Package 'TapeS'

November 26, 2024

Type Package

Title Tree Taper Curves and Sorting Based on 'TapeR'

Version 0.13.0

Description Providing new german-wide 'TapeR' Models and functions for their evaluation. Included are the most common tree species in Germany (Norway spruce, Scots pine, European larch, Douglas fir, Silver fir as well as European beech, Common/Sessile oak and Red oak). Many other species are mapped to them so that 36 tree species / groups can be processed. Single trees are defined by species code, one or multiple diameters in arbitrary measuring height and tree height. The functions then provide information on diameters along the stem, bark thickness, height of diameters, volume of the total or parts of the trunk and total and component above-ground biomass. It is also possible to calculate assortments from the taper curves. For diameter and volume estimation, uncertainty information is given.

Depends R (>= 3.5.0)

License BSD_2_clause + file LICENSE

URL https://gitlab.com/vochr/tapes

Encoding UTF-8

RoxygenNote 7.3.2

Imports methods, utils, TapeR (>= 0.5.2), Rcpp (>= 1.0.5),

LinkingTo Rcpp, RcppArmadillo

Suggests testthat, knitr, rmarkdown, rbenchmark, rBDAT (>= 0.10.0), RODBC

VignetteBuilder knitr

NeedsCompilation yes

Author Christian Vonderach [aut, cre], Edgar Kublin [aut], Gerald Kändler [aut]

Maintainer Christian Vonderach <christian.vonderach@forst.bwl.de>

Repository CRAN

Date/Publication 2024-11-26 10:50:02 UTC

Contents

Az *estimate minimum cutting diameter*

Description

estimate minimum cutting diameter based on diameter in breast height based on the functions implemented in BDAT

Usage

```
Az(sp, dbh)
```

Arguments

sp	Bdat species code [1;36], integer
dbh	vector of diameter in breast height, numeric

Details

the implemented BDAT function and parameters are used. Not all BDAT-species possess their own parameters, hence most of them are matched to one of the main tree species, especially in deciduous tree species (only parameters for beech and oak are available).

Value

vector of minimum cutting diameter [cm].

Examples

```
sp <- 1
dbh <- 30
Az(sp, dbh)
```

BaMap *function for mapping the 36 tree species to several internal functions*

Description

function for mapping the 36 tree species to several internal functions

Usage

```
BaMap(Ba = NULL, type = NULL)
```

Arguments

Ba	BDAT tree number
type	a number referring to the type to be mapped

Details

c.f. BDAT source code, line 7622, data block Ban(36, 7) type 1: Schaftform // taper form type 2: Rinde // bark type 3: Durchschnittliche Aufarbeitungsgrenze (nach EST) //average cutting diameter type 4: Höhe unverwertbares Derbholz // percentage non-merchantable coarse wood type 5: durchschnittlicher Astdurchmesser in der Krone // average branch diameter inside crown type 6: BWI-Biomasse-Funktionen // NFI-biomass functions according to Riedel & Kändler (2017) type 7: kompartimentweise Biomassefunktionen // component biomass functions according to Vonderach et al (2018) type 8: Zuordnung zu volfao // Mapping to volume according to FAO (FIX: mapping still temporary) Not included: volume tables according to Grundner and Schwappach as well as volume tables according to Krenn for small trees below 10cm dbh

Value

value(s), either a scalar, vector or matrix, with respect to tree species mapping to functions

Examples

```
BaMap(1,1) # which taper form for Norway spruce
BaMap(15,1) # which taper form for European Beech
BaMap(15,2) # which bark equation for European Beech
BaMap(,1) # return all taper form mappings
BaMap(1,) # return all mappings for Norway spruce
BaMap() # return all mappings
BaMap(, 6) # biomass mapping
BaMap(, 7) # component biomass functions
BaMap(, 8) # mapping for Vol_FAO
```

---

bark *Double Bark Thickness*

Description

Function returns double bark thickness according to Altherr et al. 1974/75/76/78/79

Usage

```
bark(Ba, Dm, relH)
```

Arguments

Ba	tree species according to BDAT, cf. tprSpeciesCode
Dm	diameter for which double bark thickness is requested
relH	relative height of Dm inside stem

Details

Function re-implemented according to Subroutine RINDE(Hhrel,Kw,Ri,Hsga,Zo), BDAT-fortran Code line 5691ff. No Functions for (historic) Heilbronner Sortierung implemented.

NB: to avoid negative double bark thickness, such values are constraint to zero. Additionally, diameter after bark reduction might not be smaller than zero, hence double bark thickness is reduce to `Dm`.

Value

double bark thickness [cm]

References

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1974). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Kiefer, Buche, Hainbuche, Esche und Roterle. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1975). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Europäische Lärche, Japanische Lärche, Schwarzkiefer, Stieleiche, Traubeneiche, Roteiche, Bergahorn und Linde. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1976). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Weymouthskiefer, Robinie, Bergulme, Birke, Marilandica-Pappel und Robusta-Pappel. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1978). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Fichte, Tanne, Douglasie und Sitka-Fichte. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1979). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Neupotz-Pappel, Regenerata-Pappel, Kirsche, Spitzahorn, Feldahorn, Aspe, Weide, Flatterulme, Tulpenbaum u. Elsbeere. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Examples

```
bark(1, 30, .1)
bark(11, 4, .1) # zero instead of -0.2497
```

`biomass` *Prediction of above-ground biomass according to NFI-functions*

Description

Prediction of total above-ground biomass for trees defined via species, dbh, d03 and height

Usage

```
biomass(spp, d13, d03, h)
```

Arguments

spp	vector of species code for biomass function [1;18]
d13	vector of diameter in breast height in centimeter
d03	vector of diameter in 30% of tree height in centimeter
h	vector of height of trees

Details

code taken from BDAT (Koeff.f).

Value

a vector of total above-ground biomass

References

Riedel, T. and G. Kaendler (2017). "Nationale Treibhausgasberichterstattung: Neue Funktionen zur Schätzung der oberirdischen Biomasse am Einzelbaum." Forstarchiv 88(2): 31-38.

---

calcVCOVsekVol *calculate VCOV-Matrix for volume segments*

Description

calculate variance-covariance matrix for volume segments from the estimated diameter and uncertainty information from TapeR-model

Usage

```
calcVCOVsekVol(estD, kovD, estL)
```

Arguments

estD	vector of estimated diameter, numeric
kovD	variance-covariance-matrix of the estimated diameter, numeric
estL	vector of segment length, numeric

Details

Calculations according to rules for products and sums of variances

Value

variance-covariance matrix of the segment volume

check_Comp *generate and/or check validity of biomass function component names*

Description

generate and/or check validity of biomass function component names

Usage

```
check_Comp(component = NULL)
```

Arguments

component	vector of biomass component names, see details

Details

If component is NULL, by default, component name for total aboveground biomass is returned. If is all, then all available component names are returned.

- stw: stump wood
- stb: stump bark
- sw: solid wood with diameter above 7cm over bark
- sb: bark of component sw
- fwb: fine wood incl. bark
- ndl: needles
- agb: total aboveground biomass

Value

a vector of component names

Examples

```
## Not run:
TapeS:::checkComp()
TapeS:::checkComp("AGB")
TapeS:::checkComp("biomass")

## End(Not run)
```

check_monotonicity *monotonicity check for taper curve*

Description

monotonicity check for taper curve

Usage

```
check_monotonicity(obj, Rfn = NULL)
```

Arguments

obj	object of class 'tprTrees'
Rfn	Rfn setting for residuals error matrix, defaults to `list(fn="sig2")`, see `resVar`.

Details

Taper curves are required to decrease monotonically. To avoid the evaluation of non-monotone taper curves, a check is done through the constructor function and an indicator (`monotone`) is set for each tree stored inside the `tprTrees`-class. As the data has been check on validity before this function is applied, we can use the tpr*-functions to evaluate the taper curve and its monotonicity. The check is done via comparison of the expected diameters along the trunk in 1m-steps and its sorted (monotonically decreasing) version using `identical`.

Value

vector of logicals, same length as `spp`.

data_coercion *coerce different data sources into class 'tprTrees'*

Description

coercion functions to make NFI, segment and 'BDAT' data available as 'tprTrees' objects

Usage

```
nfi_as_tprtrees(nfi, mapping = NULL)

seg_as_tprtrees(seg, mapping = NULL)

bdat_as_tprtrees(bdat)
```

Arguments

nfi	data.frame with tree measurements as provided by german NFI
mapping	mapping of column names
seg	data.frame with measured tree segments, see details.
bdat	data.frame holding data to process with rBDAT

Details

The coerced data is automatically checked for validity by the class constructor. For available species codes see tprSpeciesCode.

When coercing NFI (National forest inventory, BWI) data, one need to provide the columns BaTpr (species code), Bhd (Dbh, [mm]), D03, [mm] (diameter in 30% of tree height) and Hoehe (tree height, [dm]). Optionally, one can provide H1 (measurement height of Bhd, [dm]), H2 (measurement height of D03, [dm]) as well as sHt (measurement error of tree height, i.e. standard deviation [dm]); otherwise these are assumed to be 1.3m, 30% of tree height and 0 (zero), respectively.

Additionally, the NFI database stores diameter as [mm] and height as [dm]; it is *not* necessary to transform to [cm] and [m], as the function does this. Equally, sHt [dm] is transformed to sHt [m].

Keep in mind that species codes of NFI are different from the taper models for historical reasons (c.f. BDAT). Use the NFI table ('x_Ba') to map species codes beforehand (see examples).

Sectional measurements provide more information about the trunk of a tree and are usually stored in a different way. They exhibit an arbitrary amount of diameter measurements which also might vary from tree to tree. Hence, seg_as_tprtrees expects a data.frame with columns Id, BaTpr (species code), Dm (diameter measured, [cm]), Hm (height of Dm, [m]) and optionally Ht (height of tree, [m]). Tree height Ht can be included to Dm-Hm-pairs with Dm being zero (e.g. Dm=0, Hm=25). If Ht is given, it gains priority.

coercing object of class 'datBDAT' from R-Package "rBDAT" into class 'tprTrees'

Value

an object of class 'tprTrees', see tprTrees-class

Functions

- nfi_as_tprtrees(): coercion of German NFI data
- seg_as_tprtrees(): coercion of segmented data to class 'tprTrees'
- bdat_as_tprtrees(): coercion of bdat data

See Also

tprTrees-class, tprTrees, tprSpeciesCode

Examples

```
# NFI data usually stored as integer and units: diameter=[mm] and height=[dm]
nfi <- data.frame(BaTpr=1L, Bhd=300L, D03=270L, Hoehe=250L)
tpr <- nfi_as_tprtrees(nfi)
tpr
tpr@sHt # defaults to 0

# one can provide measurement heights explicitly
nfi <- data.frame(spp=1, Bhd=300, H1=12, D03=270, H=250)
nfi_as_tprtrees(nfi, mapping=c(spp="BaTpr", H="Hoehe"))

# measurement error in height
nfi <- data.frame(BaTpr=1L, Bhd=300L, D03=270L, Hoehe=250L, sHt=15)
tpr <- nfi_as_tprtrees(nfi)
tpr@sHt
## coercing sectional measurements
data(DxHx.df, package = "TapeR")
DxHx.df$BaTpr <- 1 # Norway spruce
segtprtrees <- seg_as_tprtrees(DxHx.df, mapping=c(Dx="Dm", Hx="Hm"))

## extract tree height from Dm-Hm measurements if not given explicitly
DxHx.df$Ht <- NULL # remove height, as already included with Dm=0
segtprtrees <- seg_as_tprtrees(DxHx.df, mapping=c(Dx="Dm", Hx="Hm"))
segtprtrees
if(require(rBDAT)){
  bdt <- buildTree(list(spp=1, D1=30, D2=28, H2=7, H=25))
  bdat_as_tprtrees(bdt)
}
```

Dbh *Extract pre-defined diameter*

Description

Function extracts pre-defined diameters, e.g. dbh (in 1.3m) or D03 (in 30% of tree height) for a tprTrees-object

Usage

```
Dbh(obj)

Bhd(obj)

D13(obj)

D7(obj)
```

```
D03(obj)

D005(obj)
```

Arguments

obj	a object of class 'tprTrees'

Details

a wrapper around tprDiameter to calculate specifically defined diameters like diameter in breast height (dbh), diameter in 7m above ground or in 5% and 30% of tree height.

Value

diameter(s) in predefined heights

Functions

- Dbh(): wrapper to calculate diameter in breast height
- Bhd(): German alias for function Dbh
- D13(): Height specific alias for function Dbh
- D7(): Function to calculate diameter over bark in 7m above ground
- D03(): Function to calculate diameter over bark in 30% of tree height
- D005(): Function to calculate diameter over bark in 5% of tree height

Examples

```
t <- tprTrees()
Dbh(t) # diameter in breast height (i.e. 1.3m)
Bhd(t) # same, german named function name
D13(t) # same, height related function name
D005(t) # diameter in 5% of tree height
D7(t) # diameter in height of 7m
D03(t) # diameter in 30% of tree height
```

---

estHeight *Estimate tree height by DBH according to BWI3*

Description

Function calculates expected tree height given diameter in breast height and species code

Usage

```
estHeight(d13, sp, qtl = NULL)
```

Arguments

d13	numeric vector of diameter in breast height [cm]
sp	TapeS species code, see also tprSpeciesCode
qtl	desired quantile of height, either NULL (LS-regression) or one of 17, 50, 83 (quantile regression)

Details

Function evaluates the expected value of a Pettersen-Height Function based on diameter in breast height and tree species code. The Pettersen-Function ($h = 1.3 + (a + \frac{b}{dbh})^{-3}$) was fitted on NFI 3 (BWI 3) data, using the main stand only.

d13 and sp should be of equal length or one of it can be > 1 if the other is of length 1. Then, the shorter object will be extended to match the length of the longer object. See examples.

The quantile option return tree height at quantiles 17, 50 or 83. If qtl is NULL, the result of a nonlinear least-squares regression is provided.

Value

a vector of same length as d13 or sp, with tree height in [m].

Examples

```
sp <- 1
d13 <- 30
estHeight(d13, sp)

sp <- 1
d13 <- seq(15, 50, 5)
estHeight(d13, sp)

sp <- 1:36
d13 <- 30
estHeight(d13, sp)
```

E_HDxoR_HmDm_Ht.f *Find Height of diameter under bark via uniroot*

Description

Functional equivalent to E_HDx_HmDm_HT.f, finding the height of a given diameter *without* bark, i.e. double bark thickness needs to be added on top of given diameter to find appropriate height.

Usage

```
E_HDxoR_HmDm_Ht.f(DxoR, Hm, Dm, mHt, sHt = 0, par.lme, Rfn = NULL, ...)

HxoR_root.f(Hx, DxoR, Hm, Dm, mHt, sHt, par.lme, Rfn, ...)
```

Arguments

DxoR	Scalar. Diameter under bark for which to return height.
Hm	Numeric vector of stem heights (m) along which diameter measurements were taken for calibration. Can be of length 1. Must be of same length as Dm.
Dm	Numeric vector of diameter measurements (cm) taken for calibration. Can be of length 1. Must be of same length as Hm.
mHt	Scalar. Tree height (m).
sHt	Scalar. Standard deviation of stem height. Can be 0 if height was measured without error.
par.lme	List of taper model parameters obtained by TapeR_FIT_LME.f, enhanced by the attribute 'spp', which refers to the tree species used for double bark thickness
Rfn	setting for residuals error matrix, defaults to "sig2", see details.
...	not currently used
Hx	height at which taper curve is evaluated

Details

finds height of given diameter via uniroot.

Value

A scalar. Estimated height (m) given a diameter without bark.

Functions

- HxoR_root.f(): function to be searched

Examples

```
tmp <- tprTrees()
spp <- spp(tmp)
Hm <- Hm(tmp)
Dm <- Dm(tmp)
H <- Ht(tmp)
SKP <- TapeS:::SKPar
sppSK <- BaMap(spp, 1) # tree species for taper curve
## diameter in 5m height
TapeR::E_DHx_HmDm_HT.f(c(5, 10), Hm, Dm, mHt=H, sHt = 0, par.lme = SKP[[sppSK]])$DHx
(D5m <- TapeR::E_DHx_HmDm_HT.f(c(5, 10), Hm, Dm, mHt=H, sHt = 0, par.lme = SKP[[sppSK]])$DHx)
## bark thickness of diameter in 5m height
(RiD5m <- bark(c(1,1), Dm = D5m, relH = c(5, 10)/H))
## find height of diameter without bark, which should be 5m
d5mub <- D5m - RiD5m
E_HDxoR_HmDm_Ht.f(DxoR = d5mub, Hm = Hm, Dm = Dm, mHt = H,
                  sHt = 0, par.lme = SKP[[sppSK]])
```

fnUnvd *percentage of unusable coarse wood*

Description

function extracts the percentage of unusable coarse wood according to species (beech, oak), diameter class and cutting diameter

Usage

```
fnUnvd(ba = NULL, dm = NULL, cd = NULL)
```

Arguments

ba	tree species index; see details
dm	diameter class; see details
cd	cutting diameter; see details

Details

Function extracts the percentage of unusable coarse wood according to three parameters: (i) tree species, which is 1 for using beech models and 2 for using the oak model; (ii) the 2cm-diameter class (from 8 and 60cm) and (iii) the cutting diameter ranging from 8 to 40cm.

References

Kublin and Scharnagl (1988): Verfahrens- und Programmbeschreibung zum BWI-Unterprogramm BDAT. FVA-BW 1988. ISSN: 0178-3165.

FormTariff *Tariff for taper form*

Description

evaluates tariff functions to estimate taper form, i.e. quotient of d03 by d005

Usage

```
FormTariff(spp, Dbh, Ht, inv)
```

Arguments

spp	species code of tprSpeciesCode
Dbh	diameter of considered tree at 1.3m above ground [cm]
Ht	tree height of considered tree [m]
inv	indicator for inventory (0=TapeS taper curve models, 1=NFI1, 2=NsoG, 3=IS08, 4=NFI3, 5=BDAT)

Value

quotient of d03 / d005 [unitless]

References

c.f. rBDAT::getForm respectively BDAT source code FormTarif.f

Examples

```
## dont't run
spp <- 15
Dbh <- 30
Ht <- 27
FormTariff(spp, Dbh, Ht, 0)
FormTariff(spp=c(1:2), Dbh=c(30, 30), Ht=c(27, 24), inv=0)
if(require("rBDAT")){
  FormTariff(spp, Dbh, Ht, 0)
  rBDAT::getForm(list(spp=spp, D1=Dbh, H1=1.3, H=Ht), inv=0) # different taper curves!
  FormTariff(spp, Dbh, Ht, 1)
  rBDAT::getForm(list(spp=spp, D1=Dbh, H1=1.3, H=Ht), inv=1) # identical
  FormTariff(spp, Dbh, Ht, 2)
  rBDAT::getForm(list(spp=spp, D1=Dbh, H1=1.3, H=Ht), inv=2) # identical
  FormTariff(spp, Dbh, Ht, 3)
  rBDAT::getForm(list(spp=spp, D1=Dbh, H1=1.3, H=Ht), inv=3) # identical
  FormTariff(spp, Dbh, Ht, 4)
  rBDAT::getForm(list(spp=spp, D1=Dbh, H1=1.3, H=Ht), inv=4) # identical
}
```

HtCoef *returns coefficients for Pettersen-Height model*

Description

Function to provide model coefficients for Pettersen-height model

Usage

```
HtCoef(sp = NULL, qtl = NULL)
```

Arguments

sp	BDAT species code, could be NULL then all coefficients are returned
qtl	quantile, either NULL or 17, 50, 83

Value

a data.frame with species code and coefficients

lmeSKEBLUP *diameter prediction E[d] for TapeR-object*

Description

Prediction diameter (no variances) for given tree and TapeR-object using BSpline Matrix all in C++

Usage

```
lmeSKEBLUP(xm, ym, xp, par, RV)
```

Arguments

xm	relative height of measured diameter
ym	measured diameter for calibration
xp	relative height for which diameter prediction is required
par	a TapeR-object (including padded knots vector)
RV	numeric vector holding assumed residual variance for each observation

Details

code implementation in C++ following the code base of TapeR. Bspline matrix code taken from R-package splines to avoid the need of calling R from C.

Value

a list holding several elements, perspectively only the estimated diameter

nsur *Component biomass functions*

Description

evaluation of the component biomass functions fit by nonlinear seemingly unrelated regression (NSUR) to estimate absolute or relative component mass

Usage

```
nsur(spp, dbh, ht, sth, d03, kl)
```

Arguments

spp	vector of species code for biomass component function of interval [1;8]; see BaMap for mapping of species model codes
dbh	vector of diameter in breast height; in centimeter
ht	vector of tree heights, in meter
sth	vector of stump heights, in meter
d03	vector if diameter in 30% of tree height, in centimeter
kl	vector of crown length, i.e. tree height minus height of crown base, in meter

Details

function to calculate component biomass; functions fitted using same methodology as in Vonderach et al. (2018) with slightly updated parameters as in Vonderach and Kändler (2021); species mapping as in `TapeS::BaMap(, type=7)`;

Value

a numeric matrix holding component biomass

References

Vonderach, C., G. Kändler and C. F. Dormann (2018). "Consistent set of additive biomass functions for eight tree species in Germany fit by nonlinear seemingly unrelated regression." Annals of Forest Science 75(2): 49. doi: 10.1007/s1359501807284

Vonderach, C. and G. Kändler (2021). Neuentwicklung von Schaftkurven- und Biomassemodellen für die Bundeswaldinventur auf Basis des TapeR-Pakets - Abschlussbericht zum Projekt BWI-TapeR. Freiburg: 150p.

Examples

```
nsur(spp = c(1, 6),
     dbh = c(30, 30),
     ht = c(25, 27),
     sth = c(0.25, 0.27),
     d03 = c(27, 27),
     kl = .7*c(25, 27))
```

---

nsur2 *Component biomass functions*

Description

evaluation of the component biomass functions fit by nonlinear seemingly unrelated regression (NSUR) to estimate absolute or relative component mass

Usage

```
nsur2(spp, dbh, ht)
```

Arguments

spp	vector of species code for biomass component function of interval [1;8]; see BaMap for mapping of species model codes
dbh	vector of diameter in breast height; in centimeter
ht	vector of tree heights, in meter

Details

simple function from Vonderach et al. (2018) to calculate component biomass; species mapping as in `TapeS::BaMap(, type=7)`

Value

a numeric matrix holding component biomass

References

Vonderach, C., G. Kändler and C. F. Dormann (2018). "Consistent set of additive biomass functions for eight tree species in Germany fit by nonlinear seemingly unrelated regression." Annals of Forest Science 75(2): 49. doi: 10.1007/s1359501807284

Examples

```
nsur2(spp = c(1, 6),
      dbh = c(30, 30),
      ht = c(25, 27))
```

NSURvar *estimate variance components for component biomass functions*

Description

estimate variance components for component biomass functions

Usage

```
NSURvar(
  data,
  estBM = NULL,
  comp = NULL,
  interval = "confidence",
  level = 0.95,
  adjVarPar = TRUE,
  as.list = TRUE
)
```

Arguments

data	data / predictors given for prediction by nsur incl. species code for component biomass function, see BaMap.
estBM	estimated biomass components for which variance information is required, given as data.frame, possibly use df[,,drop=FALSE]
comp	which components are required, see tprBiomass
interval	either none, confidence or prediction
level	Tolerance / confidence level, defaults 0.95
adjVarPar	should the variance information be taken from stable models? defaults to TRUE
as.list	Should the return value be a list or rbind to a data.frame? Defaults to TRUE.

Details

Estimates confidence and prediction intervals according to the methods presented in Parresol (2001).

In case, adjVarPar = TRUE, the models with instable variance estimates like Silver fir, Scots pine, Maple and Ash are, firstly, fitted by Norway spruce and European beech, respectively, and, secondly, adjusted to the expected value of the species specific model by substracting the difference to the first model. With that, more stable and imho more realistic confidence and prediction intervals are given. True, this assumes comparability of the variances between species.

Value

a data.frame with information on lower and upper bound of required interval as well as the (given) estimate and the respective mean squared error

Examples

```
d1 <- seq(42, 56, 2)
h <- estHeight(d1, 1)
data <- data.frame(spp = 1:8, # from BaMap(1, 7)
                   dbh = d1,
                   ht = h,
                   sth = 0.01*h,
                   D03 = 0.8 * d1,
                   kl = 0.7 * h)
estBM <- as.data.frame( nsur(spp = data$spp,
                             dbh = data$dbh,
                             ht = data$ht,
                             sth = data$sth,
                             d03 = data$D03,
                             kl = data$kl) )
estBM$agb <- rowSums(estBM[, -which(colnames(estBM)=="id")])
comp = c("sw", "agb")
interval = "confidence"
level = 0.95
adjVarPar = TRUE
e1 <- TapeS:::NSURvar(data, estBM, comp, interval="confidence", level=0.95, adjVarPar = TRUE)
e2 <- TapeS:::NSURvar(data, estBM, comp, interval="confidence", level=0.95, adjVarPar = FALSE)
```

```
## Not run:
par(mfrow=c(1, 2))
plot(x = data$dbh, y = e1$agb_ECBM, main="adjusted Var-Parameter", pch=data$spp,
     ylim=c(0.5*min(e1$agb_ECBM), 1.2*max(e1$agb_ECBM)), las=1,
     ylab="estimated AGB", xlab = "DBH [cm]")
invisible(sapply(1:nrow(e1), function(a){
  # a <- 1
  # lines(x = rep(data$dbh[a], 2), y = c(e2$agb_lwr[a], e2$agb_upr[a]),
  #       col="blue", lwd=2)
  rect(xleft = data$dbh[a] - 0.1, xright = data$dbh[a] + 0.1,
       ybottom = e2$agb_lwr[a], ytop = e2$agb_upr[a], border = "blue")
  lines(x = rep(data$dbh[a], 2), y = c(e1$agb_lwr[a], e1$agb_upr[a]),
        col="red", lwd=2)
}))
legend("bottomright", legend=c("Fi", "Ta", "Kie", "Dgl", "Bu", "Ei", "BAh", "Es"), pch=1:8)

## prediction intervals
e1 <- TapeS:::NSURvar(data, estBM, comp, interval="prediction", level=0.95, adjVarPar = TRUE)
e2 <- TapeS:::NSURvar(data, estBM, comp, interval="prediction", level=0.95, adjVarPar = FALSE)

plot(x = data$dbh, y = e1$agb_ECBM, main="adjusted Var-Parameter", pch=data$spp,
     ylim=c(0, 2*max(e1$agb_ECBM)), las=1,
     ylab="estimated AGB", xlab = "DBH [cm]")
invisible(sapply(1:nrow(e1), function(a){
  # a <- 1
  # lines(x = rep(data$dbh[a], 2), y = c(e2$agb_lwr[a], e2$agb_upr[a]),
  #       col="blue", lwd=2)
  rect(xleft = data$dbh[a] - 0.1, xright = data$dbh[a] + 0.1,
       ybottom = e2$agb_lwr[a], ytop = e2$agb_upr[a], border = "blue")
  lines(x = rep(data$dbh[a], 2), y = c(e1$agb_lwr[a], e1$agb_upr[a]),
        col="red", lwd=2)
}))
legend("topleft", legend=c("Fi", "Ta", "Kie", "Dgl", "Bu", "Ei", "BAh", "Es"), pch=1:8)

## one species, large diameter range
spp <- 1 # spruce
spp <- 5 # beech
spp <- 2 # silver fir
spp <- 8 # ash
d1 <- seq(7, 80, 2)
h <- estHeight(d1, spp)
data <- data.frame(spp = spp,
                   dbh = d1,
                   ht = h,
                   sth = 0.01*h,
                   D03 = 0.8 * d1,
                   kl = 0.7 * h)
estBM <- as.data.frame( nsur(spp = data$spp,
                             dbh = data$dbh,
                             ht = data$ht,
                             sth = data$sth,
                             d03 = data$D03,
```

```
                              kl = data$kl) )
estBM$agb <- rowSums(estBM[, -which(colnames(estBM)=="id")])
comp = c("sw", "agb")
interval = "confidence"
level = 0.95
adjVarPar = TRUE
e1 <- TapeS:::NSURvar(data, estBM, comp, interval="confidence", level=0.95, adjVarPar = TRUE)
e2 <- TapeS:::NSURvar(data, estBM, comp, interval="confidence", level=0.95, adjVarPar = FALSE)

par(mfrow=c(1, 2))
plot(x = data$dbh, y = e1$agb_ECBM, main="adjusted Var-Parameter", pch=data$spp,
     ylim=c(0.5*min(e1$agb_ECBM), 1.2*max(e1$agb_ECBM)), las=1,
     ylab="estimated AGB", xlab = "DBH [cm]")
invisible(sapply(1:nrow(e1), function(a){
  # a <- 1
  # lines(x = rep(data$dbh[a], 2), y = c(e2$agb_lwr[a], e2$agb_upr[a]),
  #       col="blue", lwd=2)
  rect(xleft = data$dbh[a] - 0.1, xright = data$dbh[a] + 0.1,
       ybottom = e2$agb_lwr[a], ytop = e2$agb_upr[a], border = "blue")
  lines(x = rep(data$dbh[a], 2), y = c(e1$agb_lwr[a], e1$agb_upr[a]),
        col="red", lwd=2)
}))


## prediction intervals
e1 <- TapeS:::NSURvar(data, estBM, comp, interval="prediction", level=0.95, adjVarPar = TRUE)
e2 <- TapeS:::NSURvar(data, estBM, comp, interval="prediction", level=0.95, adjVarPar = FALSE)

plot(x = data$dbh, y = e1$agb_ECBM, main="adjusted Var-Parameter", pch=data$spp,
     ylim=c(0, 2*max(e1$agb_ECBM)), las=1,
     ylab="estimated biomass", xlab = "DBH [cm]")
invisible(sapply(1:nrow(e1), function(a){
  # a <- 1
  # lines(x = rep(data$dbh[a], 2), y = c(e2$agb_lwr[a], e2$agb_upr[a]),
  #       col="blue", lwd=2)
  rect(xleft = data$dbh[a] - 0.1, xright = data$dbh[a] + 0.1,
       ybottom = e2$agb_lwr[a], ytop = e2$agb_upr[a], border = "blue")
  lines(x = rep(data$dbh[a], 2), y = c(e1$agb_lwr[a], e1$agb_upr[a]),
        col="red", lwd=2)
}))

## End(Not run)
```

`parSort` *constructor for class parSort*

Description

function to call `new()` on class `parSort`

Usage

```
parSort(
  n = 1,
  stH = 0,
  Lxh = 0,
  Hkz = 0L,
  Skz = 0L,
  Hsh = 0,
  Zsh = 0,
  Lsh = 0,
  Zab = 14,
  Lab = 0,
  Az = 0,
  LIh = 0,
  trL = 0,
  fixN = 0L,
  fixL = 0,
  fixZ = 0,
  fixA = 0,
  fixR = 0,
  ...
)
```

Arguments

n	the number of parameter sets to generate, defaults to 1
stH	stump height
Lxh	length of unusable wood at stem foot, see details
Hkz	height indicator, see details
Skz	stem indicator, see details
Hsh	height of stem wood, see details
Zsh	cutting diameter of stem wood, see details
Lsh	length of stem wood, see details
Zab	cutting diameter of upper trunk, see details
Lab	length of upper trunk, see details
Az	minimal cutting diameter, defaults to 7cm, see details
LIh	length of industrial wood, see details
trL	maximum transport length
fixN	number of fixed length assortments, see details
fixL	length of fixed length assortments, see details
fixZ	cutting diameter of fixed length assortments, see details
fixA	absolute add-on for good measure of fixed length assortments, given in cm; see details

fixR	relative add-on for good measure of fixed length assortments, given in percentage, i.e. 1% = 1; see details
...	currently unused

Details

if n is not given (or one) and any of the other parameter is given with length greater than one, n is reset to the maximum length of all parameters; care should be taken when using n and individual parameter setting for several trees.

Value

an object of class parSort, i.e. a list, each element of length n or maximum of length of defined parameters

---

parSort-class *An S4 class to represent the parameters for tree assorting.*

Description

This class represents one or multiple parameter sets holding the necessary information to specify the assortment process.

using indices i and j to subset

Usage

```
## S4 method for signature 'parSort,ANY,ANY,ANY'
x[i, j, ..., drop = FALSE]
```

Arguments

x	object from which to extract
i	index i
j	index j
...	not currently used
drop	drop dimensions, defaults to FALSE

Details

The assortment process is defined by several parameters. These follow the specification of its ancestor BDAT , but are extended to allow for fix length assortments at the tree top (industrial wood / pulp wood) and relaxes transport length and stump height.

- stH: stump height, defaults to 0, i.e. 1% of tree height
- Lxh: length of unusable wood at stem foot [m], defaults to 0 (X-Holz)
- Hkz: indicator for tree top, 0 - normal, 1 - Wipfelbruch, 2 - Gipfelbruch

  - 0 => H=H (default)
  - 1 => H=H+2
  - 2 => DBH < 30 => H=DBH; dbh > 30 => H = 30 + (DBH-30) * 0.3
- Skz: indicator for stem type, defaults to 0
  - 0 => conifer trees => no assortment restriction; deciduous trees => no assortments
  - 1 => monopodial deciduous trees => Hsh = 0.7*H
  - 2 => branching between dbh and 7m => Hsh = 5m
  - 3 => crown base < 3m => Hsh=0.1
  - 4 => dead or broken stem => Az = H*0.7
  - 5 => dead tree => non-usable wood
- Hsh: usable stem height, defaults to 0, i.e. 0.7*H
- Zsh: minimum cutting diameter under bark for stem wood [cm], defaults to 0, using parameter `Az` if estimated length < maximum length (i.e. 20m)
- Lsh: length of stem wood, defaults to 0, i.e. length unrestricted
- Zab: minimum cutting diameter under bark for top segment [cm], defaults to 0, i.e. 14cm under bark
- Lab: length of top segment, defaults to 0, i.e. length unrestricted
- Az: minimum cutting diameter over bark [cm], defaults to 0, using an exponential function given DBH to estimate Az
- LIh: length of industrial wood [m], defaults to 0, i.e. length unrestricted
- trL: maximum transport length of assortments, defaults to 0, i.e. 19m
- fixN: number of fixed length assortments at stem foot, defaults to 0 (no fixed length assortments, irrespective of other fix* parameters)
- fixZ: mininum diameter under bark for fixed length assortment at stem foot, defaults to 0
- fixL: length of fixed length assortment at stem foot, defaults to 0
- fixA: fixed length assortement add-on in [cm], defaults to 0
- fixR: fixed length assortement add-on in [%], defaults to 0

Value

a part of the original object

Methods (by generic)

- `x[i`: subsetting for class 'parSort'

Slots

`stH` stump height

`Lxh` length of unusable wood at stem foot, see details

`Hkz` height indicator, see details

`Skz` stem indicator, see details

`Hsh` height of stem wood, see details

`Zsh` cutting diameter of stem wood, see details

`Lsh` length of stem wood, see details

`Zab` cutting diameter of upper trunk, see details

`Lab` length of upper trunk, see details

`Az` minimal cutting diameter, defaults to 7cm, see details

`LIh` length of industrial wood, see details

`trL` maximum transport length

`fixN` number of fixed length assortments, see details

`fixL` length of fixed length assortments, see details

`fixZ` cutting diameter of fixed length assortments, see details

`fixA` absolute add-on for good measure of fixed length assortments, see details

`fixR` relative add-on for good measure of fixed length assortments, see details

Examples

```
parSort()
parSort(Lxh=1)
parSort(n=2)
```

`petterson` *height estimation*

Description

height estimation based on diameter in breast height and species using a Petterson-function

Usage

```
petterson(sp, d13)
```

Arguments

`sp`	vector of species code for biomass function from interval [1;18]; see `BaMap` for mapping of species model codes
`d13`	vector of diameter in breast height; in centimeter

Value

a scalar: tree height

`plot.tprTrees` *Plot taper curve for an object of class* `tprTrees`

Description

creating a plot of the taper curve of a tree, over or under bark

Usage

```
## S3 method for class 'tprTrees'
plot(
  x,
  bark = NULL,
  col.bark = NULL,
  obs = FALSE,
  assort = NULL,
  legend = FALSE,
  ...
)
```

Arguments

`x`	an object of class 'tprTrees'
`bark`	either NULL or logical; if TRUE taper curve over bark is plotted, if FALSE taper curve under bark is plotted; if NULL, both are plotted
`col.bark`	color to be used for plot of bark, if plot of taper curve over and under bark is requested
`obs`	should observations (measured/observed diameters) be added to the plot?
`assort`	assortments produced by `tprAssortment(, value="merge")`
`legend`	logical, if legend should be added
`...`	further arguments for `plot` and `points`

Details

plots the taper curve of a tree. Either over bark or under bark, or both. Elements design can partly be chosen. If assortments are given, these are added to the plot. Doing that, the assortment bottom and top position is indicated by a vertical line and mid-diameter is shown as a point with vertical dashed line. N.B. the mid-diameter shown is under bark and rounded downwards for 0.5 cm if mid-diameter < 20 and for 0.75 cm if bigger. Volume is calculated using this diameter. Reason for that behaviour is that assortment information with regard to diameter and volume reflects the legal rules for roundwood assortments (german RVR). Additionally, assortment names are indicated. One can provide assortment names in a column of `assort` named 'assortname', which will be used if available, otherwise the 'Sort'-column will be used. See Examples.

Value

No return value, called for side effects

Examples

```
## plotting the taper curve of a tree
oldpar <- par()
par(mfrow = c(1, 1))
tree <- tprTrees(spp=1L, Dm=40, Hm=1.3, H=35)
plot(tree, type = "l", las = 1, legend = TRUE)
plot(tree, bark = TRUE, las = 1)
plot(tree, bark = FALSE, las = 1, obs=TRUE) # obs incl. bark!!!
tree <- tprTrees(spp=c(1, 1), Dm = c(40, 35), Hm=c(1.3, 1.3), H = c(35, 30))
plot(tree, bark = FALSE, las = 1, legend = TRUE) # both trees are plotted
plot(tree, bark = TRUE, las = 1, legend = TRUE, obs=TRUE)

tree <- tprTrees(spp=1L, Dm=c(40, 32), Hm=c(1.3, 10.5), H=35)
plot(tree, type = "l", las = 1, legend = TRUE, obs=TRUE)

## if monotonicity is not forced:
tree <- tprTrees(spp=3L, Dm=8, Hm=1.3, H=10)
plot(tree, type = "l", las = 1, obs=TRUE, mono=FALSE)
plot(tree, type = "l", las = 1, obs=TRUE, mono=TRUE) # default

tree <- tprTrees(spp=c(1, 8), Dm = c(40, 40), Hm=c(1.3, 1.3), H = c(35, 35))
plot(tree, bark = NULL, las = 1, col.bark = "blue", legend = TRUE)
plot(tree, bark = NULL, las = 1, col.bark = "blue", legend = TRUE, obs = TRUE)
plot(tree[1, ], main = tprSpeciesCode(spp(tree[1, ]), out = "long"))
plot(tree[2, ], main = tprSpeciesCode(spp(tree[2, ]), out = "scientific"))
par(mfrow = c(1, 2))
plot(tree, bark = TRUE, las = 1)

## now add assortments into taper curve
par(mfrow = c(1, 1))
pars <- parSort(n=length(tree), Lxh=1, fixN=2, fixL=4, fixA=10)
ass <- tprAssortment(tree, pars=pars)
plot(tree, assort = ass)
plot(tree, bark = FALSE, assort = ass)
plot(tree, bark = FALSE, assort = ass, legend = TRUE)
plot(tree[1, ], assort = ass[ass$tree == 1, ], main = "first tree in subset")
plot(tree[2, ], assort = ass[ass$tree == 2, ], main = "second tree in subset")

## adding own assortment labels using column 'assortname'
ass$assortname <- ifelse(grepl("fix", ass$sort), paste0("Fix:", ass$length), ass$sort)
plot(tree, assort = ass)
par(oldpar)
```

RiPar *parameter of bark function*

Description

extract parameter of bark functions according to Altherr et al. 1974 - 1979

Usage

```
RiPar(ba = NULL, fn = NULL, par = NULL)
```

Arguments

ba	tree species code; returned by BaMap
fn	function number; see details
par	parameter; see details

Details

Function extracts the parameter according to tree species, function type and parameter number. There are three parameters in each of four functions. The first one refers to butt log (dt. Erdstamm), the second to middle log (dt. Mittelstammstück), the third to the top log (dt. Gipfelstammstück) and the fourth to the complete stem (dt. Gesamtstamm).

---

setTapeSoptions *Set and get options for the TapeS-package*

Description

Function to set and get options on how the TapeS-package works.

Usage

```
setTapeSoptions(Rfn = list(fn = "sig2"), mono = TRUE)

getTapeSoptions(name = NULL)
```

Arguments

Rfn	setting for residuals error matrix, defaults to "sig2", see details.
mono	logical, defaults to true. If calibrated taper curve is non-monotonic at stem base, an support diameter is added.
name	name of options to be returned

Details

So far, only two options are implemented: `TapeS_Rfn` and `TapeS_mono`. Teh first defaults to "sig2" (i.e. 'sigma squared') and the second to "TRUE".

The TapeR-taper curves can be evaluated in basically two ways: (i) either as defined in the TapeR-package, i.e. the diameters and volumes are estimated using the estimated error structure and find an optimal taper curve given the measured diameters or (ii) by interpolating the measured diameters, i.e. forcing the estimated taper curve through those measurements by setting the residual error structure to zero. See Kublin et al. (2013), p.987 bottom left. Technically, forcing the taper curve through the measurements is achieved by setting the residual error matrix R to zero, that is `Rfn = list(fn="zero")`. Defaults to `Rfn = list(fn="sig2")`. Besides, one can defined other functions about assumptions about the errors at the measurement positions, see `resVar` for options.

NB: Caution is required in applying `Rfn=list(fn="zero")`, since forcing the taper curve through too many points might lead to singularities or implausible results!

The option 'mono=TRUE' assures that no taper curve is generated which shows lower diameter in lower heights, possibly adding a support diameter at 1% of tree height.

Value

by defaults, sets `options()$TapeS_Rfn` to "sig2"

References

Kublin, E., Breidenbach, J., Kaendler, G. (2013) A flexible stem taper and volume prediction method based on mixed-effects B-spline regression, Eur J For Res, 132:983-997.

Examples

```
## reset option TapeS_Rfn to "sig2", i.e. model based errors by
setTapeSoptions(Rfn = list(fn="sig2"))
## or to force the taper curve through the  measurements, set
options("TapeS_Rfn" = list(fn="zero"))
## see the actual state of options by
options()[grep("^TapeS_", names(options()))]
## or easier
getTapeSoptions()
```

---

simTrees *simulating objects of class* `tprTrees`

Description

Function to simulate an object of class `tprTrees`

Usage

```
simTrees(par = NULL)
```

Arguments

par	list of lists, one for each species

Details

Function simulates trees based on given distributions and petterson height function. Dbh can be simulated using normal ('norm'), weibull or gamma distribution. Others might be added.

The par-list of each species needs the following named entries: spp - species code, n - number of trees, ddist - distribution of dbh, dpar - list of parameter of the distribution, i.e. mu and sd for normal distribution and shape and scale for weibull and gamma distribution. The latter both might use lag to offset the estimated diameter by this amount.

Value

an object of class tprTrees

See Also

petterson for the implemented height function and dnorm, dweibull and dgamma for the diameter distributions.

Examples

```
par <- list(list(spp=1, n=10, ddist="norm", dpar=list(mu=30, sd=4)),
            list(spp=3, n=5, ddist="norm", dpar=list(mu=40, sd=2)))
simTrees(par)
```

Slot_accessors *slot accessor functions for class 'tprtrees'*

Description

get and set slot values

Usage

```
spp(obj)

## S4 method for signature 'tprTrees'
spp(obj)

spp(obj) <- value

## S4 replacement method for signature 'tprTrees'
spp(obj) <- value
```

```
Dm(obj)

## S4 method for signature 'tprTrees'
Dm(obj)

Dm(obj) <- value

## S4 replacement method for signature 'tprTrees'
Dm(obj) <- value

Hm(obj)

## S4 method for signature 'tprTrees'
Hm(obj)

Hm(obj) <- value

## S4 replacement method for signature 'tprTrees'
Hm(obj) <- value

Ht(obj)

## S4 method for signature 'tprTrees'
Ht(obj)

Ht(obj) <- value

## S4 replacement method for signature 'tprTrees'
Ht(obj) <- value

sHt(obj)

## S4 method for signature 'tprTrees'
sHt(obj)

sHt(obj) <- value

## S4 replacement method for signature 'tprTrees'
sHt(obj) <- value

mono(obj)

## S4 method for signature 'tprTrees'
mono(obj)
```

Arguments

obj	object of class 'tprtrees'

value	depending on slot, see details

Details

Getting and setting the values of the different slots of 'tprTrees'-objects. For slot mono no setting function has been defined, as this slot is computed by check_monotonicity and should not be reset by users.

Setting of spp requires mode integer. For convenience, value is coerced by as.integer.

Setting spp and H, a vector of length equal length(spp(obj)) is required.

For setting slots Dm and Hm value must be a list of vectors of length equal length(spp(obj)) and the length of each vector must correspond to the length of the vectors in Hm and Dm.

Value

the accessor functions return the value of the specified slot and the setting functions update the object

Functions

- spp(): getting slot 'spp' of obj
- spp(tprTrees): method for class 'tprTrees'
- spp(obj) <- value: setting 'spp' slot of object
- spp(tprTrees) <- value: method for class 'tprTrees'
- Dm(): getting slot 'Dm' of obj
- Dm(tprTrees): method for class 'tprTrees'
- Dm(obj) <- value: setting 'Dm' slot of object
- Dm(tprTrees) <- value: method for class 'tprTrees'
- Hm(): getting slot 'Hm' of obj
- Hm(tprTrees): method for class 'tprTrees'
- Hm(obj) <- value: setting 'Hm' slot of object
- Hm(tprTrees) <- value: method for class 'tprTrees'
- Ht(): getting slot 'Ht' of obj
- Ht(tprTrees): method for class 'tprTrees'
- Ht(obj) <- value: setting 'Ht' slot of object
- Ht(tprTrees) <- value: method for class 'tprTrees'
- sHt(): getting slot 'sHt' of obj
- sHt(tprTrees): method for class 'tprTrees'
- sHt(obj) <- value: setting 'sHt' slot of object
- sHt(tprTrees) <- value: method for class 'tprTrees'
- mono(): getting slot 'monotone' of obj
- mono(tprTrees): method for class 'tprTrees'

See Also

tprTrees-class, tprTrees

tprAssortment *Functions to calculate assortments for given tree*

Description

Function calculates assortments for given tree according to assortment specification

Usage

```
tprAssortment(obj, pars = NULL, mono = TRUE, Rfn = NULL)

## S4 method for signature 'tprTrees'
tprAssortment(obj, pars = NULL, mono = TRUE, Rfn = NULL)
```

Arguments

obj	an object of class 'tprTrees'
pars	parameters to specify assortments, see parSort
mono	logical, defaults to true. If calibrated taper curve is non-monotonic at stem base, a support diameter is added.
Rfn	Rfn setting for residuals error matrix, defaults to list(fn="sig2"), see resVar.

Value

a data.frame with columns tree: tree identifier, sort: assortment name, height: beginning of assortment along trunk, length: length of assortment, mdm: mid-diameter of assortment, zdm: top-diameter of assortment and vol: volume.

Methods (by class)

- tprAssortment(tprTrees): method for class 'tprTrees'

Examples

```
## conifer wood
obj <- tprTrees(spp=c(1, 8),
                Dm=list(30, 40),
                Hm=list(1.3, 1.3),
                Ht=c(30, 40))
tprAssortment(obj)
pars <- parSort(stH=0.2, Lxh=c(1, 1.5), fixN=2, fixL=4)
(ass <- tprAssortment(obj, pars))
plot(obj, assort = ass)

## deciduous wood
obj <- tprTrees(spp=c(15),
                Dm=list(40),
                Hm=list(1.3),
```

```
                Ht=c(40))
tprAssortment(obj)
pars <- parSort(n=length(obj), Lxh=c(1), Hsh=10, Az=10)
ass <- tprAssortment(obj, pars)
plot(obj, assort=ass)
```

`tprBark` *Functions to calculate double bark thickness for given diameter at height* `Hx`

Description

Funktion evaluates the double bark thickness models developed by Altherr et al (1974-79).

Usage

```
tprBark(obj, Hx, cp = TRUE, mono = TRUE, Rfn = NULL)

## S4 method for signature 'tprTrees'
tprBark(obj, Hx, cp = TRUE, mono = TRUE, Rfn = NULL)
```

Arguments

`obj`	object of class 'tprTrees'
`Hx`	height for which double bark thickness is required
`cp`	cartesian product, i.e. apply all `Hx` to all trees, defaults to TRUE
`mono`	logical, defaults to true. If calibrated taper curve is non-monotonic at stem base, an support diameter is added.
`Rfn`	Rfn setting for residuals error matrix, defaults to `list(fn="sig2")`, see `resVar`.

Value

double bark thickness [cm]

Methods (by class)

- `tprBark(tprTrees)`: method for class 'tprTrees'

References

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1974). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Kiefer, Buche, Hainbuche, Esche und Roterle. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1975). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Europäische Lärche,

Japanische Lärche, Schwarzkiefer, Stieleiche, Traubeneiche, Roteiche, Bergahorn und Linde. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1976). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Weymouthskiefer, Robinie, Bergulme, Birke, Marilandica-Pappel und Robusta-Pappel. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1978). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Fichte, Tanne, Douglasie und Sitka-Fichte. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Altherr, E., P. Unfried, J. Hradetzky and V. Hradetzky (1979). Statistische Rindenbeziehungen als Hilfsmittel zur Ausformung und Aufmessung unentrindeten Stammholzes. Neupotz-Pappel, Regenerata-Pappel, Kirsche, Spitzahorn, Feldahorn, Aspe, Weide, Flatterulme, Tulpenbaum u. Elsbeere. Freiburg i. Br., Forstl. Versuchs- u. Forschungsanst. Baden-Württenberg.

Examples

```
## calculating bark thickness depends on diameter estimation and hence on the
## assumed residual variance at calibration.
## can be Rfn=list(fn="sig2") (default), i.e. EBLUP estimation from taper curve
## or e.g. Rfn=list(fn="zero"), i.e. force taper curve through the given measurements
options("TapeS_Rfn") # "sig2", default in TapeS
tmp <- tprTrees()
Dm(tmp); Hm(tmp) # Dbh = D(Hx=1.3) = 30cm (measured)
Dbh(tmp) # estimated via EBLUP from taper curve
tprBark(tmp, Hx = c(1.3, 5)) # bark thickness corresponds to Dbh(tmp)
(d <- tprDiameter(tmp, Hx = c(1.3, 5), bark=TRUE)) ## predicted
bark(1, d[1], 1.3/30) # the same!
bark(1, d[2], 5/30) # the same!

## if using option TapeS_Rfn = list(fn="zero"), force taper curve through measurements
setTapeSoptions(Rfn = list(fn="zero"))
options()$TapeS_Rfn
tprBark(tmp, Hx = c(1.3, 5))
bark(1, 30, 1.3/30) # the same but different to above
bark(1, d[1], 1.3/30) # cf. above
bark(1, 28, 5/30) # the same but different to above
bark(1, d[2], 1.3/30) # cf. above
```

`tprBiomass` *total aboveground and component biomass*

Description

calculate total above ground and optionally component biomass for given trees

Usage

```
tprBiomass(
  obj,
  component = NULL,
  useNFI = TRUE,
  interval = "none",
  mono = TRUE,
  Rfn = NULL
)

## S4 method for signature 'tprTrees'
tprBiomass(
  obj,
  component = NULL,
  useNFI = TRUE,
  interval = "none",
  mono = TRUE,
  Rfn = NULL
)
```

Arguments

obj	object of class 'tprTrees'
component	component for which biomass should be returned. If NULL, total aboveground biomass is returned, if 'all', all components are returned. See details.
useNFI	if TRUE, agb is estimated by the NFI-functions and component estimates are scaled so that their sum (i.e. agb) equals the estimate of the NFI functions. If FALSE, the NSUR functions are used for agb and component estimates.
interval	character to indicate whether and which type of interval is required; one of none, confidence or prediction.
mono	logical, defaults to true. If calibrated taper curve is non-monotonic at stem base, a support diameter is added.
Rfn	Rfn setting for residuals error matrix, defaults to list(fn="sig2"), see resVar.

Details

The available components are agb (= total aboveground biomass), stw (=stump wood), stb (=stump bark), sw (=solid wood with diameter above 7cm over bark), sb (=bark of component sw), fwb (=fine wood incl. bark) and ndl (=needles), if applicable. The needles-component is set to zero for deciduous tree species, no mass for leaves is available. One can request 'all' components to receive all components.

Value

a vector in case agb or only one component is requested, otherwise a matrix with one row per tree

Methods (by class)

- `tprBiomass(tprTrees)`: method for class 'tprTrees'

References

Kändler, G. and B. Bösch (2012). Methodenentwicklung für die 3. Bundeswaldinventur: Modul 3 Überprüfung und Neukonzeption einer Biomassefunktion - Abschlussbericht. Im Auftrag des Bundesministeriums für Ernährung, Landwirtschaft und Verbraucherschutz in Zusammenarbeit mit dem Institut für Waldökologie und Waldinventur des Johann Heinrich von Thünen-Instituts, FVA-BW: 71.

Kaendler (2021): Biometrische Modelle für die Ermittlung des Holzvorrats, seiner Sortimentsstruktur und der oberirdischen Biomasse im Rahmen der Bundeswaldinventur. Allg. Forst- u. J.-Ztg., 191. Jg., 5/6 83

Vonderach, C., G. Kändler and C. Dormann (2018): Consistent set of additive biomass equations for eight tree species in Germany fitted by nonlinear seemingly unrelated regression. Annals of Forest Science (2018) 75:49 doi: 10.1007/s13595-018-0728-4

Examples

```
obj <- tprTrees(spp=c(1, 15),
                Dm=list(c(30, 28), c(30, 28)),
                Hm=list(c(1, 3), c(1, 3)),
                Ht = rep(30, 2))
(tmp <- tprBiomass(obj, component="all"))

tprBiomass(obj, component=NULL) # aboveground biomass
component <- c("agb", "sw", "sb", "ndl")
tprBiomass(obj, component=component)
component <- c("sw", "sb", "ndl")
tprBiomass(obj, component="all")
# use NSUR-functions from Vonderach et al. 2018
# obs: currently sth=1% of tree height
# and kl=70% of tree height
tprBiomass(obj, component="all",  useNFI = FALSE)

## getting confidence and prediction intervals
useNFI <- FALSE
interval <- "confidence"
component <- c("sw", "agb")
mono <- TRUE
Rfn <- NULL
tprBiomass(obj, component, useNFI, interval)
tprBiomass(obj, component, useNFI, interval="none")
tprBiomass(obj, component, useNFI=TRUE, interval)
tprBiomass(obj, component, useNFI=TRUE, interval="none")
```

`tprDiameter` *Functions to extract diameters from Taper curve*

Description

Function evaluates TapeR taper curve models for given trees according to species, required height and optionally substracts double bark thickness.

Usage

```
tprDiameter(
  obj,
  Hx,
  bark = TRUE,
  interval = "none",
  cp = TRUE,
  mono = TRUE,
  Rfn = NULL
)

## S4 method for signature 'tprTrees'
tprDiameter(
  obj,
  Hx,
  bark = TRUE,
  interval = "none",
  cp = TRUE,
  mono = TRUE,
  Rfn = NULL
)
```

Arguments

`obj`	object of class 'tprTrees'
`Hx`	vector of heights for which diameter w/ or w/0 bark are required
`bark`	should diameter over or under bark be returned?
`interval`	indicator about whether 'confidence' or 'prediction' intervals are required (defaults to 'none'), optionally function returns the mean squared error of the mean and predictions ('MSE').
`cp`	cartesian product, i.e. apply all `Hx` to all trees, defaults to TRUE
`mono`	logical, defaults to true. If calibrated taper curve is non-monotonic at stem base, a support diameter is added.
`Rfn`	Rfn setting for residuals error matrix, defaults to `list(fn="sig2")`, see `resVar`.

Details

Function evaluates taper curves at required height Hx. By default (cp==TRUE), the taper curve is evaluated at Hx for each tree. If cp==FALSE, each tree is evaluated at exactly one Hx (recycled if necessary). This feature is intended for situations where diameter in relative heights are required. Then, the recycling of one height Hx (e.g. 1.3m) is not possible, since relative heights depend on absolute tree height, which might be different for each tree. Hence a call like `tprDiameter(obj, Hx=0.3*Ht(obj), cp=FALSE)` is necessary.

Value

a matrix or data.frame depending on value of `interval`. If 'none' (the default), a matrix of size [length(obj@Ht), length(Hx)] is returned, otherwise a data.frame of size [length(obj@Ht) * length(Hx), 5]. The five columns hold a tree identifier, Hx, lower confidence/prediction interval, the estimated diameter and the upper confidence/prediction interval. In case 'interval=MSE' the returned columns contain a tree identifier, Hx, the estimated diameter and mean squared error (MSE) of the mean and of the prediction. Estimates and intervals include bark or not, depending on `bark`.

Methods (by class)

- `tprDiameter(tprTrees)`: method for class 'tprTrees'

See Also

`tprDiameterCpp` for a faster implementation if no confidence or prediction information are required and `tprBark` for the applied bark reduction.

Examples

```
## prediction for new tree using implemented 'TapeR' taper curve model
obj <- tprTrees(spp=c(1, 3),
                Hm=list(c(1.3, 5), c(1.3, 5)),
                Dm=list(c(27, 25), c(27, 25)),
                Ht=c(27, 27))
hx <- c(1.3, 5, 7)
## by default, Hx applied on each tree, i.e. result is a 2x3 matrix
tprDiameter(obj, Hx = hx)

## if cp=FALSE, each tree only 'sees' one Hx, i.e. results is a vector
## (obs: length of Hx must be identical to length of obj)
tprDiameter(obj, Hx = c(1.3, 5), cp=FALSE)
tprDiameter(obj, Hx = hx, bark = FALSE)
tprDiameter(obj, Hx = hx, interval = "confidence")
tprDiameter(obj, Hx = hx, bark = FALSE, interval = "prediction")
tprDiameter(obj, Hx = hx, interval = "MSE")
tprDiameter(obj, Hx = hx, bark=FALSE, interval = "MSE")

## here same behaviour, if cp=FALSE
tprDiameter(obj, Hx = c(1.3, 5), bark = FALSE,
            interval = "prediction", cp=FALSE)
## using Cpp-implementation
## faster, but no intervals available
```

```
tprDiameterCpp(obj, Hx = hx)
tprDiameterCpp(obj, Hx = c(1.3, 5), cp=FALSE)

## prediction for objects of class 'datBDAT':
if(require(rBDAT)){
  tree <- rBDAT::buildTree(list(spp=1, D1=20:30, H1=1.3, H2=50, H=20:30))
  tree <- bdat_as_tprtrees(tree)
  tprDiameter(tree, Hx = 1.3)
}
```

`tprDiameterCpp` *Function to extract diameters from Taper curve using Rcpp*

Description

This function uses Rcpp and C-code to implement the diameter estimation of package TapeR to allow for faster estimation if no interval information is required.

Usage

```
tprDiameterCpp(obj, Hx, bark = TRUE, cp = TRUE, mono = TRUE, Rfn = NULL)

## S4 method for signature 'tprTrees'
tprDiameterCpp(obj, Hx, bark = TRUE, cp = TRUE, mono = TRUE, Rfn = NULL)
```

Arguments

`obj`	object of class 'tprTrees'
`Hx`	vector of heights for which diameter are required
`bark`	should diameter over or under bark be returned?
`cp`	cartesian product, i.e. apply all `Hx` to all trees, defaults to TRUE
`mono`	logical to decide whether a supporting diameter should be added in case the taper curve is regarded as non-monotonic. Defaults to TRUE.
`Rfn`	setting for residuals error matrix, defaults to `"sig2"`, see details.

Details

Function evaluates taper curves at required height `Hx`. By default (`cp==TRUE`), the taper curve is evaluated at `Hx` for each tree. If `cp==FALSE`, each tree is evaluated at exactly one Hx (recycled if necessary). This feature is intended for situations where diameter in relative heights are required. Then, the recycling of one height Hx (e.g. 1.3m) is not possible, since relative heights depend on absolute tree height, which might be different for each tree. Hence a call like `tprDiameter(obj, Hx=0.3*Ht(obj), cp=FALSE)` is necessary.

Value

a vector, in case only one diameter (i.e. Hx) is required per tree (`cp=FALSE`) or a matrix of size `length(trees)` x `length(Hx)` (`cp=TRUE`).

Methods (by class)

- tprDiameterCpp(tprTrees): method for class 'tprTrees'

See Also

tprDiameter if confidence or prediction intervals are required.

Examples

```
obj <- tprTrees(spp=c(1 , 3),
                Hm=list(c(1.3, 5), c(1.3, 5)),
                Dm=list(c(27, 25), c(27, 25)),
                Ht=c(27, 27))
Hx <- seq(0, 1, 0.1)
tprDiameterCpp(obj, Hx = Hx)
tprDiameterCpp(obj, Hx = Hx, bark=FALSE)
tprDiameterCpp(obj, Hx = c(1, 2), bark=FALSE, cp=FALSE)


require(rbenchmark)
benchmark(tprDiameter(obj, Hx, bark = TRUE),
          tprDiameterCpp(obj, Hx, bark = TRUE),
          replications = 10000)[,1:4]
```

---

tprHeight *Estimate height for given diameter w/ or w/o bark*

Description

Function to extract the height of given diameter w/ or w/o bark from taper curve

Usage

```
tprHeight(obj, Dx, bark = TRUE, cp = TRUE, mono = TRUE, Rfn = NULL)

## S4 method for signature 'tprTrees'
tprHeight(obj, Dx, bark = TRUE, cp = TRUE, mono = TRUE, Rfn = NULL)
```

Arguments

obj	object of class 'tprTrees'
Dx	diameter for which height is required
bark	should given diameter be considered over or under bark?
cp	cartesian product, i.e. apply all Hx to all trees, defaults to TRUE
mono	logical, defaults to true. If calibrated taper curve is non-monotonic at stem base, a support diameter is added.
Rfn	Rfn setting for residuals error matrix, defaults to list(fn="sig2"), see resVar.

Value

estimated height of given diameter

Methods (by class)

- `tprHeight(tprTrees)`: method for class 'tprTrees'

See Also

`tprDiameter`, `tprDiameterCpp`

Examples

```
obj <- tprTrees(spp=c(1, 3, 8, 15),
                Dm=list(c(30, 28), c(30, 28), c(30, 28), c(30, 28)),
                Hm=list(c(1.3, 5), c(1.3, 5), c(1.3, 5), c(1.3, 5)),
                Ht = rep(30, 4))
tprHeight(obj, Dx = c(30, 7), bark=TRUE)
tprHeight(obj, Dx = c(30, 7), bark=FALSE)

## no cartesion product between obj and Dx, i.e. cp=FALSE
## Dx is recycled if necessary
tprHeight(obj, Dx = c(30, 7), bark=FALSE, cp=FALSE)
```

---

tprSpeciesCode *Get BDAT species code or transform it to a name.*

Description

Function to get BDAT species code, or transform it to a german or english name, possibly an abbreviated version or even a scientific name

Usage

```
tprSpeciesCode(inSp = NULL, outSp = NULL)
```

Arguments

`inSp`	species information given, either numeric or character
`outSp`	character vector of names, for which information should be returned

Details

The function matches inSp to outSp. Depending on inSp, being either a numeric vector of values between 1 and 36 or a character vector of species names. Possible names are those which could be return values. One can get all names and the respective species code by calling the function with inSP=NULL and outSP=NULL (the default).

English species names and codes are taken from https://www.forestry.gov.uk/pdf/PF2011_Tree_Species.pdf/$FILE/PF2011_T while slightly adjusting the codes to be unique compared to the german codes (e.g. European larch is now ELA instead of EL).

Any given species code outside the interval [1, 36] is given the code 1 (i.e. Norway spruce), while throwing a warning. If any inSp - name is invalid, i.e. not in species list, this throws an error.

All elements of outSp, which are not colnames of the default returned data.frame, are silently dropped.

Value

vector or data.frame, depending on length of 'outSp'.

Examples

```
tprSpeciesCode(inSp=NULL, outSp=NULL) ## the default
tprSpeciesCode() ## the same
tprSpeciesCode(outSp = "scientific")
tprSpeciesCode(inSp = c(1, 2)) ## giving codes
tprSpeciesCode(inSp = c(1, 2, -1, 37)) ## values outside [1, 36] are given code 1
tprSpeciesCode(inSp = c(1, 2), outSp = c("scientific")) ## output a vector
tprSpeciesCode(inSp = c("Bu", "Fi")) ## asking for codes of abbreviated german names
tprSpeciesCode(inSp = c("Bu", "Fi", "Bu")) ## order is preserved
tprSpeciesCode(inSp = c("Buche", "Fichte")) ## asking for codes of german names
tprSpeciesCode(inSp = c("BE", "NS")) ## ... abbreviated english names
tprSpeciesCode(inSp = c("beech", "Norway spruce")) ## ... english names
tprSpeciesCode(inSp = c("Fagus sylvatica", "Picea abies")) ### ... scientific names
## not run
## tprSpeciesCode(inSp = c("Fagus sylvatica", "Picea")) ## error, 2nd name wrong
## end not run
```

---

tprTrees *constructor for class tprTrees*

Description

constructor for class tprTrees

Usage

```
tprTrees(
  spp = 1L,
  Dm = list(c(30, 28)),
  Hm = list(c(1.3, 5)),
  Ht = 30,
  sHt = rep(0, length(Ht)),
  inv = NULL,
  Rfn = NULL,
  ...
)
```

Arguments

spp	species code, see tprSpeciesCode
Dm	measurements of diameter along trunk
Hm	height of measurements along trunk
Ht	tree height
sHt	standard deviation of stem height Ht. Can be 0 if height was measured without error.
inv	indicator (0-5) for inventory to assess taper form; numeric scalar see FormTariff
Rfn	function to populate residual variance matrix R
...	arguments to be passed to initialize()

Details

constructor for a tprTrees object, includes a check on monotonicity of the taper curve.

Value

object of class tprTrees.

Examples

```
# just define a tree
tpr <- tprTrees(spp=1, Dm=30, Hm=1.3, Ht=27)
plot(tpr)
# define 2 trees with only dbh
tpr <- tprTrees(spp=c(1,3), Dm=c(30, 35), Hm=c(1.3, 1.3), Ht=c(27, 30))
plot(tpr)
# define 2 trees with several measurement
tpr <- tprTrees(spp=c(1,3), Dm=list(c(30, 28), c(35, 33, 31)),
                Hm=list(c(1.3, 8), c(1.3, 5, 8)), Ht=c(27, 30))
plot(tpr)
# define 2 trees with only dbh and inventory indicator (form)
tpr <- tprTrees(spp=c(1,3), Dm=c(30, 35), Hm=c(1.3, 1.3), Ht=c(27, 30), inv=4)
plot(tpr)
```

`tprTrees-class` *An S4 class to represent one or multiple trees.*

Description

This class represents one or multiple trees by their biometric characteristics.

using indices i and j to subset

Usage

```
## S4 method for signature 'tprTrees,ANY,ANY,ANY'
x[i, j, ..., drop = FALSE]

## S4 method for signature 'tprTrees'
length(x)

## S4 method for signature 'tprTrees'
show(object)
```

Arguments

`x`	object of class 'tprTrees'
`i`	index i
`j`	index j
`...`	not currently used
`drop`	drop dimensions, defaults to FALSE
`object`	object of class 'tprTrees'

Details

blabla

Value

a part of the original object

Methods (by generic)

- `x[i`: subsetting for class 'tprTrees'
- `length(tprTrees)`: length function for class 'tprTrees'
- `show(tprTrees)`: length function for class 'tprTrees'

Slots

`spp` species code of trees

`Dm` list of measured diameters

`Hm` list of heights of measured diameters

`Ht` total height of trees

`sHt` standard deviation of total tree height, defaults to 0 for exact height measurements without error

`monotone` logical indicator about monotonicity of taper curve

Examples

```
tprTrees() # initialise object by constructor
(tmp <- tprTrees(spp=c(1L,3L), Dm=list(c(30, 28), c(40, 38)),
                 Hm=list(c(1.3, 5), c(1.3, 5)), Ht=c(30, 40)))
```

`tprVolume` *Functions to calculate stem volume from taper curve*

Description

Function calculates stem volume from taper curve for given trees, depending definition of segment and on bark indicator. It is possible to request confidence or prediction intervals.

Usage

```
tprVolume(
  obj,
  AB = NULL,
  iAB = NULL,
  bark = NULL,
  interval = "none",
  mono = TRUE,
  Rfn = NULL
)

## S4 method for signature 'tprTrees'
tprVolume(
  obj,
  AB = list(A = 0, B = 7, sl = 2),
  iAB = c("h", "dob"),
  bark = TRUE,
  interval = "none",
  mono = TRUE,
  Rfn = NULL
)
```

Arguments

obj	object of class 'tprTrees'
AB	list with heights or diameters A and B of section for which volume over or under bark should be calculated. Additionally, add in sl for the segment length over which the integral should be calculated. See details.
iAB	character indicating how to interpret given A and B values. Either "H" (the default), "Dob" (diameter over bark) or "Dub" (diameter under bark). Could be of length one or two, depending on whether A and B are both height or diameter variables or not. See examples.
bark	should volume be returned including (TRUE) or excluding bark (FALSE)?
interval	character to indicate whether and which type of interval is required; one of none, confidence or prediction.
mono	logical, defaults to true. If calibrated taper curve is non-monotonic at stem base, a support diameter is added.
Rfn	Rfn setting for residuals error matrix, defaults to list(fn="sig2"), see resVar.

Details

The function returns total solid wood w/ bark (i.e. from H=0 to D=7cm) by default. Using AB, one can specify lower A and upper B end of segments for which volume is required, w/ or w/o bark.

iAB can be a vector of length two, indicating how to interpret A and B. Hence, one can calculate volume between a given height and a given diameter, either over or under bark. If of length one, it is assumed that the indicator applies to both A and B.

Defining interval 'confidence' or 'prediction' returns lower (lwr) and upper (upr) interval bounds on confidence level α = qt(0.025, ...). NB: The volume confidence bounds only incorporate the uncertainty of diameter estimation at a pre-fixed position (e.g. H=1.3m). If the position is given as diameter (e.g. iAB="Dob"), the absolute height position is calculated using the *estimated* diameter, hence, the uncertainty of the estimated absolute height is not (yet) included. Neither is the uncertainty of the models for bark reduction.

In contrast to the underlying R-package TapeR, which uses E_VOL_AB_HmDm_HT.f for volume calculation, this function calculates volume based on stem-section (default: 2m, see parameter AB). Additionally, with that approach, bark reduction is easily possible.

Value

if interval='none' a vector else a matrix.

Methods (by class)

- tprVolume(tprTrees): method for class 'tprTrees'

See Also

E_DHx_HmDm_HT.f for the underlying diameter calculation.

Examples

```
obj <- simTrees() # default is: simulate 10 Norway spruce with mean dbh of 40
A <- 1
B <- 10
tprVolume(obj) # default is: coarse wood volume w/ bark
tprVolume(obj, AB = list(A=A, B=B, sl=2), iAB = "H", bark=FALSE)
tprVolume(obj, AB = list(A=A, B=B, sl=0.01), iAB = "H", bark=FALSE)
tprVolume(obj, AB = list(A=A, B=B, sl=0.01), iAB = "H", bark=TRUE)

## compare against integrated taper curve volume via package TapeR
## TapeR integrates over the taper curve, while TapeS uses segments of length 'sl'
SKP <- TapeS:::SKPar
TapeR::E_VOL_AB_HmDm_HT.f(Hm=obj@Hm[[1]], Dm = obj@Dm[[1]], iDH = "H",
                          mHt = obj@Ht[1], sHt = 0, A = A, B = B,
                          par.lme=SKP[[1]])$E_VOL

## returning intervals
tprVolume(obj, interval="none")
tprVolume(obj, interval="confidence")
tprVolume(obj, interval="prediction")
tprVolume(obj, interval="prediction", bark=FALSE)
tprVolume(obj, interval="prediction", AB=list(A=0.1, B=5.1, sl=0.1), iAB="H")
```

Vfm *tree volume information*

Description

Wrapper to get specific type of volume from taper curve

Usage

```
Vfm(obj)

Efm(obj, stH = 0.01)

VolR(obj)

VolE(obj)

VolFAO(obj)

Vfm_phys(obj)

Efm_phys(obj, stH = 0.01)
```

Arguments

obj	a object of class 'tprTrees'
stH	assumed or known relative or absolute stump height, from which volume calculation should starts, defaults to 0.01

Details

wrapper functions around tprVolume, which return specific definitions of stem volume.

Function Efm uses parameter stH to define starting point, i.e. stump height, of volume calculation. stH can be defined relative to total tree height (0 < stH <= 1) or in absolute measure (unit=cm) in case stH > 1

VolE calculates as the sum of volume of default assortments (stem wood, top log, industrial wood, X-wood, non-usuable wood according to RVR. For dbh < 7cm a linear regression is applied.

VolFAO calculates tree volume starting from stump up to tree top (in contrast to german definition, which uses D=7cm over bark), and includes bark component. Stump height is defined as 1% of tree height. Volume calculation is based on 2m-sections. For trees with dbh < 7cm, tabulated values are used, see Riedel et al. (2017) for details (e.g. p.35, table 5.6).

Vfm_phys is equal to Vfm, except that the taper curve is numerically integrated, by use of section length of 0.01m. This is relevant if biomass or nutrient export is to be calculate. Numerical integration is quite slow.

Efm_phys is equal to Efm, except that the taper curve is numerically integrated, by use of section length of 0.01m. This is relevant if biomass or nutrient export is to be calculate. Numerical integration is quite slow.

Value

vector of volume estimates

Functions

- Efm(): Efm, i.e. coarse wood excl. bark from Ht=stH*Ht to Dob=7cm
- VolR(): VolR: Volume from H=0 to D=7cm over bark, measured as 2m sections
- VolE(): VolE: sum of volume of default assortments according to RVR
- VolFAO(): VolFAO: from stump to tree top incl. bark; if dbh < 7cm using tabulated values
- Vfm_phys(): Vfm_phys physical volume of tree incl. bark from A=0
- Efm_phys(): Efm_phys physical volume of tree excl. bark from A=0.1*Ht

References

Riedel, T. and Hennig, P. and Kroiher, F. and Polley, H. and Schwitzgebel, F. (2017): Die dritte Bundeswaldinventur (BWI 2012). Inventur- und Auswertemethoden. 124 pages.

Examples

```
t <- tprTrees() # constructor of class 'tprTrees'
Vfm(t)
Efm(t)
Efm(t, stH=0.01) # stump height = 1\% of tree height
Efm(t, stH=10) # stump height=10cm
VolR(t)
VolE(t)
VolFAO(t)
Vfm_phys(t) # slower since much more evaluations of taper curve (every 1 cm)
Efm_phys(t, stH=0.01) # slower since much more evaluations of taper curve (every 1 cm)
```

Index