Package 'efdm'

August 16, 2021

Title Implement European Forestry Dynamics Model

Version 0.1.0

Description An implementation of European Forestry Dynamics Model (EFDM) and an estimation algorithm for the transition probabilities.
The EFDM is a large-scale forest model that simulates the development of the forest and estimates volume of wood harvested for any given forested area. This estimate can be broken down by, for example, species, site quality, management regime and ownership category.
See Packalen et al. (2015) <doi:10.2788/153990>.

URL `https://github.com/mikkoku/efdm`

BugReports `https://github.com/mikkoku/efdm/issues`

License GPL-2

Encoding UTF-8

LazyData true

Imports stats, utils, data.table

Suggests dplyr, ggplot2, gridExtra, knitr, rmarkdown, testthat

RoxygenNote 7.1.1

VignetteBuilder knitr

Depends R (>= 2.10)

NeedsCompilation no

Author Mikko Kuronen [aut, cre] (<`https://orcid.org/0000-0002-8089-7895`>),
Minna Räty [aut] (<`https://orcid.org/0000-0001-9898-8712`>)

Maintainer Mikko Kuronen <`mikko.kuronen@luke.fi`>

Repository CRAN

Date/Publication 2021-08-16 08:00:05 UTC

R topics documented:

build_complex_statespace

Add a statespace to an activity

Description

Add a statespace to an activity

Usage

```
build_complex_statespace(
  act,
  statespace0,
  statespace1,
  factors = character(),
  by = character()
)
```

Arguments

act	Activity definition
statespace0	data.frame Statespace before transition
statespace1	data.frame Statespace after transition
factors	character Variables used by the activity
by	character Variables that split the statespace

Details

In a complex statespace it is possible to change statespaces with an activity. Since statespace is the collection of classes of variables this means that the classification changes.

Value

Activity definition with statespace

See Also

build_state_space

build_statespace *Add a statespace to an activity*

Description

Add a statespace to an activity

Usage

```
build_statespace(act, statespace, factors = character(), by = character())
```

Arguments

act	Activity definition
statespace	data.frame
factors	character Variables used by the activity
by	character Variables that split the statespace

Details

Statespace is the collection of strata. When it is added to an activity defined by define_activity the following estimatetransprobs function has sufficient information to estimate transition probabilties.

factors and by variables are used by estimatetransprobs to estimate transition probabilities for strata. Observations with different levels of factors variables are used where as observations with different levels of by variables are never used.

If statespace is changing as a result of the activity see build_complex_statespace.

Value

Activity definition with statespace

Examples

```
statespacepine <- expand.grid(species="pine", vol=1:5, age=1:3, stringsAsFactors=FALSE)
statespacespruce <- expand.grid(species="spruce", vol=1:4, age=1:3, stringsAsFactors=FALSE)
statespace <- rbind(statespacepine, statespacespruce)
act <- define_activity("nomanagement", c("vol", "age"))
act <- build_statespace(act, statespace, by="species")
```

define_activity *Define an activity*

Description

Define an activity

Usage

```
define_activity(name, dynamicvariables, probname = name)
```

Arguments

`name`	Name of activity used in reporting
`dynamicvariables`	Names of variables where changes happen
`probname`	(optional) Name of activity in activity probabilities data

Details

The set of activies in EFDM defines all possible alternatives for a forest stratum to develop during a scenario run step. Therefore activities are not only limited to forest treatments and management actions such as thinnings and final fellings but should also include 'no management' i.e. growth, if applicable. An activity may also be something else affecting the development, for example, a forest hazard: snow, wind, drought, pest damage etc.

An activity is defined with this function. A name, which is henceforth used in the EFDM R project when refering to the activity, is given. In addition, the (stetaspace) variables which are affected by the activity are named. Typically an activity affects on the age, volume or stem count of the forest, but an activity may also, for example, change land-use and then a variable related to land-use categories is essential. If the activity name does not match the name in the activaty probability data set, those can be linked here.

Defining an activity is the first step, which will be followed by

- `build_statespace` or `build_complex_statespace` and `estimatetransprobs` if using pairdata
- `transprobs<-` if not using pairdata

until an activity is fully applicable in runEFDM.

Value

An incomplete activity definition that needs to be completed with transition probabilities, see Details.

Examples

```
define_activity("nomanagement", c("vol", "age"))
```

`estimatetransprobs` *Estimate Transition Probabilities from Pairdata*

Description

Estimate Transition Probabilities from Pairdata

Usage

```
estimatetransprobs(act, pairdata, prior)
```

Arguments

`act`	Activity definition with statespace
`pairdata`	`data.frame` Observed transitions
`prior`	function or character

Details

Transition probabilities 'move' the forest areas allocated in the cells of state matrix from the initial states in the beginning of a EFDM run step to the end position. This end position will be the initial state of the next EFDM step. Length of a step (=time) in the EFDM run is typically determined by the pairdata. It is the time difference of tree observations. Note that the pairdata can be also constructed from single observation, if the other (pair) observation is estimated or modelled.

Each activity needs to have a transition probability. If no pairdata is available, transition probability matrices can be based entirelly on a prior defined with expert knowledge.

The estimation uses an iterative Bayesian algorithm that is explained in `https://github.com/ec-jrc/efdm/blob/master/documents/EFDMinstructions/Seija_Mathematics_behind_EFDM.pdf`. The transition probability estimate is the proportion of observed transitions divided by the number of all transitions from the same starting state. `prior` gives the number of prior transitions. For each `factor` variable the transitions are counted in classes of all factors before the current factor. The "most important" observations (having all classes right) is counted `length(factors)` times, the second most important observations are counted `length(factors)-1` times and so on.

If pairdata is NULL prior is used by itself.

Observations should have 'factor' and 'by' variables and statepairs with 0 and 1 suffixes to indicate before and after observations.

The estimation algorithm uses information across 'factor' variables, but not across 'by' variables.

`prior` can either `character` or `function`.

- "nochange" implies that there is one observation where state doesn't change
- "uninformative" when no observations are given all states are as likely
- `function(A,dynvar1,dynvar0)` where A is an array of zeros with dimnames(A) <- c(dynvar1, dynvar0). The function should fill A with the number of prior transitions and return it.

Value

Activity definition with transition probabilities

Examples

```
# Estimation can use observed transitions with different levels of factors.
statespace <- expand.grid(a=1:2, b=1:2, vol=1:5)
pairdata <- data.frame(a=c(1,1,2,2), b=c(1,2,1,2), vol0=c(1,1,1,1), vol1=c(2,3,4,5))
state0 <- statespace
actprob <- statespace
actprob$test <- 1
state0$area <- 0
state0$area[1] <- 1

# With by=c("a", "b") there are two observations: one from prior and the other
# from the exact combination of class levels.
act <- define_activity("test", c("vol"))
act <- build_statespace(act, statespace, by=c("a", "b"))
act1 <- estimatetransprobs(act, pairdata, "nochange")
runEFDM(state0, actprob, list(act1), 1)

act <- define_activity("test", c("vol"))
act <- build_statespace(act, statespace, factors="a", by="b")
act2 <- estimatetransprobs(act, pairdata, "nochange")
runEFDM(state0, actprob, list(act2), 1)

act <- define_activity("test", c("vol"))
act <- build_statespace(act, statespace, factors="b", by="a")
act3 <- estimatetransprobs(act, pairdata, "nochange")
runEFDM(state0, actprob, list(act3), 1)

# The order of variables in factors argument specifies the order of importance.
# Observation that differ in the first variable are counted more times.
act <- define_activity("test", c("vol"))
act <- build_statespace(act, statespace, factors=c("a", "b"))
act4 <- estimatetransprobs(act, pairdata, "nochange")
runEFDM(state0, actprob, list(act4), 1)

act <- define_activity("test", c("vol"))
act <- build_statespace(act, statespace, factors=c("b", "a"))
act5 <- estimatetransprobs(act, pairdata, "nochange")
runEFDM(state0, actprob, list(act5), 1)
```

example *Example dataset.*

Description

A listt containing the necessary datasets for EFDM forest scenario example.

Usage

```
example
```

Format

A list of data frames:

actprob Activity probabilities

noman_pairs Pair data for growth without management activities

thin_pairs Pair data for thinning

initial_state Initial forest state

drain_coef Coefficient to transform harvested areas into harvest accumulation by timber assortments

vol_coef Coefficients to transform volume classes into volumes m3/ha

income_coef Coefficients to transform harvest accumulation into income

---

prior_ff *Priors for* `estimatetransprobs`

Description

Priors for `estimatetransprobs`

Usage

```
prior_ff()

prior_grow(variable, howmuch = 1)
```

Arguments

`variable`	Name of the variable to grow
`howmuch`	Amount of growth

Details

`prior_ff` moves the forest area to the smallest classes of the given dynamic variables of the forest stratum.

`prior_grow` moves the forest to another class given by increasing `variable` by `howmuch`.

Value

Return value is used by estimatetransprobs to provide prior information on the transition probabilities.

Examples

```
statespace <- expand.grid(a=1:2, b=1:2, vol=1:15, age=1:35)
act <- define_activity("test", c("vol", "age"))
act <- build_statespace(act, statespace, by=c("a", "b"))
act1 <- estimatetransprobs(act, NULL, prior_ff())
act2 <- estimatetransprobs(act, NULL, prior_grow("age"))
```

runEFDM *Run European Forestry Dynamics Model*

Description

Run European Forestry Dynamics Model

Usage

```
runEFDM(state0, actprob, activities, n, check = TRUE)
```

Arguments

`state0`	`data.frame` Initial state
`actprob`	`data.frame` Activity probabilities
`activities`	`list` A list of activities
`n`	`integer` Number of time steps required
`check`	Check input arguments for consistency.

Details

This is the actual scenario running function, which projects the initial forest state n time steps to the future.

An activity is defined by activity name, names of dynamic variables and transition probabilities.

Value

`data.frame` State of each time step divided by activities

`transprobs<-` *Transition probabilities of an activity*

Description

Functions to get or set the transition probabilities of an activity

Usage

```
transprobs(act) <- value

transprobs(act)
```

Arguments

`act` Activity definition

`value` `data.frame` of transition probabilities

Details

The `value` should contain

- dynamic variables in the activity
- the probability of transition `prob`
- other variables affecting the transition probabilities.

Value

`data.frame` where prob=nobs/N is the transition probability from current state (with suffix 0) to next state (with suffix 1).

Examples

```
act1 <- define_activity(”test”, c(”vol”))
transprobs(act1) <- data.frame(vol0 = 1:5, vol1=c(2:5, 5), prob=1)
transprobs(act1)

if(require(”ggplot2”)) {
  ggplot(transprobs(act1)) + geom_raster(aes(x=vol0, y=vol1, fill=prob))
}
```

Index