Package 'medfate'

November 11, 2016

Type Package

Title Mediterranean Forest Simulation

Version 0.2.2

Date 2016-11-10

Author Miquel De Caceres

Maintainer Miquel De Caceres <miquelcaceres@gmail.com>

Description Functions to simulate forest dynamics using cohort-based description of vegetation.

License GPL (>= 2)

LazyLoad yes

Depends R (>= 2.10), meteoland, sp, spdep

Imports methods, Rcpp (>= 0.11.0)

LinkingTo Rcpp

NeedsCompilation yes

Repository CRAN

Date/Publication 2016-11-11 00:19:41

R topics documented:

`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 (=FALSE)`: Boolean flag to indicate console output during calculations.
- `petMode (="Input")`: Potential evapotranspiration mode. See `swb`.
- `hydraulicMode (="Input")`: Hydraulic model. See `swb`.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

`defaultSoilParams`, `SpParamsMED`, `swb`, `swbpoints`

---

defaultSoilParams — *Default soil parameters*

Description

Creates a list with default soil parameters for model functions.

Usage

```
defaultSoilParams()
```

Details

The function returns a list with default parameters. Users can change those that need to be set to other values and use the list as input for function `soil`.

Value

A list with the following elements (and default values):

- `SoilDepth (= 1000)`: Soil depth (accounting for the sum of topsoil and subsoil).
- `RockLayerDepth (= 4000)`: Depth of the rock layer (i.e. total depth).
- `TS_clay (= 25)`: Topsoil clay percentage.
- `TS_silt (= 50)`: Topsoil silt percentage.
- `TS_sand (= 25)`: Topsoil sand percentage.
- `TS_macro (= 0.1)`: Topsoil macroporosity.
- `TS_rfc (= 20)`: Topsoil percentage of rock fragment content.
- `SS_clay (= 25)`: Subsoil clay percentage.
- `SS_silt (= 50)`: Subsoil silt percentage.
- `SS_sand (= 25)`: Subsoil sand percentage.
- `SS_macro (= 0.1)`: Subsoil macroporosity.
- `SS_rfc (= 40)`: Subsoil percentage of rock fragment content.
- `RL_clay (= 25)`: Rock layer clay percentage.
- `RL_silt (= 50)`: Rock layer silt percentage.
- `RL_sand (= 25)`: Rock layer sand percentage.
- `RL_macro (= 0.1)`: Rock layer macroporosity.
- `RL_rfc (= 85)`: Rock layer percentage of rock fragment content.
- `Gsoil (= 0.5)`: Gamma parameter for infiltration.
- `Ksoil (= 0.05)`: Kappa parameter for infiltration.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

`soil`, `defaultControl`, `SpParamsMED`

Examples

```
defaultSoilParams()
```

---

exampleforest *Example forest stand*

Description

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

Usage

```
data(exampleforest)
```

Format

An object of class `forest` containing the description of the tree, sapling and shrub cohorts of a forest patch as well as the seed bank and the size of the patch.

Source

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

See Also

`forest`, `swb`, `forest2swbInput`

Examples

```
data(exampleforest)
```

`examplemeteo` *Example daily meteorology data*

Description

Example data set of meteorological input.

Usage

```
data(examplemeteo)
```

Format

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

- `DOY` Day of the year (Julian day).
- `MeanTemperature` Mean daily temperature (in degrees Celsius).
- `MinTemperature` Minimum daily temperature (in degrees Celsius).
- `MaxTemperature` Maximum daily temperature (in degrees Celsius).
- `Precipitation` Daily precipitation (in mm of water).
- `MeanRelativeHumidity` Mean daily relative humidity (in percent).
- `MinRelativeHumidity` Minimum daily relative humidity (in percent).
- `MaxRelativeHumidity` Maximum daily relative humidity (in percent).
- `Radiation` Incoming radiation (in MJ/m2).
- `WindSpeed` Wind speed (in m/s).
- `WindDirection` Wind direction (in degrees from North).
- `PET` Potential evapo-transpiration (in mm of water).

Source

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

See Also

`swb`

Examples

```
data(examplemeteo)
```

exampleSPF *Example of spatial points with forest data*

Description

An example of an object of SpatialPointsForest-class with data for 30 plots, taken from the Spanish Forest Inventory (DGCN 2005).

Usage

```
data("exampleSPF")
```

Format

The data format is that of an object SpatialPointsForest-class

Source

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

See Also

forest, exampleforest, SpatialPointsForest-class

Examples

```
data(exampleSPF)

#Plot forest coordinates
plot(exampleSPF)

#Inspect forest object corresponding to the first point
exampleSPF@forestlist[[1]]
```

extractSFIforest *Extract forest from SFI data*

Description

Creates a forest object from Spanish Forest Inventory (SFI) data (DGCN 2005).

Usage

```
extractSFIforest(SFItreeData, SFIshrubData, ID, SpParams,
                 SFIherbData = NULL, SFIcodes=NULL,
                 patchsize= 10000, setDefaults=TRUE)
translateSpeciesCodes(x, SFIcodes)
```

Arguments

`SFItreeData`	A data frame with measured tree data.
`SFIshrubData`	A data frame with measured shrub data.
`ID`	A string with the ID of the plot to be extracted.
`SpParams`	A data frame with species parameters (see details).
`SFIcodes`	A string vector (of length equal to the number of rows in `SpParams` of the SFI species codes that correspond to the model species codification. Each string may contain different coma-separated codes in order to merge SFI species into a single model species.
`SFIherbData`	A data frame with cover and mean height of the herb layer.
`patchsize`	The area of the forest stand, in square meters.
`setDefaults`	Initializes default values for missing fields in SFI data.
`x`	A data frame with a column called 'Especie'.

Details

SFI data needs to be in a specific format. Function `extractSFIforest` calls `translateSpeciesCodes` internally.

Value

An object of class `forest`.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

See Also

`SpatialPointsForest`, `forest`

fire.behaviour *Fire behaviour functions*

Description

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

Usage

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

Arguments

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

Details

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

Value

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

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

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

Note

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

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

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

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

See Also

`fuel.FCCS`

Examples

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

#Default species parameterization
data(SpParamsMED)

#Calculate fuel properties according to FCCS
fccs = fuel.FCCS(exampleforest, 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(exampleforest, 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)
```

`forest` *Forest description*

Description

Description of a forest patch.

Usage

```
## S3 method for class 'forest'
summary(object, SpParams, detailed=FALSE, ...)
## S3 method for class 'summary.forest'
print(x, digits = getOption("digits"), ...)
emptyforest(ID="", patchsize=10000)
```

Arguments

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

- `ID`: An identifier of the forest stand (a string).
- `patchsize`: The area of the forest stand, 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 roots.
  - `Z95`: Depth (in mm) corresponding to 95% of 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.
  - Z: Depth (in mm) of root system.
- 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).

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.

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 data frame 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, Centre Tecnologic Forestal de Catalunya

References

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

See Also

exampleforest, extractSFIforest, soil, SpatialPointsForest

Examples

```
data(exampleforest)
data(SpParamsMED)

summary(exampleforest, SpParamsMED)
```

Forest values *Forest description functions*

Description

Functions to calculate overall attributes of a forest object.

Usage

```
forest.BasalArea(x)
```

Arguments

x	An object of class forest.

Value

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

- forest.BasalArea: Total basal area (m2/ha).

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

swb, forest, plant.BasalArea, summary.forest

Examples

```
#Default species parameterization
data(SpParamsMED)

#Load example plot
data(exampleforest)

#A short way to obtain total basal area
forest.BasalArea(exampleforest)
```

forest2swbInput *Input variables for soil water balance*

Description

Function `forest2swbInput` takes an object of class `forest` to calculate input variables for `swb`. Function `swbInput` does the same from input vectors.

Usage

```
forest2swbInput(x, SpParams, d, gdd = NA, petMode = "Input", hydraulicMode = "Simple")
swbInput(SP, LAI, H, CR,  V, SpParams,  petMode = "Input", hydraulicMode = "Simple")
```

Arguments

`x`	An object of class `forest`.
`SpParams`	A data frame with species parameters (see `SpParamsMED` and `SpParamsMED`).
`d`	A vector containing the depth of each soil layer.
`gdd`	Growth degree days to account for leaf phenology effects (in Celsius).
`petMode`	Potential evapotranspiration mode (either 'Input' or 'PenmanMonteith').
`hydraulicMode`	Hydraulic model (either 'Simple' or 'Sperry').
`SP`	An integer vector of plant cohort species identity (first species is 0).
`LAI`	A numeric vector (same length as `SP`) with plant cohort leaf area index (one-side leaf area relative to plot area).
`H`	A numeric vector (same length as `SP`) with plant cohort total heights (in cm).
`CR`	A numeric vector (same length as `SP`) with plant cohort crown ratio values (between 0 and 1).
`V`	A numeric matrix (with three columns and as many rows as the length as `SP`) containing the proportion of roots of each plant in each soil layer.

Details

Function `forest2swbInput` extracts height and species identity from plant cohorts of `x`. For each plant cohort also calculates leaf area index and the distribution of fine roots across soil. Both `forest2swbInput` and `swbInput` find parameter values for each plant cohort according to the parameters of its species as specified in `SpParams`.

Value

A data frame with the following columns (additional columns will difer depending on the value of `hydraulicMode` and `petMode`):

- `LAI`: Leaf area index (in m2/m2).
- `SP`: Species identity (an integer).

- H: Plant height (in cm).
- V.1, V.2, V.3: The proportion of fine roots in each layer.
- RC_min: Minimum canopy (stomatal) resistance (in s/m) (for petMode = "PenmanMonteith").
- psiExtr: Water potential corresponding to 50% relative conductance (in kPa) (for hydraulicMode = "Simple").
- VC_kxmax: Maximum hydraulic conductance (in L/m2/day) (for hydraulicMode = "Sperry").
- VC_c, VC_d: Parameters of the vulnerability curve (for hydraulicMode = "Sperry").
- k: PAR extinction coefficient.
- g: Canopy water retention capacity per LAI unit (mm/LAI).
- CR: Crown ratio (crown length to total height).
- Sgdd: Growth degree days needed for leaf budburst (for winter decideous species).
- Transpiration: Plant cohort transpiration (filled with zeroes before simulations).

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

swb, soil, forest, SpParamsMED, defaultSoilParams

Examples

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

#Default species parameterization
data(SpParamsMED)

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

#Prepare input
swbInput = forest2swbInput(exampleforest,SpParamsMED, s$dVec)
```

---

fuel.properties *Fuel stratification and fuel characteristics*

Description

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

Usage

```
fuel.Stratification(object, SpParams, gdd = NA,
                      heightProfileStep = 10.0, maxHeightProfile = 5000.0,
                      bulkDensityThreshold = 0.05)
fuel.FCCS(object, ShrubCover, CanopyCover, SpParams,
          gdd = NA, heightProfileStep = 10, maxHeightProfile = 5000,
          bulkDensityThreshold = 0.05)
fuel.WindAdjustmentFactor(topShrubHeight, bottomCanopyHeight, topCanopyHeight,
                          canopyCover)
```

Arguments

`object`	An object of class `link{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`).
`gdd`	Growth degree-days.
`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.
`topShrubHeight`	Shrub stratum top height (in m).
`bottomCanopyHeight`	Canopy base height (in m).
`topCanopyHeight`	Canopy top height (in m).
`canopyCover`	Canopy percent cover.

Details

Details are described in a vignette.

Value

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

- `w`: Fine fuel loading (in kg/m2).
- `cover`: Percent cover.
- `hbc`: Height to base of crowns (in m).
- `htc`: Height to top of crowns (in m).
- `delta`: Fuel depth (in m).

- `rhob`: Fuel bulk density (in kg/m3).
- `rhop`: Fuel particle density (in kg/m3).
- `PV`: Particle volume (in m3/m2).
- `beta`: Packing ratio (unitless).
- `betarel`: Relative packing ratio (unitless).
- `etabetarel`: Reaction efficiency (unitless).
- `sigma`: Surface area-to-volume ratio (m2/m3).
- `pDead`: Proportion of dead fuels.
- `FAI`: Fuel area index (unitless).
- `h`: High heat content (in kJ/kg).
- `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.WindAdjustmentFactor` returns a value between 0 and 1.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

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

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

See Also

`fire.FCCS`

Examples

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

#Default species parameterization
data(SpParamsMED)

#Show stratification of fuels
fuel.Stratification(exampleforest, SpParamsMED)

#Calculate fuel properties according to FCCS
fccs = fuel.FCCS(exampleforest, 50,100, SpParamsMED)
fccs

fuel.WindAdjustmentFactor(fccs$htc[2], fccs$hbc[1], fccs$htc[1], fccs$cover[1])
```

`hydraulics` *Hydraulic model functions*

Description

Set of functions used in the implementation of Sperry and Love's (2015) hydraulic model.

Usage

```
hydraulics.E2psiXylem(E, psiUpstream, kxmax, c, d, psiStep = -0.01,
                  psiMax = -10.0)
hydraulics.E2psiVanGenuchten(E, psiSoil, ksmax, n, alpha,
                             psiStep = -0.01, psiMax = -10.0)
hydraulics.E2psiTwoElements(E, psiSoil, ksmax, kxmax, n, alpha, c, d,
                            psiStep = -0.001, psiMax = -10.0)
hydraulics.Ecrit(psiUpstream, kxmax, c, d)
hydraulics.EXylem(psiPlant, psiUpstream, kxmax, c, d)
hydraulics.supplyFunction(Emax, psiSoil, ksmax, kxmax, n, alpha, c, d,
                          dE = 0.1, psiMax = -10.0)
hydraulics.psiCrit(c, d)
hydraulics.regulatedPsiXylem(E, psiUpstream, kxmax, c, d, psiStep = -0.01)
hydraulics.regulatedPsiTwoElements(Emax, psiSoil, ksmax, kxmax, n, alpha,
                                   c, d, dE = 0.1, psiMax = -10.0)
hydraulics.vanGenuchtenConductance(psi, ksmax, n, alpha)
hydraulics.xylemConductance(psi, kxmax, c, d)
```

Arguments

`E`	Flow per surface unit.
`Emax`	Maximum flow per surface unit.
`psi`	Water potential (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.
`psiPlant`	Plant water potential (in MPa).
`psiSoil`	Soil water potential (in MPa).
`psiStep`	Water potential precision (in MPa).
`psiMax`	Minimum (maximum in absolute value) water potential to be considered (in MPa).
`dE`	Flow precision.
`kxmax`	Maximum xylem hydraulic conductance (defined as flow per surface unit and per pressure drop).
`ksmax`	Maximum rhizosphere hydraulic conductance (defined as flow per surface unit and per pressure drop).
`c, d`	Parameters of the Weibull function (plant vulnerability curve).
`n, alpha`	Parameters of the Van Genuchten function (rhizosphere vulnerability curve).

Details

Details of the hydraulic model are given in a vignette.

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.Ecrit`: The critical flow according to the xylem supply function and given an input soil water potential.
- `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.supplyFunction`: A list with different numeric vectors with information of the two-element supply function:
  - `E`: Flow values (supply values).
  - `FittedE`: Fitted flow values.
  - `PsiRoot`: Water potential inside the root values.
  - `PsiPlant`: Plant water potential values.
  - `dEdP`: Derivatives of the supply function.

- `hydraulics.psiCrit`: Critical water potential (in MPa) according to the xylem vulnerability curve.
- `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.
- `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).

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

See Also

swb

`Plant values` *Plant description functions*

Description

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

Usage

```
plant.BasalArea(x)
plant.LargerTreeBasalArea(x)
plant.CharacterParameter(x, SpParams, parName)
plant.Cover(x)
plant.CrownBaseHeight(x, SpParams)
plant.CrownLength(x, SpParams)
plant.CrownRatio(x, SpParams)
plant.Density(x)
plant.EquilibriumLeafLitter(x, SpParams, AET = 800)
plant.EquilibriumSmallBranchLitter(x, SpParams,
                                   smallBranchDecompositionRate = 0.81)
plant.FoliarBiomass(x, SpParams, gdd = NA)
plant.Fuel(x, SpParams, gdd = NA, includeDead = TRUE)
```

```
plant.Height(x)
plant.LAI(x, SpParams, gdd = NA)
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.
`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.

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`: Tree density (ind/ha).
- `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.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

swb, forest, summary.forest

Examples

```
#Default species parameterization
data(SpParamsMED)

#Load example plot
data(exampleforest)

#A short way to obtain total basal area
sum(plant.BasalArea(exampleforest), na.rm=TRUE)

#The same forest level function for LAI
sum(plant.LAI(exampleforest, SpParamsMED))

#The same forest level function for fuel loading
sum(plant.Fuel(exampleforest, SpParamsMED))

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

plant.SpeciesName(exampleforest, SpParamsMED)
```

---

plot.swb *Displays soil water balance results*

Description

Function `plot` plots the results of the soil water balance model (see swb), whereas function `summary` summarizes the model's output in different temporal steps (i.e. weekly, annual, ...).

Usage

```
## S3 method for class 'swb'
plot(x, yearAxis=FALSE, type="PET_Precipitation",
    xlim = NULL, ylim=NULL, xlab=NULL, ylab=NULL,...)
## S3 method for class 'swb'
summary(object, freq="years", output="DailyBalance", FUN=sum, ...)
```

Arguments

`x, object`	An object of class `swb`
`yearAxis`	A boolean to indicate whether the units of the x-axis are years (by default they are days).
`type`	The information to be plotted: • `"PET_NetPrec"`: Potential evapotranspiration and Net Precipitation. • `"Export"`: Water exported through deep drainage and surface runoff. • `"ET"`: Plant eranspiration and soil evaporation. • `"Psi"`: Soil water potential. • `"Theta"`: Soil relative water content. • `"Vol"`: Soil water volumetric content. • `"PlantStress"`: Plant cohort average daily drought stress.
`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"`, `"PlantTranspiration"` and `"SoilWaterBalance"`.
`FUN`	The function to summarize results (e.g., `sum`, `mean`, ...)
`...`	Additional parameters for functions `plot` or `summary`.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

See Also

`swb`, `swbpoints`

Examples

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

#Load example plot plant data
data(exampleforest)

#Default species parameterization
```

```
data(SpParamsMED)

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

#Prepare input
x = forest2swbInput(exampleforest,SpParamsMED, examplesoil$dVec)

#Call simulation function
S<-swb(x, examplesoil, examplemeteo)

#Plot results
plot(S)

#Monthly summary (averages) of soil water balance
summary(S, freq="months",FUN=mean, output="SoilWaterBalance")
```

---

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.

Usage

```
root.conicRS(Z, d)
root.ldrRS(Z50, Z95, d)
root.ldrProfile(Z50, Z95, d)
```

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.
Z	A vector of depths (in mm) corresponding to the root cone tip.
d	The width (in mm) corresponding to each soil layer.

Details

Function `conicRS` assumes a conic distribution of fine roots, whereas function `ldrRS` uses the linear dose response model of Schenck & Jackson (2002).

Value

Functions `conicRS` and `ldrRS` return a matrix with as many rows as elements in `Z` and three columns (corresponding to soil layers). Function `ldrProfile` returns a matrix with as many rows as elements in `Z90` and as many columns as elements in `d`. Values in all cases correspond to the proportion of fine roots in each soil layer.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

See Also

swb, forest2swbInput, soil

Examples

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

#Default species parameterization
data(SpParamsMED)

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

#Calculate conic root system for trees
V1 = root.conicRS(Z=rep(2000,nrow(exampleforest$treeData)), S$dVec)
print(V1)

#Calculate LDR root system for trees (Schenck & Jackson 2002)
V2 = root.ldrRS(Z50 = rep(200,nrow(exampleforest$treeData)),
          Z95 = rep(1000,nrow(exampleforest$treeData)), S$dVec)
print(V2)

#Plot LDR root systems (31 layers of 5 mm each)
P = root.ldrProfile(c(100,500), c(200,1500), rep(50, 31))
matplot(x=t(P), y=seq(0,1500, by=50), type="l", xlim=c(0,0.5), ylim=c(1500,0), ylab = "Depth (mm)",
        xlab="Root density")
legend("bottomright", legend=c("Z50 = 100, Z95 = 200",
       "Z50 = 200, Z95 = 1500"), col=c("black","red"), lty=1:2, bty="n")
```

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

Description

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

Usage

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

Format

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

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

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

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

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

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

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

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

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

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

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

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

`Fuel_Bed_Depth` Fuel bed depth [cm]

`Mx_dead` Dead fuel moisture of extinction [percent]

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

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

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

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

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

Source

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

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

See Also

`fire.Rothermel`

Examples

```
data(SFM_metric)
```

soil *Soil definition*

Description

Defines a soil object for its use in simulation functions.

Usage

```
soil(SoilParams, W = as.numeric(c(1,1,1)))
## S3 method for class 'soil'
print(x,  ...)
```

Arguments

SoilParams	A list of soil parameters (see defaultSoilParams.
W	A three-value numerical vector with the relative water content of the three soil layers (topsoil, subsoil and rocky layer).
x	An object of class soil.
...	Additional parameters to print.

Details

Function print prompts a description of soil characteristics and status (water content).

Value

An list of class soil with the following elements:

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

soil.psi2theta, swb, defaultSoilParams

Examples

```
s = soil(defaultSoilParams())
s
```

soil texture *Soil texture functions*

Description

Functions `soil.psi2theta` and `soil.theta2psi` calculate water potentials (psi) and relative water contents (theta) using texture data the formulae of Saxton (1986). Function `soil.USDAType` returns the USDA type for a given texture. Function `soil.vanGenuchtenParams` gives parameters for van Genuchten's conductance function for a given texture type (Leij et al. 1996).

Usage

```
soil.psi2theta(clay, sand, psi)
soil.theta2psi(clay, sand, theta)
soil.USDAType(clay, sand)
soil.vanGenuchtenParams(soilType)
```

Arguments

`clay`	Percentage of clay.
`sand`	Percentage of sand.
`psi`	Water potential (in MPa).
`theta`	Relative water content (in percent)
`soilType`	A string indicating the soil type.

Value

Function `soil.psi2theta` returns the soil water potential from soil relative water content, and the function `soil.theta2psi` does the reverse calculation. Function `soil.USDAType` returns a string. Function `soil.vanGenuchtenParams` returns a vector with three parameter values.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

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

See Also

`soil`

Examples

```
type = soil.USDAType(clay=10, sand=70)
soil.vanGenuchtenParams(type)

psi = seq(0, -6000, by=-100)
plot(psi, lapply(as.list(psi), FUN=soil.psi2theta, clay=70, sand=10),
     type="l", ylim=c(0,0.5), ylab="Theta")
lines(psi, lapply(as.list(psi), FUN=soil.psi2theta, clay=10, sand=70), lty=2)
```

`spatialForestSummary` *Forest and soil summaries over space*

Description

Functions to calculates a summary function for the forest or soil of all cells in a `SpatialPointsForest-class` or `SpatialGridForest-class`.

Usage

```
spatialForestSummary(object, summaryFunction, ...)
spatialSoilSummary(object, summaryFunction, ...)
```

Arguments

`object`	An object of class `SpatialPointsForest-class` or `SpatialGridForest-class`.
`summaryFunction`	A function that accepts objects of class `forest` or `soil`, respectively.
`...`	Additional arguments to the summary function.

Value

An object of class `SpatialPointsDataFrame` or `SpatialGridDataFrame`, depending on the input, containing the calculated statistics.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya.

See Also

`forest`, `soil`, `summary.forest`

Examples

```
#Load plot data
data(exampleSPF)

#Load species parameters
data(SpParamsMED)

#Apply summary function
y <- spatialForestSummary(exampleSPF,summary.forest, SpParamsMED)
head(y@data)

#Plot basal area
spplot(y["BA"])
```

`SpatialGridForest-class` *Class* `"SpatialGridForest"`

Description

An S4 class that represents a set of forest stands located over a grid of coordinates.

Objects from the Class

Objects can be created by calls of the form `new("SpatialGridForest", ...)`, or by calls to the function `SpatialGridForest`.

Slots

`lct`: A character vector of land cover type for each cell. Values allowed are: 'wildland', 'agriculture', 'rock' and 'static'.

`forestlist`: Object of class `"list"` containing a set of `forest` objects.

`soillist`: Object of class `"list"` containing a set of `soil` objects.

`data`: Object of class "data.frame" containing the elevation (in m), slope (in degrees) and aspect (in degrees from North) of every cell.

`grid`: Object of class `"GridTopology"`.

`bbox`: Object of class `"matrix"` with the boundary box.

`proj4string`: Object of class `"CRS"` with the projection string.

Extends

Class `"SpatialGridTopography"`, directly. Class `"SpatialGridDataFrame"`, distance 2. Class `"SpatialGrid"`, distance 3. Class `"Spatial"`, by class "SpatialGrid", distance 4.

Methods

getLCTs `signature(object = "SpatialGridForest")`: returns a SpatialGridDataFrame with the land cover types of the landscape cells.

spatialForestSummary `signature(object = "SpatialGridForest")`: calculates a summary function for all forest stands and returns an object of class `SpatialGridDataFrame-class`.

spatialSoilSummary `signature(object = "SpatialGridForest")`: calculates a summary function for the soil of all stands and returns an object of class `SpatialGridDataFrame-class`.

spplot `signature(object = "SpatialGridForest")`: allows plotting maps of the landscape state.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

`SpatialGridTopography-class`, `SpatialPointsForest-class`, `spatialForestSummary`, `forest`, `soil`

Examples

```
#Structure of the S4 object
showClass("SpatialGridForest")
```

`SpatialPointsForest` *Creates Spatial Forest Classes*

Description

Functions to create spatial forest classes from Spanish Forest Inventory (SFI) data (see details). Internally, these functions make calls to `extractSFIforest`.

Usage

```
SpatialPointsForest(SFItreeData, SFIshrubData, SFIherbData = NULL,
                    SpatialPointsIDs, elevation, slope, aspect, SpParams,
                SoilParamData = NULL, SFIcodes=NULL, control = defaultControl())
SpatialGridForest(landTopo, SFItreeData, SFIshrubData, SpatialPointsIDs,
                SpParams, SoilParamData = NULL, lctInput,
            forestLCTs, shrublandLCTs, grasslandLCTs, agricultureLCTs= numeric(0),
                rockLCTs= numeric(0), staticLCTs= numeric(0),
                SFIcodes=NULL, FMcodes = NULL, control = defaultControl())
```

Arguments

- `landTopo` An object of class `SpatialGridTopography`.
- `SFItreeData` A data frame with measured tree data.
- `SFIshrubData` A data frame with measured shrub data.
- `SFIherbData` A data frame with percent cover and mean height of the herb layer.
- `SpatialPointsIDs` An object of class `SpatialPoints-class` containing the coordinates of the forest plots. Coordinates must include row names corresponding to plot IDs.
- `elevation` A numeric vector with elevation values (in m).
- `slope` A numeric vector with slope values (in degrees).
- `aspect` A numeric vector with aspect values (in degrees from North).
- `SpParams` A data frame with species parameters (see `SpParamsMED`).
- `SoilParamData` A data frame with soil parameters for each forest stand. The rows must match `SpatialPointsIDs` or `SpatialGridIDs`, depending on the function called. If `NULL`, then the parameters given by `defaultSoilParams` are used.
- `SFIcodes` A string vector (of length equal to the number of rows in `SpParams` of the SFI species codes that correspond to the model species codification. Each string may contain different coma-separated codes in order to merge SFI species into a single model species.
- `FMcodes` A string vector with the fuel model code corresponding to each forest plot. Names of this string vector should be plot IDs.
- `control` A list of control parameters (see `defaultControl`).
- `lctInput` A character vector with the land cover type input for initialization of unsurveyed cells.
- `forestLCTs, shrublandLCTs, grasslandLCTs` Integer vectors with the land cover types that are to be considered 'wildland' (i.e. forests, shrublands or grasslands). Initialization of forest and shrublands cells will make use of SFI plot data, but in shrublands only shrubs are used. Initialization of grasslands specifies an `emptyforest`.
- `agricultureLCTs, rockLCTs, staticLCTs` Integer vectors with the land cover types that are to be considered 'agriculture', 'rock' (i.e. outcrops) or 'static' (i.e. urban, water masses). Agricultural areas are like grasslands, but they can never be vegetated with wildland. Rock cells allow surface water flow over them, but do not contain vegetation nor soil and cannot be burned. Static cells do not burn and water entering them does not flow to any other cell.

Value

An object of class `SpatialPointsForest-class` or `SpatialGridForest-class` depending on the function.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

forest, extractSFIforest

SpatialPointsForest-class
Class "SpatialPointsForest"

Description

An S4 class that represents a set of forest stands along with their spatial location

Objects from the Class

Objects can be created by calls of the form `new("SpatialPointsForest", ...)` or by calls to function `SpatialPointsForest`.

Slots

`forestlist`**:** Object of class `"list"` containing a set of `forest` objects.

`soillist`**:** Object of class `"list"` containing a set of `soil` objects.

`data`**:** Object of class "data.frame" containing the elevation (in m), slope (in degrees) and aspect (in degrees from North) of every cell.

`coords`**:** Object of class `"matrix"` with spatial coordinates.

`bbox`**:** Object of class `"matrix"` with the boundary box.

`proj4string`**:** Object of class `"CRS"` with the projection string.

Extends

Class "`SpatialPointsTopography`", directly. Class "`SpatialPointsDataFrame`", distance 2. Class "`SpatialPoints`", distance 3. Class "`Spatial`", by class "SpatialPoints", distance 4.

Methods

spatialForestSummary `signature(object = "SpatialPointsForest")`: calculates a summary function for all forest stands and returns an object of class `SpatialPointsDataFrame-class`.

spatialSoilSummary `signature(object = "SpatialPointsForest")`: calculates a summary function for the soil of all stands and returns an object of class `SpatialPointsDataFrame-class`.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

`SpatialPointsTopography`, `SpatialPointsForest`, `SpatialGridForest`, `spatialForestSummary`, `forest`

Examples

```
showClass("SpatialPointsForest")
```

Species values *Species description functions*

Description

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

Usage

```
species.BasalArea(x, SpParams)
```

Arguments

`x`	An object of class `forest`.
`SpParams`	A data frame with species parameters (see `SpParamsMED`).

Value

A vector with values for each species in `SpParams`:

- `species.BasalArea`: Species basal area (m2/ha).

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

`swb`, `forest`, `plant.BasalArea`, `summary.forest`

Examples

```
#Default species parameterization
data(SpParamsMED)

#Load example plot
data(exampleforest)

#Species basal area
species.BasalArea(exampleforest, SpParamsMED)
```

SpParamsMED *Parameter values for Mediterranean species*

Description

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

Usage

```
data("SpParamsMED")
```

Format

A data frame with 89 observations (species) on the following 42 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
- `GrowthForm` A factor with levels `Shrub`, `Tree` and `Tree/Shrub`
- `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 cummulative 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.
- `TreeType` A factor with levels `Conifer`, `Deciduous`, `Evergreen` or `Shrub`
- `SLA` Specific leaf area (mm2/mg = m2/kg).
- `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)
- `LeafDuration` Leaf duration (in years).
- `LigninPercent` Percent of lignin+cutin over dry weight in leaves.
- `ParticleDensity` Particle density (kg/m3).
- `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).

`Z_max` Maximum rooting depth (in mm).

`k` Light extinction coeficient for PAR.

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

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

`psiExtr` Water potential corresponding to 50% relative conductance (in kPa).

`RC_min` Minimum canopy resistance (in s/m)

`VC_kxmax` Maximum xylem hydraulic conductance (in L/m2/day)

`VC_c`, `VC_d` Parameters of the xylem vulnerability curve.

Details

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

Source

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

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

See Also

swb

Examples

```
data(SpParamsMED)
```

`swb` *Soil water balance model*

Description

Function `swb` is a soil water balance model that determines changes in soil moisture, soil water potentials and plant drought stress at daily steps for a given forest stand during a period specified in the input climatic data.

Usage

```
swb(x, soil, meteo, petMode = "Input", hydraulicMode = "Simple",
    latitude = NA, elevation = NA, verbose=TRUE)
```

Arguments

- `x` — A data frame with parameters for each plant cohort (see `forest2swbInput`).
- `soil` — A list containing the description of the soil (see `soil`).
- `meteo` — A data frame with daily meteorological data series. When `petMode = "Input"` 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).

  When `petMode = "PenmanMonteith"` the following columns are required:
  - `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.
- `petMode` — Potential evapotranspiration mode (either `"Input"` or `"PenmanMonteith"`). If `petMode = "Input"` the function takes column 'PET' in `meteo` as potential evapotranspiration. If `petMode = "PenmanMonteith"` the function calculates 'PET' internally, taking into account the current leaf area index of the stand.
- `hydraulicMode` — Hydraulic model (either 'Simple' or 'Sperry').
- `latitude` — Latitude (in radians).
- `elevation` — Elevation above sea level (in m).
- `verbose` — Boolean flag to indicate console output during calculations.

Details

Detailed model description is available in the vignettes section. The model using simple hydraulic model is described in De Caceres et al. (2015). WARNING: objects `x` and `soil` are modified during the simulation.

Value

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

- "petMode": Potential evapotranspiration mode (either "Input" or "PenmanMonteith").
- "hydraulicMode": Hydraulic model (either 'Simple' or 'Sperry').
- "DailyBalance": A data frame where different variables (in columns) are given for each simulated day (in rows):
  - "LAIcell": The LAI of the stand (accounting for leaf phenology) (in m2/m2).
  - "Cm": The water retention capacity of the stand (in mm) (accounting for leaf phenology).
  - "Lground": The proportion of PAR that reaches the ground (accounting for leaf phenology).
  - "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).
  - "Etot": Evapotranspiration (in mm).
  - "Esoil": Bare soil evaporation (in mm).
  - "Eplanttot": Plant transpiration (considering all soil layers) (in mm).
  - "Eplant1": Plant transpiration from soil layer 1 (in mm).
  - "Eplant2": Plant transpiration from soil layer 2 (in mm).
  - "Eplant3": Plant transpiration from soil layer 3 (in mm).
- "SoilWaterBalance": A data frame where different variables (in columns) are given for each simulated day (in rows):
  - "W1": Relative soil moisture content (relative to field capacity) in layer 1.
  - "W2": Relative soil moisture content (relative to field capacity) in layer 2.
  - "W3": Relative soil moisture content (relative to field capacity) in layer 3.
  - "ML1": Soil water volume in layer 1 (in L/m2).
  - "ML2": Soil water volume in layer 2 (in L/m2).
  - "ML3": Soil water volume in layer 3 (in L/m2).
  - "MLTot": Total soil water volume (in L/m2).
  - "psi1": Soil water potential in layer 1 (in kPa).
  - "psi2": Soil water potential in layer 2 (in kPa).
  - "psi3": Soil water potential in layer 3 (in kPa).
- "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'.
- "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'.

- "PlantTmax": A data frame with the amount of daily maximum 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'.
- "PlantPsi" (only for hydraulicMode = "Sperry"): A data frame with the average daily water potential each plant (in kPa). Days are in rows and plant cohorts are in columns. Columns in this data frame correspond to the elements in 'SP'.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

See Also

plot.swb, swbpoints, swbgrid,forest

Examples

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

#Load example plot plant data
data(exampleforest)

#Default species parameterization
data(SpParamsMED)

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

#Call simulation function
S1<-swb(forest2swbInput(exampleforest,SpParamsMED, examplesoil$dVec),
        examplesoil, examplemeteo)

#Plot results
plot(S1)

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


#Simulation with PET calculated with Penman-Monteith
examplesoil = soil(defaultSoilParams())
S2<-swb(forest2swbInput(exampleforest,SpParamsMED, examplesoil$dVec, petMode="PenmanMonteith"),
        examplesoil, examplemeteo, latitude = 0.73,
```

```
        elevation = 100, petMode="PenmanMonteith")

#Simulation with transpiration calculated using Sperry & Love
examplesoil = soil(defaultSoilParams())
S3<-swb(forest2swbInput(exampleforest,SpParamsMED, examplesoil$dVec, hydraulicMode="Sperry"),
        examplesoil, examplemeteo, hydraulicMode="Sperry")

#Simulation with PET calculated using PenmanMonteiht
#and transpiration calculated using Sperry & Love
examplesoil = soil(defaultSoilParams())
S4<-swb(forest2swbInput(exampleforest,SpParamsMED, examplesoil$dVec,
                        petMode = "PenmanMonteith", hydraulicMode="Sperry"),
        examplesoil, examplemeteo, latitude = 0.73,
        elevation = 100, petMode = "PenmanMonteith", hydraulicMode="Sperry")
```

`swb.RainInterception` *Rainfall interception*

Description

Function `swb.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

```
swb.RainInterception(Rainfall, Cm, p, ER=0.05, method="Gash1995")
```

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`).

Details

The function is prepared to accept both vectors or scalars for parameters `Cm`, `p` and `ER`. If they are supplied as vectors they should be of the same length as `Rainfall`.

Value

Returns a vector of the same length as `Rainfall` containing intercepted rain values.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

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

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

See Also

swb

Examples

```
throughfallMatrixGash<-function(P = seq(1,50, by=1), Cm = seq(1,5, by=1),
                                ER = 0.08,p=0.8) {
  m2<-P-swb.RainInterception(P,Cm[1],p,ER=ER)
  for(i in 2:length(Cm)) {
    m2<-rbind(m2,P-swb.RainInterception(P,Cm[i],p,ER=ER))
  }
  colnames(m2)<-P
  rownames(m2)<-Cm
  return(m2)
}

Cm = c(0.5,seq(1,4, by=1))
P = seq(1,50, by=1)

m2 = throughfallMatrixGash(P=P, p=0.2, Cm=Cm,ER = 0.05)
rt = sweep(m2,2,P,"/")*100
matplot(t(rt), type="l", axes=TRUE, ylab="Relative throughfall (%)",
        xlab="Gross rainfall (mm)", xlim=c(0,length(P)),
        lty=1:length(Cm), col="black", ylim=c(0,100))
title(main="p = 0.2 E/R = 0.05")
legend("bottomright",lty=1:length(Cm), legend=paste("Cm =",Cm), bty="n")
```

---

swbgrid *Soil water balance and lateral water discharge*

Description

Function swbgrid conducts daily soil water balance over a grid of cells while incorporating water runoff from upperslope cells into the current cell.

Usage

```
swbgrid(y, SpParams, meteo, dates,
           summaryFreq = "years", trackSpecies = numeric(),
           control = defaultControl())
## S3 method for class 'swbgrid'
plot(x, type = "Runon", summaryIndex = 1, spIndex = NULL, ...)
```

Arguments

`y`	An object of class `SpatialGridForest-class`.
`SpParams`	A data frame with species parameters (see `SpParamsMED`).
`meteo`	A `SpatialGridMeteorology-class` object or a data frame with two columns: 'dir' and 'filename, to indicate the path to the meteorological data.
`dates`	A `Date` object describing the days of the period to be modeled.
`summaryFreq`	Frequency in which summary layers will be produced (e.g. "years", "months", ...) (see `cut.Date`).
`trackSpecies`	An integer vector containing the indices of species for which transpiration and drought stress is to be tracked.
`control`	A list of control parameters (see `defaultControl`).
`x`	An object of class `swbgrid`.
`type`	Type of information to be drawn.
`summaryIndex`	The index of the summary to be ploted.
`spIndex`	The index of the species to be ploted (for some types).
`...`	Additional parameters to function `spplot`.

Details

Function `swbgrid` requires daily meteorological data over a grid. The user may supply an object of class `SpatialGridMeteorology` (see package 'meteoland') or a data frame with information regarding where to read meteorological data.

Value

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

- `grid`: The `GridTopology` object corresponding to the simulated area.
- `LandscapeBalance`: A data frame with as many rows as summary points and where columns are components of the water balance at the landscape level (i.e., precipitation, interception, soil evaporation, plant transpiration, ...).

Then, the following matrices are included (each with as many rows as cells and as many columns as summary points):

- `NetPrec`: Net precipitation, after accounting for interception (in mm).
- `Infiltration`: The amount of water infiltrating into the soil (in mm).

- Runon: The amount of water imported from other cells via surface runoff (in mm).
- Runoff: The amount of water exported via surface runoff (in mm).
- DeepDrainage: The amount of water exported via deep drainage (in mm).
- Esoil: Bare soil evaporation (in mm).
- Eplant: Plant transpiration (in mm).

The same list contains two three-dimensional arrays (each with dimensions number of cells, number of summary layers and number of tracked species):

- Transpiration: Total transpiration (in mm) of the tracked species for each summary period.
- DI: Drought intensity (from 0 to 1) of the tracked species for each summary period.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya.

See Also

swb, swbpoints, SpatialGridForest-class

---

swbpoints *Soil water balance model for spatially-distributed forest stands*

Description

Function swbpoints allows calling local models swb for a set of forest stands distributed in specific locations. No spatial processes are simulated.

Usage

```
swbpoints(y, SpParams, meteo, dates = NULL, control = defaultControl(),
       summaryFunction = NULL, args=NULL)
```

Arguments

y	An object of class SpatialPointsForest-class.
SpParams	A data frame with species parameters (see SpParamsMED).
meteo	Meteorology data (see details).
dates	A Date object with the days of the period to be modeled. If NULL, then the whole period of meteo is used.
control	A list of control parameters (see defaultControl).
summaryFunction	An appropriate function to calculate summaries (e.g., summary.swb).
args	List with additional arguments for the summary function.

Details

Function swbpoints accepts different formats for meteorological input (parameter meteo). If a data.frame is supplied (as in swb) then the same meteorology is used for all points (not recommended). To specify different meteorology for different points, the user can use an object of SpatialPointsMeteorology-class. Alternatively, the user can supply an object of class SpatialPointsDataFrame-class containing the meta data (columns dir and filename) of meteorological files that will be read from the disk.

Value

Function swbpoints returns a list with the result of calling swb on each forest stand. If summaryFunction is not null, then each element of the list will contain the result of the summary function.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

See Also

swb, SpatialPointsForest-class

Examples

```
#Load forest data
data(exampleSPF)

#Load species parameters
data(SpParamsMED)

#Load meteorology
data(examplemeteo)

##Example using the same meteorology for all points
a = swbpoints(exampleSPF, SpParamsMED, examplemeteo)
```

---

swbprocesses *Soil infiltration and bare soil evaporation*

Description

Function swb.SoilInfiltration calculates the amount of net precipitation 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 swb.SoilEvaporation calculates the amount of evaporation from bare soil, following Ritchie (1972).

Usage

```
swb.SoilInfiltration(NetPrec, Ssoil)
swb.SoilEvaporation(DEF,PETs, Gsoil)
```

Arguments

`NetPrec`	A numeric vector of (daily) net precipitation (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).

Details

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

Value

Function `swb.SoilInfiltration` a vector of the same length as `NetPrec` containing the daily amount of water that infiltrates into the soil (in mm of water). Function `swb.SoilEvaporation` returns the amount of water evaporated from the soil.

Author(s)

Miquel De Cáceres Ainsa, Centre Tecnologic Forestal de Catalunya

References

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

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

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

See Also

`swb`

Examples

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

#TOPSOIL LAYERS
d1 = pmin(SoilDepth, 300) #<300
#SUBSOIL LAYERS
d2 = pmax(0, pmin(SoilDepth-300,1200)) #300-1500 mm
```

```
#ROCK LAYER
d3 = 4000-(d1+d2) #From SoilDepth down to 4.0 m

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

Theta_FC1=soil.psi2theta(TS_clay, TS_sand, -33) #in m3/m3
Theta_FC2=soil.psi2theta(SS_clay, SS_sand, -33) #in m3/m3
Theta_FC3=soil.psi2theta(RL_clay, RL_sand, -33) #in m3/m3
pcTS_gravel = 1-(TS_gravel/100)
pcSS_gravel = 1-(SS_gravel/100)
pcRL_gravel = 1-(RL_gravel/100)
MaxVol1 = (d1*Theta_FC1*pcTS_gravel)
MaxVol2 = (d2*Theta_FC2*pcSS_gravel)
MaxVol3 = (d3*Theta_FC3*pcRL_gravel)
V = MaxVol1+MaxVol2+MaxVol3

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

`Vertical profiles` *Vertical profiles*

Description

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

Usage

```
vprofile.LeafAreaDensity(x, SpParams, z = NULL, gdd = NA, draw = TRUE)
vprofile.FuelBulkDensity(x, SpParams, z = NULL, gdd = NA, draw = TRUE)
vprofile.PARExtinction(x, SpParams, z = NULL, gdd = NA, draw = TRUE)
vprofile.SWRExtinction(x, SpParams, z = NULL, gdd = NA, draw = TRUE)
vprofile.WindExtinction(x, SpParams, wind20H, z = NULL,  gdd = NA, draw = TRUE)
```

Arguments

`x`	An object of class `forest`
`SpParams`	A data frame with species parameters (see `SpParamsMED`).
`z`	A numeric vector with height values.
`gdd`	Growth degree days.
`wind20H`	The value of measured wind speed at 6 m = 20ft (in m/s).
`draw`	A logical flag to indicate that a plot is desired.

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, Centre Tecnologic Forestal de Catalunya

See Also

`forest`

Examples

```
#Default species parameterization
data(SpParamsMED)

#Load example plot plant data
data(exampleforest)

vprofile.LeafAreaDensity(exampleforest, SpParamsMED)
vprofile.FuelBulkDensity(exampleforest, SpParamsMED)

vprofile.PARExtinction(exampleforest, SpParamsMED)
```

```
vprofile.SWRExtinction(exampleforest, SpParamsMED)

vprofile.WindExtinction(exampleforest, SpParamsMED, 20)
```

Index