -10.0)


hydraulics_supplyFunctionBelowground(psiSoil,
                  krhizomax, nsoil, alphasoil,
                  krootmax, rootc, rootd,
                  minFlow = 0.0, maxNsteps=400,
                  ntrial = 10, psiTol = 0.0001, ETol = 0.0001, pCrit = 0.001)

hydraulics_supplyFunctionAboveground(Erootcrown, psiRootCrown,
                  kstemmax, stemc, stemd,
                  kleafmax, leafc, leafd,
                  PLCstem)

hydraulics_supplyFunctionFineRootLeaf(psiFineRoot,
                    krootmax, rootc, rootd,
                    kstemmax, stemc, stemd,
                    kleafmax, leafc, leafd,
                    PLCstem,
                    minFlow = 0.0, maxNsteps=400,
                    ETol = 0.0001, pCrit = 0.001)
hydraulics_supplyFunctionNetworkStem1(psiSoil,
                    krhizomax, nsoil, alphasoil,
                    krootmax, rootc, rootd,
                    kstemmax, stemc, stemd,
                    PLCstem,
                    minFlow = 0.0, maxNsteps=400,
                   ntrial = 200, psiTol = 0.0001, ETol = 0.0001, pCrit = 0.001)
```

```
hydraulics_supplyFunctionNetwork(psiSoil,
                  krhizomax, nsoil, alphasoil,
                  krootmax, rootc, rootd,
                  kstemmax, stemc, stemd,
                  kleafmax, leafc, leafd,
                  PLCstem, minFlow = 0.0, maxNsteps=400,
                  ntrial = 200, psiTol = 0.0001, ETol = 0.0001, pCrit = 0.001)

hydraulics_supplyFunctionPlot(x, soil, draw = TRUE, type = "E",
                              speciesNames = FALSE, ylim=NULL)
hydraulics_regulatedPsiXylem(E, psiUpstream, kxylemmax, c, d, psiStep = -0.01)
hydraulics_regulatedPsiTwoElements(Emax, psiSoil, krhizomax, kxylemmax, n, alpha,
                                   c, d, dE = 0.1, psiMax = -10.0)
```

Arguments

`v`	Proportion of fine roots within each soil layer.
`krhizomax`	Maximum rhizosphere hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
`kxylemmax`	Maximum xylem hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
`kleafmax`	Maximum leaf hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
`kstemmax`	Maximum stem xylem hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
`krootmax`	Maximum root xylem hydraulic conductance (defined as flow per leaf surface unit and per pressure drop).
`E`	Flow per surface unit.
`Emax`	Maximum flow per surface unit.
`Erootcrown`	Flow per surface unit at the root crown.
`psi`	Water potential (in MPa).
`psiPrev`	Water potential (in MPa) in the previous time step.
`psiDownstream`	Water potential upstream (in MPa).
`psiUpstream`	Water potential upstream (in MPa). In a one-component model corresponds to soil potential. In a two-component model corresponds to the potential inside the roots.
`psiCav`	Minimum water potential (in MPa) experienced (for irreversible cavitation).
`minFlow`	Minimum flow in supply function.
`psiPlant`	Plant water potential (in MPa).
`psiFineRoot`	Water potential (in MPa) inside fine roots.
`psiSoil`	Soil water potential (in MPa). A scalar or a vector depending on the function.
`psiRhizo`	Soil water potential (in MPa) in the rhizosphere (root surface).
`psiRootCrown`	Soil water potential (in MPa) at the root crown.

psiStep	Water potential precision (in MPa).
psiTol	Precision for water potential estimates (in MPa).
psiIni	Vector of initial water potential values (in MPa).
psiMax	Minimum (maximum in absolute value) water potential to be considered (in MPa).
pCrit	Critical water potential (in MPa).
PLCstem	Proportion of loss conductance in the stem [0-1].
PLCprev	Previous proportion of loss conductance [0-1].
V	Capacity of the compartment per leaf area (in L/m2).
fapo	Apoplastic fraction (proportion) in the segment.
pi0	Full turgor osmotic potential (MPa).
eps	Bulk modulus of elasticity (MPa).
dE	Increment of flow per surface unit.
ETol	Precision for water flow per surface unit.
c, d	Parameters of the Weibull function (generic xylem vulnerability curve).
rootc, rootd	Parameters of the Weibull function for roots (root xylem vulnerability curve).
stemc, stemd	Parameters of the Weibull function for stems (stem xylem vulnerability curve).
leafc, leafd	Parameters of the Weibull function for leaves (leaf vulnerability curve).
n, alpha, l	Parameters of the Van Genuchten function (rhizosphere vulnerability curve).
nsoil, alphasoil	Parameter vectors of the Van Genuchten function (rhizosphere vulnerability curve) with one value for each soil layer.
allowNegativeFlux	A boolean to indicate wether negative flux (i.e. from plant to soil) is allowed.
maxNsteps	Maximum number of steps in the construction of supply functions.
ntrial	Maximum number of steps in Newton-Raphson optimization.
x	An object of class spwbInput.
soil	A list containing the description of the soil (see soil).
type	Plot type for hydraulics_supplyFunctionPlot, either "E", "ERhizo", "psiStem", "psiRoot", "psiRhizo" or "dEdP").
draw	A flag to indicate whether the supply function should be drawn or just returned.
speciesNames	A flag to indicate the use of species names instead of cohort names in plots.
ylim	Graphical parameter to override function defaults.
timestep	Time step in seconds.

Details

Function hydraulics_supplyFunctionPlot draws a plot of the supply function for the given soil object and network properties of each plant cohort in x. Function hydraulics_vulnerabilityCurvePlot draws a plot of the vulnerability curves for the given soil object and network properties of each plant cohort in x.

Value

Values returned for each function are:

- `hydraulics_E2psiXylem`: The plant (leaf) water potential (in MPa) corresponding to the input flow, according to the xylem supply function and given an upstream (soil or root) water potential.
- `hydraulics_E2psiVanGenuchten`: The root water potential (in MPa) corresponding to the input flow, according to the rhizosphere supply function and given a soil water potential.
- `hydraulics_E2psiTwoElements`: The plant (leaf) water potential (in MPa) corresponding to the input flow, according to the rhizosphere and plant supply functions and given an input soil water potential.
- `hydraulics_E2psiNetwork`: The rhizosphere, root crown and plant (leaf water potential (in MPa) corresponding to the input flow, according to the vulnerability curves of rhizosphere, root and stem elements in a network.
- `hydraulics_Ecrit`: The critical flow according to the xylem supply function and given an input soil water potential.
- `hydraulics_EVanGenuchten`: The flow (integral of the vulnerability curve) according to the rhizosphere supply function and given an input drop in water potential (soil and rhizosphere).
- `hydraulics_EXylem`: The flow (integral of the vulnerability curve) according to the xylem supply function and given an input drop in water potential (rhizosphere and plant).
- `hydraulics_supplyFunctionOneXylem`, `hydraulics_supplyFunctionTwoElements` and `hydraulics_supplyFunc` A list with different numeric vectors with information of the two-element supply function:
  - `E`: Flow values (supply values).
  - `FittedE`: Fitted flow values (for `hydraulics_supplyFunctionTwoElements`).
  - `Elayers`: Flow values across the roots of each soil layer (only for `hydraulics_supplyFunctionNetwork`).
  - `PsiRhizo`: Water potential values at the root surface (only for `hydraulics_supplyFunctionNetwork`).
  - `PsiRoot`: Water potential values inside the root crown (not for `hydraulics_supplyFunctionOneXylem`).
  - `PsiPlant`: Water potential values at the canopy (leaf).
  - `dEdP`: Derivatives of the supply function.
- `hydraulics_supplyFunctionPlot`: If `draw = FALSE` a list with the result of calling `hydraulics_supplyFunctionNet` for each cohort.
- `hydraulics_regulatedPsiXylem`: Plant water potential after regulation (one-element loss function) given an input water potential.
- `hydraulics_regulatedPsiTwoElements`: Plant water potential after regulation (two-element loss function) given an input soil water potential.

Author(s)

Miquel De Cáceres Ainsa, CREAF.

References

Sperry, J. S., F. R. Adler, G. S. Campbell, and J. P. Comstock. 1998. Limitation of plant water use by rhizosphere and xylem conductance: results from a model. Plant, Cell \& Environment 21:347–359.

Sperry, J. S., and D. M. Love. 2015. What plant hydraulics can tell us about responses to climate-change droughts. New Phytologist 207:14–27.

See Also

hydraulics_psi2K, hydraulics_maximumStemHydraulicConductance, spwb, soil

Examples

```
kstemmax = 4 # in mmol·m-2·s-1·MPa-1
stemc = 3
stemd = -4 # in MPa

psiVec = seq(-0.1, -7.0, by =-0.01)

#Vulnerability curve
kstem = unlist(lapply(psiVec, hydraulics_xylemConductance, kstemmax, stemc, stemd))
plot(-psiVec, kstem, type="l",ylab="Xylem conductance (mmol·m-2·s-1·MPa-1)",
xlab="Canopy pressure (-MPa)", lwd=1.5,ylim=c(0,kstemmax))
```

---

tissuemoisture *Tissue moisture functions*

Description

Set of functions used to calculate tissue moisture from water potential and viceversa.

Usage

```
moisture_symplasticRWC(psiSym, pi0, epsilon)
moisture_symplasticPsi(RWC, pi0, epsilon)
moisture_apoplasticRWC(psiApo, c, d)
moisture_apoplasticPsi(RWC, c, d)
moisture_turgorLossPoint(pi0, epsilon)
moisture_tissueRWC(psiSym, pi0, epsilon,
                   psiApo, c, d,
                   af, femb = 0)
moisture_cohortFMC(spwb, SpParams)
moisture_cohortFMCDay(spwb_day, x, SpParams)
moisture_pressureVolumeCurvePlot(x, segment="leaf",
                    fraction = "all",
                    psiVec =  seq(-0.1, -8.0, by =-0.01),
                    speciesNames = FALSE)
```

Arguments

psiSym, psiApo	Symplastic or apoplastic water potential (MPa).
RWC	Relative water content [0-1].
pi0	Full turgor osmotic potential (MPa).
epsilon	Bulk modulus of elasticity (MPa).
c,d	Parameters of the xylem vulnerability curve.
af	Apoplastic fraction (proportion) in the segment (e.g. leaf or stem).
femb	Fraction of embolized conduits.
x	An object of class `spwbInput`.
segment	Segment whose relative water content curve to plot, either `"stem"` or `"leaf"`.
fraction	Tissue fraction, either `"symplastic"`, `"apoplastic"` or `"all"`.
psiVec	Vector of water potential values to evaluate for the pressure-volume curve.
speciesNames	A flag to indicate the use of species names instead of cohort names in plots.
spwb	Object returned by function `spwb` or `growth`.
spwb_day	Object returned by function `spwb_day` or `growth_day`.
SpParams	A data frame with species parameters (see `SpParamsMED` and `SpParamsMED`).

Details

Details of the tissue moisture calculations are given in a vignette. Function `moisture_cohortFMC()` calculates the fuel moisture content of leaves of each cohort, from the results of soil water balance.

Value

Values returned for each function are:

- `moisture_symplasticRWC`: Relative water content [0-1] of the symplastic fraction.
- `moisture_apoplasticRWC`: Relative water content [0-1] of the apoplastic fraction.
- `moisture_symplasticWaterPotential`: Water potential (in MPa) of the symplastic fraction.
- `moisture_apoplasticWaterPotential`: Water potential (in MPa) of the apoplastic fraction.
- `moisture_turgorLossPoint`: Water potential (in MPa) corresponding to turgor loss point.
- `moisture_segmentRWC`: Segment relative water content [0-1].
- `moisture_cohortFMC`: A matrix with daily moisture content per dry weight (in percent) of each cohort.
- `moisture_cohortFMCDay`: A vector with moisture content per dry weight (in percent) of each cohort.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Bartlett, M.K., Scoffoni, C., Sack, L. 2012. The determinants of leaf turgor loss point and prediction of drought tolerance of species and biomes: a global meta-analysis. Ecology Letters 15: 393–405.

Hölttä, T., Cochard, H., Nikinmaa, E., Mencuccini, M. 2009. Capacitive effect of cavitation in xylem conduits: Results from a dynamic model. Plant, Cell and Environment 32: 10–21.

Martin-StPaul, N., Delzon, S., Cochard, H. 2017. Plant resistance to drought depends on timely stomatal closure. Ecology Letters 20: 1437–1447.

See Also

hydraulics_psi2K, hydraulics_supplyFunctionPlot, spwb, soil

Examples

```
psi = seq(-10,0, by=0.1)
rwc_s = rep(NA, length(psi))
for(i in 1:length(psi)) rwc_s[i] = moisture_symplasticRWC(psi[i],-3,12)
plot(psi, rwc_s, type="l", xlab="Water potential (MPa)", ylab = "Symplasmic RWC")
```

---

transp_maximumTranspirationModel

Maximum transpiration vs. LAI

Description

Builds a model of maximum transpiration (Tmax) over potential evapotranspiration (PET) for increasing leaf area index (LAI) values for each plant cohort.

Usage

```
transp_maximumTranspirationModel(x, meteo, latitude, elevation, slope, aspect,
                                 LAI_seq = c(0.1, 0.25, seq(0.5, 10, by = 0.5)),
                                  draw = TRUE)
```

Arguments

x	An object of class spwbInput, built using the 'Sperry' transpiration mode.
meteo	A data frame with daily meteorological data series.
latitude	Latitude (in degrees).
elevation, slope, aspect	Elevation above sea level (in m), slope (in degrees) and aspect (in degrees from North).
LAI_seq	Sequence of stand LAI values to be tested.
draw	Boolean flag to indicate plotting of results.

Details

This function performs a meta-modelling exercise using the Sperry transpiration model, with the aim to estimate coefficients for the equation used in the Granier transpiration model (Granier et al. 1999). The model to be fitted is: `y ~ a*LAI + b*LAI^2`, where `y` is the ratio between maximum transpiration (Tmax) and Penman's potential evapotranspiration (PET) and `LAI` is the stand LAI. Unlike the original equation of Granier et al. (1999), we fit a zero intercept model so that LAI = 0 translates into zero plant transpiration.

The function fits the model for each cohort separately, assuming it represents the whole stand. For each stand LAI value in the input sequence, the function uses simulations with Sperry transpiration and the input weather to estimate `y = Tmax/PET` as a function of stand's LAI (deciduous stands include leaf phenology). Once simulations have been conducted for each stand LAI value, the function fits a Generalized Linear Model with the above equation, assuming a Gamma distribution of residuals and an identity link.

The coefficients of the model can be used to parametrize Granier's transpiration, since coefficients `a` and `b` in the equation above correspond to parameters `Tmax_LAI` and `Tmax_LAIsq`, respectively (see `SpParamsMED`).

Value

Returns a list with as many elements as plant cohorts, each element being a `glm` model.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Granier A, Bréda N, Biron P, Villette S (1999) A lumped water balance model to evaluate duration and intensity of drought constraints in forest stands. Ecol Modell 116:269–283. https://doi.org/10.1016/S0304-3800(98)00205-1.

See Also

`spwb`, `transp_transpirationGranier`, `transp_transpirationSperry`, `SpParamsMED`

Examples

```
## Not run:
#Load example daily meteorological data
data(examplemeteo)

# Load example plot plant data
data(exampleforestMED)

# Load default species parameters
data(SpParamsMED)

# Initialize soil with default soil params
```

```
examplesoil2 = soil(defaultSoilParams(2))

# Initialize control parameters for 'Sperry' transpiration mode
control = defaultControl(transpirationMode="Sperry")

# Initialize input
x2 = forest2spwbInput(exampleforestMED,examplesoil2, SpParamsMED, control)

# Estimate maximum transpiration ratio models for each cohort
m = transp_maximumTranspirationModel(x2, examplemeteo,
                                     41.82592, elevation = 100,
                                     slope = 0, aspect = 0)

# Inspect the model for first cohort
m[[1]]

## End(Not run)
```

`transp_modes` *Transpiration modes*

Description

High-level sub-models to represent transpiration, plant hydraulics and water relations within plants. The two submodels represent a very different degree of complexity, and correspond to Granier et al. (1999) or Sperry et al. (2017).

Usage

```
transp_transpirationGranier(x, meteo, day,
                              modifyInput = TRUE)
transp_transpirationSperry(x, meteo, day,
                             latitude, elevation, slope, aspect,
                         canopyEvaporation = 0, snowMelt = 0, soilEvaporation = 0,
                             stepFunctions = NA,
                             modifyInput = TRUE)
```

Arguments

`x` An object of class `spwbInput` or `growthInput`, built using the 'Granier' or 'Sperry' transpiration modes, depending on the function to be called.

`meteo` A data frame with daily meteorological data series:

- `DOY`: Day of the year (Julian day).
- `Precipitation`: Precipitation (in mm).
- `MeanTemperature`: Mean temperature (in degrees Celsius).
- `MinTemperature`: Minimum temperature (in degrees Celsius).
- `MaxTemperature`: Maximum temperature (in degrees Celsius).

- MinRelativeHumidity: Minimum relative humidity (in percent).
- MaxRelativeHumidity: Maximum relative humidity (in percent).
- Radiation: Solar radiation (in MJ/m2/day).
- WindSpeed: Wind speed (in m/s). If not available, this column can be left with NA values.

day An integer to identify a day within meteo.

latitude Latitude (in degrees).

elevation, slope, aspect
Elevation above sea level (in m), slope (in degrees) and aspect (in degrees from North).

canopyEvaporation
Canopy evaporation (from interception) for day (mm).

soilEvaporation
Bare soil evaporation for day (mm).

snowMelt Snow melt values for day (mm).

stepFunctions An integer to indicate a simulation step for which photosynthesis and profit maximization functions are desired.

modifyInput Boolean flag to indicate that the input x object is allowed to be modified during the simulation.

Value

Function transp_transpirationGranier and transp_transpirationSperry return a list with the following elements:

- "cohorts": A data frame with cohort information, copied from spwbInput.
- "Stand": A vector of stand-level variables.
- "Plants": A data frame of results for each plant cohort. When using transp_transpirationGranier, element "Plants" includes:
  - "LAI": Leaf area index of the plant cohort.
  - "AbsorbedSWRFraction": Fraction of SWR absorbed by each cohort.
  - "Transpiration": Transpirated water (in mm) corresponding to each cohort.
  - "GrossPhotosynthesis": Gross photosynthesis (in gC/m2) corresponding to each cohort.
  - "psi": Water potential (in MPa) of the plant cohort (average over soil layers).
  - "DDS": Daily drought stress [0-1] (relative whole-plant conductance).

  When using transp_transpirationSperry, element "Plants" includes:
  - "LAI": Leaf area index of the plant cohort.
  - "Extraction": Water extracted from the soil (in mm) for each cohort.
  - "Transpiration": Transpirated water (in mm) corresponding to each cohort.
  - "GrossPhotosynthesis": Gross photosynthesis (in gC/m2) corresponding to each cohort.
  - "NetPhotosynthesis": Net photosynthesis (in gC/m2) corresponding to each cohort.

- "RootPsi": Minimum water potential (in MPa) at the root collar.
- "StemPsi": Minimum water potential (in MPa) at the stem.
- "StemPLC": Proportion of conductance loss in stem.
- "LeafPsiMin": Minimum (predawn) water potential (in MPa) at the leaf (representing an average leaf).
- "LeafPsiMax": Maximum (midday) water potential (in MPa) at the leaf (representing an average leaf).
- "LeafPsiMin_SL": Minimum (predawn) water potential (in MPa) at sunlit leaves.
- "LeafPsiMax_SL": Maximum (midday) water potential (in MPa) at sunlit leaves.
- "LeafPsiMin_SH": Minimum (predawn) water potential (in MPa) at shade leaves.
- "LeafPsiMax_SH": Maximum (midday) water potential (in MPa) at shade leaves.
- "dEdP": Overall soil-plant conductance (derivative of the supply function).
- "DDS": Daily drought stress [0-1] (relative whole-plant conductance).
- "StemRWC": Relative water content of stem tissue (including symplasm and apoplasm).
- "LeafRWC": Relative water content of leaf tissue (including symplasm and apoplasm).
- "StemSympRWC": Relative water content of symplastic stem tissue.
- "LeafSympRWC": Relative water content of symplastic leaf tissue.
- "WaterBalance": Plant water balance (extraction - transpiration).

• "Extraction": A data frame with mm of water extracted from each soil layer (in columns) by each cohort (in rows).

The remaining items are only given by `transp_transpirationSperry`:

• "EnergyBalance": When using the 'Sperry' transpiration mode, the model performs energy balance of the stand and 'EnergyBalance' is a list with the following:
  - "Temperature": A data frame with the temperature of the atmosphere ('Tatm'), canopy ('Tcan') and soil ('Tsoil.1', 'Tsoil.2', ...) for each time step.
  - "CanopyEnergyBalance": A data frame with the components of the canopy energy balance (in W/m2) for each time step.
  - "SoilEnergyBalance": A data frame with the components of the soil energy balance (in W/m2) for each time step.

• "RhizoPsi": Minimum water potential (in MPa) inside roots, after crossing rhizosphere, per cohort and soil layer.

• "Sunlitleaves" and "ShadeLeaves": Data frames for sunlit leaves and shade leaves and the following columns per cohort:
  - "LAI": Cumulative leaf area index of sunlit/shade leaves.
  - "Vmax298": Average maximum carboxilation rate for sunlit/shade leaves.
  - "Jmax298": Average maximum electron transport rate for sunlit/shade leaves.

• "ExtractionInst": Water extracted by each plant cohort during each time step.

• "PlantsInst": A list with instantaneous (per time step) results for each plant cohort:
  - "E": A data frame with the cumulative transpiration (mm) for each plant cohort during each time step.
  - "Ag": A data frame with the cumulative gross photosynthesis (gC/m2) for each plant cohort during each time step.

- "An": A data frame with the cumulative net photosynthesis (gC/m2) for each plant cohort during each time step.
- "Sunlitleaves" and "ShadeLeaves": Lists with instantaneous (for each time step) results for sunlit leaves and shade leaves and the following items:
  * "Abs_SWR": A data frame with instantaneous absorbed short-wave radiation (SWR).
  * "Net_LWR": A data frame with instantaneous net long-wave radiation (LWR).
  * "An": A data frame with instantaneous net photosynthesis (in micromol/m2/s).
  * "Ci": A data frame with instantaneous intercellular CO2 concentration (in ppm).
  * "GW": A data frame with instantaneous stomatal conductance (in mol/m2/s).
  * "VPD": A data frame with instantaneous vapour pressure deficit (in kPa).
  * "Temp": A data frame with leaf temperature (in degrees Celsius).
  * "Psi": A data frame with leaf water potential (in MPa).
- "dEdP": A data frame with the slope of the plant supply function (an estimation of whole-plant conductance).
- "RootPsi": A data frame with root crown water potential (in MPa) for each plant cohort during each time step.
- "StemPsi": A data frame with stem water potential (in MPa) for each plant cohort during each time step.
- "LeafPsi": A data frame with leaf (average) water potential (in MPa) for each plant cohort during each time step.
- "StemPLC": A data frame with the proportion loss of conductance [0-1] for each plant cohort during each time step.
- "StemRWC": A data frame with the (average) relative water content of stem tissue [0-1] for each plant cohort during each time step.
- "LeafRWC": A data frame with the relative water content of leaf tissue [0-1] for each plant cohort during each time step.
- "StemSympRWC": A data frame with the (average) relative water content of symplastic stem tissue [0-1] for each plant cohort during each time step.
- "LeafSympRWC": A data frame with the relative water content of symplastic leaf tissue [0-1] for each plant cohort during each time step.
- "PWB": A data frame with plant water balance (extraction - transpiration).

• "LightExtinction": A list of information regarding radiation balance through the canopy, as returned by function light_instantaneousLightExtinctionAbsorption.
• "CanopyTurbulence": Canopy turbulence (see wind_canopyTurbulence).
• "SupplyFunctions": If stepFunctions is not missing, a list of supply functions, photosynthesis functions and profit maximization functions.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Granier A, Bréda N, Biron P, Villette S (1999) A lumped water balance model to evaluate duration and intensity of drought constraints in forest stands. Ecol Modell 116:269–283. https://doi.org/10.1016/S0304-3800(98)00205-1.

Sperry, J. S., M. D. Venturas, W. R. L. Anderegg, M. Mencuccini, D. S. Mackay, Y. Wang, and D. M. Love. 2017. Predicting stomatal responses to the environment from the optimization of photosynthetic gain and hydraulic cost. Plant Cell and Environment 40, 816-830 (doi: 10.1111/pce.12852).

See Also

spwb_day, plot.spwb_day

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)

#Initialize soil with default soil params (2 layers)
examplesoil = soil(defaultSoilParams(2))

#Initialize control parameters
control = defaultControl("Granier")

#Initialize input
x1 = forest2spwbInput(exampleforestMED,examplesoil, SpParamsMED, control)

# Transpiration according to Granier's model, plant water potential
# and plant stress for a given day
t1 = transp_transpirationGranier(x1, examplemeteo, 1,
                            modifyInput = FALSE)

#Switch to 'Sperry' transpiration mode
control = defaultControl("Sperry")

#Initialize soil with default soil params (2 layers)
examplesoil2 = soil(defaultSoilParams(2))

#Initialize input
x2 = forest2spwbInput(exampleforestMED,examplesoil2, SpParamsMED, control)

# Transpiration according to Sperry's model
t2 = transp_transpirationSperry(x2, examplemeteo, 1,
                           latitude = 41.82592, elevation = 100, slope = 0, aspect = 0,
                           modifyInput = FALSE)
```

---

transp_stomatalregulation

Stomatal regulation

Description

Set of high-level functions used in the calculation of stomatal conductance and transpiration. Function `transp_profitMaximization` calculates gain and cost functions, as well as profit maximization from supply and photosynthesis input functions. Function `transp_stomatalRegulationPlot` produces a plot with the cohort supply functions against water potential and a plot with the cohort photosynthesis functions against water potential, both with the maximum profit values indicated.

Usage

```
transp_profitMaximization(supplyFunction, photosynthesisFunction, Gswmin, Gswmax,
                        gainModifier = 1.0, costModifier = 1.0, costWater = "dEdP")
transp_stomatalRegulationPlot(x, meteo, day, timestep,
                              latitude, elevation, slope = NA, aspect = NA,
                              type="E")
```

Arguments

`supplyFunction` Water supply function (see `hydraulics_supplyFunctionNetwork`).

`photosynthesisFunction` Function returned by `photo_photosynthesisFunction()`.

`Gswmin, Gswmax` Minimum and maximum stomatal conductance to water vapour (mol·m-2·s-1).

`gainModifier, costModifier` Modifiers (exponents) of the gain and cost functions defined in Sperry et al. (2017).

`costWater` Cost water function, either "dEdP" (default) or "kterm" (experimental).

`x` An object of class `spwbInput` built using the 'Sperry' transpiration mode.

`meteo` A data frame with daily meteorological data series:

- `DOY`: Day of the year (Julian day).
- `Precipitation`: Precipitation (in mm).
- `MeanTemperature`: Mean temperature (in degrees Celsius).
- `MinTemperature`: Minimum temperature (in degrees Celsius).
- `MaxTemperature`: Maximum temperature (in degrees Celsius).
- `MinRelativeHumidity`: Minimum relative humidity (in percent).
- `MaxRelativeHumidity`: Maximum relative humidity (in percent).
- `Radiation`: Solar radiation (in MJ/m2/day).
- `WindSpeed`: Wind speed (in m/s). If not available, this column can be left with `NA` values.

`day` An integer to identify a day within `meteo`.

`timestep` An integer between 1 and `ndailysteps` specified in `x` (see `defaultControl`).

`latitude` Latitude (in degrees).

`elevation, slope, aspect` Elevation above sea level (in m), slope (in degrees) and aspect (in degrees from North).

`type` A string with plot type, either `"E"` (transpiration flow), `"Ag"` (gross photosynthesis), `"An"` (net photosynthesis), `"Gsw"` (stomatal conductance to water vapour), `"T"` (temperature) or `"VPD"` (leaf vapour pressure deficit).

Value

Function `transp_profitMaximization` returns a list with the following elements:

- `Cost`: Cost function [0-1].
- `Gain`: Gain function [0-1].
- `Profit`: Profit function [0-1].
- `iMaxProfit`: Index corresponding to maximum profit (starting from 0).

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Sperry, J. S., M. D. Venturas, W. R. L. Anderegg, M. Mencuccini, D. S. Mackay, Y. Wang, and D. M. Love. 2017. Predicting stomatal responses to the environment from the optimization of photosynthetic gain and hydraulic cost. Plant Cell and Environment 40, 816-830 (doi: 10.1111/pce.12852).

See Also

`transp_transpirationSperry`, `hydraulics_supplyFunctionNetwork`, `biophysics_leafTemperature`, `photo_photosynthesis`, `spwb_day`, `plot.spwb_day`

Examples

```
#Load example daily meteorological data
data(examplemeteo)

#Load example plot plant data
data(exampleforestMED)

#Default species parameterization
data(SpParamsMED)


#Initialize soil with default soil params (2 layers)
examplesoil = soil(defaultSoilParams(2))

#Initialize control parameters
control = defaultControl(transpirationMode="Sperry")

#Initialize soil with default soil params (2 layers)
examplesoil2 = soil(defaultSoilParams(2))

#Initialize input
x2 = forest2spwbInput(exampleforestMED,examplesoil2, SpParamsMED, control)

# Stomatal VPD curve and chosen value for the 12th time step at day 100
transp_stomatalRegulationPlot(x2, examplemeteo, day=100, timestep = 12,
                              latitude = 41.82592, elevation = 100, type="VPD")
```

Vertical profiles *Vertical profiles*

Description

Functions to generate vertical profiles generated by an input forest object.

Usage

```
vprofile_leafAreaDensity(x, SpParams = NULL, z = NULL, gdd = NA, mode = "MED",
                         byCohorts = FALSE, bySpecies = FALSE, draw = TRUE,
                         xlim = NULL)
vprofile_rootDistribution(x, SpParams, d = NULL,  bySpecies = FALSE, draw = TRUE,
                          xlim = NULL)
vprofile_fuelBulkDensity(x, SpParams, z = NULL, gdd = NA, mode = "MED",
                         draw = TRUE, xlim = NULL)
vprofile_PARExtinction(x, SpParams, z = NULL, gdd = NA, mode = "MED",
                       draw = TRUE, xlim = c(0,100))
vprofile_SWRExtinction(x, SpParams, z = NULL, gdd = NA, mode = "MED",
                       draw = TRUE, xlim = c(0,100))
vprofile_windExtinction(x, SpParams, u = 1, windMeasurementHeight = 200,
                        boundaryLayerSize = 2000, target = "windspeed",
                        z = NULL, gdd = NA, mode = "MED",
                        draw = TRUE, xlim = NULL)
```

Arguments

x	An object of class forest
SpParams	A data frame with species parameters (see SpParamsMED).
z	A numeric vector with height values.
d	A numeric vector with soil layer widths.
gdd	Growth degree days.
mode	Calculation mode, either "MED" or "US".
byCohorts	Separate profiles for each cohort.
bySpecies	Aggregate cohort profiles by species.
u	The value of measured wind speed (in m/s).
windMeasurementHeight	Height corresponding to wind measurement (in cm over the canopy).
boundaryLayerSize	Size of the boundary layer (in cm) over the canopy.
target	Wind property to draw, either "windspeed", "kineticenergy" (turbulent kinetic energy) or "stress" (Reynold's stress).
draw	Logical flag to indicate that a plot is desired.
xlim	Limits of the x-axis.

Value

A numeric vector with values measured at each height. Units depend on the profile function:

- `vprofile_leafAreaDensity`: Cumulative LAI (m2/m2) per height bin.
- `vprofile_fuelBulkDensity`: Fuel bulk density (kg/m3) per height bin.
- `vprofile_PARExtinction`: Percent of photosynthetically active radiation (%) corresponding to each height.
- `vprofile_SWRExtinction`: Percent of shortwave radiation (%) corresponding to each height.
- `vprofile_windExtinction`: Wind speed (m/s) corresponding to each height.

Author(s)

Miquel De Cáceres Ainsa, CREAF

See Also

`forest`, `wind_canopyTurbulence`

Examples

```
#Default species parameterization
data(SpParamsMED)

#Load example plot plant data
data(exampleforestMED)

vprofile_leafAreaDensity(exampleforestMED, SpParamsMED)
vprofile_fuelBulkDensity(exampleforestMED, SpParamsMED)

vprofile_PARExtinction(exampleforestMED, SpParamsMED)
vprofile_SWRExtinction(exampleforestMED, SpParamsMED)

vprofile_windExtinction(exampleforestMED, SpParamsMED)
```

`Wind models` *Models for canopy turbulence*

Description

Models for canopy turbulence by Katul et al (2004).

Usage

```
wind_canopyTurbulenceModel(zm, Cx, hm, d0, z0, model = "k-epsilon")
wind_canopyTurbulence(zmid, LAD, canopyHeight,
                      u, windMeasurementHeight = 200, model = "k-epsilon")
```

Arguments

`zm`	A numeric vector with height values (m).
`Cx`	Effective drag = Cd x leaf area density.
`hm`	Canopy height (m).
`d0`	Zero displacement height (m).
`z0`	Momentum roughness height (m).
`zmid`	A numeric vector of mid-point heights (in cm) for canopy layers.
`LAD`	A numeric vector of leaf area density values (m3/m2).
`canopyHeight`	Canopy height (in cm).
`u`	Measured wind speed (m/s).
`windMeasurementHeight`	Height of wind speed measurement with respect to canopy height (cm).
`model`	Closure model.

Details

Implementation in Rcpp of the K-epsilon canopy turbulence models by Katul et al (2004) originally in Matlab code (https://nicholas.duke.edu/people/faculty/katul/k_epsilon_model.htm).

Value

Function `wind_canopyTurbulenceModel` returns a data frame of vertical profiles for variables:

- `z1`: Height values.
- `U1`: U/u*, where U is mean velocity and u* is friction velocity.
- `dU1`: dUdz/u*, where dUdz is mean velocity gradient and u* is friction velocity.
- `epsilon1`: epsilon/(u^3/h) where epsilon is the turbulent kinetic dissipation rate, u* is friction velocity and h is canopy height.
- `k1`: k/(u*^2), where k is the turbulent kinetic energy and u* is friction velocity.
- `uw1`: <uw>/(u*^2), where <uw> is the Reynolds stress and u* is friction velocity.
- `Lmix1`: Mixing length.

Function `wind_canopyTurbulence` returns a data frame of vertical profiles for transformed variables:

- `zmid`: Input mid-point heights (in cm) for canopy layers.
- `u`: Wind speed (m/s).
- `du`: Mean velocity gradient (1/s).
- `epsilon`: Turbulent kinetic dissipation rate.
- `k`: Turbulent kinetic energy.
- `uw`: Reynolds stress.

Author(s)

Miquel De Cáceres Ainsa, CREAF

References

Katul GG, Mahrt L, Poggi D, Sanz C (2004) One- and two-equation models for canopy turbulence. Boundary-Layer Meteorol 113:81–109. https://doi.org/10.1023/B:BOUN.0000037333.48760.e5

See Also

vprofile_windExtinction

Examples

```
#Default species parameterization
data(SpParamsMED)

#Load example plot plant data
data(exampleforestMED)

#Canopy height (in m)
h= max(exampleforestMED$treeData$Height/100)
d0 = 0.67*h
z0 = 0.08*h

#Height values (cm)
z = seq(50,1000, by=50)
zm = z/100 # (in m)

# Leaf area density
lad = vprofile_leafAreaDensity(exampleforestMED, SpParamsMED, draw = FALSE,
                               z = c(0,z))

# Effective drag
Cd = 0.2
Cx = Cd*lad

# canopy turbulence model
wind_canopyTurbulenceModel(zm, Cx,h,d0,z0)
```

Wood formation *Wood formation*

Description

Functions to initialize and expand a ring of tracheids to simulate secondary growth.

Usage

```
woodformation_initRing()
woodformation_growRing(ring, psi, Tc,
                        Nc=8.85, phi0=0.13, pi0=-0.8, CRD0=8.3,
                        Y_P=0.05, Y_T=8.0, h=0.043*1.8, s=1.8)
woodformation_relativeExpansionRate(psi, Tc, pi, phi, Y_P, Y_T)
```

Arguments

`ring`	An object of class ring returned by function `woodformation_initRing`.
`psi`	Water potential (in MPa).
`Tc`	Temperature in Celsius.
`Nc`	Number of active cells in the cambium.
`phi0`	Initial value of cell extensibility (in MPa-1 day-1)
`pi0`	Initial value of cell osmotic potential (in MPa)
`CRD0`	Initial value of cell radial diameter
`Y_P`	Turgor pressure yield threshold (in MPa)
`Y_T`	Temperature yield threshold (in Celsius)
`h`	Cell wall hardening coefficient (in day-1)
`s`	Cell wall softening coefficient (unitless)
`pi`	Osmotic potential (in MPa)
`phi`	Cell extensibility (in MPa-1 day-1)

Value

Function `woodformation_initRing()` returns a list of class 'ring', that is a list containing a data frame `cells` and two vectors: `P` and `SA`. Dataframe `cells` contains the columns "formation_date", "phi", "pi" and "CRD" and as many rows as dates processed. Vectors `P` and `SA` contain, respectively, the number of cells produced and the sapwood area corresponding to the ring of cells (assuming a tangencial radius of 20 micrometers). Function `woodformation_growRing()` modifies the input 'ring' object according to the environmental conditions given as input. Function `woodformation_relativeExpansionRate()` returns a numeric scalar with the relative expansion rate.

Note

Code modified from package xylomod by Antoine Cabon, available at GitHub

Author(s)

Antoine Cabon, CTFC

Miquel De Cáceres Ainsa, CREAF

References

Cabon A, Fernández-de-Uña L, Gea-Izquierdo G, Meinzer FC, Woodruff DR, Martínez-Vilalta J, De Cáceres M. 2020a. Water potential control of turgor-driven tracheid enlargement in Scots pine at its xeric distribution edge. New Phytologist 225: 209–221.

Cabon A, Peters RL, Fonti P, Martínez-Vilalta J, De Cáceres M. 2020b. Temperature and water potential co-limit stem cambial activity along a steep elevational gradient. New Phytologist: nph.16456.

See Also

`growth`

Index