Package ‘lefko3’

November 16, 2020

Type Package

Title Historical and Ahistorical Population Projection Matrix Analysis

Version 3.1.2

Date 2020-11-16

Description Creates matrix population models for use in population ecological analyses. Specializes on the construction of historical matrices, which are 2d matrices comprising 3 time intervals of demographic information. Estimates both raw and function-based forms of historical and standard ahistorical matrices. Methodology based on Ehrlen (2000) <doi:10.1890/0012-9658(2000)081[1675:TDOPPD]2.0.CO;2>.

Encoding UTF-8

License GPL (>= 2)

Imports Rcpp (>= 1.0.5), glmmTMB, lme4, MASS, MuMIn, pscl, stats, stringr

LinkingTo Rcpp, RcppArmadillo

LazyData true

RoxygenNote 7.1.1

Suggests knitr, popbio, rmarkdown

VignetteBuilder knitr

NeedsCompilation yes

Author Richard P. Shefferson [aut, cre]
(<https://orcid.org/0000-0002-5234-3131>),
Johan Ehrlen [aut] (<https://orcid.org/0000-0001-8539-8967>)

Maintainer Richard P. Shefferson `<cdorm@g.ecc.u-tokyo.ac.jp>`

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2020-11-16 08:00:05 UTC

R topics documented:

`aflefko2` *Create Function-based Ahistorical Age x Stage Matrix Projection Model*

Description

Function `aflefko2()` returns ahistorical age x stage MPMs corresponding to the patches and years given, including the associated component transition and fecundity matrices, a data frame detailing the characteristics of ahistorical stages, and a data frame characterizing the patch and year combinations corresponding to these matrices. Unlike `rlefko2()` and `rlefko3()`, this function currently does not currently distinguish populations.

Usage

```
aflefko2(
  year = 1,
  patch = NA,
  stageframe,
  repmatrix = NA,
  overwrite = NA,
  data = NA,
  modelsuite = NA,
  surv_model = NA,
  obs_model = NA,
  size_model = NA,
  repst_model = NA,
  fec_model = NA,
  jsurv_model = NA,
  jobs_model = NA,
  jsize_model = NA,
  jrepst_model = NA,
  paramnames = NA,
  inda = 0,
  indb = 0,
  indc = 0,
  surv_dev = 0,
  obs_dev = 0,
  size_dev = 0,
  repst_dev = 0,
  fec_dev = 0,
  jsurv_dev = 0,
  jobs_dev = 0,
  jsize_dev = 0,
  jrepst_dev = 0,
  repmod = 1,
  yearcol = "year2",
  patchcol = "patchid",
  year.as.random = FALSE,
  patch.as.random = FALSE,
  final_age = 10,
  continue = TRUE,
  randomseed = 0,
  negfec = FALSE,
  reduce = FALSE
)
```

Arguments

`year` A variable corresponding to year or observation time, or a set of such values, given in values associated with the year term used in linear model development. Can also equal `all`, in which case matrices will be estimated for all years. Defaults to `all`.

`patch` A variable designating which patches or subpopulations will have matrices estimated. Should be set to specific patch names, or to `all` if matrices should be estimated for all patches. Defaults to `all`.

`stageframe` A stageframe object that includes information on the size, observation status, propagule status, immaturity status, and maturity status of each ahistorical stage. Should also incorporate bin widths if size is continuous.

`repmatrix` A matrix composed mostly of 0s, with non-zero values for each potentially new individual (row) born to each reproductive stage (column). Entries act as multipliers on fecundity, with 1 equaling full fecundity.

`overwrite` A data frame developed with the `overwrite()` function describing transitions to be overwritten either with given values or with other estimated transitions.

`data` The original historical demographic data frame used to estimate vital rates (class `hfvdata`). The original data frame is required in order to initialize years and patches properly.

`modelsuite` An optional `lefkoMod` object holding the vital rate models. If given, then `surv_model`, `obs_model`, `size_model`, `repst_model`, `fec_model`, `jsurv_model`, `jobs_model`, `jsize_model`, `jrepst_model`, `paramnames`, `yearcol`, and `patchcol` are not required. No models should include size or reproductive status in time *t*-1.

`surv_model` A linear model predicting survival probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any survival probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

`obs_model` A linear model predicting sprouting or observation probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any observation probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

`size_model` A linear model predicting size. This can be a model of class `glm` or `glmer`, both of which require a predicted poisson variable under a log link, or a model of class `lm` or `lmer`, in which a Gaussian response is assumed. If given, then will overwrite any size model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

`repst_model` A linear model predicting reproduction probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any reproduction probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

`fec_model` A linear model predicting fecundity. This can be a model of class `glm` or `glmer`, and requires a predicted poisson variable under a log link. If given, then will overwrite any fecundity model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

`jsurv_model` A linear model predicting juvenile survival probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any juvenile survival probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

`jobs_model` A linear model predicting juvenile sprouting or observation probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any juvenile observation probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

`jsize_model` A linear model predicting juvenile size. This can be a model of class `glm` or `glmer`, both of which require a predicted poisson variable under a log link, or a model of class `lm` or `lmer`, in which a Gaussian response is assumed. If given, then will overwrite any juvenile size model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

`jrepst_model` A linear model predicting reproduction probability of a mature individual that was immature in the previous year. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any reproduction probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

`paramnames` A dataframe with two columns, the first showing the general model terms that will be used in matrix creation, and the second showing the equivalent terms used in modeling. Only required if `modelsuite` is not supplied.

`inda` A numeric value to use for individual covariate a. Defaults to 0.

`indb` A numeric value to use for individual covariate b. Defaults to 0.

`indc` A numeric value to use for individual covariate c. Defaults to 0.

`surv_dev` A numeric value to be added to the y-intercept in the linear model for survival probability.

`obs_dev` A numeric value to be added to the y-intercept in the linear model for observation probability.

`size_dev` A numeric value to be added to the y-intercept in the linear model for size.

`repst_dev` A numeric value to be added to the y-intercept in the linear model for probability of reproduction.

`fec_dev` A numeric value to be added to the y-intercept in the linear model for fecundity.

`jsurv_dev` A numeric value to be added to the y-intercept in the linear model for juvenile survival probability.

`jobs_dev` A numeric value to be added to the y-intercept in the linear model for juvenile observation probability.

`jsize_dev` A numeric value to be added to the y-intercept in the linear model for juvenile size.

`jrepst_dev` A numeric value to be added to the y-intercept in the linear model for juvenile reproduction probability.

`repmod` A scalar multiplier of fecundity. Defaults to 1.

`yearcol` The variable name or column number corresponding to year in time *t* in the dataset. Not needed if a `modelsuite` is supplied.

`patchcol` The variable name or column number corresponding to patch in the dataset. Not needed if a `modelsuite` is supplied.

`year.as.random` A logical term indicating whether coefficients for missing patches within vital rate models should be estimated as random intercepts. Defaults to FALSE, in which case missing time step coefficients are set to 0.

`patch.as.random` A logical term indicating whether coefficients for missing patches within vital rate models should be estimated as random intercepts. Defaults to FALSE, in which case missing patch coefficients are set to 0.

`final_age` The final age to model in the matrix, where the first age will be age 0.

`continue` A logical value designating whether to allow continued survival of individuals going past the final age, using the demographic characteristics of the final age.

`randomseed` A numeric value used as a seed to generate random estimates for missing time step and patch coefficients, if either `year.as.random` or `patch.as.random` is set to TRUE. Defaults to 0.

`negfec` A logical value denoting whether fecundity values estimated to be negative should be reset to 0. Defaults to FALSE.

`reduce` A logical value denoting whether to remove ahistorical stages associated solely with 0 transitions. These are only removed in cases where the associated row and column sums in ALL matrices estimated equal 0. Defaults to FALSE.

Value

If all inputs are properly formatted, then this function will return an object of class `lefkoMat`. Output includes:

`A` A list of full projection matrices in order of sorted patches and years.

`U` A list of survival-transition matrices sorted as in `A`.

`F` A list of fecundity matrices sorted as in `A`.

`hstages` Null for ahistorical matrices.

`ahstages` A data frame detailing the characteristics of associated ahistorical stages.

`labels` A data frame giving the patch and year of each matrix in order. In `aflefko2()`, only one population may be analyzed at once, and so `pop = NA`

`matrixqc` A short vector describing the number of non-zero elements in `U` and `F` matrices, and the number of annual matrices.

`modelqc` This is the `qc` portion of the modelsuite input.

Please note that this function will yield incorrect estimates if the models utilized incorporate state in time *t*-1. Only use models developed testing ahistorical effects.

Examples

```
data(lathyrus)

sizevector <- c(0, 4.6, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "Sz1nr", "Sz2nr", "Sz3nr", "Sz4nr", "Sz5nr",
                 "Sz6nr", "Sz7nr", "Sz8nr", "Sz9nr", "Sz1r", "Sz2r", "Sz3r", "Sz4r",
                 "Sz5r", "Sz6r", "Sz7r", "Sz8r", "Sz9r")
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
minima <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
maxima <- c(NA, 1, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA)
binvec <- c(0, 4.6, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
            0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5)

lathframeln <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                    indataset = indataset, binhalfwidth = binvec, propstatus = propvector,
                        minage = minima, maxage = maxima)

lathvertln <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                           individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                           sizeacol = "lnVol88", repstracol = "Intactseed88",
                           fecacol = "Intactseed88", deadacol = "Dead1988",
                           nonobsacol = "Dormant1988", stageassign = lathframeln,
                           stagesize = "sizea", censorcol = "Missing1988",
                           censorkeep = NA, NAas0 = TRUE, censor = TRUE)

lathvertln$feca2 <- round(lathvertln$feca2)
lathvertln$feca1 <- round(lathvertln$feca1)
lathvertln$feca3 <- round(lathvertln$feca3)

lathrepmln <- matrix(0, 21, 21)
lathrepmln[1, c(13:21)] <- 0.345
lathrepmln[2, c(13:21)] <- 0.054

lathover2 <- overwrite(stage3 = c("Sd", "Sdl"), stage2 = c("Sd", "Sd"),
                       givenrate = c(0.345, 0.054))

lathmodelsln2 <- modelsearch(lathvertln, historical = FALSE, approach = "lme4", suite = "main",
                             vitalrates = c("surv", "obs", "size", "repst", "fec"),
                           juvestimate = "Sdl",bestfit = "AICc&k", sizedist = "gaussian",
                             fecdist = "poisson", indiv = "individ", patch = "patchid",
                             year = "year2", age = "obsage", year.as.random = TRUE,
                          patch.as.random = TRUE, show.model.tables = TRUE, quiet = TRUE)

lathmat2age <- aflefko2(year = "all", patch = "all", stageframe = lathframeln,
                        modelsuite = lathmodelsln2, data = lathvertln,
                        repmatrix = lathrepmln, overwrite = lathover2,
                        patchcol = "patchid", yearcol = "year2", year.as.random = FALSE,
                patch.as.random = FALSE, final_age = 2, continue = TRUE, reduce = FALSE)

summary(lathmat2age)
```

cypdata — *Demographic Dataset of* Cypripedium candidum *Population, in Horizontal Format*

Description

A dataset containing the states and fates of *Cypripedium candidum* (white lady's slipper orchids), family Orchidaceae, from a population in Illinois, USA, resulting from monitoring that occurred annually between 2004 and 2009.

Usage

```
data(cypdata)
```

Format

A data frame with 77 individuals and 27 variables. Each row corresponds to an unique individual, and each variable from `size.04` on refers to the state of the individual in a particular year.

plantid A numberic variable giving a unique number to each individual.

patch A variable refering to patch within the population.

censor A variable coding for whether the data point is valid. An entry of 1 means that it is so.

Inf2.04 Number of double inflorescences in 2004.

Inf.04 Number of inflorescences in 2004.

Veg.04 Number of stems without inflorescences in 2004.

Pod.04 Number of fruits in 2004.

Inf2.05 Number of double inflorescences in 2005.

Inf.05 Number of inflorescences in 2005.

Veg.05 Number of stems without inflorescences in 2005.

Pod.05 Number of fruits in 2005.

Inf2.06 Number of double inflorescences in 2006.

Inf.06 Number of inflorescences in 2006.

Veg.06 Number of stems without inflorescences in 2006.

Pod.06 Number of fruits in 2006.

Inf2.07 Number of double inflorescences in 2007.

Inf.07 Number of inflorescences in 2007.

Veg.07 Number of stems without inflorescences in 2007.

Pod.07 Number of fruits in 2007.

Inf2.08 Number of double inflorescences in 2008.

Inf.08 Number of inflorescences in 2008.

Veg.08 Number of stems without inflorescences in 2008.

Pod.08 Number of fruits in 2008.

Inf2.09 Number of double inflorescences in 2009.

Inf.09 Number of inflorescences in 2009.

Veg.09 Number of stems without inflorescences in 2009.

Pod.09 Number of fruits in 2009.

Source

Shefferson, R.P., R. Mizuta, and M.J. Hutchings. 2017. Predicting evolution in response to climate change: the example of sprouting probability in three dormancy-prone orchid species. *Royal Society Open Science* 4(1):160647.

Examples

```
data(cypdata)

sizevector <- c(0, 0, 0, 0, 0, 0, 1, 2.5, 4.5, 8, 17.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm",
                 "Sm", "Md", "Lg", "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1, 1, 2.5, 7)

cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
                          repstatus = repvector, obsstatus = obsvector,
                          matstatus = matvector, propstatus = propvector,
                          immstatus = immvector, indataset = indataset,
                          binhalfwidth = binvec)

cypraw_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
                          patchidcol = "patch", individcol = "plantid",
                          blocksize = 4, sizeacol = "Inf2.04", sizebcol = "Inf.04",
                          sizeccol = "Veg.04", repstracol = "Inf.04",
                          repstrbcol = "Inf2.04", fecacol = "Pod.04",
                          stageassign = cypframe_raw, stagesize = "sizeadded",
                          NAas0 = TRUE, NRasRep = TRUE)

rep_cyp_raw <- matrix(0, 11, 11)
```

```
rep_cyp_raw[1:2,7:11] <- 0.5

cypover2r <- overwrite(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
                       "XSm", "Sm"), stage2 = c("SD", "SD", "P1", "P2", "P3",
                       "SL", "SL", "SL", "SL"), eststage3 = c(NA, NA, NA, NA,
                       NA, NA, "D", "XSm", "Sm"), eststage2 = c(NA, NA, NA, NA,
                       NA, NA, "XSm", "XSm", "XSm"), givenrate = c(0.1, 0.2,
                       0.2, 0.2, 0.25, 0.4, NA, NA, NA), type = c("S", "S", "S",
                       "S", "S", "S", "S", "S", "S"))

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw, year = "all",
                       patch = "all", stages = c("stage3", "stage2"),
                       size = c("size3added", "size2added"),
                       repmatrix = rep_cyp_raw, overwrite = cypover2r,
                       yearcol = "year2", patchcol = "patchid",
                       indivcol = "individ")
cypmatrix2r$A[[1]]
```

cypvert *Demographic Dataset of* Cypripedium candidum *Population, in Vertical Format*

Description

A dataset containing the states and fates of *Cypripedium candidum* (white lady's slipper orchids), family Orchidaceae, from a population in Illinois, USA, resulting from monitoring that occurred annually between 2004 and 2009. Same dataset as `cypdata`, but arranged in an ahistorical vertical format.

Usage

```
data(cypvert)
```

Format

A data frame with 77 individuals, 331 rows, and 12 variables. Each row corresponds to a specific two-year transition for a specific individual. Variable codes are similar to those for `cypdata`, but use `.2` to identify time *t* and `.3` to identify time *t*+1.

plantid A numberic variable giving a unique number to each individual.

patch A variable refering to patch within the population.

censor A variable coding for whether the data point is valid. An entry of 1 means that it is so.

year2 Year in time *t*.

Inf2.2 Number of double inflorescences in time *t*.

Inf.2 Number of inflorescences in time *t*.

Veg.2 Number of stems without inflorescences in time *t*.

Pod.2 Number of fruits in time *t*.

Inf2.3 Number of double inflorescences in time *t*+1.

Inf.3 Number of inflorescences in time *t*+1.

Veg.3 Number of stems without inflorescences in time *t*+1.

Pod.3 Number of fruits in time *t*+1.

Source

Shefferson, R.P., R. Mizuta, and M.J. Hutchings. 2017. Predicting evolution in response to climate change: the example of sprouting probability in three dormancy-prone orchid species. *Royal Society Open Science* 4(1):160647.

Examples

```
data(cypvert)

sizevector <- c(0, 0, 0, 0, 0, 0, 1, 2.5, 4.5, 8, 17.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg", "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1, 1, 2.5, 7)

cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
                          repstatus = repvector, obsstatus = obsvector,
                          matstatus = matvector, propstatus = propvector,
                          immstatus = immvector, indataset = indataset,
                          binhalfwidth = binvec)

cypframe_raw

cypraw_v2 <- historicalize3(data = cypvert, patchidcol = "patch", individcol = "plantid",
                          year2col = "year2", sizea2col = "Inf2.2", sizea3col = "Inf2.3",
                           sizeb2col = "Inf.2", sizeb3col = "Inf.3", sizec2col = "Veg.2",
                      sizec3col = "Veg.3", repstra2col = "Inf2.2", repstra3col = "Inf2.3",
                        repstrb2col = "Inf.2", repstrb3col = "Inf.3", feca2col = "Pod.2",
                           feca3col = "Pod.3", repstrrel = 2, stageassign = cypframe_raw,
                           stagesize = "sizeadded", censorcol = "censor", censor = FALSE,
                             NAas0 = TRUE, NRasRep = TRUE, reduce = TRUE)
summary(cypraw_v2)
```

elasticity3 *Calculate Elasticity of Lambda to Matrix Elements*

Description

elasticity3() is a generic function that returns the elasticity of the deterministic population growth rate, lambda, to the elements of the matrix population model. This function is made to handle very large and sparse matrices supplied as lefkoMat objects or as individual matrices.

Usage

```
elasticity3(mats)
```

Arguments

mats	A lefkoMat object, or population projection matrix, for which the stable stage distribution is desired.

Value

The value returned depends on the class of the mats argument.

See Also

elasticity3.lefkoMat()

elasticity3.matrix()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                        nonobsacol = "Dormant1988", stageassign = lathframe,
                        stagesize = "sizea", censorcol = "Missing1988",
                        censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                      stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                  stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                  overwrite = lathover3, yearcol = "year2",
                  indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
elasticity3(ehrlen3mean)
```

elasticity3.lefkoMat *Calculate Elasticity of Lambda to Matrix Elements for a lefkoMat Object*

Description

elasticity3.lefkoMat() returns the elasticities of lambda to elements of all $A matrices in an object of class lefkoMat. This function can handle large and sparse matrices, and so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'lefkoMat'
elasticity3(mats)
```

Arguments

mats	An object of class lefkoMat.

Value

This function's output depends on whether the lefkoMat object is historical or ahistorical. If the former, then it is a list with four elements. The first (h_elasmats) is a list of historical elasticity matrices, the second (ah_elasmats) is a list of sensitivity matrices in which historical elasticities have been summed by the stage in times *t* and *t*+1, to produce elasticity matrices equivalent in principle to ahistorical elasticity matrices but reflecting the effects of stage in time *t*-1. The third element (h_stages) is a data frame showing historical stage pairs, and the fourth (ah_stages) is a data frame showing the ahistorical stages.

See Also

elasticity3()

elasticity3.matrix()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
elasticity3(ehrlen3mean)
```

elasticity3.matrix *Calculate Elasticity of Lambda to Matrix Elements for a Matrix*

Description

`elasticity3.matrix()` returns the elasticities of lambda to elements of a single matrix. This function can handle large and sparse matrices, and so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'matrix'
elasticity3(mats)
```

Arguments

`mats`	An object of class `matrix`.

Value

This function returns a single elasticity matrix.

See Also

`elasticity3()`

`elasticity3.lefkoMat()`

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
```

```
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
elasticity3(ehrlen3mean$A[[1]])
```

flefko2 *Create Function-based Ahistorical Matrix Projection Model*

Description

Function flefko2() returns ahistorical MPMs corresponding to the patches and years given, including the associated component transition and fecundity matrices, a data frame detailing the characteristics of the ahistorical stages used, and a data frame characterizing the patch and year combinations corresponding to these matrices. Unlike rlefko2() and rlefko3(), this function currently does not currently distinguish populations.

Usage

```
flefko2(
  year = "all",
  patch = "all",
  stageframe,
  repmatrix = NA,
  overwrite = NA,
  data = NA,
  modelsuite = NA,
  surv_model = NA,
  obs_model = NA,
  size_model = NA,
  repst_model = NA,
  fec_model = NA,
  jsurv_model = NA,
  jobs_model = NA,
  jsize_model = NA,
  jrepst_model = NA,
  paramnames = NA,
  inda = 0,
  indb = 0,
  indc = 0,
  surv_dev = 0,
  obs_dev = 0,
  size_dev = 0,
  repst_dev = 0,
  fec_dev = 0,
  jsurv_dev = 0,
  jobs_dev = 0,
  jsize_dev = 0,
  jrepst_dev = 0,
  repmod = 1,
  yearcol = NA,
  patchcol = NA,
  year.as.random = FALSE,
  patch.as.random = FALSE,
  randomseed = 0,
  negfec = FALSE,
  reduce = FALSE
)
```

Arguments

`year` A variable corresponding to year or observation time, or a set of such values, given in values associated with the year term used in linear model development. Can also equal `all`, in which case matrices will be estimated for all years. Defaults to `all`.

`patch` A variable designating which patches or subpopulations will have matrices estimated. Should be set to specific patch names, or to `all` if matrices should be estimated for all patches. Defaults to `all`.

`stageframe` A stageframe object that includes information on the size, observation status, propagule status, immaturity status, and maturity status of each ahistorical stage. Should also incorporate bin widths if size is continuous.

`repmatrix` A matrix composed mostly of 0s, with non-zero values for each potentially new individual (row) born to each reproductive stage (column). Entries act as multipliers on fecundity, with 1 equaling full fecundity.

`overwrite` A data frame developed with the `overwrite()` function describing transitions to be overwritten either with given values or with other estimated transitions.

`data` The original historical demographic data frame used to estimate vital rates (class `hfvdata`). The original data frame is required in order to initialize years and patches properly.

`modelsuite` An optional `lefkoMod` object holding the vital rate models. If given, then `surv_model`, `obs_model`, `size_model`, `repst_model`, `fec_model`, `jsurv_model`, `jobs_model`, `jsize_model`, `jrepst_model`, `paramnames`, `yearcol`, and `patchcol` are not required. No models should include size or reproductive status in time *t*-1.

`surv_model` A linear model predicting survival probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any survival probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

obs_model
: A linear model predicting sprouting or observation probability. This can be a model of class glm or glmer, and requires a predicted binomial variable under a logit link. If given, then will overwrite any observation probability model given in modelsuite. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

size_model
: A linear model predicting size. This can be a model of class glm or glmer, both of which require a predicted poisson variable under a log link, or a model of class lm or lmer, in which a Gaussian response is assumed. If given, then will overwrite any size model given in modelsuite. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

repst_model
: A linear model predicting reproduction probability. This can be a model of class glm or glmer, and requires a predicted binomial variable under a logit link. If given, then will overwrite any reproduction probability model given in modelsuite. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

fec_model
: A linear model predicting fecundity. This can be a model of class glm or glmer, and requires a predicted poisson variable under a log link. If given, then will overwrite any fecundity model given in modelsuite. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

jsurv_model
: A linear model predicting juvenile survival probability. This can be a model of class glm or glmer, and requires a predicted binomial variable under a logit link. If given, then will overwrite any juvenile survival probability model given in modelsuite. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

jobs_model
: A linear model predicting juvenile sprouting or observation probability. This can be a model of class glm or glmer, and requires a predicted binomial variable under a logit link. If given, then will overwrite any juvenile observation probability model given in modelsuite. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

jsize_model
: A linear model predicting juvenile size. This can be a model of class glm or glmer, both of which require a predicted poisson variable under a log link, or a model of class lm or lmer, in which a Gaussian response is assumed. If given, then will overwrite any juvenile size model given in modelsuite. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

jrepst_model
: A linear model predicting reproduction probability of a mature individual that was immature in the previous year. This can be a model of class glm or glmer, and requires a predicted binomial variable under a logit link. If given, then will overwrite any reproduction probability model given in modelsuite. This model must have been developed in a modeling exercise testing only the impacts of time *t*.

paramnames
: A dataframe with two columns, the first showing the general model terms that will be used in matrix creation, and the second showing the equivalent terms used in modeling. Only required if modelsuite is not supplied.

`inda` A numeric value to use for individual covariate a. Defaults to 0.

`indb` A numeric value to use for individual covariate b. Defaults to 0.

`indc` A numeric value to use for individual covariate c. Defaults to 0.

`surv_dev` A numeric value to be added to the y-intercept in the linear model for survival probability.

`obs_dev` A numeric value to be added to the y-intercept in the linear model for observation probability.

`size_dev` A numeric value to be added to the y-intercept in the linear model for size.

`repst_dev` A numeric value to be added to the y-intercept in the linear model for probability of reproduction.

`fec_dev` A numeric value to be added to the y-intercept in the linear model for fecundity.

`jsurv_dev` A numeric value to be added to the y-intercept in the linear model for juvenile survival probability.

`jobs_dev` A numeric value to be added to the y-intercept in the linear model for juvenile observation probability.

`jsize_dev` A numeric value to be added to the y-intercept in the linear model for juvenile size.

`jrepst_dev` A numeric value to be added to the y-intercept in the linear model for juvenile reproduction probability.

`repmod` A scalar multiplier of fecundity. Defaults to 1.

`yearcol` The variable name or column number corresponding to year in time *t* in the dataset. Not needed if a `modelsuite` is supplied.

`patchcol` The variable name or column number corresponding to patch in the dataset. Not needed if a `modelsuite` is supplied.

`year.as.random` A logical term indicating whether coefficients for missing patches within vital rate models should be estimated as random intercepts. Defaults to FALSE, in which case missing time step coefficients are set to 0.

`patch.as.random`

A logical term indicating whether coefficients for missing patches within vital rate models should be estimated as random intercepts. Defaults to FALSE, in which case missing patch coefficients are set to 0.

`randomseed` A numeric value used as a seed to generate random estimates for missing time step and patch coefficients, if either `year.as.random` or `patch.as.random` is set to TRUE. Defaults to 0.

`negfec` A logical value denoting whether fecundity values estimated to be negative should be reset to 0. Defaults to FALSE.

`reduce` A logical value denoting whether to remove ahistorical stages associated solely with 0 transitions. These are only removed in cases where the associated row and column sums in ALL matrices estimated equal 0. Defaults to FALSE.

Value

If all inputs are properly formatted, then this function will return either an object of class `lefkoMat`. Output includes:

`A` A list of full projection matrices in order of sorted patches and years. All matrices output in the `matrix` class.

`U` A list of survival-transition matrices sorted as in `A`. All matrices output in the `matrix` class.

`F` A list of fecundity matrices sorted as in `A`. All matrices output in the `matrix` class.

`hstages` Null for ahistorical matrices.

`ahstages` A data frame detailing the characteristics of associated ahistorical stages.

`labels` A data frame giving the patch and year of each matrix in order. In `flefko2()`, only one population may be analyzed at once, and so `pop = NA`

`matrixqc` A short vector describing the number of non-zero elements in `U` and `F` matrices, and the number of annual matrices.

`modelqc` This is the qc portion of the modelsuite input.

Please note that this function will yield incorrect estimates if the models utilized incorporate state in time *t*-1. Only use models developed testing ahistorical effects.

Examples

```
data(lathyrus)

sizevector <- c(0, 4.6, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "Sz1nr", "Sz2nr", "Sz3nr", "Sz4nr", "Sz5nr",
                 "Sz6nr", "Sz7nr", "Sz8nr", "Sz9nr", "Sz1r", "Sz2r", "Sz3r", "Sz4r",
                 "Sz5r", "Sz6r", "Sz7r", "Sz8r", "Sz9r")
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 4.6, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
            0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5)


lathframeln <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                       obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                     indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvertln <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                             individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                             sizeacol = "lnVol88", repstracol = "Intactseed88",
                             fecacol = "Intactseed88", deadacol = "Dead1988",
                             nonobsacol = "Dormant1988", stageassign = lathframeln,
                           stagesize = "sizea", censorcol = "Missing1988",
                           censorkeep = NA, NAas0 = TRUE, censor = TRUE)

lathvertln$feca2 <- round(lathvertln$feca2)
lathvertln$feca1 <- round(lathvertln$feca1)
lathvertln$feca3 <- round(lathvertln$feca3)

lathrepmln <- matrix(0, 21, 21)
lathrepmln[1, c(13:21)] <- 0.345
lathrepmln[2, c(13:21)] <- 0.054

lathover2 <- overwrite(stage3 = c("Sd", "Sdl"), stage2 = c("Sd", "Sd"),
                       givenrate = c(0.345, 0.054))

lathmodelsln2 <- modelsearch(lathvertln, historical = FALSE, approach = "mixed", suite = "main",
                             vitalrates = c("surv", "obs", "size", "repst", "fec"),
                          juvestimate = "Sdl", bestfit = "AICc&k", sizedist = "gaussian",
                             fecdist = "poisson", indiv = "individ", patch = "patchid",
                           year = "year2", year.as.random = TRUE, patch.as.random = TRUE,
                             show.model.tables = TRUE, quiet = TRUE)

lathmat2ln <- flefko2(year = "all", patch = "all", stageframe = lathframeln,
                      modelsuite = lathmodelsln2, data = lathvertln,
                      repmatrix = lathrepmln, overwrite = lathover2,
                      patchcol = "patchid", yearcol = "year2",
                      year.as.random = FALSE, patch.as.random = FALSE,
                      reduce = FALSE)

summary(lathmat2ln)
```

`flefko3` *Create Function-based Historical Matrix Projection Model*

Description

Function `flefko3()` returns function-based historical MPMs corresponding to the patches and years given, including the associated component transition and fecundity matrices, data frames detailing the characteristics of the ahistorical stages used and historical stage pairs created, and a data frame characterizing the patch and year combinations corresponding to these matrices. Unlike `rlefko3()`, this function currently does not currently distinguish populations.

Usage

```
flefko3(
  year = "all",
  patch = "all",
  stageframe,
  repmatrix = NA,
  overwrite = NA,
  data = NA,
  modelsuite = NA,
  surv_model = NA,
  obs_model = NA,
  size_model = NA,
  repst_model = NA,
  fec_model = NA,
  jsurv_model = NA,
  jobs_model = NA,
  jsize_model = NA,
  jrepst_model = NA,
  paramnames = NA,
  inda = 0,
  indb = 0,
  indc = 0,
  surv_dev = 0,
  obs_dev = 0,
  size_dev = 0,
  repst_dev = 0,
  fec_dev = 0,
  jsurv_dev = 0,
  jobs_dev = 0,
  jsize_dev = 0,
  jrepst_dev = 0,
  repmod = 1,
  yearcol = NA,
  patchcol = NA,
  year.as.random = FALSE,
  patch.as.random = FALSE,
  randomseed = 0,
  negfec = FALSE,
  reduce = FALSE
)
```

Arguments

`year` A variable corresponding to year or observation time, or a set of such values, given in values associated with the year term used in linear model development. Can also equal `all`, in which case matrices will be estimated for all years. Defaults to `all`.

`patch` A variable designating which patches or subpopulations will have matrices estimated. Should be set to specific patch names, or to `all` if matrices should be estimated for all patches. Defaults to `all`.

`stageframe` A stageframe object that includes information on the size, observation status, propagule status, immaturity status, and maturity status of each ahistorical stage. Should also incorporate bin widths if size is continuous.

`repmatrix` A matrix composed mostly of 0s, with non-zero values for each potentially new individual (row) born to each reproductive stage (column). Entries act as multipliers on fecundity, with 1 equaling full fecundity.

`overwrite` A data frame developed with the `overwrite()` function describing transitions to be overwritten either with given values or with other estimated transitions.

`data` The original historical demographic data frame used to estimate vital rates (class `hfvdata`). The original data frame is required in order to initialize years and patches properly.

`modelsuite` An optional `lefkoMod` object holding the vital rate models. If given, then `surv_model`, `obs_model`, `size_model`, `repst_model`, `fec_model`, `jsurv_model`, `jobs_model`, `jsize_model`, `jrepst_model`, `paramnames`, `yearcol`, and `patchcol` are not required. One or more of these models should include size or reproductive status in time *t*-1.

`surv_model` A linear model predicting survival probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any survival probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing the impacts of times *t* and *t*-1.

`obs_model` A linear model predicting sprouting or observation probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any observation probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing the impacts of times *t* and *t*-1.

`size_model` A linear model predicting size. This can be a model of class `glm` or `glmer`, both of which require a predicted poisson variable under a log link, or a model of class `lm` or `lmer`, in which a Gaussian response is assumed. If given, then will overwrite any size model given in `modelsuite`. This model must have been developed in a modeling exercise testing the impacts of times *t* and *t*-1.

`repst_model` A linear model predicting reproduction probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any reproduction probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing the impacts of times *t* and *t*-1.

`fec_model` A linear model predicting fecundity. This can be a model of class `glm` or `glmer`, and requires a predicted poisson variable under a log link. If given, then will overwrite any fecundity model given in `modelsuite`. This model must have been developed in a modeling exercise testing the impacts of times *t* and *t*-1.

`jsurv_model` A linear model predicting juvenile survival probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any juvenile survival probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing the impacts of times *t* and *t*-1.

`jobs_model` A linear model predicting juvenile sprouting or observation probability. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any juvenile observation probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing the impacts of times *t* and *t*-1.

`jsize_model` A linear model predicting juvenile size. This can be a model of class `glm` or `glmer`, both of which require a predicted poisson variable under a log link, or a model of class `lm` or `lmer`, in which a Gaussian response is assumed. If given, then will overwrite any juvenile size model given in `modelsuite`. This model must have been developed in a modeling exercise testing the impacts of times *t* and *t*-1.

`jrepst_model` A linear model predicting reproduction probability of a mature individual that was immature in the previous year. This can be a model of class `glm` or `glmer`, and requires a predicted binomial variable under a logit link. If given, then will overwrite any reproduction probability model given in `modelsuite`. This model must have been developed in a modeling exercise testing the impacts of times *t* and *t*-1.

`paramnames` A dataframe with two columns, the first showing the general model terms that will be used in matrix creation, and the second showing the equivalent terms used in modeling. Only required if `modelsuite` is not supplied.

`inda` A numeric value to use for individual covariate a. Defaults to 0.

`indb` A numeric value to use for individual covariate b. Defaults to 0.

`indc` A numeric value to use for individual covariate c. Defaults to 0.

`surv_dev` A numeric value to be added to the y-intercept in the linear model for survival probability.

`obs_dev` A numeric value to be added to the y-intercept in the linear model for observation probability.

`size_dev` A numeric value to be added to the y-intercept in the linear model for size.

`repst_dev` A numeric value to be added to the y-intercept in the linear model for probability of reproduction.

`fec_dev` A numeric value to be added to the y-intercept in the linear model for fecundity.

`jsurv_dev` A numeric value to be added to the y-intercept in the linear model for juvenile survival probability.

`jobs_dev` A numeric value to be added to the y-intercept in the linear model for juvenile observation probability.

`jsize_dev` A numeric value to be added to the y-intercept in the linear model for juvenile size.

`jrepst_dev` A numeric value to be added to the y-intercept in the linear model for juvenile reproduction probability.

`repmod` A scalar multiplier of fecundity. Defaults to 1.

`yearcol` The variable name or column number corresponding to year in time *t* in the dataset. Not needed if `modelsuite` is supplied.

`patchcol` The variable name or column number corresponding to patch in the dataset. Not needed if `modelsuite` is supplied.

`year.as.random` A logical term indicating whether coefficients for missing patches within vital rate models should be estimated as random intercepts. Defaults to FALSE, in which case missing time step coefficients are set to 0.

`patch.as.random`
: A logical term indicating whether coefficients for missing patches within vital rate models should be estimated as random intercepts. Defaults to FALSE, in which case missing patch coefficients are set to 0.

`randomseed`
: A numeric value used as a seed to generate random estimates for missing time step and patch coefficients, if either `year.as.random` or `patch.as.random` is set to TRUE. Defaults to 0.

`negfec`
: A logical value denoting whether fecundity values estimated to be negative should be reset to 0. Defaults to FALSE.

`reduce`
: A logical value denoting whether to remove historical stages associated solely with 0 transitions. These are only removed in cases where the associated row and column sums in ALL matrices estimated equal 0. Defaults to FALSE.

Value

If all inputs are properly formatted, then this function will return an object of class `lefkoMat`, which is a list with the following elements:

`A`
: A list of full projection matrices in order of sorted patches and years. All matrices output in the `matrix` class.

`U`
: A list of survival-transition matrices sorted as in `A`. All matrices output in the `matrix` class.

`F`
: A list of fecundity matrices sorted as in `A`. All matrices output in the `matrix` class.

`hstages`
: A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs.

`ahstages`
: A data frame detailing the characteristics of associated ahistorical stages.

`labels`
: A data frame showing the patch and year of each matrix in order. In `flefko3()`, only one population may be analyzed at once, and so `pop = NA`.

`matrixqc`
: A short vector describing the number of non-zero elements in `U` and `F` matrices, and the number of annual matrices.

`modelqc`
: This is the `qc` portion of the `modelsuite` input.

Examples

```
data(lathyrus)

sizevector <- c(0, 4.6, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "Sz1nr", "Sz2nr", "Sz3nr", "Sz4nr", "Sz5nr",
                 "Sz6nr", "Sz7nr", "Sz8nr", "Sz9nr", "Sz1r", "Sz2r", "Sz3r", "Sz4r",
                 "Sz5r", "Sz6r", "Sz7r", "Sz8r", "Sz9r")
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
```

```
indataset <- c(0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 4.6, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
            0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5)

lathframeln <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvertln <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                           individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                           sizeacol = "lnVol88", repstracol = "Intactseed88",
                           fecacol = "Intactseed88", deadacol = "Dead1988",
                           nonobsacol = "Dormant1988", stageassign = lathframeln,
                           stagesize = "sizea", censorcol = "Missing1988",
                           censorkeep = NA, NAas0 = TRUE, censor = TRUE)

lathvertln$feca2 <- round(lathvertln$feca2)
lathvertln$feca1 <- round(lathvertln$feca1)
lathvertln$feca3 <- round(lathvertln$feca3)

lathrepmln <- matrix(0, 21, 21)
lathrepmln[1, c(13:21)] <- 0.345
lathrepmln[2, c(13:21)] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

lathmodelsln3 <- modelsearch(lathvertln, historical = TRUE, approach = "mixed", suite = "main",
                             vitalrates = c("surv", "obs", "size", "repst", "fec"),
                          juvestimate = "Sdl",bestfit = "AICc&k", sizedist = "gaussian",
                             fecdist = "poisson", indiv = "individ", patch = "patchid",
                           year = "year2",year.as.random = TRUE, patch.as.random = TRUE,
                             show.model.tables = TRUE, quiet = TRUE)

lathmat3ln <- flefko3(year = "all", patch = "all", stageframe = lathframeln,
                      modelsuite = lathmodelsln3, data = lathvertln,
                      repmatrix = lathrepmln, overwrite = lathover3,
                      patchcol = "patchid", yearcol = "year2", year.as.random = FALSE,
                      patch.as.random = FALSE, reduce = FALSE)

summary(lathmat3ln)
```

historicalize3 *Create Historical Vertical Data Frame From Ahistorical Vertical Data Frame*

Description

historicalize3() returns a vertically formatted demographic data frame organized to create historical projection matrices, given a vertically but ahistorically formatted data frame. This data frame is in standard lefko3 format and can be used in all functions in the package.

Usage

```
historicalize3(
  data,
  popidcol = 0,
  patchidcol = 0,
  individcol,
  year2col = 0,
  year3col = 0,
  xcol = 0,
  ycol = 0,
  sizea2col = 0,
  sizea3col = 0,
  sizeb2col = 0,
  sizeb3col = 0,
  sizec2col = 0,
  sizec3col = 0,
  repstra2col = 0,
  repstra3col = 0,
  repstrb2col = 0,
  repstrb3col = 0,
  feca2col = 0,
  feca3col = 0,
  fecb2col = 0,
  fecb3col = 0,
  indcova2col = 0,
  indcova3col = 0,
  indcovb2col = 0,
  indcovb3col = 0,
  indcovc2col = 0,
  indcovc3col = 0,
  alive2col = 0,
  alive3col = 0,
  dead2col = 0,
  dead3col = 0,
  obs2col = 0,
  obs3col = 0,
  nonobs2col = 0,
  nonobs3col = 0,
  repstrrel = 1,
  fecrel = 1,
  stage2col = 0,
  stage3col = 0,
  juv2col = 0,
  juv3col = 0,
  stageassign = NA,
  stagesize = NA,
  censor = FALSE,
  censorcol = 0,
  censorkeep = 0,
  spacing = NA,
  NAas0 = FALSE,
  NRasRep = FALSE,
  reduce = TRUE
)
```

Arguments

`data` The horizontal data file.

`popidcol` A variable name or column number corresponding to the identity of the population for each individual.

`patchidcol` A variable name or column number corresponding to the identity of the patch for each individual, if patches have been designated within populations.

`individcol` A variable name or column number corresponding to the unique identity of each individual.

`year2col` A variable name or column number corresponding to the year or time in time *t*.

`year3col` A variable name or column number corresponding to the year or time in time *t*+1.

`xcol` A variable name or column number corresponding to the x coordinate of each individual in Cartesian space.

`ycol` A variable name or column number corresponding to the y coordinate of each individual in Cartesian space.

`sizea2col` A variable name or column number corresponding to the primary size entry in time *t*.

`sizea3col` A variable name or column number corresponding to the primary size entry in time *t*+1.

`sizeb2col` A variable name or column number corresponding to the secondary size entry in time *t*.

`sizeb3col` A variable name or column number corresponding to the secondary size entry in time *t*+1.

`sizec2col` A variable name or column number corresponding to the tertiary size entry in time *t*.

`sizec3col` A variable name or column number corresponding to the tertiary size entry in time *t*+1.

`repstra2col` A variable name or column number corresponding to the production of reproductive structures, such as flowers, in time *t*. This can be binomial or count data, and is used to in analysis of the probability of reproduction.

`repstra3col` A variable name or column number corresponding to the production of reproductive structures, such as flowers, in time *t*+1. This can be binomial or count data, and is used to in analysis of the probability of reproduction.

`repstrb2col` A second variable name or column number corresponding to the production of reproductive structures, such as flowers, in time *t*. This can be binomial or count data.

`repstrb3col` A second variable name or column number corresponding to the production of reproductive structures, such as flowers, in time *t*+1. This can be binomial or count data.

`feca2col` A variable name or column number corresponding to fecundity in time *t*. This may represent egg counts, fruit counts, seed production, etc.

`feca3col` A variable name or column number corresponding to fecundity in time *t*+1. This may represent egg counts, fruit counts, seed production, etc.

`fecb2col` A second variable name or column number corresponding to fecundity in time *t*. This may represent egg counts, fruit counts, seed production, etc.

`fecb3col` A second variable name or column number corresponding to fecundity in time *t*+1. This may represent egg counts, fruit counts, seed production, etc.

`indcova2col` A variable name or column number corresponding to an individual covariate to be used in analysis, in time *t*.

`indcova3col` A variable name or column number corresponding to an individual covariate to be used in analysis, in time *t*+1.

`indcovb2col` A second variable name or column number corresponding to an individual covariate to be used in analysis, in time *t*.

`indcovb3col` A second variable name or column number corresponding to an individual covariate to be used in analysis, in time *t*+1.

`indcovc2col` A third variable name or column number corresponding to an individual covariate to be used in analysis, in time *t*.

`indcovc3col` A third variable name or column number corresponding to an individual covariate to be used in analysis, in time *t*+1.

`alive2col` A variable name or column number that provides information on whether an individual is alive in time *t*. If used, living status must be designated as binomial (living = 1, dead = 0).

`alive3col` A variable name or column number that provides information on whether an individual is alive in time *t*+1. If used, living status must be designated as binomial (living = 1, dead = 0).

`dead2col` A variable name or column number that provides information on whether an individual is dead in time *t*. If used, dead status must be designated as binomial (dead = 1, living = 0).

`dead3col` A variable name or column number that provides information on whether an individual is dead in time *t*+1. If used, dead status must be designated as binomial (dead = 1, living = 0).

`obs2col` A variable name or column number providing information on whether an individual is in an observable stage in time *t*. If used, observation status must be designated as binomial (observed = 1, not observed = 0).

`obs3col` A variable name or column number providing information on whether an individual is in an observable stage in time *t*+1. If used, observation status must be designated as binomial (observed = 1, not observed = 0).

`nonobs2col` A variable name or column number providing information on whether an individual is in an unobservable stage in time *t*. If used, observation status must be designated as binomial (not observed = 1, observed = 0).

`nonobs3col` A variable name or column number providing information on whether an individual is in an unobservable stage in time *t*+1. If used, observation status must be designated as binomial (not observed = 1, observed = 0).

`repstrrel` This is a scalar multiplier to make the variable represented by `repstrb2col` equivalent to the variable represented by `repstra2col`. This can be useful if two reproductive status variables have related but unequal units, for example if `repstrb2col` refers to one-flowered stems while `repstra2col` refers to two-flowered stems.

`fecrel` This is a scalar multiplier that makes the variable represented by `fecb2col` equivalent to the variable represented by `feca2col`. This can be useful if two fecundity variables have related but unequal units.

`stage2col` Optional variable name or column number corresponding to life history stage in time *t*.

`stage3col` Optional variable name or column number corresponding to life history stage in time *t*+1.

`juv2col` A variable name or column number that marks individuals in immature stages in time *t*. The `historicalize3()` function assumes that immature individuals are identified in this variable marked with a number equal to or greater than 1, and that mature individuals are marked as 0 or NA.

`juv3col` A variable name or column number that marks individuals in immature stages in time *t*+1. The `historicalize3()` function assumes that immature individuals are identified in this variable marked with a number equal to or greater than 1, and that mature individuals are marked as 0 or NA.

`stageassign` The stageframe object identifying the life history model being operationalized. Note that if `stage2col` is provided, then this stageframe is not utilized in stage designation.

`stagesize` A variable name or column number describing which size variable to use in stage estimation. Defaults to NA, and can also take `sizea`, `sizeb`, `sizec`, or `sizeadded`, depending on which size variable is chosen.

`censor` A logical variable determining whether the output data should be censored using the variable defined in `censorcol`. Defaults to FALSE.

`censorcol` A variable name or column number corresponding to a censor variable within the dataset, used to distinguish between entries to use and those to discard from analysis, or to designate entries with special issues that require further attention.

`censorkeep` The value of the censoring variable identifying data that should be included in analysis. Defaults to 0, but may take any value including NA.

`spacing` The spacing at which density should be estimated, if density estimation is desired and x and y coordinates are supplied. Given in the same units as those used in the x and y coordinates given in `xcol` and `ycol`. Defaults to NA.

`NAas0` If TRUE, then all NA entries for size and fecundity variables will be set to 0. This can help increase the sample size analyzed by `modelsearch()`, but should only be used when it is clear that this substitution is biologically realistic. Defaults to FALSE.

`NRasRep` If set to TRUE, then this function will treat non-reproductive but mature individuals as reproductive during stage zssignment. This can be useful when a matrix is desired without separation of reproductive and non-reproductive but mature stages of the same size. Only used if `stageassign` is set to a stageframe. Defaults to FALSE.

`reduce` A logical variable determining whether unused variables and some invariant state variables should be removed from the output dataset. Defaults to TRUE.

Value

If all inputs are properly formatted, then this function will output a historical vertical data frame (class `hfvdata`), meaning that the output data frame will have three consecutive years of size and reproductive data per individual per row. This data frame is in standard format for all functions used in `lefko3`, and so can be used without further modification. Note that determination of state in times *t*-1 and *t*+1 gives preference to condition in time *t* within the input dataset. Conflicts in condition in input datasets that have both times *t* and *t*+1 listed per row are resolved by using condition in time *t*.

`rowid` Unique identifier for the row of the data frame.

`popid` Unique identifier for the population, if given.

`patchid` Unique identifier for patch within population, if given.

`individ` Unique identifier for the individual.

`year2` Year or time at time *t*.

`firstseen` Year or time of first observation.

`lastseen` Year or time of last observation.

`obsage` Observed age in time *t*, assuming first observation corresponds to age = 0.

`obslifespan` Observed lifespan, given as `lastseen -firstseen + 1`.

`xpos1,xpos2,xpos3`
X position in Cartesian space in times *t*-1, *t*, and *t*+1, respectively, if provided.

`ypos1,ypos2,ypos3`
Y position in Cartesian space in times *t*-1, *t*, and *t*+1, respectively, if provided.

`sizea1,sizea2,sizea3`
Main size measurement in times *t*-1, *t*, and *t*+1, respectively.

`sizeb1,sizeb2,sizeb3`
Secondary size measurement in times *t*-1, *t*, and *t*+1, respectively.

`sizec1,sizec2,sizec3`
Tertiary measurement in times *t*-1, *t*, and *t*+1, respectively.

`size1added,size2added,size3added`
Sum of primary, secondary, and tertiary size measurements in timea *t*-1, *t*, and *t*+1, respectively.

`repstra1,repstra2,repstra3`
: Main numbers of reproductive structures in times t-1, t, and t+1, respectively.

`repstrb1,repstrb2,repstrb3`
: Secondary numbers of reproductive structures in times t-1, t, and t+1, respectively.

`repstr1added,repstr2added,repstr3added`
: Sum of primary and secondary reproductive structures in times t-1, t, and t+1, respectively.

`feca1,feca2,feca3`
: Main numbers of offspring in times t-1, t, and t+1, respectively.

`fecb1,fecb2, fecb3`
: Secondary numbers of offspring in times t-1, t, and t+1, respectively.

`fec1added,fec2added,fec3added`
: Sum of primary and secondary fecundity in times t-1, t, and t+1, respectively.

`censor1,censor2,censor3`
: Censor state values in times t-1, t, and t+1, respectively.

`juvgiven1,juvgiven2,juvgiven3`
: Binomial variable indicating whether individual is juvenile in times t-1, t, and t+1. Only given if `juvcol` is provided.

`obsstatus1,obsstatus2,obsstatus3`
: Binomial observation state in times t-1, t, and t+1, respectively.

`repstatus1,repstatus2,repstatus3`
: Binomial reproductive state in times t-1, t, and t+1, respectively.

`fecstatus1,fecstatus2,fecstatus3`
: Binomial offspring production state in times t-1, t, and t+1, respectively.

`matstatus1,matstatus2,matstatus3`
: Binomial maturity state in times t-1, t, and t+1, respectively.

`alive1,alive2,alive3`
: Binomial state as alive in times t-1, t, and t+1, respectively.

`density`
: Density of individuals per unit designated in `spacing`. Only given if spacing is not NA.

Examples

```
data(cypvert)

sizevector <- c(0, 0, 0, 0, 0, 0, 1, 2.5, 4.5, 8, 17.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg", "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1, 1, 2.5, 7)

cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
                          repstatus = repvector, obsstatus = obsvector,
                          matstatus = matvector, propstatus = propvector,
                          immstatus = immvector, indataset = indataset,
                          binhalfwidth = binvec)

cypraw_v2 <- historicalize3(data = cypvert, patchidcol = "patch", individcol = "plantid",
                            year2col = "year2", sizea2col = "Inf2.2", sizea3col = "Inf2.3",
                          sizeb2col = "Inf.2", sizeb3col = "Inf.3", sizec2col = "Veg.2",
                      sizec3col = "Veg.3", repstra2col = "Inf2.2", repstra3col = "Inf2.3",
                        repstrb2col = "Inf.2", repstrb3col = "Inf.3", feca2col = "Pod.2",
                          feca3col = "Pod.3", repstrrel = 2, stageassign = cypframe_raw,
                          stagesize = "sizeadded", censorcol = "censor", censor = FALSE,
                            NAas0 = TRUE, NRasRep = TRUE)
summary(cypraw_v2)
```

lambda3 *Dominant Eigenvalue and Deterministic Population Growth Rate Estimation*

Description

lambda3() is a generic function that returns the dominant eigenvalue of a matrix, and set of dominant eigenvalues of a set of matrices. It can handle very large and sparse matrices supplied as lefkoMat objects or as individual matrices, and can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
lambda3(mats)
```

Arguments

mats	A lefkoMat object, or a single projection matrix, for which the dominant eigenvalue is desired.

Value

The value returned depends on the class of the mats argument. See related functions for more details.

See Also

lambda3.lefkoMat()

lambda3.matrix()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                       obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                      indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
lambda3(ehrlen3mean)
```

---

lambda3.lefkoMat *Estimate Deterministic Population Growth Rates of Matrices in a lefkoMat Object*

Description

lambda3.lefkoMat() returns the dominant eigenvalues of projection matrices supplied within lefkoMat objects. This function can handle large and sparse matrices, and so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'lefkoMat'
lambda3(mats)
```

Arguments

mats	An object of class `lefkoMat`.

Value

This function returns the dominant eigenvalue of each `$A` matrix in the lefkoMat object input. For square matrices with fewer than 400 rows, this is given as the largest real part of all eigenvalues estimated via the `eig_gen()` function in the C++ Armadillo library. For larger matrices, the function assumes that matrices are sparse and uses `eigs_gen()` instead. The output includes a data frame showing the population, patch, and lambda estimate for each `$A` matrix within the object. Row names correspond to the number of the matrix within the `$A` element of the `lefkoMat` object.

See Also

`lambda3()`

`lambda3.matrix()`

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                    indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054
```

```
lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
lambda3(ehrlen3mean)
```

lambda3.matrix *Estimate Deterministic Population Growth Rate of a Projection Matrix*

Description

lambda3.matrix() returns the dominant eigenvalue of a single projection matrix. This function can handle large and sparse matrices, so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'matrix'
lambda3(mats)
```

Arguments

mats	A population projection matrix of class matrix.

Value

This function returns the dominant eigenvalue of the matrix. For square matrices with fewer than 400 rows, this is given as the largest real part of all eigenvalues estimated via the eig_gen() function in package the C++ Armadillo library. For larger matrices, the matrix is assumed to be sparse and eigs_gen() is used instead.

See Also

lambda3()

lambda3.lefkoMat()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                    obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
lambda3(ehrlen3mean$A[[1]])
```

`lathyrus` *Demographic Dataset of* Lathyrus vernus *Population*

Description

A dataset containing the states and fates of *Lathyrus vernus* (spring vetch), family Fabaceae, from a population in Sweden monitored annually from 1988 to 1991 in six study plots.

Usage

```
data(lathyrus)
```

Format

A data frame with 1119 individuals and 34 variables. Each row corresponds to a unique individual, and each variable from `Volume88` on refers to the state of the individual in a given year.

SUBPLOT A variable refering to patch within the population.

GENET A numberic variable giving a unique number to each individual.

Volume88 Aboveground volume in cubic mm in 1988.

lnVol88 Natural logarithm of `Volume88`.

FCODE88 Equals 1 if flowering and 0 if not flowering in 1988.

Flow88 Number of flowers in 1988.

Intactseed88 Number of intact mature seeds produced in 1988. Not always an integer, as in some cases seed number was estimated via linear modeling.

Dead1988 Marked as 1 if known to be dead in 1988.

Dormant1988 Marked as 1 if known to be alive but vegetatively dormant in 1988.

Missing1988 Marked as 1 if not found in 1988.

Seedling1988 Marked as 1, 2, or 3 if observed as a seedling in year *t*. Numbers refer to certainty of assignment: 1 = certain that plant is a seedling in 1988, 2 = likely that plant is a seedling in 1988, 3 = probable that plant is a seedling in 1988.

Volume89 Aboveground volume in cubic mm in 1989.

lnVol89 Natural logarithm of `Volume89`.

FCODE89 Equals 1 if flowering and 0 if not flowering in 1989.

Flow89 Number of flowers in 1989.

Intactseed89 NZumber of intact mature seeds produced in 1989. Not always an integer, as in some cases seed number was estimated via linear modeling.

Dead1989 Marked as 1 if known to be dead in 1989.

Dormant1989 Marked as 1 if known to be alive but vegetatively dormant in 1989.

Missing1989 Marked as 1 if not found in 1989.

Seedling1989 Marked as 1, 2, or 3 if observed as a seedling in year *t*. Numbers refer to certainty of assignment: 1 = certain that plant is a seedling in 1989, 2 = likely that plant is a seedling in 1989, 3 = probable that plant is a seedling in 1989.

Volume90 Aboveground volume in mm<sup>3</sup> in 1990.

lnVol90 Natural logarithm of `Volume90`.

FCODE90 Equals 1 if flowering and 0 if not flowering in 1990.

Flow90 Number of flowers in 1990.

Intactseed90 NZumber of intact mature seeds produced in 1990. Not always an integer, as in some cases seed number was estimated via linear modeling.

Dead1990 Marked as 1 if known to be dead in 1990.

Dormant1990 Marked as 1 if known to be alive but vegetatively dormant in 1990.

Missing1990 Marked as 1 if not found in 1990.

Seedling1990 Marked as 1, 2, or 3 if observed as a seedling in year *t*. Numbers refer to certainty of assignment: 1 = certain that plant is a seedling in 1990, 2 = likely that plant is a seedling in 1990, 3 = probable that plant is a seedling in 1990.

Volume91 Aboveground volume in mm<sup>3</sup> in 1991.

lnVol91 Natural logarithm of `Volume91`.

FCODE91 Equals 1 if flowering and 0 if not flowering in 1991.

Flow91 Number of flowers in 1991.

Intactseed91 NZumber of intact mature seeds produced in 1991. Not always an integer, as in some cases seed number was estimated via linear modeling.

Dead1991 Marked as 1 if known to be dead in 1991.

Dormant1991 Marked as 1 if known to be alive but vegetatively dormant in 1991.

Missing1991 Marked as 1 if not found in 1991.

Seedling1991 Marked as 1, 2, or 3 if observed as a seedling in year *t*. Numbers refer to certainty of assignment: 1 = certain that plant is a seedling in 1991, 2 = likely that plant is a seedling in 1991, 3 = probable that plant is a seedling in 1991.

Source

Ehrlen, J. 2000. The dynamics of plant populations: does the history of individuals matter? *Ecology* 81(6):1675-1684.

Examples

```
data(lathyrus)

sizevector <- c(0, 4.6, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "Sz1nr", "Sz2nr", "Sz3nr", "Sz4nr", "Sz5nr",
                 "Sz6nr", "Sz7nr", "Sz8nr", "Sz9nr", "Sz1r", "Sz2r", "Sz3r", "Sz4r",
                 "Sz5r", "Sz6r", "Sz7r", "Sz8r", "Sz9r")
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 4.6, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
            0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5)

lathframeln <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                    indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvertln <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                           individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                           sizeacol = "lnVol88", repstracol = "FCODE88",
                           fecacol = "Intactseed88", deadacol = "Dead1988",
                           nonobsacol = "Dormant1988", stageassign = lathframeln,
                           stagesize = "sizea", censorcol = "Missing1988",
                           censorkeep = NA, NAas0 = TRUE, censor = TRUE)

summary(lathvertln)
```

lefko3 *Historical and Ahistorical Population Projection Matrix Analysis*

Description

This package creates population projection matrices for use in population ecological analyses. The particular specialty of the package is the construction of historical matrices, which are 2-dimensional matrices comprising 3 time intervals of demographic information. The package is robust and can produce function-based and raw matrices for both standard ahistorical (i.e. 2 time interval) and historical analyses.

Details

The lefko package provides five categories of functions: 1. Data transformation and handling functions 2. Functions determining population characteristics from vertical data 3. Model building and selection 4. Matrix / integral projection model creation functions 5. Population dynamics analysis functions

It also includes example datasets complete with sample code.

Author(s)

Richard P. Shefferson <cdorm@g.ecc.u-tokyo.ac.jp>

Johan Ehrlén

References

Shefferson, R.P., J. Ehrlen, and S. Kurokawa. *In press*. *lefko3*: analyzing individual history through size-classified matrix population models. *Methods in Ecology and Evolution*.

`lmean` *Estimate Mean Projection Matrices*

Description

`lmean()` estimates mean projection matrices as element-wise arithmetic means.

Usage

```
lmean(mats, matsout = ”all”)
```

Arguments

`mats` A `lefkoMat` object.

`matsout` A string identifying which means to estimate. Option "pop" indicates population-level only, "patch" indicates patch-level only, and "all" indicates that both patch- and population-level means should be estimated.

Value

Yields a `lefkoMat` object with the following characteristics:

`A` A list of full mean projection matrices in order of sorted populations, patches, and years. These are typically estimated as the sums of the associated mean `U` and `F` matrices. All matrices output in the `matrix` class.

`U` A list of mean survival-transition matrices sorted as in `A`. All matrices output in the `matrix` class.

`F` A list of mean fecundity matrices sorted as in `A`. All matrices output in the `matrix` class.

`hstages` A data frame showing the pairing of ahistorical stages used to create historical stage pairs. Given if the MPM is historical.

`ahstages` A data frame detailing the characteristics of associated ahistorical stages.

`labels` A data frame detailing the order of population, patch, and year of each mean matrix. If `pop`, `patch`, or `year2` are NA in the original `labels` set, then these will be re-labeled as `A`, `1`, or `1`, respectively.

`matrixqc` A short vector describing the number of non-zero elements in `U` and `F` mean matrices, and the number of annual matrices.

`modelqc` The `qc` portion of the modelsuite input, if provided.

`dataqc` A vector showing the numbers of individuals and rows in the vertical dataset used as input.

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                    indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
ehrlen3mean$A[[1]]
```

`modelsearch` *Develop Best-fit Vital Rate Estimation Models For Matrix Development*

Description

`modelsearch()` returns both a table of vital rate estimating models and a best-fit model for each major vital rate estimated. The final output can be used as input in other functions within this package.

Usage

```
modelsearch(
  data,
  historical = TRUE,
  approach = "mixed",
  suite = "size",
  bestfit = "AICc&k",
  vitalrates = c("surv", "size", "fec"),
  surv = c("alive3", "alive2", "alive1"),
  obs = c("obsstatus3", "obsstatus2", "obsstatus1"),
  size = c("sizea3", "sizea2", "sizea1"),
  repst = c("repstatus3", "repstatus2", "repstatus1"),
  fec = c("feca3", "feca2", "feca1"),
  stage = c("stage3", "stage2", "stage1"),
  indiv = "individ",
  patch = NA,
  year = "year2",
  sizedist = "gaussian",
  fecdist = "gaussian",
  size.zero = FALSE,
  fec.zero = FALSE,
  patch.as.random = TRUE,
  year.as.random = TRUE,
  juvestimate = NA,
  juvsize = FALSE,
  fectime = 2,
  censor = NA,
  age = NA,
  indcova = NA,
  indcovb = NA,
  indcovc = NA,
  show.model.tables = TRUE,
  global.only = FALSE,
  quiet = FALSE
)
```

Arguments

`data` The vertical dataset to be used for analysis. This dataset should be of class `hfvdata`, but can also be a data frame formatted similarly to the output format provided by functions `verticalize3()` or `historicalize3()`, as long as all needed variables are properly designated.

`historical` A logical variable denoting whether to assess the effects of state in time *t*-1 in addition to state in time *t*. Defaults to TRUE.

`approach` The statistical approach to be taken for model building. The default is `lme4`, which uses the mixed model approach utilized in package `lme4`. Other options include `glm`, which uses `lm`, `glm`, and related functions in base R.

`suite` This describes the global model for each vital rate estimation and has the following possible values: `full`, includes main effects and all two-way interactions of size and reproductive status; `main`, includes main effects only of size and reproductive status; `size`, includes only size (also interactions between size in historical model); `rep`, includes only reproductive status (also interactions between status in historical model); `cons`, all vital rates estimated only as y-intercepts. If `approach = "glm"` and `year.as.random = FALSE`, then year is also included as a fixed effect, and, in the case of `full`, included in two-way interactions. Defaults to `size`.

`bestfit` A variable indicating the model selection criterion for the choice of best-fit model. The default is `AICc&k`, which chooses the best-fit model as the model with the lowest AICc or, if not the same model, then the model that has the lowest degrees of freedom among models with $\Delta AICc <= 2.0$. Alternatively, `AICc` may be chosen, in which case the best-fit model is simply the model with the lowest AICc value.

`vitalrates` A vector describing which vital rates will be estimated via linear modeling, with the following options: `surv`, survival probability; `obs`, observation probability; `size`, overall size; `repst`, probability of reproducing; and `fec`, amount of reproduction (overall fecundity). Defaults to `c("surv","size","fec")`.

`surv` A vector indicating the variable names coding for status as alive or dead in times *t*+1, *t*, and *t*-1, respectively. Defaults to `c("alive3","alive2","alive1")`.

`obs` A vector indicating the variable names coding for observation status in times *t*+1, *t*, and *t*-1, respectively. Defaults to `c("obsstatus3","obsstatus2","obsstatus1")`.

`size` A vector indicating the variable names coding for size in times *t*+1, *t*, and *t*-1, respectively. Defaults to `c("sizea3","sizea2","sizea1")`.

`repst` A vector indicating the variable names coding for reproductive status in times *t*+1, *t*, and *t*-1, respectively. Defaults to `c("repstatus3","repstatus2","repstatus1")`.

`fec` A vector indicating the variable names coding for fecundity in times *t*+1, *t*, and *t*-1, respectively. Defaults to `c("feca3","feca2","feca1")`.

`stage` A vector indicating the variables coding for stage in times *t*+1, *t*, and *t*-1. Defaults to `c("stage3","stage2","stage1")`.

`indiv` A variable indicating the variable name coding individual identity. Defaults to `individ`.

`patch` A variable indicating the variable name coding for patch, where patches are defined as permanent subgroups within the study population. Defaults to NA.

`year` A variable indicating the variable coding for observation time in time *t*. Defaults to `year2`.

`sizedist` The probability distribution used to model size. Options include `gaussian` for the Normal distribution (default), `poisson` for the Poisson distribution, and `negbin` for the negative binomial distribution.

`fecdist` The probability distribution used to model fecundity. Options include `gaussian` for the Normal distribution (default), `poisson` for the Poisson distribution, and `negbin` for the negative binomial distribution.

`size.zero` A logical variable indicating whether size distribution should be zero-inflated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE.

`fec.zero` A logical variable indicating whether fecundity distribution should be zero-inflated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE.

`patch.as.random`
If set to TRUE and `approach = "lme4"`, then `patch` is included as a random factor. If set to FALSE and `approach = "glm"`, then `patch` is included as a fixed factor. All other combinations of logical value and `approach` lead to `patch` not being included in modeling. Defaults to TRUE.

`year.as.random` If set to TRUE and `approach = "lme4"`, then `year` is included as a random factor. If set to FALSE, then `year` is included as a fixed factor. All other combinations of logical value and `approach` lead to `year` not being included in modeling. Defaults to TRUE.

`juvestimate` An optional variable denoting the stage name of the juvenile stage in the vertical dataset. If not NA, and `stage` is also given (see below), then vital rates listed in `vitalrates` other than `fec` will also be estimated from the juvenile stage to all adult stages. Defaults to NA, in which case juvenile vital rates are not estimated.

`juvsize` A logical variable denoting whether size should be used as a term in models involving transition from the juvenile stage. Defaults to FALSE, and is only used if `juvestimate` does not equal NA.

`fectime` A variable indicating which year of fecundity to use as the response term in fecundity models. Options include `2`, which refers to time *t*, and `3`, which refers to time *t*+1. Defaults to `2`.

`censor` A vector denoting the names of censoring variables in the dataset, in order from time *t*+1, followed by time *t*, and lastly followed by time *t*-1. Defaults to NA.

`age` Designates the name of the variable corresponding to age in the vertical dataset. Defaults to NA, in which case age is not included in linear models. Should only be used if building age x stage matrices.

`indcova` Vector designating the names in times *t*+1, *t*, and *t*-1 of an individual covariate. Defaults to NA.

`indcovb` Vector designating the names in times *t*+1, *t*, and *t*-1 of an individual covariate. Defaults to NA.

`indcovc` Vector designating the names in times *t*+1, *t*, and *t*-1 of an individual covariate. Defaults to NA.

`show.model.tables`
If set to TRUE, then includes full modeling tables in the output. Defaults to TRUE.

`global.only` If set to TRUE, then only global models will be built and evaluated. Defaults to FALSE.

`quiet` If set to TRUE, then model building and selection will proceed without warnings and diagnostic messages being issued. Note that this will not affect warnings and messages generated as models themselves are tested. Defaults to FALSE.

Value

This function yields an object of class `lefkoMod`, which is a list in which the first 9 elements are the best-fit models for survival, observation status, size, reproductive status, fecundity, juvenile survival, juvenile observation, juvenile size, and juvenile transition to reproduction, respectively, followed by 9 elements corresponding to the model tables for each of these vital rates, in order, followed by a single character element denoting the criterion used for model selection, and ending on a quality control vector:

`survival_model` Best-fit model of the binomial probability of survival from time *t* to time *t*+1. Defaults to 1.

`observation_model` Best-fit model of the binomial probability of observation in time *t*+1 given survival to that time. Defaults to 1.

`size_model` Best-fit model of size in time *t*+1 given survival to and observation in that time. Defaults to 1.

`repstatus_model` Best-fit model of the binomial probability of reproduction in time *t*+1, given survival to and observation in that time. Defaults to 1.

`fecundity_model` Best-fit model of fecundity in time *t*+1 given survival to, and observation and reproduction in that time. Defaults to 1.

`juv_survival_model` Best-fit model of the binomial probability of survival from time *t* to time *t*+1 of an immature individual. Defaults to 1.

`juv_observation_model` Best-fit model of the binomial probability of observation in time *t*+1 given survival to that time of an immature individual. Defaults to 1.

`juv_size_model` Best-fit model of size in time *t*+1 given survival to and observation in that time of an immature individual. Defaults to 1.

`juv_reproduction_model` Best-fit model of the binomial probability of reproduction in time *t*+1, given survival to and observation in that time of an individual that was immature in time *t*. This model is technically not a model of reproduction probability for individuals that are immature, rather reproduction probability here is given for individuals that are mature in time *t*+1 but immature in time *t*. Defaults to 1.

`survival_table` Full dredge model table of survival probability.

`observation_table` Full dredge model table of observationprobability.

`size_table` Full dredge model table of size.

`repstatus_table` Full dredge model table of reproduction probability.

`fecundity_table` Full dredge model table of fecundity.

`juv_survival_table` Full dredge model table of immature survival probability.

`juv_observation_table` Full dredge model table of immature observation probability.

`juv_size_table` Full dredge model table of immature size.

`juv_reproduction_table`
Full dredge model table of immature reproduction probability.

`criterion` Vharacter variable denoting the criterion used to determine the best-fit model.

`qc` Data frame with three variables: 1) Name of vital rate, 2) number of individuals used to model that vital rate, and 3) number of individual transitions used to model that vital rate.

The mechanics governing model building are fairly robust to errors and exceptions. The function attempts to build global models, and simplifies models automatically should model building fail. Model building proceeds through the functions `lm()` (GLM with Gaussian response), `glm()` (GLM with Poisson or binomial response), `glm.nb()` (GLM with negative binomial response), `zeroinfl()` (zero-inflated Poisson or negative binomial response), `lmer()` (mixed model with Gaussian response), `glmer()` (mixed model with binomial or Poisson response), `glmmTMB()` (mixed model with negative binomial, zero-inflated negative binomial, or zero-inflated Poisson response). See documentation related to these functions for further information.

Exhaustive model building and selection proceeds via the `dredge()` function in package `MuMIn`. This function is verbose, so that any errors and warnings developed during model building, model analysis, and model selection can be found and dealt with. Interpretations of errors during global model analysis may be found in documentation in for the functions and packages mentioned. Package `MuMIn` is used for model dredging (see dredge), and errors and warnings during dredging can be interpreted using the documentation for that package. Errors occurring during dredging lead to the adoption of the global model as the best-fit, and the user should view all logged errors and warnings to determine the best way to proceed. The `quiet = TRUE` option can be used to silence dredge warnings, but users should note that automated model selection can be viewed as a black box, and so great care should be taken to ensure that the models run make biological sense, and that model quality is prioritized.

Exhaustive model selection through dredging works best with larger datasets and fewer tested parameters. Setting `suite = "full"` may initiate a dredge that takes a dramatically long time, particularly if the model is historical, individual covariates are used, or a zero-inflated distribution is assumed. In such cases, the number of models built and tested will run at least in the millions. Small datasets will also increase the error associated with these tests, leading to adoption of simpler models overall. We do not yet offer a parallelization option for function `modelsearch()`, but plan to offer one in the future to speed this process up for particularly large global models.

Care must be taken to build models that test the impacts of state in time *t*-1 for historical models, and that do not test these impacts for ahistorical models. Ahistorical matrix modeling particularly will yield biased transition estimates if historical terms from models are ignored. This can be dealt with at the start of modeling by setting `historical = FALSE` for the ahistorical case, and `historical = TRUE` for the historical case.

Examples

```
data(lathyrus)

sizevector <- c(0, 4.6, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "Sz1nr", "Sz2nr", "Sz3nr", "Sz4nr", "Sz5nr",
                 "Sz6nr", "Sz7nr", "Sz8nr", "Sz9nr", "Sz1r", "Sz2r", "Sz3r", "Sz4r",
                 "Sz5r", "Sz6r", "Sz7r", "Sz8r", "Sz9r")
```

```
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 4.6, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
            0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5)

lathframeln <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                         obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                       indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvertln <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                           individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                           sizeacol = "lnVol88", repstracol = "FCODE88",
                           fecacol = "Intactseed88", deadacol = "Intactseed88",
                           nonobsacol = "Dormant1988", stageassign = lathframeln,
                           stagesize = "sizea", censorcol = "Missing1988",
                           censorkeep = NA, NAas0 = TRUE, censor = TRUE)

lathvertln$feca2 <- round(lathvertln$feca2)
lathvertln$feca1 <- round(lathvertln$feca1)
lathvertln$feca3 <- round(lathvertln$feca3)

lathmodelsln2 <- modelsearch(lathvertln, historical = FALSE, approach = "lme4", suite = "main",
                             vitalrates = c("surv", "obs", "size", "repst", "fec"),
                          juvestimate = "Sdl", bestfit = "AICc&k", sizedist = "gaussian",
                             fecdist = "poisson", indiv = "individ", patch = "patchid",
                          year = "year2", year.as.random = TRUE, patch.as.random = TRUE,
                             show.model.tables = TRUE)

lathmodelsln2
```

overwrite *Create an Overwrite Table for MPM Development*

Description

overwrite() returns a data frame describing which particular transitions within an ahistorical or historical projection matrix to overwrite with either given rates and probabilities, or other estimated transitions.

Usage

```
overwrite(
  stage3,
  stage2,
  stage1 = NA,
  eststage3 = NA,
  eststage2 = NA,
  eststage1 = NA,
  givenrate = NA,
  type = NA
)
```

Arguments

`stage3` The name of the stage in time *t*+1 in the transition to be replaced.

`stage2` The name of the stage in time *t* in the transition to be replaced.

`stage1` The name of the stage in time *t*-1 in the transition to be replaced. Only needed if a historical matrix is to be produced. Use `rep` if all reproductive stages are to be used, and leave empty or use `all` if all stages in stageframe are to be used.

`eststage3` The name of the stage to replace `stage3`. Only needed if a transition will be replaced by another estimated transition.

`eststage2` The name of the stage to replace `stage2`. Only needed if a transition will be replaced by another estimated transition.

`eststage1` The name of the stage to replace `stage1`. Only needed if a transition will be replaced by another estimated transition, and the matrix to be estimated is historical.

`givenrate` A fixed rate or probability to replace for the transition described by `stage3`, `stage2`, and `stage1`.

`type` A vector denoting the kind of transition that will be replaced. Should be entered as 1, S, or s for survival, or 2, F, or f for fecundity. Defaults to 1, for survival transition.

Value

A data frame that puts the above vectors together and can be used as input in `flefko3`, `flefko2`, `rlefko3`, and `rlefko2`.

Variables in this data frame include the following:

`stage3` Stage at time *t*+1 in the transition to be replaced.

`stage2` Stage at time *t* in the transition to be replaced.

`stage1` Stage at time *t*-1 in the transition to be replaced.

`eststage3` Stage at time *t*+1 in the transition to replace the transition designated by `stage3`, `stage2`, and `stage1`.

`eststage2` Stage at time *t* in the transition to replace the transition designated by `stage3`, `stage2`, and `stage1`.

`eststage1` Stage at time *t*-1 in the transition to replace the transition designated by `stage3`, `stage2`, and `stage1`.

`givenrate` A constant to be used as the value of the transition.

`convtype` Designates whether the transition is a survival-transition probability (1) or a fecundity rate (2).

Examples

```
cypover2r <- overwrite(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
                       "XSm", "Sm"), stage2 = c("SD", "SD", "P1", "P2", "P3",
                       "SL", "SL", "SL", "SL"), eststage3 = c(NA, NA, NA, NA,
                       NA, NA, "D", "XSm", "Sm"), eststage2 = c(NA, NA, NA, NA,
                       NA, NA, "XSm", "XSm", "XSm"), givenrate = c(0.1, 0.2,
                       0.2, 0.2, 0.25, 0.4, NA, NA, NA), type = c("S", "S", "S",
                       "S", "S", "S", "S", "S", "S"))

cypover2r

cypover3r <- overwrite(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
                       "SL", "SL", "D", "XSm", "Sm", "D", "XSm", "Sm"),
                       stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3",
                       "SL", "SL", "SL", "SL", "SL", "SL", "SL", "SL"),
                       stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2",
                       "P3", "SL", "P3", "P3", "P3", "SL", "SL", "SL"),
                       eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D",
                       "XSm", "Sm", "D", "XSm", "Sm"), eststage2 = c(NA, NA, NA,
                       NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
                       "XSm"), eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA,
                       "XSm", "XSm", "XSm", "XSm", "XSm", "XSm"),
                       givenrate = c(0.1, 0.1, 0.2, 0.2, 0.2, 0.2, 0.25, 0.4,
                       0.4, NA, NA, NA, NA, NA, NA), type = c("S", "S", "S",
                       "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S",
                       "S"))

cypover3r
```

---

repvalue3 *Reproductive Value Estimation*

Description

repvalue3() is a generic function that estimates returns the reproductive values of stages in a population projection matrix or a set of matrices. The specifics of estimation vary with the class of input object. This function is made to handle very large and sparse matrices supplied as lefkoMat objects or as individual matrices, and can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
repvalue3(mats)
```

Arguments

mats	A lefkoMat object, or population projection matrix.

Value

The value returned depends on the class of the `mats` argument. See related functions for details.

See Also

`repvalue3.lefkoMat()`

`repvalue3.matrix()`

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                    indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                          individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                          sizeacol = "Volume88", repstracol = "FCODE88",
                          fecacol = "Intactseed88", deadacol = "Dead1988",
                          nonobsacol = "Dormant1988", stageassign = lathframe,
                          stagesize = "sizea", censorcol = "Missing1988",
                          censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                        stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                    stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                    overwrite = lathover3, yearcol = "year2",
                    indivcol = "individ", reduce = TRUE)

ehrlen3mean <- lmean(ehrlen3)
repvalue3(ehrlen3mean)
```

`repvalue3.lefkoMat` *Estimate Reproductive Value for a lefkoMat Object*

Description

`repvalue3.lefkoMat()` returns the reproductive values for stages in a set set of population projection matrices provided as a `lefkoMat` object. This function can handle large and sparse matrices, and so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'lefkoMat'
repvalue3(mats)
```

Arguments

`mats`	An object of class `lefkoMat` object.

Value

This function returns the reproductive values for stages of a matrices within a `lefkoMat` object. The nature of the output depends on whether the `lefkoMat` object used as input is ahistorical or historical. In both cases, raw reproductive values are estimated as the left eigenvector associated with the largest real part of the dominant eigenvalue, divided by the first non-zero element of the left eigenvector. Eigen analysis is handled by the `eig_gen`() function in the C++ Armadillo library for square matrices with fewer than 400 rows, and `eigs_gen`() for larger matrices.

If an ahistorical matrix set is used as input, then the output is a data frame that includes the number of the matrix within the `$A` element of the input `lefkoMat` object, followed by the numeric stage designation, the stage name, and the reproductive value estimated (`repvalue`).

If a historical matrix set is used as input, then a list with two elements is output. The first element is a data frame showing the reproductive values given given in terms of across-year stage pairs, as estimated in the procedure described above. The order of variables in the data frame is: the matrix of origin, the stage names for stages in times *t* and *t*-1 respectively, the numeric stage designations corresponding to these stages, and the reproductive value (`rep_value`). The second element is a data frame showing the reproductive values of the basic stages in the associated stageframe. The reproductive values in this second data frame are estimated via the approach developed in Ehrlen (2000), in which each ahistorical stage's reproductive value is the average of the RVs summed by stage at time *t* weighted by the proportion of that stage pair within the historical stable stage distribution associated with the matrix.

See Also

`repvalue3()`

`repvalue3.matrix()`

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ", reduce = TRUE)

ehrlen3mean <- lmean(ehrlen3)
repvalue3(ehrlen3mean)
```

`repvalue3.matrix` *Estimate Reproductive Value for a Population Projection Matrix*

Description

`repvalue3.matrix()` returns the reproductive values for stages in a population projection matrix. The function makes no assumptions about whether the matrix is ahistorical and simply provides standard reproductive values corresponding to each row, meaning that the overall reproductive values of basic life history stages in a historical matrix are not provided (the `repvalue3.lefkoMat()`

function estimates these on the basis of stage description information provided in the lefkoMat object used as input in that function). This function can handle large and sparse matrices, and so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'matrix'
repvalue3(mats)
```

Arguments

mats	A population projection matrix.

Value

This function returns a vector data frame characterizing the reproductive values for stages of a population projection matrix. This is given as the left eigenvector associated with largest real part of the dominant eigenvalue, divided by the first non-zero element of the left eigenvector. Eigen analysis is handled by the eig_gen() function in the C++ Armadillo library for square matrices with fewer than 400 rows, and using the eigs_gen() function for larger matrices.

See Also

repvalue3()

repvalue3.lefkoMat()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                    indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                       censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                      stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                  stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                  overwrite = lathover3, yearcol = "year2",
                  indivcol = "individ", reduce = TRUE)

ehrlen3mean <- lmean(ehrlen3)
repvalue3(ehrlen3mean$A[[1]])
```

rlefko2 *Create Raw Ahistorical Population Projection Matrices*

Description

rlefko2() returns raw ahistorical MPMs, including the associated componenttransition and fecundity matrices, a data frame describing the ahistorical stages used, and a data frame describing the population, patch, and year associated with each matrix.

Usage

```
rlefko2(
  data,
  stageframe,
  year = "all",
  pop = NA,
  patch = NA,
  censor = FALSE,
  stages = NA,
  alive = c("alive3", "alive2"),
  size = c("sizea3", "sizea2"),
  repst = c("repstatus3", "repstatus2"),
  matst = c("matstatus3", "matstatus2"),
  fec = c("feca3", "feca2"),
  repmatrix = NA,
  overwrite = NA,
  yearcol = NA,
  popcol = NA,
  patchcol = NA,
  indivcol = NA,
  censorcol = NA,
  reduce = FALSE
)
```

Arguments

data
A vertical demographic data frame, with variables corresponding to the naming conventions in verticalize3().

stageframe
A stageframe object that includes information on the size, observation status, propagule status, immaturity status, and maturity status of each ahistorical stage. Should also incorporate bin widths if size is continuous.

year
A variable corresponding to year or observation time, or a set of such values, given in values associated with the year term used in linear model development. Can also equal all, in which case matrices will be estimated for all years. Defaults to all.

pop
A variable designating which populations will have matrices estimated. Should be set to specific population names, or to all if all populations should have matrices estimated.

patch
A variable designating which patches or subpopulations will have matrices estimated. Should be set to specific patch names, or to all if matrices should be estimated for all patches. Defaults to all.

censor
If TRUE, then data will be removed according to the variable set in censorcol, such that only data with censor values equal to 1 will remain. Defaults to FALSE.

stages
An optional vector denoting the names of the variables within the main vertical dataset coding for the stages of each individual in times *t*+1, *t*, and *t*-1. The names of stages in these variables should match those used in the stageframe exactly. If left blank, then rlefko3() will attempt to infer stages by matching values of alive, size, repst, and matst to characteristics noted in the associated stageframe.

alive
A vector of names of binomial variables corresponding to status as alive (1) or dead (0) in times *t*+1, *t*, and *t*-1, respectively.

size
A vector of names of variables coding size in times *t*+1 and *t*, respectively. Defaults to c("sizea3","sizea2").

repst
A vector of names of variables coding reproductive status in times *t*+1 and *t*, respectively. Defaults to c("repstatus3","repstatus2").

matst
A vector of names of variables coding maturity status in times *t*+1 and *t*, respectively. Defaults to c("matstatus3","matstatus2"). Must be supplied if stages is not provided.

fec
A vector of names of variables coding fecundity in times *t*+1 and *t*, respectively. Defaults to c("feca3","feca2").

repmatrix
A matrix composed mostly of 0s, with non-zero values for each potentially new individual (row) born to each reproductive stage (column). Entries act as multipliers on fecundity, with 1 equaling full fecundity.

`overwrite` A data frame developed with the `overwrite()` function describing transitions to be overwritten either with given values or with other estimated transitions.

`yearcol` The variable name or column number corresponding to time in time *t* in the dataset.

`popcol` The variable name or column number corresponding to the identity of the population.

`patchcol` The variable name or column number corresponding to patch in the dataset.

`indivcol` The variable name or column number coding individual identity.

`censorcol` The variable name or column number denoting the censor status. Only needed if `censor = TRUE`.

`reduce` A logical value denoting whether to remove historical stages associated with only zero transitions. These are removed only if all row and column sums in ALL matrices estimated equal 0. Defaults to FALSE.

Value

If all inputs are properly formatted, then this function will return an object of class `lefkoMat`. This includes:

`A` A list of full projection matrices in order of sorted populations, patches, and years. All matrices output in the `matrix` class.

`U` A list of survival-transition matrices sorted as in `A`. All matrices output in the `matrix` class.

`F` A list of fecundity matrices sorted as in `A`. All matrices output in the `matrix` class.

`hstages` Null for ahistorical matrices.

`ahstages` A data frame detailing the characteristics of associated ahistorical stages.

`labels` A data frame giving the population, patch, and year of each matrix in order.

`matrixqc` A short vector describing the number of non-zero elements in `U` and `F` matrices, and the number of annual matrices.

`dataqc` A vector showing the numbers of individuals and rows in the vertical dataset used as input.

Examples

```
data(cypdata)

sizevector <- c(0, 0, 0, 0, 0, 0, 1, 2.5, 4.5, 8, 17.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm",
                 "Sm", "Md", "Lg", "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
```

```
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1, 1, 2.5, 7)

cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
                          repstatus = repvector, obsstatus = obsvector,
                          matstatus = matvector, propstatus = propvector,
                          immstatus = immvector, indataset = indataset,
                          binhalfwidth = binvec)

cypraw_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
                          patchidcol = "patch", individcol = "plantid",
                          blocksize = 4, sizeacol = "Inf2.04", sizebcol = "Inf.04",
                          sizeccol = "Veg.04", repstracol = "Inf.04",
                          repstrbcol = "Inf2.04", fecacol = "Pod.04",
                          stageassign = cypframe_raw, stagesize = "sizeadded",
                          NAas0 = TRUE, NRasRep = TRUE)

rep_cyp_raw <- matrix(0, 11, 11)
rep_cyp_raw[1:2,7:11] <- 0.5

cypover2r <- overwrite(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
                       "XSm", "Sm"), stage2 = c("SD", "SD", "P1", "P2", "P3",
                       "SL", "SL", "SL", "SL"), eststage3 = c(NA, NA, NA, NA,
                       NA, NA, "D", "XSm", "Sm"), eststage2 = c(NA, NA, NA, NA,
                       NA, NA, "XSm", "XSm", "XSm"), givenrate = c(0.1, 0.2,
                       0.2, 0.2, 0.25, 0.4, NA, NA, NA), type = c("S", "S", "S",
                       "S", "S", "S", "S", "S", "S"))

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw, year = "all",
                       patch = "all", stages = c("stage3", "stage2", "stage1"),
                       size = c("size3added", "size2added"),
                       repmatrix = rep_cyp_raw, overwrite = cypover2r,
                       yearcol = "year2", patchcol = "patchid",
                       indivcol = "individ")

cypmatrix2r$A[[1]]
```

rlefko3 *Create Raw Historical Population Projection Matrices*

Description

rlefko3() returns raw historical MPMs, including the associated component transition and fecundity matrices, data frames describing the ahistorical stages used and the historical paired stages, and a data frame describing the population, patch, and year associated with each matrix.

Usage

```
rlefko3(
  data,
  stageframe,
  year = "all",
  pop = NA,
  patch = NA,
  censor = FALSE,
  stages = NA,
  alive = c("alive3", "alive2", "alive1"),
  size = c("sizea3", "sizea2", "sizea1"),
  repst = c("repstatus3", "repstatus2", "repstatus1"),
  matst = c("matstatus3", "matstatus2", "matstatus1"),
  fec = c("feca3", "feca2", "feca1"),
  repmatrix = NA,
  overwrite = NA,
  yearcol = NA,
  popcol = NA,
  patchcol = NA,
  indivcol = NA,
  censorcol = NA,
  reduce = FALSE
)
```

Arguments

`data` A vertical demographic data frame, with variables corresponding to the naming conventions in `verticalize3()`.

`stageframe` A stageframe object that includes information on the size, observation status, propagule status, immaturity status, and maturity status of each ahistorical stage. Should also incorporate bin widths if size is continuous.

`year` A variable corresponding to year or observation time, or a set of such values, given in values associated with the year term used in linear model development. Can also equal `all`, in which case matrices will be estimated for all years. Defaults to `all`.

`pop` A variable designating which populations will have matrices estimated. Should be set to specific population names, or to `all` if all populations should have matrices estimated.

`patch` A variable designating which patches or subpopulations will have matrices estimated. Should be set to specific patch names, or to `all` if matrices should be estimated for all patches. Defaults to `all`.

`censor` If TRUE, then data will be removed according to the variable set in `censorcol`, such that only data with censor values equal to 1 will remain. Defaults to FALSE.

`stages` An optional vector denoting the names of the variables within the main vertical dataset coding for the stages of each individual in times *t*+1, *t*, and *t*-1. The names of stages in these variables should match those used in the `stageframe` exactly. If left blank, then `rlefko3()` will attempt to infer stages by matching values of `alive`, `size`, `repst`, and `matst` to characteristics noted in the associated `stageframe`.

`alive` A vector of names of binomial variables corresponding to status as alive (1) or dead (0) in times *t*+1, *t*, and *t*-1, respectively.

`size` A vector of names of variables coding size in times *t*+1, *t*, and *t*-1, respectively. Defaults to `c("sizea3","sizea2","sizea1")`.

`repst` A vector of names of variables coding reproductive status in times *t*+1, *t*, and *t*-1, respectively. Defaults to `c("repstatus3","repstatus2","repstatus1")`.

`matst` A vector of names of variables coding maturity status in times *t*+1, *t*, and *t*-1, respectively. Defaults to `c("matstatus3","matstatus2","matstatus1")`. Must be supplied if `stages` is not provided.

`fec` A vector of names of variables coding fecundity in times *t*+1, *t*, and *t*-1, respectively. Defaults to `c("feca3","feca2","feca1")`.

`repmatrix` A matrix composed mostly of 0s, with non-zero values for each potentially new individual (row) born to each reproductive stage (column). Entries act as multipliers on fecundity, with 1 equaling full fecundity.

`overwrite` A data frame developed with the `overwrite()` function describing transitions to be overwritten either with given values or with other estimated transitions.

`yearcol` The variable name or column number corresponding to time in time *t* in the dataset.

`popcol` The variable name or column number corresponding to the identity of the population.

`patchcol` The variable name or column number corresponding to patch in the dataset.

`indivcol` The variable name or column number coding individual identity.

`censorcol` The variable name or column number denoting the censor status. Only needed if `censor = TRUE`.

`reduce` A logical value denoting whether to remove historical stages associated exclusively with zero transitions. These are removed only if all row and column sums in ALL matrices estimated equal 0. Defaults to FALSE.

Value

If all inputs are properly formatted, then this function will return either an object of class `lefkoMat`. Output includes:

`A` A list of full projection matrices in order of sorted populations, patches, and years. All matrices output in the `matrix` class.

`U` A list of survival-transition matrices sorted as in `A`. All matrices output in the `matrix` class.

`F` A list of fecundity matrices sorted as in `A`. All matrices output in the `matrix` class.

`hstages` A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs.

`ahstages` A data frame detailing the characteristics of associated ahistorical stages.

`labels` A data frame giving the population, patch, and year of each matrix in order.

`matrixqc` A short vector describing the number of non-zero elements in `U` and `F` matrices, and the number of annual matrices.

`dataqc` A vector showing the numbers of individuals and rows in the vertical dataset used as input.

Examples

```
data(cypdata)

sizevector <- c(0, 0, 0, 0, 0, 0, 1, 2.5, 4.5, 8, 17.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm",
                 "Sm", "Md", "Lg", "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1, 1, 2.5, 7)

cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
                          repstatus = repvector, obsstatus = obsvector,
                          matstatus = matvector, propstatus = propvector,
                          immstatus = immvector, indataset = indataset,
                          binhalfwidth = binvec)

cypraw_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
                          patchidcol = "patch", individcol = "plantid",
                          blocksize = 4, sizeacol = "Inf2.04", sizebcol = "Inf.04",
                          sizeccol = "Veg.04", repstracol = "Inf.04",
                          repstrbcol = "Inf2.04", fecacol = "Pod.04",
                          stageassign = cypframe_raw, stagesize = "sizeadded",
                          NAas0 = TRUE, NRasRep = TRUE)

rep_cyp_raw <- matrix(0, 11, 11)
rep_cyp_raw[1:2,7:11] <- 0.5

cypover3r <- overwrite(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
                       "SL", "SL", "D", "XSm", "Sm", "D", "XSm", "Sm"),
                       stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3",
                       "SL", "SL", "SL", "SL", "SL", "SL", "SL", "SL"),
                       stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2",
                       "P3", "SL", "P3", "P3", "P3", "SL", "SL", "SL"),
                       eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "D",
                       "XSm", "Sm", "D", "XSm", "Sm"), eststage2 = c(NA, NA, NA,
                       NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
                       "XSm"), eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA,
                       "XSm", "XSm", "XSm", "XSm", "XSm", "XSm"),
                       givenrate = c(0.1, 0.1, 0.2, 0.2, 0.2, 0.2, 0.25, 0.4,
                       0.4, NA, NA, NA, NA, NA, NA), type = c("S", "S", "S",
                       "S", "S", "S", "S", "S", "S", "S", "S", "S", "S", "S",
                       "S"))
```

```
cypmatrix3r <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw, year = "all",
                       patch = "all", stages = c("stage3", "stage2", "stage1"),
                       size = c("size3added", "size2added", "size1added"),
                       repmatrix = rep_cyp_raw, overwrite = cypover3r,
                       yearcol = "year2", patchcol = "patchid", indivcol = "individ")

summary(cypmatrix3r)
```

sensitivity3 *Calculate Sensitivity of Lambda to Matrix Elements*

Description

sensitivity3() is a generic function that returns the sensitivity of the deterministic population growth rate, lambda, to the elements of the matrix population model. This function is made to handle very large and sparse matrices supplied as lefkoMat objects or as individual matrices.

Usage

```
sensitivity3(mats)
```

Arguments

mats	A lefkoMat object, or population projection matrix, for which the stable stage distribution is desired.

Value

The value returned depends on the class of the mats argument.

See Also

sensitivity3.lefkoMat()

sensitivity3.matrix()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
```

```
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                    indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                          individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                          sizeacol = "Volume88", repstracol = "FCODE88",
                          fecacol = "Intactseed88", deadacol = "Dead1988",
                          nonobsacol = "Dormant1988", stageassign = lathframe,
                          stagesize = "sizea", censorcol = "Missing1988",
                          censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                        stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                    stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                    overwrite = lathover3, yearcol = "year2",
                    indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
sensitivity3(ehrlen3mean)
```

sensitivity3.lefkoMat *Calculate Sensitivity of Lambda to Matrix Elements for a lefkoMat Object*

Description

sensitivity3.lefkoMat() returns the sensitivities of lambda to elements of all $A matrices in an object of class lefkoMat. This function can handle large and sparse matrices, and so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'lefkoMat'
sensitivity3(mats)
```

Arguments

mats	An object of class lefkoMat.

Value

The output from this function depends on whether the input is a historical or ahistorical `lefkoMat` object. If the latter, then this function returns a list with two elements, the first being a list of sensitivity matrices corresponding to the `$A` matrices in a `lefkoMat` object, and the second being a data frame showing the order of stages. If the former, then the output also includes the equivalent sensitivities to ahistorical transitions estimated using the historical matrix, and a data frame detailing the order of historical paired stages.

See Also

`sensitivity3()`

`sensitivity3.matrix()`

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")
```

```
ehrlen3mean <- lmean(ehrlen3)
sensitivity3(ehrlen3mean)
```

sensitivity3.matrix *Calculate Sensitivity of Lambda to Matrix Elements for a Matrix*

Description

sensitivity3.matrix() returns the sensitivities of lambda to elements of a single matrix. This function can handle large and sparse matrices, and so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'matrix'
sensitivity3(mats)
```

Arguments

mats	An object of class matrix.

Value

This function returns a single sensitivity matrix.

See Also

sensitivity3()

sensitivity3.lefkoMat()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)
```

```
lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
sensitivity3(ehrlen3mean$A[[1]])
```

sf_create *Create a Stageframe for Population Matrix Projection Analysis*

Description

sf_create() returns a data frame describing each ahistorical life history stage in the life history model. This data frame can be used as input into MPM creation functions such as flefko3(), where it determines how each stage is treated during matrix estimation.

Usage

```
sf_create(
  sizes,
  stagenames = NA,
  repstatus = 1,
  obsstatus = 1,
  propstatus = NA,
  immstatus = NA,
  matstatus = 1,
  minage = NA,
  maxage = NA,
  indataset = NA,
  binhalfwidth = 0.5,
  ipmbins = 100,
  roundsize = 5
)
```

Arguments

`sizes` A numeric vector of the typical or representative size of each life history stage.

`stagenames` An optional vector of stage names, in the same order as elements in sizes. If an IPM or function-based matrix with many stages is desired, then two stages that occur within the dataset and represent the lower and upper size limits of the IPM must be marked as `ipm` in this vector. These stages must be mature stages, and should have all characteristics other than size equal. If two or more groups of stages, each with its own characteristics, are to be developed for an IPM, then an even number of stages with two stages marking the minimum and maximum size of each group should be marked, with all other characteristics equal within each group.

`repstatus` A vector denoting the binomial reproductive status of each life history stage. Defaults to 1.

`obsstatus` A vector denoting the binomial observation status of each life history stage. Defaults to 1, but may be changed for unobservable stages.

`propstatus` A vector denoting whether each life history stage is a propagule. Such stages are generally only used in fecundity estimation. Defaults to NA.

`immstatus` A vector denoting whether each stage is immature. Must be composed of binomial values if given. Defaults to NA.

`matstatus` A vector denoting whether each stage is mature. Must be composed of binomial values if given. Defaults to 1 for all stages defined in `sizes`.

`minage` An optional vector denoting the minimum age at which a stage can occur. Only used in age x stage matrix development. Defaults to NA.

`maxage` An optional vector denoting the maximum age at which a stage should occur. Only used in age x stage matrix development. Defaults to NA.

`indataset` A vector designating which stages are found within the dataset. While `rlefko2()` and `rlefko3()` can use all stages in the input dataset, `flefko3()` and `flefko2()` can only handle size-classified stages with non-overlapping combinations of size and reproductive status, plus one immature stage. Stages that do not actually exist within the dataset should be marked as 0 in this vector.

`binhalfwidth` A numeric vector giving the half-width of size bins. Required to classify individuals appropriately within size classes. Defaults to 0.5 for all sizes.

`ipmbins` If an IPM is desired, then this parameter sets the number of stages to create for that IPM. This number is in addition to any stages that are not size-classified. Defaults to 100, and numbers greater than this yield a warning about the loss of statistical power and increasing chance of matrix over-parameterization resulting from increasing numbers of stages.

`roundsize` This parameter sets the precision of size classification, and equals the number of digits used in rounding sizes. Defaults to 5.

Value

A data frame of class `stageframe`, which includes information on the stage name, size, reproductive status, observation status, propagule status, immaturity status, maturity status, presence within the core dataset, counts of similarly sized stages, raw bin half-width, and the minimum, center, and maximum of each size bin, as well as its width. If minimum and maximum ages were specified, then these are also included. Also includes an empty string variable that can be used to describe stages meaningfully. This object can be used as the `stageframe` input for `flefko3()` `flefko2()`, `rlefko3()`, and `rlefko2()`.

Variables in this data frame include the following:

`stagenames` The unique names of the stages to be analyzed.

`size` The typical or representative size at which each stage occurs.

`repstatus` A binomial variable showing whether each stage is reproductive.

`obsstatus` A binomial variable showing whether each stage is observable.

`propstatus` A binomial variable showing whether each stage is a propagule.

`immstatus` A binomial variable showing whether each stage can occur as immature.

`matstatus` A binomial variable showing whether each stage occurs in maturity.

`indataset` A binomial variable describing whether each stage occurs in the input dataset.

`binhalfwidth_raw`
The half-width of the size bin, as input.

`min_age` The minimum age at which the stage may occur.

`max_age` The maximum age at which the stage may occur.

`sizebin_min` The minimum size at which the stage may occur.

`sizebin_max` The maximum size at which the stage may occur.

`sizebin_center` The centroid of the size bin at which the stage may occur.

`sizebin_width` The width of the size bin corresponding to the stage.

`comments` A text field for stage descriptions.

Examples

```
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 2.5, 4.5, 8, 17.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm", "Sm", "Md", "Lg",
                 "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1, 1, 2.5, 7)

cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
                          repstatus = repvector, obsstatus = obsvector,
                          matstatus = matvector, propstatus = propvector,
                          immstatus = immvector, indataset = indataset,
                          binhalfwidth = binvec)

cypframe_raw$comments[(cypframe_raw$stagenames == "SD")] <- "Dormant seed"
cypframe_raw$comments[(cypframe_raw$stagenames == "P1")] <- "1st yr protocorm"
cypframe_raw$comments[(cypframe_raw$stagenames == "P2")] <- "2nd yr protocorm"
```

```
cypframe_raw$comments[(cypframe_raw$stagenames == "P3")] <- "3rd yr protocorm"
cypframe_raw$comments[(cypframe_raw$stagenames == "SL")] <- "Seedling"
cypframe_raw$comments[(cypframe_raw$stagenames == "D")] <- "Dormant adult"
cypframe_raw$comments[(cypframe_raw$stagenames == "XSm")] <- "Extra small adult (1 shoot)"
cypframe_raw$comments[(cypframe_raw$stagenames == "Sm")] <- "Small adult (2-3 shoots)"
cypframe_raw$comments[(cypframe_raw$stagenames == "Md")] <- "Medium adult (4-5 shoots)"
cypframe_raw$comments[(cypframe_raw$stagenames == "Lg")] <- "Large adult (6-10 shoots)"
cypframe_raw$comments[(cypframe_raw$stagenames == "XLg")] <- "Extra large adult (>10 shoots)"

cypframe_raw
```

sf_distrib *Test for overdispersion and zero inflation in size and fecundity*

Description

Function `sf_distrib` takes a historically formatted vertical data as input and tests whether size and fecundity data are dispersed according to a Poisson distribution (where mean = variance), and whether the number of 0s exceeds expectations.

Usage

```
sf_distrib(data, size = NA, fec = NA, repst = NA)
```

Arguments

`data`	A historical vertical data file, which is a data frame of class `hfvdata`.
`size`	The name or column number of the variable corresponding to size.
`fec`	The name or column number of the variable corresponding to fecundity. Note that the name of the variable should correspond to the proper time, either time *t* or time *t*-1.
`repst`	The name or column number of the variable corresponding to reproductive status in time *t*. Required if fecundity distribution will be tested.

Value

Produces text describing the degree and significance of overdispersion and zero inflation. The tests are chi-squared score tests based on the expectations of mean = variance, and 0s as abundant as predicted by the value of lambda estimated from the dataset. See van der Broek (1995) for more details.

Examples

```
data(lathyrus)

sizevector <- c(0, 4.6, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "Sz1nr", "Sz2nr", "Sz3nr", "Sz4nr", "Sz5nr",
                "Sz6nr", "Sz7nr", "Sz8nr", "Sz9nr", "Sz1r", "Sz2r", "Sz3r", "Sz4r",
                 "Sz5r", "Sz6r", "Sz7r", "Sz8r", "Sz9r")
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 4.6, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
            0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5)

lathframeln <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                         obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                         indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvertln <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                           individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                           sizeacol = "lnVol88", repstracol = "Intactseed88",
                           fecacol = "Intactseed88", deadacol = "Dead1988",
                           nonobsacol = "Dormant1988", stageassign = lathframeln,
                           stagesize = "sizea", censorcol = "Missing1988",
                           censorkeep = NA, NAas0 = TRUE, censor = TRUE)

lathvertln$feca2 <- round(lathvertln$feca2)
lathvertln$feca1 <- round(lathvertln$feca1)
lathvertln$feca3 <- round(lathvertln$feca3)

sf_distrib(lathvertln, fec = "feca2", repst = "repstatus2")
```

stablestage3 *Stable Stage Distribution Estimation*

Description

stablestage3() is a generic function that returns the stable stage distribution for a population projection matrix or set of matrices. This function is made to handle very large and sparse matrices supplied as lefkoMat objects or as individual matrices, and can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
stablestage3(mats)
```

Arguments

mats	A lefkoMat object, or population projection matrix, for which the stable stage distribution is desired.

Value

The value returned depends on the class of the mats argument. See related functions for details.

See Also

stablestage3.lefkoMat()

stablestage3.matrix()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
stablestage3(ehrlen3mean)
```

stablestage3.lefkoMat *Estimate Stable Stage Distribution for a lefkoMat Object*

Description

stablestage3.lefkoMat() returns the stable stage distributions of all $A matrices in an object of class lefkoMat. This function can handle large and sparse matrices, and so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'lefkoMat'
stablestage3(mats)
```

Arguments

mats	An object of class lefkoMat.

Value

This function returns the stable stage distributions corresponding to the matrices in a lefkoMat object. For square matrices with fewer than 400 rows, the stable stage distribution is given as the right eigenvector associated with largest real part of all eigenvalues estimated via the eig_gen() function in the C++ Armadillo library divided by the sum of the associated right eigenvector. For larger matrices, the function assumes that the matrix is sparse and conducts a similar calculation but using the eigs_gen() for sparse matrix eigen analysis.

The output depends on whether the lefkoMat object used as input is ahistorical or historical. If the former, then a single data frame is output, which includes the number of the matrix within the $A element of the input lefkoMat object, followed by the stage id (numeric and assigned through sf_create()), the stage name, and the estimated proportion of the stable stage distribution (ss_prop).

If a historical matrix is used as input, then two data frames are output into a list object. The $hist element contains a data frame where the stable stage distribution is given in terms of across-year stage pairs. The structure includes the matrix number, the numeric stage designations for stages in times *t* and *t*-1, respectively, followed by the respective stage names, and ending with the estimated proportion of the stable stage distribution for that stage within its matrix (ss_prop). The $ahist element contains the stable stage distribution in stages as given in the original stageframe. It includes a data frame with the matrix of origin, the numeric stage designation, stage name, and the stable stage distribution estimated as the sum of distribution elements from $hist corresponding to the equivalent stage in time *t*, irrespective of stage in time *t*-1.

See Also

stablestage3()

stablestage3.matrix()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
```

```
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                       obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                      indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
stablestage3(ehrlen3mean)
```

stablestage3.matrix *Estimate Stable Stage Distribution for a Population Projection Matrix*

Description

stablestage3.matrix() returns the stable stage distribution for a population projection matrix. This function can handle large and sparse matrices, and so can be used with large historical matrices, IPMs, age x stage matrices, as well as smaller ahistorical matrices.

Usage

```
## S3 method for class 'matrix'
stablestage3(mats)
```

Arguments

mats A population projection matrix of class matrix.

Value

This function returns the stable stage distribution corresponding to the input matrix. For square matrices with fewer than 400 rows, the stable stage distribution is given as the right eigenvector associated with largest real part of the eigenvalues estimated for the matrix via the eig_gen() function in the C++ Armadillo library, divided by the sum of the associated right eigenvector. For larger matrices, the matrix is assumed to be sparse and the calculation is conducted similarly but using eig_gens() instead.

See Also

stablestage3()

stablestage3.lefkoMat()

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                     obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                   indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)

lathrepm <- matrix(0, 7, 7)
lathrepm[1, 6] <- 0.345
lathrepm[2, 6] <- 0.054

lathover3 <- overwrite(stage3 = c("Sd", "Sd", "Sdl"), stage2 = c("Sd", "Sd", "Sd"),
                       stage1 = c("Sd", "rep", "rep"), givenrate = c(0.345, 0.054))

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = c(1989, 1990),
                   stages = c("stage3", "stage2", "stage1"), repmatrix = lathrepm,
                   overwrite = lathover3, yearcol = "year2",
                   indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
stablestage3(ehrlen3mean$A[[1]])
```

summary.lefkoMat   *Summary of Class "lefkoMat"*

Description

A function to simplify the viewing of basic information describing the matrices produced through functions `flefko3()`, `flefko2()`, `rlefko3()`, and `rlefko2()`.

Usage

```
## S3 method for class 'lefkoMat'
summary(object, ...)
```

Arguments

`object`	An object of class `lefkoMat`.
`...`	Other parameters.

Value

A summary of the object, showing the number of each type of matrix, the number of annual matrices, the number of estimated (non-zero) elements across all matrices and per matrix, the number of unique transitions in the dataset, and the number of individuals.

Examples

```
data(cypdata)

sizevector <- c(0, 0, 0, 0, 0, 0, 1, 2.5, 4.5, 8, 17.5)
stagevector <- c("SD", "P1", "P2", "P3", "SL", "D", "XSm",
                 "Sm", "Md", "Lg", "XLg")
repvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
obsvector <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
immvector <- c(0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1, 1, 2.5, 7)

cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
                          repstatus = repvector, obsstatus = obsvector,
                          matstatus = matvector, propstatus = propvector,
                          immstatus = immvector, indataset = indataset,
                          binhalfwidth = binvec)

cypraw_v1 <- verticalize3(data = cypdata, noyears = 6, firstyear = 2004,
                          patchidcol = "patch", individcol = "plantid",
                          blocksize = 4, sizeacol = "Inf2.04", sizebcol = "Inf.04",
                          sizeccol = "Veg.04", repstracol = "Inf.04",
                          repstrbcol = "Inf2.04", fecacol = "Pod.04",
                          stageassign = cypframe_raw, stagesize = "sizeadded",
                          NAas0 = TRUE, NRasRep = TRUE)

rep_cyp_raw <- matrix(0, 11, 11)
rep_cyp_raw[1:2,7:11] <- 0.5

cypover2r <- overwrite(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
                       "XSm", "Sm"), stage2 = c("SD", "SD", "P1", "P2", "P3",
                       "SL", "SL", "SL", "SL"), eststage3 = c(NA, NA, NA, NA,
                       NA, NA, "D", "XSm", "Sm"), eststage2 = c(NA, NA, NA, NA,
                       NA, NA, "XSm", "XSm", "XSm"), givenrate = c(0.1, 0.2,
                       0.2, 0.2, 0.25, 0.4, NA, NA, NA), type = c("S", "S", "S",
                       "S", "S", "S", "S", "S", "S"))

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw, year = "all",
                       patch = "all", stages = c("stage3", "stage2", "stage1"),
                       size = c("size3added", "size2added"),
                       repmatrix = rep_cyp_raw, overwrite = cypover2r,
                       yearcol = "year2", patchcol = "patchid",
                       indivcol = "individ")

summary(cypmatrix2r)
```

summary.lefkoMod *Summary of Class "lefkoMod"*

Description

A function to sumarize the viewable output for an R object of class lefkoMod. This function shows the best-fit models, summarizes the numbers of models in the model tables, shows the criterion used to determine the best-fit models, and provides some basic quality control information.

Usage

```
## S3 method for class 'lefkoMod'
summary(object, ...)
```

Arguments

object	An R object of class lefkoMod resulting from modelsearch().
...	Other parameters.

Value

A summary of the object, showing the best-fit models for all vital rates, with constants of 0 or 1 used for unestimated models. This is followed by a summary of the number of models tested per vital rate, and a table showing the names of the parameters used to model vital rates and represent tested factors. At the end is a section describing the number of individuals and individual transitions used to estimate each vital rate best-fit model.

Examples

```
data(lathyrus)

sizevector <- c(0, 4.6, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9)
stagevector <- c("Sd", "Sdl", "Dorm", "Sz1nr", "Sz2nr", "Sz3nr", "Sz4nr", "Sz5nr",
                 "Sz6nr", "Sz7nr", "Sz8nr", "Sz9nr", "Sz1r", "Sz2r", "Sz3r", "Sz4r",
                 "Sz5r", "Sz6r", "Sz7r", "Sz8r", "Sz9r")
repvector <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
obsvector <- c(0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
matvector <- c(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 4.6, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
            0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5)

lathframeln <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                      obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                    indataset = indataset, binhalfwidth = binvec, propstatus = propvector)

lathvertln <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                           individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                           sizeacol = "lnVol88", repstracol = "Intactseed88",
                           fecacol = "Intactseed88", deadacol = "Dead1988",
                           nonobsacol = "Dormant1988", stageassign = lathframeln,
                           stagesize = "sizea", censorcol = "Missing1988",
                           censorkeep = NA, NAas0 = TRUE, censor = TRUE)

lathvertln$feca2 <- round(lathvertln$feca2)
lathvertln$feca1 <- round(lathvertln$feca1)
lathvertln$feca3 <- round(lathvertln$feca3)

lathmodelsln2 <- modelsearch(lathvertln, historical = FALSE, approach = "lme4", suite = "main",
                               vitalrates = c("surv", "obs", "size", "repst", "fec"),
                           juvestimate = "Sdl", bestfit = "AICc&k", sizedist = "gaussian",
                               fecdist = "poisson", indiv = "individ", patch = "patchid",
                            year = "year2", year.as.random = TRUE, patch.as.random = TRUE,
                               show.model.tables = TRUE, quiet = TRUE)

summary(lathmodelsln2)
```

verticalize3 *Create Historical Vertical Data Frames From Horizontal Data Frames*

Description

verticalize3() returns a vertically formatted demographic data frame organized to create historical projection matrices, given a horizontally formatted input data frame.

Usage

```
verticalize3(
  data,
  noyears,
  firstyear,
  popidcol = 0,
  patchidcol = 0,
  individcol = 0,
  blocksize,
  xcol = 0,
  ycol = 0,
  juvcol = 0,
  sizeacol,
  sizebcol = 0,
  sizeccol = 0,
  repstracol = 0,
  repstrbcol = 0,
  fecacol = 0,
  fecbcol = 0,
  indcovacol = 0,
  indcovbcol = 0,
  indcovccol = 0,
  aliveacol = 0,
  deadacol = 0,
  obsacol = 0,
  nonobsacol = 0,
  censorcol = 0,
  repstrrel = 1,
  fecrel = 1,
  stagecol = 0,
  stageassign = NA,
  stagesize = NA,
  censorkeep = 0,
  censor = FALSE,
  spacing = NA,
  NAas0 = FALSE,
  NRasRep = FALSE,
  reduce = TRUE
)
```

Arguments

`data` The horizontal data file.

`noyears` The number of years or observation periods in the dataset.

`firstyear` The first year or time of observation.

`popidcol` A variable name or column number corresponding to the identity of the population for each individual.

`patchidcol` A variable name or column number corresponding to the identity of the patch for each individual, if patches have been designated within populations.

`individcol` A variable name or column number corresponding to the identity of each individual.

`blocksize` The number of variables corresponding to each time in the input dataset designated in `data`.

`xcol` A variable name or column number corresponding to the x coordinate of each individual in Cartesian space.

`ycol` A variable name or column number corresponding to the y coordinate of each individual in Cartesian space.

`juvcol` A variable name or column number that marks individuals in immature stages within the dataset. The `verticalize3()` function assumes that immature individuals are identified in this variable marked with a number equal to or greater than 1, and that mature individuals are marked as 0 or NA.

`sizeacol` A variable name or column number corresponding to the size entry associated with the first year or observation time in the dataset.

`sizebcol` A second variable name or column number corresponding to the size entry associated with the first year or observation time in the dataset.

`sizeccol` A third variable name or column number corresponding to the size entry associated with the first year or observation time in the dataset.

`repstracol` A variable name or column number corresponding to the production of reproductive structures, such as flowers, associated with the first year or observation period in the input dataset. This can be binomial or count data, and is used to in analysis of the probability of reproduction.

`repstrbcol` A second variable name or column number corresponding to the production of reproductive structures, such as flowers, associated with the first year or observation period in the input dataset. This can be binomial or count data, and is used to in analysis of the probability of reproduction.

`fecacol` A variable name or column number denoting fecundity associated with the first year or observation time in the input dataset. This may represent egg counts, fruit counts, seed production, etc.

`fecbcol` A second variable name or column number denoting fecundity associated with the first year or observation time in the input dataset. This may represent egg counts, fruit counts, seed production, etc.

`indcovacol` A variable name or column number corresponding to an individual covariate to be used in analysis.

`indcovbcol` A variable name or column number corresponding to an individual covariate to be used in analysis.

`indcovccol` A second variable name or column number corresponding to an individual covariate to be used in analysis.

`aliveacol` A variable name or column number that provides information on whether an individual is alive at a given time. If used, living status must be designated as binomial (living = 1, dead = 0).

`deadacol` A variable name or column number that provides information on whether an individual is alive at a given time. If used, dead status must be designated as binomial (dead = 1, living = 0).

`obsacol` A variable name or column number providing information on whether an individual is in an observable stage at a given time. If used, observation status must be designated as binomial (observed = 1, not observed = 0).

`nonobsacol` A variable name or column number providing information on whether an individual is in an unobservable stage at a given time. If used, observation status must be designated as binomial (not observed = 1, observed = 0).

`censorcol` A variable name or column number corresponding to the first entry of a censor variable, used to distinguish between entries to use and entries not to use, or to designate entries with special issues that require further attention. If used, this should be associated with the first year or observation time, and all other years or times must also have censor columns.

`repstrrel` This is a scalar multiplier on variable `repstrbcol` to make it equivalent to `repstracol`. This can be useful if two reproductive status variables have related but unequal units, for example if `repstracol` refers to one-flowered stems while `repstrbcol` refers to two-flowered stems. Defaults to 1.

`fecrel` This is a scalar multiplier on variable `fecbcol` to make it equivalent to `fecacol`. This can be useful if two fecundity variables have related but unequal units. Defaults to 1.

`stagecol` Optional variable name or column number corresponding to life history stage at a given time.

`stageassign` The stageframe object identifying the life history model being operationalized. Note that if `stagecol` is provided, then this stageframe is not used for stage designation.

`stagesize` A variable name or column number describing which size variable to use in stage estimation. Defaults to NA, and can also take `sizea`, `sizeb`, `sizec`, or `sizeadded`, depending on which size variable is chosen.

`censorkeep` The value of the censor variable identifying data to be included in analysis. Defaults to 0, but may take any value including NA.

`censor` A logical variable determining whether the output data should be censored using the variable defined in `censorcol`. Defaults to FALSE.

`spacing` The spacing at which density should be estimated, if density estimation is desired and x and y coordinates are supplied. Given in the same units as those used in the x and y coordinates given in `xcol` and `ycol`. Defaults to NA.

`NAas0` If TRUE, then all NA entries for size and fecundity variables will be set to 0. This can help increase the sample size analyzed by `modelsearch()`, but should only be used when it is clear that this substitution is biologically realistic. Defaults to FALSE.

`NRasRep` If TRUE, then will treat non-reproductive but mature individuals as reproductive during stage assignment. This can be useful when a matrix is desired without separation of reproductive and non-reproductive but mature stages of the same size. Only used if `stageassign` is set to a stageframe. Defaults to FALSE.

`reduce` A logical variable determining whether unused variables and some invariant state variables should be removed from the output dataset. Defaults to TRUE.

Value

If all inputs are properly formatted, then this function will output a historical vertical data frame (class `hfvdata`), meaning that the output data frame will have three consecutive times of size and reproductive data per individual per row. This data frame is in standard format for all functions used in `lefko3`, and so can be used without further modification.

Variables in this data frame include the following:

`rowid` Unique identifier for the row of the data frame.

`popid` Unique identifier for the population, if given.

`patchid` Unique identifier for patch within population, if given.

`individ` Unique identifier for the individual.

`year2` Year or time at time *t*.

`firstseen` Year or time of first observation.

`lastseen` Year or time of last observation.

`obsage` Observed age in time *t*, assuming first observation corresponds to age = 0.

`obslifespan` Observed lifespan, given as `lastseen -firstseen + 1`.

`xpos1,xpos2,xpos3`
X position in Cartesian space in times *t*-1, *t*, and *t*+1, respectively, if provided.

`ypos1,ypos2,ypos3`
Y position in Cartesian space in times *t*-1, *t*, and *t*+1, respectively, if provided.

`sizea1,sizea2,sizea3`
Main size measurement in times *t*-1, *t*, and *t*+1, respectively.

`sizeb1,sizeb2,sizeb3`
Secondary size measurement in times *t*-1, *t*, and *t*+1, respectively.

`sizec1,sizec2,sizec3`
Tertiary measurement in times *t*-1, *t*, and *t*+1, respectively.

`size1added,size2added,size3added`
Sum of primary, secondary, and tertiary size measurements in timea *t*-1, *t*, and *t*+1, respectively.

`repstra1,repstra2,repstra3`
Main numbers of reproductive structures in times *t*-1, *t*, and *t*+1, respectively.

`repstrb1,repstrb2,repstrb3`
: Secondary numbers of reproductive structures in times *t*-1, *t*, and *t*+1, respectively.

`repstr1added,repstr2added,repstr3added`
: Sum of primary and secondary reproductive structures in times *t*-1, *t*, and *t*+1, respectively.

`feca1,feca2,feca3`
: Main numbers of offspring in times *t*-1, *t*, and *t*+1, respectively.

`fecb1,fecb2, fecb3`
: Secondary numbers of offspring in times *t*-1, *t*, and *t*+1, respectively.

`fec1added,fec2added,fec3added`
: Sum of primary and secondary fecundity in times *t*-1, *t*, and *t*+1, respectively.

`censor1,censor2,censor3`
: Censor state values in times *t*-1, *t*, and *t*+1, respectively.

`juvgiven1,juvgiven2,juvgiven3`
: Binomial variable indicating whether individual is juvenile in times *t*-1, *t*, and *t*+1. Only given if `juvcol` is provided.

`obsstatus1,obsstatus2,obsstatus3`
: Binomial observation state in times *t*-1, *t*, and *t*+1, respectively.

`repstatus1,repstatus2,repstatus3`
: Binomial reproductive state in times *t*-1, *t*, and *t*+1, respectively.

`fecstatus1,fecstatus2,fecstatus3`
: Binomial offspring production state in times *t*-1, *t*, and *t*+1, respectively.

`matstatus1,matstatus2,matstatus3`
: Binomial maturity state in times *t*-1, *t*, and *t*+1, respectively.

`alive1,alive2,alive3`
: Binomial state as alive in times *t*-1, *t*, and *t*+1, respectively.

`density`
: Density of individuals per unit designated in `spacing`. Only given if spacing is not NA.

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sdl", "VSm", "Sm", "VLa", "Flo", "Dorm")
repvector <- c(0, 0, 0, 0, 0, 1, 0)
obsvector <- c(0, 1, 1, 1, 1, 1, 0)
matvector <- c(0, 0, 1, 1, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1, 1, 1)
binvec <- c(0, 100, 11, 103, 3500, 3800, 0.5)

lathframe <- sf_create(sizes = sizevector, stagenames = stagevector, repstatus = repvector,
                       obsstatus = obsvector, matstatus = matvector, immstatus = immvector,
                      indataset = indataset, binhalfwidth = binvec, propstatus = propvector)
```

```
lathvert <- verticalize3(lathyrus, noyears = 4, firstyear = 1988, patchidcol = "SUBPLOT",
                         individcol = "GENET", blocksize = 9, juvcol = "Seedling1988",
                         sizeacol = "Volume88", repstracol = "FCODE88",
                         fecacol = "Intactseed88", deadacol = "Dead1988",
                         nonobsacol = "Dormant1988", stageassign = lathframe,
                         stagesize = "sizea", censorcol = "Missing1988",
                         censorkeep = NA, censor = TRUE)
summary(lathvert)
```

Index