Package ‘medfate’

March 23, 2020

**Type**  Package  
**Title**  Mediterranean Forest Simulation  
**Version**  0.9.0  
**Date**  2020-03-05  
**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>].

**License**  GPL (>= 2)

**URL**  [https://vegmod.github.io/software/medfate](https://vegmod.github.io/software/medfate)

**LazyLoad**  yes

**Depends**  R (>= 3.5.0), sp

**Imports**  spdep, ggplot2, methods, meteoland (>= 0.8.1), Rcpp (>= 0.12.12)

**Suggests**  knitr, GSIF, rmarkdown, rjson

**LinkingTo**  Rcpp, meteoland

**Encoding**  UTF-8

**NeedsCompilation**  yes

**VignetteBuilder**  utils, knitr

**RoxygenNote**  6.1.1

**Author**  Miquel De Cáceres [aut, cre],  
            Shengli Huang [aut],  
            Víctor Granda [aut],  
            Antoine Cabon [aut]

**Maintainer**  Miquel De Cáceres <miquelcaceres@gmail.com>

**Repository**  CRAN

**Date/Publication**  2020-03-23 02:00:02 UTC
R topics documented:

biophysics ................................................................. 3
conductancefunctions .................................................. 4
defaultControl .......................................................... 7
defaultSoilParams ....................................................... 9
exampleforest ............................................................ 10
examplemeteo ............................................................. 10
fire_behaviour ............................................................ 11
forest ................................................................. 15
Forest manipulation ........................................................ 16
forest_mapWoodyTables .................................................. 17
fuel_properties ............................................................ 18
growth ................................................................. 21
hydrology_rainInterception ............................................. 24
hydrology_soilWaterInputs ............................................. 26
light ................................................................. 28
modifySpParams .......................................................... 31
pheno_updateLeaves ..................................................... 32
photo ................................................................. 33
Plant values .............................................................. 36
plot.spwb ............................................................... 38
plot.spwb_day ............................................................ 41
root ................................................................. 44
scalingconductance ...................................................... 46
SFM_metric ................................................................. 49
soil ................................................................. 50
soil hydrology ............................................................. 52
soil texture and hydraulics ............................................... 54
soil thermodynamics ........................................................ 57
soilgridsParams ............................................................ 58
Species values .............................................................. 60
SpParams ................................................................. 61
spwb ................................................................. 64
spwbInput ............................................................... 70
spwb_day ............................................................... 73
spwb_ladrCalibration .................................................... 76
spwb_ladrOptimization .................................................. 77
spwb_maximumTranspirationRatioPlot ................................. 80
spwb_resistances .......................................................... 82
spwb_sensitivity ........................................................... 83
spwb_stress .............................................................. 84
spwb_validation ............................................................ 85
spwb_waterUseEfficiency ................................................ 86
Stand values .............................................................. 88
supplyfunctions ............................................................. 89
tissuemoisture ............................................................. 94
transpiration ............................................................. 96
**biophysics**

*Physical and biophysical utility functions*

**Description**

Set of functions used in the calculation of biophysical variables.

**Usage**

```r
biophysics_leafTemperature(absRad, airTemperature, u, E, leafWidth = 1.0)
biophysics_radiationDiurnalPattern(t, daylength)
biophysics_temperatureDiurnalPattern(t, tmin, tmax, daylength)
```

**Arguments**

- `u` Wind speed above the leaf boundary layer (in m/s).
- `airTemperature` Air temperature (in °C).
- `tmin, tmax` Minimum and maximum daily temperature (°C).
- `absRad` Absorbed long- and short-wave radiation (in W·m-2).
- `E` Transpiration flow (in mmol H2O·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).

**Value**

Values returned for each function are:

- `biophysics_leafTemperature`: leaf temperature (in °C)
- `biophysics_radiationDiurnalPattern`: the proportion of daily radiation corresponding to the input time in seconds after sunrise.
- `biophysics_temperatureDiurnalPattern`: diurnal pattern of temperature assuming a sinusoidal pattern with $T = T_{min}$ at sunrise and $T = (T_{min} + T_{max})/2$ at sunset.

**Author(s)**

Miquel De Cáceres Ainsa, CTFC
References

Campbell, G. S., and J. M. Norman. 1998. AN INTRODUCTION TO ENVIRONMENTAL BIO-


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.

potential, reference crop and pan evaporation using standard meteorological data: a pragmatic syn-

See Also

spwb

conductancefunctions

Descriptive conductance functions

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_psi2Weibull(psi50, psi88)

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 con-
ductance.
ws Exponent of the whole-plant relative conductance Weibull function.
v Proportion of fine roots within each soil layer.
### conductancefunctions

**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**
- Water potentials (in MPa) corresponding to 50% and 88% of percent loss of conductance.

### 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, CTFC.

References

See Also
hydraulics_supplyFunctionPlot, hydraulics_maximumStemHydraulicConductance, spwb, soil

Examples

#Manual display of vulnerability curve
kstemmax = 4 # in mmol·m⁻²·s⁻¹·MPa⁻¹
steme = 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()

#Switch to 'Sperry' transpiration mode
control$transpirationMode="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 control parameters for models

Description

Creates a list with global default parameters for simulation models.

Usage

defaultControl()

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:

- **verbose (=TRUE)**: Boolean flag to indicate console output during calculations.
- **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.
- **soilFunctions (="SX")**: Soil water retention curve and conductivity functions, either 'SX' (for Saxton) or 'VG' (for Van Genuchten).
- **snowpack (=TRUE)**: Boolean flag to indicate the simulation of snow accumulation and melting.
- **drainage (=TRUE)**: Boolean flag to indicate the simulation of deep drainage.
- **leafPhenology (=TRUE)**: Boolean flag to indicate the simulation of leaf phenology for winter-deciduous species.
- **unlimitedSoilWater (=FALSE)**: Boolean flag to indicate the simulation of plant transpiration assuming that soil water is always at field capacity.
- **plantWaterPools (=FALSE)**: Boolean flag to indicate the simulation of water balance assuming that each cohort has its own water pools (mixing between pools of different cohorts is regulated via poolOverlapFactor).
- **poolOverlapFactor (=0.2)**: Maximum daily rate of mixing between water pools of different cohorts (actual rate depends on the proportion of fine roots, so the actual mixing rate is normally small for deep layers).
- **transpirationMode (="Granier")**: Transpiration model (either 'Granier' or 'Sperry'). See spwbInput.
- **cuticularTranspiration (=TRUE)**: Boolean flag to indicate the simulation of cuticular transpiration (at night or when stomata are fully closed) according to minimum stomatal conductance.
• **gainModifier, costModifier (= 1)**: Modifiers (exponents) of the gain and cost functions defined in Sperry et al. (2016).

• **ndailysteps (= 24)**: Number of steps into which each day is divided for determination of stomatal conductance, transpiration and photosynthesis (24 equals 1-hour intervals).

• **verticalLayerSize (= 100)**: The size of vertical layers (in cm) for photosynthesis calculation.

• **capacitance (=FALSE)**: Whether the effect of plant water compartments is considered in simulations.

• **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, "total" - instantaneous complete refilling).

• **refillMaximumRate (= 0.05)**: Maximum rate of daily refilling of embolized conduits as sapwood area per leaf area (in cm2-m-2-day-1).

• **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.

• **taper (= TRUE)**: Whether taper of xylem conduits is accounted for when calculating above-ground stem conductance from xylem conductivity.

• **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.

• **defaultWindSpeed (= 2.5)**: Default wind speed value (in m/s) to be used when missing from data.

• **Catm (=386)**: Atmospheric CO2 concentration (in micromol·mol-1 = ppm).

• **fracLeafResistance (= NA)**: Fraction of plant total resistance (leaf+stem+root) that corresponds to leaves. This fraction is used if VLeaf_kmax = NA.

• **fracRootResistance (= 0.40)**: Fraction of plant total resistance (leaf+stem+root) that corresponds to root system. If fracRootResistance = NA then root conductance is calculated from root xylem conductivity and root distribution.

• **averageFracRhizosphereResistance (=0.15)**: Fraction to total continuum (leaf+stem+root+rhizosphere) resistance that corresponds to rhizosphere (averaged across soil water potential values).

• **storagePool (="none")**: Whether carbon storage pools are considered (either "none", "one" or "two").

**Author(s)**

Miquel De Cáceres Ainsa, CTFC

**See Also**

spwbInput, spwb
defaultSoilParams

---

**defaultSoilParams**  
*Default soil parameters*

---

**Description**

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

**Usage**

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

**Arguments**

- `n`  
  An integer with the number of soil layers.

**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 for each layer.

**Author(s)**

Miquel De Cáceres Ainsa, CTFC

**See Also**

- `soil`, `soilgridsParams`, `defaultControl`, `SpParamsMED`

**Examples**

```r
defaultSoilParams(4)
```
**exampleforest**  
*Example forest stands*

**Description**

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

**Usage**

```r
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**


**See Also**

`forest`, `spwb`, `forest2spwbInput`

**Examples**

```r
data(exampleforestMED)
```

---

**examplemeteo**  
*Example daily meteorology data*

**Description**

Example data set of meteorological input.

**Usage**

```r
data(examplemeteo)
```
fire_behaviour

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/m^2).
- 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

```r
data(examplemeteo)
```

---

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

```r
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)
```
fire behaviour

**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$'I_R_surf [kJ/m2/min]'`: Intensity of the surface fire reaction.
- `SurfaceFire$'q_surf [kJ/m2]'`: Heat sink of the surface fire.
- `SurfaceFire$'x_surf'`: Propagating flux ratio of the surface fire.
- `SurfaceFire$'I_R_litter [kJ/m2/min]'`: Intensity of the litter fire reaction.
- `SurfaceFire$'q_litter [kJ/m2]'`: Heat sink of the litter fire.
- `SurfaceFire$'x_litter'`: Propagating flux ratio of the litter fire.
• **SurfaceFire**$' $ROS_s**: Spread rate of the surface fire (without accounting for faster spread in the litter layer).
• **SurfaceFire**$' $ROS_l**: Spread rate of the litter fire.
• **SurfaceFire**$' $ROS_w**: Maximum surface fire spread rate according to wind speed.
• **SurfaceFire**$' $ROS**: Final spread rate of the surface fire.
• **SurfaceFire**$' $I_b**: Fireline intensity of the surface fire.
• **SurfaceFire**$' $FL**: Flame length of the surface fire.
• **CrownFire**$' $I_R**: Intensity of the canopy fire reaction.
• **CrownFire**$' $q_c**: Heat sink of the crown fire.
• **CrownFire**$' $xi**: Propagating flux ratio of the crown fire.
• **CrownFire**$' $WAF**: Wind speed adjustment factor for crown fires.
• **CrownFire**$' $ROS**: Spread rate of the crown fire.
• **CrownFire**$' $I_c**: Crown initiation ratio.
• **CrownFire**$' $I_b**: Fireline intensity of the crown fire.
• **CrownFire**$' $FL**: Flame length of the crown fire.
• **FirePotentials**$' $RP**: Surface fire reaction potential ([0-9]).
• **FirePotentials**$' $SP**: Surface fire spread rate potential ([0-9]).
• **FirePotentials**$' $FP**: Surface fire flame length potential ([0-9]).
• **FirePotentials**$' $SFP**: Surface fire potential ([0-9]).
• **FirePotentials**$' $IC**: Crown initiation potential ([0-9]).
• **FirePotentials**$' $TC**: Crown-to-crown transmission potential ([0-9]).
• **FirePotentials**$' $RC**: Crown fire spread rate potential ([0-9]).
• **FirePotentials**$' $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, CTFC
References


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)
Description

Description of a forest stand.

Usage

```r
## 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.
mtree, 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, CTFC

See Also
texampleforestMED, forest_mapWoodyTables

Examples
data(exampleforestMED)
data(SpParamsMED)

summary(exampleforestMED, SpParamsMED)

---

Description
Functions to manipulate a forest object.

Usage
forest_mergeTrees(x)
forest_mapWoodyTables

Arguments
x
An object of class forest.

Value
Depending on the function:
• forest_mergeTrees: Another forest object with merged trees.

Author(s)
Miquel De Cáceres Ainsa, CTFC

See Also
spwb, forest, summary.forest

Description
Mapping functions to facilitate building forest objects from forest plot data

Usage
forest_mapWoodyTables(x, y, mapping, SpParams, plot.size = NULL)
forest_mapTreeTable(x, mapping, SpParams, plot.size = NULL)
forest_mapShrubTable(y, mapping, SpParams)

Arguments
x
A data frame with tree records in rows and attributes in columns.
y
A data frame with shrub records in rows and attributes in columns.
mapping
A named character vector to specify mappings of columns in x or y into attributes of treeData and shrubData data frames. Accepted names (and the corresponding specifications for the columns in x and y) are:
• "Species": Species code (should follow codes in SpParams).
• "Species.name": Species name. In this case, the species code will be drawn by matching names with species names in SpParams.
• "N": Tree density (in ind./ha).
• "plot.size": Plot size (in m2) to which each plot record refers to.
• "DBH": Diameter at breast height (in cm).
• "Height": Tree or shrub height (in cm).
• "Z50": Depth (in mm) corresponding to 50% of fine roots.
• "Z95": Depth (in mm) corresponding to 95% of fine roots.
SpParams: A data frame with species parameters (see SpParamsMED).

plot.size: The size of plot sampled area (in m²). Alternatively, 'plot.size' can be a column in x and specified in mapping to indicate that trees have been measured in different subplots and, therefore, they represent different densities per hectare.

Value

Functions forest_mapTreeTable and forest_mapShrubTable return a data frame with the structure of treeData and shrubData from forest objects. Function forest_mapWoodyTable returns directly a forest object.

Author(s)

Miquel De Cáceres Ainsa, CTFC

See Also

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). Function `fuel_cohortFineFMC` calculates the fuel moisture content of leaves and twigs of each cohort, from the results of soil water balance.

**Usage**

```r
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)
fuel_cohortFineFMC(spwb, x)
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. taken from the result of fuel_cohortFineFMC).
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.
spwb       Object returned by function spwb.
topShrubHeight Shrub stratum top height (in m).
bottomCanopyHeight Canopy base height (in m).
topCanopyHeight  Canopy top height (in m).
canopyCover  Canopy percent cover.
x           An object of class spwbInput.

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).
- `etaF`: Flammability modifier (between 1 and 2).
- `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, Centre Tecnologic Forestal de Catalunya

**References**


**See Also**

`fire_FCCS`, `spwb`
Examples

```r
# 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 forest growth model that calculates changes in leaf area, sapwood area and structural variables for all plant cohorts in a given forest stand during a period specified in the input climatic data.

Usage

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

Arguments

- **x**
  - An object of class `growthInput`.
- **soil**
  - A list containing the description of the soil (see `soil`).
- **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.

**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.
- "spwbInput": A copy of the object x of class spwbInput given as input.
- "soilInput": A copy of the object soil of class soil given as input.
- "WaterBalance": A data frame where different variables (in columns) are given for each simulated day (in rows):
  - "PET": Potential evapotranspiration (in mm).
  - "Rainfall": Input precipitation (in mm).
  - "NetPrec": Net precipitation, 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).
  - "Transpiration": Plant transpiration (considering all soil layers) (in mm).
- "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).
- "PlantExt.1", ..., "PlantExt.k": Plant extraction from each soil layer (in mm).
- "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):
  - "GDD": Cumulative growth degree days.
  - "LAIcell": LAI of the stand (accounting for leaf phenology) (in m²/m²).
  - "LAIcelldead": LAI of the stand corresponding to dead leaves (in m²/m²).
  - "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).
- "PlantTranspiration": A data frame with the amount of daily transpiration (in mm) for each plant cohort. Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
- "PlantPhotosynthesis": A data frame with the amount of daily photosynthesis (in g C · m⁻²) for each plant cohort. Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
- "PlantRespiration": A data frame with the amount of daily maintenance respiration (in g C · m⁻²) for each plant cohort. Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
- "PlantCStorageFast": A data frame with the daily amount of fast-dynamics carbon reserves (as proportion of total storage capacity) for an average individual of each plant cohort. Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
- "PlantCStorageSlow": A data frame with the daily amount of slow-dynamics carbon reserves (as proportion of total storage capacity) for an average individual of each plant cohort. Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
- "PlantSAgrowth": A data frame with the daily amount of newly created sapwood area (in cm²) for an average individual of each plant cohort. Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
- "PlantSA": A data frame with the daily amount of sapwood area (in cm²) for an average individual of each plant cohort. Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
- "PlantStress": A data frame with the amount of daily stress suffered by each plant cohort (relative whole-plant conductance). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
- "PlantPsi": A data frame with the average daily water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
- "PlantLAIded": A data frame with the dead leaf area index (in m² · m⁻²) for each plant cohort. Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.

• "PlantLAIlive": A data frame with the live leaf area index (in m2 · m-2) for each plant cohort. Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in `SP`.

Note

Objects x and soil are modified during the simulation.

Author(s)

Miquel De Cáceres Ainsa, CTFC

See Also

growthInput, 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 (2 layers)
examplesoil = soil(defaultSoilParams(2))

#Initialize control parameters
control = defaultControl()

#Initialize input
x = forest2growthInput(exampleforestMED, examplesoil, SpParamsMED, control)

#Call simulation function
G1<-growth(x, examplesoil, examplemeteo, elevation = 100)

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, CTFC
References


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,
   drainage = 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 (°C).
- **rad**: Solar radiation (in MJ/m²/day).
- **elevation**: Altitude above sea level (m).
- **Cm**: Canopy water storage capacity.
- **LgroundPAR**: Proportion of photosynthetically-active 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.
- **drainage**: Boolean flag to indicate the simulation of deep drainage.
- **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, CTFC

See Also
spwb_day, hydrology_rainInterception, hydrology_soilEvaporation

light

Light extinction and absorption 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, LWR_diffuse, 
ntimesteps = 24, canopyMode= "sunshade", 
trunkExtinctionFraction = 0.1)
light_cohortAbsorbedSWRFraction(z, x, SpParams, gdd = NA)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAIme</td>
<td>A numeric matrix of live expanded LAI values per vegetation layer (row) and cohort (column).</td>
</tr>
<tr>
<td>LAImd</td>
<td>A numeric matrix of dead LAI values per vegetation layer (row) and cohort (column).</td>
</tr>
<tr>
<td>LAImx</td>
<td>A numeric matrix of maximum LAI values per vegetation layer (row) and cohort (column).</td>
</tr>
<tr>
<td>k</td>
<td>A vector of light extinction coefficients.</td>
</tr>
<tr>
<td>kb</td>
<td>A vector of direct light extinction coefficients.</td>
</tr>
</tbody>
</table>
kd A vector of diffuse light extinction coefficients.
Ib\(0\) Above-canopy direct incident radiation.
Id\(0\) 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 vector 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 vector of short-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).
LWR\_diffuse A vector with diffuse longwave radiation for different subdaily time steps.
ntimesteps Number of subdaily time steps.
canopyMode Indicates how crowns should be described to calculate photosynthesis/extinction. Accepted values are “sunshade” (distinguishes photosynthesis in sun leaves from shade leaves) and “multilayer” (distinguishes photosynthesis of sun leaves and shade leaves in each canopy layer).
trunkExtinctionFraction Fraction of extinction due to trunks (for winter deciduous forests).
xs 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

Function codification adapted from Anten & Bastiaans (2016). 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, CTFC
References


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((1-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,
modifySpParams

Modify species/cohort parameters

Description

Routines to modify input species parameters or model input objects

Usage

modifySpParams(SpParams, customParams, subsetSpecies = TRUE)
modifyCohortParams(x, customParams)

Arguments

x              A model input object of class spwbInput.
SpParams       A species parameter data frame, typically SpParamsMED.
customParams   A data frame with new parameter values (see details).
subsetSpecies A flag to indicate that the only species mentioned in customParams should be returned.

Details

Data frame customParams should have as many rows as species or cohorts and as many columns as parameters to modify. An additional column is needed to identify species/cohorts. In the case of modifySpParams the column should be named ‘SpIndex’ (to match the corresponding column of SpParamsMED), while in modifyCohortParams the column should be named ‘Cohort’ and will be matched with the cohort names given by spwbInput or growthInput.
Value

Function `modifySpParams` returns a modified species parameter data frame. Function `modifyCohortParams` returns a modified `spwbInput` object.

Author(s)

Miquel De Cáceres Ainsa, CTFC

See Also

`spwbInput`, `SpParamsMED`

Examples

```r
## TO DO
```

---

`pheno_updateLeaves`  
*Leaf phenology*

Description

Function `pheno_leafDevelopmentStatus` returns the fenological status ([0-1]) of species according to the growth degree days required for leaf flush. Function `pheno_updateLeaves` updates the status of expanded leaves and dead leaves of object `x` given the temperature and wind of a given day.

Usage

```r
pheno_leafDevelopmentStatus(Sgdd, gdd)
pheno_updateLeaves(x, doy, tmean, wind, Tbase = 5)
```

Arguments

- **Sgdd**: A numeric vector of growth degree days for leaf budburst (in Celsius).
- **gdd**: Growth degree days (in Celsius)
- **x**: An object of class `spwbInput`.
- **doy**: Day of the year.
- **tmean**: Average day temperature (in Celsius).
- **wind**: Average day wind speed (in m/s).
- **Tbase**: Base temperature to update growth degree days (in Celsius).

Value

Function `pheno_leafDevelopmentStatus` returns a vector of values between 0 and 1.
Note
Function pheno_updateLeaves modifies the object x.

Author(s)
Miquel De Cáceres Ainsa, CTFC

See Also
spwb

---

**Photosynthesis submodel functions**

**Description**
Set of functions used in the calculation of photosynthesis.

**Usage**

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

**Arguments**

- **leaf temp** Leaf temperature (in °C).
- **Oi** Oxigen concentration (mmol/mol).
Vmax298, Vmax298SL, Vmax298SH
Maximum Rubisco carboxylation rate per leaf area at 298°K (i.e. 25 °C) (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 298°K (i.e. 25 °C) (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).

Patm
Atmospheric air pressure (in kPa).

Tair
Air temperature (in °C).

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).

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 Oxygen (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).
  - **WaterVaporConductance**: Leaf vapor conductance (mol * s-1 * m-2).
  - **Photosynthesis**: Gross photosynthesis (micromol*s-1*m-2).
  - **NetPhotosynthesis**: Net photosynthesis, after discounting autotrophic respiration (micromol*s-1*m-2).
- **photo_sunshadePhotosynthesisFunction** and **photo_multilayerPhotosynthesisFunction**: Return a data frame with the following columns:
  - **Photosynthesis**: 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, Centre Tecnologic Forestal de Catalunya

References


Plant values

See Also

`hydraulics_supplyFunctionNetwork`, `biophysics_leafTemperature`, `spwb`

---

### Plant values

#### Plant description functions

**Description**

Functions to calculate attributes of plants in a `forest` object.

**Usage**

```r
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)
plant_equilibriumSmallBranchLitter(x, SpParams,
  smallBranchDecompositionRate = 0.81)
plant_foliarBiomass(x, SpParams, gdd = NA, mode = "MED")
plant_fuel(x, SpParams, gdd = NA, includeDead = TRUE, mode = "MED")
plant_height(x)
plant_ID(x)
plant_LAI(x, SpParams, gdd = NA, mode = "MED")
plant_parameter(x, SpParams, parName)
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.
Plant 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, Centre Tecnologic Forestal de Catalunya

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 Displays simulation results

Description

Function plot plots the results of the soil plant water balance model (see spwb), plant water balance model (see pwb) or the forest growth model (see growth), whereas function summary summarizes the model's output in different temporal steps (i.e. weekly, annual, ...).

Usage

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

- `x`, `object` An object of class spwb.
- `type` The information to be plotted:
  - "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.
• "PlantTranspirationPerLeaf": Plant cohort transpiration per leaf area.
• "PlantPhotosynthesis": Plant cohort photosynthesis.
• "PlantPhotosynthesisPerLeaf": Plant cohort photosynthesis per leaf area.
• "PlantWUE": Plant cohort daily water use efficiency (photosynthesis over transpiration).
• "PlantAbsorbedSWR": Plant cohort absorbed short wave radiation (only for transpirationMode = "Sperry").
• "PlantAbsorbedSWRPerLeaf": Plant cohort absorbed short wave radiation per leaf area (only for transpirationMode = "Sperry").
• "PlantAbsorbedLWR": Plant cohort absorbed long wave radiation (only for transpirationMode = "Sperry").
• "PlantAbsorbedLWRPerLeaf": Plant cohort absorbed 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").
• "PlantRespiration": Plant cohort respiration (only for plot.growth).
• "PlantRespirationPerLeaf": Plant cohort respiration per leaf area (only for plot.growth).
• "PlantRespirationPerIndividual": Respiration per individual (only for plot.growth).
• "PlantCarbonBalance": Plant cohort carbon balance (only for plot.growth).
• "PlantCarbonBalancePerLeaf": Plant cohort carbon balance per leaf area (only for plot.growth).
• "PlantCarbonBalancePerIndividual": Carbon balance per individual (only for plot.growth).
• "PlantCstorageFast": Amount of fast-dynamics carbon reserves (only for plot.growth).
• "PlantCstorageSlow": Amount of slow-dynamics carbon reserves (only for plot.growth).
• "PlantSA": Amount of sapwood area in an individual (only for plot.growth).
• "PlantSAgrowth": Amount of newly-created sapwood area (only for plot.growth).
• "PlantRelativeSAgrowth": Amount of newly-created sapwood area per sapwood area (only for plot.growth).
• "PlantLAIlive": Plant cohort leaf area index of live leaves (only for plot.growth).
• "PlantLAIdead": Plant cohort leaf area index of dead leaves (only for plot.growth).

bySpecies Allows aggregating output by species, before calculating summaries or 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).

xlim Range of values for x.
ylim Range of values for y.
xlab x-axis label.
ylab y-axis label.
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, ...)
... Additional parameters for function plot or summary.
Author(s)

Miquel De Cáceres Ainsa, CTFC

References


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()

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

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

#Plot results
plot(S1)

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

plot.spwb_day Displays simulation results for one day
Description

Functions to plot the results of the soil water balance model for one day (see `spwb_day`) or the transpiration calculations with Sperry (see `transp_transpirationSperry`).

Usage

## 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` or `pwb_day`.

- **type**  
  The information to be plotted:
  - "LeafPsi": Leaf water potential.
  - "StemPsi": (Upper) stem water potential.
  - "RootPsi": Root crown water potential.
  - "StemPLC": (Average) percentage of loss conductance in the stem conduits.
  - "StemRWC": (Average) relative water content in the stem symplasm.
  - "LeafRWC": 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.
  - "PlantTranspirationPerLeaf": Plant cohort transpiration per leaf area.
  - "PlantPhotosynthesis": Plant cohort daily net photosynthesis per ground area.
  - "PlantPhotosynthesisPerLeaf": Plant cohort daily net photosynthesis per leaf area.
  - "PlantWaterBalance": Difference between water extraction from the soil and transpired water per ground 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).
  - "LeafPhotosynthesis": Instantaneous net photosynthesis per leaf area (differentiates sunlit and shade leaves).
  - "LeafAbsorbedSWR": Absorbed short wave radiation per leaf area (differentiates sunlit and shade leaves).
  - "LeafVPD": Leaf vapour pressure deficit (differentiates sunlit and shade leaves).
• "LeafStomatalConductance": Leaf stomatal conductance (differentiates	sunlit and shade leaves).
• "LeafTemperature": Leaf temperature (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 (differenti-
tiates sunlit and shade leaves).
• "Temperature": Above-canopy, inside-canopy and soil temperature.
• "CanopyEnergyBalance": Canopy energy balance components.
• "SoilEnergyBalance": Soil energy balance components.

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.
xlabel x-axis label.
ylabel y-axis label.
... Additional parameters for function plot.

Note
Only for soil plant water balance simulations using transpirationMode = "Sperry".

Author(s)
Miquel De Cáceres Ainsa, CTFC

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
ccontrol = defaultControl()
ccontrol$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
ccontrol$transpirationMode="Sperry"

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

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

---

**root**  
**Distribution of fine roots**

**Description**
Functions to calculate the distribution of fine roots within the soil, given root system parameters and soil layer definition (layer widths).

**Usage**
- root_conicDistribution(Zcone, d)
- root_ldrDistribution(Z50, Z95, d)
- root_xylemConductanceProportions(v, d, depthWidthRatio = 1)
- root_rootLengths(v, d, depthWidthRatio = 1.0)
- root_rhizosphereOverlapProportions(V, LAAllive, poolOverlapFactor)

**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**  
  Proportions of fine roots, as returned by functions root_conicDistribution or root_ldrDistribution.
- **depthWidthRatio**  
  Ratio between radius of the soil layer with the largest radius and maximum rooting depth.
- **V**  
  Matrix of proportions of fine roots (cohorts x soil layers).
LAIlive  Vector of Leaf Area Index values per cohort.

poolOverlapFactor  Factor to regulate the influence of LAI on root overlap.

Details

Function root_conicDistribution assumes a conic distribution of fine roots, whereas function root_ldrDistribution distributes fine roots according to the linear dose response model of Schenck & Jackson (2002). Function root_xylemConductanceProportions calculates the proportion of total root xylem conductance that can be attributed to each layer, according to layer widths and the proportion of fine roots (Sperry et al. 2016). Function root_rhizosphereOverlapProportions calculates the proportion of occupied root space that is shared with other cohorts, per cohort and soil layer.

Value

Functions root_conicDistribution and root_ldrDistribution return a matrix with as many rows as elements in Z (or Z50) and as many columns as soil layers. Values in all cases correspond to the proportion of fine roots in each soil layer. Function root_xylemConductanceProportions returns a vector of proportions of the same length as the number of layers.

Author(s)

Miquel De Cáceres Ainsa, CTFC

References


See Also

spwb, spwb_ldrOptimization, forest2spwbInput, soil

Examples

#Load example plot plant data
data(exampleforestMED)

data(SpParamsMED)

ntree = nrow(exampleforestMED$treeData)

S = soil(defaultSoilParams())

#Calculate conic root system for trees
scalingconductance

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)

# Equal xylem conductance proportions for a cone distribution
# (assuming depth-width ratio 1)
root_xylemConductanceProportions(V1[,1], S$dVec)

# Xylem conductance proportions for LDR distribution
root_xylemConductanceProportions(V2[,1], S$dVec)

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)
hydraulics_maximumRootHydraulicConductance(xylemConductivity, Al2As,
v, widths, depthWidthRatio = 1.0)
hydraulics_maximumStemHydraulicConductance(xylemConductivity, refheight, Al2As, height,
 angiosperm = TRUE, taper = FALSE)
hydraulics_referenceConductivityHeightFactor(refheight, height)
hydraulics_terminalConduitRadius(height)
hydraulics_taperFactorSavage(height)
scalingconductance

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.
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).
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).
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 (in cm).
angiosperm A boolean flag to indicate an angiosperm species.
taper A boolean flag to indicate correction by taper of xylem conduits (Christoffersen et al. 2017).
widths Soil layer depths (in mm).
depthWidthRatio Ratio between radius of the soil layer with the largest radius and maximum rooting depth.
SLA Specific leaf area (mm²·mg⁻¹).
wd Wood density (g·cm⁻³).
lld 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, Centre Tecnologic Forestal de Catalunya

References


See Also

- `hydraulics_psi2K`
- `hydraulics_supplyFunctionPlot`
- `spwb`
- `soil`

Examples

```r
kstemmax = 4 # in mmol·m⁻²·s⁻¹·MPa⁻¹
stemp = 3
stend = -4 # in MPa
```

**Description**

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

**Usage**

```r
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**


soil

Soil initialization

Description
Initialize 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", ...)

Arguments
- SoilParams: A list of soil parameters (see 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.
- ...: 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.

Value
An 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 °C) of each layer.

See Also
fire_Rothermel

Examples
data(SFM_metric)
• 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).
• usda_Type: USDA texture type.
• 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.
• macro: Macroporosity for each layer (estimated using Stolf et al. 2011).
• rfc: Percentage of rock fragment content for each layer.

Author(s)
Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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 infiltration.
**soil hydrology**

**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, CTFC

**References**


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

**See Also**

`spwb`, `hydrology_soilWaterInputs`

**Examples**

```r
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

load("soiltexture.R")
load("soilhydraulics.R")
load("soilhydrology.R")
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

Description

Functions soil_psi2thetaSX and soil_theta2psiSX calculate water potentials (psi) 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. Functions codesoils_psi2thetaVG and soil_theta2psiVG to the same calculations as before, but using the Van Genuchten - Mualem equations (Wösten & van Genuchten 1988). Function soil_USDAType returns the USDA type for a given texture. Function soil_vanGenuchtenParamsCarsel gives parameters for van Genuchten-Mualem equations for a given texture type (Leij et al. 1996), whereas function soil_vanGenuchtenParamsToth gives parameters for van Genuchten-Mualem equations for a given texture, organic matter and bulk.
Correspondingly, functions `soil_waterFC` and `soil_thetaFC` calculate the water volume (in mm or as percent of soil volume) of each soil layer at field capacity, according to a given water retention model. Functions `soil_waterWP` and `soil_thetaWP` calculate the water volume (in mm or as percent of soil volume) of each soil layer at wilting point (-1.5 MPa), and functions `soil_waterSAT`, `soil_thetaSATSX` and `soil_thetaSAT` calculate the saturated water volume (in mm or as percent of soil volume) of each soil layer. Functions `soil_psi`, `soil_water` and `soil_theta` return the current water potential and water content (in mm or as % volume) of the soil object, according to a given water retention model. Function `soil_rockWeight2Volume` transform rock percentage from weight to volume basis.

### Usage

```r
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_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).
- **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).

**Value**

Functions `soil_psi2thetaSX` and `soil_psi2thetaVG` return the soil water potential (in MPa) from soil volumetric water content, and functions `soil_theta2psiSX` and `soil_theta2psiVG` do the reverse calculation returning water potential in MPa. Function `soil_USDAType` returns a string. Function `soil_vanGenuchtenParamsToth` and `soil_vanGenuchtenParamsCarsel` return a vector with four parameter values (alpha, n, theta_res and theta_sat, where alpha is in MPa-1). Function `soil_waterTableDepth` returns water table depth in mm from surface.

**Author(s)**

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

**References**


**See Also**

`soil`
soil thermodynamics

Examples

```r
# 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 °C/m) and temporal temperature change (in °C/s) given soil layer texture and water content (and possibly including heat flux from above).

Usage

```r
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 °C) 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/mºK) for each soil layer. Function `soil_thermalCapacity` returns a vector with values of heat storage capacity (J/m³ºK) 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, Centre Tecnologic Forestal de Catalunya

References


See Also

`soil`

Examples

```r
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)
```

---

soilgridsParams

*SoilGrids soil description fetcher*

Description

soilgridsParams takes a vector of depths and returns a list of soil characteristics ready to use with `soil` function (requires package GSIF).

Usage

```r
soilgridsParams(points, depths = c(300, 500, 1200), verbose = FALSE)
```
soilgridsParams

Arguments

points An object of class (or subclass) SpatialPoints with a valid CRS definition.
depths A numeric vector indicating the desired depths, in mm
verbose A logical flag to include a progress bar while processing the output of the query to the SoilGrids REST API.

Details

This function connects with the SoilGrids REST API (https://rest.soilgrids.org) to retrieve the soil physical and chemical characteristics for a site, selected by its coordinates. Also, in case the depths are not the default ones in the SoilGrids API, the function uses the trapezoidal rule to calculate the values for the desired depths (as described in Hengl et al. 2007). To do this, it uses internally the rjson and GSIF R packages. Although soil layer definition is taken from the input, the function includes absolute depth to bedrock and depth to the R horizon from SoilGrids, in case the user wants to redefine soil definition. Input coordinates are transformed to longitude/latitude within the function.

Value

If only one point is supplied, the function returns a list with the following elements:

• soilparams: A data frame containing the soil characteristics ready to use with the soil function.
• soilgrids_Rhorizondepth: Depth to the R horizon (in mm).
• soilgrids_Rhorizonprob: Probability of the R horizon.
• soilgrids_absolutesoildepth: Absolute depth to bedrock (in mm).

If more than one point is supplied, the function returns a list with as many elements as points, each one containing the mentioned list.

Author(s)

Víctor Granda, CREAF/CTFC
Miquel De Cáceres Ainsa, CTFC

References


See Also

soil, defaultSoilParams
Examples

```r
## Not run:
library(GSIF)
library(rjson)
coords_sp <- SpatialPoints(cbind(long = -5.6333, lat = 42.6667),
                           CRS("+proj=longlat +datum=WGS84"))
foo <- soilgridsParams(coords_sp, depths = c(300, 600, 1200))
foo_soil <- soil(foo$soilparams)
foo_soil
## End(Not run)
```

---

### Species values

#### Species description functions

**Description**

Functions to calculate attributes of a `forest` object by species.

**Usage**

```r
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)
```

**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**

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).

**Description**

A data set of species parameter values, resulting from bibliographic search, fit to empirical data or expert-based guesses.

**Usage**

```r
data("SpParamsMED")
data("SpParamsUS")```

**Format**

A data frame with 89 observations (species) on the following 77 variables:

- **Name**: A factor with taxon names (mostly species names).
- **IFNcodes**: A factor with levels corresponding to species codes in the Third Spanish forest inventory (DGCN 2005.)
- **SpIndex**: A numeric vector of the species index.
- **Group**: Either "Gymnosperm" or "Angiosperm".
- **Order**: Taxonomic order.

**See Also**

`spwb, forest, plant_basalArea, summary.forest`
Family  Taxonomical family.
GrowthForm  A factor with levels Shrub, Tree and Tree/Shrub.
TreeType  A factor with levels Conifer, Deciduous, Evergreen or Shrub
Hmed  Median plant height (in cm).
Hmax  Maximum plant height (in cm).
Z50  Rooting depth (in mm) corresponding to 50% of fine roots.
Z95  Rooting depth (in mm) corresponding to 95% of fine roots.
Zmax  Maximum rooting depth (in mm).
a_ash  Regression coefficient relating the square of shrub height with shrub area.
a_bsh, b_bsh  Allometric coefficients relating phytovolume with dry weight of shrub individuals.
cr  Ratio between crown length and total height (for shrubs).
a_fbt, b_fbt, c_fbt, d_fbt  Regression coefficients used to calculate foliar biomass of an individual tree from its dbh and the cumulative basal area of larger trees.
a_cr, b_1cr, b_2cr, b_3cr, c_1cr, c_2cr  Regression coefficients used to calculate crown ratio of trees.
a_cw, b_cw  Regression coefficients used to calculated crown width of trees.
SLA  Specific leaf area (mm^2/mg = m^2/kg).
LeafDensity  Density of leaf tissue (dry weight over volume).
r635  Ratio between the weight of leaves plus branches and the weight of leaves alone for branches of 6.35 mm.
pDead  Proportion of total fine fuels that are dead.
maxFMC  Maximum fuel moisture (in percent of dry weight)
minFMC  Minimum fuel moisture (in percent of dry weight)
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.
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.
Cstoragepmax  Maximum storage capacity, expressed as the fraction of C per total respiratory of C.
LeafDuration  Leaf duration (in years).
LigninPercent  Percent of lignin+cutin over dry weight in leaves.
ParticleDensity  Particle density (kg/m^3).
LeafLitterFuelType  Fuel type for leaf litter, with levels Broadleaved, LongLinear, Scale and ShortLinear.
Flammability  Flammability modifier (either 1 or 2 for normal or high, respectively).
SAV  Surface-area-to-volume ratio of the small fuel (1h) fraction (leaves and branches < 6.35mm) (m²/m³).

HeatContent  High fuel heat content (kJ/kg).

fH0min  Minimum value of the height-to-diameter ratio (dimensionless).

fH0max  Maximum value of the height-to-diameter ratio (dimensionless).

gammaSWR  Reflectance (albedo) for SWR (gammaPAR = 0.8*gammaSWR).

alphaSWR  Absorbance coefficient for SWR (alphaPAR = alphaSWR * 1.35).

kPAR  Extinction coefficient for PAR (kSWR = kPAR/1.35).

g  Canopy water storage capacity per LAI unit (in mm/LAI).

Sgdd  Growth degree days for leaf budburst (in Celsius).

A12As  Leaf area to sapwood area ratio (in m²·m⁻²).

WUE  Water use efficiency for carbon assimilation (g C /mm water).

WoodC  Wood carbon content per dry weight (g C /g dry).

WoodDensity  Wood density (at 0 percent humidity!).

RGRmax  Maximum relative growth rate (in basal area or sapwood area) (in cm²·cm⁻²).

Psi_Extract  Water potential corresponding to 50% relative conductance (in MPa).

pRootDisc  Relative root conductance leading to hydraulic disconnection from a soil layer.

Gwmin  Minimum stomatal conductance to water vapor per leaf area unit (in mol·s⁻¹·m⁻²).

Gwmax  Maximum stomatal conductance to water vapor per leaf area unit (in mol·s⁻¹·m⁻²).

VCleaf_kmax  Leaf hydraulic conductance (in mmol H₂O·s⁻¹·m⁻²·MPa⁻¹).

VCleaf_c, VCleaf_d  Parameters of the leaf vulnerability curve (VCleaf_d in MPa).

Kmax_stemxylem  Sapwood-specific hydraulic conductivity of stem xylem (in kg H₂O·s⁻¹·m⁻¹·MPa⁻¹).

VCstem_c, VCstem_d  Parameters of the stem xylem vulnerability curve (VCstem_d in MPa).

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

VCroot_c, VCroot_d  Parameters of the root xylem vulnerability curve (VCroot_d in MPa).

LeafWidth  Leaf width (in cm).

Narea  Nitrogen concentration per leaf area (in g·m⁻²).

Vmax298  Maximum Rubisco carboxilation rate at 25°C (in micromol CO₂·s⁻¹·m⁻²).

Jmax298  Maximum electron transport rate at 25°C (in micromol electrons·s⁻¹·m⁻²).

Details

See details of parameterization in De Caceres et al. (2015) and De Caceres et al. (submitted).

Source


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 and plant transpiration and drought stress at daily steps for a given forest stand during a period specified in the input climatic data. Additionally, the function also calculates plant net photosynthesis. 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 transpiration and photosynthesis processes are conducted with different level of detail depending on the transpiration mode.

**Usage**

```r
spwb(x, soil, meteo, latitude = NA, elevation = NA, slope = NA, aspect = NA)
pwb(x, soil, meteo, W, latitude = NA, elevation = NA, slope = NA, aspect = NA, 
    canopyEvaporation = numeric(0), 
    snowMelt = numeric(0), 
    soilEvaporation = numeric(0))
spwb_resetInputs(x, soil, from = NULL, day = NA)
```

**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. 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.

\( W \) 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). 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. Elevation is also required for 'Granier' if snowpack dynamics are simulated.

### from
An object of class `spwb` storing the results of a previous simulation, including values of state variables. If `from = NULL`, state variables are set to their defaults (i.e. soil moisture set to field capacity and cumulative growth degree days set to zero).

### day
An integer with the day from which state variable values stored in `from` should be copied. If missing, values are copied from the first day of stored values.

### 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
Detailed model description is available in the vignettes section. 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 and are illustrated by function `transp_transpirationSperry`. Function `spwb_resetInputs()` allows resetting state variables in `x` and `soil` to their defaults, or to copy values of state variables from a previous `spwb()` simulation results stored in `from`.

### 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).
• "soilInput": An (unmodified) copy of the object soil of class soil given as input (note that soil 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").

• "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):
  - "GDD": Cumulative growth degree days.
  - "LAIcell": LAI of the stand (accounting for leaf phenology) (in m²/m²).
  - "LAIcelldead": 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).
• "PlantLAI": A data frame with the daily leaf area index for each plant cohort. Days are in rows and plant cohorts are in columns.
• "PlantAbsorbedSWRFraction": A data frame with the fraction of SWR absorbed by each plant cohort. Days are in rows and plant cohorts are in columns.
• "PlantTranspiration": A data frame with the amount of daily transpiration (in mm) for each plant cohort. Days are in rows and plant cohorts are in columns.
• "PlantPhotosynthesis": A data frame with the amount of daily net photosynthesis (in g C·m⁻²) for each plant cohort. Days are in rows and plant cohorts are in columns.
• "PlantPsi": A data frame with the average daily water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
• "PlantStress": A data frame with the amount of daily stress [0-1] suffered by each plant cohort (relative whole-plant conductance). Days are in rows and plant cohorts are in columns.
• "subdaily": A list of objects of class spwb_day, one per day simulated (only if required in control parameters, see defaultControl).

If transpirationMode="Sperry" the list also includes the following elements:

• "PlantAbsorbedSWR": A data frame with the daily SWR absorbed by each plant cohort. Days are in rows and plant cohorts are in columns.
• "PlantAbsorbedLWR": A data frame with the daily LWR absorbed by each plant cohort. Days are in rows and plant cohorts are in columns.
• "dEdP": A data frame with mean daily values of soil-plant conductance (derivative of the supply function) for each plant cohort. Days are in rows and plant cohorts in columns.
• "EnergyBalance": A data frame with the daily values of energy balance components for the soil and the canopy.
• "Temperature": A data frame with the daily values of minimum/mean/maximum temperatures for the atmosphere (input), canopy and soil.
• "LeafPsiMin": A data frame with the minimum (midday) daily (average) leaf water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
• "LeafPsiMax": A data frame with the maximum (predawn) daily (average) leaf water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
• "LeafPsiMin_SL": A data frame with the minimum (midday) daily sunlit leaf water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
• "LeafPsiMax_SL": A data frame with the maximum (predawn) daily sunlit leaf water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.
• "LeafPsiMin_SH": A data frame with the minimum (midday) daily shade leaf water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.

• "LeafPsiMax_SH": A data frame with the maximum (predawn) daily shade leaf water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.

• "LeafRWC": A data frame with the average daily leaf relative water content of each plant (in percent). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.

• "StemPsi": A data frame with the minimum daily stem water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.

• "StemPLC": A data frame with the average daily proportion of stem conductance loss of each plant ([0-1]). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.

• "StemRWC": A data frame with the average daily stem relative water content of each plant (in percent). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.

• "RootPsi": A data frame with the minimum daily root water potential of each plant (in MPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.

• "RhizoPsi": A list of data frames (one per plant cohort) with the minimum daily root water potential of each plant (in MPa). In each data frame, days are in rows and layers are in columns.

Note

State variables stored in objects x and soil are modified during the simulation. Use function spwb_resetInputs() to reset state variables to defaults. Daily transpiration and photosynthesis values are stored in columns Transpiration and Photosynthesis of object x. Water content relative to field capacity (vector W) of soil is also modified.

Author(s)

Miquel De Cáceres Ainsa, CTFC

References


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 (2 layers)
examplesoil = soil(defaultSoilParams(2))

# Initialize control parameters
control = defaultControl()

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

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

# Plot results
plot(S1)

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

## Not run:
# Initialize soil with default soil params (2 layers)
examplesoil2 = soil(defaultSoilParams(2))

# Switch to 'Sperry' transpiration mode
control$transpirationMode="Sperry"

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

# Call simulation function (11 days)
d = 100:110
S2<-spwb(x2, examplesoil2, examplemeteo[d,], latitude = 41.82592, elevation = 100)

# Run the model with 'Sperry' transpiration mode using the water balance of
# simulated with the 'Granier' model
P2<-pwb(x2, examplesoil2, examplemeteo[d,], latitude = 41.82592, elevation = 100,
W = as.matrix(S1$Soil[d, c("W.1", "W.2")]),
canopyEvaporation = S1$WaterBalance$Interception[d],
snowMelt = S1$WaterBalance$Snowmelt[d],
soilEvaporation = S1$WaterBalance$SoilEvaporation[d])

## End(Not run)
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 does the same from input data. Function forest2aboveground calculates aboveground variables that may be used in spwbInput and growthInput functions. Similarly, function forest2belowground calculates belowground root distribution that may be used in spwbInput and growthInput functions.

Usage

forest2aboveground(x, SpParams, gdd = NA, mode = "MED")
forest2belowground(x, soil, SpParams)
forest2growthInput(x, soil, SpParams, control)
forest2spwbInput(x, soil, SpParams, control)
growthInput(above, Z, V, soil, SpParams, control)
spwbInput(above, V, 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, LAI_live, LAI_dead, H and CR.
Z A numeric vector with cohort rooting depths (in cms).
V A numeric matrix (with as many columns as soil layers and as many rows as the length as SP) containing the proportion of roots of each plant in each soil layer.

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).
- **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`).
- **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 list. If `control$transpirationMode = "Granier"` it contains a single element:
  - **V**: A matrix with the proportion of fine roots 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).

- **paramsBase**: A data frame with columns:
  - **kPAR**: PAR extinction coefficient.
  - **g**: Canopy water retention capacity per LAI unit (mm/LAI).
  - **Sgdd**: Growth degree days needed for leaf budburst (for winter deciduous species).

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

- **paramsTransp**: A data frame with transpiration parameters. If `control$transpirationMode = "Granier"`, columns are:
  - **Psi_Extract**: Water potential corresponding to 50% relative conductance (in MPa).
  - **WUE**: Water use efficiency for carbon assimilation (g C/mm water).

  If `control$transpirationMode = "Sperry"` columns are:
  - **Gwmax**: Maximum stomatal conductance to water vapor (in mol H2O·m-2·s-1).
  - **Vmax298**: Maximum Rubisco carboxylation rate at 25°C (in micromol CO2·s-1·m-2).
  - **Jmax298**: Maximum rate of electron transport at 25°C (in micromol photons·s-1·m-2).
  - **VCroot_c**, **VCroot_d**: Parameters of the root xylem vulnerability curve.
- `xylem_kmax`: Sapwood-specific hydraulic conductivity of stem xylem (in kg H2O·s⁻¹·m⁻²).
- `VCstem_kmax`: Maximum stem xylem conductance values of each cohort.
- `VCstem_c`, `VCstem_d`: Parameters of the stem xylem vulnerability curve.

**Transpiration**: Plant cohort transpiration of the current day (mm of water = L/m²; filled with zeroes before simulations).

**Photosynthesis**: Plant cohort photosynthesis of the current day (g C/m²; filled with zeroes before simulations).

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 columns:
  - `SLA`: Specific leaf area (mm²/mg = m²/kg).
  - `A12As`: Leaf area to sapwood area ratio (in m²·m⁻²).

**Author(s)**

Miquel De Cáceres Ainsa, CTFC

**See Also**

`spwb`, `soil`, `forest`, `SpParamsMED`, `defaultSoilParams`, `plant_ID`

**Examples**

```R
# 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())

# Belowground parameters
below = forest2belowground(exampleforestMED, examplesoil, SpParamsMED)
below

# Initialize control parameters
control = defaultControl()
```
# Prepare spwb input
spwbInput(above, below, 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 complex transpiration mode
control$transpirationMode = "Sperry"
forest2spwbInput(exampleforestMED, examplesoil, SpParamsMED, control)

---

spwb_day

**Soil-plant water balance for a single day**

### Description

Function `spwb_day` performs water balance for a single day.

### Usage

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

### Arguments

- **x**: An object of class `spwbInput`.
- **soil**: A list containing the description of the soil (see `soil`).
- **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.
Value

An object (a list) of class `spwb_day` with the following elements:

- "cohorts": A data frame with cohort information, copied from `spwbInput`.
- "EnergyBalance": When using the 'Sperry' transpiration mode, the model performs energy balance of the stand and 'EnergyBalance' is returned (see `transp_transpirationSperry`).
- "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`).
- "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.
- "RhizoPsi": Minimum water potential (in MPa) inside roots, after crossing rhizosphere, per cohort and soil layer.
- "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`.
- "WindExtinction": A numeric vector with the wind speed (in m/s) for each plant cohort.

Note

Objects x and soil are modified during the simulation. Daily transpiration and photosynthesis values are stored in columns Transpiration and Photosynthesis of object x. Water content relative to field capacity (vector W) of soil is also modified.

Author(s)

Miquel De Cáceres Ainsa, CTFC

References

See Also

spwbInput, spwb, plot.spwb_day, spwb_ldrOptimization, forest

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()

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

# Simulate one day only
x1 = forest2spwbInput(exampleforestMED, examplesoil, SpParamsMED, control)
d = 100
sd1 <- spwb_day(x1, examplesoil, 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])

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

# Switch to 'Sperry' transpiration mode
control$transpirationMode="Sperry"

# Simulate one day only
x2 = forest2spwbInput(exampleforestMED, examplesoil2, SpParamsMED, control)
d = 100
sd2 <- spwb_day(x2, examplesoil2, 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.spwb.day()')
plot(sd2)
spwb_ldrCalibration  Calibration of root distribution

Description

The function spwb_ldrCalibration calibrates the species root distribution within spwb, given the arguments x, meteo, soil, 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, soil, 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.
soil  A list containing the description of the soil (see soil).
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. Useful 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 parameters 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, Centre Tecnologic Forestal de Catalunya
Antoine Cabon, Centre Tecnologic Forestal de Catalunya
Miquel De Cáceres Ainsa, CTFC

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, soil and psi_crit.

Usage
spwb_ldrExploration(x, soil, 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`.
- **soil**: A list containing the description of the soil (see `soil`).
- **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. Useful 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 equal 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 (as defined in the argument `soil`) 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, CTFC
Miquel De Cáceres Ainsa, CTFC

References


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()

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

Run exploration
y = spwb_lkrExploration(x = x, soil = examplesoil, meteo = examplemeteo, elevation = 100)

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

## End(Not run)

**spwb_maximumTranspirationRatioPlot**

*Maximum transpiration vs. LAI*

**Description**

Builds a curve of maximum transpiration (Tmax) over potential evapotranspiration (ETP) for increasing leaf area index (LAI) values.

**Usage**

```r
spwb_maximumTranspirationRatioPlot(x, soil, meteo, latitude, elevation, slope, aspect, 
n days = 100, 
LAI_seq = c(0.1, 0.25, seq(0.5, 10, by = 0.5))
```
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.
- **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.
- **ndays**: Number of days to subset for curve building.
- **LAI_seq**: Sequence of stand LAI values to be tested.

Details

TO BE DONE

Author(s)

Miquel De Cáceres Ainsa, CTFC

See Also

- `spwb`

Examples

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

#Initialize control parameters
control = defaultControl()

#Switch to 'Sperry' transpiration mode
control$transpirationMode="Sperry"

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

#Run optimization
```
spwb_maximumTranspirationRatioPlot(x2, examplesoil2, examplemeteo, 
   41.82592, elevation = 100, 
   slope = 0, aspect = 0, ndays = 20)

## End(Not run)

---

**spwb_resistances**

Soil-plant resistances

**Description**

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

**Usage**

```r
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, CTFC

**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 = NA,  
    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 (if NA parameter values of all cohort are modified).
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 below$V, below$VCroot_kmax and below$VGrhizo_kmax.
- Modifying LAI_live also affects LAI_expanded.
- Modifying VCroot_kmax from paramsTransp affects both paramsTransp$VCroot_kmax and below$VCroot_kmax.
- Modifying WaterStorage affects simultaneously paramsWaterStorage$Vleaf and paramsWaterStorage$Vsapwood.
- Modifying c affects simultaneously paramsTransp$VCleaf_c, paramsTransp$VCstem_c and paramsTransp$VCroot_c.
- Modifying d affects simultaneously paramsTransp$VCleaf_d, paramsTransp$VCstem_d and paramsTransp$VCroot_d.
- Modifying Plant_kmax from paramsTransp affects paramsTransp$VCleaf_kmax, paramsTransp$VCstem_kmax, paramsTransp$VCroot_kmax and below$VCroot_kmax.
• Modifying Al2As from paramsAnatomy affects paramsWaterStorage$Vsapwood, paramsTransp$VCstem_kmax, paramsTransp$VCroot_kmax and below$VCroot_kmax.
• Setting paramName = "Vmax298/Jmax298" affects both paramsTransp$Vmax298 and paramsTransp$Jmax298.

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, CTFC

See Also
spwb, summary.spwb

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).
spwb_validation

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, CTFC

References


See Also

spwb, summary.spwb

Description

Compares model predictions against observed values

Usage

spwb_validation(x, measuredData, type = "SWC", cohort = NULL, draw = TRUE, plotType = "dynamics")

Arguments

x
measuredData
type
cohort
draw
plotType

An object of class spwb or pwb.
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).
A string with the kind of model output to be evaluated. Accepted values are "SWC" (soil moisture content), "ETR" (total evapotranspiration), "E" (transpiration per leaf area) and "WP" (plant water potentials).
A string of the cohort to be compared (e.g. "T1_68"). If NULL results for the first cohort will be evaluated.
A boolean flag to indicate that plots should be produced.
Plot type to draw, either "dynamics" or "scatter".
Details

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

- "SWC": A column named "SWC" should be present, containing soil moisture content in percent volume.
- "ETR": A column named "ETR" should be present, containing stand’s evapotranspiration in mm/day. Observed values will be compared against modelled evapotranspiration (i.e. sum of transpiration, soil evaporation and interception loss) as well as 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 mm/day on a leaf area basis.
- "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.

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.

Value

A ggplot or evaluation statistics (a vector or a data frame depending on type):

- MAE: Mean absolute error.
- Bias: Mean deviation (positive values correspond to model overestimations).
- R2: Squared linear correlation coefficient.

Author(s)

Miquel De Cáceres Ainsa, CTFC

See Also

spwb

Description

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

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

Arguments

x
An object of class spwb or pwb.

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) photosynthesis over plant transpiration
- "Stand An/E": Stand 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 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 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, CTFC

See Also

spwb
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, CTFC

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)

Hydraulic supply functions

Description

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

Usage

```r
hydraulics_EXylem(psiPlant, psiUpstream,
kxylemmmax, 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, kxylemmmax, c, d, pCrit = 0.001)
hydraulics_E2psiXylem(E, psiUpstream,
kxylemmmax, c, d, psiCav = 0)
hydraulics_E2psiXylemUp(E, psiDownstream,
kxylemmmax, c, d, psiCav = 0)
hydraulics_E2psiVanGenuchten(E, psiSoil, krhizomax, n, alpha,
psiStep = -0.0001, psiMax = -10.0)
hydraulics_E2psiTwoElements(E, psiSoil, krhizomax, kxylemmmax, 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,
supplyfunctions

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)

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 whether 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.
supplyfunctions

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_supplyFunctionNetwork`**: 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_supplyFunctionNetwork` 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, CTFC.
References


See Also

`hydraulics_psi2K, hydraulics_maximumStemHydraulicConductance, spwb, soil`

Examples

```r
kstemmax = 4 # in mmol·m^-2·s^-1·MPa^-1
stcm = 3
stmd = -4 # in MPa

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

# Vulnerability curve
kstem = unlist(lapply(psiVec, hydraulics_xylemConductance, kstemmax, stcm, stmd))
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_tissueRWC(psiSym, pi0, epsilon, psiApo, c, d, af, femb = 0)`
- `moisture_tissueFMC(RWC, density, d0 = 1.54)`
- `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.
- density: Tissue density (g·cm⁻¹).
- d0: Matric density (g·cm⁻¹).

Details

Details of the tissue moisture calculations are given in a vignette.

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_segmentRWC`: Segment relative water content [0-1].

Author(s)

Miquel De Cáceres Ainsa, CTFC

References


transpiration

See Also

hydraulics_psi2K, hydraulics_supplyFunctionPlot, spwb, soil

Examples

```r
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")
```

transpiration  Transpiration submodel functions

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

```r
transp_profitMaximization(supplyFunction, photosynthesisFunction, Gwmin, Gwmax, gainModifier = 1.0, costModifier = 1.0)
transp_transpirationGranier(x, soil, meteo, day, modifyInputX = TRUE, modifyInputSoil = TRUE)
transp_transpirationSperry(x, soil, meteo, day, latitude, elevation, slope, aspect, canopyEvaporation = 0, snowMelt = 0, soilEvaporation = 0, stepFunctions = NA, modifyInputX = TRUE, modifyInputSoil = TRUE)
transp_stomatalRegulationPlot(x, soil, 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()`.
- `Gwmin, Gwmax`: Minimum and maximum stomatal conductance (mol·m⁻²·s⁻¹).
- `gainModifier, costModifier`: Modifiers (exponents) of the gain and cost functions defined in Sperry et al. (2016).
- `x`: An object of class `spwbInput` built using the 'Sperry' transpiration mode.
soil  An object of class soil.

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).

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.

type  A string with plot type, either "E" (transpiration flow), "An" (net photosynthesis), "Gw" (stomatal conductance), "T" (temperature) or "VPD" (vapour pressure deficit).

modifyInputX, modifyInputSoil  Boolean flags to indicate that the input x and soil objects are allowed to be modified during the simulation.

Details

Details of the transpiration submodel are given in a vignette.

Value

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

- "cohorts": A data frame with cohort information, copied from spwbInput.
- "Plants": A data frame of results for each plant cohort:
  - "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.
- "Photosynthesis": Net 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:
- "Extraction": Water extracted from the soil (in mm) for each cohort.
- "RootPsi": Minimum water potential (in MPa) at the root collar.
- "StemPsi": Minimum water potential (in MPa) at the 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.
- "StemPLC": Proportion of conductance loss in stem.
- "StemRWC": Relative water content of symplastic stem tissue.
- "LeafRWC": Relative water content of symplastic leaf tissue.
- "dEdP": Overall soil-plant conductance (derivative of the supply function).

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.
- "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.
  - "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:
transpiration

- "LAI": Leaf area index of (sunlit or shade) leaves of the plant cohort.
- "Abs_SWR": A data frame with instantaneous absorbed short-wave radiation (SWR).
- "Abs_LWR": A data frame with instantaneous absorbed 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 mmol/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).
- "dEdPinst": A data frame with the slope of the plant supply function (an estimation of whole-plant conductance).
- "PsiRoot": A data frame with root crown water potential (in MPa) for each plant cohort during each time step.
- "PsiStem": A data frame with stem water potential (in MPa) for each plant cohort during each time step.
- "PsiLeaf": A data frame with leaf (average) water potential (in MPa) for each plant cohort during each time step.
- "PLCstem": A data frame with the proportion loss of conductance [0-1] for each plant cohort during each time step.
- "RWCstem": A data frame with the (average) relative water content of symplastic stem tissue [0-1] for each plant cohort during each time step.
- "RWCleaf": 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_instantaneousLightExtinctionAbsortion`.
- "WindExtinction": A numeric vector with the wind speed (in m/s) for each plant cohort.
- "SupplyFunctions": If `stepFunctions` is not missing, a list of supply functions, photosynthesis functions and profit maximization functions.

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, CTFC

**References**

See Also

`hydraulics_supplyFunctionNetwork`, `biophysics_leafTemperature`, `photo_photosynthesis`, `spwb`

Examples

```r
# 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()

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

# Transpiration according to Granier's model, plant water potential and plant stress for a given day
transp_transpirationGranier(x, examplesoil, examplemeteo, 1, modifyInputX = FALSE, modifyInputSoil = FALSE)

# Switch to 'Sperry' transpiration mode
control$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, examplesoil2, examplemeteo, day = 100, timestep = 12, latitude = 41.82592, elevation = 100, type = "VPD")
```

Description

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

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, wind20H, 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.
wind20H The value of measured wind speed at 6 m = 20ft (in m/s).
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, CTFC
See Also

forest

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, 20)
Index

*Topic datasets
  exampleforest, 10
  examplemeteo, 10
  SFM_metric, 49
  SpParams, 61
biophysics, 3
biophysics_leafTemperature, 36, 100
biophysics_leafTemperature (biophysics), 3
biophysics_radiationDiurnalPattern (biophysics), 3
biophysics TEMPERATUREDIURNALPATTERN (biophysics), 3
conductancefunctions, 4
cut.Date, 40, 84, 87
Date, 21, 64
defaultcontrol, 7, 9, 67, 70, 71, 97
defaultsoilparams, 9, 50, 51, 59, 72
emptyforest (forest), 15
exampleforest, 10
exampleforestMED, 16
exampleforestMED (exampleforest), 10
exampleforestUS (exampleforest), 10
examplemeteo, 10
fire behaviour, 11
fire_FCCS, 20
fire_FCCS (fire behaviour), 11
fire_Rothermel, 49, 50
fire_Rothermel (fire behaviour), 11
forest, 10, 15, 16–19, 24, 29, 36, 37, 60, 61, 68, 70, 72, 75, 88, 100–102
Forest manipulation, 16
forest2aboveground (spwbInput), 70
forest2belowground (spwbInput), 70
forest2growthInput (spwbInput), 70
forest2spwbInput (spwbInput), 70
forest_mapShrubTable (forest_mapWoodyTables), 17
forest_mapTreeTable (forest_mapWoodyTables), 17
forest_mapWoodyTables, 16, 17
forest_mergeTrees (Forest manipulation), 16
fuel_cohortFineFMC (fuel_properties), 18
fuel_FCCS, 12, 14
fuel_FCCS (fuel_properties), 18
fuel_properties, 18
fuel_stratification (fuel_properties), 18
fuel_windAdjustmentFactor (fuel_properties), 18
growth, 21, 38, 70
growthInput, 21, 24, 31
growthInput (spwbInput), 70
hydraulics_averagePsi (conductancefunctions), 4
hydraulics_averageRhizosphereResistancePercent (scalingconductance), 46
hydraulics_E2psiAboveground (supplyfunctions), 89
hydraulics_E2psiBelowground (supplyfunctions), 89
hydraulics_E2psiFineRootLeaf (supplyfunctions), 89
hydraulics_E2psiNetwork (supplyfunctions), 89
hydraulics_E2psiNetworkStem1 (supplyfunctions), 89
hydraulics_E2psiTwoElements (supplyfunctions), 89
hydraulics_E2psiVanGenuchten (supplyfunctions), 89
hydraulics_E2psiXylem (supplyfunctions), 89
hydraulics_E2psiXylemUp (supplyfunctions), 89
hydraulics_ECapacitance (supplyfunctions), 89
hydraulics_ECrit (supplyfunctions), 89
hydraulics_EVanGenuchten (supplyfunctions), 89
hydraulics_EXylem (supplyfunctions), 89
hydraulics_findRhizosphereMaximumConductance (scalingconductance), 46
hydraulics_K2Psi (conductancefunctions), 4
hydraulics_leafWaterCapacity (scalingconductance), 46
hydraulics_maximumRootHydraulicConductance (scalingconductance), 46
hydraulics_maximumSoilPlantConductance (scalingconductance), 46
hydraulics_maximumStemHydraulicConductance (scalingconductance), 46
hydraulics_maximumStemHydraulicConductance, 6, 94
hydraulics_maximumStemHydraulicConductance (scalingconductance), 46
hydraulics_psi2K, 48, 94, 96
hydraulics_psi2K (conductancefunctions), 4
hydraulics_psi2Weibull (conductancefunctions), 4
hydraulics_psiCrit (conductancefunctions), 4
hydraulics_referenceConductivityHeightFactor (scalingconductance), 46
hydraulics_regulatedPsiTwoElements (supplyfunctions), 89
hydraulics_regulatedPsiXylem (supplyfunctions), 89
hydraulics_soilPlantResistances, 82
hydraulics_soilPlantResistances (scalingconductance), 46
hydraulics_stemWaterCapacity (scalingconductance), 46
hydraulics_supplyFunctionAboveground (supplyfunctions), 89
hydraulics_supplyFunctionBelowground (supplyfunctions), 89
hydraulics_supplyFunctionFineRootLeaf (supplyfunctions), 89
hydraulics_supplyFunctionNetwork, 36, 96, 100
hydraulics_supplyFunctionNetwork (supplyfunctions), 89
hydraulics_supplyFunctionNetworkStem1 (supplyfunctions), 89
hydraulics_supplyFunctionOneXylem (supplyfunctions), 89
hydraulics_supplyFunctionPlot, 6, 48, 96
hydraulics_supplyFunctionPlot (supplyfunctions), 89
hydraulics_supplyFunctionThreeElements (supplyfunctions), 89
hydraulics_supplyFunctionTwoElements (supplyfunctions), 89
hydraulics_taperFactorSavage (scalingconductance), 46
hydraulics_terminalConduitRadius (scalingconductance), 46
hydraulics_vanGenuchtenConductance (conductancefunctions), 4
hydraulics_vulnerabilityCurvePlot (conductancefunctions), 4
hydraulics_xylemConductance (conductancefunctions), 4
hydraulics_xylemPsi (conductancefunctions), 4
hydrology_erFactor (hydrology_rainInterception), 24
hydrology_infiltrationAmount (soil hydrology), 52
hydrology_infiltrationRepartition (soil hydrology), 52
hydrology_interceptionPlot (hydrology_rainInterception), 24
hydrology_rainInterception, 24, 28
hydrology_snowMelt (soil hydrology), 52
hydrology_soilEvaporation, 28
hydrology_soilEvaporation (soil hydrology), 52
hydrology_soilEvaporationAmount (soil hydrology), 52
hydrology_soilInfiltrationPercolation (hydrology_soilWaterInputs), 26
hydrology_soilWaterInputs, 26, 53
light, 28
light_cohortAbsorbedSWRFraction
  (light), 28
light_cohortSunlitShadeAbsorbedRadiation
  (light), 28
light_instantaneousLightExtinctionAbsortion,
  74, 99
light_instantaneousLightExtinctionAbsortion
  (light), 28
light_layerIrradianceFraction (light), 28
light_layerIrradianceFractionBottomUp
  (light), 28
light_instantaneousLightExtinctionAbsortion
  (light), 28
modifyCohortParams (modifySpParams), 31
modifySpParams, 31
moisture_apoplasticPsi
  (tissuemoisture), 94
moisture_apoplasticRWC
  (tissuemoisture), 94
moisture_pressureVolumeCurvePlot
  (tissuemoisture), 94
moisture_symplasticPsi
  (tissuemoisture), 94
moisture_symplasticRWC
  (tissuemoisture), 94
moisture_tissueFMC (tissuemoisture), 94
moisture_tissueRWC (tissuemoisture), 94
pheno_leafDevelopmentStatus
  (pheno_updateLeaves), 32
pheno_updateLeaves, 32
plot, 33
photo, 33
photo_electronLimitedPhotosynthesis
  (photo), 33
photo_GammaTemp (photo), 33
photo_KmTemp (photo), 33
photo_leafPhotosynthesisFunction
  (photo), 33
photo_multilayerPhotosynthesisFunction
  (photo), 33
photo_photosynthesis, 100
photo_photosynthesis (photo), 33
photo_rubiscoLimitedPhotosynthesis
  (photo), 33
photo_sunshadePhotosynthesisFunction
  (photo), 33
photo_VmaxTemp (photo), 33
Plant values, 36
plant_basalArea, 61, 88
plant_basalArea (Plant values), 36
plant_characterParameter (Plant values), 36
plant_cover (Plant values), 36
plant_crownBaseHeight (Plant values), 36
plant_crownLength (Plant values), 36
plant_crownRatio (Plant values), 36
plant_density, 71
plant_density (Plant values), 36
plant_equilibriumLeafLitter (Plant values), 36
plant_equilibriumSmallBranchLitter
  (Plant values), 36
plant_foliarBiomass (Plant values), 36
plant_fuel (Plant values), 36
plant_height (Plant values), 36
plant_ID, 71, 72
plant_ID (Plant values), 36
plant_LAII (Plant values), 36
plant_equilibriumSmallBranchLitter
  (Plant values), 36
plant_phytovolume (Plant values), 36
plant_species (Plant values), 36
plant_speciesName (Plant values), 36
plot_growth (plot.spwb), 38
plot_pwb (plot.spwb), 38
plot_pwb_day (plot.spwb_day), 41
plot_spwb, 38, 43, 68, 82
plot_spwb_day, 41, 75
print, 16
print.soil (soil), 50
print.summary.forest (forest), 15
pwb, 38, 70, 85
pwb (spwb), 64
root, 44
root_conicDistribution (root), 44
root_ldrvDistribution, 76, 77, 79
root_ldrvDistribution (root), 44
root_rhizosphereOverlapProportions
  (root), 44
root_rootLengths (root), 44
root_xylemConductanceProportions (root), 44
sapply, 83, 84
scalingconductance, 46
SFM_metric, 49
soil, 5, 6, 9, 21, 22, 27, 45, 48, 50, 52, 56–59, 64, 66, 70, 72, 73, 76–79, 81, 83, 92, 94, 96, 97
soil hydrology, 52
soil texture and hydraulics, 54
soil thermodynamics, 57
soil_psi (soil texture and hydraulics), 54
soil_psi2thetaSX, 51
soil_psi2thetaSX (soil texture and hydraulics), 54
soil_psi2thetaVG, 51
soil_psi2thetaVG (soil texture and hydraulics), 54
soil_retentionCurvePlot (soil texture and hydraulics), 54
soil_rockWeight2Volume (soil texture and hydraulics), 54
soil_temperatureChange (soil thermodynamics), 57
soil_temperatureGradient (soil thermodynamics), 57
soil_thermalCapacity (soil thermodynamics), 57
soil_thermalConductivity (soil thermodynamics), 57
soil_theta (soil texture and hydraulics), 54
soil_theta2psiSX (soil texture and hydraulics), 54
soil_theta2psiVG (soil texture and hydraulics), 54
soil_thetaFC (soil texture and hydraulics), 54
soil_thetaSAT (soil texture and hydraulics), 54
soil_thetaSATSX (soil texture and hydraulics), 54
soil_thetaWP (soil texture and hydraulics), 54
soil_USDAType (soil texture and hydraulics), 54
soil_vanGenuchtenParamsCarsel (soil texture and hydraulics), 54
soil_vanGenuchtenParamsToth (soil texture and hydraulics), 54
soil_water (soil texture and hydraulics), 54
soil_waterExtractable (soil texture and hydraulics), 54
soil_waterFC (soil texture and hydraulics), 54
soil_waterSAT (soil texture and hydraulics), 54
soil_waterTableDepth (soil texture and hydraulics), 54
soil_waterWP (soil texture and hydraulics), 54
soilgridsParams, 9, 58
Species values, 60
species_basalArea (Species values), 60
species_cover (Species values), 60
species_density (Species values), 60
species_foliarBiomass (Species values), 60
species_fuel (Species values), 60
species_LAI (Species values), 60
species_phytovolume (Species values), 60
SpParams, 61
SpParamsMED, 9, 15, 18, 19, 25, 29, 31, 32, 36, 60, 70, 72, 88, 101
SpParamsMED (SpParams), 61
SpParamsUS (SpParams), 61
spwb, 4, 6–8, 10, 11, 17, 19, 20, 26, 30, 33, 36–38, 41, 43, 45, 48, 51, 53, 61, 64, 64, 70, 72, 74–79, 81–87, 94, 96, 100
spwb_day, 7, 28, 42, 43, 67, 68, 73
spwb_ldrCalibration, 76
spwb_ldrExploration (spwb_ldrOptimization), 77
spwb_ldrOptimization, 45, 68, 75, 77, 77
spwb_maximumTranspirationRatioPlot, 80
spwb_resetInputs (spwb), 64
spwb_resistances, 82
spwb_sensitivity, 83
spwb_stress, 84
spwb_validation, 85
spwb_waterUseEfficiency, 86
spwbInput, 5, 7, 8, 19, 22, 25, 31, 32, 64, 65, 68, 70, 73–76, 78, 81, 83, 92, 95–97
Stand values, 88
stand_basalArea (Stand values), 88
stand_foliarBiomass (Stand values), 88
stand_fuel (Stand values), 88
INDEX

stand_LAI (Stand values), 88
stand_phytovolume (Stand values), 88
summary, 16
summary.forest, 17, 37, 61, 88
summary.forest (forest), 15
summary.growth (plot.spwb), 38
summary.pwb (plot.spwb), 38
summary.spwb, 84, 85
summary.spwb (plot.spwb), 38
supplyfunctions, 89
tissuemoisture, 94
transp_profitMaximization
  (transpiration), 96
transp_stomatalRegulationPlot
  (transpiration), 96
transp_transpirationGranier, 65, 74
transp_transpirationGranier
  (transpiration), 96
transp_transpirationSperry, 42, 65, 74
transp_transpirationSperry
  (transpiration), 96
transpiration, 96
Vertical profiles, 100
vprofile_fuelBulkDensity (Vertical profiles), 100
vprofile_leafAreaDensity (Vertical profiles), 100
vprofile_PARExtinction (Vertical profiles), 100
vprofile_rootDistribution (Vertical profiles), 100
vprofile_SWRExtinction (Vertical profiles), 100
vprofile_windExtinction (Vertical profiles), 100