

Package ‘lefk03’

June 25, 2022

Type Package

Title Historical and Ahistorical Population Projection Matrix Analysis

Version 5.3.0

Date 2022-06-24

Description Complete analytical environment for the construction and analysis of matrix population models and integral projection models. Includes the ability to construct historical matrices, which are 2d matrices comprising 3 consecutive times of demographic information. Estimates both raw and function-based forms of historical and standard ahistorical matrices. It also estimates function-based age-by-stage matrices and raw and function-based Leslie matrices.

Encoding UTF-8

License GPL (>= 2)

URL <http://www.sheffersonlab.com/projects>

Imports Rcpp (>= 1.0.5), glmmTMB, lme4, MASS, methods, MuMIn, pscl, SparseM, stats, VGAM, grDevices

LinkingTo Rcpp, RcppArmadillo, BH

LazyData true

RoxygenNote 7.2.0

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 2022-06-24 22:40:02 UTC

R topics documented:

actualstage3	3
add_IM	6
afleko2	11
arleko2	20
beverton3	25
cond_diff	27
cond_hmpm	29
create_IM	31
create_pm	38
cypdata	39
cypvert	42
delete_IM	44
density_input	48
density_vr	52
diff_IM	55
elasticity3	58
elasticity3.lefkoMat	60
elasticity3.list	63
elasticity3.matrix	66
fleko2	68
fleko3	78
fleslie	89
f_projection3	93
hfv_qc	102
historicalize3	107
hist_null	115
image3	116
image3.lefkoElas	119
image3.lefkoMat	121
image3.lefkoSens	123
image3.list	126
image3.matrix	128
lambda3	130
lambda3.lefkoMat	133
lambda3.matrix	135
lathyrus	138
lefko3	142
lmean	143
logistic3	145
ltre3	147
ltre3.lefkoMat	149
modelsearch	153
overwrite	163
plot.lefkoProj	165
projection3	167
repvalue3	172

repvalue3.lefkoMat	175
repvalue3.matrix	178
ricker3	181
rlefko2	182
rlefko3	187
rleslie	193
sensitivity3	197
sensitivity3.lefkoMat	199
sensitivity3.list	202
sensitivity3.matrix	205
sf_create	208
sf_distrib	213
slambda3	217
stablestage3	220
stablestage3.lefkoMat	222
stablestage3.matrix	225
start_input	228
subset_IM	230
summary.lefkoCondMat	234
summary.lefkoElas	237
summary.lefkoLTRE	239
summary.lefkoMat	240
summary.lefkoMod	242
summary.lefkoProj	243
summary_hfv	247
supplemental	248
usher3	252
verticalize3	254
vrn_import	264

Index**270**

actualstage3	<i>Calculate Actual Stage, Age, Stage-Pair, or Age-Stage Distributions</i>
--------------	--

Description

Function `actualstage3()` shows the frequencies and proportions of each stage, stage pair, age-stage, or age in each year.

Usage

```
actualstage3(
  data,
  check_stage = TRUE,
  check_age = FALSE,
  historical = FALSE,
  year2 = "year2",
```

```

indices = c("stage3index", "stage2index", "stage1index"),
stagecol = c("stage3", "stage2", "stage1"),
agecol = "obsage",
remove.stage = "NotAlive"
)

```

Arguments

<code>data</code>	A demographic dataset in hfv format.
<code>check_stage</code>	A logical value indicating whether to assess frequencies and proportions of stages. Defaults to TRUE.
<code>check_age</code>	A logical value indicating whether to assess frequencies and proportions of ages. Defaults to FALSE.
<code>historical</code>	A logical value indicating whether the stage structure should be ahistorical (FALSE) or historical (TRUE). Defaults to FALSE.
<code>year2</code>	A string value indicating the name of the variable coding for monitoring occasion at time t .
<code>indices</code>	A vector of three strings, indicating the stage indices for times $t+1$, t , and $t-1$, respectively, in data. Defaults to <code>c("stage3index", "stage2index", "stage1index")</code> .
<code>stagecol</code>	A vector of three strings, indicating the stage name columns for times $t+1$, t , and $t-1$, respectively, in data. Defaults to <code>stagecol = c("stage3", "stage2", "stage1")</code> .
<code>agecol</code>	A single string indicating the age of individuals in time t . Defaults to <code>obsage</code> .
<code>remove.stage</code>	A string vector indicating the names of stages to remove from consideration. Defaults to <code>"NotAlive"</code> .

Value

A data frame with the following variables:

<code>rowid</code>	A string identifier term, equal to the monitoring occasion in time t and the stage index.
<code>stageindex</code>	The stageframe index of the stage. Only output if <code>check_stage = TRUE</code> .
<code>stage</code>	The name of each stage, or NA. Only output if <code>check_stage = TRUE</code> .
<code>stage2</code>	The name of the stage in time t . Only output if <code>check_stage = TRUE</code> .
<code>stage1</code>	The name of the stage in time $t-1$, or NA. Only output if <code>check_stage = TRUE</code> .
<code>age</code>	The age at time t . Only output if <code>check_age = TRUE</code> .
<code>year2</code>	Monitoring occasion in time t .
<code>frequency</code>	The number of individuals in the respective stage and time.
<code>actual_prop</code>	The proportion of individuals alive in time t in the respective stage.

Notes

This function produces frequencies and proportions of stages in hfv formatted data using stage index variables rather than stage name variables, and so requires the former. The latter is only required if the user wants to know the associated stage names.

Frequencies and proportions will be calculated for all times, including the last time, which is generally found in the stage3 columns of the last year2 entry in object data. The default is to treat the year2 entry for that time as $\max(\text{year2}) + 1$.

If `check_stage = TRUE` and `check_age = FALSE`, then this function will assess frequencies and proportions of stages or historical stage-pairs. If both `check_stage = TRUE` and `check_age = TRUE`, then this function will assess frequencies and proportions of age-stages. If `check_stage = FALSE` and `check_age = TRUE`, then the frequencies and proportions of ages only will be assessed.

Note that no stageframe is required for this function to operate. Stage names and their order are inferred directly from the object data.

Examples

```
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
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, 0, 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, 0, 1, 1, 1, 1, 1)
binvec <- c(0, 0, 0, 0, 0, 0.5, 0.5, 1.5, 1.5, 3.5, 5)
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
  "3rd yr protocorm", "Seedling", "Dormant adult",
  "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
  "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
  "Extra large adult (>14 shoots)")
cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
  repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
  propstatus = propvector, immstatus = immvector, indataset = indataset,
  binhalfwidth = binvec, comments = comments)

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)

all_stage_props <- actualstage3(cypraw_v1)
all_stage_props
```

add_1M *Add Matrices to lefkoMat Object*

Description

Function add_1M() adds matrices to lefkoMat objects.

Usage

```
add_1M(
  1M,
  Amats = NA,
  Umats = NA,
  Fmats = NA,
  UFdecomp = FALSE,
  entrystage = 1,
  pop = NA,
  patch = NA,
  year = NA
)
```

Arguments

1M	The lefkoMat object to add matrices to.
Amats	Either a single A matrix, or a list of A matrices. Not necessary if Umats and Fmats are both provided.
Umats	Either a single U matrix, or a list of U matrices. Not necessary if Amats and Fmats are both provided, or if UFdecomp = TRUE and entrystage is provided.
Fmats	Either a single F matrix, or a list of U matrices. Not necessary if Amats and Umats are both provided, or if UFdecomp = TRUE and entrystage is provided.
UFdecomp	A logical value indicating whether U and F matrices should be inferred from A matrices and the given entrystage. Defaults to TRUE.
entrystage	The stage or stages produced by reproductive individuals. Used to determine which transitions are reproductive for U-F decomposition. Defaults to 1, which corresponds to the first stage in the stageframe.
pop	The population designation for each matrix. If object 1M includes only a single population, then defaults to that designation. Otherwise requires a designation as input.
patch	The patch designation for each matrix. If object 1M includes only a single patch, then defaults to that designation. Otherwise requires a designation as input.
year	The designation for occasion at time t corresponding to each matrix. Cannot be left empty.

Value

A lefkoMat object incorporating the new matrices within the object input in 1M.

Notes

This function will not allow matrices of different dimension from those input in object IM to be added to that object.

Two of Amats, Umats, and Fmats must be provided for this function to proceed. Also, if Amats, Umats, and Fmats are all provided, then this function will default to replacing Amats with the sum of the respective Umats and Fmats.

See Also

[create_IM\(\)](#)

[delete_IM\(\)](#)

[subset_IM\(\)](#)

Examples

```
# These matrices are of 9 populations of the plant species Anthyllis
# vulneraria, and were originally published in Davison et al. (2010) Journal
# of Ecology 98:255-267 (doi: 10.1111/j.1365-2745.2009.01611.x).
```

```
sizevector <- c(1, 1, 2, 3) # These sizes are not from the original paper
stagevector <- c("Sd1", "Veg", "SmFlo", "LFlo")
repvector <- c(0, 0, 1, 1)
obsvector <- c(1, 1, 1, 1)
matvector <- c(0, 1, 1, 1)
immvector <- c(1, 0, 0, 0)
propvector <- c(0, 0, 0, 0)
indataset <- c(1, 1, 1, 1)
binvec <- c(0.5, 0.5, 0.5, 0.5)
```

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

```
# POPN C 2003-2004
```

```
XC3 <- matrix(c(0, 0, 1.74, 1.74,
0.208333333, 0, 0, 0.057142857,
0.041666667, 0.076923077, 0, 0,
0.083333333, 0.076923077, 0.066666667, 0.028571429), 4, 4, byrow = TRUE)
```

```
# 2004-2005
```

```
XC4 <- matrix(c(0, 0, 0.3, 0.6,
0.32183908, 0.142857143, 0, 0,
0.16091954, 0.285714286, 0, 0,
0.252873563, 0.285714286, 0.5, 0.6), 4, 4, byrow = TRUE)
```

```
# 2005-2006
```

```
XC5 <- matrix(c(0, 0, 0.50625, 0.675,
0, 0, 0, 0.035714286,
0.1, 0.068965517, 0.0625, 0.107142857,
```

```
0.3, 0.137931034, 0, 0.071428571), 4, 4, byrow = TRUE)

# POPN E 2003-2004
XE3 <- matrix(c(0, 0, 2.44, 6.569230769,
0.196428571, 0, 0, 0,
0.125, 0.5, 0, 0,
0.160714286, 0.5, 0.133333333, 0.076923077), 4, 4, byrow = TRUE)

XE4 <- matrix(c(0, 0, 0.45, 0.646153846,
0.06557377, 0.090909091, 0.125, 0,
0.032786885, 0, 0.125, 0.076923077,
0.049180328, 0, 0.125, 0.230769231), 4, 4, byrow = TRUE)

XE5 <- matrix(c(0, 0, 2.85, 3.99,
0.083333333, 0, 0, 0,
0, 0, 0, 0,
0.416666667, 0.1, 0, 0.1), 4, 4, byrow = TRUE)

# POPN F 2003-2004
XF3 <- matrix(c(0, 0, 1.815, 7.058333333,
0.075949367, 0, 0.05, 0.083333333,
0.139240506, 0, 0, 0.25,
0.075949367, 0, 0, 0.083333333), 4, 4, byrow = TRUE)

XF4 <- matrix(c(0, 0, 1.233333333, 7.4,
0.223880597, 0, 0.111111111, 0.142857143,
0.134328358, 0.272727273, 0.166666667, 0.142857143,
0.119402985, 0.363636364, 0.055555556, 0.142857143), 4, 4, byrow = TRUE)

XF5 <- matrix(c(0, 0, 1.06, 3.372727273,
0.073170732, 0.025, 0.033333333, 0,
0.036585366, 0.15, 0.1, 0.136363636,
0.06097561, 0.225, 0.166666667, 0.272727273), 4, 4, byrow = TRUE)

# POPN G 2003-2004
XG3 <- matrix(c(0, 0, 0.245454545, 2.1,
0, 0, 0.045454545, 0,
0.125, 0, 0.090909091, 0,
0.125, 0, 0.090909091, 0.333333333), 4, 4, byrow = TRUE)

XG4 <- matrix(c(0, 0, 1.1, 1.54,
0.111111111, 0, 0, 0,
0, 0, 0, 0,
0.111111111, 0, 0, 0), 4, 4, byrow = TRUE)

XG5 <- matrix(c(0, 0, 0, 1.5,
0, 0, 0, 0,
0.090909091, 0, 0, 0,
0.545454545, 0.5, 0, 0.5), 4, 4, byrow = TRUE)

# POPN L 2003-2004
XL3 <- matrix(c(0, 0, 1.785365854, 1.856521739,
0.128571429, 0, 0, 0.010869565,
```



```
0.028571429, 0, 0, 0,
0.014285714, 0, 0, 0.02173913), 4, 4, byrow = TRUE)

XL4 <- matrix(c(0, 0, 14.25, 16.625,
0.131443299, 0.057142857, 0, 0.25,
0.144329897, 0, 0, 0,
0.092783505, 0.2, 0, 0.25), 4, 4, byrow = TRUE)

XL5 <- matrix(c(0, 0, 0.594642857, 1.765909091,
0, 0, 0.017857143, 0,
0.021052632, 0.018518519, 0.035714286, 0.045454545,
0.021052632, 0.018518519, 0.035714286, 0.068181818), 4, 4, byrow = TRUE)

# POPN O 2003-2004
XO3 <- matrix(c(0, 0, 11.5, 2.775862069,
0.6, 0.285714286, 0.333333333, 0.24137931,
0.04, 0.142857143, 0, 0,
0.16, 0.285714286, 0, 0.172413793), 4, 4, byrow = TRUE)

XO4 <- matrix(c(0, 0, 3.78, 1.225,
0.28358209, 0.171052632, 0, 0.166666667,
0.084577114, 0.026315789, 0, 0.055555556,
0.139303483, 0.447368421, 0, 0.305555556), 4, 4, byrow = TRUE)

XO5 <- matrix(c(0, 0, 1.542857143, 1.035616438,
0.126984127, 0.105263158, 0.047619048, 0.054794521,
0.095238095, 0.157894737, 0.19047619, 0.082191781,
0.111111111, 0.223684211, 0, 0.356164384), 4, 4, byrow = TRUE)

# POPN Q 2003-2004
XQ3 <- matrix(c(0, 0, 0.15, 0.175,
0, 0, 0, 0,
0, 0, 0, 0,
1, 0, 0, 0), 4, 4, byrow = TRUE)

XQ4 <- matrix(c(0, 0, 0, 0.25,
0, 0, 0, 0,
0, 0, 0, 0,
1, 0.666666667, 0, 1), 4, 4, byrow = TRUE)

XQ5 <- matrix(c(0, 0, 0, 1.428571429,
0, 0, 0, 0.142857143,
0.25, 0, 0, 0,
0.25, 0, 0, 0.571428571), 4, 4, byrow = TRUE)

# POPN R 2003-2004
XR3 <- matrix(c(0, 0, 0.7, 0.6125,
0.25, 0, 0, 0.125,
0, 0, 0, 0,
0.25, 0.166666667, 0, 0.25), 4, 4, byrow = TRUE)

XR4 <- matrix(c(0, 0, 0, 0.6,
0.285714286, 0, 0, 0,
```

```

0.285714286, 0.333333333, 0, 0,
0.285714286, 0.333333333, 0, 1), 4, 4, byrow = TRUE)

XR5 <- matrix(c(0, 0, 0.7, 0.6125,
0, 0, 0, 0,
0, 0, 0, 0,
0.333333333, 0, 0.333333333, 0.625), 4, 4, byrow = TRUE)

# POPN S 2003-2004
XS3 <- matrix(c(0, 0, 2.1, 0.816666667,
0.166666667, 0, 0, 0,
0, 0, 0, 0,
0, 0, 0, 0.166666667), 4, 4, byrow = TRUE)

XS4 <- matrix(c(0, 0, 0, 7,
0.333333333, 0.5, 0, 0,
0, 0, 0, 0,
0.333333333, 0, 0, 1), 4, 4, byrow = TRUE)

XS5 <- matrix(c(0, 0, 0, 1.4,
0, 0, 0, 0,
0, 0, 0, 0.2,
0.111111111, 0.75, 0, 0.2), 4, 4, byrow = TRUE)

mats_list <- list(XC3, XC4, XC5, XE3, XE4, XE5, XF3, XF4, XF5, XG3, XG4, XG5,
  XL3, XL4, XL5, X03, X04, X05, XQ3, XQ4, XQ5, XR3, XR4, XR5, XS3, XS4, XS5)

yr_ord <- c(1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1,
  2, 3, 1, 2, 3)

pch_ord <- c(1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6, 7, 7, 7,
  8, 8, 8, 9, 9, 9)

anth_lefkoMat <- create_IM(mats_list, anthframe, hstages = NA, historical = FALSE,
  poporder = 1, patchorder = pch_ord, yearorder = yr_ord)

# POPN H (EXCLUDED FROM ANALYSIS B/C OF UNREALISTIC ELASTICITIES)
XH3 <- matrix(c(0, 0, 0.1125, 1.05,
0.2, 0, 0, 0,
0, 0.5, 0, 0,
0.2, 0.5, 0, 0), 4, 4, byrow = TRUE)

XH3u <- matrix(c(0, 0, 0, 0,
0.2, 0, 0, 0,
0, 0.5, 0, 0,
0.2, 0.5, 0, 0), 4, 4, byrow = TRUE)

XH4 <- matrix(c(0, 0, 0, 0,
0, 0, 0.5, 0,
0.8, 0.5, 0.25, 0.25,
0.2, 0, 0, 0.75), 4, 4, byrow = TRUE)

XH4u <- matrix(c(0, 0, 0, 0,

```

```

0, 0, 0.5, 0,
0.8, 0.5, 0.25, 0.25,
0.2, 0, 0, 0.75), 4, 4, byrow = TRUE)

XH5 <- matrix(c(0, 0, 0.2, 1.05,
0, 0, 0, 0,
0.001, 0.001, 0.333333333, 0, #ELEMENTS (3,1),(4,1),(3,2) REPLACED W NONZERO
0.001, 0, 0, 0), 4, 4, byrow = TRUE)

XH5u <- matrix(c(0, 0, 0, 0,
0, 0, 0, 0,
0.001, 0.001, 0.333333333, 0, #ELEMENTS (3,1),(4,1),(3,2) REPLACED W NONZERO
0.001, 0, 0, 0), 4, 4, byrow = TRUE)

anth_lefkoMat <- add_lm(anth_lefkoMat, Amats = list(XH3, XH4, XH5),
  Umats = list(XH3u, XH4u, XH5u), patch = c(10, 10, 10), year = c(1, 2, 3))

anth_lefkoMat

```

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 occasions given, including the associated component transition and fecundity matrices, data frames detailing the characteristics of ahistorical stages and the exact age-stage combinations corresponding to rows and columns in estimated matrices, and a data frame characterizing the patch and occasion combinations corresponding to these matrices.

Usage

```

aflefko2(
  year = "all",
  patch = "all",
  stageframe,
  supplement = NULL,
  repmatrix = NULL,
  overwrite = NULL,
  data = NULL,
  modelsuite = NULL,
  surv_model = NULL,
  obs_model = NULL,
  size_model = NULL,
  sizeb_model = NULL,
  sizec_model = NULL,
  repst_model = NULL,

```

```

fec_model = NULL,
jsurv_model = NULL,
jobs_model = NULL,
jsize_model = NULL,
jsizeb_model = NULL,
jsizec_model = NULL,
jrepst_model = NULL,
jmatst_model = NULL,
paramnames = NULL,
inda = NULL,
indb = NULL,
indc = NULL,
surv_dev = 0,
obs_dev = 0,
size_dev = 0,
sizeb_dev = 0,
sizec_dev = 0,
repst_dev = 0,
fec_dev = 0,
jsurv_dev = 0,
jobs_dev = 0,
jsize_dev = 0,
jsizeb_dev = 0,
jsizec_dev = 0,
jrepst_dev = 0,
jmatst_dev = 0,
density = NA,
repmo = 1,
random.inda = FALSE,
random.indb = FALSE,
random.indc = FALSE,
final_age = NA,
continue = TRUE,
prebreeding = TRUE,
negfec = FALSE,
ipm_method = "CDF",
reduce = FALSE,
err_check = FALSE,
exp_tol = 700,
theta_tol = 1e+08
)

```

Arguments

year	A variable corresponding to the observation occasion, or a set of such values, given in values associated with the year term used in linear model development. Defaults to "all", in which case matrices will be estimated for all occasions.
patch	A variable designating which patches or subpopulations will have matrices esti-

	<p>mated. Defaults to "all", but can also be set to specific patch names or a vector thereof.</p>
stageframe	<p>An object of class stageframe. These objects are generated by function <code>sf_create()</code>, and include information on the size, observation status, propagule status, reproduction status, immaturity status, maturity status, stage group, size bin widths, and other key characteristics of each ahistorical stage.</p>
supplement	<p>An optional data frame of class <code>lefkoSD</code> that provides supplemental data that should be incorporated into the MPM. Three kinds of data may be integrated this way: transitions to be estimated via the use of proxy transitions, transition overwrites from the literature or supplemental studies, and transition multipliers for survival and fecundity. This data frame should be produced using the <code>supplemental()</code> function. Can be used in place of or in addition to an overwrite table (see <code>overwrite</code> below) and a reproduction matrix (see <code>repmatrix</code> below).</p>
repmatrix	<p>An optional reproduction matrix. This matrix is composed mostly of 0s, with non-zero entries acting as element identifiers and multipliers for fecundity (with 1 equaling full fecundity). If left blank, and no <code>supplement</code> is provided, then <code>afleko2()</code> will assume that all stages marked as reproductive produce offspring at 1x that of estimated fecundity, and that offspring production will yield the first stage noted as propagule or immature. Must be the dimensions of an ahistorical stage-based matrix.</p>
overwrite	<p>An optional data frame developed with the <code>overwrite()</code> function describing transitions to be overwritten either with given values or with other estimated transitions. Note that this function supplements overwrite data provided in <code>supplement</code>.</p>
data	<p>The historical vertical demographic data frame used to estimate vital rates (class <code>hfvdata</code>), which is required to initialize times and patches properly. Variable names should correspond to the naming conventions in <code>verticalize3()</code> and <code>historicalize3()</code>. Not required if option <code>modelsuite</code> is set to a <code>vrn_input</code> object.</p>
modelsuite	<p>One of two kinds of lists. The first is a <code>lefkoMod</code> object holding the vital rate models and associated metadata. Alternatively, an object of class <code>vrn_input</code> may be provided. If given, then <code>surv_model</code>, <code>obs_model</code>, <code>size_model</code>, <code>sizeb_model</code>, <code>sizec_model</code>, <code>repst_model</code>, <code>fec_model</code>, <code>jsurv_model</code>, <code>jobs_model</code>, <code>jsize_model</code>, <code>jsizeb_model</code>, <code>jsizec_model</code>, <code>jrepst_model</code>, <code>jmatst_model</code>, and <code>paramnames</code> are not required. No models should include size or reproductive status in occasion $t-1$. Although this is optional input, it is recommended, and without it all vital rate model inputs (named <code>XX_model</code>) are required.</p>
surv_model	<p>A linear model predicting survival probability. This can be a model of class <code>glm</code> or <code>glmer</code>, and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t.</p>
obs_model	<p>A linear model predicting sprouting or observation probability. This can be a model of class <code>glm</code> or <code>glmer</code>, and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t.</p>

size_model	A linear model predicting primary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
sizeb_model	A linear model predicting secondary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
sizec_model	A linear model predicting tertiary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
repst_model	A linear model predicting reproduction probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
fec_model	A linear model predicting fecundity. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
jsurv_model	A linear model predicting juvenile survival probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
jobs_model	A linear model predicting juvenile sprouting or observation probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
jsize_model	A linear model predicting juvenile primary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
jsizeb_model	A linear model predicting juvenile secondary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
jsizec_model	A linear model predicting juvenile tertiary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
jrepst_model	A linear model predicting reproduction probability of a mature individual that was immature in time t . This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .

jmatst_model	A linear model predicting maturity probability of an individual that was immature in time t . This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
paramnames	A data frame with three columns, the first describing all terms used in linear modeling, the second (must be called <code>mainparams</code>) giving the general model terms that will be used in matrix creation, and the third showing the equivalent terms used in modeling (must be named <code>modelparams</code>). Function <code>create_pm()</code> can be used to create a skeleton <code>paramnames</code> object, which can then be edited. Only required if <code>modelsuite</code> is not supplied.
inda	Can be a single value to use for individual covariate <code>a</code> in all matrices, a pair of values to use for times t and $t-1$ in historical matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to <code>NULL</code> .
indb	Can be a single value to use for individual covariate <code>b</code> in all matrices, a pair of values to use for times t and $t-1$ in historical matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to <code>NULL</code> .
indc	Can be a single value to use for individual covariate <code>c</code> in all matrices, a pair of values to use for times t and $t-1$ in historical matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to <code>NULL</code> .
surv_dev	A numeric value to be added to the y-intercept in the linear model for survival probability. Defaults to 0 .
obs_dev	A numeric value to be added to the y-intercept in the linear model for observation probability. Defaults to 0 .
size_dev	A numeric value to be added to the y-intercept in the linear model for primary size. Defaults to 0 .
sizeb_dev	A numeric value to be added to the y-intercept in the linear model for secondary size. Defaults to 0 .
sizec_dev	A numeric value to be added to the y-intercept in the linear model for tertiary size. Defaults to 0 .
repst_dev	A numeric value to be added to the y-intercept in the linear model for probability of reproduction. Defaults to 0 .
fec_dev	A numeric value to be added to the y-intercept in the linear model for fecundity. Defaults to 0 .
jsurv_dev	A numeric value to be added to the y-intercept in the linear model for juvenile survival probability. Defaults to 0 .
jobs_dev	A numeric value to be added to the y-intercept in the linear model for juvenile observation probability. Defaults to 0 .
jsize_dev	A numeric value to be added to the y-intercept in the linear model for juvenile primary size. Defaults to 0 .
jsizeb_dev	A numeric value to be added to the y-intercept in the linear model for juvenile secondary size. Defaults to 0 .
jsizec_dev	A numeric value to be added to the y-intercept in the linear model for juvenile tertiary size. Defaults to 0 .

jrepst_dev	A numeric value to be added to the y-intercept in the linear model for juvenile reproduction probability. Defaults to 0.
jmatst_dev	A numeric value to be added to the y-intercept in the linear model for juvenile maturity probability. Defaults to 0.
density	A numeric value indicating density value to use to propagate matrices. Only needed if density is an explanatory term used in one or more vital rate models. Defaults to NA.
repmo	A scalar multiplier of fecundity. Defaults to 1.
random.ind	A logical value denoting whether to treat individual covariate a as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random.indb	A logical value denoting whether to treat individual covariate b as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random.indc	A logical value denoting whether to treat individual covariate c as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
final_age	The final age to model in the matrix, where the first age will be age 0. Defaults to the maximum age in the dataset.
continue	A logical value designating whether to allow continued survival of individuals past the final age noted in the stageframe, using the demographic characteristics of the final age. Defaults to TRUE.
prebreeding	A logical value indicating whether the life history model is a pre-breeding model. Defaults to TRUE.
negfec	A logical value denoting whether fecundity values estimated to be negative should be reset to 0. Defaults to FALSE.
ipm_method	A string indicating what method to use to estimate size transition probabilities, if size is treated as continuous. Options include: "midpoint", which utilizes the midpoint method; and "CDF", which uses the cumulative distribution function. Defaults to "CDF".
reduce	A logical value denoting whether to remove age-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.
err_check	A logical value indicating whether to append matrices of vital rate probabilities associated with each matrix to the lefkoMat object generated. These matrices are developed internally and can be used for error checking (see element out in Value section below for details). Defaults to FALSE.
exp_tol	A numeric value used to indicate a maximum value to set exponents to in the core kernel to prevent numerical overflow. Defaults to 700.
theta_tol	A numeric value used to indicate a maximum value to theta as used in the negative binomial probability density kernel. Defaults to 100000000, but can be reset to other values during error checking.

Value

If all inputs are properly formatted, then this function will return an object of class `lefkoMat`, which is a list that holds the matrix projection model and all of its metadata. The structure has the following elements:

<code>A</code>	A list of full projection matrices in order of sorted patches and occasions. All matrices output in R's <code>matrix</code> class.
<code>U</code>	A list of survival transition matrices sorted as in <code>A</code> . All matrices output in R's <code>matrix</code> class.
<code>F</code>	A list of fecundity matrices sorted as in <code>A</code> . All matrices output in R's <code>matrix</code> class.
<code>hstages</code>	A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs. Set to <code>NA</code> for age-by-stage MPMs.
<code>agestages</code>	A data frame showing the stage number and stage name corresponding to <code>ahstages</code> , as well as the associated age, of each row in each age-by-stage matrix.
<code>ahstages</code>	A data frame detailing the characteristics of associated ahistorical stages, in the form of a modified stageframe that includes status as an entry stage through reproduction.
<code>labels</code>	A data frame giving the patch and year of each matrix in order. In <code>afleko2()</code> , only one population may be analyzed at once.
<code>matrixqc</code>	A short vector describing the number of non-zero elements in <code>U</code> and <code>F</code> matrices, and the number of annual matrices.
<code>modelqc</code>	This is the <code>qc</code> portion of the <code>modelsuite</code> input.
<code>prob_out</code>	An optional element only added if <code>err_check = TRUE</code> . This is a list of vital rate probability matrices, with 7 columns in the order of survival, observation probability, reproduction probability, primary size transition probability, secondary size transition probability, tertiary size transition probability, and probability of juvenile transition to maturity.
<code>allstages</code>	An optional element only added if <code>err_check = TRUE</code> . This is a data frame giving the values used to determine each matrix element capable of being estimated.

Notes

Unlike `rleko2()`, `rleko3()`, `arleko2()`, and `rleslie()`, this function does not currently distinguish populations. Users wishing to use the same vital rate models across populations should label them as patches (though we do not advise this approach, as populations should typically be treated as statistically independent).

This function will yield incorrect estimates if the models utilized incorporate state in occasion $t-1$. Only use models developed testing for ahistorical effects.

The default behavior of this function is to estimate fecundity with regards to transitions specified via associated fecundity multipliers in either `supplement` or `repmatrix`. If both of these fields are left empty, then fecundity will be estimated at full for all transitions leading from reproductive stages to immature and propagule stages. However, if a `supplement` is provided and a `repmatrix` is not, or if `repmatrix` is set to 0, then only fecundity transitions noted in the `supplement` will be set to non-zero values. To use the default behavior of setting all reproductive stages to reproduce

at full fecundity into immature and propagule stages but also incorporate given or proxy survival transitions, input those given and proxy transitions through the `overwrite` option.

The reproduction matrix (field `repmatrix`) may only be supplied as `ahistorical`. If provided as `historical`, then `afleko2()` will fail and produce an error.

Stageframes used in this function should include ages for minimum and maximum age for each stage. NAs are treated as 0s in minimum age, and as `final_age` for maximum age.

Users may at times wish to estimate MPMs using a dataset incorporating multiple patches or subpopulations, but without discriminating between those patches or subpopulations. Should the aim of analysis be a general MPM that does not distinguish these patches or subpopulations, the `modelsearch()` run should not include patch terms.

Input options including multiple variable names must be entered in the order of variables in occasion $t+1$ and t . Rearranging the order will lead to erroneous calculations, and may lead to fatal errors.

Care should be taken to match the random status of year and patch to the states of those variables within the `modelsuite`. If they do not match, then they will be treated as zeroes in vital rate estimation.

The `ipm_method` function gives the option of using two different means of estimating the probability of size transition. The midpoint method ("`midpoint`") refers to the method in which the probability is estimated by first estimating the probability associated with transition from the exact size at the midpoint of the size class using the corresponding probability density function, and then multiplying that value by the bin width of the size class. Doak et al. 2021 (Ecological Monographs) noted that this method can produce biased results, with total size transitions associated with a specific size not totaling to 1.0 and even specific size transition probabilities capable of being estimated at values greater than 1.0. The alternative and default method, "`CDF`", uses the corresponding cumulative density function to estimate the probability of size transition as the cumulative probability of size transition at the greater limit of the size class minus the cumulative probability of size transition at the lower limit of the size class. The latter method avoids this bias. Note, however, that both methods are exact and unbiased for the Poisson and negative binomial distributions.

Under the Gaussian and gamma size distributions, the number of estimated parameters may differ between the two `ipm_method` settings. Because the midpoint method has a tendency to incorporate upward bias in the estimation of size transition probabilities, it is more likely to yield non-zero values when the true probability is extremely close to 0. This will result in the `summary.lefkoMat` function yielding higher numbers of estimated parameters than the `ipm_method = "CDF"` yields in some cases.

Using the `err_check` option will produce a matrix of 7 columns, each characterizing a different vital rate. The product of each row yields an element in the associated U matrix. The number and order of elements in each column of this matrix matches the associated matrix in column vector format. Use of this option is generally for the purposes of debugging code.

Individual covariates are treated as categorical only if they are set as random terms. Fixed categorical individual covariates are currently not allowed. However, such terms may be supplied if the `modelsuite` option is set to a `vrn_input` object. In that case, the user should also set the logical random switch for the individual covariate to be used to `TRUE` (e.g., `random.indv = TRUE`).

See Also

[fleko3\(\)](#)

[fleko2\(\)](#)

```
fleslie()
arleko2()
rleko3()
rleko2()
rleslie()
```

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)
obsvector <- c(0, 1, 0, 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)
immvector <- c(1, 1, 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)

indataset <- c(0, 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)
maxima <- c(NA, 1, 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)

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

lathvertIn <- 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 = lathframeIn,
stagesize = "sizea", censorcol = "Missing1988", censorkeep = NA,
NAas0 = TRUE, censor = TRUE)

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

lathmodelsIn2 <- modelsearch(lathvertIn, 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", age = "obsage",
```

```

year.as.random = TRUE, patch.as.random = TRUE, show.model.tables = TRUE,
quiet = TRUE)

# Here we use supplemental() to provide overwrite and reproductive info
lathsupp2 <- supplemental(stage3 = c("Sd", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "rep", "rep"),
  givenrate = c(0.345, 0.054, NA, NA),
  multiplier = c(NA, NA, 0.345, 0.054),
  type = c(1, 1, 3, 3), stageframe = lathframe1n, historical = FALSE)

lathmat2age <- arlefko2(year = "all", patch = "all",
  stageframe = lathframe1n, modelsuite = lathmodels1n2, data = lathvert1n,
  supplement = lathsupp2, final_age = 3, continue = TRUE, reduce = FALSE)

summary(lathmat2age)

```

arlefko2

Create Raw Ahistorical Age x Stage Matrix Projection Model

Description

Function `arlefko2()` returns raw ahistorical age x stage MPMs corresponding to the patches and occasion times given, including the associated component transition and fecundity matrices, data frames detailing the characteristics of ahistorical stages and the exact age-stage combinations corresponding to rows and columns in estimated matrices, and a data frame characterizing the patch and occasion time combinations corresponding to these matrices.

Usage

```

arlefko2(
  data,
  stageframe,
  year = "all",
  pop = NA,
  patch = NA,
  censor = FALSE,
  stages = NA,
  alive = c("alive3", "alive2"),
  size = c("sizea3", "sizea2"),
  sizeb = c(NA, NA),
  sizec = c(NA, NA),
  repst = c("repstatus3", "repstatus2"),
  matst = c("matstatus3", "matstatus2"),
  fec = c("feca3", "feca2"),
  supplement = NULL,
  repmatrix = NULL,
  overwrite = NULL,

```

```

agecol = "obsage",
yearcol = NA,
popcol = NA,
patchcol = NA,
indivcol = NA,
censorcol = NA,
censorkeep = 0,
final_age = NA,
continue = TRUE,
prebreeding = TRUE,
reduce = FALSE
)

```

Arguments

data	A vertical demographic data frame, with variables corresponding to the naming conventions in <code>verticalize3()</code> and <code>historicalize3()</code> .
stageframe	A stageframe object that includes information on the size, observation status, propagule status, reproduction status, immaturity status, and maturity status of each ahistorical stage. Should also incorporate bin widths if size is continuous.
year	A variable corresponding to observation occasion, or a set of such values, given in values associated with the year term used in linear model development. Defaults to "all", in which case matrices will be estimated for all occasions.
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 NA, in which case patch designations are ignored.
censor	If TRUE, then data will be removed according to the variable set in <code>censorcol</code> , such that only data with censor values equal to <code>censorkeep</code> 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 occasions $t+1$ and t . The names of stages in these variables should match those used in the <code>stageframe</code> exactly. If left blank, then <code>arlefko2()</code> will attempt to infer stages by matching values of <code>alive</code> , <code>size</code> , <code>repst</code> , and <code>matst</code> to characteristics noted in the associated <code>stageframe</code> .
alive	A vector of names of binomial variables corresponding to status as alive (1) or dead (0) in occasions $t+1$ and t , respectively.
size	A vector of names of variables coding the primary size variable in occasions $t+1$ and t , respectively. Defaults to <code>c("sizea3", "sizea2")</code> .
sizeb	A vector of names of variables coding the secondary size variable in occasions $t+1$ and t , respectively. Defaults to <code>c(NA, NA)</code> .

sizec	A vector of names of variables coding the tertiary size variable in occasions $t+1$ and t , respectively. Defaults to <code>c(NA, NA)</code> .
repst	A vector of names of variables coding reproductive status in occasions $t+1$ and t , respectively. Defaults to <code>c("repstatus3", "repstatus2")</code> . Must be supplied if stages is not provided.
matst	A vector of names of variables coding maturity status in occasions $t+1$ and t , respectively. Defaults to <code>c("matstatus3", "matstatus2")</code> . Must be supplied if stages is not provided.
fec	A vector of names of variables coding fecundity in occasions $t+1$ and t , respectively. Defaults to <code>c("feca3", "feca2")</code> .
supplement	An optional data frame of class <code>lefkoSD</code> that provides supplemental data that should be incorporated into the MPM. Three kinds of data may be integrated this way: transitions to be estimated via the use of proxy transitions, transition overwrites from the literature or supplemental studies, and transition multipliers for survival and fecundity. This data frame should be produced using the <code>supplemental()</code> function. Can be used in place of or in addition to an overwrite table (see <code>overwrite</code> below) and a reproduction matrix (see <code>repmatrix</code> below).
repmatrix	An optional reproduction matrix. This matrix is composed mostly of 0s, with non-zero entries acting as element identifiers and multipliers for fecundity (with 1 equaling full fecundity). If left blank, and no supplement is provided, then <code>arlefko2()</code> will assume that all stages marked as reproductive produce offspring at 1x that of estimated fecundity, and that offspring production will yield the first stage noted as propagule or immature. To prevent this behavior, input just <code>0</code> , which will result in fecundity being estimated only for transitions noted in supplement above. Must be the dimensions of an ahistorical stage-based matrix.
overwrite	An optional data frame developed with the <code>overwrite()</code> function describing transitions to be overwritten either with given values or with other estimated transitions. Note that this function supplements overwrite data provided in supplement.
agecol	The variable name or column number coding for age in time t .
yearcol	The variable name or column number corresponding to occasion 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.
ensorcol	The variable name or column number denoting the censor status. Only needed if <code>ensor = TRUE</code> .
ensorkeep	The value of the censor variable denoting data elements to keep. Defaults to <code>0</code> .
final_age	The final age to model in the matrix. Defaults to the maximum age in the dataset.
continue	A logical value designating whether to allow continued survival of individuals past the final age noted in the stageframe, using the demographic characteristics of the final age. Defaults to <code>TRUE</code> .

prebreeding	A logical value indicating whether the life history model is a pre-breeding model. Defaults to TRUE.
reduce	A logical value denoting whether to remove historical stages associated with only zero transitions. These are removed only if the respective 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 that holds the matrix projection model and all of its metadata. Its structure is a list with the following elements:

A	A list of full projection matrices in order of sorted patches and occasions. All matrices output in R's <code>matrix</code> class.
U	A list of survival transition matrices sorted as in A. All matrices output in R's <code>matrix</code> class.
F	A list of fecundity matrices sorted as in A. All matrices output in R's <code>matrix</code> class.
hstages	A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs. Set to NA for age-by-stage MPMs.
agestages	A data frame showing the stage number and stage name corresponding to ahstages, as well as the associated age, of each row in each age-by-stage matrix.
ahstages	A data frame detailing the characteristics of associated ahistorical stages, in the form of a modified stageframe that includes status as an entry stage through reproduction.
labels	A data frame giving the patch and year of each matrix in order. In <code>arlefko2()</code> , only one population may be analyzed at once, and so <code>pop = NA</code>
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.
prob_out	An optional element only added if <code>err_check = TRUE</code> . This is a list of vital rate probability matrices, with 6 columns in the order of survival, observation probability, reproduction probability, primary size transition probability, secondary size transition probability, and tertiary size transition probability.

Notes

The default behavior of this function is to estimate fecundity with regards to transitions specified via associated fecundity multipliers in either `supplement` or `repmatrix`. If both of these fields are left empty, then fecundity will be estimated at full for all transitions leading from reproductive stages to immature and propagule stages. However, if a `supplement` is provided and a `repmatrix` is not, or if `repmatrix` is set to 0, then only fecundity transitions noted in the `supplement` will be set to non-zero values. To use the default behavior of setting all reproductive stages to reproduce at full fecundity into immature and propagule stages but also incorporate given or proxy survival transitions, input those given and proxy transitions through the `overwrite` options.

The reproduction matrix (field `repmatrix`) may only be supplied as `ahistorical`. If provided as `historical`, then `rlefko2()` will fail and produce an error.

Users may at times wish to estimate MPMs using a dataset incorporating multiple patches or subpopulations. Should the aim of analysis be a general MPM that does not distinguish these patches or subpopulations, the `patchcol` variable should be left to `NA`, which is the default. Otherwise the variable identifying patch needs to be named.

Input options including multiple variable names must be entered in the order of variables in occasion $t+1$ and t . Rearranging the order WILL lead to erroneous calculations, and may lead to fatal errors.

Although this function is capable of assigning stages given an input stageframe, it lacks the power of `verticalize3()` and `historicalize3()` in this regard. Users are strongly encouraged to use the latter two functions for stage assignment.

See Also

`flefko3()`
`flefko2()`
`aflefko2()`
`fleslie()`
`rlefko3()`
`rlefko2()`
`rleslie()`

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)
minagevec <- c(1, 1, 2, 3, 4, 5, 5, 5, 5, 5, 5) # Might have to subtract 1 from everything
maxagevec <- c(rep(NA, 11))

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

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, age_offset = 4)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type =c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cyp_mats <- arlefko2(data = cypraw_v1, stageframe = cypframe_raw, year = "all",
  patch = NA, censor = FALSE, stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), fec = c("feca3", "feca2"),
  supplement = cypsupp2r, agecol = "obsage", yearcol = "year2",
  patchcol = "patchid", indivcol = "individ", prebreeding = TRUE, final_age = NA,
  continue = TRUE, reduce = FALSE)
summary(cyp_mats)

```

beverton3

Two-parameter Beverton-Holt function

Description

Function `beverton3()` creates a vector of values produced by the two-parameter Beverton-Holt function as applied with a user-specified time lag. The two-parameter Beverton-Holt function is given as $\phi_{t+1} = \phi_t \alpha / (1 + \beta n_t)$. Here, if no separate_N vector is provided, then $n_t = \phi_t$.

Usage

```

beverton3(
  start_value,
  alpha,
  beta,
  time_steps = 100L,
  time_lag = 1L,
  pre0_subs = FALSE,
  pre0_value = 0,
  substoch = 0L,
  separate_N = NULL
)

```

Arguments

start_value	A positive number to start the return vector in time 0.
alpha	The alpha parameter in the two-parameter Beverton-Holt function. Must be non-negative.
beta	The beta parameter in the two-parameter Beverton-Holt function. Must be non-negative.
time_steps	The number of time steps to run the projection. Must be a positive integer.
time_lag	A positive integer denoting the number of time steps back for the value of phi in the two-parameter Beverton-Holt function.
pre0_subs	A logical value indicating whether to use a number other than that given in start_value for values of phi lagged from times prior to time 0.
pre0_value	A positive number to use for phi lagged from times prior to time 0. Only used if pre0_subs = TRUE.
substoch	An integer value indicating the kind of substochasticity to use. Values include: 0, no substochasticity enforced (the default); 1, all numbers must be non-negative; and 2, all numbers should be forced to the interval [0, 1].
separate_N	An optional numeric vector with values of N in each time, if phi is to be treated as different from N in the two-parameter model.

Value

A numeric vector of values showing values projected under the two-parameter Beverton-Holt function.

Examples

```

trial_run1 <- beverton3(1, alpha = 0.5, beta = 0.009)
plot(trial_run1)

trial_run2 <- beverton3(1, alpha = 0.5, beta = 0.9)
plot(trial_run2)

trial_run3 <- beverton3(1, alpha = 1, beta = 0.009)
plot(trial_run3)

trial_run4 <- beverton3(1, alpha = 1, beta = 0.9)
plot(trial_run4)

trial_run5 <- beverton3(1, alpha = 5, beta = 0.009)
plot(trial_run5)

trial_run6 <- beverton3(1, alpha = 5, beta = 0.9)
plot(trial_run6)

used_Ns <- c(10, 15, 12, 14, 14, 150, 15, 1, 5, 7, 9, 14, 13, 16, 17, 19,
            25, 26)
trial_run7 <- beverton3(1, alpha = 1, beta = 0.009, separate_N = used_Ns)
plot(trial_run7)

```

cond_diff	<i>Extract Conditional Ahistorical Difference Matrices</i>
-----------	--

Description

Function `cond_diff()` takes a set of historical difference matrices resulting from function `diff_lm()` and decomposes them into ahistorical difference matrices conditional upon stage in time $t-1$.

Usage

```
cond_diff(lDiff, ref = 1L, matchchoice = NULL, err_check = NULL)
```

Arguments

<code>lDiff</code>	An object of class <code>lefkDiff</code> .
<code>ref</code>	Choice of <code>mpm</code> to use as reference. Defaults to 1, which means that the <code>ahstages</code> , <code>hstages</code> , and <code>labels</code> elements for <code>mpm1</code> will be used for all calculations. Only 1 and 2 are possible inputs.
<code>matchchoice</code>	A character denoting whether to use A, U, or F matrices. Defaults to A matrices.
<code>err_check</code>	A logical value denoting whether to include a data frame of element equivalence from the conditional matrices to the original matrices. Used only for debugging purposes. Defaults to FALSE.

Value

A `lefkCondDiff` object, with the following elements:

<code>Mcond</code>	A multi-level list holding the conditional matrices derived from the input <code>lefkDiff</code> object. The top level of the list corresponds to each historical difference matrix in turn, and the lower level corresponds to each stage in time $t-1$, with individual conditional matrices named for the latter.
<code>hstages</code>	A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs.
<code>ahstages</code>	A data frame detailing the characteristics of associated ahistorical stages.
<code>labels</code>	A data frame showing the patch and year of each input full A matrix in order.

Examples

```
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
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, 1.5, 1.5, 3.5, 5)
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
  "3rd yr protocorm", "Seedling", "Dormant adult",
  "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
  "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
  "Extra large adult (>14 shoots)")
cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
  repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
  propstatus = propvector, immstatus = immvector, indataset = indataset,
  binhalfwidth = binvec, comments = comments)

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)

seeds_per_pod <- 5000

cypsupp2_raw <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
  "XSm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep", "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "XSm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "XSm", "XSm", NA, NA),
  givenrate = c(0.03, 0.15, 0.1, 0.1, 0.1, 0.05, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, (0.5 * seeds_per_pod),
    (0.5 * seeds_per_pod)),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)
cypsupp3_raw <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
  "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat", "mat", "SD", "P1"),
  stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
  "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
  stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "P3",
  "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D", "XSm", "Sm",
  "mat", "mat", "mat", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "D", "XSm", "Sm", NA, NA),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.1, 0.1, 0.2, 0.2, 0.2, 0.2, 0.25, NA, NA, NA, NA, NA,
  NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
  NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  type_t12 = c(1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
  stageframe = cypframe_raw, historical = TRUE)

cypmatrix2rp <- rlfko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2"),

```

```

size = c("size3added", "size2added"), supplement = cypsups2_raw,
yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
year = "all", stages = c("stage3", "stage2"),
size = c("size3added", "size2added"), supplement = cypsups2_raw,
yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix3rp <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
size = c("size3added", "size2added", "size1added"), supplement = cypsups3_raw,
yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix3r <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
year = "all", stages = c("stage3", "stage2", "stage1"),
size = c("size3added", "size2added", "size1added"), supplement = cypsups3_raw,
yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix2r_3 <- hist_null(cypmatrix2r)
cypmatrix2r_3 <- delete_lm(cypmatrix2r_3, year = 2004)
diff_r <- diff_lm(cypmatrix3r, cypmatrix2r_3)

cypmatrix2rp_3 <- hist_null(cypmatrix2rp)
cypmatrix2rp_3 <- delete_lm(cypmatrix2rp_3, year = 2004)
diff_rp <- diff_lm(cypmatrix3rp, cypmatrix2rp_3)

condr1 <- cond_diff(diff_r, ref = 1)
condr2 <- cond_diff(diff_r, ref = 2)

condrp1 <- cond_diff(diff_rp, matchchoice = "U", ref = 1)
condrp2 <- cond_diff(diff_rp, matchchoice = "F", ref = 2)

```

cond_hmpm

Extract Conditional Ahistorical Matrices from Historical MPM

Description

Function `cond_hmpm()` takes historical MPMs and decomposes them into ahistorical matrices conditional upon stage in time $t-1$. In effect, the function takes each historical matrix within a `lefkoMat` object, and forms one ahistorical matrix for each stage in time $t-1$.

Usage

```
cond_hmpm(hmpm, matchchoice = NULL, err_check = NULL)
```

Arguments

<code>hmpm</code>	A historical matrix projection model of class <code>lefkoMat</code> .
<code>matchchoice</code>	A character denoting whether to use A, U, or F matrices. Defaults to A matrices.

`err_check` A logical value denoting whether to include a data frame of element equivalence from the conditional matrices to the original matrices. Used only for debugging purposes. Defaults to FALSE.

Value

A `lefkoCondMat` object, with the following elements:

`Mcond` A multi-level list holding the conditional A matrices derived from the input `lefkoMat` object. The top level of the list corresponds to each historical matrix in turn, and the lower level corresponds to each stage in time $t-1$, with individual conditional matrices named for the latter.

`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 input full A matrix in order.

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, 0, 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, 0, 1, 1, 1, 1, 1)
binvec <- c(0, 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)

cypsupp3r <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
  "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat", "mat", "SD", "P1"),
  stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
  "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
  stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "P3", "P3",
  "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D", "XSm", "Sm",
```

```

      "mat", "mat", "mat", NA, NA),
eststage2 = c(NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "D", "XSm", "Sm", NA, NA),
eststage1 = c(NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "XSm", "XSm", "XSm", NA, NA),
givenrate = c(0.1, 0.1, 0.2, 0.2, 0.2, 0.2, 0.2, 0.25, NA, NA, NA, NA, NA, NA,
  NA, NA, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
  NA, 0.5, 0.5),
type = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
type_t12 = c(1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
stageframe = cypframe_raw, historical = TRUE)

cypmatrix3r <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added", "size1added"),
  supplement = cypsupsup3r, yearcol = "year2", patchcol = "patchid",
  indivcol = "individ")

cypcondmats <- cond_hmpm(cypmatrix3r)
summary(cypcondmats)

```

create_IM

Create lefkoMat Object from Given Input Matrices

Description

Function `create_IM()` creates `lefkoMat` objects from supplied matrices and extra information.

Usage

```

create_IM(
  mats,
  stageframe,
  hstages = NA,
  agestages = NA,
  historical = FALSE,
  agebystage = FALSE,
  UFdecomp = TRUE,
  entrystage = 1,
  poporder = 1,
  patchorder = 1,
  yearorder = NA
)

```

Arguments

<code>mats</code>	A list of A matrices.
<code>stageframe</code>	A stageframe describing all stages utilized.
<code>hstages</code>	A data frame outlining the order of historical stages, if matrices provided in <code>mats</code> are historical. Defaults to NA.
<code>agestages</code>	A data frame outlining the order of ahistorical age-stages, if age-by-stage matrices are provided.
<code>historical</code>	A logical value indicating whether input matrices are historical or not. Defaults to FALSE.
<code>agebystage</code>	A logical value indicating whether input matrices are ahistorical age-by-stage matrices. If TRUE, then object <code>agestages</code> is required. Defaults to FALSE.
<code>UFdecomp</code>	A logical value indicating whether U and F matrices should be inferred. Defaults to TRUE.
<code>entrystage</code>	The stage or stages produced by reproductive individuals. Used to determine which transitions are reproductive for U-F decomposition. Defaults to 1, which corresponds to the first stage in the stageframe.
<code>poporder</code>	The order of populations in the list supplied in object <code>mats</code> . Defaults to 1.
<code>patchorder</code>	The order of patches in the list supplied in object <code>mats</code> . Defaults to 1.
<code>yearorder</code>	The order of monitoring occasions in the list supplied in object <code>mats</code> . Defaults to NA, which leads to each matrix within each population-patch combination being a different monitoring occasion.

Value

A `lefkMat` object incorporating the matrices input in object `mats` as object `A`, their U and F decompositions in objects `U` and `F` (if requested), the provided stageframe as object `ahstages`, the order of historical stages as object `hstages` (if `historical = TRUE`), the order of matrices as object `labels`, and a short quality control section used by the `summary.lefkMat()` function.

Notes

U and F decomposition assumes that elements holding fecundity values are to be interpreted solely as fecundity rates. Users wishing to split these elements between fecundity and survival should do so manually after running this function.

Age-by-stage MPMs require an `agestages` data frame outlining the order of age-stages. This data frame has 3 variables: `stage_id`, which is the number of the stage as labelled by the equivalently named variable in the stageframe; `stage`, which is the official name of the stage as given in the equivalently named variable in the stageframe; and `age`, which of course gives the age associated with the stage at that time. The number of rows must be equal to the number of rows and columns of each entered matrix.

See Also

[add_IM\(\)](#)
[delete_IM\(\)](#)
[subset_IM\(\)](#)

Examples

```

# These matrices are of 9 populations of the plant species Anthyllis
# vulneraria, and were originally published in Davison et al. (2010) Journal
# of Ecology 98:255-267 (doi: 10.1111/j.1365-2745.2009.01611.x).

sizevector <- c(1, 1, 2, 3) # These sizes are not from the original paper
stagevector <- c("Sd1", "Veg", "SmFlo", "LFlo")
repvector <- c(0, 0, 1, 1)
obsvector <- c(1, 1, 1, 1)
matvector <- c(0, 1, 1, 1)
immvector <- c(1, 0, 0, 0)
propvector <- c(0, 0, 0, 0)
indataset <- c(1, 1, 1, 1)
binvec <- c(0.5, 0.5, 0.5, 0.5)

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

# POPN C 2003-2004
XC3 <- matrix(c(0, 0, 1.74, 1.74,
0.2083333333, 0, 0, 0.057142857,
0.0416666667, 0.076923077, 0, 0,
0.0833333333, 0.076923077, 0.066666667, 0.028571429), 4, 4, byrow = TRUE)

# 2004-2005
XC4 <- matrix(c(0, 0, 0.3, 0.6,
0.32183908, 0.142857143, 0, 0,
0.16091954, 0.285714286, 0, 0,
0.252873563, 0.285714286, 0.5, 0.6), 4, 4, byrow = TRUE)

# 2005-2006
XC5 <- matrix(c(0, 0, 0.50625, 0.675,
0, 0, 0, 0.035714286,
0.1, 0.068965517, 0.0625, 0.107142857,
0.3, 0.137931034, 0, 0.071428571), 4, 4, byrow = TRUE)

# POPN E 2003-2004
XE3 <- matrix(c(0, 0, 2.44, 6.569230769,
0.196428571, 0, 0, 0,
0.125, 0.5, 0, 0,
0.160714286, 0.5, 0.133333333, 0.076923077), 4, 4, byrow = TRUE)

XE4 <- matrix(c(0, 0, 0.45, 0.646153846,
0.06557377, 0.090909091, 0.125, 0,
0.032786885, 0, 0.125, 0.076923077,
0.049180328, 0, 0.125, 0.230769231), 4, 4, byrow = TRUE)

XE5 <- matrix(c(0, 0, 2.85, 3.99,
0.0833333333, 0, 0, 0,
0, 0, 0, 0,
0, 0, 0, 0,
0, 0, 0, 0), 4, 4, byrow = TRUE)

```

```
0.416666667, 0.1, 0, 0.1), 4, 4, byrow = TRUE)

# POPN F 2003-2004
XF3 <- matrix(c(0, 0, 1.815, 7.058333333,
0.075949367, 0, 0.05, 0.083333333,
0.139240506, 0, 0, 0.25,
0.075949367, 0, 0, 0.083333333), 4, 4, byrow = TRUE)

XF4 <- matrix(c(0, 0, 1.233333333, 7.4,
0.223880597, 0, 0.111111111, 0.142857143,
0.134328358, 0.272727273, 0.166666667, 0.142857143,
0.119402985, 0.363636364, 0.055555556, 0.142857143), 4, 4, byrow = TRUE)

XF5 <- matrix(c(0, 0, 1.06, 3.372727273,
0.073170732, 0.025, 0.033333333, 0,
0.036585366, 0.15, 0.1, 0.136363636,
0.06097561, 0.225, 0.166666667, 0.272727273), 4, 4, byrow = TRUE)

# POPN G 2003-2004
XG3 <- matrix(c(0, 0, 0.245454545, 2.1,
0, 0, 0.045454545, 0,
0.125, 0, 0.090909091, 0,
0.125, 0, 0.090909091, 0.333333333), 4, 4, byrow = TRUE)

XG4 <- matrix(c(0, 0, 1.1, 1.54,
0.111111111, 0, 0, 0,
0, 0, 0, 0,
0.111111111, 0, 0, 0), 4, 4, byrow = TRUE)

XG5 <- matrix(c(0, 0, 0, 1.5,
0, 0, 0, 0,
0.090909091, 0, 0, 0,
0.545454545, 0.5, 0, 0.5), 4, 4, byrow = TRUE)

# POPN L 2003-2004
XL3 <- matrix(c(0, 0, 1.785365854, 1.856521739,
0.128571429, 0, 0, 0.010869565,
0.028571429, 0, 0, 0,
0.014285714, 0, 0, 0.02173913), 4, 4, byrow = TRUE)

XL4 <- matrix(c(0, 0, 14.25, 16.625,
0.131443299, 0.057142857, 0, 0.25,
0.144329897, 0, 0, 0,
0.092783505, 0.2, 0, 0.25), 4, 4, byrow = TRUE)

XL5 <- matrix(c(0, 0, 0.594642857, 1.765909091,
0, 0, 0.017857143, 0,
0.021052632, 0.018518519, 0.035714286, 0.045454545,
0.021052632, 0.018518519, 0.035714286, 0.068181818), 4, 4, byrow = TRUE)

# POPN O 2003-2004
XO3 <- matrix(c(0, 0, 11.5, 2.775862069,
0.6, 0.285714286, 0.333333333, 0.24137931,
```

```
0.04, 0.142857143, 0, 0,
0.16, 0.285714286, 0, 0.172413793), 4, 4, byrow = TRUE)

X04 <- matrix(c(0, 0, 3.78, 1.225,
0.28358209, 0.171052632, 0, 0.166666667,
0.084577114, 0.026315789, 0, 0.055555556,
0.139303483, 0.447368421, 0, 0.305555556), 4, 4, byrow = TRUE)

X05 <- matrix(c(0, 0, 1.542857143, 1.035616438,
0.126984127, 0.105263158, 0.047619048, 0.054794521,
0.095238095, 0.157894737, 0.19047619, 0.082191781,
0.111111111, 0.223684211, 0, 0.356164384), 4, 4, byrow = TRUE)

# POPN Q 2003-2004
XQ3 <- matrix(c(0, 0, 0.15, 0.175,
0, 0, 0, 0,
0, 0, 0, 0,
1, 0, 0, 0), 4, 4, byrow = TRUE)

XQ4 <- matrix(c(0, 0, 0, 0.25,
0, 0, 0, 0,
0, 0, 0, 0,
1, 0.666666667, 0, 1), 4, 4, byrow = TRUE)

XQ5 <- matrix(c(0, 0, 0, 1.428571429,
0, 0, 0, 0.142857143,
0.25, 0, 0, 0,
0.25, 0, 0, 0.571428571), 4, 4, byrow = TRUE)

# POPN R 2003-2004
XR3 <- matrix(c(0, 0, 0.7, 0.6125,
0.25, 0, 0, 0.125,
0, 0, 0, 0,
0.25, 0.166666667, 0, 0.25), 4, 4, byrow = TRUE)

XR4 <- matrix(c(0, 0, 0, 0.6,
0.285714286, 0, 0, 0,
0.285714286, 0.333333333, 0, 0,
0.285714286, 0.333333333, 0, 1), 4, 4, byrow = TRUE)

XR5 <- matrix(c(0, 0, 0.7, 0.6125,
0, 0, 0, 0,
0, 0, 0, 0,
0.333333333, 0, 0.333333333, 0.625), 4, 4, byrow = TRUE)

# POPN S 2003-2004
XS3 <- matrix(c(0, 0, 2.1, 0.816666667,
0.166666667, 0, 0, 0,
0, 0, 0, 0,
0, 0, 0, 0.166666667), 4, 4, byrow = TRUE)

XS4 <- matrix(c(0, 0, 0, 7,
0.333333333, 0.5, 0, 0,
```

```

0, 0, 0, 0,
0.333333333, 0, 0, 1), 4, 4, byrow = TRUE)

XS5 <- matrix(c(0, 0, 0, 1.4,
0, 0, 0, 0,
0, 0, 0, 0.2,
0.111111111, 0.75, 0, 0.2), 4, 4, byrow = TRUE)

mats_list <- list(XC3, XC4, XC5, XE3, XE4, XE5, XF3, XF4, XF5, XG3, XG4, XG5,
  XL3, XL4, XL5, X03, X04, X05, XQ3, XQ4, XQ5, XR3, XR4, XR5, XS3, XS4, XS5)

yr_ord <- c(1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1,
  2, 3, 1, 2, 3)

pch_ord <- c(1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6, 7, 7, 7,
  8, 8, 8, 9, 9, 9)

anth_lefkoMat <- create_IM(mats_list, anthframe, hstages = NA, historical = FALSE,
  poporder = 1, patchorder = pch_ord, yearorder = yr_ord)

anth_lefkoMat

# A theoretical example showcasing historical matrices

sizevector <- c(1, 2, 3) # These sizes are not from the original paper
stagevector <- c("Sdl", "Veg", "Flo")
repvector <- c(0, 0, 1)
obsvector <- c(1, 1, 1)
matvector <- c(0, 1, 1)
immvector <- c(1, 0, 0)
propvector <- c(1, 0, 0)
indataset <- c(1, 1, 1)
binvec <- c(0.5, 0.5, 0.5)

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

A1 <- matrix(c(0.10, 0, 0, 0.12, 0, 0, 0.15, 0, 0,
  0.15, 0, 0, 0.17, 0, 0, 0.20, 0, 0,
  0.20, 0, 0, 0.22, 0, 0, 0.25, 0, 0,
  0, 0.20, 0, 0, 0.22, 0, 0, 0.25, 0,
  0, 0.25, 0, 0, 0.27, 0, 0, 0.30, 0,
  0, 0.30, 0, 0, 0.32, 0, 0, 0.35, 0,
  0, 0, 2.00, 0, 0, 3.00, 0, 0, 4.00,
  0, 0, 0.35, 0, 0, 0.37, 0, 0, 0.40,
  0, 0, 0.40, 0, 0, 0.42, 0, 0, 0.45), 9, 9, byrow = TRUE)

A2 <- matrix(c(0.10, 0, 0, 0.12, 0, 0, 0.15, 0, 0,
  0.15, 0, 0, 0.17, 0, 0, 0.20, 0, 0,
  0.20, 0, 0, 0.22, 0, 0, 0.25, 0, 0,
  0, 0.20, 0, 0, 0.22, 0, 0, 0.25, 0,

```

```

0, 0.25, 0, 0, 0.27, 0, 0, 0.30, 0,
0, 0.30, 0, 0, 0.32, 0, 0, 0.35, 0,
0, 0, 5.00, 0, 0, 6.00, 0, 0, 7.00,
0, 0, 0.35, 0, 0, 0.37, 0, 0, 0.40,
0, 0, 0.40, 0, 0, 0.42, 0, 0, 0.45), 9, 9, byrow = TRUE)

A3 <- matrix(c(0.10, 0, 0, 0.12, 0, 0, 0.15, 0, 0,
0.15, 0, 0, 0.17, 0, 0, 0.20, 0, 0,
0.20, 0, 0, 0.22, 0, 0, 0.25, 0, 0,
0, 0.20, 0, 0, 0.22, 0, 0, 0.25, 0,
0, 0.25, 0, 0, 0.27, 0, 0, 0.30, 0,
0, 0.30, 0, 0, 0.32, 0, 0, 0.35, 0,
0, 0, 8.00, 0, 0, 9.00, 0, 0, 10.00,
0, 0, 0.35, 0, 0, 0.37, 0, 0, 0.40,
0, 0, 0.40, 0, 0, 0.42, 0, 0, 0.45), 9, 9, byrow = TRUE)

B1 <- matrix(c(0.10, 0, 0, 0.12, 0, 0, 0.15, 0, 0,
0.15, 0, 0, 0.17, 0, 0, 0.20, 0, 0,
0.20, 0, 0, 0.22, 0, 0, 0.25, 0, 0,
0, 0.20, 0, 0, 0.22, 0, 0, 0.25, 0,
0, 0.25, 0, 0, 0.27, 0, 0, 0.30, 0,
0, 0.30, 0, 0, 0.32, 0, 0, 0.35, 0,
0, 0, 11.00, 0, 0, 12.00, 0, 0, 13.00,
0, 0, 0.35, 0, 0, 0.37, 0, 0, 0.40,
0, 0, 0.40, 0, 0, 0.42, 0, 0, 0.45), 9, 9, byrow = TRUE)

B2 <- matrix(c(0.10, 0, 0, 0.12, 0, 0, 0.15, 0, 0,
0.15, 0, 0, 0.17, 0, 0, 0.20, 0, 0,
0.20, 0, 0, 0.22, 0, 0, 0.25, 0, 0,
0, 0.20, 0, 0, 0.22, 0, 0, 0.25, 0,
0, 0.25, 0, 0, 0.27, 0, 0, 0.30, 0,
0, 0.30, 0, 0, 0.32, 0, 0, 0.35, 0,
0, 0, 14.00, 0, 0, 15.00, 0, 0, 16.00,
0, 0, 0.35, 0, 0, 0.37, 0, 0, 0.40,
0, 0, 0.40, 0, 0, 0.42, 0, 0, 0.45), 9, 9, byrow = TRUE)

B3 <- matrix(c(0.10, 0, 0, 0.12, 0, 0, 0.15, 0, 0,
0.15, 0, 0, 0.17, 0, 0, 0.20, 0, 0,
0.20, 0, 0, 0.22, 0, 0, 0.25, 0, 0,
0, 0.20, 0, 0, 0.22, 0, 0, 0.25, 0,
0, 0.25, 0, 0, 0.27, 0, 0, 0.30, 0,
0, 0.30, 0, 0, 0.32, 0, 0, 0.35, 0,
0, 0, 17.00, 0, 0, 18.00, 0, 0, 19.00,
0, 0, 0.35, 0, 0, 0.37, 0, 0, 0.40,
0, 0, 0.40, 0, 0, 0.42, 0, 0, 0.45), 9, 9, byrow = TRUE)

histmats <- list(A1, A2, A3, B1, B2, B3)
stageframe <- exframe
pch_ord <- c("A", "A", "A", "B", "B", "B")
yr_ord <- c(1, 2, 3, 1, 2, 3)

hist_trial <- create_LM(histmats, exframe, historical = TRUE, UFdecomp = TRUE,
  entrystage = 1, patchorder = pch_ord, yearorder = yr_ord)

```

hist_trial

create_pm	<i>Creates a Skeleton Paramnames Object for Use in Function-based Modeling</i>
-----------	--

Description

Creates a simple skeleton paramnames object that can be entered as input in functions [flefko2\(\)](#), [flefko3\(\)](#), and [aflefko2\(\)](#).

Usage

```
create_pm(name_terms = FALSE)
```

Arguments

name_terms	A logical value indicating whether to start each variable name as none if FALSE, or as the default modelparams name if TRUE. Defaults to FALSE.
------------	---

Value

A three column data frame, of which the first describes the parameters in reasonably plain English, the second gives the name of the parameter within the MPM generating functions, and the third is to be edited with the names of the variables as they appear in the models.

Notes

The third column in the resulting object should be edited with the names only of those variables actually used in vital rate modeling. This paramnames object should apply to all models used in a single MPM building exercise. So, for example, if the models used include random terms, then they should all have the same random terms. Fixed terms can vary, however.

Examples

```
our_pm <- create_pm()
our_pm
```

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 29 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 numeric variable giving a unique number to each individual.

patch A variable referring to patch within the population.

X An X coordinate for the plant within the population.

Y A Y coordinate for the plant within the population.

sensor 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

```
# Cyripedium example using blocksize
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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
```



```

type =c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix2r$A[[intersect(which(cypmatrix2r$labels$patch == "A"),
  which(cypmatrix2r$labels$year2 == 2004))]]

lambda3(cypmatrix2r)

# Cyripedium example using partial repeat patterns with blocksize and part
# explicit variable name cast
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", indivcol = "plantid", blocksize = 4,
  sizeacol = "Inf2.04", sizebcol = "Inf.04", sizeccol = "Veg.04",
  repstracol = c("Inf.04", "Inf.05", "Inf.06", "Inf.07", "Inf.08", "Inf.09"),
  repstrbcol = c("Inf2.04", "Inf2.05", "Inf2.06", "Inf2.07", "Inf2.08", "Inf2.09"),
  fecacol = "Pod.04", stageassign = cypframe_raw, stagesize = "sizeadded",
  NAas0 = TRUE, NRasRep = TRUE)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type =c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,

```

```

year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
size = c("size3added", "size2added"), supplement = cypsups2r,
yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix2r$A[[intersect(which(cypmatrix2r$labels$patch == "A"),
  which(cypmatrix2r$labels$year2 == 2004))]]

lambda3(cypmatrix2r)

```

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, 322 rows, and 14 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 occasion t and `.3` to identify occasion $t+1$.

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

patch A variable referring to patch within the population.

X An X coordinate for the plant within the population.

Y A Y coordinate for the plant within the population.

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

year2 Year in occasion t .

Inf2.2 Number of double inflorescences in occasion t .

Inf.2 Number of inflorescences in occasion t .

Veg.2 Number of stems without inflorescences in occasion t .

Pod.2 Number of fruits in occasion t .

Inf2.3 Number of double inflorescences in occasion $t+1$.

Inf.3 Number of inflorescences in occasion $t+1$.

Veg.3 Number of stems without inflorescences in occasion $t+1$.

Pod.3 Number of fruits in occasion $t+1$.

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, 0, 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",
  fecaa2col = "Pod.2", fecaa3col = "Pod.3", repstrrel = 2,
  stageassign = cypframe_raw, stagesize = "sizeadded", censorcol = "censor",
  censor = FALSE, NAas0 = TRUE, NRasRep = TRUE, reduce = TRUE)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rleko2(data = cypraw_v2, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")
```

```

cypmatrix2r$A[[intersect(which(cypmatrix2r$labels$patch == "A"),
  which(cypmatrix2r$labels$year2 == 2004))]]

lambda3(cypmatrix2r)

```

delete_IM

Delete Matrices from lefkoMat Object

Description

Function `delete_IM()` deletes matrices from `lefkoMat` objects.

Usage

```
delete_IM(lM, mat_num = NA, pop = NA, patch = NA, year = NA)
```

Arguments

<code>lM</code>	The <code>lefkoMat</code> object to delete matrices from.
<code>mat_num</code>	Either a single integer corresponding to the matrix to remove within the <code>labels</code> element of <code>lM</code> , or a vector of such integers.
<code>pop</code>	The population designation for matrices to remove. Only used if <code>mat_num</code> is not given.
<code>patch</code>	The patch designation for matrices to remove. Only used if <code>mat_num</code> is not given.
<code>year</code>	The time t designation for matrices to remove. Only used if <code>mat_num</code> is not given.

Value

A `lefkoMat` object in which the matrices specified in `lM` have been removed.

Notes

If `mat_num` is not provided, then at least one of `pop`, `patch`, or `year` must be provided. If at least two of `pop`, `patch`, and `year` are provided, then function `delete_IM()` will identify matrices to remove as the intersection of provided inputs.

See Also

[create_IM\(\)](#)
[add_IM\(\)](#)
[subset_IM\(\)](#)

Examples

```

# These matrices are of 9 populations of the plant species Anthyllis
# vulneraria, and were originally published in Davison et al. (2010) Journal
# of Ecology 98:255-267 (doi: 10.1111/j.1365-2745.2009.01611.x).

sizevector <- c(1, 1, 2, 3) # These sizes are not from the original paper
stagevector <- c("Sd1", "Veg", "SmFlo", "LFlo")
repvector <- c(0, 0, 1, 1)
obsvector <- c(1, 1, 1, 1)
matvector <- c(0, 1, 1, 1)
immvector <- c(1, 0, 0, 0)
propvector <- c(0, 0, 0, 0)
indataset <- c(1, 1, 1, 1)
binvec <- c(0.5, 0.5, 0.5, 0.5)

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

# POPN C 2003-2004
XC3 <- matrix(c(0, 0, 1.74, 1.74,
0.2083333333, 0, 0, 0.057142857,
0.0416666667, 0.076923077, 0, 0,
0.0833333333, 0.076923077, 0.066666667, 0.028571429), 4, 4, byrow = TRUE)

# 2004-2005
XC4 <- matrix(c(0, 0, 0.3, 0.6,
0.32183908, 0.142857143, 0, 0,
0.16091954, 0.285714286, 0, 0,
0.252873563, 0.285714286, 0.5, 0.6), 4, 4, byrow = TRUE)

# 2005-2006
XC5 <- matrix(c(0, 0, 0.50625, 0.675,
0, 0, 0, 0.035714286,
0.1, 0.068965517, 0.0625, 0.107142857,
0.3, 0.137931034, 0, 0.071428571), 4, 4, byrow = TRUE)

# POPN E 2003-2004
XE3 <- matrix(c(0, 0, 2.44, 6.569230769,
0.196428571, 0, 0, 0,
0.125, 0.5, 0, 0,
0.160714286, 0.5, 0.133333333, 0.076923077), 4, 4, byrow = TRUE)

XE4 <- matrix(c(0, 0, 0.45, 0.646153846,
0.06557377, 0.090909091, 0.125, 0,
0.032786885, 0, 0.125, 0.076923077,
0.049180328, 0, 0.125, 0.230769231), 4, 4, byrow = TRUE)

XE5 <- matrix(c(0, 0, 2.85, 3.99,
0.0833333333, 0, 0, 0,
0, 0, 0, 0,
0, 0, 0, 0,
0, 0, 0, 0), 4, 4, byrow = TRUE)

```

```
0.416666667, 0.1, 0, 0.1), 4, 4, byrow = TRUE)

# POPN F 2003-2004
XF3 <- matrix(c(0, 0, 1.815, 7.058333333,
0.075949367, 0, 0.05, 0.083333333,
0.139240506, 0, 0, 0.25,
0.075949367, 0, 0, 0.083333333), 4, 4, byrow = TRUE)

XF4 <- matrix(c(0, 0, 1.233333333, 7.4,
0.223880597, 0, 0.111111111, 0.142857143,
0.134328358, 0.272727273, 0.166666667, 0.142857143,
0.119402985, 0.363636364, 0.055555556, 0.142857143), 4, 4, byrow = TRUE)

XF5 <- matrix(c(0, 0, 1.06, 3.372727273,
0.073170732, 0.025, 0.033333333, 0,
0.036585366, 0.15, 0.1, 0.136363636,
0.06097561, 0.225, 0.166666667, 0.272727273), 4, 4, byrow = TRUE)

# POPN G 2003-2004
XG3 <- matrix(c(0, 0, 0.245454545, 2.1,
0, 0, 0.045454545, 0,
0.125, 0, 0.090909091, 0,
0.125, 0, 0.090909091, 0.333333333), 4, 4, byrow = TRUE)

XG4 <- matrix(c(0, 0, 1.1, 1.54,
0.111111111, 0, 0, 0,
0, 0, 0, 0,
0.111111111, 0, 0, 0), 4, 4, byrow = TRUE)

XG5 <- matrix(c(0, 0, 0, 1.5,
0, 0, 0, 0,
0.090909091, 0, 0, 0,
0.545454545, 0.5, 0, 0.5), 4, 4, byrow = TRUE)

# POPN L 2003-2004
XL3 <- matrix(c(0, 0, 1.785365854, 1.856521739,
0.128571429, 0, 0, 0.010869565,
0.028571429, 0, 0, 0,
0.014285714, 0, 0, 0.02173913), 4, 4, byrow = TRUE)

XL4 <- matrix(c(0, 0, 14.25, 16.625,
0.131443299, 0.057142857, 0, 0.25,
0.144329897, 0, 0, 0,
0.092783505, 0.2, 0, 0.25), 4, 4, byrow = TRUE)

XL5 <- matrix(c(0, 0, 0.594642857, 1.765909091,
0, 0, 0.017857143, 0,
0.021052632, 0.018518519, 0.035714286, 0.045454545,
0.021052632, 0.018518519, 0.035714286, 0.068181818), 4, 4, byrow = TRUE)

# POPN O 2003-2004
XO3 <- matrix(c(0, 0, 11.5, 2.775862069,
0.6, 0.285714286, 0.333333333, 0.24137931,
```

```
0.04, 0.142857143, 0, 0,
0.16, 0.285714286, 0, 0.172413793), 4, 4, byrow = TRUE)

X04 <- matrix(c(0, 0, 3.78, 1.225,
0.28358209, 0.171052632, 0, 0.166666667,
0.084577114, 0.026315789, 0, 0.055555556,
0.139303483, 0.447368421, 0, 0.305555556), 4, 4, byrow = TRUE)

X05 <- matrix(c(0, 0, 1.542857143, 1.035616438,
0.126984127, 0.105263158, 0.047619048, 0.054794521,
0.095238095, 0.157894737, 0.19047619, 0.082191781,
0.111111111, 0.223684211, 0, 0.356164384), 4, 4, byrow = TRUE)

# POPN Q 2003-2004
XQ3 <- matrix(c(0, 0, 0.15, 0.175,
0, 0, 0, 0,
0, 0, 0, 0,
1, 0, 0, 0), 4, 4, byrow = TRUE)

XQ4 <- matrix(c(0, 0, 0, 0.25,
0, 0, 0, 0,
0, 0, 0, 0,
1, 0.666666667, 0, 1), 4, 4, byrow = TRUE)

XQ5 <- matrix(c(0, 0, 0, 1.428571429,
0, 0, 0, 0.142857143,
0.25, 0, 0, 0,
0.25, 0, 0, 0.571428571), 4, 4, byrow = TRUE)

# POPN R 2003-2004
XR3 <- matrix(c(0, 0, 0.7, 0.6125,
0.25, 0, 0, 0.125,
0, 0, 0, 0,
0.25, 0.166666667, 0, 0.25), 4, 4, byrow = TRUE)

XR4 <- matrix(c(0, 0, 0, 0.6,
0.285714286, 0, 0, 0,
0.285714286, 0.333333333, 0, 0,
0.285714286, 0.333333333, 0, 1), 4, 4, byrow = TRUE)

XR5 <- matrix(c(0, 0, 0.7, 0.6125,
0, 0, 0, 0,
0, 0, 0, 0,
0.333333333, 0, 0.333333333, 0.625), 4, 4, byrow = TRUE)

# POPN S 2003-2004
XS3 <- matrix(c(0, 0, 2.1, 0.816666667,
0.166666667, 0, 0, 0,
0, 0, 0, 0,
0, 0, 0, 0.166666667), 4, 4, byrow = TRUE)

XS4 <- matrix(c(0, 0, 0, 7,
0.333333333, 0.5, 0, 0,
```

```

0, 0, 0, 0,
0.3333333333, 0, 0, 1), 4, 4, byrow = TRUE)

XS5 <- matrix(c(0, 0, 0, 1.4,
0, 0, 0, 0,
0, 0, 0, 0.2,
0.1111111111, 0.75, 0, 0.2), 4, 4, byrow = TRUE)

mats_list <- list(XC3, XC4, XC5, XE3, XE4, XE5, XF3, XF4, XF5, XG3, XG4, XG5,
  XL3, XL4, XL5, X03, X04, X05, XQ3, XQ4, XQ5, XR3, XR4, XR5, XS3, XS4, XS5)

yr_ord <- c(1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1,
  2, 3, 1, 2, 3)

pch_ord <- c(1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6, 7, 7, 7,
  8, 8, 8, 9, 9, 9)

anth_lefkoMat <- create_LM(mats_list, anthframe, hstages = NA, historical = FALSE,
  poporder = 1, patchorder = pch_ord, yearorder = yr_ord)

smaller_anth_LM <- delete_LM(anth_lefkoMat, patch = 3)
smaller_anth_LM

```

density_input

Create a Data Frame of Density Dependence Relationships in Matrix Elements

Description

Function `density_input()` provides all necessary data to incorporate density dependence into a `lefkoMat` object, a list of matrices, or a single matrix. Four forms of density dependence are allowed, including the Ricker function, the Beverton-Holt function, the Usher function, and the logistic function. In each case, density must have an effect with at least a one time-step delay (see Notes). The resulting data frame provides a guide for other `lefko3` functions to modify matrix elements by density.

Usage

```

density_input(
  mpm,
  stage3,
  stage2,
  stage1 = NA,
  age2 = NA,
  style = 1,
  time_delay = 1,
  alpha = NA,
  beta = NA,

```



```

    type = NA,
    type_t12 = NA
  )

```

Arguments

mpm	The <code>lefkoMat</code> object that will be subject to density dependent projection.
stage3	A vector showing the name or number of the stage in occasion $t+1$ in the transitions to be affected by density. Abbreviations for groups of stages are also usable (see Notes).
stage2	A vector showing the name or number of the stage in occasion t in the transition to be affected by density. Abbreviations for groups of stages are also usable (see Notes).
stage1	A vector showing the name or number of the stage in occasion $t-1$ in the transition to be affected by density. Only needed if a historical MPM is used. Abbreviations for groups of stages are also usable (see Notes).
age2	A vector showing the age of the stage in occasion t in the transition to be affected by density. Only needed if an age-by-stage MPM is used.
style	A vector coding for the style of density dependence on each transition subject to density dependence. Options include 1, <code>ricker</code> , <code>ric</code> , or <code>r</code> for the Ricker function; 2, <code>beverton</code> , <code>bev</code> , and <code>b</code> for the Beverton-Holt function; 3, <code>usher</code> , <code>ush</code> , and <code>u</code> for the Usher function; and 4, <code>logistic</code> , <code>log</code> , and <code>l</code> for the logistic function. If only a single code is provided, then all noted transitions are assumed to be subject to this style of density dependence. Defaults to <code>ricker</code> .
time_delay	An integer vector indicating the number of occasions back on which density dependence operates. Defaults to 1, and may not equal any integer less than 1. If a single number is input, then all noted transitions are assumed to be subject to this time delay.
alpha	A vector indicating the numeric values to use as the alpha term in the two parameter Ricker, Beverton-Holt, or Usher function, or the value of the carrying capacity K to use in the logistic equation (see Notes section for more on this term). If a single number is provided, then all noted transitions are assumed to be subject to this value of alpha.
beta	A vector indicating the numeric values to use as the beta term in the two parameter Ricker, Beverton-Holt, or Usher function. Used to indicate whether to use K as a hard limit in the logistic equation (see section Notes below). If a single number is provided, then all noted transitions are assumed to be subject to this value of beta.
type	A vector denoting the kind of transition between occasions t and $t+1$ to be replaced. This should be entered as 1, <code>S</code> , or <code>s</code> for the replacement of a survival transition; or 2, <code>F</code> , or <code>f</code> for the replacement of a fecundity transition. If empty or not provided, then defaults to 1 for survival transition.
type_t12	An optional vector denoting the kind of transition between occasions $t-1$ and t . Only necessary if a historical MPM in deVries format is desired. This should be entered as 1, <code>S</code> , or <code>s</code> for a survival transition; or 2, <code>F</code> , or <code>f</code> for a fecundity transitions. Defaults to 1 for survival transition, with impacts only on the construction of deVries-format hMPMs.

Value

A data frame of class `lefkoDens`. This object can be used as input in function `projection3()`.

Variables in this object include the following:

<code>stage3</code>	Stage at occasion $t+1$ in the transition to be replaced.
<code>stage2</code>	Stage at occasion t in the transition to be replaced.
<code>stage1</code>	Stage at occasion $t-1$ in the transition to be replaced, if applicable.
<code>age2</code>	Age at occasion t in the transition to be replaced, if applicable.
<code>style</code>	Style of density dependence, coded as 1, 2, 3, or 4 for the Ricker, Beverton-Holt, Usher, or logistic function, respectively.
<code>time_delay</code>	The time delay on density dependence, in time steps.
<code>alpha</code>	The value of alpha in the Ricker, Beverton-Holt, or Usher function, or the value of carrying capacity, K , in the logistic function.
<code>beta</code>	The value of beta in the Ricker, Beverton-Holt, or Usher function.
<code>type</code>	Designates whether the transition from occasion t to occasion $t+1$ is a survival transition probability (1), or a fecundity rate (2).
<code>type_t12</code>	Designates whether the transition from occasion $t-1$ to occasion t is a survival transition probability (1), a fecundity rate (2).

Notes

This function provides inputs when density dependence is operationalized directly on matrix elements. It can be used in both `projection3()` and `f_projection3()`. Users wishing to modify vital rate functions by density dependence functions for use in function-based projections with function `f_projection3()` should use function `density_vr()` to provide the correct inputs.

The parameters `alpha` and `beta` are applied according to the two-parameter Ricker function, the two-parameter Beverton-Holt function, the two-parameter Usher function, or the one-parameter logistic function. Although the default is that a 1 time step delay is assumed, greater time delays can be set through the `time_delay` option.

Entries in `stage3`, `stage2`, and `stage1` can include abbreviations for groups of stages. Use `rep` if all reproductive stages are to be used, `nrep` if all mature but non-reproductive stages are to be used, `mat` if all mature stages are to be used, `immat` if all immature stages are to be used, `prop` if all propagule stages are to be used, `npr` if all non-propagule stages are to be used, `obs` if all observable stages are to be used, `nobs` if all unobservable stages are to be used, and leave empty or use `all` if all stages in `stageframe` are to be used.

When using the logistic function, it is possible that the time delay used in density dependent simulations will cause matrix elements to become negative. To prevent this behavior, set the associated `beta` term to `1.0`. Doing so will set K as the hard limit in the logistic equation, essentially setting a minimum limit at 0 for all matrix elements modified.

See Also

[start_input\(\)](#)

[projection3\(\)](#)

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfeko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

ehrlen3mean <- lmean(ehrlen3)

e3d <- density_input(ehrlen3mean, stage3 = c("Sd", "Sd1"),
  stage2 = c("rep", "rep"), stage1 = c("all", "all"), style = 1,
  time_delay = 1, alpha = 1, beta = 0, type = c(2, 2), type_t12 = c(1, 1))

lathproj <- projection3(ehrlen3, nreps = 5, stochastic = TRUE, substoch = 2,
  density = e3d)

```

density_vr	<i>Create a Data Frame of Density Dependence Relationships in Vital Rates</i>
------------	---

Description

Function `density_vr()` provides all necessary data to incorporate density dependence into the vital rate functions used to create matrices in function-based projections using function `f_projection3()`. Four forms of density dependence are allowed, including the Ricker function, the Beverton-Holt function, the Usher function, and the logistic function. In each case, density must have an effect with at least a one time-step delay (see Notes).

Usage

```
density_vr(
  density_yn = c(FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
                 FALSE, FALSE, FALSE, FALSE),
  style = c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),
  time_delay = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
  alpha = c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),
  beta = c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
)
```

Arguments

density_yn	A 14 element logical vector denoting whether each vital rate is subject to density dependence. The order of vital rates is: survival probability, observation probability, primary size transition, secondary size transition, tertiary size transition, reproductive status probability, fecundity rate, juvenile survival probability, juvenile observation probability, juvenile primary size transition, juvenile secondary size transition, juvenile tertiary size transition, juvenile reproductive status probability, and juvenile maturity status probability. Defaults to a vector of 14 FALSE values.
style	A 14 element vector coding for the style of density dependence on each vital rate. Options include 0: no density dependence, 1, ricker, ric, or r for the Ricker function; 2, beverton, bev, and b for the Beverton-Holt function; 3, usher, ush, and u for the Usher function; and 4, logistic, log, and l for the logistic function. Defaults to 14 values of 1.
time_delay	A 14 element vector indicating the number of occasions back on which density dependence operates. Defaults to 14 values of 1, and may not include any number less than 1.
alpha	A 14 element vector indicating the numeric values to use as the alpha term in the two parameter Ricker, Beverton-Holt, or Usher function, or the value of the carrying capacity K to use in the logistic equation (see Notes for more on this term). Defaults to 14 values of 0.

beta A 14 element vector indicating the numeric values to use as the beta term in the two parameter Ricker, Beverton-Holt, or Usher function. Used to indicate whether to use K as a hard limit in the logistic equation (see Notes below). Defaults to 14 values of \emptyset .

Value

A data frame of class `lefkodensVR` with 14 rows, one for each vital rate in the order of: survival probability, observation probability, primary size transition, secondary size transition, tertiary size transition, reproductive status probability, fecundity rate, juvenile survival probability, juvenile observation probability, juvenile primary size transition, juvenile secondary size transition, juvenile tertiary size transition, juvenile reproductive status probability, and juvenile maturity status probability. This object can be used as input in function `f_projection3()`.

Variables in this object include the following:

<code>vital_rate</code>	The vital rate to be modified.
<code>density_yn</code>	Logical value indicating whether vital rate will be subject to density dependence.
<code>style</code>	Style of density dependence, coded as 1, 2, 3, 4, or \emptyset for the Ricker, Beverton-Holt, Usher, or logistic function, or no density dependence, respectively.
<code>time_delay</code>	The time delay on density dependence, in time steps.
<code>alpha</code>	The value of alpha in the Ricker, Beverton-Holt, or Usher function, or the value of carrying capacity, K , in the logistic function.
<code>beta</code>	The value of beta in the Ricker, Beverton-Holt, or Usher function.

Notes

This function provides inputs when density dependence is operationalized directly on vital rates. It can be used only in function `f_projection3()`. Users wishing to modify matrix elements directly by density dependence functions for use in function-based or raw projections with functions `projection3()` and `f_projection3()` should use function `density_input()` to provide the correct inputs.

The parameters `alpha` and `beta` are applied according to the two-parameter Ricker function, the two-parameter Beverton-Holt function, the two-parameter Usher function, or the one-parameter logistic function. Although the default is that a 1 time step delay is assumed, greater time delays can be set through the `time_delay` option.

When using the logistic function, it is possible that the time delay used in density dependent simulations will cause matrix elements to become negative. To prevent this behavior, set the associated beta term to $1 \cdot \emptyset$. Doing so will set K as the hard limit in the logistic equation, essentially setting a minimum limit at \emptyset for all matrix elements modified.

See Also

[density_input\(\)](#)
[f_projection3\(\)](#)

Examples

```

# Lathyrus projection example with historical matrices
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", "Sd1", "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)
obsvector <- c(0, 1, 0, 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)
immvector <- c(1, 1, 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)
indataset <- c(0, 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)

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)

lathmodelsln3 <- modelsearch(lathvertln, historical = TRUE,
approach = "mixed", suite = "main",
vitalrates = c("surv", "obs", "size", "repst", "fec"), juvestimate = "Sd1",
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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "mat", "Sd", "Sd1"),
stage2 = c("Sd", "Sd", "Sd", "Sd", "Sd1", "rep", "rep"),
stage1 = c("Sd", "rep", "Sd", "rep", "Sd", "mat", "mat"),
eststage3 = c(NA, NA, NA, NA, "mat", NA, NA),
eststage2 = c(NA, NA, NA, NA, "Sd1", NA, NA),
eststage1 = c(NA, NA, NA, NA, "Sd1", NA, NA),
givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, NA, 0.345, 0.054),
type = c(1, 1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1, 1))

```

```

stageframe = lathframe1n, historical = TRUE)

# While we do not use MPMs to initialize f_projections3(), we do use MPMs to
# initialize functions start_input() and density_input().
lathmat31n <- flefko3(year = "all", patch = "all", stageframe = lathframe1n,
  modelsuite = lathmodels1n3, data = lathvert1n, supplement = lathsupp3,
  reduce = FALSE)

e3m_sv <- start_input(lathmat31n, stage2 = "Sd", stage1 = "Sd", value = 1000)

dyn7 <- c(TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
  FALSE, FALSE, FALSE, FALSE, FALSE)
dst7 <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
dal7 <- c(0.5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
dbe7 <- c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

e3d_vr <- density_vr(density_yn = dyn7, style = dst7, alpha = dal7,
  beta = dbe7)

trial7_dvr <- f_projection3(format = 1, data = lathvert1n,
  modelsuite = lathmodels1n3, stageframe = lathframe1n, nreps = 2,
  times = 1000, stochastic = TRUE, standardize = FALSE, growthonly = TRUE,
  integeronly = FALSE, substoch = 0, sp_density = 0, start_frame = e3m_sv,
  density_vr = e3d_vr)
summary(trial7_dvr)

```

diff_1M

Calculate Difference Matrices Between lefkoMat Objects of Equal Dimensions

Description

Function `diff_1M()` takes two `lefkoMat` objects with completely equal dimensions, including both the size and number of matrices, and gives the matrix differences between each corresponding set.

Usage

```
diff_1M(mpm1, mpm2)
```

Arguments

<code>mpm1</code>	The first <code>lefkoMat</code> object.
<code>mpm2</code>	The second <code>lefkoMat</code> object.

Value

An object of class `lefkoDiff`, which is a set of A, U, and F matrices corresponding to the differences between each set of matrices, followed by the `hstages`, `ahstages`, and `labels` elements from each input `lefkoMat` object. Elements labelled with a 1 at the end refer to `mpm1`, while those labelled 2 at the end refer to `mpm2`.

Notes

The exact difference is calculated as the respective matrix in `mpm1` minus the corresponding matrix in `mpm2`.

This function first checks to see if the number of matrices is the same, and then whether the matrix dimensions are the same. If the two sets differ in at least one of these characteristics, then the function will yield a fatal error.

If the lengths and dimensions of the input `lefkoMat` objects are the same, then this will check if the `labels` element is essentially the same. If not, then the function will yield a warning, but will still operate.

Examples

```
sizevector <- c(0, 0, 0, 0, 0, 0, 1, 3, 6, 11, 19.5)
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, 1.5, 1.5, 3.5, 5)
comments <- c("Dormant seed", "1st yr protocorm", "2nd yr protocorm",
  "3rd yr protocorm", "Seedling", "Dormant adult",
  "Extra small adult (1 shoot)", "Small adult (2-4 shoots)",
  "Medium adult (5-7 shoots)", "Large adult (8-14 shoots)",
  "Extra large adult (>14 shoots)")
cypframe_raw <- sf_create(sizes = sizevector, stagenames = stagevector,
  repstatus = repvector, obsstatus = obsvector, matstatus = matvector,
  propstatus = propvector, immstatus = immvector, indataset = indataset,
  binhalfwidth = binvec, comments = comments)

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)

seeds_per_pod <- 5000

cypsupp2_raw <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "SL", "D",
  "XSm", "SD", "P1"),
```



```

stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep", "rep"),
eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "XSm", NA, NA),
eststage2 = c(NA, NA, NA, NA, NA, NA, "XSm", "XSm", NA, NA),
givenrate = c(0.03, 0.15, 0.1, 0.1, 0.1, 0.05, NA, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, (0.5 * seeds_per_pod),
(0.5 * seeds_per_pod)),
type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
stageframe = cypframe_raw, historical = FALSE)
cypsupp3_raw <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3",
"SL", "SL", "SL", "D", "D", "SD", "P1"),
stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
"rep", "rep"),
stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "SL", "P3",
"SL", "mat", "mat"),
eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "D", NA, NA),
eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", NA, NA),
eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", NA, NA),
givenrate = c(0.01, 0.05, 0.10, 0.20, 0.1, 0.1, 0.05, 0.05, 0.05, NA, NA,
NA, NA),
multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
(0.5 * seeds_per_pod), (0.5 * seeds_per_pod)),
type = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
type_t12 = c(1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1),
stageframe = cypframe_raw, historical = TRUE)

cypmatrix2rp <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
year = "all", patch = "all", stages = c("stage3", "stage2"),
size = c("size3added", "size2added"), supplement = cypsupp2_raw,
yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
year = "all", stages = c("stage3", "stage2"),
size = c("size3added", "size2added"), supplement = cypsupp2_raw,
yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix3rp <- rlefk3(data = cypraw_v1, stageframe = cypframe_raw,
year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix3r <- rlefk3(data = cypraw_v1, stageframe = cypframe_raw,
year = "all", stages = c("stage3", "stage2", "stage1"),
size = c("size3added", "size2added", "size1added"), supplement = cypsupp3_raw,
yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix2r_3 <- hist_null(cypmatrix2r)
cypmatrix2r_3 <- delete_lm(cypmatrix2r_3, year = 2004)
diff_r <- diff_lm(cypmatrix3r, cypmatrix2r_3)

cypmatrix2rp_3 <- hist_null(cypmatrix2rp)
cypmatrix2rp_3 <- delete_lm(cypmatrix2rp_3, year = 2004)
diff_rp <- diff_lm(cypmatrix3rp, cypmatrix2rp_3)

```

Description

`elasticity3()` is a generic function that returns the elasticity of the population growth rate to the elements of the matrices in a matrix population model. Currently, this function estimates both deterministic and stochastic elasticities, where the growth rate is λ in the former case and the log of the stochastic λ in the latter case. This function is made to handle very large and sparse matrices supplied as `lefkoMat` objects, as lists of matrices, and as individual matrices.

Usage

```
elasticity3(mats, ...)
```

Arguments

<code>mats</code>	A <code>lefkoMat</code> object, or population projection matrix, for which the stable stage distribution is desired.
<code>...</code>	Other parameters.

Value

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

See Also

[elasticity3.lefkoMat\(\)](#)
[elasticity3.matrix\(\)](#)
[elasticity3.list\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfeko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
elasticity3(ehrlen3mean)

# Cypridium example
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, 0, 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)

```

```

cypsups2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type =c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

elasticity3(cypmatrix2r)

```

elasticity3.lefkoMat *Estimate Elasticity of Population Growth Rate of a lefkoMat Object*

Description

elasticity3.lefkoMat() returns the elasticities of population growth rate to elements of all \$A matrices in an object of class lefkoMat. If deterministic, then λ is taken as the population growth rate. If stochastic, then stochastic λ , or the stochastic growth rate, is taken as the population growth rate. 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,
  stochastic = FALSE,
  steps = 10000,
  time_weights = NA,
  sparse = "auto",
  append_mats = FALSE,
  ...
)

```

Arguments

mats	An object of class lefkoMat.
stochastic	A logical value determining whether to conduct a deterministic (FALSE) or stochastic (TRUE) elasticity analysis. Defaults to FALSE.

steps	The number of occasions to project forward in stochastic simulation. Defaults to 10,000.
time_weights	Numeric vector denoting the probabilistic weightings of annual matrices. Defaults to equal weighting among occasions.
sparse	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
append_mats	A logical value indicating whether to include the original A, U, and F matrices in the output lefkoElas object.
...	Other parameters.

Value

This function returns an object of class `lefkoElas`, which is a list with 8 elements. The first, `h_elasmats`, is a list of historical elasticity matrices (NULL if an `ahMPM` is used as input). The second, `ah_elasmats`, is a list of either ahistorical elasticity matrices if an `ahMPM` is used as input, or, if an `hMPM` is used as input, then the result is a list of elasticity matrices in which historical elasticities have been summed by the stage in occasions t and $t+1$ to produce historically-corrected elasticity matrices, which are equivalent in dimension to ahistorical elasticity matrices but reflect the effects of stage in occasion $t-1$. The third element, `h_stages`, is a data frame showing historical stage pairs (NULL if `ahMPM` used as input). The fourth element, `agestages`, shows age-stage combinations in the order used in age-by-stage MPMs, if supplied. The fifth element, `ah_stages`, is a data frame showing the order of ahistorical stages. The last 3 elements are the A, U, and F portions of the input.

Notes

Deterministic elasticities are estimated as eqn. 9.72 in Caswell (2001, *Matrix Population Models*). Stochastic elasticities are estimated as eqn. 14.99 in Caswell (2001). Note that stochastic elasticities are of the stochastic λ , while stochastic sensitivities are with regard to the log of the stochastic λ .

See Also

[elasticity3\(\)](#)
[elasticity3.matrix\(\)](#)
[elasticity3.list\(\)](#)
[summary.lefkoElas\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "F1o", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "indiv")

elasticity3(ehrlen3, stochastic = TRUE)

# Cypridium example
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, 0, 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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", individcol = "individ")

elasticity3(cypmatrix2r)

```

elasticity3.list

Estimate Elasticity of Population Growth Rate of a List of Matrices

Description

`elasticity3.list()` 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 'list'
elasticity3(
  mats,
  stochastic = FALSE,
  steps = 10000,
  time_weights = NA,
  historical = FALSE,
  sparse = "auto",
  append_mats = FALSE,
  ...
)

```

Arguments

<code>mats</code>	A list of objects of class <code>matrix</code> .
<code>stochastic</code>	A logical value determining whether to conduct a deterministic (FALSE) or stochastic (TRUE) elasticity analysis. Defaults to FALSE.
<code>steps</code>	The number of occasions to project forward in stochastic simulation. Defaults to 10,000.
<code>time_weights</code>	Numeric vector denoting the probabilistic weightings of annual matrices. Defaults to equal weighting among occasions.
<code>historical</code>	A logical value denoting whether the input matrices are historical. Defaults to FALSE.
<code>sparse</code>	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
<code>append_mats</code>	A logical value indicating whether to include the original matrices input as object <code>mats</code> in the output <code>lefkoElas</code> object.
<code>...</code>	Other parameters.

Value

This function returns an object of class `lefkoElas`, which is a list with 8 elements. The first, `h_elasmats`, is a list of historical elasticity matrices, though in the standard list case it returns a NULL value. The second, `ah_elasmats`, is a list of ahistorical elasticity matrices. The third element, `h_stages`, the fourth element, `agestages`, and the fifth element, `ah_stages`, are set to NULL. The last 3 elements are the original A matrices in element A, followed by NULL values for the U and F elements.

Notes

Deterministic elasticities are estimated as eqn. 9.72 in Caswell (2001, *Matrix Population Models*). Stochastic elasticities are estimated as eqn. 14.99 in Caswell (2001). Note that stochastic elasticities are of stochastic λ , while stochastic sensitivities are with regard to the log of the stochastic λ .

See Also

[elasticity3\(\)](#)
[elasticity3.lefkoMat\(\)](#)
[elasticity3.matrix\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

elasticity3(ehrlen3$A, stochastic = TRUE)

# Cypridium example
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, 0, 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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

elasticity3(cypmatrix2r$A)

```

elasticity3.matrix *Estimate Elasticity of Population Growth Rate of a Single Matrix*

Description

`elasticity3.matrix()` returns the elasticities of lambda to elements of a single matrix. Because this handles only one matrix, the elasticities are inherently deterministic and based on the dominant eigen value as the best metric of the population growth rate. 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, sparse = "auto", ...)

```

Arguments

<code>mats</code>	An object of class <code>matrix</code> .
<code>sparse</code>	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
<code>...</code>	Other parameters.

Value

This function returns a single elasticity matrix.

See Also

[elasticity3\(\)](#)
[elasticity3.lefkoMat\(\)](#)
[elasticity3.list\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")
```

```

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

# Cypripedium example
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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rleeko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", individcol = "individ")

elasticity3(cypmatrix2r$A[[1]])

```

Description

Function `flefko2()` returns ahistorical MPMs corresponding to the patches and occasions 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 occasion combinations corresponding to these matrices.

Usage

```
flefko2(
  year = "all",
  patch = "all",
  stageframe,
  supplement = NULL,
  repmatrix = NULL,
  overwrite = NULL,
  data = NULL,
  modelsuite = NULL,
  surv_model = NULL,
  obs_model = NULL,
  size_model = NULL,
  sizeb_model = NULL,
  sizec_model = NULL,
  repst_model = NULL,
  fec_model = NULL,
  jsurv_model = NULL,
  jobs_model = NULL,
  jsize_model = NULL,
  jsizeb_model = NULL,
  jsizec_model = NULL,
  jrepst_model = NULL,
  jmatst_model = NULL,
  paramnames = NULL,
  inda = NULL,
  indb = NULL,
  indc = NULL,
  surv_dev = 0,
  obs_dev = 0,
  size_dev = 0,
  sizeb_dev = 0,
  sizec_dev = 0,
  repst_dev = 0,
  fec_dev = 0,
  jsurv_dev = 0,
  jobs_dev = 0,
  jsize_dev = 0,
  jsizeb_dev = 0,
  jsizec_dev = 0,
  jrepst_dev = 0,
```

```

jmatst_dev = 0,
density = NA,
repmatrix = 1,
random.inda = FALSE,
random.indb = FALSE,
random.indc = FALSE,
negfec = FALSE,
ipm_method = "CDF",
reduce = FALSE,
err_check = FALSE,
exp_tol = 700,
theta_tol = 1e+08
)

```

Arguments

year	A variable corresponding to the observation occasion, or a set of such values, given in values associated with the year term used in linear model development. Defaults to "all", in which case matrices will be estimated for all occasions.
patch	A variable designating which patches or subpopulations will have matrices estimated. Defaults to "all", but can also be set to specific patch names or a vector thereof.
stageframe	An object of class <code>stageframe</code> . These objects are generated by function <code>sf_create()</code> , and include information on the size, observation status, propagule status, reproduction status, immaturity status, maturity status, stage group, size bin widths, and other key characteristics of each ahistorical stage.
supplement	An optional data frame of class <code>lefkoSD</code> that provides supplemental data that should be incorporated into the MPM. Three kinds of data may be integrated this way: transitions to be estimated via the use of proxy transitions, transition overwrites from the literature or supplemental studies, and transition multipliers for survival and fecundity. This data frame should be produced using the <code>supplemental()</code> function. Can be used in place of or in addition to an overwrite table (see <code>overwrite</code> below) and a reproduction matrix (see <code>repmatrix</code> below).
repmatrix	An optional reproduction matrix. This matrix is composed mostly of 0s, with non-zero entries acting as element identifiers and multipliers for fecundity (with 1 equaling full fecundity). If left blank, and no <code>supplement</code> is provided, then <code>flefko2()</code> will assume that all stages marked as reproductive produce offspring at 1x that of estimated fecundity, and that offspring production will yield the first stage noted as propagule or immature. Must be the dimensions of an ahistorical matrix.
overwrite	An optional data frame developed with the <code>overwrite()</code> function describing transitions to be overwritten either with given values or with other estimated transitions. Note that this function supplements <code>overwrite</code> data provided in <code>supplement</code> .
data	The historical vertical demographic data frame used to estimate vital rates (class <code>hfvdata</code>), which is required to initialize times and patches properly. Variable

names should correspond to the naming conventions in `verticalize3()` and `historicalize3()`. Not required if option `modelsuite` is set to a `vrn_input` object.

<code>modelsuite</code>	One of two kinds of lists. The first is a <code>lefk0Mod</code> object holding the vital rate models and associated metadata. Alternatively, an object of class <code>vrn_input</code> may be provided. If given, then <code>surv_model</code> , <code>obs_model</code> , <code>size_model</code> , <code>sizeb_model</code> , <code>sizec_model</code> , <code>repst_model</code> , <code>fec_model</code> , <code>jsurv_model</code> , <code>jobs_model</code> , <code>jsize_model</code> , <code>jsizeb_model</code> , <code>jsizec_model</code> , <code>jrepst_model</code> , <code>jmatst_model</code> , and <code>paramnames</code> are not required. No models should include size or reproductive status in occasion $t-1$. Although this is optional input, it is recommended, and without it all vital rate model inputs (named <code>XX_model</code>) are required.
<code>surv_model</code>	A linear model predicting survival probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>obs_model</code>	A linear model predicting sprouting or observation probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>size_model</code>	A linear model predicting primary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>sizeb_model</code>	A linear model predicting secondary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>sizec_model</code>	A linear model predicting tertiary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>repst_model</code>	A linear model predicting reproduction probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>fec_model</code>	A linear model predicting fecundity. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>jsurv_model</code>	A linear model predicting juvenile survival probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>jobs_model</code>	A linear model predicting juvenile sprouting or observation probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable

	under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>jsize_model</code>	A linear model predicting juvenile primary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>jsizeb_model</code>	A linear model predicting juvenile secondary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>jsizec_model</code>	A linear model predicting juvenile tertiary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>jrepst_model</code>	A linear model predicting reproduction probability of a mature individual that was immature in time t . This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>jmatst_model</code>	A linear model predicting maturity probability of an individual that was immature in time t . This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
<code>paramnames</code>	A data frame with three columns, the first describing all terms used in linear modeling, the second (must be called <code>mainparams</code>) giving the general model terms that will be used in matrix creation, and the third showing the equivalent terms used in modeling (must be named <code>modelparams</code>). Function <code>create_pm()</code> can be used to create a skeleton <code>paramnames</code> object, which can then be edited. Only required if <code>modelsuite</code> is not supplied.
<code>inda</code>	Can be a single value to use for individual covariate <code>a</code> in all matrices, a pair of values to use for times t and $t-1$ in historical matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to <code>NULL</code> .
<code>indb</code>	Can be a single value to use for individual covariate <code>b</code> in all matrices, a pair of values to use for times t and $t-1$ in historical matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to <code>NULL</code> .
<code>indc</code>	Can be a single value to use for individual covariate <code>c</code> in all matrices, a pair of values to use for times t and $t-1$ in historical matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to <code>NULL</code> .
<code>surv_dev</code>	A numeric value to be added to the y-intercept in the linear model for survival probability. Defaults to <code>0</code> .
<code>obs_dev</code>	A numeric value to be added to the y-intercept in the linear model for observation probability. Defaults to <code>0</code> .
<code>size_dev</code>	A numeric value to be added to the y-intercept in the linear model for primary size. Defaults to <code>0</code> .

sizeb_dev	A numeric value to be added to the y-intercept in the linear model for secondary size. Defaults to 0.
sizec_dev	A numeric value to be added to the y-intercept in the linear model for tertiary size. Defaults to 0.
repst_dev	A numeric value to be added to the y-intercept in the linear model for probability of reproduction. Defaults to 0.
fec_dev	A numeric value to be added to the y-intercept in the linear model for fecundity. Defaults to 0.
jsurv_dev	A numeric value to be added to the y-intercept in the linear model for juvenile survival probability. Defaults to 0.
jobs_dev	A numeric value to be added to the y-intercept in the linear model for juvenile observation probability. Defaults to 0.
jsize_dev	A numeric value to be added to the y-intercept in the linear model for juvenile primary size. Defaults to 0.
jsizeb_dev	A numeric value to be added to the y-intercept in the linear model for juvenile secondary size. Defaults to 0.
jsizec_dev	A numeric value to be added to the y-intercept in the linear model for juvenile tertiary size. Defaults to 0.
jrepst_dev	A numeric value to be added to the y-intercept in the linear model for juvenile reproduction probability. Defaults to 0.
jmatst_dev	A numeric value to be added to the y-intercept in the linear model for juvenile maturity probability. Defaults to 0.
density	A numeric value indicating density value to use to propagate matrices. Only needed if density is an explanatory term used in one or more vital rate models. Defaults to NA.
repmo	A scalar multiplier of fecundity. Defaults to 1.
random.ind	A logical value denoting whether to treat individual covariate a as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random.indb	A logical value denoting whether to treat individual covariate b as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random.indc	A logical value denoting whether to treat individual covariate c as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
negfec	A logical value denoting whether fecundity values estimated to be negative should be reset to 0. Defaults to FALSE.
ipm_method	A string indicating what method to use to estimate size transition probabilities, if size is treated as continuous. Options include: "midpoint", which utilizes the midpoint method; and "CDF", which uses the cumulative distribution function. Defaults to "CDF".
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.

err_check	A logical value indicating whether to append matrices of vital rate probabilities associated with each matrix to the <code>lefkoMat</code> object generated. These matrices are developed internally and can be used for error checking (see element out in Value section below for details). Defaults to FALSE.
exp_tol	A numeric value used to indicate a maximum value to set exponents to in the core kernel to prevent numerical overflow. Defaults to 700.
theta_tol	A numeric value used to indicate a maximum value to theta as used in the negative binomial probability density kernel. Defaults to 100000000, but can be reset to other values during error checking.

Value

If all inputs are properly formatted, then this function will return an object of class `lefkoMat`, which is a list that holds the matrix projection model and all of its metadata. The structure has the following elements:

A	A list of full projection matrices in order of sorted patches and occasion times. All matrices output in R's <code>matrix</code> class.
U	A list of survival transition matrices sorted as in A. All matrices output in R's <code>matrix</code> class.
F	A list of fecundity matrices sorted as in A. All matrices output in R's <code>matrix</code> class.
hstages	A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs. Set to NA for ahistorical matrices.
agestages	A data frame showing age-stage pairs. In this function, it is set to NA. Only used in output to function <code>aflefko2()</code> .
ahstages	A data frame detailing the characteristics of associated ahistorical stages, in the form of a modified stageframe that includes status as an entry stage through reproduction.
labels	A data frame giving the population, patch, and year of each matrix in order. In <code>flefko2()</code> , only one population may be analyzed at once.
matrixqc	A short vector describing the number of non-zero elements in U and F matrices, and the number of matrices.
modelqc	This is the qc portion of the <code>modelsuite</code> input.
prob_out	An optional element only added if <code>err_check = TRUE</code> . This is a list of vital rate probability matrices, with 7 columns in the order of survival, observation probability, reproduction probability, primary size transition probability, secondary size transition probability, tertiary size transition probability, and probability of juvenile transition to maturity.
allstages	An optional element only added if <code>err_check = TRUE</code> . This is a data frame giving the values used to determine each matrix element capable of being estimated.

Notes

Unlike `rlefko2()`, `rlefko3()`, `arlefko2()`, and `rleslie()`, this function does not currently distinguish populations. Users wishing to use the same vital rate models across populations should label them as patches (though we do not advise this approach, as populations should typically be treated as statistically independent).

This function will yield incorrect estimates if the models utilized incorporate state in occasion $t-1$. Only use models developed testing for ahistorical effects.

The default behavior of this function is to estimate fecundity with regards to transitions specified via associated fecundity multipliers in either `supplement` or `repmatrix`. If both of these fields are left empty, then fecundity will be estimated at full for all transitions leading from reproductive stages to immature and propagule stages. However, if a `supplement` is provided and a `repmatrix` is not, or if `repmatrix` is set to \emptyset , then only fecundity transitions noted in the `supplement` will be set to non-zero values. To use the default behavior of setting all reproductive stages to reproduce at full fecundity into immature and propagule stages, but also incorporate given or proxy survival transitions, input those given and proxy transitions through the `overwrite` option.

The reproduction matrix (field `repmatrix`) may only be supplied as ahistorical. If provided as historical, then `flefko2()` will fail and produce an error.

Users may at times wish to estimate MPMs using a dataset incorporating multiple patches or subpopulations, but without discriminating between those patches or subpopulations. Should the aim of analysis be a general MPM that does not distinguish these patches or subpopulations, the `modelsearch()` run should not include patch terms.

Input options including multiple variable names must be entered in the order of variables in occasion $t+1$ and t . Rearranging the order will lead to erroneous calculations, and may lead to fatal errors.

Care should be taken to match the random status of year and patch to the states of those variables within the `modelsuite`. If they do not match, then they will be treated as zeroes in vital rate estimation.

The `ipm_method` function gives the option of using two different means of estimating the probability of size transition. The midpoint method ("`midpoint`") refers to the method in which the probability is estimated by first estimating the probability associated with transition from the exact size at the midpoint of the size class using the corresponding probability density function, and then multiplying that value by the bin width of the size class. Doak et al. 2021 (Ecological Monographs) noted that this method can produce biased results, with total size transitions associated with a specific size not totaling to 1.0 and even specific size transition probabilities capable of being estimated at values greater than 1.0. The alternative and default method, "`CDF`", uses the corresponding cumulative density function to estimate the probability of size transition as the cumulative probability of size transition at the greater limit of the size class minus the cumulative probability of size transition at the lower limit of the size class. The latter method avoids this bias. Note, however, that both methods are exact and unbiased for negative binomial and Poisson distributions.

Under the Gaussian and gamma size distributions, the number of estimated parameters may differ between the two `ipm_method` settings. Because the midpoint method has a tendency to incorporate upward bias in the estimation of size transition probabilities, it is more likely to yield non-zero values when the true probability is extremely close to 0. This will result in the `summary.lfekoMat` function yielding higher numbers of estimated parameters than the `ipm_method = "CDF"` yields in some cases.

Using the `err_check` option will produce a matrix of 7 columns, each characterizing a different vital rate. The product of each row yields an element in the associated U matrix. The number and

order of elements in each column of this matrix matches the associated matrix in column vector format. Use of this option is generally for the purposes of debugging code.

Individual covariates are treated as categorical only if they are set as random terms. Fixed categorical individual covariates are currently not allowed. However, such terms may be supplied if the `modelsuite` option is set to a `vrn_input` object. In that case, the user should also set the logical random switch for the individual covariate to be used to TRUE (e.g., `random.indata = TRUE`).

See Also

[flefko3\(\)](#)
[aflefko2\(\)](#)
[arlefko2\(\)](#)
[fleslie\(\)](#)
[rlefko3\(\)](#)
[rlefko2\(\)](#)
[rleslie\(\)](#)

Examples

```
# Lathyrus example
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", "Sd1", "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, 0, 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,
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)

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$fece2 <- round(lathvertln$fece2)
lathvertln$fece1 <- round(lathvertln$fece1)
lathvertln$fece3 <- round(lathvertln$fece3)

lathmodelsln2 <- modelsearch(lathvertln, historical = FALSE,
  approach = "mixed", suite = "main",
  vitalrates = c("surv", "obs", "size", "repst", "fec"), juvestimate = "Sd1",
  bestfit = "AICc&k", sizedist = "gaussian", fecdist = "poisson",
  indiv = "indiv", patch = "patchid", year = "year2",
  year.as.random = TRUE, patch.as.random = TRUE, show.model.tables = TRUE,
  quiet = TRUE)

# Here we use supplemental to provide overwrite and reproductive info
lathsupp2 <- supplemental(stage3 = c("Sd", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "rep", "rep"),
  givenrate = c(0.345, 0.054, NA, NA),
  multiplier = c(NA, NA, 0.345, 0.054),
  type = c(1, 1, 3, 3), stageframe = lathframeIn, historical = FALSE)

lathmat2ln <- flefko2(year = "all", patch = "all", stageframe = lathframeIn,
  modelsuite = lathmodelsln2, data = lathvertln, supplement = lathsupp2,
  reduce = FALSE)

summary(lathmat2ln)

#Cypripedium example using three size metrics for classification
data(cypdata)
sizevector.f <- c(0, 0, 0, 0, 0, 0, seq(1, 12, by = 1), seq(0, 9, by = 1),
  seq(0, 8, by = 1), seq(0, 7, by = 1), seq(0, 6, by = 1), seq(0, 5, by = 1),
  seq(0, 4, by = 1), seq(0, 3, by = 1), 0, 1, 2, 0, 1, 0,
  0, 0, 1, 0)
sizebvector.f <- c(0, 0, 0, 0, 0, 0, rep(0, 12), rep(1, 10), rep(2, 9),
  rep(3, 8), rep(4, 7), rep(5, 6), rep(6, 5), rep(7, 4), rep(8, 3), 9, 9, 10,
  0, 1, 1, 2)
sizecvector.f <- c(0, 0, 0, 0, 0, 0, rep(0, 12), rep(0, 10), rep(0, 9),
  rep(0, 8), rep(0, 7), rep(0, 6), rep(0, 5), rep(0, 4), 0, 0, 0, 0, 0, 0,
  1, 1, 1, 1)
stagevector.f <- c("DS", "P1", "P2", "P3", "Sd1", "Dorm", "V1 I0 D0",
  "V2 I0 D0", "V3 I0 D0", "V4 I0 D0", "V5 I0 D0", "V6 I0 D0", "V7 I0 D0",
  "V8 I0 D0", "V9 I0 D0", "V10 I0 D0", "V11 I0 D0", "V12 I0 D0", "V0 I1 D0",
  "V1 I1 D0", "V2 I1 D0", "V3 I1 D0", "V4 I1 D0", "V5 I1 D0", "V6 I1 D0",
  "V7 I1 D0", "V8 I1 D0", "V9 I1 D0", "V0 I2 D0", "V1 I2 D0", "V2 I2 D0",
  "V3 I2 D0", "V4 I2 D0", "V5 I2 D0", "V6 I2 D0", "V7 I2 D0", "V8 I2 D0",
  "V0 I3 D0", "V1 I3 D0", "V2 I3 D0", "V3 I3 D0", "V4 I3 D0", "V5 I3 D0",
  "V6 I3 D0", "V7 I3 D0", "V0 I4 D0", "V1 I4 D0", "V2 I4 D0", "V3 I4 D0",
  "V4 I4 D0", "V5 I4 D0", "V6 I4 D0", "V0 I5 D0", "V1 I5 D0", "V2 I5 D0",
  "V3 I5 D0", "V4 I5 D0", "V5 I5 D0", "V0 I6 D0", "V1 I6 D0", "V2 I6 D0",
  "V3 I6 D0", "V4 I6 D0", "V0 I7 D0", "V1 I7 D0", "V2 I7 D0", "V3 I7 D0",
  "V0 I8 D0", "V1 I8 D0", "V2 I8 D0", "V0 I9 D0", "V1 I9 D0", "V0 I10 D0",
  "V0 I0 D1", "V0 I1 D1", "V1 I1 D1", "V0 I2 D1")
repvector.f <- c(0, 0, 0, 0, 0, 0, rep(0, 13), rep(1, 59))

```

```

obsvector.f <- c(0, 0, 0, 0, 0, 0, rep(1, 71))
matvector.f <- c(0, 0, 0, 0, 0, rep(1, 72))
immvector.f <- c(0, 1, 1, 1, 1, rep(0, 72))
propvector.f <- c(1, rep(0, 76))
indataset.f <- c(0, 0, 0, 0, 0, rep(1, 72))
binvec.f <- c(0, 0, 0, 0, 0, rep(0.5, 72))
binbvec.f <- c(0, 0, 0, 0, 0, rep(0.5, 72))
bincvec.f <- c(0, 0, 0, 0, 0, rep(0.5, 72))

vertframe.f <- sf_create(sizes = sizevector.f, sizesb = sizebvector.f,
  sizesc = sizecvector.f, stagenames = stagevector.f, repstatus = repvector.f,
  obsstatus = obsvector.f, propstatus = propvector.f, immstatus = immvector.f,
  matstatus = matvector.f, indataset = indataset.f, binhalfwidth = binvec.f,
  binhalfwidthb = binbvec.f, binhalfwidthc = bincvec.f)

vert.data.f <- verticalize3(cypdata, noyears = 6, firstyear = 2004,
  individcol = "plantid", blocksize = 4, sizeacol = "Veg.04",
  sizebcol = "Inf.04", sizeccol = "Inf2.04", repstracol = "Inf.04",
  repstrbcol = "Inf2.04", fecacol = "Pod.04", censorcol = "censor",
  censorkeep = 1, censorRepeat = FALSE, stageassign = vertframe.f,
  stagesize = "sizeabc", NAas0 = TRUE, censor = FALSE)

vertmodels2f <- modelsearch(vert.data.f, historical = FALSE, suite = "main",
  sizeb = c("sizeb3", "sizeb2", "sizeb1"), sizec = c("sizec3", "sizec2", "sizec1"),
  approach = "glm", vitalrates = c("surv", "obs", "size", "repst", "fec"),
  sizedist = "negbin", sizebdist = "poisson", sizecdist = "poisson",
  fecdist = "poisson", patch.as.random = TRUE, year.as.random = TRUE)

vertsupp2f <- supplemental(stage3 = c("DS", "P1", "P2", "P3", "Sd1", "Sd1",
  "Dorm", "V1 I0 D0", "V2 I0 D0", "V3 I0 D0", "DS", "P1"),
  stage2 = c("DS", "DS", "P1", "P2", "P3", "Sd1", "Sd1", "Sd1", "Sd1", "Sd1",
  "rep", "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "Dorm", "V1 I0 D0", "V2 I0 D0",
  "V3 I0 D0", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "V1 I0 D0", "V1 I0 D0", "V1 I0 D0",
  "V1 I0 D0", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, 0.40, NA, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, 0.5 * 5000, 0.5 * 5000),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3), stageframe = vertframe.f,
  historical = FALSE)

vert.mats.f2 <- flefko2(stageframe = vertframe.f, supplement = vertsupp2f,
  data = vert.data.f, modelsuite = vertmodels2f)
summary(vert.mats.f2)

```

Description

Function `flefko3()` returns function-based historical MPMs corresponding to the patches and occasions 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 occasion combinations corresponding to these matrices.

Usage

```
flefko3(  
  year = "all",  
  patch = "all",  
  stageframe,  
  supplement = NULL,  
  repmatrix = NULL,  
  overwrite = NULL,  
  data = NULL,  
  modelsuite = NULL,  
  surv_model = NULL,  
  obs_model = NULL,  
  size_model = NULL,  
  sizeb_model = NULL,  
  sizec_model = NULL,  
  repst_model = NULL,  
  fec_model = NULL,  
  jsurv_model = NULL,  
  jobs_model = NULL,  
  jsize_model = NULL,  
  jsizeb_model = NULL,  
  jsizec_model = NULL,  
  jrepst_model = NULL,  
  jmatst_model = NULL,  
  paramnames = NULL,  
  inda = NULL,  
  indb = NULL,  
  indc = NULL,  
  surv_dev = 0,  
  obs_dev = 0,  
  size_dev = 0,  
  sizeb_dev = 0,  
  sizec_dev = 0,  
  repst_dev = 0,  
  fec_dev = 0,  
  jsurv_dev = 0,  
  jobs_dev = 0,  
  jsize_dev = 0,  
  jsizeb_dev = 0,  
  jsizec_dev = 0,  
  jrepst_dev = 0,  
)
```

```

jmatst_dev = 0,
density = NA,
repmat = 1,
random.inda = FALSE,
random.indb = FALSE,
random.indc = FALSE,
negfec = FALSE,
format = "ehrlen",
ipm_method = "CDF",
reduce = FALSE,
err_check = FALSE,
exp_tol = 700,
theta_tol = 1e+08
)

```

Arguments

year	A variable corresponding to the observation occasion, or a set of such values, given in values associated with the year term used in linear model development. Defaults to "all", in which case matrices will be estimated for all occasions.
patch	A variable designating which patches or subpopulations will have matrices estimated. Defaults to "all", but can also be set to specific patch names or a vector thereof.
stageframe	An object of class <code>stageframe</code> . These objects are generated by function <code>sf_create()</code> , and include information on the size, observation status, propagule status, reproduction status, immaturity status, maturity status, stage group, size bin widths, and other key characteristics of each ahistorical stage.
supplement	An optional data frame of class <code>lefkoSD</code> that provides supplemental data that should be incorporated into the MPM. Three kinds of data may be integrated this way: transitions to be estimated via the use of proxy transitions, transition overwrites from the literature or supplemental studies, and transition multipliers for survival and fecundity. This data frame should be produced using the <code>supplemental()</code> function. Can be used in place of or in addition to an overwrite table (see <code>overwrite</code> below) and a reproduction matrix (see <code>repmat</code> below).
repmat	An optional reproduction matrix. This matrix is composed mostly of 0s, with non-zero entries acting as element identifiers and multipliers for fecundity (with 1 equaling full fecundity). If left blank, and no supplement is provided, then <code>flefko3()</code> will assume that all stages marked as reproductive produce offspring at 1x that of estimated fecundity, and that offspring production will yield the first stage noted as propagule or immature. May be the dimensions of either a historical or an ahistorical matrix. If the latter, then all stages will be used in occasion $t-1$ for each suggested ahistorical transition.
overwrite	An optional data frame developed with the <code>overwrite()</code> function describing transitions to be overwritten either with given values or with other estimated transitions. Note that this function supplements overwrite data provided in supplement.

data	The historical vertical demographic data frame used to estimate vital rates (class <code>hfvdata</code>), which is required to initialize times and patches properly. Variable names should correspond to the naming conventions in <code>verticalize3()</code> and <code>historicalize3()</code> . Not required if option <code>modelsuite</code> is set to a <code>vrn_input</code> object.
modelsuite	One of two kinds of lists. The first is a <code>lefk0Mod</code> object holding the vital rate models and associated metadata. Alternatively, an object of class <code>vrn_input</code> may be provided. If given, then <code>surv_model</code> , <code>obs_model</code> , <code>size_model</code> , <code>sizeb_model</code> , <code>sizec_model</code> , <code>repst_model</code> , <code>fec_model</code> , <code>jsurv_model</code> , <code>jobs_model</code> , <code>jsize_model</code> , <code>jsizeb_model</code> , <code>jsizec_model</code> , <code>jrepst_model</code> , <code>jmatst_model</code> , and <code>paramnames</code> are not required. One or more of these models should include size or reproductive status in occasion $t-1$. Although this is optional input, it is recommended, and without it all vital rate model inputs (named <code>XX_model</code>) are required.
surv_model	A linear model predicting survival probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
obs_model	A linear model predicting sprouting or observation probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
size_model	A linear model predicting primary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
sizeb_model	A linear model predicting secondary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
sizec_model	A linear model predicting tertiary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
repst_model	A linear model predicting reproduction probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
fec_model	A linear model predicting fecundity. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
jsurv_model	A linear model predicting juvenile survival probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.

jobs_model	A linear model predicting juvenile sprouting or observation probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
jsize_model	A linear model predicting juvenile primary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
jsizeb_model	A linear model predicting juvenile secondary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
jsizec_model	A linear model predicting juvenile tertiary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
jrepst_model	A linear model predicting reproduction probability of a mature individual that was immature in time t . This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
jmatst_model	A linear model predicting maturity probability of an individual that was immature in time t . This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
paramnames	A data frame with three columns, the first describing all terms used in linear modeling, the second (must be called <code>mainparams</code>) giving the general model terms that will be used in matrix creation, and the third showing the equivalent terms used in modeling (must be named <code>modelparams</code>). Function <code>create_pm()</code> can be used to create a skeleton <code>paramnames</code> object, which can then be edited. Only required if <code>modelsuite</code> is not supplied.
inda	Can be a single value to use for individual covariate <code>a</code> in all matrices, a pair of values to use for times t and $t-1$ in historical matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to <code>NULL</code> .
indb	Can be a single value to use for individual covariate <code>b</code> in all matrices, a pair of values to use for times t and $t-1$ in historical matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to <code>NULL</code> .
indc	Can be a single value to use for individual covariate <code>c</code> in all matrices, a pair of values to use for times t and $t-1$ in historical matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to <code>NULL</code> .
surv_dev	A numeric value to be added to the y-intercept in the linear model for survival probability. Defaults to 0 .
obs_dev	A numeric value to be added to the y-intercept in the linear model for observation probability. Defaults to 0 .

size_dev	A numeric value to be added to the y-intercept in the linear model for primary size. Defaults to 0.
sizeb_dev	A numeric value to be added to the y-intercept in the linear model for secondary size. Defaults to 0.
sizec_dev	A numeric value to be added to the y-intercept in the linear model for tertiary size. Defaults to 0.
repst_dev	A numeric value to be added to the y-intercept in the linear model for probability of reproduction. Defaults to 0.
fec_dev	A numeric value to be added to the y-intercept in the linear model for fecundity. Defaults to 0.
jsurv_dev	A numeric value to be added to the y-intercept in the linear model for juvenile survival probability. Defaults to 0.
jobs_dev	A numeric value to be added to the y-intercept in the linear model for juvenile observation probability. Defaults to 0.
jsize_dev	A numeric value to be added to the y-intercept in the linear model for juvenile primary size. Defaults to 0.
sizeb_dev	A numeric value to be added to the y-intercept in the linear model for juvenile secondary size. Defaults to 0.
sizec_dev	A numeric value to be added to the y-intercept in the linear model for juvenile tertiary size. Defaults to 0.
jrepst_dev	A numeric value to be added to the y-intercept in the linear model for juvenile reproduction probability. Defaults to 0.
jmatst_dev	A numeric value to be added to the y-intercept in the linear model for juvenile maturity probability. Defaults to 0.
density	A numeric value indicating density value to use to propagate matrices. Only needed if density is an explanatory term used in one or more vital rate models. Defaults to NA.
repmo	A scalar multiplier of fecundity. Defaults to 1.
random.ind	A logical value denoting whether to treat individual covariate a as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random.indb	A logical value denoting whether to treat individual covariate b as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random.indc	A logical value denoting whether to treat individual covariate c as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
negfec	A logical value denoting whether fecundity values estimated to be negative should be reset to 0. Defaults to FALSE.
format	A string indicating whether to estimate matrices in ehrs format or deVries format. The latter adds one extra prior stage to account for the prior state of newborns. Defaults to ehrs format.

ipm_method	A string indicating what method to use to estimate size transition probabilities, if size is treated as continuous. Options include: "midpoint", which utilizes the midpoint method; and "CDF", which uses the cumulative distribution function. Defaults to "CDF".
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.
err_check	A logical value indicating whether to append matrices of vital rate probabilities associated with each matrix to the lefkoMat object generated. These matrices are developed internally and can be used for error checking (see element out in Value section below for details). Defaults to FALSE.
exp_tol	A numeric value used to indicate a maximum value to set exponents to in the core kernel to prevent numerical overflow. Defaults to 700.
theta_tol	A numeric value used to indicate a maximum value to theta as used in the negative binomial probability density kernel. Defaults to 100000000, but can be reset to other values during error checking.

Value

If all inputs are properly formatted, then this function will return an object of class `lefkoMat`, which is a list that holds the matrix projection model and all of its metadata. The structure has the following elements:

A	A list of full projection matrices in order of sorted patches and occasion times. All matrices output in R's <code>matrix</code> class.
U	A list of survival transition matrices sorted as in A. All matrices output in R's <code>matrix</code> class.
F	A list of fecundity matrices sorted as in A. All matrices output in R's <code>matrix</code> class.
hstages	A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs.
agestages	A data frame showing age-stage pairs. In this function, it is set to NA. Only used in output to function <code>aflefko2()</code> .
ahstages	A data frame detailing the characteristics of associated ahistorical stages, in the form of a modified stageframe that includes status as an entry stage through reproduction.
labels	A data frame giving the population, patch, and year of each matrix in order. In <code>flefko3()</code> , only one population may be analyzed at once.
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 <code>modelsuite</code> input.
prob_out	An optional element only added if <code>err_check = TRUE</code> . This is a list of vital rate probability matrices, with 7 columns in the order of survival, observation probability, reproduction probability, primary size transition probability, secondary size transition probability, tertiary size transition probability, and probability of juvenile transition to maturity.

`allstages` An optional element only added if `err_check = TRUE`. This is a data frame giving the values used to determine each matrix element capable of being estimated.

Notes

Unlike `rlefko2()`, `rlefko3()`, `arlefko2()`, and `rleslie()`, this function does not currently distinguish populations. Users wishing to use the same vital rate models across populations should label them as patches (though we do not advise this approach, as populations should typically be treated as statistically independent).

The default behavior of this function is to estimate fecundity with regards to transitions specified via associated fecundity multipliers in either `supplement` or `repmatrix`. If both of these fields are left empty, then fecundity will be estimated at full for all transitions leading from reproductive stages to immature and propagule stages. However, if a `supplement` is provided and a `repmatrix` is not, or if `repmatrix` is set to \emptyset , then only fecundity transitions noted in the `supplement` will be set to non-zero values. To use the default behavior of setting all reproductive stages to reproduce at full fecundity into immature and propagule stages, but also incorporate given or proxy survival transitions, input those given and proxy transitions through the `overwrite` option.

If used, the reproduction matrix (field `repmatrix`) may be supplied as either historical or ahistorical. If provided as ahistorical, then `flefko3()` will assume that all historical transitions involving stages noted for occasions t and $t+1$ should be set to the respective fecundity multipliers noted.

Users may at times wish to estimate MPMs using a dataset incorporating multiple patches or subpopulations, but without discriminating between those patches or subpopulations. Should the aim of analysis be a general MPM that does not distinguish these patches or subpopulations, the `modelsearch()` run should not include patch terms.

Input options including multiple variable names must be entered in the order of variables in occasion $t+1$, t , and $t-1$. Rearranging the order will lead to erroneous calculations, and will may lead to fatal errors.

The `ipm_method` function gives the option of using two different means of estimating the probability of size transition. The midpoint method ("`midpoint`") refers to the method in which the probability is estimated by first estimating the probability associated with transition from the exact size at the midpoint of the size class using the corresponding probability density function, and then multiplying that value by the bin width of the size class. Doak et al. 2021 (Ecological Monographs) noted that this method can produce biased results, with total size transitions associated with a specific size not totaling to 1.0 and even specific size transition probabilities capable of being estimated at values greater than 1.0. The alternative and default method, "`CDF`", uses the corresponding cumulative density function to estimate the probability of size transition as the cumulative probability of size transition at the greater limit of the size class minus the cumulative probability of size transition at the lower limit of the size class. This latter method avoids this bias. Note, however, that both methods are exact and unbiased for negative binomial and Poisson distributions.

Under the Gaussian and gamma size distributions, the number of estimated parameters may differ between the two `ipm_method` settings. Because the midpoint method has a tendency to incorporate upward bias in the estimation of size transition probabilities, it is more likely to yield non-zero values when the true probability is extremely close to 0. This will result in the `summary.lfkoMat` function yielding higher numbers of estimated parameters than the `ipm_method = "CDF"` yields in some cases.

Using the `err_check` option will produce a matrix of 7 columns, each characterizing a different vital rate. The product of each row yields an element in the associated U matrix. The number and order

of elements in each column of this matrix matches the associated matrix in column vector format. Use of this option is generally for the purposes of debugging code. Individual covariates are treated as categorical only if they are set as random terms. Fixed categorical individual covariates are currently not allowed. However, such terms may be supplied if the `modelsuite` option is set to a `vrn_input` object. In that case, the user should also set the logical random switch for the individual covariate to be used to TRUE (e.g., `random.indata = TRUE`).

See Also

```
fleeko2()
afleeko2()
arleeko2()
fleslie()
rleeko3()
rleeko2()
rleslie()
```

Examples

```
# Lathyrus example
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", "Sd1", "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)
obsvector <- c(0, 1, 0, 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)
immvector <- c(1, 1, 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)
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)

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

lathvertIn <- 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 = lathframeIn, stagesize = "sizea",
  censorcol = "Missing1988", censorkeep = NA, NAas0 = TRUE, censor = TRUE)
```

```

lathvertln$fece2 <- round(lathvertln$fece2)
lathvertln$fece1 <- round(lathvertln$fece1)
lathvertln$fece3 <- round(lathvertln$fece3)

lathmodelsln3 <- modelsearch(lathvertln, historical = TRUE,
  approach = "mixed", suite = "main",
  vitalrates = c("surv", "obs", "size", "repst", "fec"), juvestimate = "Sd1",
  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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "mat", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "Sd1", "rep", "rep"),
  stage1 = c("Sd", "rep", "Sd", "rep", "Sd", "mat", "mat"),
  eststage3 = c(NA, NA, NA, NA, "mat", NA, NA),
  eststage2 = c(NA, NA, NA, NA, "Sd1", NA, NA),
  eststage1 = c(NA, NA, NA, NA, "Sd1", NA, NA),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, 0.345, 0.054),
  type = c(1, 1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframeIn, historical = TRUE)

lathmat3ln <- flefko3(year = "all", patch = "all", stageframe = lathframeIn,
  modelsuite = lathmodelsln3, data = lathvertln, supplement = lathsupp3,
  reduce = FALSE)

summary(lathmat3ln)

#Cypripedium example using three size metrics for classification
data(cypdata)
sizevector.f <- c(0, 0, 0, 0, 0, 0, seq(1, 12, by = 1), seq(0, 9, by = 1),
  seq(0, 8, by = 1), seq(0, 7, by = 1), seq(0, 6, by = 1), seq(0, 5, by = 1),
  seq(0, 4, by = 1), seq(0, 3, by = 1), 0, 1, 2, 0, 1, 0,
  0, 0, 1, 0)
sizebvector.f <- c(0, 0, 0, 0, 0, 0, rep(0, 12), rep(1, 10), rep(2, 9),
  rep(3, 8), rep(4, 7), rep(5, 6), rep(6, 5), rep(7, 4), rep(8, 3), 9, 9, 10,
  0, 1, 1, 2)
sizecvector.f <- c(0, 0, 0, 0, 0, 0, rep(0, 12), rep(0, 10), rep(0, 9),
  rep(0, 8), rep(0, 7), rep(0, 6), rep(0, 5), rep(0, 4), 0, 0, 0, 0, 0, 0,
  1, 1, 1, 1)
stagevector.f <- c("DS", "P1", "P2", "P3", "Sd1", "Dorm", "V1 I0 D0",
  "V2 I0 D0", "V3 I0 D0", "V4 I0 D0", "V5 I0 D0", "V6 I0 D0", "V7 I0 D0",
  "V8 I0 D0", "V9 I0 D0", "V10 I0 D0", "V11 I0 D0", "V12 I0 D0", "V0 I1 D0",
  "V1 I1 D0", "V2 I1 D0", "V3 I1 D0", "V4 I1 D0", "V5 I1 D0", "V6 I1 D0",
  "V7 I1 D0", "V8 I1 D0", "V9 I1 D0", "V0 I2 D0", "V1 I2 D0", "V2 I2 D0",
  "V3 I2 D0", "V4 I2 D0", "V5 I2 D0", "V6 I2 D0", "V7 I2 D0", "V8 I2 D0",
  "V0 I3 D0", "V1 I3 D0", "V2 I3 D0", "V3 I3 D0", "V4 I3 D0", "V5 I3 D0",
  "V6 I3 D0", "V7 I3 D0", "V0 I4 D0", "V1 I4 D0", "V2 I4 D0", "V3 I4 D0",
  "V4 I4 D0", "V5 I4 D0", "V6 I4 D0", "V0 I5 D0", "V1 I5 D0", "V2 I5 D0",
  "V3 I5 D0", "V4 I5 D0", "V5 I5 D0", "V0 I6 D0", "V1 I6 D0", "V2 I6 D0",
  "V3 I6 D0", "V4 I6 D0", "V0 I7 D0", "V1 I7 D0", "V2 I7 D0", "V3 I7 D0",
  "V0 I8 D0", "V1 I8 D0", "V2 I8 D0", "V0 I9 D0", "V1 I9 D0", "V0 I10 D0",
  "V0 I0 D1", "V0 I1 D1", "V1 I1 D1", "V0 I2 D1")

```



```

type_t12 = c(1, 1, 2, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
  1, 1, 1),
stageframe = vertframe.f, historical = TRUE)

vert.mats.f3 <- flefko3(stageframe = vertframe.f, supplement = vertsupp3f,
  data = vert.data.f, modelsuite = vertmodels3f)
summary(vert.mats.f3)

```

fleslie

Create Function-based Age-based (Leslie) Matrix Projection Model

Description

Function `fleslie()` returns age-based (Leslie) MPMs corresponding to the patches and occasions given, including the associated component transition and fecundity matrices, data frames detailing the characteristics of the exact ages corresponding to rows and columns in estimated matrices, and a data frame characterizing the patch and occasion combinations corresponding to these matrices.

Usage

```

fleslie(
  year = "all",
  patch = "all",
  prebreeding = TRUE,
  data = NULL,
  modelsuite = NULL,
  surv_model = NULL,
  fec_model = NULL,
  paramnames = NULL,
  start_age = NA,
  last_age = NA,
  fecage_min = NA,
  fecage_max = NA,
  continue = TRUE,
  inda = NULL,
  indb = NULL,
  indc = NULL,
  surv_dev = 0,
  fec_dev = 0,
  density = NA,
  repmod = 1,
  random.inda = FALSE,
  random.indb = FALSE,
  random.indc = FALSE,
  negfec = FALSE,
  exp_tol = 700,

```

```

    theta_tol = 1e+08
  )

```

Arguments

year	A variable corresponding to observation occasion, or a set of such values, given in values associated with the year term used in linear model development. Defaults to "all", in which case matrices will be estimated for all occasions.
patch	A variable designating which patches or subpopulations will have matrices estimated. Defaults to "all", but can also be set to specific patch names or a vector thereof.
prebreeding	A logical value indicating whether the life history model is a pre-breeding model. Defaults to TRUE.
data	The historical vertical demographic data frame used to estimate vital rates (class <code>hfvdata</code>). The original data frame is generally required in order to initialize occasions and patches properly, and to assess the range of ages observed in the population. Not required if option <code>modelsuite</code> is set to a <code>vrn_input</code> object.
modelsuite	One of two optional lists. The first is an optional <code>lefkoMod</code> object holding the vital rate models. Alternatively, an object of class <code>vrn_input</code> may be provided. If given, then <code>surv_model</code> , <code>fec_model</code> , and <code>paramnames</code> are not required. No models should include size or reproductive status in any occasion, nor should they include any variable for occasion $t-1$. Note that the <code>modelsuite</code> must have been created from a <code>modelsearch()</code> run in which <code>vitalrates = c("surv", "fec")</code> and the <code>suite</code> option was set to either <code>age</code> or <code>cons</code> .
surv_model	A linear model predicting survival probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
fec_model	A linear model predicting fecundity. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing only the impacts of occasion t .
paramnames	A data frame with three columns, the first describing all terms used in linear modeling, the second (must be called <code>mainparams</code>) giving the general model terms that will be used in matrix creation, and the third showing the equivalent terms used in modeling (must be named <code>modelparams</code>). Function <code>create_pm()</code> can be used to create a skeleton <code>paramnames</code> object, which can then be edited. Only required if <code>modelsuite</code> is not supplied.
start_age	The age from which to start the matrix. Defaults to NA, in which case age 1 is used if <code>prebreeding = TRUE</code> , and age 0 is used if <code>prebreeding = FALSE</code> .
last_age	The final age to use in the matrix. Defaults to NA, in which case the highest age in the dataset is used.
fecage_min	The minimum age at which reproduction is possible. Defaults to NA, which is interpreted to mean that fecundity should be assessed starting in the minimum age observed in the dataset.

fecage_max	The maximum age at which reproduction is possible. Defaults to NA, which is interpreted to mean that fecundity should be assessed until the final observed age.
continue	A logical value designating whether to allow continued survival of individuals past the final age noted in the stageframe, using the demographic characteristics of the final age. Defaults to TRUE.
inda	Can be a single value to use for individual covariate a in all matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to NULL.
indb	Can be a single value to use for individual covariate b in all matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to NULL.
indc	Can be a single value to use for individual covariate c in all matrices, or a vector of such values corresponding to each occasion in the dataset. Defaults to NULL.
surv_dev	A numeric value to be added to the y-intercept in the linear model for survival probability. Defaults to 0.
fec_dev	A numeric value to be added to the y-intercept in the linear model for fecundity. Defaults to 0.
density	A numeric value indicating density value to use to propagate matrices. Only needed if density is an explanatory term used in linear models. Defaults to NA.
repmod	A scalar multiplier of fecundity. Defaults to 1.
random.inda	A logical value denoting whether to treat individual covariate a as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random.indb	A logical value denoting whether to treat individual covariate b as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random.indc	A logical value denoting whether to treat individual covariate c as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
negfec	A logical value denoting whether fecundity values estimated to be negative should be reset to 0. Defaults to FALSE.
exp_tol	A numeric value used to indicate a maximum value to set exponents to in the core kernel to prevent numerical overflow. Defaults to 700.
theta_tol	A numeric value used to indicate a maximum value to theta as used in the negative binomial probability density kernel. Defaults to 100000000, but can be reset to other values during error checking.

Value

If all inputs are properly formatted, then this function will return an object of class `lefkoMat`, which is a list that holds the matrix projection model and all of its metadata. Its structure has the following elements:

A	A list of full projection matrices in order of sorted patches and occasions. All matrices output in R's <code>matrix</code> class.
---	--

U	A list of survival transition matrices sorted as in A. All matrices output in R's matrix class.
F	A list of fecundity matrices sorted as in A. All matrices output in R's matrix class.
hstages	Set to NA for Leslie MPMs.
agestages	Set to NA for Leslie MPMs.
ahstages	A data frame detailing the characteristics of associated ages, in the form of a modified stageframe including reproduction status.
labels	A data frame giving the patch and year of each matrix in order. In fleslie(), only one population may be analyzed at once.
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.

Notes

Unlike [rlefko2\(\)](#), [rlefko3\(\)](#), [arlefko2\(\)](#), and [rleslie\(\)](#), this function does not currently distinguish populations.

This function will yield incorrect estimates if the models utilized incorporate state in occasion $t-1$, or any size or reproductive status terms.

Users may at times wish to estimate MPMs using a dataset incorporating multiple patches or subpopulations, but without discriminating between those patches or subpopulations. Should the aim of analysis be a general MPM that does not distinguish these patches or subpopulations, the [modelsearch\(\)](#) run should not include patch terms.

Input options including multiple variable names must be entered in the order of variables in occasion $t+1$ and t . Rearranging the order will lead to erroneous calculations, and may lead to fatal errors.

Care should be taken to match the random status of year and patch to the states of those variables within the modelsuite. If they do not match, then they will be treated as zeroes in vital rate estimation.

Individual covariates are treated as categorical only if they are set as random terms. Fixed categorical individual covariates are currently not allowed. However, such terms may be supplied if the modelsuite option is set to a `vrn_input` object. In that case, the user should also set the logical random switch for the individual covariate to be used to TRUE (e.g., `random.indc = TRUE`).

See Also

[flefko3\(\)](#)
[flefko2\(\)](#)
[aflefko2\(\)](#)
[arlefko2\(\)](#)
[rlefko3\(\)](#)
[rlefko2\(\)](#)
[rleslie\(\)](#)

Examples

```

data(lathyrus)

lathvert_base <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
  patchidcol = "SUBPLOT", individcol = "GENET", blocksize = 9,
  sizeacol = "Volume88", repstracol = "FCODE88", fecacol = "Intactseed88",
  deadacol = "Dead1988", censorcol = "Missing1988", censorkeep = NA,
  censor = TRUE, NRasRep = TRUE, N0asObs = TRUE)

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

lathvert_age <- subset(lathvert_base, firstseen > 1988)

lathmodels2_age <- modelsearch(lathvert_age, historical = FALSE,
  approach = "mixed", suite = "cons", bestfit = "AICc&k", age = "obsage",
  vitalrates = c("surv", "fec"), fecdist = "poisson", indiv = "individ",
  year = "year2", show.model.tables = TRUE, quiet = TRUE)

lathmat2fleslie <- fleslie(year = "all", data = lathvert_age,
  modelsuite = lathmodels2_age, fecage_min = 1)

summary(lathmat2fleslie)

```

f_projection3

Project Function-based Matrix Projection Model

Description

Function `f_projection3()` develops and projects function-based matrix models. Unlike `projection3()`, which uses matrices provided as input via already created `lefkoMat` objects, function `f_projection3()` creates matrices at each time step from vital rate models and parameter inputs provided. Projections may be deterministic or stochastic, and may be density dependent in either case. Replicates may also be produced.

Usage

```

f_projection3(
  format,
  prebreeding = TRUE,
  start_age = NA_integer_,
  last_age = NA_integer_,
  fecage_min = NA_integer_,
  fecage_max = NA_integer_,
  cont = TRUE,

```

```
stochastic = FALSE,
standardize = FALSE,
growthonly = TRUE,
repvalue = FALSE,
integeronly = FALSE,
substoch = 0L,
ipm_method = "CDF",
nreps = 1L,
times = 10000L,
repmo = 1,
exp_tol = 700,
theta_tol = 1e+08,
random_inda = FALSE,
random_indb = FALSE,
random_indc = FALSE,
err_check = FALSE,
quiet = FALSE,
data = NULL,
stageframe = NULL,
supplement = NULL,
repmatrix = NULL,
overwrite = NULL,
modelsuite = NULL,
paramnames = NULL,
year = NULL,
patch = NULL,
sp_density = NULL,
ind_terms = NULL,
dev_terms = NULL,
surv_model = NULL,
obs_model = NULL,
size_model = NULL,
sizeb_model = NULL,
sizec_model = NULL,
repst_model = NULL,
fec_model = NULL,
jsurv_model = NULL,
jobs_model = NULL,
jsize_model = NULL,
jsizeb_model = NULL,
jsizec_model = NULL,
jrepst_model = NULL,
jmatst_model = NULL,
start_vec = NULL,
start_frame = NULL,
tweights = NULL,
density = NULL,
density_vr = NULL
```

)

Arguments

format	An integer indicating the kind of function-based MPM to create. Possible choices include: 1, Ehrlen-format historical MPM; 2, deVries-format historical MPM; 3, ahistorical MPM; 4, age-by-stage MPM; and 5, Leslie (age-based) MPM.
prebreeding	A logical value indicating whether the life history model is a pre-breeding model. Only used in Leslie and age-by-stage MPMs. Defaults to TRUE.
start_age	The age from which to start the matrix. Defaults to NA, in which case age 1 is used if prebreeding = TRUE, and age 0 is used if prebreeding = FALSE.
last_age	The final age to use in the matrix. Defaults to NA, in which case the highest age in the dataset is used.
fecage_min	The minimum age at which reproduction is possible. Defaults to NA, which is interpreted to mean that fecundity should be assessed starting in the minimum age observed in the dataset.
fecage_max	The maximum age at which reproduction is possible. Defaults to NA, which is interpreted to mean that fecundity should be assessed until the final observed age.
cont	A logical value designating whether to allow continued survival of individuals past the final age noted in the stageframe, using the demographic characteristics of the final age. Defaults to TRUE.
stochastic	A logical value denoting whether to conduct a stochastic projection or a deterministic / cyclical projection.
standardize	A logical value denoting whether to re-standardize the population size to 1.0 at each occasion. Used in density-independent simulations in which it is more important to know the general trend in population growth than the explicit growth rate. Defaults to FALSE.
growthonly	A logical value indicating whether to produce only the projected population size at each occasion (TRUE), or also to produce vectors showing the stage distribution at each occasion (FALSE). Defaults to TRUE.
repvalue	A logical value indicating whether to calculate reproductive value vectors at each time step. Can only be set to TRUE if growthonly = FALSE. Setting to TRUE may dramatically increase the duration of calculations. Defaults to FALSE.
integeronly	A logical value indicating whether to round the number of individuals projected in each stage at each occasion to the nearest integer. Defaults to FALSE.
substoch	An integer value indicating whether to force survival- transition matrices to be substochastic in density dependent and density independent simulations. Defaults to 0, which does not enforce substochasticity. Alternatively, 1 forces all survival-transition elements to range from 0.0 to 1.0, and forces fecundity to be non-negative; and 2 forces all column rows in the survival-transition matrices to total no more than 1.0, in addition to the actions outlined for option 1. Both settings 1 and 2 change negative fecundity elements to 0.0.

ipm_method	A string indicating what method to use to estimate size transition probabilities, if size is treated as continuous. Options include: "midpoint", which utilizes the midpoint method; and "CDF", which uses the cumulative distribution function. Defaults to "CDF".
nreps	The number of replicate projections. Defaults to 1.
times	Number of occasions to iterate per replicate. Defaults to 10000.
repmo	A scalar multiplier of fecundity. Defaults to 1.
exp_tol	A numeric value used to indicate a maximum value to set exponents to in the core kernel to prevent numerical overflow. Defaults to 700.
theta_tol	A numeric value used to indicate a maximum value to theta as used in the negative binomial probability density kernel. Defaults to 100000000, but can be reset to other values during error checking.
random_inda	A logical value denoting whether to treat individual covariate a as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random_indb	A logical value denoting whether to treat individual covariate b as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
random_indc	A logical value denoting whether to treat individual covariate c as a random, categorical variable. Otherwise is treated as a fixed, numeric variable. Defaults to FALSE.
err_check	A logical value indicating whether to append extra output for debugging purposes. Defaults to FALSE.
quiet	A logical value indicating whether warning messages should be suppressed. Defaults to FALSE.
data	The historical vertical demographic data frame used to estimate vital rates (class hfvddata), which is required to initialize times and patches properly. Variable names should correspond to the naming conventions in verticalize3() and historicalize3() .
stageframe	An object of class stageframe. These objects are generated by function sf_create() , and include information on the size, observation status, propagule status, reproduction status, immaturity status, maturity status, stage group, size bin widths, and other key characteristics of each ahistorical stage. Required for all MPM formats except Leslie MPMs.
supplement	An optional data frame of class lefkoSD that provides supplemental data that should be incorporated into the MPM. Three kinds of data may be integrated this way: transitions to be estimated via the use of proxy transitions, transition overwrites from the literature or supplemental studies, and transition multipliers for survival and fecundity. This data frame should be produced using the supplemental() function. Can be used in place of or in addition to an overwrite table (see overwrite below) and a reproduction matrix (see repmatrix below).
repmatrix	An optional reproduction matrix. This matrix is composed mostly of 0s, with non-zero entries acting as element identifiers and multipliers for fecundity (with

1 equaling full fecundity). If left blank, and no supplement is provided, then `flecko3()` will assume that all stages marked as reproductive produce offspring at 1x that of estimated fecundity, and that offspring production will yield the first stage noted as propagule or immature. May be the dimensions of either a historical or an ahistorical matrix. If the latter, then all stages will be used in occasion $t-1$ for each suggested ahistorical transition.

overwrite	An optional data frame developed with the <code>overwrite()</code> function describing transitions to be overwritten either with given values or with other estimated transitions. Note that this function supplements overwrite data provided in supplement.
modelsuite	A <code>lefkMod</code> object, at minimum with all required best-fit vital rate models and a <code>paramnames</code> data frame, and following the naming conventions used in this package. If given, then <code>surv_model</code> , <code>obs_model</code> , <code>size_model</code> , <code>sizeb_model</code> , <code>sizec_model</code> , <code>repst_model</code> , <code>fec_model</code> , <code>jsurv_model</code> , <code>jobs_model</code> , <code>jsize_model</code> , <code>jsizeb_model</code> , <code>jsizec_model</code> , <code>jrepst_model</code> , <code>jmatst_model</code> , <code>paramnames</code> , <code>yearcol</code> , and <code>patchcol</code> are not required. Although this is optional input, it is recommended, and without it separate vital rate model inputs (named <code>XX_model</code>) are required.
paramnames	A data frame with three columns, the first describing all terms used in linear modeling, the second (must be called <code>mainparams</code>) giving the general model terms that will be used in matrix creation, and the third showing the equivalent terms used in modeling (must be named <code>modelparams</code>). Function <code>create_pm()</code> can be used to create a skeleton <code>paramnames</code> object, which can then be edited. Only required if <code>modelsuite</code> is not supplied.
year	Either a single integer value corresponding to the year to project, or a vector of <code>times</code> elements with the year to use at each time step. Defaults to NA, in which the first year in the set of years in the dataset is projected. If a vector shorter than <code>times</code> is supplied, then this vector will be cycled.
patch	A value of NA, a single string value corresponding to the patch to project, or a vector of <code>times</code> elements with the patch to use at each time step. If a vector shorter than <code>times</code> is supplied, then this vector will be cycled. Note that this function currently does not handle multiple projections for different patches in the same run.
sp_density	Either a single numeric value of spatial density to use in vital rate models in all time steps, or a vector of <code>times</code> elements of such numeric values. Defaults to NA.
ind_terms	An optional data frame with 3 columns and <code>times</code> rows giving the values of individual covariates a, b, and c, respectively, for each projected time. Unused terms must be set to 0 (use of NA will produce errors.)
dev_terms	An optional data frame with 14 columns and <code>times</code> rows showing the values of the deviation terms to be added to each linear vital rate. The column order should be: 1: survival, 2: observation, 3: primary size, 4: secondary size, 5: tertiary size, 6: reproduction, 7: fecundity, 8: juvenile survival, 9: juvenile observation, 10: juvenile primary size, 11: juvenile secondary size, 12: juvenile tertiary size, 13: juvenile reproduction, and 14: juvenile maturity transition. Unused terms must be set to 0 (use of NA will produce errors.)

<code>surv_model</code>	A linear model predicting survival probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>obs_model</code>	A linear model predicting sprouting or observation probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>size_model</code>	A linear model predicting primary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>sizeb_model</code>	A linear model predicting secondary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>sizec_model</code>	A linear model predicting tertiary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>repst_model</code>	A linear model predicting reproduction probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>fec_model</code>	A linear model predicting fecundity. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>jsurv_model</code>	A linear model predicting juvenile survival probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>jobs_model</code>	A linear model predicting juvenile sprouting or observation probability. This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>jsize_model</code>	A linear model predicting juvenile primary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>jsizeb_model</code>	A linear model predicting juvenile secondary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
<code>jsizec_model</code>	A linear model predicting juvenile tertiary size. This can be a model of class <code>glm</code> , <code>glmer</code> , <code>glmmTMB</code> , <code>zeroinfl</code> , <code>vglm</code> , <code>lm</code> , or <code>lmer</code> . Ignored if <code>modelsuite</code> is

	provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
jrepst_model	A linear model predicting reproduction probability of a mature individual that was immature in time t . This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
jmatst_model	A linear model predicting maturity probability of an individual that was immature in time t . This can be a model of class <code>glm</code> or <code>glmer</code> , and requires a predicted binomial variable under a logit link. Ignored if <code>modelsuite</code> is provided. This model must have been developed in a modeling exercise testing the impacts of occasions t and $t-1$.
start_vec	An optional numeric vector denoting the starting stage distribution for the projection. Defaults to a single individual of each stage.
start_frame	An optional data frame characterizing stages, age-stages, or stage-pairs that should be set to non-zero values in the starting vector, and what those values should be. Can only be used with <code>lefkMat</code> objects.
tweights	An optional numeric vector denoting the probabilistic weightings of year terms. Defaults to equal weighting among occasions.
density	An optional data frame describing the matrix elements that will be subject to density dependence, and the exact kind of density dependence that they will be subject to. The data frame used should be an object of class <code>lefkDens</code> , which is the output from function <code>density_input()</code> .
density_vr	An optional data frame describing density dependence relationships in vital rates, if such relationships are to be assumed. The data frame must be of class <code>lefkDensVR</code> , which is the output from the function <code>density_vr()</code> .

Value

A list of class `lefkProj`, which always includes the first three elements of the following, and also includes the remaining elements below when a `lefkMat` object is used as input:

projection	A list of lists of matrices showing the total number of individuals per stage per occasion. The first list corresponds to each pop-patch followed by each population (this top-level list is a single element in <code>f_projection3()</code>). The inner list corresponds to replicates within each pop-patch or population.
stage_dist	A list of lists of the actual stage distribution in each occasion in each replicate in each pop-patch or population. The list structure is the same as in <code>projection3()</code> .
rep_value	A list of lists of the actual reproductive value in each occasion in each replicate in each pop-patch or population. The list structure is the same as in <code>projection3()</code> .
pop_size	A list of matrices showing the total population size in each occasion per replicate (row within data frame) per pop-patch or population (list element). Only a single pop-patch or population is allowed in <code>f_projection3()</code>
labels	A data frame showing the order of populations and patches in item projection.
ahstages	The original stageframe used in the study.

<code>hstages</code>	A data frame showing the order of historical stage pairs.
<code>agestages</code>	A data frame showing the order of age-stage pairs.
<code>labels</code>	A short data frame indicating the population (always 1, and patch (either the numeric index of the single chosen patch, or 1 in all other cases).)
<code>control</code>	A short vector indicating the number of replicates and the number of occasions projected per replicate.
<code>density</code>	The data frame input under the <code>density</code> option. Only provided if input by the user.
<code>density_vr</code>	The data frame input under the <code>density_vr</code> option. Only provided if input by the user.

Notes

Population projection can be a very time-consuming activity, and it is most time-consuming when matrices need to be created at each time step. We have created this function to be as quick as possible, but some options will slow the analysis down. First, the `err_check` option should always be set to `FALSE`, as the added created output will not only slow the analysis down but also potentially crash the memory if matrices are large enough. Second, the `repvalue` option should be set to `FALSE` unless reproductive values are genuinely needed, since this step requires concurrent backward projection and so in some cases may double total run time. Finally, if the only needed data is the total population size and actual age/stage structure at each time step, then setting `growthonly = TRUE` will yield the quickest possible run time.

Projections with large matrices may take a long time to run. To assess the likely running time, try using a low number of iterations on a single replicate first. For example, set `nreps = 1` and `times = 10` for a trial run. If a full run is set and takes too long, press the `STOP` button in RStudio to cancel the projection run, or click `esc`.

This function currently allows three forms of density dependence. The first modifies matrix elements on the basis of the input provided in option `density`, and so alters matrix elements once the matrix has already been created. The second form alters the vital rates estimated, and so estimates matrix elements using vital rate values already modified by `density`. This second form uses the input provided in option `density_vr`. These two forms of density dependence utilize the projected population size at some time to make these alterations. The third form of density dependence also alters the vital rates, but using spatial density supplied via option `sp_density` and only in vital rates in which spatial density is included as a fixed factor in the associated vital rate model.

Consistently positive population growth can quickly lead to population size numbers larger than can be handled computationally. In that circumstance, a continuously rising population size will suddenly become `NaN` for the remainder of the projection.

This function does not reduce the dimensionality of matrices developed for projection.

See Also

[projection3\(\)](#)
[flefko3\(\)](#)
[flefko2\(\)](#)
[aflefko2\(\)](#)
[fleslie\(\)](#)

Examples

```

# Lathyrus projection example with historical matrices
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", "Sd1", "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)
obsvector <- c(0, 1, 0, 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)
immvector <- c(1, 1, 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)
indataset <- c(0, 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)

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)

lathmodelsln3 <- modelsearch(lathvertln, historical = TRUE,
approach = "mixed", suite = "main",
vitalrates = c("surv", "obs", "size", "repst", "fec"), juvestimate = "Sd1",
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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "mat", "Sd", "Sd1"),
stage2 = c("Sd", "Sd", "Sd", "Sd", "Sd1", "rep", "rep"),
stage1 = c("Sd", "rep", "Sd", "rep", "Sd", "mat", "mat"),
eststage3 = c(NA, NA, NA, NA, "mat", NA, NA),
eststage2 = c(NA, NA, NA, NA, "Sd1", NA, NA),
eststage1 = c(NA, NA, NA, NA, "Sd1", NA, NA),
givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, NA, 0.345, 0.054),
type = c(1, 1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1, 1))

```

```

stageframe = lathframe1n, historical = TRUE)

# While we do not use MPMs to initialize f_projections3(), we do use MPMs to
# initialize functions start_input() and density_input().
lathmat3ln <- flefko3(year = "all", patch = "all", stageframe = lathframe1n,
  modelsuite = lathmodels1n3, data = lathvert1n, supplement = lathsupp3,
  reduce = FALSE)

e3m_sv <- start_input(lathmat3ln, stage2 = "Sd", stage1 = "Sd", value = 1000)

e3d <- density_input(lathmat3ln, stage3 = c("Sd", "Sd1"),
  stage2 = c("rep", "rep"), stage1 = c("all", "all"), style = 1,
  time_delay = 1, alpha = 1, beta = 0, type = c(2, 2), type_t12 = c(1, 1))

trial7 <- f_projection3(format = 1, data = lathvert1n,
  modelsuite = lathmodels1n3, stageframe = lathframe1n, nreps = 2,
  times = 1000, stochastic = TRUE, standardize = FALSE, growthonly = TRUE,
  integeronly = FALSE, substoch = 0)
summary(trial7)

# Now with density dependence and a set start vector
trial7a <- f_projection3(format = 1, data = lathvert1n,
  modelsuite = lathmodels1n3, stageframe = lathframe1n, nreps = 2,
  times = 1000, stochastic = TRUE, standardize = FALSE, growthonly = TRUE,
  integeronly = FALSE, substoch = 0, sp_density = 0, start_frame = e3m_sv,
  density = e3d)
summary(trial7a)

```

hfv_qc

Check Quality and Distributions of hfv Datasets

Description

Function `hfv_qc()` tests the overall quality of hfv datasets, and also runs a series of tests to assess which statistical distributions match the variables within these datasets. The input format is equivalent to the input format of function `modelsearch()`, allowing users to assess vital rate variable distributions assuming the same internal dataset subsetting used by the latter function and simply copy and pasting the parameter options from one function to the other.

Usage

```

hfv_qc(
  data,
  stageframe = NULL,
  historical = TRUE,
  suite = "size",
  vitalrates = c("surv", "size", "fec"),

```

```

surv = c("alive3", "alive2", "alive1"),
obs = c("obsstatus3", "obsstatus2", "obsstatus1"),
size = c("sizea3", "sizea2", "sizea1"),
sizeb = c(NA, NA, NA),
sizec = c(NA, NA, NA),
repst = c("repstatus3", "repstatus2", "repstatus1"),
fec = c("feca3", "feca2", "feca1"),
stage = c("stage3", "stage2", "stage1"),
matstat = c("matstatus3", "matstatus2", "matstatus1"),
indiv = "individ",
patch = NA,
year = "year2",
density = NA,
patch.as.random = TRUE,
year.as.random = TRUE,
juvestimate = NA,
juvsize = FALSE,
fectime = 2,
censor = NA,
age = NA,
indcova = NA,
indcovb = NA,
indcovc = NA,
random.indcova = FALSE,
random.indcovb = FALSE,
random.indcovc = FALSE,
test.group = FALSE,
...
)

```

Arguments

<code>data</code>	The vertical dataset to be used for analysis. This dataset should be of class <code>hfvdata</code> , but can also be a data frame formatted similarly to the output format provided by functions <code>verticalize3()</code> or <code>historicalize3()</code> , as long as all needed variables are properly designated.
<code>stageframe</code>	The stageframe characterizing the life history model used. Optional unless <code>test.group = TRUE</code> , in which case it is required. Defaults to <code>NULL</code> .
<code>historical</code>	A logical variable denoting whether to assess the effects of state in occasion $t-1$, in addition to state in occasion t . Defaults to <code>TRUE</code> .
<code>suite</code>	This describes the global model for each vital rate estimation, and has the following possible values: <code>full</code> , includes main effects and all two-way interactions of size and reproductive status; <code>main</code> , includes main effects only of size and reproductive status; <code>size</code> , includes only size (also interactions between size in historical model); <code>rep</code> , includes only reproductive status (also interactions between status in historical model); <code>age</code> , all vital rates estimated with age and y-intercepts only; <code>cons</code> , all vital rates estimated only as y-intercepts. Defaults to <code>size</code> .

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). May also be set to vitalrates = "leslie", which is equivalent to setting c("surv", "fec") for a Leslie MPM. This choice also determines how internal data subsetting for vital rate model estimation will work. Defaults to c("surv", "size", "fec").
surv	A vector indicating the variable names coding for status as alive or dead in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("alive3", "alive2", "alive1").
obs	A vector indicating the variable names coding for observation status in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("obsstatus3", "obsstatus2", "obsstatus1").
size	A vector indicating the variable names coding for the primary size variable on occasions $t+1$, t , and $t-1$, respectively. Defaults to c("sizea3", "sizea2", "sizea1").
sizeb	A vector indicating the variable names coding for the secondary size variable on occasions $t+1$, t , and $t-1$, respectively. Defaults to c(NA, NA, NA), in which case sizeb is not used.
sizec	A vector indicating the variable names coding for the tertiary size variable on occasions $t+1$, t , and $t-1$, respectively. Defaults to c(NA, NA, NA), in which case sizec is not used.
repst	A vector indicating the variable names coding for reproductive status in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("repstatus3", "repstatus2", "repstatus1").
fec	A vector indicating the variable names coding for fecundity in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("feca3", "feca2", "feca1").
stage	A vector indicating the variable names coding for stage in occasions $t+1$, t , and $t-1$. Defaults to c("stage3", "stage2", "stage1").
matstat	A vector indicating the variable names coding for maturity status in occasions $t+1$, t , and $t-1$. Defaults to c("matstatus3", "matstatus2", "matstatus1").
indiv	A text value indicating the variable name coding individual identity. Defaults to "individ".
patch	A text value indicating the variable name coding for patch, where patches are defined as permanent subgroups within the study population. Defaults to NA.
year	A text value indicating the variable coding for observation occasion t . Defaults to year2.
density	A text value indicating the name of the variable coding for spatial density, should the user wish to test spatial density as a fixed factor affecting vital rates. Defaults to NA.
patch.as.random	If set to TRUE and approach = "mixed", 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.

<code>year.as.random</code>	If set to TRUE and <code>approach = "mixed"</code> , 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 <code>approach</code> lead to year not being included in modeling. Defaults to TRUE.
<code>juvestimate</code>	An optional variable denoting the stage name of the juvenile stage in the vertical dataset. If not NA, and <code>stage</code> is also given (see below), then vital rates listed in <code>vitalrates</code> other than <code>fec</code> will also be estimated from the juvenile stage to all adult stages. Defaults to NA, in which case juvenile vital rates are not estimated.
<code>juvsize</code>	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 <code>juvestimate</code> does not equal NA.
<code>fectime</code>	A variable indicating which year of fecundity to use as the response term in fecundity models. Options include 2, which refers to occasion t , and 3, which refers to occasion $t+1$. Defaults to 2.
<code>censor</code>	A vector denoting the names of censoring variables in the dataset, in order from occasion $t+1$, followed by occasion t , and lastly followed by occasion $t-1$. Defaults to NA.
<code>age</code>	Designates the name of the variable corresponding to age in time t in the vertical dataset. Defaults to NA, in which case age is not included in linear models. Should only be used if building Leslie or age x stage matrices.
<code>indcova</code>	Vector designating the names in occasions $t+1$, t , and $t-1$ of an individual covariate. Defaults to NA.
<code>indcovb</code>	Vector designating the names in occasions $t+1$, t , and $t-1$ of a second individual covariate. Defaults to NA.
<code>indcovc</code>	Vector designating the names in occasions $t+1$, t , and $t-1$ of a third individual covariate. Defaults to NA.
<code>random.indcova</code>	A logical value indicating whether <code>indcova</code> should be treated as a random categorical factor, rather than as a fixed factor. Defaults to FALSE.
<code>random.indcovb</code>	A logical value indicating whether <code>indcovb</code> should be treated as a random categorical factor, rather than as a fixed factor. Defaults to FALSE.
<code>random.indcovc</code>	A logical value indicating whether <code>indcovc</code> should be treated as a random categorical factor, rather than as a fixed factor. Defaults to FALSE.
<code>test.group</code>	A logical value indicating whether to include the group variable from the input <code>stageframe</code> as a fixed categorical variable in linear models. Defaults to FALSE.
<code>...</code>	Other parameters.

Value

This function yields text output describing the subsets to be used in linear vital rate modeling. No value or object is returned.

Notes

This function is meant to handle input as would be supplied to function `modelsearch()`. To use most easily, users may copy all input parameters from a call to function `modelsearch()`, and paste directly within this function. The exact subsets used in the `modelsearch()` run will also be created here.

Examples

```

# Lathyrus example
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", "Sd1", "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, 0, 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)
matvector <- c(0, 0, 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)
propvector <- c(1, 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)
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)

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)

hfv_qc(lathvertln, historical = TRUE, suite = "main",
vitalrates = c("surv", "obs", "size", "repst", "fec"), juvestimate = "Sd1",
indiv = "individ", patch = "patchid", year = "year2", year.as.random = TRUE,
patch.as.random = TRUE)

lathmodelsln3 <- modelsearch(lathvertln, historical = TRUE,
approach = "mixed", suite = "main",
vitalrates = c("surv", "obs", "size", "repst", "fec"), juvestimate = "Sd1",
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)

# Here we use supplemental() to provide overwrite and reproductive info
lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "mat", "Sd", "Sd1"),
stage2 = c("Sd", "Sd", "Sd", "Sd", "Sd1", "rep", "rep"),
stage1 = c("Sd", "rep", "Sd", "rep", "Sd", "mat", "mat"),

```

```

eststage3 = c(NA, NA, NA, NA, "mat", NA, NA),
eststage2 = c(NA, NA, NA, NA, "Sd1", NA, NA),
eststage1 = c(NA, NA, NA, NA, "Sd1", NA, NA),
givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, NA, 0.345, 0.054),
type = c(1, 1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe1n, historical = TRUE)

lathmat3ln <- flfko3(year = "all", patch = "all", stageframe = lathframe1n,
  modelsuite = lathmodels1n3, data = lathvert1n, supplement = lathsupp3,
  reduce = FALSE)

summary(lathmat3ln)

```

historicalize3	<i>Create Historical Vertical Data Frame from Ahistorical Vertical Data Frame</i>
----------------	---

Description

Function `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 `hfvddata` 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,
  fec2col = 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,
NOasObs = FALSE,
prebreeding = TRUE,
age_offset = 0,
reduce = TRUE,
quiet = FALSE
)

```

Arguments

<code>data</code>	The horizontal data file.
<code>popidcol</code>	A variable name or column number corresponding to the identity of the population for each individual.
<code>patchidcol</code>	A variable name or column number corresponding to the identity of the patch or subpopulation for each individual, if patches have been designated within populations.
<code>individcol</code>	A variable name or column number corresponding to the unique identity of each individual.

year2col	A variable name or column number corresponding to occasion t (year or time).
year3col	A variable name or column number corresponding to occasion $t+1$ (year or time).
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 occasion t .
sizea3col	A variable name or column number corresponding to the primary size entry in occasion $t+1$.
sizeb2col	A variable name or column number corresponding to the secondary size entry in occasion t .
sizeb3col	A variable name or column number corresponding to the secondary size entry in occasion $t+1$.
sizec2col	A variable name or column number corresponding to the tertiary size entry in occasion t .
sizec3col	A variable name or column number corresponding to the tertiary size entry in occasion $t+1$.
repstra2col	A variable name or column number corresponding to the production of reproductive structures, such as flowers, in occasion t . This can be binomial or count data, and is used 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 occasion $t+1$. This can be binomial or count data, and is used 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 occasion 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 occasion $t+1$. This can be binomial or count data.
feca2col	A variable name or column number corresponding to fecundity in occasion t . This may represent egg counts, fruit counts, seed production, etc.
feca3col	A variable name or column number corresponding to fecundity in occasion $t+1$. This may represent egg counts, fruit counts, seed production, etc.
fecb2col	A second variable name or column number corresponding to fecundity in occasion t . This may represent egg counts, fruit counts, seed production, etc.
fecb3col	A second variable name or column number corresponding to fecundity in occasion $t+1$. 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 occasion t .
indcova3col	A variable name or column number corresponding to an individual covariate to be used in analysis, in occasion $t+1$.

indcovb2col	A variable name or column number corresponding to a second individual covariate to be used in analysis, in occasion t .
indcovb3col	A variable name or column number corresponding to a second individual covariate to be used in analysis, in occasion $t+1$.
indcovc2col	A variable name or column number corresponding to a third individual covariate to be used in analysis, in occasion t .
indcovc3col	A variable name or column number corresponding to a third individual covariate to be used in analysis, in occasion $t+1$.
alive2col	A variable name or column number that provides information on whether an individual is alive in occasion 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 occasion $t+1$. 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 occasion t . If used, dead status must be designated as binomial (living = 0, dead = 1).
dead3col	A variable name or column number that provides information on whether an individual is dead in occasion $t+1$. If used, dead status must be designated as binomial (living = 0, dead = 1).
obs2col	A variable name or column number providing information on whether an individual is in an observable stage in occasion 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 occasion $t+1$. 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 occasion t . If used, observation status must be designated as binomial (observed = 0, not observed = 1).
nonobs3col	A variable name or column number providing information on whether an individual is in an unobservable stage in occasion $t+1$. If used, observation status must be designated as binomial (observed = 0, not observed = 1).
repstrrel	This is a scalar multiplier making 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 making 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 occasion t .
stage3col	Optional variable name or column number corresponding to life history stage in occasion $t+1$.

juv2col	A variable name or column number that marks individuals in immature stages in occasion t . Function <code>historicalize3()</code> 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 occasion $t+1$. Function <code>historicalize3()</code> 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 <code>stage2col</code> 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 <code>sizea</code> , <code>sizeb</code> , <code>sizec</code> , <code>sizeab</code> , <code>sizebc</code> , <code>sizeac</code> , <code>sizeabc</code> , or <code>sizeadded</code> , depending on which size variable within the input dataset is chosen. Note that the variable(s) chosen should be presented in the order of the primary, secondary, and tertiary variables in the stageframe input with <code>stageassign</code> . For example, choosing <code>sizeb</code> assumes that this size variable in the dataset is the primary variable in the stageframe.
censor	A logical variable determining whether the output data should be censored using the variable defined in <code>censorcol</code> . 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 <code>xcol</code> and <code>ycol</code> . 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 <code>modelsearch()</code> , 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 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 <code>stageassign</code> is set to a valid stageframe. Defaults to FALSE.
NOasObs	If TRUE, then will treat individuals that are interpreted as not observed in the dataset as though they were observed during stage assignment. This can be useful when a MPM is desired without separation of observable and unobservable stages. Only used if <code>stageassign</code> is set to a stageframe. Defaults to FALSE.
prebreeding	A logical term indicating whether the life history model is pre-breeding. If so, then 1 is added to all ages. Defaults to TRUE.
age_offset	A number to add automatically to all values of age at time t . Defaults to 0.

reduce	A logical variable determining whether unused variables and some invariant state variables should be removed from the output dataset. Defaults to TRUE.
quiet	A logical variable indicating whether to silence warnings. Defaults to FALSE.

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 occasions $t-1$ and $t+1$ gives preference to condition in occasion t within the input dataset. Conflicts in condition in input datasets that have both occasions t and $t+1$ listed per row are resolved by using condition in occasion t .

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 in occasion t .
firstseen	Occasion of first observation.
lastseen	Occasion of last observation.
obsage	Observed age in occasion t , assuming first observation corresponds to age = 0.
obs lifespan	Observed lifespan, given as <code>lastseen - firstseen + 1</code> .
xpos1, xpos2, xpos3	X position in Cartesian space in occasions $t-1$, t , and $t+1$, respectively, if provided.
ypos1, ypos2, ypos3	Y position in Cartesian space in occasions $t-1$, t , and $t+1$, respectively, if provided.
sizea1, sizea2, sizea3	Main size measurement in occasions $t-1$, t , and $t+1$, respectively.
sizeb1, sizeb2, sizeb3	Secondary size measurement in occasions $t-1$, t , and $t+1$, respectively.
sizec1, sizec2, sizec3	Tertiary size measurement in occasions $t-1$, t , and $t+1$, respectively.
size1added, size2added, size3added	Sum of primary, secondary, and tertiary size measurements in occasions $t-1$, t , and $t+1$, respectively.
repstra1, repstra2, repstra3	Main numbers of reproductive structures in occasions $t-1$, t , and $t+1$, respectively.
repstrb1, repstrb2, repstrb3	Secondary numbers of reproductive structures in occasions $t-1$, t , and $t+1$, respectively.

repstr1added, repstr2added, repstr3added	Sum of primary and secondary reproductive structures in occasions $t-1$, t , and $t+1$, respectively.
feca1, feca2, feca3	Main numbers of offspring in occasions $t-1$, t , and $t+1$, respectively.
fecb1, fecb2, fecb3	Secondary numbers of offspring in occasions $t-1$, t , and $t+1$, respectively.
fec1added, fec2added, fec3added	Sum of primary and secondary fecundity in occasions $t-1$, t , and $t+1$, respectively.
censor1, censor2, censor3	Censor status values in occasions $t-1$, t , and $t+1$, respectively.
juvgiven1, juvgiven2, juvgiven3	Binomial variable indicating whether individual is juvenile in occasions $t-1$, t , and $t+1$. Only given if juvcol is provided.
obsstatus1, obsstatus2, obsstatus3	Binomial observation status in occasions $t-1$, t , and $t+1$, respectively.
repstatus1, repstatus2, repstatus3	Binomial reproductive status in occasions $t-1$, t , and $t+1$, respectively.
fecstatus1, fecstatus2, fecstatus3	Binomial offspring production status in occasions $t-1$, t , and $t+1$, respectively.
matstatus1, matstatus2, matstatus3	Binomial maturity status in occasions $t-1$, t , and $t+1$, respectively.
alive1, alive2, alive3	Binomial status as alive in occasions $t-1$, t , and $t+1$, respectively.
density	Density of individuals per unit designated in spacing. Only given if spacing is not NA.

Notes

Warnings that some individuals occur in state combinations that do not match any stages in the stageframe used to assign stages, and that some individuals match characteristics of several stages in the stageframe, are common when first working with a dataset. Typically, these situations can be identified as NoMatch entries in stage3, although such entries may crop up in stage1 and stage2, as well. In some cases, these warnings will arise with no concurrent NoMatch entries. These are important warnings and suggest that there is likely a problem with the stageframe. The most common such problems are: 1) stages have significant overlap in characteristics, with the most common being overlapping size bins caused by erroneous definitions of size bin halfwidths; and 2) some individuals exist in states not defined within the stageframe.

In some datasets with unobservable stages, observation status (obsstatus) might not be inferred properly if a single size variable is used that does not yield sizes greater than 0 in all cases in which individuals were observed. Such situations may arise, for example, in plants when leaf number is the dominant size variable used, but individuals occasionally occur with inflorescences but no leaves. In this instances, it helps to mark related variables as sizeb and sizec, because observation status will be interpreted in relation to all 3 size variables. Alternatively, observation status may be input via obs2col and obs3col to force computation with given values (although this requires

all instances of observation and non-observation to be known and coded ahead of time). Further analysis can then utilize only a single size variable, of the user's choosing. Similar issues can arise in reproductive status (repstatus).

Juvenile designation should only be used when juveniles fall outside of the size classification scheme used in determining stages. If juveniles are to be size classified along the size spectrum that adults also fall on, then it is best to treat juveniles as mature but not reproductive.

Care should be taken to avoid variables with negative values indicating size, fecundity, or reproductive or observation status. Negative values can be interpreted in different ways, typically reflecting estimation through other algorithms rather than actual measured data. Variables holding negative values can conflict with data management algorithms in ways that are difficult to predict.

Unusual errors (e.g. "Error in pjf...") may occur in cases where the variables are improperly passed, or where seemingly numeric variables include text and so get automatically converted to string variables.

Density estimation is performed as a count of individuals alive and within the radius specified in spacing of the respective individual at some point in time.

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, 0, 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, 0, 1, 1, 1, 1, 1)
binvec <- c(0, 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",
  fecal2col = "Pod.2", fecal3col = "Pod.3", repstrrel = 2,
  stageassign = cypframe_raw, stagesize = "sizeadded", censorcol = "censor",
  censor = FALSE, NAas0 = TRUE, NRasRep = TRUE, reduce = TRUE)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
```

```

      "rep"),
      eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
      eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
      givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
      multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
      type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
      stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v2, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix2r$A[[intersect(which(cypmatrix2r$labels$patch == "A"),
  which(cypmatrix2r$labels$year2 == 2004))]]

lambda3(cypmatrix2r)

```

 hist_null

Create Historical MPMs Assuming No Influence of Individual History

Description

Function `hist_null()` uses ahistorical MPMs to create the equivalent MPMs in the structure of historical MPMs. These MPMs have the same dimensions and stage structure of hMPMs but assume no influence of individual history, and so can be compared to actual hMPMs.

Usage

```
hist_null(mpm, format = 1, err_check = FALSE)
```

Arguments

<code>mpm</code>	An ahistorical MPM of class <code>lefkoMat</code> .
<code>format</code>	An integer stipulating whether historical matrices should be produced in Ehrlen format (1) or deVries format (2).
<code>err_check</code>	A logical value indicating whether to output the main index used to sort elements in the matrices.

Value

An object of class `lefkoMat`, with the same list structure as the input object, but with A, U, and F elements replaced with lists of historically-structured matrices, and with element `hstages` changed from NA to an index of stage pairs corresponding to the rows and columns of the new matrices. If `err_check = TRUE`, then a data frame showing the values used to determine element index values is also exported.

Examples

```

sizevector <- c(1, 1, 2, 3)
stagevector <- c("Sd1", "Veg", "SmFlo", "LFlo")
repvector <- c(0, 0, 1, 1)
obsvector <- c(1, 1, 1, 1)
matvector <- c(0, 1, 1, 1)
immvector <- c(1, 0, 0, 0)
propvector <- c(0, 0, 0, 0)
indataset <- c(1, 1, 1, 1)
binvec <- c(0.5, 0.5, 0.5, 0.5)

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

# POPN C 2003-2004
XC3 <- matrix(c(0, 0, 1.74, 1.74,
0.208333333, 0, 0, 0.057142857,
0.041666667, 0.076923077, 0, 0,
0.083333333, 0.076923077, 0.066666667, 0.028571429), 4, 4, byrow = TRUE)

# 2004-2005
XC4 <- matrix(c(0, 0, 0.3, 0.6,
0.32183908, 0.142857143, 0, 0,
0.16091954, 0.285714286, 0, 0,
0.252873563, 0.285714286, 0.5, 0.6), 4, 4, byrow = TRUE)

mats_list <- list(XC3, XC4)
yr_ord <- c(1, 2)
pch_ord <- c(1, 1)

anth_lefkoMat <- create_IM(mats_list, anthframe, hstages = NA, historical = FALSE,
  poporder = 1, patchorder = pch_ord, yearorder = yr_ord)

anth_lefkoMat

nullmodel1 <- hist_null(anth_lefkoMat, 1) # Ehrlen format
nullmodel2 <- hist_null(anth_lefkoMat, 2) # deVries format

```

 image3

Create Matrix Image

Description

Function `image3()` is a generic function that creates matrix plots. It acts as a wrapper for the `image()` function in package `SparseM`, conducting all necessary conversions and automating image production across all or just specific matrices.

Usage

```
image3(mats, ...)
```

Arguments

<code>mats</code>	A <code>lefkoMat</code> object, or a single projection matrix, for which the dominant eigenvalue is desired.
<code>...</code>	Other parameters

Value

Produces a single matrix image, or a series of images, depending on the input. Non-zero elements appear as red space, while zero elements appear as white space.

See Also

[image3.lefkoMat\(\)](#)

[image3.matrix\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
```

```

eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

image3(ehrlen3, used = 1, type = "U")

# Cyripedium example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,

```

```

    yearcol = "year2", patchcol = "patchid", indivcol = "individ")
image3(cypmatrix2r, used = 1, type = "U")

```

image3.lefkoElas	<i>Create Matrix Image(s) for lefkoElas Object</i>
------------------	--

Description

Function `image3.lefkoElas` plots matrix images for elasticity matrices supplied within `lefkoElas` objects. This function operates as a wrapper the `image()` function in package `SparseM`, conducting all necessary conversions and automating image production across all or just specific matrices.

Usage

```

## S3 method for class 'lefkoElas'
image3(mats, used = "all", type = "a", ...)

```

Arguments

<code>mats</code>	A <code>lefkoElas</code> object.
<code>used</code>	A numeric value or vector designating the matrices to plot. Can also take the value "all", which plots all matrices. Defaults to "all".
<code>type</code>	Character value indicating whether to plot "a"historical or "h"istorical elasticity matrices. Defaults to "a"historical, but will plot a historical elasticity matrix image if no ahistorical elasticity matrix exists.
<code>...</code>	Other parameters.

Value

Plots a matrix image, or series of matrix images, denoting non-zero elements as red space and zero elements as white space.

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlelko3(data = lathvert, stageframe = lathframe,
  year = c(1989, 1990), stages = c("stage3", "stage2", "stage1"),
  supplement = lathsupp3, yearcol = "year2", indivcol = "indiv")

ehrlen_elas <- elasticity3(ehrlen3)

image3(ehrlen_elas, used = 1, type = "h")

# Cypripedium example
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, 0, 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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypelas <- elasticity3(cypmatrix2r)

image3(cypelas, used = 1, type = "a")

```

image3.lefkoMat

Create Matrix Image(s) for lefkoMat Object

Description

Function `image3.lefkoMat` plots matrix images for matrices supplied within `lefkoMat` objects. This function operates as a wrapper for the `image()` function in package `SparseM`, conducting all necessary conversions and automating image production across all or just specific matrices.

Usage

```

## S3 method for class 'lefkoMat'
image3(mats, used = "all", type = "A", ...)

```

Arguments

<code>mats</code>	A <code>lefkoMat</code> object.
<code>used</code>	A numeric value or vector designating the matrices to plot. Can also take the value "all", which plots all matrices. Defaults to "all".
<code>type</code>	Character value indicating whether to plot A, U, or F matrices. Defaults to "A".
<code>...</code>	Other parameters.

Value

Plots a matrix image, or series of matrix images, denoting non-zero elements as red space and zero elements as white space.

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

image3(ehrlen3, used = 1, type = "U")

# Cyripedium example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

image3(cypmatrix2r, used = 1, type = "U")

```

Description

Function `image3.lefkoSens` plots matrix images for sensitivity matrices supplied within `lefkoSens` objects. This function operates as a wrapper the `image()` function in package `SparseM`, conducting all necessary conversions and automating image production across all or just specific matrices.

Usage

```
## S3 method for class 'lefkoSens'
image3(mats, used = "all", type = "a", ...)
```

Arguments

<code>mats</code>	A <code>lefkoSens</code> object.
<code>used</code>	A numeric value or vector designating the matrices to plot. Can also take the value "all", which plots all matrices. Defaults to "all".
<code>type</code>	Character value indicating whether to plot "a"historical or "h"istorical sensitivity matrices. Defaults to "a"historical, but will plot a historical sensitivity matrix image if no ahistorical sensitivity matrix exists.
<code>...</code>	Other parameters.

Value

Plots a matrix image, or series of matrix images, denoting non-zero elements as red space and zero elements as white space.

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
```

```

eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe,
  year = c(1989, 1990), stages = c("stage3", "stage2", "stage1"),
  supplement = lathsupp3, yearcol = "year2", indivcol = "individ")

ehrlen_sens <- sensitivity3(ehrlen3)

image3(ehrlen_sens, used = 1, type = "h")

# Cypripedium example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsups2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

```

```

cypmatrix2r <- r1efko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypsens <- sensitivity3(cypmatrix2r)
image3(cypsens, used = 1, type = "a")

```

image3.list

Create Matrix Images for Matrices in a List

Description

Function `image3.matrix` plots matrix images for matrices contained in a list of matrices. This function operates as a wrapper for the `image()` function in package `SparseM`, conducting all necessary conversions and automating image production across all or just specific matrices.

Usage

```

## S3 method for class 'list'
image3(mats, used = "all", ...)

```

Arguments

<code>mats</code>	A list class object.
<code>used</code>	A numeric vector of projection matrices within <code>mats</code> to represent as matrix images. Can also take the text value "all", which will produce images of all matrices. Defaults to "all".
<code>...</code>	Other parameters.

Value

Plots a matrix image, or series of matrix images, denoting non-zero elements as red space and zero elements as white space.

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep"),
  stage1 = c("Sd", "rep", "Sd", "rep", "all", "all"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054),
  type = c(1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfeko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

image3(ehrlen3$A, used = 1)

# Cyripedium example
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, 0, 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, 0, 1, 1, 1, 1, 1)
binvec <- c(0, 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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

image3(cypmatrix2r$A, used = 1)

```

image3.matrix

Create a Matrix Image for a Single Matrix

Description

Function `image3.matrix` plots a matrix image for a single matrix. This function operates as a wrapper for the `image()` function in package `SparseM`, conducting all necessary conversions and automating image production across all or just specific matrices.

Usage

```

## S3 method for class 'matrix'
image3(mats, ...)

```

Arguments

<code>mats</code>	A matrix class object.
<code>...</code>	Other parameters.

Value

Plots a matrix image, or series of matrix images, denoting non-zero elements as red space and zero elements as white space.

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

image3(ehrlen3$U[[1]])

# Cypripedium example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsups2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

image3(cypmatrix2r$U[[1]])

```

lambda3

Estimate Dominant Eigenvalue and Deterministic Population Growth Rate

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.

... Other parameters.

Value

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

See Also

[lambda3.lefkoMat\(\)](#)
[lambda3.matrix\(\)](#)
[slambda3\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "Flo", "Dorm")
repvector <- c(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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
```

```

type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
lambda3(ehrlen3mean)

# Cypridium example
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, 0, 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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

```

```
lambda3(cypmatrix2r)
```

lambda3.lefkoMat	<i>Estimate Deterministic Population Growth Rates of lefkoMat Matrices</i>
------------------	--

Description

lambda3.lefkoMat() returns the dominant eigenvalues of all 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 ahistorical matrices.

Usage

```
## S3 method for class 'lefkoMat'
lambda3(mats, sparse = "auto", ...)
```

Arguments

mats	An object of class lefkoMat.
sparse	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
...	Other parameters.

Value

This function returns the dominant eigenvalue of each \$A matrix in mats. The output includes a data frame showing the population, patch, and lambda estimate for each A matrix. Row names correspond to the order of the matrix within the A element of mats.

Notes

The sparse option allows the function to utilize underlying methods of either dense or sparse matrix manipulation in order to speed up processing time and prevent memory shortages. Under the auto setting, the function will determine whether the matrix is sparse and act accordingly. For extremely large, sparse matrices, the user may simply set sparse = "yes" to save time further and force the use of sparse format in calculations.

See Also

[lambda3\(\)](#)

[lambda3.matrix\(\)](#)

[slambda3\(\)](#)

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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 = "size",
  censorcol = "Missing1988", censorkeep = NA, censor = TRUE)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlelko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
lambda3(ehrlen3mean)

# Cyripedium example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsups2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

lambda3(cypmatrix2r)

```

lambda3.matrix	<i>Estimate Deterministic Population Growth Rate of Single Projection Matrix</i>
----------------	--

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 ahistorical matrices.

Usage

```
## S3 method for class 'matrix'
lambda3(mats, sparse = "auto", ...)
```

Arguments

<code>mats</code>	A population projection matrix of class <code>matrix</code> .
<code>sparse</code>	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
<code>...</code>	Other parameters.

Value

This function returns the dominant eigenvalue of the matrix.

Notes

The sparse option allows the function to utilize underlying methods of either dense or sparse matrix manipulation in order to speed up processing time and prevent memory shortages. Under the auto setting, the function will determine whether the matrix is sparse and act accordingly. For extremely large sparse matrices, the user may simply set `sparse = "yes"` to save time further and force the use of sparse format in calculations.

See Also

[lambda3\(\)](#)
[lambda3.lefkoMat\(\)](#)
[slambda3\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "indiv")

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

# Cypripedium example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",

```

```

    "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
    "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

lambda3(cypmatrix2r$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 referring to patch within the population.

GENET A numeric 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.

InVol89 Natural logarithm of Volume89.

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

Flow89 Number of flowers in 1989.

Intactseed89 Number 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^3 in 1990.

InVol90 Natural logarithm of Volume90.

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

Flow90 Number of flowers in 1990.

Intactseed90 Number 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^3 in 1991.

InVol91 Natural logarithm of Volume91.

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

Flow91 Number of flowers in 1991.

Intactseed91 Number 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

```
# Lathyrus example using blocksize - when repeated patterns exist in variable
# order
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

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

lambda3(ehrlen3mean)
```

```

# Lathyrus example without blocksize - when no repeated patterns exist in
# variable order and all variables names are specified
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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",
  juvcol = c("Seedling1988", "Seedling1989", "Seedling1990", "Seedling1991"),
  sizeacol = c("Volume88", "Volume89", "Volume90", "Volume91"),
  repstracol = c("FCODE88", "FCODE89", "FCODE90", "FCODE91"),
  fecacol = c("Intactseed88", "Intactseed89", "Intactseed90", "Intactseed91"),
  deadacol = c("Dead1988", "Dead1989", "Dead1990", "Dead1991"),
  nonobsacol = c("Dormant1988", "Dormant1989", "Dormant1990", "Dormant1991"),
  censorcol = c("Missing1988", "Missing1989", "Missing1990", "Missing1991"),
  stageassign = lathframe, stagesize = "size",
  censorkeep = NA, censor = TRUE)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

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

lambda3(ehrlen3mean)

```

Description

This package creates population matrix projection models (MPMs) for use in population ecological analyses. It presents a complete working environment for the construction and analysis of ALL kinds of MPMs and IPMs, including age, stage, and age-by-stage versions. Its specialty is the estimation of historical MPMs, which are 2d matrices comprising 3 monitoring occasions (2 time steps or periods) of demographic information. The package constructs both function-based and raw MPMs for both standard ahistorical (i.e. 2 occasions, 1 time step) and historical analyses, has functions for complex density-dependent and independent, and stochastic and cyclical, projections, and also includes the automatic calculation of quality control metrics throughout every step of analysis. It also includes powerful functions to standardize demographic datasets.

Details

The lefk03 package provides seven 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 and projection functions
6. Functions describing, summarizing, or visualizing MPMs and derived structures
7. Extra functions used to illustrate core theory and ideas.

lefk03 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. Ehrlén, and S. Kurokawa. 2021. *lefk03*: analyzing individual history through size-classified matrix population models. *Methods in Ecology and Evolution* 12(2): 378-382.

lmean

Estimate Mean Projection Matrices

Description

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

Usage

```
lmean(mats, matsout = "all")
```

Arguments

<code>mats</code>	A <code>lefkoMat</code> object.
<code>matsout</code>	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. Defaults to "all".

Value

Yields a `lefkoMat` object with the following characteristics:

<code>A</code>	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 <code>U</code> and <code>F</code> matrices. All matrices output in the <code>matrix</code> class.
<code>U</code>	A list of mean survival-transition matrices sorted as in <code>A</code> . All matrices output in the <code>matrix</code> class.
<code>F</code>	A list of mean fecundity matrices sorted as in <code>A</code> . All matrices output in the <code>matrix</code> class.
<code>hstages</code>	A data frame showing the pairing of ahistorical stages used to create historical stage pairs. Given if the MPM is historical.
<code>ahstages</code>	A data frame detailing the characteristics of associated ahistorical stages.
<code>labels</code>	A data frame detailing the order of population, patch, and year of each mean matrix. If <code>pop</code> , <code>patch</code> , or <code>year2</code> are <code>NA</code> in the original <code>labels</code> set, then these will be re-labeled as <code>A</code> , <code>1</code> , or <code>1</code> , respectively.
<code>matrixqc</code>	A short vector describing the number of non-zero elements in <code>U</code> and <code>F</code> mean matrices, and the number of annual matrices.
<code>modelqc</code>	This is the <code>qc</code> portion of the <code>modelsuite</code> input. Only output from <code>lefkoMat</code> objects resulting from function-based estimation.
<code>dataqc</code>	A vector showing the numbers of individuals and rows in the vertical dataset used as input. Only output from <code>lefkoMat</code> objects resulting from raw matrix estimation.

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

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

# Cyripedium example
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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cyp2mean <- lmean(cypmatrix2r)
cyp2mean

```

logistic3

Two-parameter logistic function

Description

Function `logistic3()` creates a vector of values produced by the logistic function as applied with a user-specified time lag. The logistic function is given as $\phi_{t+1} = \phi_t \lambda (1 - n_t/K)$. Here, if no separate_N vector is provided, then $n_t = \phi_t$. If λ is not provided, then it defaults to 1.0 .

Usage

```
logistic3(
```

```

start_value,
alpha,
beta = 0,
lambda = 1,
time_steps = 100L,
time_lag = 1L,
pre0_subs = FALSE,
pre0_value = 0,
substoch = 0L,
separate_N = NULL
)

```

Arguments

start_value	A positive number to start the return vector in time 0.
alpha	The carrying capacity K.
beta	If set to some positive number, then this number is the maximum value of phi to enforce. Otherwise, equals 0 and enforces no limit.
lambda	The value of the discrete population growth rate to use. Equal to the natural logarithm of the instantaneous growth rate, r.
time_steps	The number of time steps to run the projection. Must be a positive integer.
time_lag	A positive integer denoting the number of time steps back for the value of phi in the logistic function.
pre0_subs	A logical value indicating whether to use a number other than that given in start_value for values of phi lagged from times prior to time 0.
pre0_value	A positive number to use for phi lagged from times prior to time 0. Only used if pre0_subs = TRUE.
substoch	An integer value indicating the kind of substochasticity to use. Values include: 0, no substochasticity enforced (the default); 1, all numbers must be non-negative; and 2, all numbers should be forced to the interval [0, 1].
separate_N	An optional numeric vector with values of N in each time, if phi is to be treated as different from N in the logistic model.

Value

A numeric vector of values showing values projected under the- logistic function.

Examples

```

trial_run1 <- logistic3(1, alpha = 5)
plot(trial_run1)

trial_run2 <- logistic3(1, alpha = 5, beta = 5)
plot(trial_run2)

trial_run3 <- logistic3(1, alpha = 100)
plot(trial_run3)

```

```
trial_run4 <- logistic3(1, alpha = 100, beta = 50)
plot(trial_run4)

trial_run5 <- logistic3(1, alpha = 500)
plot(trial_run5)

trial_run6 <- logistic3(1, alpha = 500, beta = 501)
plot(trial_run6)

used_Ns <- c(10, 15, 12, 14, 14, 150, 15, 1, 5, 7, 9, 14, 13, 16, 17, 19,
            25, 26)
trial_run7 <- logistic3(1, alpha = 500, beta = 501, separate_N = used_Ns)
plot(trial_run7)
```

ltre3

Conduct a Life Table Response Experiment

Description

`ltre3()` is a generic function that returns life table response experiment (LTRE) or stochastic LTRE matrices for the input projection matrices.

Usage

```
ltre3(mats, refmats, ...)
```

Arguments

<code>mats</code>	A <code>lefkoMat</code> object, population projection matrix, or list of population projection matrices.
<code>refmats</code>	A reference <code>lefkoMat</code> object, or matrix, for use as the control. If missing, then is set to the same object as <code>mats</code> .
<code>...</code>	Other parameters.

Value

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

Notes

Deterministic LTRE is one-way, fixed, and based on the sensitivities of the matrix midway between each input matrix and the reference matrix, per Caswell (2001, *Matrix Population Models*, Sinauer Associates, MA, USA). Stochastic LTRE is per Davison et al. (2010, doi: 10.1111/j.1365-2745.2009.01611.x).

See Also

[ltre3.lefkoMat\(\)](#)
[summary.lefkoLTRE\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "indiv")

ltre3(ehrlen3)

# Cyripedium example
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, 0, 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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "indiv")

ltre3(cypmatrix2r)

```

Description

ltre3.lefkoMat() returns a set of matrices of one-way LTRE (life table response experiment) or stochastic LTRE matrices contributions.

Usage

```
## S3 method for class 'lefkoMat'
ltre3(
  mats,
  refmats = NA,
  ref = NA,
  stochastic = FALSE,
  steps = 10000,
  burnin = 3000,
  time_weights = NA,
  sparse = "auto",
  rseed = NA,
  append_mats = FALSE,
  ...
)
```

Arguments

<code>mats</code>	An object of class <code>lefkoMat</code> .
<code>refmats</code>	A reference <code>lefkoMat</code> object, or matrix, for use as the control. If missing, then is set to the same object as <code>mats</code> .
<code>ref</code>	A numeric value indicating which matrix or matrices in <code>refmats</code> to use as the control. The numbers used must correspond to the number of the matrices in the <code>labels</code> element of the associated <code>lefkoMat</code> object. The default setting, <code>NA</code> , uses all entries in <code>refmats</code> .
<code>stochastic</code>	A logical value determining whether to conduct a deterministic (<code>FALSE</code>) or stochastic (<code>TRUE</code>) elasticity analysis. Defaults to <code>FALSE</code> .
<code>steps</code>	The number of occasions to project forward in stochastic simulation. Defaults to 10,000.
<code>burnin</code>	The number of initial steps to ignore in stochastic projection when calculating stochastic elasticities. Must be smaller than <code>steps</code> . Defaults to 3000.
<code>time_weights</code>	Numeric vector denoting the probabilistic weightings of all matrices. Defaults to equal weighting among matrices.
<code>sparse</code>	A string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
<code>rseed</code>	Optional numeric value corresponding to the random seed for stochastic simulation.
<code>append_mats</code>	A logical value denoting whether to include the original A, U, and F matrices in the returned <code>lefkoLTRE</code> object. Defaults to <code>FALSE</code> .
<code>...</code>	Other parameters.

Value

This function returns an object of class `lefkoLTRE`. This includes a list of LTRE matrices as object `ltre_det` if a deterministic LTRE is called for, or a list of mean-value LTRE matrices as object

ltre_mean and a list of SD-value LTRE matrices as object ltre_sd if a stochastic LTRE is called for. This is followed by the stageframe as object ahstages, the order of historical stages as object hstages, the age-by-stage order as object agestages, the order of matrices as object labels, and, if requested, the original A, U, and F matrices.

Notes

Deterministic LTRE is one-way, fixed, and based on the sensitivities of the matrix midway between each input matrix and the reference matrix, per Caswell (2001, Matrix Population Models, Sinauer Associates, MA, USA). Stochastic LTRE is simulated per Davison et al. (2010) Journal of Ecology 98:255-267 (doi: 10.1111/j.1365-2745.2009.01611.x).

Default behavior for stochastic LTRE uses the full population provided in mats as the reference if no refmats and ref is provided. If no refmats is provided but ref is, then the matrices noted in ref are used as the reference matrix set. Year and patch order is utilized from object mats, but not from object refmats, in which each matrix is assumed to represent a different year from one population. This function cannot currently handle multiple populations within the same mats object (although such analysis is possible if these populations are designated as patches instead).

See Also

[ltre3\(\)](#)
[summary.lefkoLTRE\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
```

```

stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

ltre3(ehrlen3, stochastic = TRUE)

# Cypripedium example
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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

```



```

cypmatrix2r <- rlfko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

ltre3(cypmatrix2r)

```

modelsearch

Develop Best-fit Vital Rate Estimation Models for MPM Development

Description

Function `modelsearch()` runs exhaustive model building and selection for each vital rate needed to estimate a function-based MPM or IPM. It returns best-fit models for each vital rate, model table showing all models tested, and model quality control data. The final output can be used as input in other functions within this package.

Usage

```

modelsearch(
  data,
  stageframe = NULL,
  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"),
  sizeb = c(NA, NA, NA),
  sizec = c(NA, NA, NA),
  repst = c("repstatus3", "repstatus2", "repstatus1"),
  fec = c("feca3", "feca2", "feca1"),
  stage = c("stage3", "stage2", "stage1"),
  matstat = c("matstatus3", "matstatus2", "matstatus1"),
  indiv = "individ",
  patch = NA,
  year = "year2",
  density = NA,
  sizedist = "gaussian",
  sizebdist = NA,
  sizecdist = NA,
  fecdist = "gaussian",
  size.zero = FALSE,
  sizeb.zero = FALSE,

```

```

sizec.zero = FALSE,
size.trunc = FALSE,
sizeb.trunc = FALSE,
sizec.trunc = FALSE,
fec.zero = FALSE,
fec.trunc = FALSE,
patch.as.random = TRUE,
year.as.random = TRUE,
juvestimate = NA,
juvsize = FALSE,
jsize.zero = FALSE,
jsizeb.zero = FALSE,
jsizec.zero = FALSE,
jsize.trunc = FALSE,
jsizeb.trunc = FALSE,
jsizec.trunc = FALSE,
fectime = 2,
censor = NA,
age = NA,
indcova = NA,
indcovb = NA,
indcovc = NA,
random.indcova = FALSE,
random.indcovb = FALSE,
random.indcovc = FALSE,
test.group = FALSE,
show.model.tables = TRUE,
global.only = FALSE,
accuracy = TRUE,
quiet = FALSE
)

```

Arguments

data	The vertical dataset to be used for analysis. This dataset should be of class <code>hfvdata</code> , but can also be a data frame formatted similarly to the output format provided by functions <code>verticalize3()</code> or <code>historicalize3()</code> , as long as all needed variables are properly designated.
stageframe	The stageframe characterizing the life history model used. Optional unless <code>test.group = TRUE</code> , in which case it is required. Defaults to <code>NULL</code> .
historical	A logical variable denoting whether to assess the effects of state in occasion $t-1$, in addition to state in occasion t . Defaults to <code>TRUE</code> .
approach	The statistical approach to be taken for model building. The default is "mixed", which uses the mixed model approach utilized in packages <code>lme4</code> and <code>glmmTMB</code> . Other options include "glm", which uses generalized linear modeling assuming that all factors are fixed.
suite	This describes the global model for each vital rate estimation, and has the following possible values: <code>full</code> , includes main effects and all two-way interac-

tions 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); age, all vital rates estimated with age and y-intercepts only; 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 $\Delta AICc \leq 2.0$. 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). May also be set to vitalrates = "leslie", which is equivalent to setting c("surv", "fec") for a Leslie MPM. This choice also determines how internal data subsetting for vital rate model estimation will work. Defaults to c("surv", "size", "fec").
surv	A vector indicating the variable names coding for status as alive or dead in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("alive3", "alive2", "alive1").
obs	A vector indicating the variable names coding for observation status in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("obsstatus3", "obsstatus2", "obsstatus1").
size	A vector indicating the variable names coding for the primary size variable on occasions $t+1$, t , and $t-1$, respectively. Defaults to c("sizea3", "sizea2", "sizea1").
sizedb	A vector indicating the variable names coding for the secondary size variable on occasions $t+1$, t , and $t-1$, respectively. Defaults to c(NA, NA, NA), in which case sizedb is not used.
sizec	A vector indicating the variable names coding for the tertiary size variable on occasions $t+1$, t , and $t-1$, respectively. Defaults to c(NA, NA, NA), in which case sizec is not used.
repst	A vector indicating the variable names coding for reproductive status in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("repstatus3", "repstatus2", "repstatus1").
fec	A vector indicating the variable names coding for fecundity in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("feca3", "feca2", "feca1").
stage	A vector indicating the variable names coding for stage in occasions $t+1$, t , and $t-1$. Defaults to c("stage3", "stage2", "stage1").
matstat	A vector indicating the variable names coding for maturity status in occasions $t+1$, t , and $t-1$. Defaults to c("matstatus3", "matstatus2", "matstatus1").

<code>indiv</code>	A text value indicating the variable name coding individual identity. Defaults to "individ".
<code>patch</code>	A text value indicating the variable name coding for patch, where patches are defined as permanent subgroups within the study population. Defaults to NA.
<code>year</code>	A text value indicating the variable coding for observation occasion t . Defaults to <code>year2</code> .
<code>density</code>	A text value indicating the name of the variable coding for spatial density, should the user wish to test spatial density as a fixed factor affecting vital rates. Defaults to NA.
<code>sizedist</code>	The probability distribution used to model primary size. Options include "gaussian" for the Normal distribution (default), "poisson" for the Poisson distribution, "negbin" for the negative binomial distribution (quadratic parameterization), and "gamma" for the Gamma distribution.
<code>sizebdist</code>	The probability distribution used to model secondary size. Options include "gaussian" for the Normal distribution, "poisson" for the Poisson distribution, "negbin" for the negative binomial distribution (quadratic parameterization), and "gamma" for the Gamma distribution. Defaults to NA.
<code>sizedist</code>	The probability distribution used to model tertiary size. Options include "gaussian" for the Normal distribution, "poisson" for the Poisson distribution, "negbin" for the negative binomial distribution (quadratic parameterization), and "gamma" for the Gamma distribution. Defaults to NA.
<code>fecdist</code>	The probability distribution used to model fecundity. Options include "gaussian" for the Normal distribution (default), "poisson" for the Poisson distribution, "negbin" for the negative binomial distribution (quadratic parameterization), and "gamma" for the Gamma distribution.
<code>size.zero</code>	A logical variable indicating whether the primary size distribution should be zero-inflated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE.
<code>sizeb.zero</code>	A logical variable indicating whether the secondary size distribution should be zero-inflated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE.
<code>sizec.zero</code>	A logical variable indicating whether the tertiary size distribution should be zero-inflated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE.
<code>size.trunc</code>	A logical variable indicating whether the primary size distribution should be zero-truncated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE. Cannot be TRUE if <code>size.zero = TRUE</code> .
<code>sizeb.trunc</code>	A logical variable indicating whether the secondary size distribution should be zero-truncated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE. Cannot be TRUE if <code>sizeb.zero = TRUE</code> .
<code>sizec.trunc</code>	A logical variable indicating whether the tertiary size distribution should be zero-truncated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE. Cannot be TRUE if <code>sizec.zero = TRUE</code> .

fec.zero	A logical variable indicating whether the fecundity distribution should be zero-inflated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE.
fec.trunc	A logical variable indicating whether the fecundity distribution should be zero-truncated. Only applies to the Poisson and negative binomial distributions. Defaults to FALSE. Cannot be TRUE if fec.zero = TRUE.
patch.as.random	If set to TRUE and approach = "mixed", 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 = "mixed", 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.
jsize.zero	A logical variable indicating whether the primary size distribution of juveniles should be zero-inflated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE.
jsizeb.zero	A logical variable indicating whether the secondary size distribution of juveniles should be zero-inflated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE.
jsizec.zero	A logical variable indicating whether the tertiary size distribution of juveniles should be zero-inflated. Only applies to Poisson and negative binomial distributions. Defaults to FALSE.
jsize.trunc	A logical variable indicating whether the primary size distribution in juveniles should be zero-truncated. Defaults to FALSE. Cannot be TRUE if jsize.zero = TRUE.
jsizeb.trunc	A logical variable indicating whether the secondary size distribution in juveniles should be zero-truncated. Defaults to FALSE. Cannot be TRUE if jsizeb.zero = TRUE.
jsizec.trunc	A logical variable indicating whether the tertiary size distribution in juveniles should be zero-truncated. Defaults to FALSE. Cannot be TRUE if jsizec.zero = TRUE.
fectime	A variable indicating which year of fecundity to use as the response term in fecundity models. Options include 2, which refers to occasion t , and 3, which refers to occasion $t+1$. Defaults to 2.
censor	A vector denoting the names of censoring variables in the dataset, in order from occasion $t+1$, followed by occasion t , and lastly followed by occasion $t-1$. Defaults to NA.

<code>age</code>	Designates the name of the variable corresponding to age in time t in the vertical dataset. Defaults to NA, in which case age is not included in linear models. Should only be used if building Leslie or age x stage matrices.
<code>indcov_a</code>	Vector designating the names in occasions $t+1$, t , and $t-1$ of an individual covariate. Defaults to NA.
<code>indcov_b</code>	Vector designating the names in occasions $t+1$, t , and $t-1$ of a second individual covariate. Defaults to NA.
<code>indcov_c</code>	Vector designating the names in occasions $t+1$, t , and $t-1$ of a third individual covariate. Defaults to NA.
<code>random.indcov_a</code>	A logical value indicating whether <code>indcov_a</code> should be treated as a random categorical factor, rather than as a fixed factor. Defaults to FALSE.
<code>random.indcov_b</code>	A logical value indicating whether <code>indcov_b</code> should be treated as a random categorical factor, rather than as a fixed factor. Defaults to FALSE.
<code>random.indcov_c</code>	A logical value indicating whether <code>indcov_c</code> should be treated as a random categorical factor, rather than as a fixed factor. Defaults to FALSE.
<code>test.group</code>	A logical value indicating whether to include the group variable from the input <code>stageframe</code> as a fixed categorical variable in linear models. Defaults to FALSE.
<code>show.model.tables</code>	If set to TRUE, then includes full modeling tables in the output. Defaults to TRUE.
<code>global.only</code>	If set to TRUE, then only global models will be built and evaluated. Defaults to FALSE.
<code>accuracy</code>	A logical value indicating whether to test accuracy of models. See Notes section for details on how accuracy is assessed. Defaults to TRUE.
<code>quiet</code>	If set to TRUE, then model building and selection will proceed with most warnings and diagnostic messages silenced. Defaults to FALSE.

Value

This function yields an object of class `lefkoMod`, which is a list in which the first 14 elements are the best-fit models for survival, observation status, primary size, secondary size, tertiary size, reproductive status, fecundity, juvenile survival, juvenile observation, juvenile primary size, juvenile secondary size, juvenile tertiary size, juvenile transition to reproduction, and juvenile transition to maturity, respectively. This is followed by 14 elements corresponding to the model tables for each of these vital rates, in order, followed by a data frame showing the order and names of variables used in modeling, followed by a single character element denoting the criterion used for model selection, and ending on a data frame with quality control data:

<code>survival_model</code>	Best-fit model of the binomial probability of survival from occasion t to occasion $t+1$. Defaults to 1.
<code>observation_model</code>	Best-fit model of the binomial probability of observation in occasion $t+1$ given survival to that occasion. Defaults to 1.
<code>size_model</code>	Best-fit model of the primary size metric on occasion $t+1$ given survival to and observation in that occasion. Defaults to 1.

sizeb_model	Best-fit model of the secondary size metric on occasion $t+1$ given survival to and observation in that occasion. Defaults to 1.
sizec_model	Best-fit model of the tertiary size metric on occasion $t+1$ given survival to and observation in that occasion. Defaults to 1.
repstatus_model	Best-fit model of the binomial probability of reproduction in occasion $t+1$, given survival to and observation in that occasion. Defaults to 1.
fecundity_model	Best-fit model of fecundity in occasion $t+1$ given survival to, and observation and reproduction in that occasion. Defaults to 1.
juv_survival_model	Best-fit model of the binomial probability of survival from occasion t to occasion $t+1$ of an immature individual. Defaults to 1.
juv_observation_model	Best-fit model of the binomial probability of observation in occasion $t+1$ given survival to that occasion of an immature individual. Defaults to 1.
juv_size_model	Best-fit model of the primary size metric on occasion $t+1$ given survival to and observation in that occasion of an immature individual. Defaults to 1.
juv_sizeb_model	Best-fit model of the secondary size metric on occasion $t+1$ given survival to and observation in that occasion of an immature individual. Defaults to 1.
juv_sizec_model	Best-fit model of the tertiary size metric on occasion $t+1$ given survival to and observation in that occasion of an immature individual. Defaults to 1.
juv_reproduction_model	Best-fit model of the binomial probability of reproduction in occasion $t+1$, given survival to and observation in that occasion of an individual that was immature in occasion 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 occasion $t+1$ but immature in occasion t . Defaults to 1.
juv_maturity_model	Best-fit model of the binomial probability of becoming mature in occasion $t+1$, given survival to that occasion of an individual that was immature in occasion t . Defaults to 1.
survival_table	Full dredge model table of survival probability.
observation_table	Full dredge model table of observation probability.
size_table	Full dredge model table of the primary size variable.
sizeb_table	Full dredge model table of the secondary size variable.
sizec_table	Full dredge model table of the tertiary size variable.
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 primary size in immature individuals.
juv_sizeb_table	Full dredge model table of secondary size in immature individuals.
juv_sizec_table	Full dredge model table of tertiary size in immature individuals.
juv_reproduction_table	Full dredge model table of immature reproduction probability.
juv_maturity_table	Full dredge model table of the probability of an immature individual transitioning to maturity.
criterion	Character variable denoting the criterion used to determine the best-fit model.
qc	Data frame with five variables: 1) Name of vital rate, 2) number of individuals used to model that vital rate, 3) number of individual transitions used to model that vital rate, 4) parameter distribution used to model the vital rates, and 5) accuracy of model, given as detailed in Notes section.

Notes

Setting `suite = "cons"` prevents the inclusion of size and reproductive status as fixed, independent factors in modeling. However, it does not prevent any other terms from being included. Density, age, individual covariates, individual identity, patch, and year may all be included.

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, Gamma, or binomial response), `glm.nb()` (GLM with negative binomial response), `zeroinfl()` (GLM with zero-inflated Poisson or negative binomial response), `vglm()` (GLM with zero-truncated Poisson or negative binomial response), `lmer()` (mixed model with Gaussian response), `glmer()` (mixed model with binomial, Poisson, or Gamma response), and `glmmTMB()` (mixed model with negative binomial, or zero-truncated or zero-inflated Poisson or negative binomial response). See documentation related to these functions for further information. Any response term that is invariable in the dataset will lead to a best-fit model for that response represented by a single constant value.

When `modelsearch()` is called, it first builds global models for all vital rates and runs them. If a global model fails, then the function proceeds by dropping any two-way interactions and trying again. If this fails, then function will continue to drop key terms (typically random terms or individual covariates) until something runs. The last attempt if running a mixed set of models is to try a `glm` version of the original failed model, and use that as a global model if it runs properly. Finally, if all attempts fail, then the function returns a 1.

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 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 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.

Accuracy of vital rate models is calculated differently depending on vital rate and assumed distribution. For all vital rates assuming a binomial distribution, including survival, observation status, reproductive status, and juvenile version of these, accuracy is calculated as the percent of predicted responses equal to actual responses. In all other models, accuracy is actually the conditional R-squared using package MuMIn's `r.squaredGLMM()` function, estimated via the delta method. When this method fails, `modelsearch()` calculates McFadden's pseudo R-squared. If this fails, then NA is returned.

Care must be taken to build models that test the impacts of state in occasion $t-1$ 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.

This function handles generalized linear models (GLMs) under zero-inflated distributions using the `zeroinfl()` function, and zero-truncated distributions using the `vglm()` function. Model dredging may fail with these functions, leading to the global model being accepted as the best-fit model. However, model dredges of mixed models work for all distributions. We encourage the use of mixed models in all cases.

The negative binomial and truncated negative binomial distributions use the quadratic structure emphasized in Hardin and Hilbe (2018, 4th Edition of *Generalized Linear Models and Extensions*). The truncated negative binomial distribution may fail to predict size probabilities correctly when dispersion is near that expected of the Poisson distribution. To prevent this problem, we have integrated a cap on the overdispersion parameter. However, when using this distribution, please check the matrix column sums to make sure that they do not predict survival greater than 1.0. If they do, then please use either the negative binomial distribution or the zero-truncated Poisson distribution.

If density dependence is explored through function `modelsearch()`, then the interpretation of density is not the full population size but rather the spatial density term included in the dataset.

Users building vital rate models for Leslie matrices must set `vitalrates = c("surv", "fec")` or `vitalrates = "leslie"` rather than the default, because only survival and fecundity should be estimated in these cases. Also, the `suite` setting can be set to either `age` or `cons`, as the results will be exactly the same.

Examples

```

# Lathyrus example
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", "Sd1", "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)
obsvector <- c(0, 1, 0, 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)
immvector <- c(1, 1, 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)
indataset <- c(0, 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)

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)

lathmodelsln3 <- modelsearch(lathvertln, historical = TRUE,
approach = "mixed", suite = "main",
vitalrates = c("surv", "obs", "size", "repst", "fec"), juvestimate = "Sd1",
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)

# Here we use supplemental() to provide overwrite and reproductive info
lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "mat", "Sd", "Sd1"),
stage2 = c("Sd", "Sd", "Sd", "Sd", "Sd1", "rep", "rep"),
stage1 = c("Sd", "rep", "Sd", "rep", "Sd", "mat", "mat"),
eststage3 = c(NA, NA, NA, NA, "mat", NA, NA),
eststage2 = c(NA, NA, NA, NA, "Sd1", NA, NA),
eststage1 = c(NA, NA, NA, NA, "Sd1", NA, NA),
givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, NA, 0.345, 0.054),

```

```

type = c(1, 1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe1n, historical = TRUE)

lathmat3ln <- flefko3(year = "all", patch = "all", stageframe = lathframe1n,
  modelsuite = lathmodels1n3, data = lathvert1n, supplement = lathsupp3,
  reduce = FALSE)

summary(lathmat3ln)

```

 overwrite

Create 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,
  type_t12 = NA
)

```

Arguments

stage3	The name of the stage in occasion $t+1$ in the transition to be replaced. Abbreviations for groups of stages are also allowed (see Notes).
stage2	The name of the stage in occasion t in the transition to be replaced. Abbreviations for groups of stages are also allowed (see Notes).
stage1	The name of the stage in occasion $t-1$ in the transition to be replaced. Only needed if a historical matrix is to be produced. Abbreviations for groups of stages are also allowed (see Notes).
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 between occasions t and $t+1$ to be replaced. This should be entered as 1, S, or s for the replacement of a survival transition; or 2, F, or f for the replacement of a fecundity transition. If empty or not provided, then defaults to 1 for survival transition.
type_t12	An optional vector denoting the kind of transition between occasions $t-1$ and t . Only necessary if a historical MPM in deVries format is desired. This should be entered as 1, S, or s for a survival transition; or 2, F, or f for a fecundity transitions. Defaults to 1 for survival transition, with impacts only on the construction of deVries-format hMPMs.

Value

A data frame that puts the above vectors together and can be used as input in `flefko3()`, `flefko2()`, `rlefko3()`, `rlefko2()`, and `aflefko2()`.

Variables in this data frame include the following:

stage3	Stage at occasion $t+1$ in the transition to be replaced.
stage2	Stage at occasion t in the transition to be replaced.
stage1	Stage at occasion $t-1$ in the transition to be replaced.
eststage3	Stage at occasion $t+1$ in the transition to replace the transition designated by stage3, stage2, and stage1.
eststage2	Stage at occasion t in the transition to replace the transition designated by stage3, stage2, and stage1.
eststage1	Stage at occasion $t-1$ 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 from occasion t to occasion $t+1$ is a survival-transition probability (1) or a fecundity rate (2).
convtype_t12	Designates whether the transition from occasion $t-1$ to occasion t is a survival transition probability (1), a fecundity rate (2).

Notes

Entries in stage3, stage2, and stage1 can include abbreviations for groups of stages. Use `rep` if all reproductive stages are to be used, `nrep` if all mature but non-reproductive stages are to be used, `mat` if all mature stages are to be used, `immat` if all immature stages are to be used, `prop` if all propagule stages are to be used, `npr` if all non-propagule stages are to be used, and leave empty or use `all` if all stages in stageframe are to be used.

Examples

```

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

cypover2r

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

cypover3r

```

plot.lefkoProj

Plot Projection Simulations

Description

Function `plot.lefkoProj()` produces plots of `lefkoProj` objects. Acts as a convenient wrapper for the `plot.default()` function.

Usage

```

## S3 method for class 'lefkoProj'
plot(
  x,
  variable = "popsize",
  style = "time",
  repl = "all",
  patch = "pop",
  auto_ylim = TRUE,
  auto_col = TRUE,
  auto_lty = TRUE,

```

```

    auto_title = FALSE,
    ...
  )

```

Arguments

x	A lefkoProj object.
variable	The focus variable of the plot to produce. Defaults to "popsize", which produces line plots of the popsize element in object x.
style	A string denoting the kind of plot to produce. Currently limited to "timeseries", which shows variable against time on the x axis. Other choices include "statespace", which plots variable at one time on the x axis against the same variable in the next time on the y axis.
repl	The replicate to plot. Defaults to "all", in which case all replicates are plotted.
patch	The patch to plot, as labeled in the labels element in object x. Defaults to "pop", in which case only the final population-level projection is plotted. Can also be set to "all", in which case projections for all patches and population in the labels element are plotted.
auto_ylim	A logical value indicating whether the maximum of the y axis should be determined automatically. Defaults to TRUE, but reverts to FALSE if any setting for ylim is given.
auto_col	A logical value indicating whether to shift the color of lines associated with each patch automatically. Defaults to TRUE, but reverts to FALSE if any setting for col is given.
auto_lty	A logical value indicating whether to shift the line type associated with each replicate automatically. Defaults to TRUE, but reverts to FALSE if any setting for lty is given.
auto_title	A logical value indicating whether to add a title to each plot. The plot is composed of the concatenated population and patch names. Defaults to FALSE.
...	Other parameters used by functions plot.default() and lines().

Value

A plot of the results of a [projection3\(\)](#) run.

Notes

Output plots are currently limited to time series and state space plots of population size.

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep"),
  stage1 = c("Sd", "rep", "Sd", "rep", "all", "all"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054),
  type = c(1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfeko3(data = lathvert, stageframe = lathframe,
  year = c(1989, 1990), stages = c("stage3", "stage2", "stage1"),
  repmatrix = lathrepm, supplement = lathsupp3, yearcol = "year2",
  individcol = "individ")

lathproj <- projection3(ehrlen3, nreps = 5, stochastic = TRUE)
plot(lathproj)

```

Description

Function `projection3()` runs projection simulations. It projects the population and patches forward in time by a user-defined number of occasions. A given set of matrices is utilized and not recreated, although elements may be altered if density dependence is set. Projections may be deterministic or stochastic, and may be density dependent in either case. If deterministic, then projections will be cyclical if matrices exist covering multiple occasions for each population or patch.

If stochastic, then annual matrices will be shuffled within patches and populations. Also produces replicates if set.

Usage

```
projection3(
  mpm,
  nreps = 1L,
  times = 10000L,
  historical = FALSE,
  stochastic = FALSE,
  standardize = FALSE,
  growthonly = TRUE,
  integeronly = FALSE,
  substoch = 0L,
  sub_warnings = TRUE,
  quiet = FALSE,
  year = NULL,
  start_vec = NULL,
  start_frame = NULL,
  tweights = NULL,
  density = NULL
)
```

Arguments

mpm	A matrix projection model of class <code>lefkMat</code> , or a list of full matrix projection matrices.
nreps	The number of replicate projections.
times	Number of occasions to iterate per replicate. Defaults to 10,000.
historical	An optional logical value only used if object <code>mpm</code> is a list of matrices, rather than a <code>lefkMat</code> object. Defaults to <code>FALSE</code> for the former case, and overridden by information supplied in the <code>lefkMat</code> object for the latter case.
stochastic	A logical value denoting whether to conduct a stochastic projection or a deterministic / cyclical projection.
standardize	A logical value denoting whether to re-standardize the population size to 1.0 at each occasion. Defaults to <code>FALSE</code> .
growthonly	A logical value indicating whether to produce only the projected population size at each occasion, or a vector showing the stage distribution followed by the reproductive value vector followed by the full population size at each occasion. Defaults to <code>TRUE</code> .
integeronly	A logical value indicating whether to round the number of individuals projected in each stage at each occasion to the nearest integer. Defaults to <code>FALSE</code> .
substoch	An integer value indicating whether to force survival- transition matrices to be substochastic in density dependent simulations. Defaults to <code>0</code> , which does not force substochasticity. Alternatively, <code>1</code> forces all survival-transition elements to

range from 0.0 to 1.0, and forces fecundity to be non-negative; and 2 forces all column rows in the survival-transition matrices to total no more than 1.0, in addition to the actions outlined for option 1.

sub_warnings	A logical value indicating whether to warn the user if density dependence yields matrix values outside of the realm of possibility. Generally, this means that survival-transition elements altered to values outside of the interval [0, 1], and negative fecundity values, will both yield warnings. Defaults to TRUE, but becomes FALSE if quiet = TRUE.
quiet	A logical value indicating whether to suppress warnings. Defaults to FALSE.
year	Either a single integer value corresponding to the year to project, or a vector of times elements with the year to use at each time step. If a vector shorter than times is supplied, then this vector will be cycled. If not provided, then all annual matrices will be cycled within patches or populations.
start_vec	An optional numeric vector denoting the starting stage distribution for the projection. Defaults to a single individual of each stage.
start_frame	An optional data frame characterizing stages, age-stages, or stage-pairs that should be set to non-zero values in the starting vector, and what those values should be. Can only be used with lefkoMat objects.
tweights	An optional numeric vector denoting the probabilistic weightings of annual matrices. Defaults to equal weighting among occasions.
density	An optional data frame describing the matrix elements that will be subject to density dependence, and the exact kind of density dependence that they will be subject to. The data frame used should be an object of class lefkoDens, which is the output from function density_input() .

Value

A list of class lefkoProj, which always includes the first three elements of the following, and also includes the remaining elements below when a lefkoMat object is used as input:

projection	A list of lists of matrices showing the total number of individuals per stage per occasion. The first list corresponds to each pop-patch followed by each population. The inner list corresponds to replicates within each pop-patch or population.
stage_dist	A list of lists of the actual stage distribution in each occasion in each replicate in each pop-patch or population. The list order is the same as in projection.
rep_value	A list of lists of the actual reproductive value in each occasion in each replicate in each pop-patch or population. The list order is the same as in projection.
pop_size	A list of data frames showing the total population size in each occasion per replicate (row within data frame) per pop-patch or population (list element).
labels	A data frame showing the order of populations and patches in item projection.
ahstages	The original stageframe used in the study.
hstages	A data frame showing the order of historical stage pairs.
agestages	A data frame showing the order of age-stage pairs.

control	A short vector indicating the number of replicates and the number of occasions projected per replicate.
density	The data frame input under the density option. Only provided if input by the user.

Notes

Projections are run both at the patch level and at the population level. Population level estimates will be noted at the end of the data frame with 0 entries for patch designation.

Weightings given in `tweights` do not need to sum to 1. Final weightings used will be based on the proportion per element of the sum of elements in the user-supplied vector.

Starting vectors can be input in one of two ways: 1) as `start_vec` input, which is a vector of numbers of the numbers of individuals in each stage, stage pair, or age-stage, with the length of the vector necessarily as long as there are rows in the matrices of the MPM; or 2) as `start_frame` input, which is a data frame showing only those stages, stage pairs, or age-stages that should begin with more than 0 individuals, and the numbers of individuals that those stages should start with (this object is created using the `start_input()` function). If both are provided, then `start_frame` takes precedence and `start_vec` is ignored. If neither is provided, then `projection3()` automatically assumes that each stage, stage pair, or age-stage begins with a single individual. Importantly, if a `lefkoMat` object is not used, and a list of matrices is provided instead, then `start_frame` cannot be utilized and a full `start_vec` must be provided to conduct a simulation with starting numbers of individuals other than 1 per stage.

The resulting data frames in `element projection` are separated by `pop-patch` according to the order provided in `element labels`, but the matrices for each element of projection have the result of each replicate stacked in order on top of one another without any break or indication. Results for each replicate must be separated using the information provided in elements `control` and the 3 stage descriptor elements.

Density dependent projections are automatically set up if object `density` is input. If this object is not included, then density independent projections will be set up. Note that currently, density dependent projections can only be performed with `lefkoMat` objects.

The stage distributions and reproductive values produced are not the asymptotic values as would be given by the standardized right and left eigenvectors associated with the dominant eigenvalue of a matrix, but are vectors describing these values at the specific points in time projected. See equations 14.86 and 14.88 and section 14.4 on Sensitivity and Elasticity Analysis under Environmental Stochasticity in Caswell (2001, Matrix Population Models, Sinauer Associates) for more details.

Consistently positive population growth can quickly lead to population size numbers larger than can be handled computationally. In that circumstance, a continuously rising population size will suddenly become NaN for the remainder of the projection.

Users wishing to run a projection of a single patch in a `lefkoMat` object with multiple patches should subset the MPM first to contain only the patch needed. This can be accomplished with the `subset_lm()` function.

See Also

`start_input()`
`density_input()`
`f_projection3()`

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep"),
  stage1 = c("Sd", "rep", "Sd", "rep", "all", "all"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054),
  type = c(1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe,
  year = c(1989, 1990), stages = c("stage3", "stage2", "stage1"),
  supplement = lathsupp3, yearcol = "year2", individcol = "individ")

lathproj <- projection3(ehrlen3, nreps = 5, stochastic = TRUE)

# Cypripedium example
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, 0, 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, 0, 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)

cypsupp3r <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
  "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat", "mat", "SD", "P1"),
  stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
  "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
  stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "P3", "P3",
  "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D", "XSm", "Sm",
  "mat", "mat", "mat", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "D", "XSm", "Sm", NA, NA),
  eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.1, 0.1, 0.2, 0.2, 0.2, 0.2, 0.25, NA, NA, NA, NA, NA, NA,
  NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
  NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  type_t12 = c(1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
  stageframe = cypframe_raw, historical = TRUE)

cypmatrix3r <- rlfko3(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added", "size1added"),
  supplement = cypsupp3r, yearcol = "year2",
  patchcol = "patchid", individcol = "individ")

cypstoch <- projection3(cypmatrix3r, nreps = 5, stochastic = TRUE)

```

 repvalue3

Estimate Reproductive Value

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 ahistorical matrices.

Usage

```
repvalue3(mats, ...)
```

Arguments

`mats` A `lefkoMat` object, or population projection matrix.
`...` Other parameters.

Value

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

See Also

[repvalue3.lefkoMat\(\)](#)
[repvalue3.matrix\(\)](#)

Examples

```
# Lathyrus deterministic example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
```

```

stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
repvalue3(ehrlen3mean)

# Cypridium stochastic example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

```

```

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsuff2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

repvalue3(cypmatrix2r, stochastic = TRUE)

```

repvalue3.lefkoMat *Estimate Reproductive Value Vectors of Matrices in a lefkoMat Object*

Description

repvalue3.lefkoMat() returns the reproductive values for stages in a 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 ahistorical matrices.

Usage

```

## S3 method for class 'lefkoMat'
repvalue3(
  mats,
  stochastic = FALSE,
  times = 10000,
  tweights = NA,
  seed = NA,
  sparse = "auto",
  ...
)

```

Arguments

mats	An object of class lefkoMat object.
stochastic	A logical value indicating whether to use deterministic (FALSE) or stochastic (TRUE) analysis. Defaults to FALSE.
times	An integer variable indicating number of occasions to project if using stochastic analysis. Defaults to 10000.
tweights	An optional vector indicating the probability weighting to use for each matrix in stochastic simulations. If not given, then defaults to equal weighting.
seed	A number to use as a random number seed.
sparse	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
...	Other parameters.

Value

This function returns the asymptotic reproductive value vectors if deterministic analysis is chosen, and long-run mean reproductive value vectors if stochastic analysis is chosen.

The output depends on whether the `lefkoMat` object used as input is `ahistorical` or `historical`, and whether the analysis is deterministic or stochastic. If deterministic and `ahistorical`, 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 reproductive value vector (`rep_value`). If stochastic and `ahistorical`, then a single data frame is output starting with the number of the population-patch (`matrix_set`), a string concatenating the names of the population and the patch (`poppatch`), the assigned stage id number (`stage_id`), and the stage name (`stage`), and the long-run mean reproductive value vector (`rep_value`).

If a historical matrix is used as input, then two data frames are output into a list object. The `hist` element describes the historical stage-pair reproductive values, while the `ahist` element describes the stage reproductive values. If deterministic, then `hist` contains a data frame including the matrix number (`matrix`), the numeric stage designations for stages in occasions t and $t-1$, (`stage_id_2` and `stage_id_1`, respectively), followed by the respective stage names (`stage_2` and `stage_1`), and ending with the estimated reproductive values (`rep_value`). The associated `ahist` element is as before. If stochastic, then the `hist` element contains a single data frame with the number of the population-patch (`matrix_set`), a string concatenating the names of the population and the patch (`poppatch`), the assigned stage id numbers in times t and $t-1$ (`stage_id_2` and `stage_id_1`, respectively), and the associated stage names (`stage_2` and `stage_1`, respectively), and the long-run mean reproductive values (`rep_value`). The associated `ahist` element is as before in the `ahistorical`, stochastic case.

In addition to the data frames noted above, stochastic analysis will result in the additional output of a list of matrices containing the actual projected reproductive value vectors across all projected occasions, in the order of population-patch combinations in the `lefkoMat` input.

Notes

In stochastic analysis, the projected mean reproductive value vector is the arithmetic mean across the final projected 1000 occasions if the simulation is at least 2000 projected occasions long. If between 500 and 2000 projected occasions long, then only the final 200 are used, and if fewer than 500 occasions are used, then all are used. Note that because reproductive values in stochastic simulations can change greatly in the initial portion of the run, we encourage a minimum 2000 projected occasions per simulation, with 10000 preferred.

See Also

`repvalue3()`
`repvalue3.matrix()`

Examples

```
# Lathyrus deterministic example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
```



```

stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
repvalue3(ehrlen3mean)

# Cypripedium stochastic example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsups2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

repvalue3(cypmatrix2r, stochastic = TRUE)

```

repvalue3.matrix	<i>Estimate Reproductive Value Vector for a Single Population Projection Matrix</i>
------------------	---

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, sparse = "auto", ...)

```

Arguments

<code>mats</code>	A population projection matrix.
<code>sparse</code>	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
<code>...</code>	Other parameters.

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.

See Also

[repvalue3\(\)](#)
[repvalue3.lefkoMat\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
```

```

givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

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

# Cyripedium example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,

```

```

yearcol = "year2", patchcol = "patchid", indivcol = "individ")
repvalue3(cypmatrix2r$A[[1]])

```

ricker3

*Two-parameter Ricker function***Description**

Function `ricker3()` creates a vector of values produced by the two-parameter Ricker function as applied with a user-specified time lag. The two-parameter Ricker function is given as $\phi_{t+1} = \phi_t \alpha e^{-\beta n_t}$. Here, if no separate_N vector is provided, then $n_t = \phi_t$.

Usage

```

ricker3(
  start_value,
  alpha,
  beta,
  time_steps = 100L,
  time_lag = 1L,
  pre0_subs = FALSE,
  pre0_value = 0,
  substoch = 0L,
  separate_N = NULL
)

```

Arguments

<code>start_value</code>	A positive number to start the return vector in time 0.
<code>alpha</code>	The alpha parameter in the two-parameter Ricker function. Must be non-negative.
<code>beta</code>	The beta parameter in the two-parameter Ricker function.
<code>time_steps</code>	The number of time steps to run the projection. Must be a positive integer.
<code>time_lag</code>	A positive integer denoting the number of time steps back for the value of phi in the two-parameter Ricker function.
<code>pre0_subs</code>	A logical value indicating whether to use a number other than that given in <code>start_value</code> for values of phi lagged from times prior to time 0.
<code>pre0_value</code>	A positive number to use for phi lagged from times prior to time 0. Only used if <code>pre0_subs = TRUE</code> .
<code>substoch</code>	An integer value indicating the kind of substochasticity to use. Values include: 0, no substochasticity enforced (the default); 1, all numbers must be non-negative; and 2, all numbers should be forced to the interval [0, 1].
<code>separate_N</code>	An optional numeric vector with values of N in each time, if phi is to be treated as different from N in the two-parameter model.

Value

A numeric vector of values showing values projected under the two- parameter Ricker function.

Examples

```
trial_run1 <- ricker3(1, alpha = 0.5, beta = -0.009)
plot(trial_run1)

trial_run2 <- ricker3(1, alpha = 0.5, beta = 0.009)
plot(trial_run2)

trial_run3 <- ricker3(1, alpha = 1, beta = -0.009)
plot(trial_run3)

trial_run4 <- ricker3(1, alpha = 1, beta = 0.009)
plot(trial_run4)

trial_run5 <- ricker3(1, alpha = 5, beta = -0.009)
plot(trial_run5)

trial_run6 <- ricker3(1, alpha = 5, beta = 0.009)
plot(trial_run6)

used_Ns <- c(10, 15, 12, 14, 14, 150, 15, 1, 5, 7, 9, 14, 13, 16, 17, 19,
  25, 26)
trial_run7 <- ricker3(1, alpha = 1, beta = -0.009, separate_N = used_Ns)
plot(trial_run7)
```

rlefko2

Create Raw Ahistorical Matrix Projection Model

Description

Function `rlefko2()` returns raw ahistorical MPMs, including the associated component transition and fecundity matrices, a data frame describing the ahistorical stages used, and a data frame describing the population, patch, and occasion time 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"),
sizeb = c(NA, NA),
sizec = c(NA, NA),
repst = c("repstatus3", "repstatus2"),
matst = c("matstatus3", "matstatus2"),
fec = c("feca3", "feca2"),
supplement = NULL,
repmatrix = NULL,
overwrite = NULL,
yearcol = NA,
popcol = NA,
patchcol = NA,
indivcol = NA,
censorcol = NA,
censorkeep = 0,
reduce = FALSE
)

```

Arguments

data	A vertical demographic data frame, with variables corresponding to the naming conventions in verticalize3() and historicalize3() .
stageframe	A stageframe object that includes information on the size, observation status, propagule status, reproduction status, immaturity status, and maturity status of each ahistorical stage.
year	A variable corresponding to observation occasion, or a set of such values, given in values associated with the year term used in vital rate model development. Can also equal "all", in which case matrices will be estimated for all occasion times. 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 NA, in which case patch designations are ignored..
censor	If TRUE, then data will be removed according to the variable set in censorcol, such that only data with censor values equal to censorkeep 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 occasions $t+1$ and t . The names of stages in these variables should match those used in the stageframe exactly. If left blank, then <code>rlefk2()</code> 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 occasions $t+1$ and t , respectively.

size	A vector of names of variables coding the primary size variable in occasions $t+1$ and t , respectively. Defaults to <code>c("sizea3", "sizea2")</code> .
sizeb	A vector of names of variables coding the secondary size variable in occasions $t+1$ and t , respectively. Defaults to <code>c(NA, NA)</code> .
sizec	A vector of names of variables coding the tertiary size variable in occasions $t+1$ and t , respectively. Defaults to <code>c(NA, NA)</code> .
repst	A vector of names of variables coding reproductive status in occasions $t+1$ and t , respectively. Defaults to <code>c("repstatus3", "repstatus2")</code> . Must be supplied if stages is not provided.
matst	A vector of names of variables coding maturity status in occasions $t+1$ and t , respectively. Defaults to <code>c("matstatus3", "matstatus2")</code> . Must be supplied if stages is not provided.
fec	A vector of names of variables coding fecundity in occasions $t+1$ and t , respectively. Defaults to <code>c("feca3", "feca2")</code> .
supplement	An optional data frame of class <code>lefk2SD</code> that provides supplemental data that should be incorporated into the MPM. Three kinds of data may be integrated this way: transitions to be estimated via the use of proxy transitions, transition overwrites from the literature or supplemental studies, and transition multipliers for fecundity. This data frame should be produced using the <code>supplemental()</code> function. Should be used in place of or in addition to an overwrite table (see <code>overwrite</code> below) and a reproduction matrix (see <code>repmatrix</code> below).
repmatrix	An optional reproduction matrix. This matrix is composed mostly of 0s, with non-zero entries acting as element identifiers and multipliers for fecundity (with 1 equaling full fecundity). If left blank, and no supplement is provided, then <code>rlefk2()</code> will assume that all stages marked as reproductive produce offspring at 1x that of estimated fecundity, and that offspring production will yield the first stage noted as propagule or immature. To prevent this behavior, input just <code>0</code> , which will result in fecundity being estimated only for transitions noted in supplement above. Must be the dimensions of an ahistorical matrix.
overwrite	An optional data frame developed with the <code>overwrite()</code> function describing transitions to be overwritten either with given values or with other estimated transitions. Note that this function supplements overwrite data provided in supplement.
yearcol	The variable name or column number corresponding to occasion 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.
sensorcol	The variable name or column number denoting the censor status. Only needed if <code>sensor = TRUE</code> .
sensorkeep	The value of the censor variable denoting data elements to keep. Defaults to <code>0</code> .
reduce	A logical value denoting whether to remove historical stages associated with only zero transitions. These are removed only if the respective row and column sums in ALL matrices estimated equal 0. Defaults to <code>FALSE</code> .

Value

If all inputs are properly formatted, then this function will return an object of class `lefkoMat`, which is a list that holds the matrix projection model and all of its metadata. Its structure is a list with the following elements:

<code>A</code>	A list of full projection matrices in order of sorted populations, patches, and occasions. All matrices output in the <code>matrix</code> class.
<code>U</code>	A list of survival transition matrices sorted as in <code>A</code> . All matrices output in the <code>matrix</code> class.
<code>F</code>	A list of fecundity matrices sorted as in <code>A</code> . All matrices output in the <code>matrix</code> class.
<code>hstages</code>	A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs. Set to <code>NA</code> for ahistorical matrices.
<code>agestages</code>	A data frame showing age-stage pairs. In this function, it is set to <code>NA</code> . Only used in output to function <code>aflefko2()</code> .
<code>ahstages</code>	A data frame detailing the characteristics of associated ahistorical stages, in the form of a modified stageframe that includes status as an entry stage through reproduction.
<code>labels</code>	A data frame giving the population, patch, and year of each matrix in order.
<code>matrixqc</code>	A short vector describing the number of non-zero elements in <code>U</code> and <code>F</code> matrices, and the number of annual matrices.
<code>dataqc</code>	A vector showing the numbers of individuals and rows in the vertical dataset used as input.

Notes

The default behavior of this function is to estimate fecundity with regards to transitions specified via associated fecundity multipliers in either `supplement` or `repmatrix`. If both of these fields are left empty, then fecundity will be estimated at full for all transitions leading from reproductive stages to immature and propagule stages. However, if a `supplement` is provided and a `repmatrix` is not, or if `repmatrix` is set to 0, then only fecundity transitions noted in the `supplement` will be set to non-zero values. To use the default behavior of setting all reproductive stages to reproduce at full fecundity into immature and propagule stages but also incorporate given or proxy survival transitions, input those given and proxy transitions through the `overwrite` options.

The reproduction matrix (field `repmatrix`) may only be supplied as ahistorical. If provided as historical, then `rlefko2()` will fail and produce an error.

Users may at times wish to estimate MPMs using a dataset incorporating multiple patches or subpopulations. Should the aim of analysis be a general MPM that does not distinguish these patches or subpopulations, the `patchcol` variable should be left to `NA`, which is the default. Otherwise the variable identifying patch needs to be named.

Input options including multiple variable names must be entered in the order of variables in occasion $t+1$ and t . Rearranging the order WILL lead to erroneous calculations, and may lead to fatal errors.

Although this function is capable of assigning stages given an input stageframe, it lacks the power of `verticalize3()` and `historicalize3()` in this regard. Users are strongly encouraged to use the latter two functions for stage assignment.

See Also

```

flefko3()
flefko2()
aflefko2()
arlefko2()
fleslie()
rlefko3()
rleslie()

```

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp2 <- supplemental(stage3 = c("Sd", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "rep", "rep"),
  givenrate = c(0.345, 0.054, NA, NA),
  multiplier = c(NA, NA, 0.345, 0.054),
  type = c(1, 1, 3, 3), stageframe = lathframe, historical = FALSE)

ehrlen2 <- rlefko2(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2"), supplement = lathsupp2, yearcol = "year2",
  individcol = "individ")

summary(ehrlen2)

# Cypridium example

```

```

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, 0, 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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cypmatrix2r$A[[1]]

```

Description

Function `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 occasion time 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"),
  sizeb = c(NA, NA, NA),
  sizec = c(NA, NA, NA),
  repst = c("repstatus3", "repstatus2", "repstatus1"),
  matst = c("matstatus3", "matstatus2", "matstatus1"),
  fec = c("feca3", "feca2", "feca1"),
  supplement = NULL,
  repmatrix = NULL,
  overwrite = NULL,
  yearcol = NA,
  popcol = NA,
  patchcol = NA,
  indivcol = NA,
  censorcol = NA,
  censorkeep = 0,
  format = "ehrlen",
  reduce = FALSE,
  err_check = FALSE
)
```

Arguments

<code>data</code>	A vertical demographic data frame, with variables corresponding to the naming conventions in verticalize3() and historicalize3() .
<code>stageframe</code>	A stageframe object that includes information on the size, observation status, propagule status, reproduction status, immaturity status, and maturity status of each ahistorical stage.
<code>year</code>	A variable corresponding to observation occasion, or a set of such values, given in values associated with the year term used in vital rate model development. Can also equal "all", in which case matrices will be estimated for all occasions. 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 NA, in which case patch designations are ignored..
censor	If TRUE, then data will be removed according to the variable set in censorcol, such that only data with censor values equal to censorkeep 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 occasions $t+1$, t , and $t-1$. The names of stages in these variables should match those used in the stageframe exactly. If left blank, then rlefk3() 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 occasions $t+1$, t , and $t-1$, respectively.
size	A vector of names of variables coding the primary size variable in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("sizea3", "sizea2", "sizea1").
sizeb	A vector of names of variables coding the secondary size variable in occasions $t+1$, t , and $t-1$, respectively. Defaults to c(NA, NA, NA).
sizec	A vector of names of variables coding the tertiary size variable in occasions $t+1$, t , and $t-1$, respectively. Defaults to c(NA, NA, NA).
repst	A vector of names of variables coding reproductive status in occasions $t+1$, t , and $t-1$, respectively. Defaults to c("repstatus3", "repstatus2", "repstatus1"). Must be supplied if stages is not provided.
matst	A vector of names of variables coding maturity status in occasions $t+1$, 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 occasions $t+1$, t , and $t-1$, respectively. Defaults to c("feca3", "feca2", "feca1").
supplement	An optional data frame of class lefkoSD that provides supplemental data that should be incorporated into the MPM. Three kinds of data may be integrated this way: transitions to be estimated via the use of proxy transitions, transition overwrites from the literature or supplemental studies, and transition multipliers for fecundity. This data frame should be produced using the supplemental() function. Should be used in place of or in addition to an overwrite table (see overwrite below) and a reproduction matrix (see repmatrix below).
repmatrix	An optional reproduction matrix. This matrix is composed mostly of 0s, with non-zero entries acting as element identifiers and multipliers for fecundity (with 1 equaling full fecundity). If left blank, and no supplement is provided, then rlefk3() will assume that all stages marked as reproductive produce offspring at 1x that of estimated fecundity, and that offspring production will yield the first stage noted as propagule or immature. To prevent this behavior, input just

	\emptyset , which will result in fecundity being estimated only for transitions noted in supplement above. May be the dimensions of either a historical or an ahistorical matrix. If the latter, then all stages will be used in occasion $t-1$ for each suggested ahistorical transition.
overwrite	An optional data frame developed with the <code>overwrite()</code> function describing transitions to be overwritten either with given values or with other estimated transitions. Note that this function supplements overwrite data provided in supplement.
yearcol	The variable name or column number corresponding to occasion 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.
ensorcol	The variable name or column number denoting the censor status. Only needed if <code>ensor = TRUE</code> .
ensorkeep	The value of the censor variable denoting data elements to keep. Defaults to \emptyset .
format	A string indicating whether to estimate matrices in <code>ehr1en</code> format or <code>deVries</code> format. The latter adds one unborn prior stage to account for the prior state of newborns. Defaults to <code>ehr1en</code> format.
reduce	A logical value denoting whether to remove historical stages associated exclusively with zero transitions. These are removed only if the respective row and column sums in ALL matrices estimated equal 0. Defaults to <code>FALSE</code> .
err_check	A logical value indicating whether to append extra information used in matrix calculation within the output list. Used for development debugging purposes. Defaults to <code>FALSE</code> .

Value

If all inputs are properly formatted, then this function will return an object of class `lefkoMat`, which is a list that holds the matrix projection model and all of its metadata. Its structure is a list with the following elements:

A	A list of full projection matrices in order of sorted populations, patches, and occasions. All matrices output in the <code>matrix</code> class.
U	A list of survival transition matrices sorted as in A. All matrices output in the <code>matrix</code> class.
F	A list of fecundity matrices sorted as in A. All matrices output in the <code>matrix</code> class.
hstages	A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs.
agestages	A data frame showing age-stage pairs. In this function, it is set to <code>NA</code> . Only used in output to function <code>aflefko2()</code> .
ahstages	A data frame detailing the characteristics of associated ahistorical stages, in the form of a modified stageframe that includes status as an entry stage through reproduction.

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.

Notes

The default behavior of this function is to estimate fecundity with regards to transitions specified via associated fecundity multipliers in either `supplement` or `repmatrix`. If both of these fields are left empty, then fecundity will be estimated at full for all transitions leading from reproductive stages to immature and propagule stages. However, if a `supplement` is provided and a `repmatrix` is not, or if `repmatrix` is set to 0, then only fecundity transitions noted in the `supplement` will be set to non-zero values. To use the default behavior of setting all reproductive stages to reproduce at full fecundity into immature and propagule stages but incorporate given or proxy survival transitions, input those given and proxy transitions through the `overwrite` option.

The reproduction matrix (field `repmatrix`) may be supplied as either historical or ahistorical. If provided as ahistorical, then `flefko3()` will assume that all historical transitions involving stages noted for occasions t and $t+1$ should be set to the respective fecundity multipliers noted.

Users may at times wish to estimate MPMs using a dataset incorporating multiple patches or subpopulations. Should the aim of analysis be a general MPM that does not distinguish these patches or subpopulations, the `patchcol` variable should be left to NA, which is the default. Otherwise the variable identifying patch needs to be named.

Input options including multiple variable names must be entered in the order of variables in occasion $t+1$, t , and $t-1$. Rearranging the order WILL lead to erroneous calculations, and may lead to fatal errors.

Although this function is capable of assigning stages given an input stageframe, it lacks the power of `verticalize3()` and `historicalize3()` in this regard. Users are strongly encouraged to use the latter two functions for stage assignment.

See Also

[flefko3\(\)](#)
[flefko2\(\)](#)
[aflefko2\(\)](#)
[arlefko2\(\)](#)
[fleslie\(\)](#)
[rlefko2\(\)](#)
[rleslie\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
```

```

stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk03(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

summary(ehrlen3)

# Cypripedium example
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, 0, 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)

cypsupp3r <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
  "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat", "mat", "SD", "P1"),
  stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
  "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
  stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "P3", "P3",
  "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D", "XSm", "Sm",
  "mat", "mat", "mat", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "D", "XSm", "Sm", NA, NA),
  eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.1, 0.1, 0.2, 0.2, 0.2, 0.25, NA, NA, NA, NA, NA, NA,
  NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
  NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  type_t12 = c(1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
  stageframe = cypframe_raw, historical = TRUE)

cypmatrix3r <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added", "size1added"),
  supplement = cypsupp3r, yearcol = "year2", patchcol = "patchid",
  individcol = "individ")

summary(cypmatrix3r)

```

Description

Function `rleslie()` returns raw Leslie MPMs, including the associated component transition and fecundity matrices, a data frame describing the ages used, and a data frame describing the population, patch, and occasion time associated with each matrix.

Usage

```
rleslie(
  data,
  start_age = NA,
  last_age = NA,
  continue = TRUE,
  fecage_min = NA,
  fecage_max = NA,
  alive = c("alive3", "alive2", "alive1"),
  repst = c("repstatus3", "repstatus2", "repstatus1"),
  fec = c("feca3", "feca2", "feca1"),
  agecol = "obsage",
  year = "all",
  pop = NA,
  patch = NA,
  yearcol = NA,
  popcol = NA,
  patchcol = NA,
  indivcol = NA,
  censor = FALSE,
  censorcol = NA,
  censorkeep = 0,
  fectime = 2,
  fecmod = 1,
  prebreeding = TRUE
)
```

Arguments

<code>data</code>	A vertical demographic data frame, with variables corresponding to the naming conventions in <code>verticalize3()</code> .
<code>start_age</code>	The age from which to start the matrix. Defaults to NA, age 1 is used if <code>prebreeding = TRUE</code> , and age 0 is used if <code>prebreeding = FALSE</code> .
<code>last_age</code>	The final age to use in the matrix. Defaults to NA, in which case the highest age in the dataset is used.
<code>continue</code>	A logical value designating whether to allow continued survival of individuals past the final age noted in the stageframe, using the demographic characteristics of the final age. Defaults to TRUE.
<code>fecage_min</code>	The minimum age at which reproduction is possible. Defaults to NA, which is interpreted to mean that fecundity should be assessed starting in the minimum age observed in the dataset.
<code>fecage_max</code>	The maximum age at which reproduction is possible. Defaults to NA, which is interpreted to mean that fecundity should be assessed until the final observed age.
<code>alive</code>	A vector of names of binomial variables corresponding to status as alive (1) or dead (0) in occasions $t+1$ and t , respectively.

repst	A vector of names of variables coding reproductive status in occasions $t+1$ and t , respectively. Defaults to <code>c("repstatus3", "repstatus2")</code> .
fec	A vector of names of variables coding fecundity in occasions $t+1$ and t , respectively. Defaults to <code>c("feca3", "feca2")</code> .
agecol	The name or column number of the variable coding for age in data. Defaults to <code>"obsage"</code> .
year	A variable corresponding to observation occasion, or a set of such values, given in values associated with the year term used in vital rate model development. Can also equal <code>"all"</code> , in which case matrices will be estimated for all occasion times. Defaults to <code>"all"</code> .
pop	A variable designating which populations will have matrices estimated. Should be set to specific population names, or to <code>"all"</code> 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 <code>"all"</code> if matrices should be estimated for all patches. Defaults to <code>"all"</code> .
yearcol	The variable name or column number corresponding to occasion 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.
ensor	If TRUE, then data will be removed according to the variable set in <code>ensorcol</code> , such that only data with censor values equal to <code>ensorkeep</code> will remain. Defaults to FALSE.
ensorcol	The variable name or column number denoting the censor status. Only needed if <code>ensor = TRUE</code> .
ensorkeep	The value of the censor variable denoting data elements to keep. Defaults to <code>0</code> .
fectime	An integer indicating whether to estimate fecundity using the variable given for <code>fec</code> in time t (2) or time $t+1$ (3).
fecmod	A scalar multiplier for fecundity. Defaults to <code>1.0</code> .
prebreeding	A logical value indicating whether the life history model is a pre-breeding model. Defaults to TRUE.

Value

If all inputs are properly formatted, then this function will return an object of class `lefkMat`, which is a list that holds the matrix projection model and all of its metadata. Its structure is a list with the following elements:

A	A list of full projection matrices in order of sorted populations, patches, and occasions. All matrices output in the <code>matrix</code> class.
U	A list of survival transition matrices sorted as in A. All matrices output in the <code>matrix</code> class.

F	A list of fecundity matrices sorted as in A. All matrices output in the <code>matrix</code> class.
hstages	A data frame matrix showing the pairing of ahistorical stages used to create historical stage pairs. Set to NA for ahistorical matrices.
agestages	A data frame showing age-stage pairs. In this function, it is set to NA. Only used in output to function <code>aflefk2()</code> .
ahstages	A data frame detailing the characteristics of associated ahistorical stages, in the form of a modified stageframe that includes status as an entry stage through reproduction.
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.

Notes

In order to accomodate survival to time $t+1$ in the final year of a study, the maximum age assessed if no input `last_age` is provided is one time step past the final described age.

Users may at times wish to estimate MPMs using a dataset incorporating multiple patches or sub-populations. Should the aim of analysis be a general MPM that does not distinguish these patches or subpopulations, the `patchcol` variable should be left to NA, which is the default. Otherwise the variable identifying patch needs to be named.

Input options including multiple variable names must be entered in the order of variables in occasion $t+1$ and t . Rearranging the order WILL lead to erroneous calculations, and may lead to fatal errors.

See Also

[flefk3\(\)](#)
[flefk2\(\)](#)
[aflefk2\(\)](#)
[arlefk2\(\)](#)
[fleslie\(\)](#)
[rlefk3\(\)](#)
[rlefk2\(\)](#)

Examples

```
# Cypripedium example
data(cypdata)

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",
```

```

NAas0 = TRUE, NRasRep = TRUE)

cyp_r1 <- rleslie(data = cypraw_v1, start_age = 0, last_age = 4, continue = TRUE,
  fecage_min = 3, fecage_max = 4, year = "all", pop = NA, patch = "all",
  yearcol = "year2", popcol = NA, patchcol = "patchid", indivcol = "individ")
cyp_r1

```

sensitivity3

Estimate Sensitivity of Population Growth Rate to Matrix Elements

Description

sensitivity3() is a generic function that returns the sensitivity of the population growth rate to the elements of the matrices in a matrix population model. Currently, this function estimates both deterministic and stochastic sensitivities, where the growth rate is λ in the former case and the log of the stochastic λ in the latter case. This function is made to handle very large and sparse matrices supplied as lefkoMat objects, as lists of matrices, and as individual matrices.

Usage

```
sensitivity3(mats, ...)
```

Arguments

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

Value

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

See Also

[sensitivity3.lefkoMat\(\)](#)
[sensitivity3.matrix\(\)](#)
[sensitivity3.list\(\)](#)

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "indiv")

ehrlen3mean <- lmean(ehrlen3)
sensitivity3(ehrlen3mean)

# Cypripedium example
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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", individcol = "individ")

sensitivity3(cypmatrix2r)

```

sensitivity3.lefkoMat *Estimate Sensitivity of Population Growth Rate of a lefkoMat Object*

Description

sensitivity3.lefkoMat() returns the sensitivities of population growth rate to elements of all \$A matrices in an object of class lefkoMat. If deterministic, then λ is taken as the population growth rate. If stochastic, then the log of stochastic λ , or the log stochastic growth rate, is taken as the population growth rate. 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,
  stochastic = FALSE,
  steps = 10000,
  time_weights = NA,
  sparse = "auto",
  append_mats = FALSE,

```

```
    ...
  )
```

Arguments

<code>mats</code>	An object of class <code>lefkoMat</code> .
<code>stochastic</code>	A logical value determining whether to conduct a deterministic (FALSE) or stochastic (TRUE) sensitivity analysis. Defaults to FALSE.
<code>steps</code>	The number of occasions to project forward in stochastic simulation. Defaults to 10000.
<code>time_weights</code>	Numeric vector denoting the probabilistic weightings of annual matrices. Defaults to equal weighting among occasions.
<code>sparse</code>	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
<code>append_mats</code>	A logical value indicating whether to include the original A, U, and F matrices in the output <code>lefkoSens</code> object.
<code>...</code>	Other parameters.

Value

This function returns an object of class `lefkoSens`, which is a list of 8 elements. The first, `h_sensmats`, is a list of historical sensitivity matrices (NULL if an ahMPM is used as input). The second, `ah_elasmats`, is a list of either ahistorical sensitivity matrices if an ahMPM is used as input, or, if an hMPM is used as input, then the result is a list of ahistorical matrices based on the equivalent historical dependencies assumed in the input historical matrices. The third element, `h_stages`, is a data frame showing historical stage pairs (NULL if an ahMPM used as input). The fourth element, `agestages`, show the order of age-stage combinations, if age-by-stage MPMs have been supplied. The fifth element, `ah_stages`, is a data frame showing the order of ahistorical stages. The last 3 elements are the A, U, and F portions of the input.

Notes

Deterministic sensitivities are estimated as eqn. 9.14 in Caswell (2001, *Matrix Population Models*). Stochastic sensitivities are estimated as eqn. 14.97 in Caswell (2001). Note that stochastic sensitivities are of the log of the stochastic λ .

See Also

```
sensitivity3()
sensitivity3.matrix()
sensitivity3.list()
```

Examples

```
# Lathyrus example
data(lathyrus)
```



```

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

sensitivity3(ehrlen3, stochastic = TRUE)

# Cypridium example
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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

sensitivity3(cypmatrix2r)

```

sensitivity3.list

Estimate Sensitivity of Population Growth Rate of a List of Matrices

Description

`sensitivity3.list()` returns the sensitivities of population growth rate to elements of matrices supplied in a list. The sensitivity analysis can be deterministic or stochastic, but if the latter then at least two *A* matrices must be included in the list. 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 'list'
sensitivity3(
  mats,
  stochastic = FALSE,
  steps = 10000,

```

```

    time_weights = NA,
    historical = FALSE,
    sparse = "auto",
    append_mats = FALSE,
    ...
  )

```

Arguments

<code>mats</code>	An object of class <code>matrix</code> .
<code>stochastic</code>	A logical value determining whether to conduct a deterministic (FALSE) or stochastic (TRUE) sensitivity analysis. Defaults to FALSE.
<code>steps</code>	The number of occasions to project forward in stochastic simulation. Defaults to 10,000.
<code>time_weights</code>	Numeric vector denoting the probabilistic weightings of annual matrices. Defaults to equal weighting among occasions.
<code>historical</code>	A logical value indicating whether matrices are historical. Defaults to FALSE.
<code>sparse</code>	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
<code>append_mats</code>	A logical value indicating whether to include the original matrices input as object <code>mats</code> in the output <code>lefkoSens</code> object. Defaults to FALSE.
<code>...</code>	Other parameters.

Value

This function returns an object of class `lefkoSens`, which is a list of 8 elements. The first, `h_sensmats`, is a list of historical sensitivity matrices (NULL if an `ahMPM` is used as input). The second, `ah_elasmats`, is a list of ahistorical sensitivity matrices if an `ahMPM` is used as input (NULL if an `hMPM` is used as input). The third element, `h_stages`, the fourth element, `age_stages`, and the fifth element, `ah_stages`, are NULL. The last 3 elements include the original `A` matrices supplied (as the `A` element), followed by NULLs for the `U` and `F` elements.

Notes

Deterministic sensitivities are estimated as eqn. 9.14 in Caswell (2001, *Matrix Population Models*). Stochastic sensitivities are estimated as eqn. 14.97 in Caswell (2001). Note that stochastic sensitivities are with regard to the log of the stochastic λ .

Currently, this function does not estimate equivalent ahistorical stochastic sensitivities for input historical matrices, due to the lack of guidance input on the order of stages (guidance is provided within `lefkMat` objects).

See Also

[sensitivity3\(\)](#)
[sensitivity3.lefkMat\(\)](#)
[sensitivity3.matrix\(\)](#)

Examples

```

# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

sensitivity3(ehrlen3$A)

# Cypripedium example
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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlfko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", individcol = "individ")

sensitivity3(cypmatrix2r$A)

```

sensitivity3.matrix *Estimate Sensitivity of Population Growth Rate of a Single Matrix*

Description

sensitivity3.matrix() returns the sensitivities of λ to elements of a single matrix. Because this handles only one matrix, the sensitivities are inherently deterministic and based on the dominant eigen value as the best metric of the population growth rate. 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, sparse = "auto", ...)

```

Arguments

<code>mats</code>	An object of class <code>matrix</code> .
<code>sparse</code>	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
<code>...</code>	Other parameters.

Value

This function returns a single deterministic sensitivity matrix.

See Also

[sensitivity3\(\)](#)
[sensitivity3.lefkoMat\(\)](#)
[sensitivity3.list\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
```

```

multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

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

# Cyripedium example
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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

```

```
sensitivity3(cypmatrix2r$A[[1]])
```

 sf_create

 Create Stageframe for Population Matrix Projection Analysis

Description

Function `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 including `flefko3()`, `flefko2()`, `aflefko2()`, `rlefko3()`, and `rlefko2()`, in which it determines how each stage is treated during matrix estimation.

Usage

```
sf_create(
  sizes,
  stagenames = NULL,
  sizesb = NULL,
  sizesc = NULL,
  repstatus = NULL,
  obsstatus = NULL,
  propstatus = NULL,
  matstatus = NULL,
  immstatus = NULL,
  minage = NULL,
  maxage = NULL,
  indataset = NULL,
  binhalfwidth = NULL,
  binhalfwidthb = NULL,
  binhalfwidthc = NULL,
  group = NULL,
  comments = NULL,
  roundsize = 5L,
  roundsizeb = 5L,
  roundsizec = 5L,
  ipmbins = 100L,
  ipmbinsb = NA_integer_,
  ipmbinsc = NA_integer_
)
```

Arguments

<code>sizes</code>	A numeric vector of the typical or representative size of each life history stage. If making function-based MPMs, then this should be a vector composed of the midpoints of each size bin. If denoting the boundary of an automated size classification group, then should denote the absolute minimum size of that group, or the absolute size of that group (see Notes).
--------------------	--

stagenames	A vector of stage names, in the same order as elements in sizes. Can also be set to ipm for automated size classification (see Notes section).
sizesb	An optional numeric vector for a second size metric for each life history stage. Only to be used if stages are defined by at least two size metrics in all cases. Same issues apply as in sizes.
sizesc	An optional numeric vector for a third size metric for each life history stage. Only to be used if stages are defined by at least three size metrics in all cases. Same issues apply as in sizes.
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 0.
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.
immstatus	A vector denoting whether each stage is immature. Must be composed of binomial values if given. Defaults to the complement of vector matstatus.
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 <code>rlefko2()</code> and <code>rlefko3()</code> can use all stages in the input dataset, <code>flefko3()</code> and <code>flefko2()</code> can only handle size-classified stages with non-overlapping combinations of size and status variables. 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.
binhalfwidthb	A numeric vector giving the half-width of size bins used for the optional second size metric. Required to classify individuals appropriately with two or three size classes. Defaults to 0.5 for all sizes.
binhalfwidthc	A numeric vector giving the half-width of size bins used for the optional third size metric. Required to classify individuals appropriately with three size classes. Defaults to 0.5 for all sizes.
group	An integer vector providing information on each respective stage's size classification group. If used, then function-based MPM creation functions <code>flefko2()</code> , <code>flefko3()</code> , and <code>aflefko2()</code> will estimate transitions only within these groups and for allowed cross-group transitions noted within the supplement table. Defaults to 0.
comments	An optional vector of text entries holding useful text descriptions of all stages.
roundsize	This parameter sets the precision of size classification, and equals the number of digits used in rounding sizes. Defaults to 5.

roundsizeb	This parameter sets the precision of size classification in the optional second size metric, and equals the number of digits used in rounding sizes. Defaults to 5.
roundsizec	This parameter sets the precision of size classification in the optional third size metric, and equals the number of digits used in rounding sizes. Defaults to 5.
ipmbins	An integer giving the number of size bins to create using the primary size classification variable. 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.
ipmbinsb	An optional integer giving the number of size bins to create using the secondary size classification variable. This number is in addition to any stages that are not size classified, as well as in addition to any automated size classification using the primary and tertiary size variables. Defaults to NA, and must be set to a positive integer for automated size classification to progress.
ipmbinsc	An optional integer giving the number of size bins to create using the tertiary size classification variable. This number is in addition to any stages that are not size classified, as well as in addition to any automated size classification using the primary and secondary size variables. Defaults to NA, and must be set to a positive integer for automated size classification to progress.

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, stage group classification, 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:

stage	The unique names of the stages to be analyzed.
size	The typical or representative size at which each stage occurs.
size_b	Size at which each stage occurs in terms of a second size variable, if one exists.
size_c	Size at which each stage occurs in terms of a third size variable, if one exists.
min_age	The minimum age at which the stage may occur.
max_age	The maximum age at which the stage may occur.
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.
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 midpoint of the size bin at which the stage may occur.
sizebin_width	The width of the size bin corresponding to the stage.
binhalfwidthb_raw	The half-width of the size bin of a second size variable, as input.
sizebinb_min	The minimum size at which the stage may occur.
sizebinb_max	The maximum size at which the stage may occur.
sizebinb_center	The midpoint of the size bin at which the stage may occur, in terms of a second size variable.
sizebinb_width	The width of the size bin corresponding to the stage, in terms of a second size variable.
binhalfwidthc_raw	The half-width of the size bin of a third size variable, as input.
sizebinc_min	The minimum size at which the stage may occur, in terms of a third size variable.
sizebinc_max	The maximum size at which the stage may occur, in terms of a third size variable.
sizebinc_center	The midpoint of the size bin at which the stage may occur, in terms of a third size variable.
sizebinc_width	The width of the size bin corresponding to the stage, in terms of a third size variable.
group	An integer denoting the size classification group that the stage falls within.
comments	A text field for stage descriptions.

Notes

If an IPM or function-based matrix with automated size classification is desired, then two stages that occur within the dataset and represent the lower and upper size limits of the IPM must be marked with `ipm` in the `stagenames` vector. These stages should have all characteristics other than size equal, and the size input for whichever size will be classified automatically must include the minimum in one stage and the maximum in the other. The actual characteristics of the first stage encountered in the inputs will be used as the template for the creation of these sizes. Note that `ipm` refers to size classification with the primary size variable. To automate size classification with the secondary size variable, use `ipmb`, and to automate size classification with the tertiary size variable, use `ipmc`. To nest automated size classifications, use `ipmab` for the primary and secondary size variables, `ipmac` for the primary and tertiary size variables, `ipmbc` for the secondary and tertiary size variables, and `ipmabc` for all three size variables. The primary size variable can also be set with `ipma`.

If two or more groups of stages, each with its own characteristics, are to be developed for an IPM or function-based MPM, then an even number of stages with two stages marking the minimum and maximum size of each group should be marked with the same code as given above, with all other characteristics equal within each group.

Stage classification groups set with the group variable create zones within function-based matrices in which survival transitions are estimated. These groups should not be set if transitions are possible between all stages regardless of group. To denote specific transitions as estimable between stage groups, use the `supplemental()` function.

If importing an IPM rather than building one with `lefko3`: Using the `vrn_input` approach to building function-based MPMs with provided linear model slope coefficients requires careful attention to the stageframe. Although no `hfv` data frame needs to be entered in this instance, stages for which vital rates are to be estimated via linear models parameterized with coefficients provided via function `vrn_import()` should be marked as occurring within the dataset, while stages for which the provided coefficients should not be used should be marked as not occurring within the dataset.

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")
```

```

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

# Cypridium example
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, 0, 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, 0, 1, 1, 1, 1, 1)
binvec <- c(0, 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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlfko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", individcol = "individ")

cyp2mean <- lmean(cypmatrix2r)
cyp2mean

```

sf_distrib

*Test Overdispersion and Zero Inflation in Size and Fecundity Distributions***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,
  sizea = NA,
  sizeb = NA,
  sizec = NA,
  obs3 = NA,
  fec = NA,
  repst = NA,
  zisizea = TRUE,
  zisizeb = TRUE,
  zisizec = TRUE,
  zifec = TRUE,
  fectime = 2,
  show.size = TRUE,
  show.fec = TRUE
)
```

Arguments

<code>data</code>	A historical vertical data file, which is a data frame of class <code>hfvdata</code> .
<code>sizea</code>	A vector holding the name or column number of the variables corresponding to primary size in occasions $t+1$ and t . Input only if <code>sizea</code> is to be tested.
<code>sizeb</code>	A vector holding the name or column number of the variables corresponding to secondary size in occasions $t+1$ and t . Input only if <code>sizeb</code> is to be tested.
<code>sizec</code>	A vector holding the name or column number of the variables corresponding to tertiary size in occasions $t+1$ and t . Input only if <code>sizec</code> is to be tested.
<code>obs3</code>	The name or column number of the variable corresponding to observation status in occasion $t+1$. This should be used if observation status will be used as a vital rate to absorb states of size = 0.
<code>fec</code>	A vector holding the names or column numbers of the variables corresponding to in occasions $t+1$ and t . Input only if <code>fec</code> is to be tested.
<code>repst</code>	A vector holding the names or column numbers of the variables corresponding to reproductive status in occasions $t+1$ and t . If not provided, then fecundity will be tested without subsetting to only reproductive individuals.

zysizea	A logical value indicating whether to conduct a test of zero inflation in primary size. Defaults to TRUE.
zysizeb	A logical value indicating whether to conduct a test of zero inflation in secondary size. Defaults to TRUE.
zysizec	A logical value indicating whether to conduct a test of zero inflation in tertiary size. Defaults to TRUE.
zifec	A logical value indicating whether to conduct a test of zero inflation in fecundity. Defaults to TRUE.
fectime	An integer indicating whether to treat fecundity as occurring in time t (2) or time $t+1$ (3). Defaults to 2.
show.size	A logical value indicating whether to show the output for tests of size. Defaults to TRUE.
show.fec	A logical value indicating whether to show the output for tests of fecundity. Defaults to TRUE.

Value

Produces text describing the degree and significance of difference from expected dispersion, and the degree and significance of 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.

Notes

This function subsets the data in the same way as `modelsearch()` before testing underlying distributions, making the output much more appropriate than a simple analysis of size and fecundity variables in data.

The specific test used for overdispersion is a chi-squared test of the dispersion parameter estimated using a generalized linear model predicting the response given size in occasion t , under a quasi-Poisson distribution.

The specific test used for zero-inflation is the chi-squared test presented in van der Broek (1995).

Examples

```
# Lathyrus example
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", "Sd1", "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)
obsvector <- c(0, 1, 0, 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)
immvector <- c(1, 1, 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)
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)

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

lathvertIn <- verticalize3(lathyrus, noyears = 4, firstyear = 1988,
  patchidcol = "SUBPLOT", individcol = "GENET", blocksize = 9,
  juvcol = "Seedling1988", sizeacol = "lnVol188", repstracol = "Intactseed88",
  fecacol = "Intactseed88", deadacol = "Dead1988",
  nonobsacol = "Dormant1988", stageassign = lathframeIn, stagesize = "sizea",
  censorcol = "Missing1988", censorkeep = NA, NAas0 = TRUE, censor = TRUE)

lathvertIn$fec2 <- round(lathvertIn$fec2)
lathvertIn$fec1 <- round(lathvertIn$fec1)
lathvertIn$fec3 <- round(lathvertIn$fec3)

# The following will only test fecundity, since size is Gaussian.
# Zero-inflation will not be assessed in this example, since 0 values in
# fecundity have been excluded in the life history model.

sf_distrib(lathvertIn, sizea = c("sizea3", "sizea2"), fec = c("fec3", "fec2"),
  repst = c("repstatus3", "repstatus2"), zifec = FALSE)

# Cypridium example
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, 0, 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, 0, 1, 1, 1, 1, 1)
binvec <- c(0, 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)

```



```
sf_distrib(cypraw_v1, sizea = c("size3added", "size2added"),
  fec = c("feca3", "feca2"), repst = c("repstatus3", "repstatus2"),
  zisizea = TRUE)
```

slambda3

*Estimate Stochastic Population Growth Rate***Description**

Function `slambda3()` estimates the stochastic population growth rate, a , defined as the long-term arithmetic mean of the log population growth rate estimated per simulated occasion. This function can handle both `lefkMat` objects and lists of full A matrices as input.

Usage

```
slambda3(
  mpm,
  times = 10000L,
  historical = FALSE,
  dense_only = FALSE,
  tweights = NULL
)
```

Arguments

<code>mpm</code>	A matrix projection model of class <code>lefkMat</code> , or a list of full matrix projection matrices.
<code>times</code>	Number of occasions to iterate. Defaults to 10000.
<code>historical</code>	An optional logical value only used if object <code>mpm</code> is a list of matrices, rather than a <code>lefkMat</code> object. Defaults to <code>FALSE</code> for the former case, and overridden by information supplied in the <code>lefkMat</code> object for the latter case.
<code>dense_only</code>	A logical value indicating whether to force matrices to be run in dense format. Defaults to <code>FALSE</code> , and should only be used if errors occur when running under default conditions.
<code>tweights</code>	Numeric vector denoting the probabilistic weightings of annual matrices. Defaults to equal weighting among occasions.

Value

A data frame with the following variables:

<code>pop</code>	The identity of the population.
<code>patch</code>	The identity of the patch.
<code>a</code>	Estimate of stochastic growth rate, estimated as the arithmetic mean of the log population growth rate across simulated occasions.

var	The estimated variance of a .
sd	The standard deviation of a .
se	The standard error of a .

Notes

The log stochastic population growth rate, a , is as given in equation 2 of Tuljapurkar, Horvitz, and Pascarella 2003. This term is estimated via projection of randomly sampled matrices, similarly to the procedure outlined in Box 7.4 of Morris and Doak (2002).

Stochastic growth rate is estimated both at the patch level and at the population level. Population level estimates will be noted at the end of the data frame with 0 entries for patch designation.

Weightings given in `tweights` do not need to sum to 1. Final weightings used will be based on the proportion per element of the sum of elements in the user-supplied vector.

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep"),
  stage1 = c("Sd", "rep", "Sd", "rep", "all", "all"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054),
  type = c(1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfko3(data = lathvert, stageframe = lathframe,
  year = c(1989, 1990), stages = c("stage3", "stage2", "stage1"),
```

```

    supplement = lathsupp3, yearcol = "year2", individcol = "individ")

slambda3(ehrlen3)

# Cypripedium example
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, 0, 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, 0, 1, 1, 1, 1, 1)
binvec <- c(0, 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)

cypsupp3r <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
  "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat", "mat", "SD", "P1"),
  stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
  "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
  stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "P3", "P3",
  "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D", "XSm", "Sm",
  "mat", "mat", "mat", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "D", "XSm", "Sm", NA, NA),
  eststage1 = c(NA, NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.1, 0.1, 0.2, 0.2, 0.2, 0.2, 0.25, NA, NA, NA, NA, NA, NA,
  NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
  NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  type_t12 = c(1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
  stageframe = cypframe_raw, historical = TRUE)

cypmatrix3r <- rlefk3(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added", "size1added"),

```

```

supplement = cypsupp3r, yearcol = "year2",
patchcol = "patchid", indivcol = "individ")

cypstoch <- slambda3(cypmatrix3r, dense_only = TRUE)
cypstoch

```

stablestage3

Estimate Stable Stage Distribution

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 ahistorical matrices.

Usage

```
stablestage3(mats, ...)
```

Arguments

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

Value

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

See Also

[stablestage3.lefkoMat\(\)](#)
[stablestage3.matrix\(\)](#)

Examples

```

# Lathyrus deterministic example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

ehrlen3mean <- lmean(ehrlen3)
stablestage3(ehrlen3mean)

# Cypripedium stochastic example
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, 0, 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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsups2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
    "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

stablestage3(cypmatrix2r, stochastic = TRUE)

```

stablestage3.lefkoMat *Estimate Stable Stage Distribution of Matrices in lefkoMat Object*

Description

stablestage3.lefkoMat() returns the deterministic stable stage distributions of all A matrices in an object of class lefkoMat, as well as the long-run projected mean stage distribution in stochastic analysis. 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 ahistorical matrices.

Usage

```

## S3 method for class 'lefkoMat'
stablestage3(
  mats,
  stochastic = FALSE,
  times = 10000,
  tweights = NA,
  seed = NA,
  sparse = "auto",
  ...
)

```

Arguments

<code>mats</code>	An object of class <code>lefkoMat</code> .
<code>stochastic</code>	A logical value indicating whether to use deterministic (FALSE) or stochastic (TRUE) analysis. Defaults to FALSE.
<code>times</code>	An integer variable indicating number of occasions to project if using stochastic analysis. Defaults to 10000.
<code>tweights</code>	An optional vector indicating the probability weighting to use for each matrix in stochastic simulations. If not given, then defaults to equal weighting.
<code>seed</code>	A number to use as a random number seed.
<code>sparse</code>	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
<code>...</code>	Other parameters.

Value

This function returns the stable stage distributions (and long-run mean stage distributions in stochastic analysis) corresponding to the matrices in a `lefkoMat` object.

The output depends on whether the `lefkoMat` object used as input is ahistorical or historical, and whether the analysis is deterministic or stochastic. If deterministic and ahistorical, 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 stochastic and ahistorical, then a single data frame is output starting with the number of the population-patch (`matrix_set`), a string concatenating the names of the population and the patch (`poppatch`), the assigned stage id number (`stage_id`), and the stage name (`stage`), and the long-run average 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 describes the historical stage-pair distribution, while the `ahist` element describes the stage distribution. If deterministic, then `hist` contains a data frame including the matrix number (`matrix`), the numeric stage designations for stages in occasions t and $t-1$, (`stage_id_2` and `stage_id_1`, respectively), followed by the respective stage names (`stage_2` and `stage_1`), and ending with the estimated stable stage-pair distribution. The associated `ahist` element is as before. If stochastic, then the `hist` element contains a single data frame with the number of the population-patch (`matrix_set`), a string concatenating the names of the population and the patch (`poppatch`), the assigned stage id numbers in times t and $t-1$ (`stage_id_2` and `stage_id_2`, respectively), and the associated stage names (`stage_2` and `stage_1`, respectively), and the long-run average stage distribution (`ss_prop`). The associated `ahist` element is as before in the ahistorical, stochastic case.

In addition to the data frames noted above, stochastic analysis will result in the additional output of a list of matrices containing the actual projected stage distributions across all projected occasions, in the order of population-patch combinations in the `lefkoMat` input.

Notes

In stochastic analysis, the projected mean distribution is the arithmetic mean across the final 1000 projected occasions if the simulation is at least 2000 projected occasions long. If between 500 and 2000 projected occasions long, then only the final 200 are used, and if fewer than 500 occasions are

used, then all are used. Note that because stage distributions in stochastic simulations can change greatly in the initial portion of the run, we encourage a minimum of 2000 projected occasions per simulation, with 10000 preferred.

See Also

[stablestage3\(\)](#)
[stablestage3.matrix\(\)](#)

Examples

```
# Lathyrus deterministic example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "indiv")

ehrlen3mean <- lmean(ehrlen3)
```



```

stablestage3(ehrlen3mean)

# Cypripedium stochastic example
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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsups2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlfko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

stablestage3(cypmatrix2r, stochastic = TRUE)

```

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, sparse = "auto", ...)
```

Arguments

mats	A population projection matrix of class matrix.
sparse	A text string indicating whether to use sparse matrix encoding ("yes") or dense matrix encoding ("no"). Defaults to "auto".
...	Other parameters.

Value

This function returns the stable stage distribution corresponding to the input matrix.

See Also

[stablestage3\(\)](#)
[stablestage3.lefkoMat\(\)](#)

Examples

```
#Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfeko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

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

# Cypripedium stochastic example
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, 0, 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)

# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),

```

```

eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
type =c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
stageframe = cyframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cyframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsups2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

stablestage3(cypmatrix2r$A[[1]])

```

start_input

Create a Starting Vector for Population Projection

Description

Function `start_input()` creates a data frame summarizing the non-zero elements of the start vector for use in population projection analysis via function `projection3()`.

Usage

```
start_input(mpm, stage2, stage1 = NA, age2 = NA, value = 1)
```

Arguments

mpm	The <code>lefkMat</code> object to be used in projection analysis.
stage2	A vector showing the name or number of a stage in occasion t that should be set to a positive number of individuals in the start vector. Abbreviations for groups of stages are also usable (see Notes). This input is required and has no default input.
stage1	A vector showing the name or number of a stage in occasion $t-1$ that should be set to a positive number of individuals in the start vector. Abbreviations for groups of stages are also usable (see Notes). This is only used for historical MPMs, since the rows of hMPMs correspond to stage-pairs in times t and $t-1$ together. Only required for historical MPMs, and will result in errors if otherwise used.
age2	A vector showing the age of each respective stage in occasion t that should be set to a positive number of individuals in the start vector. Only used for age-by-stage MPMs. Defaults to NA.
value	A vector showing the values, in order, of the number of individuals set for the stage or stage-pair in question. Defaults to 1.

Value

A list of class `lefkoStart`, with 4 objects, which can be used as input in function `projection3()`. The last three include the `ahstages`, `hstages`, and `agestages` objects from the `lefkoMat` object supplied in `mpm`. The first element in the list is a data frame with the following variables:

<code>stage2</code>	Stage at occasion t .
<code>stage_id_2</code>	The stage number associated with <code>stage2</code> .
<code>stage1</code>	Stage at occasion $t-1$, if historical. Otherwise NA.
<code>stage_id_1</code>	The stage number associated with <code>stage1</code> .
<code>age2</code>	The age of individuals in <code>stage2</code> and, if applicable, <code>stage1</code> . Only used in age-by-stage MPMs.
<code>row_num</code>	A number indicating the respective starting vector element.
<code>value</code>	Number of individuals in corresponding stage or stage-pair.

Notes

Entries in `stage2`, and `stage1` can include abbreviations for groups of stages. Use `rep` if all reproductive stages are to be used, `nrep` if all mature but non-reproductive stages are to be used, `mat` if all mature stages are to be used, `immat` if all immature stages are to be used, `prop` if all propagule stages are to be used, `npr` if all non-propagule stages are to be used, `obs` if all observable stages are to be used, `nobs` if all unobservable stages are to be used, and leave empty or use `all` if all stages in `stageframe` are to be used.

See Also

[density_input\(\)](#)

[projection3\(\)](#)

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "indiv")

ehrlen3mean <- lmean(ehrlen3)

e3m_sv <- start_input(ehrlen3mean, stage2 = "Sd", stage1 = "Sd", value = 1000)

lathproj <- projection3(ehrlen3, nreps = 5, times = 100, stochastic = TRUE,
  start_frame = e3m_sv)

```

subset_LM

Create New lefkoMat Object as Subset of Another lefkoMat Object

Description

Function `subset_LM()` creates a new `lefkoMat` object from a subset of matrices in another `lefkoMat` object.

Usage

```
subset_LM(lM, mat_num = NA, pop = NA, patch = NA, year = NA)
```

Arguments

<code>lM</code>	The <code>lefkoMat</code> object to select matrices from.
<code>mat_num</code>	Either a single integer corresponding to the matrix to select within the labels element of <code>lM</code> , or a vector of such integers.
<code>pop</code>	The population designation for matrices to select. Only used if <code>mat_num</code> is not given.

patch	The patch designation for matrices to select. Only used if mat_num is not given.
year	The time t designation for matrices to select. Only used if mat_num is not given.

Value

A lefkoMat object composed of the matrices specified in the options.

Notes

If mat_num is not provided, then at least one of pop, patch, or year must be provided. If at least two of pop, patch, and year are provided, then function subset_IM() will identify matrices as the intersection of provided inputs.

See Also

[create_IM\(\)](#)
[add_IM\(\)](#)
[delete_IM\(\)](#)

Examples

```
# These matrices are of 9 populations of the plant species Anthyllis
# vulneraria, and were originally published in Davison et al. (2010) Journal
# of Ecology 98:255-267 (doi: 10.1111/j.1365-2745.2009.01611.x).

sizevector <- c(1, 1, 2, 3) # These sizes are not from the original paper
stagevector <- c("Sd1", "Veg", "SmFlo", "LFlo")
repvector <- c(0, 0, 1, 1)
obsvector <- c(1, 1, 1, 1)
matvector <- c(0, 1, 1, 1)
immvector <- c(1, 0, 0, 0)
propvector <- c(0, 0, 0, 0)
indataset <- c(1, 1, 1, 1)
binvec <- c(0.5, 0.5, 0.5, 0.5)

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

# POPN C 2003-2004
XC3 <- matrix(c(0, 0, 1.74, 1.74,
0.208333333, 0, 0, 0.057142857,
0.041666667, 0.076923077, 0, 0,
0.083333333, 0.076923077, 0.066666667, 0.028571429), 4, 4, byrow = TRUE)

# 2004-2005
XC4 <- matrix(c(0, 0, 0.3, 0.6,
0.32183908, 0.142857143, 0, 0,
0.16091954, 0.285714286, 0, 0,
0.252873563, 0.285714286, 0.5, 0.6), 4, 4, byrow = TRUE)
```

```
# 2005-2006
XC5 <- matrix(c(0, 0, 0.50625, 0.675,
0, 0, 0, 0.035714286,
0.1, 0.068965517, 0.0625, 0.107142857,
0.3, 0.137931034, 0, 0.071428571), 4, 4, byrow = TRUE)

# POPN E 2003-2004
XE3 <- matrix(c(0, 0, 2.44, 6.569230769,
0.196428571, 0, 0, 0,
0.125, 0.5, 0, 0,
0.160714286, 0.5, 0.133333333, 0.076923077), 4, 4, byrow = TRUE)

XE4 <- matrix(c(0, 0, 0.45, 0.646153846,
0.06557377, 0.090909091, 0.125, 0,
0.032786885, 0, 0.125, 0.076923077,
0.049180328, 0, 0.125, 0.230769231), 4, 4, byrow = TRUE)

XE5 <- matrix(c(0, 0, 2.85, 3.99,
0.083333333, 0, 0, 0,
0, 0, 0, 0,
0.416666667, 0.1, 0, 0.1), 4, 4, byrow = TRUE)

# POPN F 2003-2004
XF3 <- matrix(c(0, 0, 1.815, 7.058333333,
0.075949367, 0, 0.05, 0.083333333,
0.139240506, 0, 0, 0.25,
0.075949367, 0, 0, 0.083333333), 4, 4, byrow = TRUE)

XF4 <- matrix(c(0, 0, 1.233333333, 7.4,
0.223880597, 0, 0.111111111, 0.142857143,
0.134328358, 0.272727273, 0.166666667, 0.142857143,
0.119402985, 0.363636364, 0.055555556, 0.142857143), 4, 4, byrow = TRUE)

XF5 <- matrix(c(0, 0, 1.06, 3.372727273,
0.073170732, 0.025, 0.033333333, 0,
0.036585366, 0.15, 0.1, 0.136363636,
0.06097561, 0.225, 0.166666667, 0.272727273), 4, 4, byrow = TRUE)

# POPN G 2003-2004
XG3 <- matrix(c(0, 0, 0.245454545, 2.1,
0, 0, 0.045454545, 0,
0.125, 0, 0.090909091, 0,
0.125, 0, 0.090909091, 0.333333333), 4, 4, byrow = TRUE)

XG4 <- matrix(c(0, 0, 1.1, 1.54,
0.111111111, 0, 0, 0,
0, 0, 0, 0,
0.111111111, 0, 0, 0), 4, 4, byrow = TRUE)

XG5 <- matrix(c(0, 0, 0, 1.5,
0, 0, 0, 0,
0.090909091, 0, 0, 0,
```



```
0.545454545, 0.5, 0, 0.5), 4, 4, byrow = TRUE)

# POPN L 2003-2004
XL3 <- matrix(c(0, 0, 1.785365854, 1.856521739,
0.128571429, 0, 0, 0.010869565,
0.028571429, 0, 0, 0,
0.014285714, 0, 0, 0.02173913), 4, 4, byrow = TRUE)

XL4 <- matrix(c(0, 0, 14.25, 16.625,
0.131443299, 0.057142857, 0, 0.25,
0.144329897, 0, 0, 0,
0.092783505, 0.2, 0, 0.25), 4, 4, byrow = TRUE)

XL5 <- matrix(c(0, 0, 0.594642857, 1.765909091,
0, 0, 0.017857143, 0,
0.021052632, 0.018518519, 0.035714286, 0.045454545,
0.021052632, 0.018518519, 0.035714286, 0.068181818), 4, 4, byrow = TRUE)

# POPN O 2003-2004
XO3 <- matrix(c(0, 0, 11.5, 2.775862069,
0.6, 0.285714286, 0.333333333, 0.24137931,
0.04, 0.142857143, 0, 0,
0.16, 0.285714286, 0, 0.172413793), 4, 4, byrow = TRUE)

XO4 <- matrix(c(0, 0, 3.78, 1.225,
0.28358209, 0.171052632, 0, 0.166666667,
0.084577114, 0.026315789, 0, 0.055555556,
0.139303483, 0.447368421, 0, 0.305555556), 4, 4, byrow = TRUE)

XO5 <- matrix(c(0, 0, 1.542857143, 1.035616438,
0.126984127, 0.105263158, 0.047619048, 0.054794521,
0.095238095, 0.157894737, 0.19047619, 0.082191781,
0.111111111, 0.223684211, 0, 0.356164384), 4, 4, byrow = TRUE)

# POPN Q 2003-2004
XQ3 <- matrix(c(0, 0, 0.15, 0.175,
0, 0, 0, 0,
0, 0, 0, 0,
1, 0, 0, 0), 4, 4, byrow = TRUE)

XQ4 <- matrix(c(0, 0, 0, 0.25,
0, 0, 0, 0,
0, 0, 0, 0,
1, 0.666666667, 0, 1), 4, 4, byrow = TRUE)

XQ5 <- matrix(c(0, 0, 0, 1.428571429,
0, 0, 0, 0.142857143,
0.25, 0, 0, 0,
0.25, 0, 0, 0.571428571), 4, 4, byrow = TRUE)

# POPN R 2003-2004
XR3 <- matrix(c(0, 0, 0.7, 0.6125,
0.25, 0, 0, 0.125,
```

```

0, 0, 0, 0,
0.25, 0.166666667, 0, 0.25), 4, 4, byrow = TRUE)

XR4 <- matrix(c(0, 0, 0, 0.6,
0.285714286, 0, 0, 0,
0.285714286, 0.333333333, 0, 0,
0.285714286, 0.333333333, 0, 1), 4, 4, byrow = TRUE)

XR5 <- matrix(c(0, 0, 0.7, 0.6125,
0, 0, 0, 0,
0, 0, 0, 0,
0.333333333, 0, 0.333333333, 0.625), 4, 4, byrow = TRUE)

# POPN S 2003-2004
XS3 <- matrix(c(0, 0, 2.1, 0.816666667,
0.166666667, 0, 0, 0,
0, 0, 0, 0,
0, 0, 0, 0.166666667), 4, 4, byrow = TRUE)

XS4 <- matrix(c(0, 0, 0, 7,
0.333333333, 0.5, 0, 0,
0, 0, 0, 0,
0.333333333, 0, 0, 1), 4, 4, byrow = TRUE)

XS5 <- matrix(c(0, 0, 0, 1.4,
0, 0, 0, 0,
0, 0, 0, 0.2,
0.111111111, 0.75, 0, 0.2), 4, 4, byrow = TRUE)

mats_list <- list(XC3, XC4, XC5, XE3, XE4, XE5, XF3, XF4, XF5, XG3, XG4, XG5,
  XL3, XL4, XL5, X03, X04, X05, XQ3, XQ4, XQ5, XR3, XR4, XR5, XS3, XS4, XS5)

yr_ord <- c(1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1,
  2, 3, 1, 2, 3)

pch_ord <- c(1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 6, 7, 7, 7,
  8, 8, 8, 9, 9, 9)

anth_lefkoMat <- create_lm(mats_list, anthframe, hstages = NA, historical = FALSE,
  poporder = 1, patchorder = pch_ord, yearorder = yr_ord)

smaller_anth_lm <- subset_lm(anth_lefkoMat, patch = c(1, 2, 3),
  year = c(1, 2))
smaller_anth_lm

```

Description

This function provides basic information summarizing the characteristics of conditional matrices derived from a lefkoCondMat object.

Usage

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

Arguments

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

Value

A text summary of the object shown on the console, showing the number of historical matrices, as well as the number of conditional matrices nested within each historical matrix.

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
```

```

eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

lathcondmats <- cond_hmpm(ehrlen3)
summary(lathcondmats)

# Cypripedium example
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)

cypsupp3r <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
  "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat", "mat", "SD", "P1"),
  stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
  "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
  stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "P3", "P3",
  "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D", "XSm", "Sm",
  "mat", "mat", "mat", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "D", "XSm", "Sm", NA, NA),
  eststage1 = c(NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.1, 0.1, 0.2, 0.2, 0.2, 0.2, 0.25, NA, NA, NA, NA, NA, NA,
  NA, NA, NA, NA, NA),

```

```

multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
  NA, 0.5, 0.5),
type = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
type_t12 = c(1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
stageframe = cypraw_v1, historical = TRUE)

cypmatrix3r <- rlefko3(data = cypraw_v1, stageframe = cypraw_v1,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added", "size1added"),
  supplement = cypsupsup3r, yearcol = "year2", patchcol = "patchid",
  indivcol = "indiv")

cypcondmats <- cond_hmpm(cypmatrix3r)

summary(cypcondmats)

```

```
summary.lefkoElas      Summarize lefkoElas Objects
```

Description

Function `summary.lefkoElas()` summarizes `lefkoElas` objects. Particularly, it breaks down elasticity values by the kind of ahistorical and, if applicable, historical transition.

Usage

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

Arguments

```
object      A lefkoElas object.
...         Other parameters currently not utilized.
```

Value

A list composed of 2 data frames. The first, `hist`, is a data frame showing the summed elasticities for all 16 kinds of historical transition per matrix, with each column corresponding to each elasticity matrix in order. The second, `ahist`, is a data frame showing the summed elasticities for all 4 kinds of ahistorical transition per matrix, with each column corresponding to each elasticity matrix in order.

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
```

```

stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

lathsupp2 <- supplemental(stage3 = c("Sd", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "rep", "rep"),
  givenrate = c(0.345, 0.054, NA, NA),
  multiplier = c(NA, NA, 0.345, 0.054),
  type = c(1, 1, 3, 3), stageframe = lathframe, historical = FALSE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "indiv")

ehrlen2 <- rlefk2(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2"), supplement = lathsupp2,
  yearcol = "year2", indivcol = "indiv")

ehrlen3elas <- elasticity3(ehrlen3)
ehrlen2elas <- elasticity3(ehrlen2)

summary(ehrlen3elas)
summary(ehrlen2elas)

```

summary.lefkoLTRE	<i>Summarize lefkoLTRE Objects</i>
-------------------	------------------------------------

Description

Function `summary.lefkoLTRE()` summarizes `lefkoLTRE` objects. Particularly, it breaks down LTRE contributions by the kind of ahistorical and, if applicable, historical transition.

Usage

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

Arguments

<code>object</code>	A <code>lefkoLTRE</code> object.
<code>...</code>	Other parameters currently not utilized.

Value

A list composed of 2 (if deterministic) or 4 (if stochastic) data frames. If deterministic, then `hist_det` is a data frame showing the summed LTRE contributions for all 16 kinds of historical transition per matrix, with each column corresponding to each A matrix in order, followed by all summed positive and all summed negative contributions. Object `ahist_det` is a data frame showing the summed LTRE contributions for all 4 kinds of ahistorical transition per matrix, with order as before, followed by summed positive and summed negative contributions. If stochastic, then `hist_mean` and `hist_sd` are the summed LTRE contributions for the mean vital rates and variability in vital rates, respectively, according to all 16 historical transition types, followed by summed positive and negative contributions, and `ahist_mean` and `ahist_sd` are the equivalent ahistorical versions.

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

lathsupp2 <- supplemental(stage3 = c("Sd", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "rep", "rep"),
  givenrate = c(0.345, 0.054, NA, NA),
  multiplier = c(NA, NA, 0.345, 0.054),
  type = c(1, 1, 3, 3), stageframe = lathframe, historical = FALSE)

ehrlen3 <- rlefko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

ehrlen2 <- rlefko2(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2"), supplement = lathsupp2,
  yearcol = "year2", indivcol = "individ")

ehrlen3ltre <- ltre3(ehrlen3)
summary(ehrlen3ltre)

```

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()`, `rlefko2()`, `aflefko2()`, `rleslie()`, and `fleslie()`.

Usage

```

## S3 method for class 'lefkoMat'
summary(object, colsums = TRUE, ...)

```


Arguments

object	An object of class lefkoMat.
colsums	A logical value indicating whether column sums should be shown for U matrices, allowing users to check stage survival probabilities. Defaults to TRUE.
...	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, the number of individuals, and summaries of the column sums of the survival-transition matrices. This function will also yield warnings if any survival-transition matrices include elements outside of the interval [0,1], if any fecundity matrices contain negative elements, and if any matrices include NA values.

Notes

Under the Gaussian and gamma size distributions, the number of estimated parameters may differ between the two ipm_method settings. Because the midpoint method has a tendency to incorporate upward bias in the estimation of size transition probabilities, it is more likely to yield non-zero values when the true probability is extremely close to 0. This will result in the summary.lefkoMat function yielding higher numbers of estimated parameters than the ipm_method = "CDF" yields in some cases.

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, 0, 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)

```

```
# Here we use supplemental() to provide overwrite and reproductive info
cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

summary(cypmatrix2r)
```

summary.lefkoMod

Summary of Class "lefkoMod"

Description

A function to summarize objects 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 currently not utilized.

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 numbers of individuals and of individual transitions used to estimate each vital rate best-fit model, along with the accuracy of each binomial model.

Examples

```

# Lathyrus example
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", "Sd1", "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, 0, 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)
matvector <- c(0, 0, 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)
propvector <- c(1, 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)
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)

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

lathvertIn <- 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 = lathframeIn,
stagesize = "sizea", censorcol = "Missing1988", censorkeep = NA,
NAas0 = TRUE, censor = TRUE)

lathvertIn$fecac2 <- round(lathvertIn$fecac2)
lathvertIn$fecac1 <- round(lathvertIn$fecac1)
lathvertIn$fecac3 <- round(lathvertIn$fecac3)

lathmodelsIn2 <- modelsearch(lathvertIn, historical = FALSE,
approach = "mixed", suite = "main",
vitalrates = c("surv", "obs", "size", "repst", "fec"), juvestimate = "Sd1",
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(lathmodelsIn2)

```

Description

Function `summary.lefkoProj()` summarizes `lefkoProj` objects. Particularly, it breaks down the data frames provided in the `projection` element in ways meaningful for those running simulations.

Usage

```
## S3 method for class 'lefkoProj'
summary(
  object,
  threshold = 1,
  inf_alive = TRUE,
  milepost = c(0, 0.25, 0.5, 0.75, 1),
  ext_time = FALSE,
  ...
)
```

Arguments

<code>object</code>	A <code>lefkoProj</code> object.
<code>threshold</code>	A threshold population size to be searched for in projections. Defaults to 1.
<code>inf_alive</code>	A logical value indicating whether to treat infinitely large population size as indicating that the population is still extant. If <code>FALSE</code> , then the population is considered extinct. Defaults to <code>TRUE</code> .
<code>milepost</code>	A numeric vector indicating at which points in the projection to assess detailed results. Can be input as integer values, in which case each number must be between 1 and the total number of occasions projected in each projection, or decimals between 0 and 1, which would then be translated into the corresponding projection steps of the total. Defaults to <code>c(0, 0.25, 0.50, 0.75, 1.00)</code> .
<code>ext_time</code>	A logical value indicating whether to output extinction times per population-patch. Defaults to <code>FALSE</code> .
<code>...</code>	Other parameters currently not utilized.

Value

Apart from a statement of the results, this function outputs a list with the following elements:

<code>milepost_sums</code>	A data frame showing the number of replicates at each of the milepost times that is above the threshold population/patch size.
<code>extinction_times</code>	A dataframe showing the numbers of replicates going extinct (<code>ext_reps</code>) and mean extinction time (<code>ext_time</code>) per population-patch. If <code>ext_time = FALSE</code> , then only outputs NA.

Notes

The `inf_alive` option assesses whether replicates have reached a value of `NaN`. If `inf_alive = TRUE` and a value of `NaN` is found, then the replicate is considered extant if the preceding value is

above the extinction threshold. If the setting is `inf_alive = FALSE`, then a value of `NaN` is considered evidence of extinction.

Extinction time is calculated on the basis of whether the replicate ever falls below a single individual. A replicate with a positive population size below 0.0 that manages to rise above 1.0 individual is still considered to have gone extinct the first time it crossed below 1.0.

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "VSm", "Sm", "VL", "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

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep"),
  stage1 = c("Sd", "rep", "Sd", "rep", "all", "all"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054),
  type = c(1, 1, 1, 1, 3, 3), type_t12 = c(1, 2, 1, 2, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlefk3(data = lathvert, stageframe = lathframe,
  year = c(1989, 1990), stages = c("stage3", "stage2", "stage1"),
  repmatrix = lathrepm, supplement = lathsupp3, yearcol = "year2",
  indivcol = "indiv")

lathproj <- projection3(ehrlen3, nreps = 5, stochastic = TRUE)
summary(lathproj)
```

```

# Cypripedium example
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)

cypsupp3r <- supplemental(stage3 = c("SD", "SD", "P1", "P1", "P2", "P3", "SL",
  "D", "XSm", "Sm", "D", "XSm", "Sm", "mat", "mat", "mat", "SD", "P1"),
  stage2 = c("SD", "SD", "SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "SL",
  "SL", "SL", "D", "XSm", "Sm", "rep", "rep"),
  stage1 = c("SD", "rep", "SD", "rep", "SD", "P1", "P2", "P3", "P3", "P3",
  "SL", "SL", "SL", "SL", "SL", "SL", "mat", "mat"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, NA, "D", "XSm", "Sm", "D", "XSm", "Sm",
  "mat", "mat", "mat", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "D", "XSm", "Sm", NA, NA),
  eststage1 = c(NA, NA, NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", "XSm", "XSm",
  "XSm", "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.1, 0.1, 0.2, 0.2, 0.2, 0.2, 0.25, NA, NA, NA, NA, NA, NA,
  NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
  NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  type_t12 = c(1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
  stageframe = cypframe_raw, historical = TRUE)

cypmatrix3r <- rlefko3(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added", "size1added"),
  supplement = cypsupp3r, yearcol = "year2",
  patchcol = "patchid", indivcol = "individ")

```

```
cypstoch <- projection3(cypmatrix3r, nreps = 5, stochastic = TRUE)
summary(cypstoch, ext_time = TRUE)
```

summary_hfv

Summary of Class "hfvdata"

Description

A function to simplify the viewing of basic information describing demographic data in historical vertical format (data frames of class hfvdata).

Usage

```
summary_hfv(
  object,
  popid = "popid",
  patchid = "patchid",
  individ = "individ",
  year2id = "year2",
  full = TRUE,
  ...
)
```

Arguments

object	An object of class hfvdata.
popid	A string denoting the name of the variable denoting population identity.
patchid	A string denoting the name of the variable denoting patch identity.
individ	A string denoting the name of the variable denoting individual identity.
year2id	A string denoting the name of the variable denoting the year in time t .
full	A logical value indicating whether to include basic data frame summary information in addition to hfvdata-specific summary information. Defaults to TRUE.
...	Other parameters.

Value

A summary of the object. The first line shows the numbers of populations, patches, individuals, and time steps. If `full = TRUE`, then this is followed by a standard data frame summary of the hfv dataset.

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)

summary_hfv(cypraw_v1)

```

supplemental

Create a Data Frame of Supplemental Data for MPM Development

Description

Function `supplemental()` provides all necessary supplemental data for matrix estimation, particularly bringing together data on proxy rates, data to overwrite existing rates, identified reproductive transitions complete, and fecundity multipliers.

Usage

```

supplemental(
  stage3,
  stage2,
  stage1 = NA,
  eststage3 = NA,
  eststage2 = NA,
  eststage1 = NA,
  givenrate = NA,
  multiplier = 1,

```



```

    type = NA,
    type_t12 = NA,
    stageframe,
    historical = TRUE
  )

```

Arguments

stage3	The name of the stage in occasion $t+1$ in the transition to be replaced. Abbreviations for groups of stages are also usable (see Notes).
stage2	The name of the stage in occasion t in the transition to be replaced. Abbreviations for groups of stages are also usable (see Notes).
stage1	The name of the stage in occasion $t-1$ in the transition to be replaced. Only needed if a historical matrix is to be produced. Abbreviations for groups of stages are also usable (see Notes).
eststage3	The name of the stage to replace stage3 in a proxy transition. Only needed if a transition will be replaced by another estimated transition.
eststage2	The name of the stage to replace stage2 in a proxy transition. Only needed if a transition will be replaced by another estimated transition.
eststage1	The name of the stage to replace stage1 in a proxy historical transition. Only needed if a transition will be replaced by another estimated transition, and the matrix to be estimated is historical. Stage NotAlive is also possible for raw hMPMs as a means of handling the prior stage for individuals entering the population in occasion t .
givenrate	A fixed rate or probability to replace for the transition described by stage3, stage2, and stage1.
multiplier	A vector of numeric multipliers for fecundity or for proxy transitions. Defaults to 1.
type	A vector denoting the kind of transition between occasions t and $t+1$ to be replaced. This should be entered as 1, S, or s for the replacement of a survival transition; 2, F, or f for the replacement of a fecundity transition; or 3, R, or r for a fecundity multiplier. If empty or not provided, then defaults to 1 for survival transition.
type_t12	An optional vector denoting the kind of transition between occasions $t-1$ and t . Only necessary if a historical MPM in deVries format is desired. This should be entered as 1, S, or s for a survival transition; or 2, F, or f for a fecundity transitions. Defaults to 1 for survival transition, with impacts only on the construction of deVries-format hMPMs.
stageframe	The stageframe being used to produce the MPMs in the study.
historical	A logical value indicating whether the MPMs intended will be historical or ahistorical. Defaults to TRUE.

Value

A data frame of class `lefkoSD`. This object can be used as input in `flefko3()`, `flefko2()`, `rlefko3()`, `rlefko2()`, and `aflefko2()`.

Variables in this object include the following:

stage3	Stage at occasion $t+1$ in the transition to be replaced.
stage2	Stage at occasion t in the transition to be replaced.
stage1	Stage at occasion $t-1$ in the transition to be replaced.
eststage3	Stage at occasion $t+1$ in the transition to replace the transition designated by stage3, stage2, and stage1.
eststage2	Stage at occasion t in the transition to replace the transition designated by stage3, stage2, and stage1.
eststage1	Stage at occasion $t-1$ 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.
multiplier	A multiplier for proxy transitions or for fecundity.
convtype	Designates whether the transition from occasion t to occasion $t+1$ is a survival transition probability (1), a fecundity rate (2), or a fecundity multiplier (3).
convtype_t12	Designates whether the transition from occasion $t-1$ to occasion t is a survival transition probability (1), a fecundity rate (2).

Notes

Negative values are not allowed in `givenrate` and `multiplier` input.

Fecundity multiplier data supplied via the `supplemental()` function acts in the same way as non-zero entries supplied via a reproductive matrix, but gets priority in all matrix creations. Thus, in cases where fecundity multipliers are provided for the same function via the reproductive matrix and function `supplemental()`, the latter is used.

Entries in `stage3`, `stage2`, and `stage1` can include abbreviations for groups of stages. Use `rep` if all reproductive stages are to be used, `nrep` if all mature but non-reproductive stages are to be used, `mat` if all mature stages are to be used, `immat` if all immature stages are to be used, `prop` if all propagule stages are to be used, `npr` if all non-propagule stages are to be used, `obs` if all observable stages are to be used, `nobs` if all unobservable stages are to be used, and leave empty or use `all` if all stages in `stageframe` are to be used. Also use `groupX` to denote all stages in group X (e.g. `group1` will use all stages in the respective `stageframe`'s group 1).

Examples

```
# Lathyrus example
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfeko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

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

# Cypripedium example
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, 0, 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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type =c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefk2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", indivcol = "individ")

cyp2mean <- lmean(cypmatrix2r)
cyp2mean

```

usher3

Two-parameter Usher function

Description

Function `usher3()` creates a vector of values produced by the two- parameter Usher function as applied with a user-specified time lag. The Usher function is given as $\phi_{t+1} = \phi_t / (1 + e^{\alpha t + \beta})$. Here, if no `separate_N` vector is provided, then $n_t = \phi_t$.

Usage

```

usher3(
  start_value,
  alpha,
  beta,
  time_steps = 100L,
  time_lag = 1L,
  pre0_subs = FALSE,
  pre0_value = 0,
  substoch = 0L,
  separate_N = NULL
)

```

Arguments

<code>start_value</code>	A positive number to start the return vector in time 0.
<code>alpha</code>	The alpha parameter in the two-parameter Usher function.
<code>beta</code>	The beta parameter in the two-parameter Usher function.
<code>time_steps</code>	The number of time steps to run the projection. Must be a positive integer.
<code>time_lag</code>	A positive integer denoting the number of time steps back for the value of phi in the two-parameter Usher function.
<code>pre0_subs</code>	A logical value indicating whether to use a number other than that given in <code>start_value</code> for values of phi lagged from times prior to time 0.
<code>pre0_value</code>	A positive number to use for phi lagged from times prior to time 0. Only used if <code>pre0_subs = TRUE</code> .
<code>substoch</code>	An integer value indicating the kind of substochasticity to use. Values include: 0, no substochasticity enforced (the default); 1, all numbers must be non-negative; and 2, all numbers should be forced to the interval [0, 1].
<code>separate_N</code>	An optional numeric vector with values of N in each time, if phi is to be treated as different from N in the two-parameter model.

Value

A numeric vector of values showing values projected under the two-parameter Usher function.

Examples

```

trial_run1 <- usher3(1, alpha = -0.5, beta = 0.005)
plot(trial_run1)

trial_run2 <- usher3(1, alpha = 0.5, beta = 0.005)
plot(trial_run2)

trial_run3 <- usher3(1, alpha = -5, beta = 0.005)
plot(trial_run3)

trial_run4 <- usher3(1, alpha = 5, beta = 0.005)
plot(trial_run4)

trial_run5 <- usher3(1, alpha = -25, beta = 0.005)
plot(trial_run5)

trial_run6 <- usher3(1, alpha = 25, beta = 0.005)
plot(trial_run6)

used_Ns <- c(10, 15, 12, 14, 14, 150, 15, 1, 5, 7, 9, 14, 13, 16, 17, 19,
            25, 26)
trial_run7 <- usher3(1, alpha = -0.5, beta = 0.005, separate_N = used_Ns)
plot(trial_run7)

```

`verticalize3`*Create Historical Vertical Data Frame from Horizontal Data Frame*

Description

Function `verticalize3()` returns a vertically formatted demographic data frame organized to create historical projection matrices, given a horizontally formatted input data frame. It also handles stage assignments if given an appropriate `stageframe`.

Usage

```
verticalize3(  
  data,  
  noyears,  
  firstyear = 1,  
  popidcol = 0,  
  patchidcol = 0,  
  individcol = 0,  
  blocksize = NA,  
  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,  
  censorRepeat = FALSE,  
  censor = FALSE,  
  coordsRepeat = FALSE,  
  spacing = NA,
```

```

NAas0 = FALSE,
NRasRep = FALSE,
NOasObs = FALSE,
prebreeding = TRUE,
age_offset = 0,
reduce = TRUE,
a2check = FALSE,
quiet = FALSE
)

```

Arguments

data	The horizontal data file. A valid data frame is required as input.
noyears	The number of years or observation occasions in the dataset. A valid integer is required as input.
firstyear	The first year or occasion of observation. Defaults to 1.
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 or subpopulation 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 occasion in the input dataset designated in data, if a set pattern of variables is used for each observation occasion in the data frame used as input. If such a pattern is not used, and all variable names are properly noted as character vectors in the other input variables, then this may be set to NA. Defaults to NA.
xcol	A variable name(s) or column number(s) corresponding to the X coordinate of each individual, or of each individual at each occasion, in Cartesian space. Can refer to the only instance, the first instance, or all instances of X variables. In the last case, the values should be entered as a vector.
ycol	A variable name(s) or column number(s) corresponding to the Y coordinate of each individual, or of each individual at each occasion, in Cartesian space. Can refer to the only instance, the first instance, or all instances of Y variables. In the last case, the values should be entered as a vector.
juvcol	A variable name(s) or column number(s) that marks individuals in immature stages within the dataset. This 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. Can refer to the first instance, or all instances of these variables. In the latter case, the values should be entered as a vector.
sizeacol	A variable name(s) or column number(s) corresponding to the size entry associated with the first year or observation occasion in the dataset. Can refer to the first instance, or all instances of these variables. In the latter case, the values should be entered as a vector. This variable should refer to the first size variable in the stageframe, unless <code>stagesize = "sizeadded"</code> .

sizebcol	A second variable name(s) or column number(s) corresponding to the size entry associated with the first year or observation occasion in the dataset. Can refer to the first instance, or all instances of these variables. In the latter case, the values should be entered as a vector. This variable should refer to the second size variable in the stageframe, unless <code>stagesize = "sizeadded"</code> .
sizeccol	A third variable name(s) or column number(s) corresponding to the size entry associated with the first year or observation occasion in the dataset. Can refer to the first instance, or all instances of these variables. In the latter case, the values should be entered as a vector. This variable should refer to the third size variable in the stageframe, unless <code>stagesize = "sizeadded"</code> .
repstracol	A variable name(s) or column number(s) 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 analyze the probability of reproduction. Can refer to the first instance, or all instances of these variables. In the latter case, the values should be entered as a vector.
repstrbcol	A second variable name(s) or column number(s) 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 analyze the probability of reproduction. Can refer to the first instance, or all instances of these variables. In the latter case, the values should be entered as a vector.
fecacol	A variable name(s) or column number(s) denoting fecundity associated with the first year or observation occasion in the input dataset. This may represent egg counts, fruit counts, seed production, etc. Can refer to the first instance, or all instances of these variables. In the latter case, the values should be entered as a vector.
fecbcol	A second variable name(s) or column number(s) denoting fecundity associated with the first year or observation occasion in the input dataset. This may represent egg counts, fruit counts, seed production, etc. Can refer to the first instance, or all instances of these variables. In the latter case, the values should be entered as a vector.
indcovacol	A variable name(s) or column number(s) corresponding to an individual covariate to be used in analysis. Can refer to the only instance, the first instance, or all instances of these variables. In the last case, the values should be entered as a vector.
indcovbcol	A variable name(s) or column number(s) corresponding to an individual covariate to be used in analysis. Can refer to the only instance, the first instance, or all instances of these variables. In the last case, the values should be entered as a vector.
indcovccol	A second variable name(s) or column number(s) corresponding to an individual covariate to be used in analysis. Can refer to the only instance, the first instance, or all instances of these variables. In the last case, the values should be entered as a vector.
aliveacol	Variable name(s) or column number(s) providing information on whether an individual is alive at a given occasion. If used, living status must be designated

	as binomial (living = 1, dead = 0). Can refer to the first instance of a living status variable in the dataset, or a full vector of all living status variables in temporal order.
deadacol	Variable name(s) or column number(s) providing information on whether an individual is alive at a given occasion. If used, dead status must be designated as binomial (dead = 1, living = 0). Can refer to the first instance of a dead status variable in the dataset, or a full vector of all dead status variables in temporal order.
obsacol	A variable name(s) or column number(s) providing information on whether an individual is in an observable stage at a given occasion. If used, observation status must be designated as binomial (observed = 1, not observed = 0). Can refer to the first instance of an observation status variable in the dataset, or a full vector of all observation status variables in temporal order.
nonobsacol	A variable name(s) or column number(s) providing information on whether an individual is in an unobservable stage at a given occasion. If used, observation status must be designated as binomial (not observed = 1, observed = 0). Can refer to the first instance of a non-observation status variable in the dataset, or a full vector of all non-observation status variables in temporal order.
sensorcol	A variable name(s) or column number(s) 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. Can refer to the first instance of a censor status variable in the dataset, or a full vector of all censor status variables in temporal order. Can also refer to a single censor status variable used for the entire individual, if <code>singlecensor = TRUE</code> .
repstrrel	This is a scalar multiplier on variable <code>repstrbcol</code> to make it equivalent to <code>repstracol</code> . This can be useful if two reproductive status variables have related but unequal units, for example if <code>repstracol</code> refers to one-flowered stems while <code>repstrbcol</code> refers to two-flowered stems. Defaults to 1.
fecrel	This is a scalar multiplier on variable <code>fecbcol</code> to make it equivalent to <code>fecacol</code> . This can be useful if two fecundity variables have related but unequal units. Defaults to 1.
stagecol	Optional variable name(s) or column number(s) corresponding to life history stage at a given occasion. Can refer to the first instance of a stage identity variable in the dataset, or a full vector of all stage identity variables in temporal order.
stageassign	The stageframe object identifying the life history model being operationalized. Note that if <code>stagecol</code> 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 <code>sizea</code> , <code>sizeb</code> , <code>sizec</code> , <code>sizeab</code> , <code>sizebc</code> , <code>sizeac</code> , <code>sizeabc</code> , or <code>sizeadded</code> , depending on which size variable within the input dataset is chosen. Note that the variable(s) chosen should be presented in the order of the primary, secondary, and tertiary variables in the stageframe input with <code>stageassign</code> . For example, choosing <code>sizeb</code> assumes that this size is the primary variable in the stageframe.

sensorkeep	The value of the censor variable identifying data to be included in analysis. Defaults to 0, but may take any value including NA. Note that if NA is the value to keep, then this function will alter all NAs to 0 values, and all other values to 1, treating 0 as the new value to keep.
sensorRepeat	A logical value indicating whether the censor variable is a single column, or whether it repeats across occasion blocks. Defaults to FALSE.
censor	A logical variable determining whether the output data should be censored using the variable defined in <code>sensorcol</code> . Defaults to FALSE.
coordsRepeat	A logical value indicating whether X and Y coordinates correspond to single X and Y columns. If TRUE, then each observation occasion has its own X and Y variables. 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 <code>xcol</code> and <code>ycol</code> . 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 <code>modelsearch()</code> , 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 MPM is desired without separation of reproductive and non-reproductive but mature stages of the same size. Only used if <code>stageassign</code> is set to a stageframe. Defaults to FALSE.
NOasObs	If TRUE, then will treat individuals that are interpreted as not observed in the dataset as though they were observed during stage assignment. This can be useful when a MPM is desired without separation of observable and unobservable stages. Only used if <code>stageassign</code> is set to a stageframe. Defaults to FALSE.
prebreeding	A logical term indicating whether the life history model is pre-breeding. If so, then 1 is added to all ages. Defaults to TRUE.
age_offset	A number to add automatically to all values of age at time t . Defaults to 0.
reduce	A logical variable determining whether unused variables and some invariant state variables should be removed from the output dataset. Defaults to TRUE.
a2check	A logical variable indicating whether to retain all data with living status at occasion t equal to 0. Defaults to FALSE, and should be kept FALSE except to inspect potential errors in the dataset.
quiet	A logical variable indicating whether to silence warnings. Defaults to FALSE.

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 occasions of size and reproductive data per individual per row. This data frame is in standard format for all functions used in `lefk3`, 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 occasion t .
firstseen	Occasion of first observation.
lastseen	Occasion of last observation.
obsage	Observed age in occasion t , assuming first observation corresponds to age = 0.
obslifespan	Observed lifespan, given as $\text{lastseen} - \text{firstseen} + 1$.
xpos1, xpos2, xpos3	X position in Cartesian space in occasions $t-1$, t , and $t+1$, respectively, if provided.
ypos1, ypos2, ypos3	Y position in Cartesian space in occasions $t-1$, t , and $t+1$, respectively, if provided.
sizea1, sizea2, sizea3	Main size measurement in occasions $t-1$, t , and $t+1$, respectively.
sizeb1, sizeb2, sizeb3	Secondary size measurement in occasions $t-1$, t , and $t+1$, respectively.
sizec1, sizec2, sizec3	Tertiary measurement in occasions $t-1$, t , and $t+1$, respectively.
size1added, size2added, size3added	Sum of primary, secondary, and tertiary size measurements in occasions $t-1$, t , and $t+1$, respectively.
repstra1, repstra2, repstra3	Main numbers of reproductive structures in occasions $t-1$, t , and $t+1$, respectively.
repstrb1, repstrb2, repstrb3	Secondary numbers of reproductive structures in occasions $t-1$, t , and $t+1$, respectively.
repstr1added, repstr2added, repstr3added	Sum of primary and secondary reproductive structures in occasions $t-1$, t , and $t+1$, respectively.
feca1, feca2, feca3	Main numbers of offspring in occasions $t-1$, t , and $t+1$, respectively.
fecb1, fecb2, fecb3	Secondary numbers of offspring in occasions $t-1$, t , and $t+1$, respectively.
fec1added, fec2added, fec3added	Sum of primary and secondary fecundity in occasions $t-1$, t , and $t+1$, respectively.
sensor1, sensor2, sensor3	Censor state values in occasions $t-1$, t , and $t+1$, respectively.
juvgiven1, juvgiven2, juvgiven3	Binomial variable indicating whether individual is juvenile in occasions $t-1$, t , and $t+1$. Only given if <code>juvcol</code> is provided.

obsstatus1,obsstatus2,obsstatus3	Binomial observation state in occasions $t-1$, t , and $t+1$, respectively.
repstatus1,repstatus2,repstatus3	Binomial reproductive state in occasions $t-1$, t , and $t+1$, respectively.
fecstatus1,fecstatus2,fecstatus3	Binomial offspring production state in occasions $t-1$, t , and $t+1$, respectively.
matstatus1,matstatus2,matstatus3	Binomial maturity state in occasions $t-1$, t , and $t+1$, respectively.
alive1,alive2,alive3	Binomial state as alive in occasions $t-1$, t , and $t+1$, respectively.
density	Radial density of individuals per unit designated in spacing. Only given if spacing is not NA.

Notes

In some datasets on species with unobservable stages, observation status (`obsstatus`) might not be inferred properly if a single size variable is used that does not yield sizes greater than 0 in all cases in which individuals were observed. Such situations may arise, for example, in plants when leaf number is the dominant size variable used, but individuals occasionally occur with inflorescences but no leaves. In this instances, it helps to mark related variables as `sizeb` and `sizec`, because observation status will be interpreted in relation to all 3 size variables. Further analysis can then utilize only a single size variable, of the user's choosing. Similar issues can arise in reproductive status (`repstatus`).

Juvenile designation should only be used when juveniles fall outside of the size classification scheme used in determining stages. If juveniles are to be size classified along the size spectrum that adults also fall on, then it is best to treat juveniles as mature but not reproductive.

Warnings that some individuals occur in state combinations that do not match any stages in the stageframe used to assign stages are common when first working with a dataset. Typically, these situations can be identified as `NoMatch` entries in `stage3`, although such entries may crop up in `stage1` and `stage2`, as well. In rare cases, these warnings will arise with no concurrent `NoMatch` entries, which indicates that the input dataset contained conflicting state data at once suggesting that the individual is in some stage but is also dead. The latter is removed if the conflict occurs in occasion t or $t-1$, as only living entries are allowed in time t and time $t-1$ may involve living entries as well as non-living entries immediately prior to birth.

Care should be taken to avoid variables with negative values indicating size, fecundity, or reproductive or observation status. Negative values can be interpreted in different ways, typically reflecting estimation through other algorithms rather than actual measured data. Variables holding negative values can conflict with data management algorithms in ways that are difficult to predict.

Unusual errors (e.g. "Error in .pfj...") may occur in cases where the variables are improperly passed, where seemingly numeric variables include text, or where the `blocksize` is improperly set.

Density estimation is performed as a count of individuals alive and within the radius specified in spacing of the respective individual at some point in time.

If a censor variable is included for each monitoring occasion, and the `blocksize` option is set, then the user must set `sensorRepeat = TRUE` in order to censor the correct transitions. Failing this step will likely lead to the loss of a large portion of the data as all data for entire individuals will be excluded.

Examples

```

# Lathyrus example using blocksize - when repeated patterns exist in variable
# order
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfeko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", individcol = "individ")

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

# Lathyrus example without blocksize - when no repeated patterns exist in
# variable order and all variables names are specified
data(lathyrus)

sizevector <- c(0, 100, 13, 127, 3730, 3800, 0)
stagevector <- c("Sd", "Sd1", "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",
  juvcol = c("Seedling1988", "Seedling1989", "Seedling1990", "Seedling1991"),
  sizeacol = c("Volume88", "Volume89", "Volume90", "Volume91"),
  repstracol = c("FCODE88", "FCODE89", "FCODE90", "FCODE91"),
  fecacol = c("Intactseed88", "Intactseed89", "Intactseed90", "Intactseed91"),
  deadacol = c("Dead1988", "Dead1989", "Dead1990", "Dead1991"),
  nonobsacol = c("Dormant1988", "Dormant1989", "Dormant1990", "Dormant1991"),
  censorcol = c("Missing1988", "Missing1989", "Missing1990", "Missing1991"),
  stageassign = lathframe, stagesize = "sizea",
  censorkeep = NA, censor = TRUE)

lathsupp3 <- supplemental(stage3 = c("Sd", "Sd", "Sd1", "Sd1", "Sd", "Sd1", "mat"),
  stage2 = c("Sd", "Sd", "Sd", "Sd", "rep", "rep", "Sd1"),
  stage1 = c("Sd", "rep", "Sd", "rep", "npr", "npr", "Sd"),
  eststage3 = c(NA, NA, NA, NA, NA, NA, "mat"),
  eststage2 = c(NA, NA, NA, NA, NA, NA, "Sd1"),
  eststage1 = c(NA, NA, NA, NA, NA, NA, "NotAlive"),
  givenrate = c(0.345, 0.345, 0.054, 0.054, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, 0.345, 0.054, NA),
  type = c(1, 1, 1, 1, 3, 3, 1), type_t12 = c(1, 2, 1, 2, 1, 1, 1),
  stageframe = lathframe, historical = TRUE)

ehrlen3 <- rlfeko3(data = lathvert, stageframe = lathframe, year = "all",
  stages = c("stage3", "stage2", "stage1"), supplement = lathsupp3,
  yearcol = "year2", indivcol = "individ")

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

# Cypripedium example using blocksize
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, 0, 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)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlfko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", individcol = "individ")

cyp2mean <- lmean(cypmatrix2r)
cyp2mean

# Cyripedium example using partial repeat patterns with blocksize and part
# explicit variable name cast
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 = c("Inf.04", "Inf.05", "Inf.06", "Inf.07", "Inf.08", "Inf.09"),
  repstrbcol = c("Inf2.04", "Inf2.05", "Inf2.06", "Inf2.07", "Inf2.08", "Inf2.09"),
  fecacol = "Pod.04", stageassign = cypframe_raw, stagesize = "sizeadded",
  NAas0 = TRUE, NRasRep = TRUE)

cypsupp2r <- supplemental(stage3 = c("SD", "P1", "P2", "P3", "SL", "D",
  "XSm", "Sm", "SD", "P1"),
  stage2 = c("SD", "SD", "P1", "P2", "P3", "SL", "SL", "SL", "rep",
  "rep"),
  eststage3 = c(NA, NA, NA, NA, NA, "D", "XSm", "Sm", NA, NA),
  eststage2 = c(NA, NA, NA, NA, NA, "XSm", "XSm", "XSm", NA, NA),
  givenrate = c(0.10, 0.20, 0.20, 0.20, 0.25, NA, NA, NA, NA, NA),
  multiplier = c(NA, NA, NA, NA, NA, NA, NA, NA, 0.5, 0.5),
  type = c(1, 1, 1, 1, 1, 1, 1, 1, 3, 3),
  stageframe = cypframe_raw, historical = FALSE)

cypmatrix2r <- rlefko2(data = cypraw_v1, stageframe = cypframe_raw,
  year = "all", patch = "all", stages = c("stage3", "stage2", "stage1"),
  size = c("size3added", "size2added"), supplement = cypsupp2r,
  yearcol = "year2", patchcol = "patchid", individcol = "individ")

cyp2mean <- lmean(cypmatrix2r)
cyp2mean

```

vrm_import

Import Vital Rate Model Factor Values for Function-based MPM Development

Description

Function `vrm_import()` builds a skeleton list holding data frames and vectors that can be used to import coefficient values for the factors of the vital rate models used to build function-based MPMs or run function-based projections.

Usage

```

vrm_import(
  years = NULL,
  patches = c(1),
  groups = c(0),
  interactions = FALSE,
  zi = FALSE,
  cat.indcova = NULL,
  cat.indcovb = NULL,

```



```

    cat.indcovc = NULL,
    dist.sizea = "gaussian",
    dist.sizeb = "constant",
    dist.sizec = "constant",
    dist.fec = "gaussian",
    trunc.sizea = FALSE,
    trunc.sizeb = FALSE,
    trunc.sizec = FALSE,
    trunc.fec = FALSE,
    use.juv = FALSE
  )

```

Arguments

years	A numeric vector of the years or times at time t to be modeled.
patches	A string or numeric vector of the patch names to be modeled.
groups	An integer vector of stage groups to be modeled. Defaults to a vector with a single element with value 0.
interactions	A logical value indicating whether to include two-way interactions between main effects (TRUE), or only main effects (FALSE). Defaults to FALSE.
zi	A logical value indicating whether to include coefficients for the binomial components of zero-inflation models. Defaults to FALSE.
cat.indcova	If individual covariate a is categorical, then this term should equal a string vector of the names of the categories. Defaults to NULL, in which case individual covariate a is either not used or is numeric.
cat.indcovb	If individual covariate b is categorical, then this term should equal a string vector of the names of the categories. Defaults to NULL, in which case individual covariate b is either not used or is numeric.
cat.indcovc	If individual covariate c is categorical, then this term should equal a string vector of the names of the categories. Defaults to NULL, in which case individual covariate c is either not used or is numeric.
dist.sizea	A string value giving the distribution of the variable coding primary size. Can equal "none", "gamma", "gaussian", "poisson", "negbin", or "constant". Defaults to "gaussian".
dist.sizeb	A string value giving the distribution of the variable coding secondary size. Can equal "none", "gamma", "gaussian", "poisson", "negbin", or "constant". Defaults to "constant".
dist.sizec	A string value giving the distribution of the variable coding tertiary size. Can equal "none", "gamma", "gaussian", "poisson", "negbin", or "constant". Defaults to "constant".
dist.fec	A string value giving the distribution of the variable coding fecundity. Can equal "none", "gamma", "gaussian", "poisson", or "negbin". Defaults to "gaussian".
trunc.sizea	A logical value indicating whether the distribution of the primary size variable should be zero-truncated. Defaults to FALSE. Currently only works with the Poisson and negative binomial distributions.

<code>trunc.sizeb</code>	A logical value indicating whether the distribution of the secondary size variable should be zero-truncated. Defaults to FALSE. Currently only works with the Poisson and negative binomial distributions.
<code>trunc.sizec</code>	A logical value indicating whether the distribution of the tertiary size variable should be zero-truncated. Defaults to FALSE. Currently only works with the Poisson and negative binomial distributions.
<code>trunc.fec</code>	A logical value indicating whether the distribution of the fecundity variable should be zero-truncated. Defaults to FALSE. Currently only works with the Poisson and negative binomial distributions.
<code>use.juv</code>	A logical value indicating whether to utilize juvenile vital rates. If FALSE, then all juvenile vital rates will be set to constant distributions. Defaults to FALSE.

Value

A list of class `vrm_input`, with up to 13 elements including:

<code>vrm_frame</code>	A data frame holding the main slope coefficients for the linear vital rate models.
<code>year_frame</code>	A data frame holding the main slope coefficients for the year at time t terms in the linear vital rate models.
<code>patch_frame</code>	A data frame holding the main slope coefficients for the patch terms in the linear vital rate models.
<code>group2_frame</code>	A data frame holding the main slope coefficients for the stage group terms in time t in the linear vital rate models.
<code>group1_frame</code>	A data frame holding the main slope coefficients for the stage group terms in time $t-1$ in the linear vital rate models.
<code>dist_frame</code>	A data frame giving the distributions of all variables, including primary, secondary, and tertiary size, and fecundity. Some variables begin as constant.
<code>indcova2_frame</code>	A data frame holding the main slope coefficients for the categorical individual covariate a terms in time t in the linear vital rate models.
<code>indcova1_frame</code>	A data frame holding the main slope coefficients for the categorical individual covariate a terms in time $t-1$ in the linear vital rate models.
<code>indcovb2_frame</code>	A data frame holding the main slope coefficients for the categorical individual covariate b terms in time t in the linear vital rate models.
<code>indcovb1_frame</code>	A data frame holding the main slope coefficients for the categorical individual covariate b terms in time $t-1$ in the linear vital rate models.
<code>indcovc2_frame</code>	A data frame holding the main slope coefficients for the categorical individual covariate c terms in time t in the linear vital rate models.
<code>indcovc1_frame</code>	A data frame holding the main slope coefficients for the categorical individual covariate c terms in time $t-1$ in the linear vital rate models.
<code>st_frame</code>	A data frame holding values of sigma or theta for use in Gaussian or negative binomial response terms, respectively.

The first element, called `vrm_frame`, is a data frame with the following 18 variables:

<code>main_effect_1</code>	The main effect for which coefficients are to be entered.
----------------------------	---

main_1_defined	A more natural explanation of main_effect_1.
main_effect_2	If given, then indicates another effect in a two-way interaction with main_effect_1.
main_2_defined	A more natural explanation of main_effect_2.
surv	A vector of coefficients for the factors in the model of adult survival.
obs	A vector of coefficients for the factors in the model of adult observation status.
sizea	A vector of coefficients for the factors in the model of adult primary size.
sizeb	A vector of coefficients for the factors in the model of adult secondary size.
sizec	A vector of coefficients for the factors in the model of adult tertiary size.
repst	A vector of coefficients for the factors in the model of adult reproductive status.
fec	A vector of coefficients for the factors in the model of adult fecundity.
jsurv	A vector of coefficients for the factors in the model of juvenile survival.
jobs	A vector of coefficients for the factors in the model of juvenile observation status.
jsize	A vector of coefficients for the factors in the model of juvenile primary size.
jsizeb	A vector of coefficients for the factors in the model of juvenile secondary size.
jsizec	A vector of coefficients for the factors in the model of juvenile tertiary size.
jrepst	A vector of coefficients for the factors in the model of juvenile reproductive status, for individuals maturing in the current time step.
jmat	A vector of coefficients for the factors in the model of maturity status, for individuals capable of maturing at the current time step.
sizea_zi	A vector of coefficients for the factors in the binomial component of the zero-inflated model of adult primary size, if zero-inflated models are being used.
sizeb_zi	A vector of coefficients for the factors in the binomial component of the zero-inflated model of adult secondary size, if zero-inflated models are being used.
sizec_zi	A vector of coefficients for the factors in the binomial component of the zero-inflated model of adult tertiary size, if zero-inflated models are being used.
fec_zi	A vector of coefficients for the factors in the binomial component of the zero-inflated model of fecundity, if zero-inflated models are being used.
jsizea_zi	A vector of coefficients for the factors in the binomial component of the zero-inflated model of juvenile primary size, if zero-inflated models are being used.
jsizeb_zi	A vector of coefficients for the factors in the binomial component of the zero-inflated model of juvenile secondary size, if zero-inflated models are being used.
jsizec_zi	A vector of coefficients for the factors in the binomial component of the zero-inflated model of juvenile tertiary size, if zero-inflated models are being used.

Notes

All coefficients across all data frames are initially set to \emptyset . After using this function to create the skeleton list, all relevant coefficient values should be set to non-zero values equal to the respective slope from the appropriate linear model, and any vital rate model to be used should have its distribution set to "binom", "gaussian", "gamma", "poisson", or "negbin". Unused vital rates should

be set to "constant", and the first element of the corresponding column in `vrm_frame` (corresponding to the y-intercept) should be set to the constant value to utilize (generally 1). If no values are manually edited, then function-based MPM generator functions will not be able to generate valid MPMs.

Users should never change the labels or the order of terms in the data frames and vectors produced via this function, nor should they ever change the names of core list elements in the `vrm_input` object. Doing so will result either in fatal errors or erroneous matrix calculations.

Using the `vrm_input` approach to building function-based MPMs requires careful attention to the stageframe. Although no `hfv` data frame needs to be entered, stages for which vital rates are to be estimated via linear models parameterized with coefficients provided via function `vrm_import()` should be marked as occurring within the dataset, while stages for which the provided coefficients should not be used should be marked as not occurring within the dataset.

Coefficients added to zero-inflation models can only be added to primary size, secondary size, tertiary size, fecundity, and the juvenile versions of primary, secondary, and tertiary size. Care must be taken to include zero-inflated coefficients only for variables without size-truncated distributions. Adding such terms will result in fatal errors during matrix creation.

Examples

```
data(lathyrus)

sizevector <- c(0, 100, 0, 1, 7100)
stagevector <- c("Sd", "Sd1", "Dorm", "ipm", "ipm")
repvector <- c(0, 0, 0, 1, 1)
obsvector <- c(0, 1, 0, 1, 1)
matvector <- c(0, 0, 1, 1, 1)
immvector <- c(1, 1, 0, 0, 0)
propvector <- c(1, 0, 0, 0, 0)
indataset <- c(0, 1, 1, 1, 1)
binvec <- c(0, 100, 0.5, 1, 1)
comments <- c("Dormant seed", "Seedling", "Dormant", "ipm adult stage",
             "ipm adult stage")
lathframeipm <- sf_create(sizes = sizevector, stagenames = stagevector,
                        repstatus = repvector, obsstatus = obsvector, propstatus = propvector,
                        immstatus = immvector, matstatus = matvector, comments = comments,
                        indataset = indataset, binhalfwidth = binvec, ipmbins = 100, roundsize = 3)

lathsupp2 <- supplemental(stage3 = c("Sd", "Sd1", "Sd", "Sd1"),
                        stage2 = c("Sd", "Sd", "rep", "rep"),
                        givenrate = c(0.345, 0.054, NA, NA),
                        multiplier = c(NA, NA, 0.345, 0.054),
                        type = c(1, 1, 3, 3), stageframe = lathframeipm, historical = FALSE)

lath_vrm <- vrm_import(years = c(1988:1990), zi = TRUE, dist.fec = "negbin",
                      use.juv = TRUE)

lath_vrm$vrm_frame$surv[1] <- 2.32571
lath_vrm$vrm_frame$surv[2] <- 0.00109
lath_vrm$vrm_frame$obs[1] <- 2.230
lath_vrm$vrm_frame$sizea[1] <- 164.0695
```

```

lath_vrm$vrms_frame$sizea[2] <- 0.6211
lath_vrm$vrms_frame$fec[1] <- 1.517
lath_vrm$vrms_frame$fec_zi[1] <- 6.252765
lath_vrm$vrms_frame$fec_zi[2] <- -0.007313
lath_vrm$vrms_frame$jssurv[1] <- 1.03
lath_vrm$vrms_frame$jjobs[1] <- 10.390
lath_vrm$vrms_frame$jsizea[1] <- 3.0559
lath_vrm$vrms_frame$jsizea[2] <- 0.8482

lath_vrm$year_frame$fec[c(1:3)] <- c(-0.41749627, 0.51421684, -0.07964038)
lath_vrm$year_frame$fec_zi[c(1:3)] <- c(3.741475e-07, -7.804715e-08,
-2.533755e-07)
lath_vrm$year_frame$sizea[c(1:3)] <- c(96.3244, -240.8036, 144.4792)
lath_vrm$year_frame$jjobs[c(1:3)] <- c(-0.7459843, 0.6118826, -0.9468618)
lath_vrm$year_frame$jsizea[c(1:3)] <- c(0.5937962, 1.4551236, -2.0489198)

lath_vrm$dist_frame$dist[2] <- "binom"
lath_vrm$dist_frame$dist[9] <- "binom"

lath_vrm$st_frame[3] <- 503.6167
lath_vrm$st_frame[7] <- 0.2342114
lath_vrm$st_frame[10] <- 5.831

lath_vrm$vrms_frame$sizeb[1] <- 1
lath_vrm$vrms_frame$sizec[1] <- 1
lath_vrm$vrms_frame$repst[1] <- 1
lath_vrm$vrms_frame$jsizeb[1] <- 1
lath_vrm$vrms_frame$jsizec[1] <- 1
lath_vrm$vrms_frame$jrepst[1] <- 1
lath_vrm$vrms_frame$jmatst[1] <- 1

lathmat2_importipm <- flefko2(stageframe = lathframeipm,
  modelsuite = lath_vrm, supplement = lathsupp2, reduce = FALSE)

summary(lathmat2_importipm)

```

Index

- * **datasets**
 - cypdata, 39
 - cypvert, 42
 - lathyrus, 138
- actualstage3, 3
- add_lM, 6, 32, 44, 231
- aflefko2, 11, 24, 38, 76, 86, 92, 100, 164, 186, 191, 196, 208, 209, 240, 249
- arlefko2, 17, 19, 20, 75, 76, 85, 86, 92, 186, 191, 196
- beverton3, 25
- cond_diff, 27
- cond_hmpm, 29
- create_lM, 7, 31, 44, 231
- create_pm, 15, 38, 72, 82, 90, 97
- cypdata, 39
- cypvert, 42
- delete_lM, 7, 32, 44, 231
- density_input, 48, 53, 99, 169, 170, 229
- density_vr, 52, 99
- diff_lM, 27, 55
- dredge, 160, 161
- elasticity3, 58, 61, 64, 67
- elasticity3.lefkoMat, 58, 60, 64, 67
- elasticity3.list, 58, 61, 63, 67
- elasticity3.matrix, 58, 61, 64, 66
- f_projection3, 53, 93, 170
- flefko2, 18, 24, 38, 68, 86, 92, 100, 164, 186, 191, 196, 208–210, 240, 249
- flefko3, 18, 24, 38, 76, 78, 92, 100, 164, 186, 191, 196, 208–210, 240, 249
- fleslie, 19, 24, 76, 86, 89, 100, 186, 191, 196, 240
- glm, 160
- glm.nb, 160
- glmer, 160
- glmmTMB, 160
- hfv_qc, 102
- hist_null, 115
- historicalize3, 13, 21, 24, 71, 81, 96, 103, 107, 154, 183, 185, 188, 191
- image, 116, 119, 121, 123, 126, 128
- image3, 116
- image3.lefkoElas, 119
- image3.lefkoMat, 117, 121
- image3.lefkoSens, 123
- image3.list, 126
- image3.matrix, 117, 128
- lambda3, 130, 133, 136
- lambda3.lefkoMat, 131, 133, 136
- lambda3.matrix, 131, 133, 135
- lathyrus, 138
- lefko3, 142
- lm, 160
- lmean, 143
- lmer, 160
- logistic3, 145
- ltre3, 147, 151
- ltre3.lefkoMat, 148, 149
- modelsearch, 102, 111, 153, 215, 242, 258
- overwrite, 13, 22, 70, 80, 97, 163, 184, 190
- plot.lefkoProj, 165
- projection3, 50, 93, 99, 100, 166, 167, 228, 229
- r.squaredGLMM, 161
- repvalue3, 172, 176, 179
- repvalue3.lefkoMat, 173, 175, 178, 179
- repvalue3.matrix, 173, 176, 178

ricker3, 181
rlefko2, 17, 19, 24, 75, 76, 85, 86, 92, 164,
182, 191, 196, 208–210, 240, 249
rlefko3, 17, 19, 24, 75, 76, 85, 86, 92, 164,
186, 187, 196, 208–210, 240, 249
rleslie, 17, 19, 24, 75, 76, 85, 86, 92, 186,
191, 193, 240

sensitivity3, 197, 200, 203, 206
sensitivity3.lefkoMat, 197, 199, 203, 206
sensitivity3.list, 197, 200, 202, 206
sensitivity3.matrix, 197, 200, 203, 205
sf_create, 13, 70, 80, 96, 176, 208, 223
sf_distrib, 213
slambda3, 131, 133, 136, 217
stablestage3, 220, 224, 226
stablestage3.lefkoMat, 220, 222, 226
stablestage3.matrix, 220, 224, 225
start_input, 50, 170, 228
subset_lM, 7, 32, 44, 170, 230
summary.lefkoCondMat, 234
summary.lefkoElas, 61, 237
summary.lefkoLTRE, 148, 151, 239
summary.lefkoMat, 32, 240
summary.lefkoMod, 242
summary.lefkoProj, 243
summary_hfv, 247
supplemental, 13, 22, 70, 80, 96, 184, 189,
212, 248

usher3, 252

verticalize3, 13, 21, 24, 71, 81, 96, 103,
154, 183, 185, 188, 191, 194, 254
vglm, 160, 161
vrm_import, 212, 264

zeroinfl, 160, 161