Package ‘popbio’

September 26, 2019

Version 2.6
Author Chris Stubben, Brook Milligan, Patrick Nantel
Maintainer Chris Stubben <chris.stubben@hci.utah.edu>
Title Construction and Analysis of Matrix Population Models
License GPL-3
Encoding UTF-8
LazyData true
Suggests quadprog
Description Construct and analyze projection matrix models from a demography study of marked individuals classified by age or stage. The package covers methods described in Matrix Population Models by Caswell (2001) and Quantitative Conservation Biology by Morris and Doak (2002).

RoxygenNote 6.1.1
NeedsCompilation no
Repository CRAN
Date/Publication 2019-09-26 04:20:02 UTC

R topics documented:

01.Introduction .......................................................... 3
02.Caswell ............................................................. 3
03.Morris ............................................................... 5
aq.census .............................................................. 6
aq.matrix ............................................................ 7
aq.trans ............................................................... 8
betaval ................................................................. 9
boot.transitions ..................................................... 11
calathea ............................................................. 13
countCDFxt ........................................................ 14
damping.ratio ....................................................... 15
eigen.analysis ....................................................... 16
elasticity ............................................................ 18
Introduction to the popbio Package

Description

Popbio is a package for the construction and analysis of matrix population models. First, the package consists of the R translation of Matlab code found in Caswell (2001) or Morris and Doak (2002). A list of converted functions within each book can be accessed using help(Caswell) and help(Morris) within R, or by following the links to 02.Caswell and 03.Morris from the help content pages.

Second, the popbio package includes functions to estimate vital rates and construct projection matrices from raw census data typically collected in plant demography studies. In these studies, vital rates can often be estimated directly from annual censuses of tagged individuals using transition frequency tables. To estimate vital rates in animal demography using capture-recapture methods, try the Rcapture or mra package instead.

Finally, the package includes plotting methods and sample datasets consisting of either published projection matrices or annual census data from demography studies. Three sample demonstrations illustrate some of the package capabilities (Caswell, fillmore and stage.classify). A description of the package in the Journal of Statistical Software is available at http://www.jstatsoft.org/v22/i11.

Author(s)

Chris Stubben

References

To cite the popbio package in publications, type citation('popbio'). For details on matrix population models, see


Description

Chapter 2. Age-classified matrix models

pop.projection section 2.2. Projection of population growth rates.

Chapter 4. Stage-classified matrix models
\textbf{lambda} section 4.4. Returns the dominant eigenvalue
\textbf{stable.stage} section 4.5. Returns the stable stage distribution (right eigenvector)
\textbf{reproductive.value} section 4.6. Returns the reproductive value (left eigenvector)
\textbf{damping.ratio} section 4.7. Returns the damping ratio
\textbf{eigen.analysis} section 4.8. Computes eigenvalues and vectors, including the dominant eigenvalue, stable stage distribution, reproductive value, damping ratio, sensitivities, and elasticities. Since version 2.0, these are now included as separate functions as well.

\section*{Chapter 5. Events in the Life Cycle}
\textbf{fundamental.matrix} section 5.3.1. Calculate age-specific survival from a stage classified matrix using the fundamental matrix \( N \)
\textbf{net.reproductive.rate} section 5.3.4. Calculate the net reproductive rate of a stage classified matrix using the dominant eigenvalue of the matrix \( R \).
\textbf{generation.time} section 5.3.5. Calculate the generation time of a stage-classified matrix

Age-specific survivorship and fertility curves in Fig 5.1 and 5.2 are now included in \texttt{demo(Caswell)}.

\section*{Chapter 6. Parameter estimation}
\textbf{projection.matrix} section 6.1.1. Estimate vital rates and construct a projection matrix using transition frequency tables
\textbf{QPmat} section 6.2.2. Construct a projection matrix from a time series of individuals per stage using Wood’s quadratic programming method. Requires \texttt{quadprog} library.

\section*{Chapter 9. Sensitivity analysis}
\textbf{sensitivity} section 9.1. Calculate sensitivities
\textbf{elasticity} section 9.2. Calculate elasticities
\textbf{secder} section 9.7. Second derivatives of eigenvalues

\section*{Chapter 10. Life Table Response Experiments}
\textbf{LTRE} section 10.1 and 10.2. Fixed designs in LTREs. See \texttt{demo(Caswell)} for variance decomposition in random design (Fig 10.10).

\section*{Chapter 12. Statistical inference}
\textbf{boot.transitions} section 12.1.4. Resample observed census transitions in a stage-fate data frame
\textbf{resample} section 12.1.5.2. Resample transitions in a projection matrix from a multinomial distribution (and fertilities from a log normal)

\section*{Chapter 14. Environmental stochasticity}
\textbf{stoch.growth.rate} section 14.3. Calculate the log stochastic growth rate by simulation and Tuljapukar’s approximation
\textbf{stoch.sens} section 14.4.1. Sensitivity and elasticity of stochastic growth rate from numerical simulations
Chapter 15. Demographic stochasticity

The example uses the whale dataset to create a plot like figure 15.3.

Author(s)

Chris Stubben

Description

Chapter 3

grizzly  Table 3.1. Grizzly bear population counts. The example includes code to calculate mean, variance and confidence intervals using regression and other procedures
extCDF  Box 3.3. Count-based extinction time cumulative distribution function
countCDFxt  Box 3.4. Count-based extinction probabilities with bootstrap confidence intervals

Chapter 7

stoch.projection  Box 7.3. Project stochastic growth from a sequence of matrices
stoch.growth.rate  Box 7.4. Calculate the log stochastic growth rate by Tuljapukar’s approximation and by simulation
stoch.quasi.ext  Box 7.5. Estimate quasi-extinction threshold

Chapter 8

Kendall  Box 8.2. Kendall’s method to correct for sampling variation
betaval  Box 8.3. Generate beta-distributed random numbers
lnorms  Box 8.4. Generate random lognormal values
stretchbetaval  Box 8.5. Generate stretched beta-distributed random numbers
vitalsim  Box 8.10. Calculate stochastic growth rate and extinction time CDF using vital rates
multiresultm  Box 8.11. Incorporate demographic stochasticity into population projections

Chapter 9

vitalsens  Box 9.1. Vital rate sensitivity and elasticity
aq.census

Annual census data for Aquilegia chrysantha

Description

Demography census data from *Aquilegia chrysantha* in Fillmore Canyon, Organ Mountains, New Mexico, 1996-2003.

Usage

aq.census

Format

A data frame with 2853 rows on the following 8 variables:

- **plot** Plot number
- **year** Year of census
- **plant** Plant id number
- **status** Plant status recorded in field: dead, dormant, recruit0 (with cotyledons only), recruit1, flowering or vegetative.
- **rose** Total number of rosettes
- **leaf** Total number of leaves
- **infl** Total number of infloresences or flowering stalks
- **fruits** Total number of mature fruits

Details

This sample data set includes census data from 10 of the 15 total demography plots established in 1995.

Source

Data set owners: Brook Milligan, Chris Stubben, Allan Strand

See Also

aq.trans for annual transitions with stage and fate in same row

Examples

```r
head2(aq.census)
sv <- table(aq.census$status, aq.census$year)
sv
stage.vector.plot(sv[-1, ], prop = FALSE)
```
Create a projection matrix for Aquilegia

Description

Creates a projection matrix for *Aquilegia* from annual transition data, assuming new seeds and seed bank seeds have an equal chance for successful germination and equal survival rates.

Usage

```r
aq.matrix(trans, recruits, summary = TRUE, seed.survival = 0.126, seed.bank.size = 10000, seeds.per.fruit = 120, ...)
```

Arguments

- `trans`: A data frame with transitions listing ordered stages and fates and counts of mature fruits.
- `recruits`: The number of observed recruits in year \( t + 1 \).
- `summary`: Output projection matrix and summaries. Otherwise output transition table with added individual fertilities.
- `seed.survival`: Estimated seed survival rate for both new seeds and seed bank. Default is 12.6 percent survival.
- `seed.bank.size`: Estimated size of the seed bank. Seed bank and new seeds contribute to a common germinant pool with equal chance for germination. Default is 10,000 seeds in seed bank.
- `seeds.per.fruit`: The number of seeds produced per mature fruit. Default is 120 seeds.
- `...`: Additional arguments passed to `projection.matrix`.

Details

Adds individual fertilities to annual transitions using a prebreeding census.

Value

If summary is TRUE, a list with

- `recruits`: total number of recruits
- `seed.survival`: seed survival rate
- `seed.bank`: total number of seeds in seed bank
- `seeds.from.plants`: total number of new seeds just released from fruits
- `recruitment.rate`: recruitment rate calculated as recruits/(seed.bank.size + seeds.from.plants)
- `A`: a projection matrix
lambda population growth rate
n initial population vector
n1 final population vector

If summary is FALSE, a data frame with individual fertilities added to the transition data frame only.

Author(s)
Chris Stubben

See Also
projection.matrix

Examples

x <- subset(aq.trans, year==1996)
## number of recruits in 1997
rec <- nrow(subset(aq.trans, year==1997 & stage == "recruit"))
aq.matrix(x, recruits=rec)
aq.matrix(x, recruits=rec, seed.survival=.7, seed.bank=3000)

aq.trans

Annual transition data for Aquilegia chrysantha

Description
Transition data listing stages and fates from Aquilegia chrysantha in Fillmore Canyon, Organ Mountains, New Mexico, 1996-2003.

Usage
aq.trans

Format
A data frame with 1637 rows on the following 9 variables:
plot Plot number
year Starting year of census
plant Plant id number
stage Initial stage class with ordered factor levels seed < recruit < small < large < flower.
leaf Total number of leaves
rose Total number of rosettes
fruits Total number of mature fruits
fate Final stage class or fate with levels seed < recruit < small < large < flower < dead
rose2 Final number of rosettes
The five stage classes include seeds in the seed bank, new recruits or seedlings, small vegetative plants with 1 rosette, large vegetative plants with 2 or more rosettes, and flowering plants. Stage classes were assigned to census plants using a combination of status and size data recorded in the field. See `demo(stage.classify)` for more details.

### Source
Data set owners: Brook Milligan, Chris Stubben, Allan Strand

### See Also
`aq.census`

### Examples
```r
head2(aq.trans)
sv <- table(aq.trans$stage, aq.trans$year)
addmargins(sv)
stage.vector.plot(sv[-1, ], prop = FALSE, main = "Aquilegia stage vectors")
## plot proportions with barplot
## use xpd to draw legend outside plot boundaries
op <- par(mar = c(5, 4, 4, 1), xpd = TRUE)
x <- barplot(prop.table(sv[-1, ], 2),
  las = 1, col = rainbow(4), yaxt = "n", xlab = "Year", ylab = "Proportion in stage class"
)
yrs <- substr(colnames(sv), 3, 4)
axis(1, x, yrs)
legend(2.7, 1.25, rev(rownames(sv)[-1]), fill = rev(rainbow(4)), bty = "n", ncol = 2)
par(op)
```

#### betaval
**Generate beta-distributed random numbers**

**Description**
Calculates a random number from a beta distribution and uses the R function pbeta(x,vv,ww).

**Usage**
```r
betaval(mn, sdev, fx = runif(1))
```

**Arguments**
- **mn**: mean rate between 0 and 1
- **sdev**: standard deviation
- **fx**: cumulative distribution function, default is a random number between 0 and 1
Details

This function is used by \texttt{vitalsim}

Value

a random beta value

Author(s)

Original MATLAB code by Morris and Doak (2002: 277-278), adapted to R by Patrick Nantel, 20 June 2005

Source

converted Matlab code from Box 8.3 in Morris and Doak (2002)

References


See Also

Beta Distribution \texttt{rbeta}

Examples

\begin{verbatim}
betaval(.5, sd=.05)
betaval(.5, sd=.05)
## histogram with mean=0.5 and sd=0.05
x <- sapply(1:100, function(x) betaval(0.5, 0.05))
hist(x, seq(0,1,.025), col="green", ylim=c(0,25), xlab="Value",
main="Beta distribution with mean=0.5 and sd=0.05")
# generates a graph similar to Figure 8.2 A in Morris & Doak (2002:264)
# a much simpler version of BetaDemo in Box 8.3
x <- matrix(numeric(3*1000), nrow=3)
sd <-c(.05, .25, .45)
for (i in 1:3){
  for (j in 1:1000){
    x[i,j]<-betaval(.5,sd[i])
  }
}
plot(0,0,xlim=c(0,1), ylim=c(0,.4), type='n', xlab='Value', ylab='Frequency'
main="Examples of beta distributions")
for (i in 1:3){
  h <- hist(x[i,], plot=FALSE, breaks=seq(0,1,.02))
  lines(h$mids, h$counts/1000, type='l', col=1+i, lwd=2, lty=i)
}
legend(0.5,0.4, c("(0.50, 0.05)", "(0.50, 0.25)", "(0.50, 0.45)"),
lty=1:3, lwd=2, col=2:4, title="mean and sd")
\end{verbatim}
boot.transitions  

**Bootstrap observed census transitions**

**Description**

Calculate bootstrap distributions of population growth rates (lambda), stage vectors, and projection matrix elements by randomly sampling with replacement from a stage-fate data frame of observed transitions.

**Usage**

```r
boot.transitions(transitions, iterations, by.stage.counts = FALSE, ...)
```

**Arguments**

- `transitions`: a stage-fate data frame with stage or age class in the current census, fate in the subsequent census, and one or more fertility columns.
- `iterations`: Number of bootstrap iterations.
- `by.stage.counts`: Resample transitions with equal probability (default) or by subsets of initial stage counts.
- `...`: additional options passed to `projection.matrix`.

**Value**

A list with 3 items:

- `lambda`: A vector containing bootstrap values for lambda.
- `matrix`: A matrix containing bootstrap transition matrices with one projection matrix per row.
- `vector`: A matrix containing bootstrap stage vectors with one stage vector per row.

**Author(s)**

Chris Stubben

**References**

see Morris and Doak 2005 in [http://esapubs.org/Archive/mono/M075/004/appendix-A.htm](http://esapubs.org/Archive/mono/M075/004/appendix-A.htm) for resampling by stage class counts.

**See Also**

`projection.matrix`
Examples

```r
## create stage-fate dataframe using merge and subset
trans01 <- subset(
  merge(test.census, test.census, by="plant", sort=FALSE),
  year.x==2001 & year.y==2002)
## format column and row names
trans01 <- trans01[,c(1:4,6)]
colnames(trans01)[2:5] <- c("year", "stage", "fruits", "fate")
rownames(trans01) <- 1:nrow(trans01)
# order stage columns corresponding to matrix
trans01$stage <- ordered(trans01$stage,
  levels = c("seedling", "vegetative", "reproductive"))
## add individual fertilities using prebreeding census with no seed bank
## based on the proportional reproductive outputs of flowering plants
## and the total number of seedlings at the end of the projection interval
seedlings <- nrow(subset(test.census, year==2002 & stage=="seedling")
trans01$seedling <- trans01$fruits/sum(trans01$fruits) * seedlings
trans01
## Step by step instructions for bootstrapping dataframe
n <- nrow(trans01)
set.seed(77)
x <- sample(n, replace=TRUE)
x
bt <- trans01[x,]
btp
projection.matrix(bt)
## or resample by stage class counts
lapply(split(trans01, trans01$stage, drop=TRUE),
  function(x) x[sample(nrow(x), replace=TRUE),])
## using boot.transitions
boot.transitions(trans01, 5)
boot.transitions(trans01, 5, by.stage=TRUE)
## Aquilegia example
x <- subset(aq.trans, year==1996)
# calculate lambda, seed survival and recruitment rate using aq.matrix
rec <- nrow(subset(aq.trans, year==1997 & stage="recruit")
aq.96 <- aq.matrix(x, rec)
# add individual fertilities to data frame only
aq.96.trans <- aq.matrix(x, rec, summary=FALSE)
# pass estimated transitions in aq.96 to projection matrix
aq.96.boot <- boot.transitions(aq.96.trans, 200,
  add=c(1,1, aq.96$seed.survival, 2,1, aq.96$recruitment.rate) )
# calculate percentile intervals using quantile()
ci <- quantile(aq.96.boot$lambda, c(0.025,0.975) )
aq.96$lambda
ci
# plot histogram
hist(aq.96.boot$lambda, col="green", xlab="Lambda",
  main=paste('Bootstrap estimates of population\ngrowth rate from 1996-1997'))
abline(v=ci, lty=3)
```
Projection matrices for a tropical understory herb

Description

Projection matrices for a tropical understory herb (*Calathea ovandensis*) for plots 1-4 in years 1982-1985 and the pooled matrix. Matrices were constructed using a post-breeding census with 8 size classes: seed, seedling, juvenile, pre-reproductive, and 4 reproductive classes divided by leaf area.

Usage

calathea

Format

A list of 17 matrices ordered by plot then year, with the pooled matrix last.

Source

Table 7 in Horvitz and Schemske (1995). The pooled matrix is from Table 8.

References


Examples

calathea

## Single matrix

```
calathea[[11]]
```

image2(calathea[[11]], text.cex = .8)
title(paste("Calathea", names(calathea[[11]])), line = 3)

## MEAN matrix (exclude pooled matrix)

```
mean(calathea[-17])
```

## all plot 1

calathea[1:4]

## all 1982 matrices

calathea[ grep("1982", names(calathea)) ]

# OR

```
# calathea[seq(1,16,4)]
# split(calathea, 1:4)[[1]]
```

## Growth rates -see Figure 7

```
x <- sapply(calathea[-17], lambda)
x <- matrix(x, nrow = 4, byrow = TRUE, dimnames = list(paste("plot", 1:4), 1982:1985))
x
```

```
matplot2(x, type = "b", ylab = "Growth rate", main = "Calathea growth rates")
```
countCDFxt  
*Count-based extinction probabilities and bootstrap confidence intervals*

**Description**

This function takes parameters derived from population counts and calculates the probability of extinction with bootstrap confidence intervals for a density-independent model, using a diffusion approximation.

**Usage**

```r
countCDFxt(mu, sig2, nt, Nc, Ne, tq = nt, tmax = 50, Nboot = 500, plot = TRUE)
```

**Arguments**

- `mu` estimated value of mean mu
- `sig2` estimated value of sample variance
- `nt` number of transitions in the data set
- `Nc` current population size
- `Ne` quasi-extinction threshold
- `tq` length of the census (in years), default is number of transitions
- `tmax` latest time to calculate extinction probability, default 50
- `Nboot` number of bootstrap samples for calculating confidence intervals for extinction probabilities, default 500
- `plot` draw extinction time CDF plot with log-scale on y-axis

**Details**

converted Matlab code from Box 3.4 in Morris and Doak (2002)

**Value**

The function plots the cumulative probabilities of quasi-extinction through time with 95% confidence intervals. It also returns a data frame with the extinction time CDF for the best parameter estimates (Gbest), and the lower and upper bootstrap confidence limits for extinction probabilities (Glo, Gup).

**Author(s)**

Adapted to R by Patrick Nantel, 4 May 2005, from program `extprob` of Morris & Doak (2002: 79-86)
**damping.ratio**

References

Dennis et al. 1991, Ecological Monographs 61: 115-143


See Also

extCDF

Examples

```r
## plot like Figure 3.8 in Morris and Doak (2002).
logN <- log(grizzly$N[-1]/grizzly$N[-39])
countCDFxt(mu=mean(logN), sig2=var(logN), nt=38, tq=38, Nc=99, Ne=20)
```

---

**damping.ratio**  
**Damping ratio**

Description

Calculate the damping ratio of a projection matrix

Usage

```r
damping.ratio(A)
```

Arguments

- `A`: A projection matrix

Details

see section 4.7 in Caswell (2001)

Value

Damping ratio

Note

The damping ratio is calculated by dividing the dominant eigenvalue by the eigenvalue with the second largest magnitude.

Author(s)

Chris Stubben
References


See Also

lambda

Examples

```r
## whale converges slowly to stable stage distribution
matplot2(pop.projection(whale, c(1,1,1,1), 60)$stage.vectors,
prop=TRUE, legend=NA,
main=paste("whale damping ratio = ", round(damping.ratio(whale),3) ) )
# Calathea - compare to Table 12 in Horvitz and Schemske (1995)
x <- sapply(calathea[-17], damping.ratio)
x <- matrix(x, nrow=4, byrow=TRUE, dimnames= list(paste("plot", 1:4), 1982:1985))
x
matplot2(x, type="b", ylab="Damping ratio", main="Calathea")
```

eigen.analysis

Eigenvalue and eigenvector analysis of a projection matrix

Description

Calculate population growth rate and other demographic parameters from a projection matrix model using matrix algebra

Usage

eigen.analysis(A, zero = FALSE)

Arguments

A 
A projection matrix

zero 
Set sensitivities for unobserved transitions to zero, default is FALSE

Details

The calculation of eigenvalues and eigenvectors partly follows Matlab code in section 4.8.1 (p. 107) in Caswell (2001). Since popbio version 2.0, each part returned by eigen.analysis is now included as a separate function.
Value

A list with 6 items

- `lambda1` dominant eigenvalue with largest real part
- `stable.stage` proportional stable stage distribution
- `sensitivities` matrix of eigenvalue sensitivities
- `elasticities` matrix of eigenvalue elasticities
- `repro.value` reproductive value scaled so `v[1]=1`
- `damping.ratio` damping ratio

Note

If matrix `A` is singular, then `eigen.analysis` will return elasticities, sensitivities, and reproductive values with NAs.

Author(s)

Original code by James Holland Jones, Stanford University, August 2005

References


See Also

eigen and `pop.projection`

Examples

```r
## Imprimitive matrix
A <- matrix(c(0,0,2,.3,0,0,0,.6,0), nrow=3,byrow=TRUE)
A
ev <- eigen(A)
ev$values
Mod(ev$values)
lmax <- which.max(Re(ev$values))
lmax
Re(ev$values)[lmax]
## damping ratio is NA
eigen.analysis(A)
## cycles every 3 years
stage.vector.plot(pop.projection(A, c(1,1,1), 10)$stage.vectors)
## Teasel
a <- eigen.analysis(teasel)
a
barplot(a$stable.stage, col="green", ylim=c(0,1),
       ylab="Stable stage proportion", xlab="Stage class", main="Teasel")
box()
```
elasticity

Elasticity analysis of a projection matrix

Description

Calculate the elasticities of eigenvalues to changes in the projection matrix elements.

Usage

```r
elasticity(A)
```

Arguments

- `A` A projection matrix

Details

see section 9.2 in Caswell (2001)

Value

An elasticity matrix

Author(s)

Chris Stubben

References


See Also

- `sensitivity`
Examples

```r
elas <- elasticity(teasel)
image2(elas, mar=c(1,3.5,5,1) )
  title("Teasel elasticity matrix", line=2.5)
# Summed elasticities for teasel.
# fertility in last column, stasis P on diagonal, and growth in bottom-left triangle
  c(F=sum(elas[,6]), P=sum(diag(elas)), G=sum(elas[row(elas)>col(elas)]))

elas <- elasticity(tortoise[["med.high"]])
image2(elas, mar=c(1,3.5,5,1), log=FALSE)
  title("Tortoise elasticity matrix", line=2.5)
# Summed elasticities for tortoise (see example 9.4)
# fertility in top row, stasis on diagonal, and growth on subdiagonal
  c(F=sum(elas[1,]), P=sum(diag(elas)), G=sum(elas[row(elas)==col(elas)+1]))
```

extCDF

Count-based extinction time cumulative distribution function

Description

Returns the extinction time cumulative distribution function using parameters derived from population counts.

Usage

```r
extCDF(mu, sig2, Nc, Ne, tmax = 50)
```

Arguments

- `mu` : estimated value of mean mu
- `sig2` : estimated value of sample variance
- `Nc` : current population size
- `Ne` : quasi-extinction threshold
- `tmax` : latest time to calculate extinction probability, default 50

Details

converted Matlab code from Box 3.3 and equation 3.5 in Morris and Doak 2002

Value

A vector with the cumulative probabilities of quasi-extinction from t=0 to t=tmax.

Author(s)

Chris Stubben
References

See Also
countCDFxt for bootstrap confidence intervals

Examples

```
logN <- log(grizzly$N[-1]/grizzly$N[-39])
mu <- mean(logN)
sig2 <- var(logN)
## grizzly cdf (log scale)
ex <- extCDF(mu, sig2, Nc=99, Ne=20)
plot(ex, log='y', type='l', pch=16, col="blue", yaxt='n',
xlab="Years", ylab="Quasi-extinction probability",
main="Yellowstone Grizzly bears")
pwrs <- seq(-15,-5,5)
axis(2, at = 10^pwrs, labels=parse(text=paste("10^", pwrs, sep = "")), las=1)
##plot like fig 3.10 (p 90)
n <- seq(20, 100, 2)
exts <- numeric(length(n))
for (i in 1:length(n)){
  ex <- extCDF(mu, sig2, Nc=n[i], Ne=20)
exts[i] <- ex[50]
}
plot(n, exts, type='l', las=1,
xlab="Current population size",
ylab="Probability of quasi-extinction by year 50")
```

fundamental.matrix  

**Fundamental matrix and age-specific survival**

Description
Age-specific survival calculations from stage-classified matrices. Includes the mean, variance and coefficient of variation (cv) of the time spent in each stage class and the mean and variance of the time to death.

Usage

```
fundamental.matrix(A, ...)
```

Arguments

- `A` projection matrix
- `...` additional items are passed to `splitA` and are used to split A into T and F matrices
Details

see section 5.3.1 in Caswell (2001).

Value

A list with 5 items

N  fundamental matrix or mean of the time spent in each stage class
var  variance of the time spent in each stage class
cv  coefficient of variation (sd/mean)
meaneta  mean of time to death
vareta  variance of time to death

Author(s)

Chris Stubben

References


See Also

see generation.time and net.reproductive.rate for other age-specific traits

Examples

fundamental.matrix(whale)
Details

see section 5.3.5 in Caswell (2001).

Value

Generation time. If the transition matrix is singular, then NA is returned.

Author(s)

Chris Stubben

References


See Also

see fundamental.matrix and net.reproductive.rate for other age-specific traits

Examples

generation.time(whale)
## fertilities in last column
generation.time(teasel, r=1:6, c=6)
## Plot 3 from Calathea
sapply(calathea[9:12], generation.time)

grizzly

Population sizes of grizzly bears in Yellowstone from 1959-1997

Description

Estimated number of adult female grizzly bears in the Greater Yellowstone population from 1959-1997.

Usage

grizzly

Format

A data frame with 39 rows on the following 2 variables.

year  Year of census
N    Estimated number of female grizzlies
Source

Table 3.1 in Morris and Doak 2002. Original data from Eberhardt et al. 1986 and Haroldson 1999.

References


Examples

grizzly
## plot like Fig 3.6 (p. 66)
plot(grizzly$year, grizzly$N,
    type = "o", pch = 16, las = 1, xlab = "Year",
    ylab = "Adult females", main = "Yellowstone grizzly bears"
)
## calaculate log(Nt+1/Nt)
nt <- length(grizzly$N) ## number transitions
logN <- log(grizzly$N[-1] / grizzly$N[-nt])
## Mean and var
mean <- mean(logN), var = var(logN))
## or using linear regression
## transformation for unequal variances (p. 68)
x <- sqrt(grizzly$year[-1] - grizzly$year[-length(grizzly$year)])
y <- logN / x
mod <- lm(y ~ 0 + x)
summary(mod)
## plot like Fig 3.7
plot(x, y,
    xlim = c(0, 1.2), ylim = c(-.3, .3), pch = 16, las = 1,
    xlab = expression(paste("Sqrt time between censuses ", (t[t + 1] - t[i])^{
        1 / 2
    })),
    ylab = expression(log(N[t + 1] / N[t]) / (t[t + 1] - t[i])^{
        1 / 2
    })),
    main = expression(paste("Estimating ", mu, " and ", sigma^2, " using regression")))
)
abline(mod)
## MEAN (slope)
mu <- coef(mod)
## VAR (mean square in analysis of variance table)
sig2 <- anova(mod)["Mean Sq"]
c(mean = mu, var = sig2)
## Confidence interval for mean (page 72)
confint(mod, 1)
## Confidence interval for sigma 2 (equation 3.13)
df1 <- length(logN) - 1
dx1 * sig2 / qchisq(c(.975, .025), df = df1)
## test for outliers using dffits (p.74)
dffits(mod)[dffits(mod) > 2 * sqrt(1 / 38) ]
## plot like fig 3.11
plot(grizzly$N[-nt], logN, 
  pch = 16, xlab = "Number of females in year T", 
  ylab = expression(log(N[t + 1] / N[t])), 
  main = "Grizzly log population growth rates"
) 
cor(grizzly$N[-nt], logN) 
abline(lm(logN ~ grizzly$N[-nt]), lty = 3)

head2

Return the first and last part of a matrix or dataframe

Description
Returns the first and last rows using output from both head and tail and separates the two parts with dots. Useful for viewing ordered datasets such as longitudinal census data.

Usage
head2(x, head = 3, tail = 1, dotrows = 1)

Arguments
- x: A matrix or dataframe
- head: The number of first rows
- tail: The number of last rows
- dotrows: The number of rows of dots

Value
A smaller object with first and last rows only

Author(s)
Chris Stubben

Examples
head2(aq.trans)
hudcorrs

Correlation matrices for Hudsonia vital rates

Description
Within year and between year correlation matrices from Hudsonia montana vital rates. Correlations were calculated from first 13 growth and survival rates only, since fertility rates vary little.

Usage
hudcorrs

Format
A list with 2 correlation matrices, corrin (within year correlation) and corrout (between year correlation).

Source
The correlation matrices in Morris and Doak 2002 include some correlations > 1. A corrected set of correlations was sent by the D. Doak on 8/4/2007.

References

See Also
vitalsim

Examples
hudcorrs

hudmxdef

Matrix definition program for Hudsonia vital rates

Description
Creates a projection matrix from Hudsonia vital rates (survival, growth, and reproduction). Growth rates are defined as a set of binomial choices as in Table 8.4 B in Morris and Doak (2002).

Usage
hudmxdef(vrs)
Arguments

vrs

Vital rate means in hudvrs

Value

A projection matrix

Author(s)

Chris Stubben

References


See Also

vitalsim

Examples

hudmxdef(hudvrs$mean)

---

hudsonia

*Projection matrices for mountain golden heather*

Description

Projection matrices for the mountain golden heather (*Hudsonia montana*) for years 1985 through 1988 with 6 size classes: seeds, seedlings, and 4 size classes divided by plant area.

Usage

hudsonia

Format

A list of 4 matrices from 1985-1988

Source

Table 6.7 in Morris and Doak 2002. The original data is from Frost 1990.

References

hudsiona
sapply(hudsiona, lambda)
## mean matrix
x <- mean(hudsiona)
image2(x, mar = c(1, 4, 5.5, 1))
title("Hudsiona mean matrix", line = 2.5)
lambda(x)
# variance
var2(hudsiona)

## Examples

hudvrs

### Description

Best Kendall estimates of vital rate means (9 growth, 4 survival, and 11 fertility rates) for *Hudsonia montana*.

### Usage

hudvrs

### Format

A data frame with 24 rows and 2 columns:

- **mean**: vital rate means
- **var**: vital rate variances

### Source

Data listed in Box 8.10 for the `vitalsim` function. See also Table 8.5 in Morris and Doak (2002).

### References


### Examples

hudvrs
hudmxdef(hudvrs$mean)
**Display a matrix image**

**Description**

Creates a grid of colored rectangles to display a projection, elasticity, sensitivity or other matrix.

**Usage**

```r
image2(x, col = c("white", rev(heat.colors(23))), breaks, log = TRUE,
    border = NA, box.offset = 0.1, round = 3, cex, text.cex = 1,
    text.col = "black", mar = c(1, 3, 3, 1), labels = 2:3,
    label.offset = 0.1, label.cex = 1, srt = 90)
```

**Arguments**

- **x**  
  A numeric matrix with row and column names
- **col**  
  A vector of colors for boxes
- **breaks**  
  A numeric vector of break points or number of intervals into which `x` is to be cut. Default is the length of `col`
- **log**  
  Cut values in `x` using a log scale, default TRUE
- **border**  
  The border color for boxes, default is no borders
- **box.offset**  
  Percent reduction in box size (a number between 0 and 1), default is 10% reduction
- **round**  
  Number of decimal places to display values of `x` in each box
- **cex**  
  Magnification size of text and labels, if specified this will replace values in both `text.cex` and `label.cex`
- **text.cex**  
  Magnification size of text in cells only
- **text.col**  
  Color of text in cells, use NA to skip text labels
- **mar**  
  Margins on four sides of plot
- **labels**  
  A vector giving sides of the plot (1=bottom, 2=left, 3=top, 4=right) for row and column labels
- **label.offset**  
  Amount of space between label and boxes
- **label.cex**  
  Magnification size of labels
- **srt**  
  String rotation for labels on top and bottom of matrix

**Value**

A image plot of the matrix
Note

`#` The minimum value in `x` is usually assigned to the first color category and the rest of the values are then cut into equally spaced intervals. This was added to show transitions with very low probabilities in a new color category, e.g., `2e-06` would usually be grouped with `0` using `image`. Note if all elements > 0, then the first color will not be used.

Author(s)

Chris Stubben

See Also

`image`

Examples

```r
A <- calathea[[11]]
op <- par(mfrow=c(2,2))
image2(A, text.cex=.8)
## with gray border and labels on bottom right
image2( A, text.cex=.8, border="gray70", labels=c(1,4), mar=c(3,1,1,3))
## no text or box offset
image2( A, box.offset=0, text.col=NA)
# set zeros to NA to print everything but zero
A[A == 0] <- NA
image2( A, box.offset=0, text.cex=.8)
## if comparing two or more matrices, get the log10 range
## of values (not including zero) and pass to breaks
x <- unlist(calathea[-17])
x <- log10(range(x[x!=0]))
par(mfrow=c(4,4))
for(i in 1:16){
  A <- calathea[[i]]
  A[A == 0] <- NA
  image2(A, cex=.7, box.offset=0, breaks=seq(x[1], x[2], len=24))
  title(names(calathea[i]), line=3)
}
par(op)
```

---

**Kendall**

Find the best Kendall’s estimates of mean and environmental variance for beta-binomial vital rates

**Description**

Finds the best estimates of mean and environmental variance for beta-binomial vital rates, using a brute force search for the best adjusted estimates from a very large number of combinations of different possible mean and variance values.
Usage

```r
Kendall(rates, grades = 1000, maxvar = 0.2, minvar = 1e-05, maxmean = 1, minmean = 0.01)
```

Arguments

- `rates`: a matrix or dataframe with four columns: Rate identifier, Year, Total number of starting individuals, Number growing (or surviving).
- `grades`: number of different levels of means and variances to try, default is 1000.
- `maxvar`: maximum variance to search over, default is 0.20. The maximum possible is 0.25 and searching a narrower range will improve the accuracy of the answer.
- `minvar`: minimum variance to search, default is 0.00001.
- `maxmean`: maximum limit on the mean values to search, default 1.
- `minmean`: minimum limit on the mean values to search, default 0.01.

Details

converted Matlab code from Box 8.2 in Morris and Doak (2002)

Value

A list with estimates and confidence intervals

- `est`: a matrix with 5 columns: (1) estimated mean, (2) Kendall’s MLE mean, (3) estimated variance, (4) Kendall’s MLE variance, (5) Kendall’s unbiased MLE variance.
- `ci`: a matrix with 95% confidence limits for the Kendall’s mean and unbiased variance estimates with 4 columns: (1) low and (3) high mean limits, (3) low and (4) high variance limits.

Note

may deliver warning messages of 'no finite arguments to min; returning Inf', indicating use of very low values for variance, but this is not a malfunction.

Author(s)

Adapted to R from Morris \\& Doak (2002: 267-270) by Patrick Nantel.

References


### lambda

Population growth rate

**See Also**

varEst

**Examples**

```r
## desert tortoise input from Box 8.2 - compare results to Table 8.3
tor <- data.frame(rate=rep(c("g4","g5","g6"), each=3),
    year=rep(1:3,3),
    start=c(17,15,7,22,19,4,32,31,10),
    grow=c(8,1,0,5,5,0,2,1,0)
)
## use fewer grades for faster loop
tor.est<-Kendall(tor, grades=200)
tor.est
wp.est <- Kendall(woodpecker, grades=200)
wp.est
```

---

### Description

Calculates the population growth rate of a projection matrix

### Usage

```r
lambda(A)
```

### Arguments

- `A` A projection matrix

### Details

see section 4.4 in Caswell (2001)

### Value

The dominant eigenvalue

### Note

The built-in `eigen` function returns eigenvalues in decreasing order of magnitude or modulus. The dominant eigenvalue of imprimitive matrices with $d$ eigenvalues of equal modulus is the one with the largest real part (`which.max(Re(eigen(A)$values))`).
Author(s)

Chris Stubben

References


See Also

eigen and pop.projection

Examples

A <- matrix(c(0,0,2,.3,0,0,.6,0), nrow=3,byrow=TRUE)
lambda(A)
Re(eigen(A)$values)
sapply(hudsonia, lambda)

lnorms

Generate random lognormal values for fertility rates

Description

Converts standard normal random values to lognormals with defined means and variances

Usage

lnorms(n, mean = 2, var = 1)

Arguments

n number of observations
mean mean value of the fertility rate, default 2
var variance of the vital rate (not standard deviation), default 1

Details

converted Matlab code from Box 8.4 in Morris and Doak (2002)

Value

A vector of random lognormal values

Note

This function could probably be replaced with built-in functions for the Log Normal Distribution rlnorm
Author(s)


References


See Also

stretchbetaval

Examples

lnorms(1)
# Generate lognormal random fertilities
# for a population of 1000 mature individuals with mean fertility of
# 3 and inter-individual variance in fertility of 1.5.
rndfert <- lnorms(1000, 3, 1.5)
summary(rndfert)
hist(rndfert, 40, main="Lognormal random fertilities",
xlab="Fertility rate", col="blue")

logi.hist.plot    Plot logistic regression

Description

Plot combined graphs for logistic regressions

Usage

logi.hist.plot(independ, depend, logi.mod = 1, type = "dit",
    boxp = TRUE, rug = FALSE, ylabel = "Probability",
    ylabel2 = "Frequency", xlabel = "", mainlabel = "", las.h = 1,
    counts = FALSE, ...)

Arguments

independ    explanatory variable
depend     dependent variable, typically a logical vector
logi.mod    type of fitting, 1 = logistic; 2 = "gaussian" logistic
type       type of representation, "dit" = dit plot; "hist" = histogram
boxp        TRUE = with box plots, FALSE = without
rug TRUE = with rug plots, FALSE = without
ylabel y-axis label
ylabel2 2nd y-axis label
xlabel x-axix label
mainlabel overall title for plot
las.h orientation of axes labels (0 = vertical, 1 = horizontal
counts add counts above histogram bars
... additional options passed to logi.hist

Value
A logistic regression plot

Author(s)
M. de la Cruz Rot

References

Examples
aq.trans$survived <- aq.trans$fate!="dead"
a <- subset(aq.trans, leaf<50 & stage!="recruit", c(leaf,survived))
logi.hist.plot(a$leaf, a$survived,
  type="hist", boxp=FALSE, counts=TRUE, int=10,
  ylabel="Survival probability", ylabel2="Number of plants",
  xlab="Number of leaves")
b <- glm(survived ~ leaf, binomial, data=a)
summary(b)

LTRE

Description
Evaluate sensitivities in a fixed Life Table Response Experiment (LTRE)

Usage
LTRE(trts, ref)
**Arguments**

- `trts` A treatment matrix or a list of two or more treatment matrices
- `ref` A reference matrix

**Details**

Sensitivities are evaluated midway between the treatment and reference matrices as described in section 10.1.1 in Caswell (2001).

**Value**

A matrix of contributions (equation 10.4 in Caswell) or a list of matrices with one matrix of contributions per treatment

**Note**


**Author(s)**

Chris Stubben

**See Also**

Check the demo(Caswell) for variance decomposition in a random design using killer whale.

**Examples**

```r
#### Calathea ovandensis
calathea_pool <- calathea[['pooled']]
## Create plots like FIGURE 7 in Horvitz et al 1997
plots <- split(calathea[-17], rep(1:4,each=4))
## use Mean matrix since pooled not available by plot
plots <- lapply(plots, mean)
Cm <- LTRE(plots, calathea_pool)
pe <- sapply(Cm, sum)
barplot(pe, xlab="Plot", ylab="Plot effect", ylim=c(-.25, .25),
col="blue", las=1)
abline(h=0)
box()
title(expression(italic("Calathea ovandensis")))

##YEARS -- split recycles vector
yrs <- split(calathea[-17], 1:4)
yrs <- lapply(yrs, mean)
```
names(yrs) <- 1982:1985
cm <- LTRE(yrs, calathea_pool)
ye <- sapply(cm, sum)
barchart(ye, xlab="Year", ylab="Year effect", ylim=c(-.25, .25), col="blue", las=1)
abline(h=0)
box()
title(expression(italic("Calathea ovandensis"))

## INTERACTION

Cm <- LTRE(calathea[-17], calathea_pool)
ie <- sapply(Cm, sum)

## minus plot, year effects
ie<- ie - rep(ie, each=4) - rep(ye, 4)
names(ie) <- NULL
names(ie)[seq(1,16,4)] <- 1:4
barchart(ie, xlab="Plot (years 82-83 to 85-86)", ylab="Interaction effect", ylim=c(-.25, .25), col="blue", las=1)
abline(h=0)
box()
title(expression(italic("Calathea ovandensis")))

#### Mimulus

## Pooled M. cardinalis reference matrix kindly provided by Amy Angert 1/2/2008.
m_card_pool <- matrix(c(1.99e-01, 8.02e+02, 5.82e+03, 3.05e+04,
                        2.66e-05, 7.76e-02, 2.31e-02, 1.13e-03,
                        7.94e-06, 8.07e-02, 3.22e-01, 2.16e-01,
                        2.91e-07, 1.58e-02, 1.15e-01, 6.01e-01), byrow=TRUE, nrow=4)

## Population effects using pooled population matrices
card <- subset(monkeyflower, species=="cardinalis" & year=="pooled")

## split rows into list of 4 matrices
Atrt <- lapply(split(as.matrix(card[,4:19]), 1:4), matrix, nrow=4, byrow=TRUE)
names(Atrt) <- card$site
Cm <- LTRE(Atrt, m_card_pool)
x <- sapply(Cm, sum)
x
names(x) <- c("BU", "RP", "WA", "CA")

## Plot like Figure 2A in Angert (2006)
op <- par(mar=c(5,5,4,1))
barchart(x, xlab="Population", ylab="", xlim=c(0,6.5), ylim=c(-.4, .4),
        las=1, space=.5, col="blue")
abline(h=0)

mtext(expression(paste(sum(a[i,j]), " contributions")), 2, 3.5)
title(expression(paste(italic("M. cardinalis"), " Population effects")))
box()

## and Plot like Figure 3A
x <- matrix(unlist(Cm), nrow=4, byrow=TRUE)
colnames(x) <- paste("a", rep(1:4, each=4), 1:4, sep="")
bp <- barchart(x[1:2,], beside=TRUE, ylim=c(-.2,.2), las=1,
xlab="Transition", ylab="", xaxt="n")
mtext(expression(paste("Contribution of ", a[i,j], " to variation in ", lambda)), 2, 3.5)

## rotate labels
text(bp[1,]-.5, -2.2, labels=colnames(x), srt=45, xpd=TRUE)
title(expression(paste(italic("M. cardinalis"), " Range center"))))
matplot2

box()
par(op)

---

**matplot2**

*Plot a matrix*

**Description**

Plot the rows of a matrix. Useful for displaying a matrix of stage vectors, survival rates and sensitivities.

**Usage**

```r
matplot2(x, proportions = FALSE, legend = "topright", xlab = NULL,
         ylab = NULL, type = "l", las = 1, pch = c(15:18, 1:3), lwd = 1,
         lty = 1:nrow(x), col = rainbow(nrow(x)), lcex = 1, lbty = "o",
         lcol = 1, ltitle = NULL, lsort = TRUE, ...)```

**Arguments**

- `x` a matrix
- `proportions` If `TRUE`, then plot proportional changes
- `legend` a `legend` keyword or vector of x,y coordinates, defaults to top-right corner
- `xlab` a label for the x axis
- `ylab` a label for the y axis
- `type` plot type, default line
- `las` style of axis labels, default horizontal
- `pch` point types
- `lwd` line width
- `lty` line type
- `col` color
- `lcex` legend size expansion
- `lbty` legend box type
- `lcol` number of columns in legend
- `ltitle` legend title
- `lsort` sort legend by decreasing order of mean number in row
- `...` additional options are passed to `plot` function

**Value**

A matrix plot
Note

Only a few basic legend options are available. For more control, set legend=NA and run separately.

Author(s)

Chris Stubben

See Also

matplot and stage.vector.plot

Examples

# survival rates
x <- calathea[9:12]
x <- sapply(x, function(x) colSums(splitA(x, r=1:2)$T))
matplot2(t(x), legend="bottomright", ylab="Survival",
main="Calathea survival curves")
# Growth rates - do not sort legend
x <- sapply(calathea[-17], lambda)
x <- matrix(x, nrow=4, byrow=TRUE, dimnames=list(paste("plot", 1:4), 1982:1985))
matplot2(x, type='b', lsort=FALSE, ylab="Growth rate", main="Calathea growth rates")
# Convergence to stable stage (excluding seeds)
x <- pop.projection(calathea[[7]], rep(1,8), 10)
matplot2(x$stage.vectors[-1,], prop=TRUE,
main="Calathea stage vectors", lcex=.7)

matrix2

Square matrices

Description

Create a square matrix from a given set of values

Usage

matrix2(x, stages, byrow = TRUE)

Arguments

x a vector of matrix elements
stages a vector of row names (also assigned to columns)
byrow fill matrix by rows , default TRUE

Value

a square matrix
mean.list

Author(s)
Chris Stubben

See Also
matrix

Examples

# Centaurea corymbosa from Freville 2004
ceco <- c(0, 0.5, 0.905, 0.368, 0.639, 0.025, 0.001, 0.152, 0.051)
stages <- c("seedling", "vegetative", "flowering")
# shortcut for
matrix(ceco, nrow=3, byrow=TRUE, dimnames=list(stages, stages))
matrix2(ceco, stages)

mean.list  Mean matrix

Description
Calculates mean matrix from a list of matrices

Usage

## S3 method for class 'list'
mean(x, ...)

Arguments

x  A list of two or more matrices
...

Additional arguments passed to rowMeans

Details
Returns the mean matrix from a list of matrices using a combination of unlist and rowMeans. See example for details.

Value
The mean matrix

Note
S3 method for the mean of a list of matrices
**monkeyflower**

**Author(s)**

Chris Stubben

**See Also**

var2

**Examples**

```
mean(hudsonia)
# or
x <- matrix(unlist(hudsonia), ncol=length(hudsonia) )
matrix2(rowMeans(x), colnames(hudsonia[[1]]))
```

---

**monkeyflower**

*Projection matrices for monkeyflower*

**Description**

Pooled and annual projection matrices of central and marginal populations of monkeyflowers (*Mimulus cardinalis* and *M. lewisii*)

**Usage**

monkeyflower

**Format**

A data frame with 32 matrices, arranged with one matrix per row

species  M. cardinalis or M. lewisii
site     Study site
year     Start year of projection interval or pooled for all three years
a11      matrix element a11; seed to seed transition or seed bank survival
a12      matrix element a12; small nr to seed - fertility
a13      matrix element a13; large nr to seed - fertility
a14      matrix element a14; reprod to seed - fertility
a21      matrix element a21; seed to small nr - growth
a22      matrix element a22; small nr to small nr - stasis
a23      matrix element a23; large nr to small nr - regress
a24      matrix element a24; reprod to small nr - regress
a31      matrix element a31; seed to large nr - growth
a32      matrix element a32; small nr to large nr - growth
a33 matrix element a33; large nr to large nr - stasis
a34 matrix element a34; reprod to large nr - regress
a41 matrix element a41; seed to reprod - growth
a42 matrix element a42; small nr to reprod - growth
a43 matrix element a43; large nr to reprod - growth
a44 matrix element a44; reprod to reprod - stasis

Details

Matrix constructed using a post-breeding census with four stage classes: Seeds, small non-reproductive, large non-reproductive, and reproductive.

Source

http://www.esapubs.org/archive/ecol/E087/126/appendix-E.htm

References


Examples

monkeyflower

## convert M. cardinalis rows to list of 16 matrices
A <- subset(monkeyflower, species == "cardinalis")
# use as.matrix to convert data.frame to numeric matrix
A <- split(as.matrix(A[, 4:19]), paste(A$site, A$year))
stages <- c("seed", "sm.nr", "lg.nr", "repro")
## convert to list of 16 matrices
A <- lapply(A, matrix, nrow = 4, byrow = TRUE, dimnames = list(stages, stages))
A[8]
image2(A[[8]], round = 8, mar = c(1, 3, 4.5, 1))
title(paste("M. cardinalis - ", names(A[8])), line = 2.5)
## plot like figure 1A
x <- matrix(sapply(A, lambda), ncol = 4)
colnames(x) <- c("BU", "CA", "RP", "WA")
rownames(x) <- c(2000:2002, "pooled")
x <- x[, c(1, 3, 4, 2)]
cols <- gray(0:3 / 3)[c(1, 3, 2, 4)]
barplot(x, beside = TRUE, las = 1, col = cols, ylim = c(0, 2),
ylab = "Population growth rate", main = "Mimulus cardinalis")
box()
abline(h = 1, lwd = .5)
legend(1, 1.95, rownames(x), fill = cols, bty = "n")
Incorporate demographic stochasticity into population projections

Description

Generates multinomial random numbers for state transitions and lognormal or binomial (for clutch size=1) random numbers for fertilities and returns a vector of the number of individuals per stage class at t+1.

Usage

```r
multiresultm(n, T, F, varF = NULL)
```

Arguments

- `n`: the vector of numbers of individuals per class at t
- `T`: a transition T matrix
- `F`: a fertility F matrix
- `varF`: a matrix of inter-individual variance in fertilities, default is NULL for simulating population where clutch size = 1, so that fertilities give the probabilities of birth

Details

Adapted from Matlab code in Box 8.11 in Morris and Doak (2002) and section 15.1.3 in Caswell (2001)

Value

A vector of the number of individuals per class at t+1.

Author(s)

Patrick Nantel

References


Examples

\[
x <- \text{splitA}(\text{whale})
\]
\[
\text{whaleT} <- x$T
\]
\[
\text{whaleF} <- x$F
\]
\[
\text{multiresultm}(c(1,9,9,9), \text{whaleT}, \text{whaleF})
\]
\[
\text{multiresultm}(c(1,9,9,9), \text{whaleT}, \text{whaleF})
\]

## create graph similar to Fig 15.3 a

\[
\text{reps} <- 10 \quad \# \text{number of trajectories}
\]
\[
\text{tmax} <- 200 \quad \# \text{length of the trajectories}
\]
\[
\text{totalpop} <- \text{matrix}(0, \text{tmax}, \text{reps}) \quad \# \text{initializes totalpop matrix to store trajectories}
\]
\[
\text{nzero} <- c(1,1,1,1) \quad \# \text{starting population size}
\]
\[
\text{for} (j \text{ in 1:reps}) {
\text{n} <- \text{nzero}
\text{for} (i \text{ in 1:tmax}) {
\text{n} <- \text{multiresultm}(\text{n}, \text{whaleT}, \text{whaleF})
\text{totalpop}[i,j] <- \text{sum(n)}
\}
}
\]
\[
\text{matplot(\text{totalpop}, \text{type} = '1', \text{log}="y",}
\]
\[
\text{xlab = 'Time (years)', ylab = 'Total population')}
\]

nematode

Population densities for the sugarbeet cyst nematode

Description

A time-series of population vectors for the sugarbeet cyst nematode *Heterodera schachtii*. Individuals were classified into three stages (J2, J3+J4, and adult) and densities (per 60 cc of soil) were averaged over four replicates, measured every two days, for 10 days.

Usage

nematode

Format

A matrix listing densities from 3 stage classes over 6 time periods

Source


References

Sinauer, Sunderland, Massachusetts.
net.reproductive.rate

See Also

QPmat

Examples

nematode
stage.vector.plot(nematode,
    prop = FALSE, log = "y", ylim = c(.3, 200),
    xlab = "Time", ylab = "Nematode density"
)

net.reproductive.rate  Net reproductive rate

Description

Calculates the net reproductive rate of a stage classified matrix using the dominant eigenvalue of
the matrix R

Usage

net.reproductive.rate(A, ...)

Arguments

A          projection matrix

...        additional items are passed to splitA and are used to split A into T and F ma-
trices

Value

Net reproductive rate. If the transition matrix is singular, then NA is returned.

Author(s)

Chris Stubben

References

edition. Sinauer, Sunderland, Massachusetts, USA.

See Also

see fundamental.matrix and generation.time for other age-specific traits
pfister.plot  

Examples

```r
net.reproductive.rate(whale)  
## fertilities in last column
net.reproductive.rate(teasel, r=1:6, c=6)  
## Plot 3 from Calathea - values are not the same as p. 105 in Caswell.
sapply(calathea[9:12], net.reproductive.rate)
```

Description

Create log-log plots of variance vs. sensitivity and CV vs. elasticity in matrix elements. Plots are based on Figure 2 in Pfister(1998)

Usage

```r
pfister.plot(A)
```

Arguments

- `A`: A list of two or more projection matrices

Details

Calculates mean, variance and coefficient of variation (CV) of matrix elements from a list of two or more projection matrices. The sensitivity and elasticity matrices are then calculated from the mean matrix using `eigen.analysis`

Value

Creates two log-log plots similar to Figure 2 in Pfister(1998) and outputs a data.frame with 5 columns listing mean, variance, CV, sensitivity and elasticity for matrix elements with a mean and variance > 0

Author(s)

Chris Stubben

References

Examples

```r
## 4 Hudsonia matrices
pfister.plot(hudsonia)

## 3 Mimulus cardinalis matrices at Carlon
mim <- subset(monkeyflower, species == "cardinalis" &
  site == "Carlon" & year != "pooled", select = c(4:19))

## convert data frame to list of matrices using split
mim1 <- split(mim, 2000:2002)
mim2 <- lapply(mim1, matrix, nrow=4, byrow=TRUE)
vr1 <- pfister.plot(mim2)

## PLOT using labels
plot(vr1$cv, vr1$elas, xlab="CV", ylab="Elasticity", log="xy", type='n'

# Split matrix elements into transitions representing F (fertility),
# S (survival), G (growth), and R (retrogression).
# Fertility on top row, survival on diagonal, growth is above diagonal
# and retrogression below diagonal.
rownames(vr1)
y2 <- expression(S[11],G[21],G[31],G[41],
  F[12],S[22],G[32],G[42],
  F[13],R[23],S[33],G[43],
  F[14],R[34],S[44])
text(vr1$cv, vr1$elas, y2)

### add trend line
abline(lm(log10(vr1$elas)~log10(vr1$cv)), col="red")

## include Spearman's rank correlation
a <- cor.test(vr1$cv, vr1$elas, method="spearman")
a
text(10, .0015, substitute(rho == x, list(x=round(a$estimate,2))), col="blue")
```

---

**pop.projection**

*Calculate population growth rates by projection*

**Description**

Calculates the population growth rate and stable stage distribution by repeated projections of the equation $n(t+1) = An(t)$

**Usage**

```r
pop.projection(A, n, iterations = 20)
```

**Arguments**

- **A**
  - A projection matrix
- **n**
  - An initial age or stage vector
- **iterations**
  - Number of iterations
Details

Eventually, structured populations will convergence to a stable stage distribution where each new stage vector is changing by the same proportion (lambda).

Value

A list with 5 items

- lambda: Estimate of lambda using change between the last two population counts
- stable.stage: Estimate of stable stage distribution using proportions in last stage vector
- stage.vector: A matrix with the number of projected individuals in each stage class
- pop.sizes: Total number of projected individuals
- pop.changes: Proportional change in population size

Author(s)

Chris Stubben

References

see section 2.2 in Caswell 2001

See Also

stage.vector.plot to plot stage vectors

Examples

```r
## mean matrix from Freville et al 2004
stages <- c("seedling", "vegetative", "flowering")
A <- matrix(c( 0, 0, 5.905, 0.368, 0.639, 0.025, 0.001, 0.152, 0.051 ), nrow=3, byrow=TRUE, dimnames=list(stages,stages))
n <- c(5,5,5)
p <- pop.projection(A,n, 15)
p
damping.ratio(A)
stage.vector.plot(p$stage.vectors, col=2:4)
A <- whale
#n <- c(4,38,36,22)
#n <- c(5,5,5,5)
p <- pop.projection(A,n, 15)
p
stage.vector.plot(p$stage.vectors, col=2:4, ylim=c(0, 0.6))
## convergence is slow with damping ratio close to 1
damping.ratio(A)
pop.projection(A, n, 100)$pop.changes
```
Construct projection matrix models using transition frequency tables

Description

Construct an age or stage-structure projection model from a transition table listing stage in time $t$, fate in time $t+1$, and one or more individual fertility columns.

Usage

projection.matrix(transitions, stage = NULL, fate = NULL, fertility = NULL, sort = NULL, add = NULL, TF = FALSE)

Arguments

- **transitions**: a stage-fate data frame with stage or age class in the current census, fate in the subsequent census, and one or more fertility columns
- **stage**: a column name or position of the stage column in the stage-fate data frame. Defaults to "stage".
- **fate**: name of the fate column in the stage-fate data frame. Defaults to "fate"
- **fertility**: one or more names of fertility columns in the stage-fate data frame. By default, any column names matching stage class names are assumed to contain individual fertilities
- **sort**: a vector listing stage classes that correspond to the rows and columns of the desired projection matrix. Currently, names in this vector must match a level in the stage column. Also, this option should only be used if stages are not ordered, since the default is to sort by levels in the stage column.
- **add**: a vector listing row, column and value, used to add estimated transitions to the transition matrix (e.g., a transition from seed bank to seedling). May be repeated.
- **TF**: output separate transition (T) and fertility (F) matrices. Default is FALSE and outputs a single projection matrix $A$

Details

The state transition rates are estimated using transition frequency tables (see section 6.1.1, Caswell 2001), so this technique will most likely apply to demographic studies of plants or other sessile organisms where individuals are tagged and then consistently relocated in annual censuses. The fertility rates are calculated by averaging individuals fertilities by stage class; therefore, some care should be taken to correctly estimate individual fertilities based on the timing of the census.

Value

The default output is a single projection matrix $A$. If the TF flag is true, then a list with 2 items where $A=T+F$
projection.matrix

Note

Individual fertilities should be the total number of offspring at the end of the census interval. Therefore, fertilities should include offspring survival in a prebreeding censuses (and more than one offspring class may be present). In a postbreeding census, new offspring were born just before the census, so the fertility rate is just the number of offspring in this case.

Author(s)

Chris Stubben

Examples

trans01 <- subset(merge(test.census, test.census, by = "plant", sort = FALSE),
year.x==2001 & year.y==2002 )
## Add individual fertilities using "anonymous reproduction" based on the
## proportional reproductive outputs of flowering plants and the total number
## of seedlings at the end of the projection interval
trans01$seedferts <- trans01$fruits.x/sum(trans01$fruits.x) * 5
trans01
stages <- c("seedling", "vegetative", "reproductive")
## three ways to specify columns
projection.matrix(trans01, stage.x, stage.y, seedferts, stages)
projection.matrix(trans01, 3, 6, 8, c(3,4,2))
projection.matrix(trans01, "stage.x", "stage.y", "seedferts", stages)
## BEST to use column default (fertility column (seedling) now matches stage class name)
names(trans01)[c(3, 6, 8)] <- c("stage", "fate", "seedling")
# AND order stages in dataframe
trans01$stage <- ordered(trans01$stage, stages)
projection.matrix(trans01)
projection.matrix(trans01, TF=TRUE)
## Example using Aquilegia data
sf <- subset(aq.trans, year==1998 & plot==909, c(year, plant, stage, fruits, fate))
## rows and columns of final matrix
levels(sf$stage)
## seedlings next year
seedlings <- nrow(subset(aq.trans, plot==909 & year==1999 & stage="recruit"))
## ADD individual fertility estimates for recruits and seeds assuming seed bank and
## new seeds contribute to a common germinant pool with equal chance of recruitment
seed.survival <- .4
seed.bank.size <- 1000
seeds.per.fruit <- 50
seeds.from.plants <- sum(sf$fruits)*seeds.per.fruit
recruitment.rate <- seedlings/(seed.bank.size + seeds.from.plants)
## add two fertility columns
sf$recruit <- sf$fruits/sum(sf$fruits) * seeds.from.plants * recruitment.rate
sf$seed <- sf$fruits * seeds.per.fruit * seed.survival
## add seed bank survival and seed bank recruitment rate to transition matrix
A <- projection.matrix(sf, add=c(1,1, seed.survival, 2,1, recruitment.rate ))
A
max(Re(eigen(A)$values))
**QPmat**

*Build a projection matrix from a time series of individuals (or densities) per stage*

**Description**

Builds one projection matrix from a time series of number (or densities) of individuals per stage (size classes or life stages) using Wood’s quadratic programming method. The matrix model also requires a constraint matrix \( C \), vector \( b \), and vector listing nonzero elements of desired projection matrix.

**Usage**

\[
\text{QPmat}(nout, C, b, \text{nonzero})
\]

**Arguments**

- **nout**: A time series of population vectors
- **C**: constraint matrix
- **b**: \( b \) vector
- **nonzero**: indices of the non-zero elements of the transition matrix (counting by column)

**Details**

converted Matlab code from Example 6.3 in Caswell (2001)

**Value**

A projection matrix.

**Note**

This function requires \texttt{solve.QP} in the \texttt{quadprog} package.

**Author(s)**

Adapted to R by Patrick Nantel

**Examples**

```r
## Not run:
## list nonzero elements
nonzero <- c(1, 2, 5, 6, 7, 9)
## create C matrix
C <- rbind(diag(-1,6), c(1,1,0,0,0,0), c(0,0,1,1,0,0), c(0,0,0,0,0,1))
## calculate b (transpose is not necessary - either way works)
b <- apply(C, 1, max)
QPmat(nematode, C, b, nonzero)
```
reproductive.value

## End(Not run)

---

reproductive.value  Reproductive value

**Description**

Calculates the reproductive values of a projection matrix

**Usage**

```r
reproductive.value(A)
```

**Arguments**

- `A` A projection matrix

**Details**

see section 4.5 in Caswell (2001)

**Value**

A vector containing the scaled reproductive values so v[1]=1

**Author(s)**

Chris Stubben

**References**


**Examples**

```r
v <- reproductive.value(teasel)
v
dotchart(log10(v), pch=16, xlab="Reproductive value (log10)")
```
resample  

Resample a projection matrix

Description

Resample a projection matrix using a multinomial distribution for transitions and a log normal distribution for fertilities

Usage

resample(A, n, fvar = 1.5, ...)

Arguments

A  
a projection matrix
n  
either a stage vector with the number of transitions to sample in each column or a single value that is applied to all columns
fvar  
either a vector of different fertility variances or a single variance of fertility (default 1.5) that is applied to all rates
...  
additional items are passed to splitA and are used to split A into T and F matrices

Details

The projection matrix A is first split into separate transition and fertility matrices. Dead fates are added to the transition matrix and the columns are then sampled from a Multinomial distribution based on the size in each corresponding stage class in n. The fertility rates are sampled from a Log Normal distribution using the lnorms function. The variance can be a single value which is applied to all rates, or vector of different values to apply to each rate. In this case, the values are recycled to match the number of non-zero fertilities.

Value

A resampled projection matrix

Note

see section 12.1.5.2 on parametric bootstrap in Caswell (2001)

Author(s)

Chris Stubben

See Also

boot.transitions
Examples

```r
A <- hudsonia[[1]]
lambda(A)
## NOTE fertilities are in first two rows, so use r=1:2 for splitting this matrix
resample(A, 100, r=1:2)
## set higher fvar in stage 4 and 6
## because there are two fertilities per stage (8 total), need to repeat values
resample(A,1000, fvar=c(1.5, 1.5, 3, 3), r=1:2)
## OR resample based on number of plants surveyed
# data from table 6.4 and box 7.3)
# resample(A, n, r=1:2)
# OR resample based on number of plants surveyed
n <- c(4264, 30, 16, 24, 5)
## create a list with 1000 resampled matrices
x <- lapply(1:1000, function(x) resample(A, n, r=1:2))
mean(x)
## use var2 to check variances, especially if using different fvar values
var2(x)
## growth rates
y <- sapply(x, lambda)
quantile( y, c(0.025, .975) )
hist(y, br=30, col="palegreen", xlab="Lambda", main="1985 Hudsonia growth rates")
abline(v=quantile(y, c(0.025, .975)), lty=3)
## double the sample size (and quadruple seedlings) and you may be able to detect a decline
n <- n * 2
x <- lapply(1:1000, function(x) resample(A, n * 2, r=1:2))
quantile( sapply(x, lambda), c(0.025, .975) )
```

---

**secder**

*Second derivatives of the dominant eigenvalue*

**Description**

Calculates the second derivatives of the dominant eigenvalue of the demographic projection matrix for all non-zero transitions with respect to one specified transition.

**Usage**

```r
secder(A, k, l)
```

**Arguments**

- `A` **projection matrix**
- `k` **row index for the specified transition**
- `l` **column index for the specified transition**
Details

Function copied from demogR package after it was removed from CRAN. See section 9.7 in Caswell 2001.

Value

A square matrix of the same rank as A where each element $s_{ij}$ is the second derivative of the dominant eigenvalue of A.

Note

The eigenvalue second derivatives are essential for calculating both perturbation analyses of the eigenvalue elasticities and stochastic sensitivities.

Author(s)

James Holland Jones

References


See Also

eigen.analysis

Examples

```r
## eigenvalue second derivatives of the US projection matrix from 1967
## with respect to infant survival
x1 <- c(0, 0.0010478, 0.0820086, 0.2884376, 0.3777064,
  0.2647110, 0.1405144, 0.0585568, 0.0134388, 0.0003327)
x2 <- diag(c(0.9972036, 0.9983625, 0.9978063, 0.9967535,
  0.9961039, 0.9948677, 0.9923658, 0.9885968, 0.9828676))
usa <- rbind(x1, cbind(x2,0))
sd21 <- secder(usa,2,1)
sd21
```
sensitivity

Sensitivity analysis of a projection matrix

Description

Calculate the sensitivities of eigenvalues to changes in the projection matrix elements

Usage

sensitivity(A, zero = FALSE)

Arguments

A A projection matrix

zero Set sensitivities for unobserved transitions to zero, default is FALSE

Details

see section 9.1 in Caswell (2001)

Value

A sensitivity matrix

Author(s)

Chris Stubben

References


See Also

elasticity

Examples

sens <- sensitivity(teasel)
## IMAGE plot with smaller boxes
image2(sens, mar=c(1,3.5,5,1), box.offset=.1)
title("Sensitivity matrix using image2", line=2.5)
## MATPLOT
matplot2(sens, log='y', type='b', yaxt='n', ltitle="Fate",
ylab=expression(paste("Sensitivity of \lambda",sep="")),
main="Sensitivity matrix using matplot2")
pwrs <- -4:1
axis(2, 10^pwrs, parse(text=paste("10^", pwrs, sep = "")), las=1)
splitA

**Description**

Splits a projection matrix into transition and fertility matrices where \( A = T + F \)

**Usage**

\[
\text{splitA}(A, r = 1, c = -1)
\]

**Arguments**

- \( A \): a projection matrix
- \( r \): rows containing fertilities (default is first row) OR a logical matrix where TRUE is the location of a fertility value OR a complete fertility matrix
- \( c \): columns containing fertilities, default is all columns except first

**Details**

see section 5.1 in Caswell (2001)

**Value**

A list with T and F matrices

**Note**

By default, the fertility matrix will include elements in the first row (except first element). In some cases, it is not possible to split a projection matrix using only row and column indexes. Therefore, a logical matrix (where TRUE is the location of a fertility value) or the complete fertility matrix is also accepted.

**Author(s)**

Chris Stubben

**References**


**See Also**

functions like `generation.time` and `net.reproductive.rate` use `splitA` to split the matrix
stable.stage

stable.stage

### Description

Calculates the stable stage distribution of a projection matrix

### Usage

```r
stable.stage(A)
```

### Arguments

- `A`: A projection matrix

### Details

see section 4.5 in Caswell (2001)

### Value

A vector containing the stable stage distribution

### Author(s)

Chris Stubben

### References

Examples

```r
w <- stable.stage(teasel)
w
barplot(w, col="green", ylim=c(0,1), las=1, main="Teasel",
       ylab="Stable stage proportion", xlab="Stage class")
box()
```

stage.vector.plot

Plot stage vector projections

Description

Plots short-term dynamics and convergence to stage stage distribution using stage vector projections

Usage

```r
stage.vector.plot(stage.vectors, proportions = TRUE,
                  legend.coords = "topright", ylim = NULL, xlab = "Years",
                  ylab = NULL, col = rainbow(8), ...)
```

Arguments

- `stage.vectors`: a matrix listing stage class vectors in columns
- `proportions`: plot proportional changes or total numbers, defaults to proportions
- `legend.coords`: a legend keyword or vector of x,y coordinates, defaults to top-right corner
- `ylim`: the y limits of the plot, defaults to min and max values in stage.vectors
- `xlab`: a label for the x axis
- `ylab`: a label for the y axis
- `col`: vector of line colors, defaults to rainbow(8)
- `...`: additional options are passed to `plot` function

Details

see section 2.2 in Caswell 2001

Value

A plot of stage or age class projections

Author(s)

Chris Stubben

See Also

see `pop.projection`
Examples

```r
## matrix from Example 2.1 in Caswell
A <- matrix2(c(0, 0.3, 0, 1, 0, 0.5, 5, 0, 0), 1:3)
n <- c(1,0,0)
p <- pop.projection(A,n,60)
## Plots in Figure 2.3
stage.vector.plot(p$stage.vector[,1:15], col='black', las=1, prop=FALSE)
stage.vector.plot(p$stage.vector[,1:40], col=2:4, las=1)
## log-scale with custom y-axis
stage.vector.plot(p$stage.vector, col=2:4, prop=FALSE, ylim=c(.01, 10), log='y', legend="bottomright", yaxt='n')
pwrs <- -2:1
# major ticks
axis(2, at = 10^pwrs, labels=parse(text=paste("10^{" , pwrs, sep = "")), las=1, tcl= -.6)
# minor ticks
axis(2, at = 1:9 * rep(10^pwrs[-1] / 10, each = 9),
     tcl = -.3, labels = FALSE)
```

---

**stoch.growth.rate**  
*Log stochastic growth rate*

**Description**

Calculates the log stochastic growth rate by Tuljapukar’s approximation and by simulation.

**Usage**

```r
stoch.growth.rate(matrices, prob = NULL, maxt = 50000,
verbose = TRUE)
```

**Arguments**

- **matrices**: a list with two or more projection matrices, or a matrix with one projection matrix per column, with elements filled by columns.
- **prob**: a vector of probability weights used by `sample` for selecting the projection matrices, defaults to equal probabilities.
- **maxt**: number of time intervals, default 50000.
- **verbose**: Print comment at start of time 1, 10000, 20000, etc.

**Details**

converted Matlab code from Box 7.4 in Morris and Doak (2002)
Value

A list with 3 items

- approx: log stochastic growth rate by Tuljapukar’s approximation
- sim: log stochastic growth rate by simulation
- sim.CI: confidence interval for simulation

Author(s)

Chris Stubben

References


See Also

stoch.projection to output population sizes from simulation

Examples

```r
sgr <- stoch.growth.rate(hudsonia)
sgr
exp(sgr$approx)
```

---

**stoch.projection**

Project stochastic growth from a sequence of matrices

Description

Projects stochastic growth using whole matrix selection techniques in an independently and identically distributed (iid) environment from a set of two or more projection matrices

Usage

```r
stoch.projection(matrices, n0, tmax = 50, nreps = 5000, prob = NULL, nmax = NULL, sumweight = rep(1, length(n0)), verbose = FALSE)
```
Arguments

- matrices: a list with two or more projection matrices
- n0: initial population vector
- tmax: number of time steps or projection intervals to predict future population size
- nreps: number of iterations
- prob: a vector of probability weights used by sample for selecting the projection matrices; defaults to equal probabilities
- nmax: a maximum number of individuals beyond which population projections cannot exceed. Default is no density dependence
- sumweight: A vector of ones and zeros used to omit stage classes when checking density threshold. Default is to sum across all stage classes
- verbose: Print comments at start of iteration 1, 100, 200, 300, etc.

Details

converted Matlab code from Box 7.3 in Morris and Doak (2002) with nmax option added to introduce simple density dependence

Value

A matrix listing final population sizes by stage class with one iteration per row.

Author(s)

Chris Stubben

References


Examples

```r
n <- c(4264, 3,30,16,25,5)
names(n) <- c("seed", "seedlings", "tiny", "small", "medium", "large")
## use equal and unequal probabilities for matrix selection
x.eq <- stoch.projection(hudsonia, n, nreps=1000)
x.uneq <- stoch.projection(hudsonia, n, nreps=1000, prob=c(.2,.2,.2,.4))
hist(apply(x.eq, 1, sum), xlab="Final population size at t=50", ylab="", xaxt="n", yaxt="n", col="green", breaks=seq(0,5000, 100), main="Var")
par(new=TRUE)
## use transparency for overlapping distributions - may not work on all systems
hist(apply(x.uneq, 1, sum), xlab="Final population size at t=50", main="Var")
legend(2500,200, c("equal", "unequal"), cex.main=1)
```
Description

Estimate the quasi-extinction probability by simulation for a structured population in an independently and identically distributed stochastic environment.

Usage

```r
stoch.quasi.ext(matrices, n0, Nx, tmax = 50, maxruns = 10,
    nreps = 5000, prob = NULL, sumweight = NULL, verbose = TRUE)
```

Arguments

- `matrices`: a list with two or more projection matrices, or a matrix with one projection matrix per column, with elements filled by columns.
- `n0`: initial population vector.
- `Nx`: quasi-extinction threshold.
- `tmax`: number of time steps or projection intervals.
- `maxruns`: number of times to simulate cumulative distribution function.
- `nreps`: number of iterations.
- `prob`: a vector of probability weights used by `sample` for selecting the projection matrices.
- `sumweight`: A vector of ones and zeros used to omit stage classes when checking quasi-extinction threshold. Default is to sum across all stage classes.
- `verbose`: Print comment at start of run 1,2,3, etc.

Details

Converted Matlab code from Box 7.5 in Morris and Doak (2002).

Value

A matrix with quasi-extinction probabilities for each run by columns.

Author(s)

Chris Stubben
stoch.sens

References

See Also
stoch.projection

Examples
n <- c(4264, 3, 30, 16, 25, 5)
names(n) <- c("seed", "seedlings", "tiny", "small", "medium", "large")
## exclude seeds using sumweight. Using 100 nreps for speed
x <- stoch.quasi.ext(hudsonia, n, Nx=10, nreps=100, sumweight=c(0,1,1,1,1,1))
matplot(x, xlab="Years", ylab="Quasi-extinction probability", type='l', lty=1, col=rainbow(10), las=1,
main="Time to reach a quasi-extinction threshold of 10 above-ground individuals")

stoch.sens

Stochastic growth rate sensitivity

Description
Calculates the sensitivity of the stochastic growth rate to perturbations in the mean demographic projection matrix

Usage
stoch.sens(A, tlimit = 100)

Arguments
A a list of matrices
tlimit time limit, default 100

Details
Function copied from demogR package after it was removed from CRAN. See section 14.4.1 in Caswell 2001.

Value
A list with two elements:
sensitivities sensitivities of the stochastic growth rate
elasticities elasticities of the stochastic growth rate
Author(s)

James Holland Jones

References


See Also

eigen.analysis

Examples

stoch.sens(hudsonia)

---

stretchbetaval

**Stretched beta-distributed random numbers**

Description

Generate a stretched beta number with mean, standard deviation, minimum and maximum values and CDF value for bounded fertility estimates

Usage

stretchbetaval(mn, std, minb, maxb, fx)

Arguments

- **mn**: mean of a fertility rate
- **std**: standard deviation
- **minb**: minimum value
- **maxb**: maximum value
- **fx**: Cumulative Distribution Function value

Details

converted Matlab code from Box 8.5 in Morris and Doak (2002)
Value

Returns a stretched beta number with mean mn, standard deviation std, minimum and maximum values (minb, maxb) and CDF value fx.

Author(s)

Adapted to R by Patrick Nantel, 11 July 2005.

References


See Also

betaval

Examples

stretchbetaval(3, 1.2, 1, 20, runif(1))
# Generates stretchbeta random
# fertilities for a population of 1000 mature individuals (Ni) with mean
# fertility (f) of 3.0 and inter-individual variance in fertility (varF) of 1.5.
Ni <- 1000
f <- 2.5
varF <- 1
fmin <- 1
fmax <- 5
rndfert <- numeric(Ni)
for(i in 1:Ni) rndfert[i] <- stretchbetaval(f, sqrt(varF), fmin, fmax, runif(1))
hist(rndfert, 20, main="Stretched beta-distributed random fertilities",
     xlab="Fertility rate", col="blue")

Projection matrix for teasel

Description

Projection matrix with six stage classes for the plant teasel

Usage

teasel

Format

A 6 x 6 matrix
Source

Example 5.2 in Caswell 2001.

References


Examples

tea
image2(teasel, mar = c(1, 3.5, 5, 1), box.offset = .1)
title("Teasel projection matrix", line = 2.5)
# fertilities for a monocarpic plant in a prebreeding census in last column
splitA(teasel, r = 1:6, c = 6)
lambda(teasel)

---

test.census Census data for hypothetical plant

Description

Three years of census data for a hypothetical plant with three stage classes

Usage

  test.census

Format

A data frame with 41 census rows and 4 columns:

  plant  Plant id number
  year   Year of census
  stage  Stage class: seedling, vegetative, or reproductive
  fruits Total number of fruits

Examples

test.census
stages <- c("seedling", "vegetative", "reproductive")
## Cross-tabulate stage vectors and order rows by stage
sv <- table(test.census$stage, test.census$year)[stages, ]
sv
stage.vector.plot(sv)
## set xaxt="n" to avoid fractions of a year (2002.5)
stage.vector.plot(sv, prop = FALSE, xaxt = "n", las = 1)
## Convert census data to state-fate transition table using reshape
```
reshape(test.census, direction = "wide", idvar = "plant", timevar = "year")
```
## Convert census data to state-fate transition table using merge
```
trans <- subset(
    merge(test.census, test.census, by = "plant", sort = FALSE),
    year.x == year.y - 1
)
```
## Format column and row names
```
trans <- trans[, c(1:4, 6)]
colnames(trans)[2:5] <- c("year", "stage", "fruits", "fate")
rownames(trans) <- 1:nrow(trans)
```
## Order stage and fate columns
```
trans$stage <- ordered(trans$stage, levels = stages)
trans$fate <- ordered(trans$fate, levels = c(stages, "dead"))
```
## Select transitions for 2001-2002 and count offspring (seedlings)
```
trans01 <- subset(trans, year == 2001)
seedlings <- nrow(subset(test.census, year == 2002 & stage == "seedling"))
```
## Add individual fertilities using "anonymous reproduction" based on the
## proportional reproductive outputs of flowering plants and the total number
## of seedlings at the end of the projection interval
```
trans01$seedling <- trans01$fruits / sum(trans01$fruits) * seedlings
```
## Create transition frequency table and build T matrix
```
tf <- table(trans01$fate, trans01$stage)
```
## remove "dead" fate from matrix
```
T.mat <- prop.table(tf, 2)[-4,]
```
## Summarize stage-specific fertility rates and build F matrix
```
fert <- tapply(trans01$seedling, trans01$stage, mean)
F.mat <- T.mat * 0
F.mat[1,] <- fert
```
## The final projection matrix is just
```
T.mat + F.mat
```
## OR use projection matrix function
```
projection.matrix(trans01)
```
---

### tortoise  
**Description**

Projection matrices for the desert tortoise *Gopherus agassizii* with 4 different fertility estimates (low, medium low, medium high, and high)
Usage
tortoise

Format
A list of 4 matrices

Source

References

Examples
tortoise
A <- tortoise["med.high"]
# log color scale not needed
image2(A, mar = c(1, 3.5, 5, 1), log = FALSE, box.off = .1)
title("Tortoise projection matrix", line = 3)
splitA(A)
lambda(A)
sapply(tortoise, lambda)

---

var2

Variance matrix

Description
Calculates the variances from a list of matrices

Usage
var2(x)

Arguments
x A list of two or more matrices

Value
A matrix containing variances

Author(s)
Chris Stubben
varEst

Examples

var2(hudsonia)

---

**varEst**

*Estimate the variance of beta-binomial vital rates*

**Description**

Finds the best estimates of mean and environmental variance for beta-binomial vital rates using the approximation method of Akcakaya (2002)

**Usage**

```r
varEst(rates, weighted = 1)
```

**Arguments**

- `rates`: a matrix or dataframe with four columns: Rate identifier, Year, Total number of starting individuals, Number surviving (or growing)
- `weighted`: either 1 for weighted average demographic variance, or 0 for unweighted average, default is 1

**Value**

A matrix with 3 columns: (1) total observed variance, (2) estimate of variance due to demographic stochasticity, and (3) estimate of variance due to environmental stochasticity.

**Author(s)**


**References**


**See Also**

Kendall

**Examples**

```r
varEst(woodpecker)
```
Vitalsens

Vital rate sensitivities and elasticities

Description
Calculates deterministic sensitivities and elasticities of lambda to lower-level vital rates using partial derivatives

Usage
vitalsens(elements, vitalrates)

Arguments
elements An object of mode expression with all matrix elements represented by zeros or symbolic vital rates
vitalrates A list of vital rates with names matching expressions in elements above

Details
Vital rate sensitivities and elasticities are discussed in example 9.3 and 9.6 in Caswell (2001). Also see Chapter 9 and Box 9.1 for Matlab code in Morris and Doak (2002).

Value
A dataframe with vital rate estimates, sensitivities, and elasticities

Note
The element expressions should return the actual matrix element estimates after evaluating the variables using eval below.
A<-sapply(elements,eval,vitalrates,NULL)
In addition, these expressions should be arranged by rows so the following returns the projection matrix.
matrix(A,nrow=sqrt(length(elements)),byrow=TRUE)

Author(s)
Chris Stubben. Based on code posted by Simon Blomberg to R-help mailing list

References
Examples

```r
## emperor goose in Morris and Doak 2002.
goose.vr <- list( Ss0=0.1357, Ss1=0.8926, Sf2=0.6388, Sf3=0.8943)
goose.el <- expression( 0, 0, Sf2*Ss1,Sf3*Ss1, Ss0,0, 0, 0, 0, Ss1,0, 0, 0, Ss0, 0, Ss1)

## first plot effects of changing vital rates -- Figure 9.1
n <- length(goose.vr)
vrsen <- matrix(numeric(n*length(vr)), ncol=n, dimnames=list(vr, names(goose.vr)))
for (h in 1:n) {
goose.vr2 <- list( Ss0=0.1357, Ss1=0.8926, Sf2=0.6388, Sf3=0.8943)
for (i in 1:length(vr)) {
goose.vr2[[h]] <- vr[i]
A <- matrix(sapply(goose.el, eval,goose.vr2 , NULL), nrow=sqrt(length(goose.el)), byrow=TRUE)
vrsen[i,h] <- max(Re(eigen(A)$values))
}
}

matplot(rownames(vrsen), vrsen, type='l', lwd=2, las=1, ylab="Goose population growth", main="Effects of changing goose vital rates")

legend(.8, .4, vrn, lty=1:4, lwd=2, col=1:4, cex=1.2)

## then calculate sensitivities -- Table 9.1
x <- vitalsens(goose.el, goose.vr)
x

sum(x$elasticity)

barplot(t(x[,2:3]), beside=TRUE, legend=TRUE, las=1, xlab="Vital rate", main="Goose vital rate sensitivity and elasticity")

aline(h=0)

## Table 7 endangered lesser kestral in Hiraldo et al 1996
kest.vr <- list(b = 0.9321, co = 0.3847, ca = 0.925, so = 0.3409, sa = 0.7107)
kest.el <- expression( co*b*so, ca*b*so, sa, sa)
x <- vitalsens(kest.el, kest.vr)
x

sum(x$elasticity)

barplot(t(x[,2:3]), beside=TRUE, las=1, xlab="Vital rate", main="Kestral vital rate sensitivity and elasticity")

legend(1,1, rownames(t(x[,2:3])), fill=grey.colors(2))
aline(h=0)
```

**Description**

Calculates the extinction time CDF and stochastic growth rate by sampling vital rates from a beta, stretched beta, or lognormal distribution and includes within-year, auto- and cross-correlations

**Usage**

\[
vitalsim(vrmeans, vrvars, corrin, corrout, makemx, n0, yrspan, Ne = 500, 
\text{tmax} = 50, \text{runs} = 500, \text{vrtypes} = \text{NULL}, \text{vrmins} = \text{NULL}, 
\text{vrmaxs} = \text{NULL}, \text{sumweight} = \text{NULL})
\]

**Arguments**

- **vrmeans**: means of vital rates
- **vrvars**: variance of vital rates
- **corrin**: within year correlation
- **corrout**: between year correlations
- **makemx**: a function that creates a square projection matrix from a vector of \text{vrmeans}
- **n0**: initial population vector
- **yrspan**: the number of years of correlations to build into the M12 matrix
- **Ne**: quasi-extinction threshold
- **tmax**: latest time to calculate extinction probability, default 50
- **runs**: the number of trajectories, default is 500. 1000 is recommended
- **vrtypes**: identifies the distribution for each rate in \text{vrmeans} where 1 = beta, 2 = stretched beta, 3 = lognormal, default is all ones
- **vrmins**: minimum value for each vital rate; use zeros for rates that are not stretched betas, default is all zeros
- **vrmaxs**: maximum value for each vital rate; use zeros for rates that are not stretched betas, default is all zeros
- **sumweight**: a vector of weights, with 0 to omit a class and 1 to include it when computing the summed density to compare to the quasi-extinction threshold, default is to include all classes

**Details**

Vital rates used must be either fertility values or binomial probabilities, i.e., probabilities for events with only two possible outcomes (such as survival). Means and variances of the vital rates should preferably be corrected to remove sampling errors and demographic stochasticity. Note that this version of the function does not simulate demographic stochasticity and is density-independent.

**Value**

Plots a histogram of log stochastic growth rates and the cumulative probability of quasi-extinction and returns a list with 4 items:

- **detLambda**: the deterministic population growth rate computed from the mean matrix
stochlambda  the mean stochastic growth rate with 95% confidence intervals.
logLambdas    a vector of all log stochastic growth rates in first plot
CDFExt        a vector of cumulative probabilities of quasi-extinction in second plot

Note

The correlation matrices for Hudsonia in Morris and Doak 2002 include some correlations > 1. A corrected set of correlations was sent by D. Doak on 8/4/2007. Therefore the results from the simulation below are different than the book.

Author(s)

Original MATLAB code from Box 8.10 in Morris and Doak (2002). Adapted to R by Patrick Nantel, 12 July 2005

See Also

hudmxdef, hudvrs and hucorrs

Examples

```r
## set vrtypes
hudvrtypes <- c(rep(1,13), rep(3,5), rep(1,6))
## run Full model- using 100 runs here for speed
full <- vitalsim(hudvrs$mean, hudvrs$var, hudcorrs$corrin, 
hudcorrs$corrout, hudmxdef, vrtypes=hudvrtypes, 
n0=c(4264,3,30,16,25,5), yrspan=20 , runs=100)
## deterministic and stochastic lambda
full[1:2]
## log stochastic lambda
log(full$stochLambda)
sd(full$logLambdas)
## SKIP the next two simulations- however, sample output is included for plotting
#NO between year correlations so corrout = diag(0,13) - all zeros
# no.between <- vitalsim(hudvrs$mean, hudvrs$var, hudcorrs$corrin, 
# diag(0,13), hudmxdef, vrtypes=hudvrtypes,
# n0=c(4264,3,30,16,25,5), yrspan=20 )
no.between <- list(CDFExt=c(rep(0,40),0.01,0.04,0.12,0.15, 
0.20,0.31,0.49,0.58,0.72,0.78))
#NO correlations so corrout = diag(0,13) AND corrin=diag(13) - ones on diagonal
# no.corr<-vitalsim(hudvrs$mean, hudvrs$var, diag(13),
# diag(0,13), hudmxdef, vrtypes=hudvrtypes,
# n0=c(4264,3,30,16,25,5), yrspan=20 )
no.corr <- list(CDFExt=c(rep(0,39),0.03,0.03,0.06,0.12,0.20, 
0.30,0.42,0.52,0.65,0.76,0.83))
## Figure 8.3 with corrected correlation matrices for full model
matplot(cbind(a=full$CDFExt, no.between$CDFExt, no.corr$CDFExt), type='l',
ylim=c(0,1), lty=1:3, col=2:4, lwd=2, las=1,
 xlab="Years into the future", ylab="Cumulative probability of quasi-extinction")
legend(2,1, c("Full model", "No between-year correlations", "No correlations"),
 lty=1:3, col=2:4, lwd=2)
```
Projection matrix for killer whale

**Description**

Projection matrix for killer whales with 4 size classes: yearling, juvenile, mature and post-reproductive.

**Usage**

`whale`  

**Format**

A 4 x 4 matrix

**Source**

Example 5.1 in Caswell (2001)

**References**


**Examples**

```r
whale
splitA(whale)
lambda(whale)
sensitivity(whale)
# plot sensitivity
matplot2(sensitivity(whale),
    type = "b", legend = "topleft", ltitle = "Fate",
    main = "Killer Whale sensitivity")
```

Survivorship data for adult and juvenile Acorn Woodpeckers

**Description**

Number of juvenile and adult Acorn Woodpeckers and survival in the Water Canyon, New Mexico population, reconstructed from Stacey and Taper (1992).

**Usage**

`woodpecker`
Format

A data frame with 18 rows and 4 columns

rate  Adult or juvenile stage
year  Year
start  Total number of starting individuals
surv  Number surviving to spring

Source


References


See Also

Kendall and varEst

Examples

woodpecker
x <- subset(woodpecker, rate == "adult")
plot(x$year, x$start,
    type = "o", pch = 16,
    ylab = "Number of adults", xlab = "Year",
    main = "Acorn Woodpeckers in Water Canyon"
)
## stage-specific survival rate
x <- aggregate(
    list(Nstart = woodpecker$start, Nsurv = woodpecker$surv),
    list(stage = woodpecker$rate), sum
)
 x$survival <- x[, 3] / x[, 2]
 x
Index

*Topic datasets
aq.census, 6
aq.trans, 8
calathea, 13
grizzly, 22
hudcorrs, 25
hudsonia, 26
hudvrs, 27
monkeyflower, 40
nematode, 43
teasel, 65
test.census, 66
tortoise, 67
whale, 74
woodpecker, 74

*Topic documentation
01.Introduction, 3
02.Caswell, 3
03.Morris, 5
01.Introduction, 3
02.Caswell, 3, 5
03.Morris, 3, 5
aq.census, 6, 9
aq.matrix, 7
aq.trans, 6, 8
betaval, 5, 9, 65
boot.transitions, 4, 11, 52
calathea, 13
Caswell (02.Caswell), 3
countCDFxt, 5, 14, 20
cut, 28
damping.ratio, 4, 15
eigen, 17, 31, 32
eigen.analysis, 4, 16, 45, 54, 64
elasticity, 4, 18, 55
eval, 70
expression, 70
extCDF, 5, 15, 19
fundamental.matrix, 4, 20, 22, 44
generation.time, 4, 21, 24, 44, 56
grizzly, 5, 22
head, 24
head2, 24
hudcorrs, 25, 73
hudmxdef, 25, 73
hudsonia, 26
hudvrs, 26, 27, 73
image, 29
image2, 28
Kendall, 5, 29, 69, 75
lambda, 4, 16, 31
legend, 37, 58
levels, 48
list, 59, 61, 62
lnorms, 5, 32, 52
logi.hist.plot, 33
LTRE, 4, 34
matplot, 38
matplot2, 37
matrix, 39
matrix2, 38
mean, 39
mean.list, 39
names, 70
nematode, 43
net.reproductive.rate, 4, 21, 22, 44, 56

ordered, 7, 48

pfister.plot, 45
plot, 37, 58
pop.projection, 3, 17, 32, 46, 58
popbio(01.Introduction), 3
projection.matrix, 4, 7, 8, 11, 48

QPmat, 4, 44, 50

rbeta, 10
reproductive.value, 4, 51
resample, 4, 52
rlnorm, 32
rowMeans, 39

sample, 59, 61, 62
secder, 4, 53
sensitivity, 4, 18, 55
splitA, 20, 21, 44, 52, 56
stable.stage, 4, 57
stage.vector.plot, 38, 47, 58
stoch.growth.rate, 4, 5, 59
stoch.projection, 5, 60, 60, 63
stoch.quasi.ext, 5, 62
stoch.sens, 4, 63
stretchbeta.val, 5, 33, 64

tail, 24
teaasel, 65
test.census, 66
tortoise, 67

unlist, 39

var2, 40, 68
varEst, 31, 69, 75
vitalsens, 5, 70
vitalsim, 5, 10, 25–27, 71

whale, 5, 74
woodpecker, 74