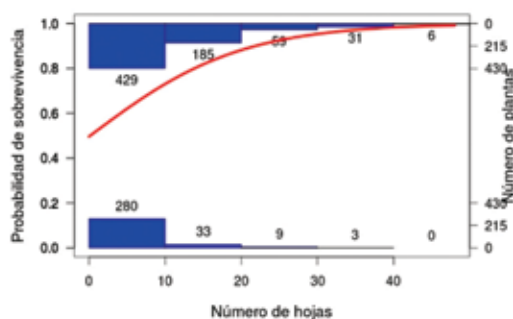
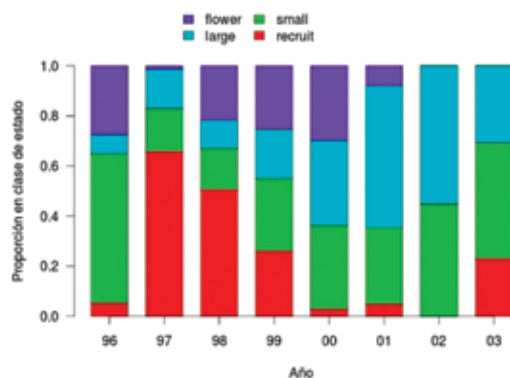
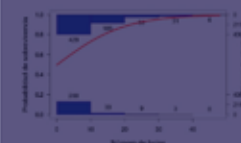
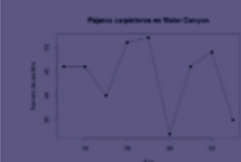
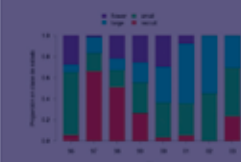
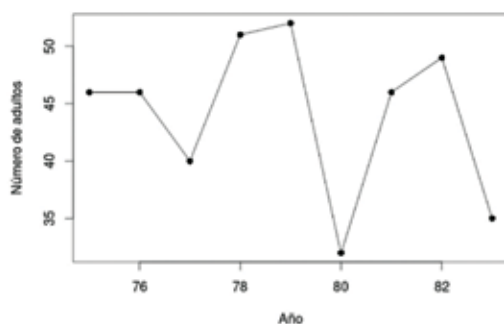


Introducción a la Ecología de poblaciones con el lenguaje R y el paquete popbio



Pájaros carpinteros en Water Canyon



Casa abierta al tiempo

UNIVERSIDAD AUTÓNOMA METROPOLITANA
UNIDAD XOCHIMILCO División de Ciencias Biológicas y de la Salud

José A. Arévalo R.
Christopher Stubben
Brook Milligan
María del C. Mandujano S.
Jordan K. Golubov F.





RECTOR GENERAL

Dr. Salvador Vega y León

SECRETARIO GENERAL

Mtro. Norberto Manjarrez Álvarez

UNIVERSIDAD AUTÓNOMA METROPOLITANA-XOCHIMILCO

RECTORA

Dra. Patricia E. Alfaro Moctezuma

SECRETARIO

Lic. G. Joaquín Jiménez Mercado

DIVISIÓN DE CIENCIAS BIOLÓGICAS Y DE LA SALUD

DIRECTOR

Mtro. Rafael Díaz García

SECRETARIA ACADÉMICA

Dra. Leonor Sánchez Pérez

RESPONSABLE DEL PROGRAMA EDITORIAL

Lic. Zyanya Patricia Ruiz Chapoy

COMITÉ EDITORIAL

Esp. Marco Antonio Díaz Franco

Dra. Norma Ramos Ibáñez

Mtro. Alejandro Meléndez Herrada

Dr. Jordan Golubov Figueroa

M. en C. Dorys Primavera Orea Coria

Dr. Román Espinosa Cervántes

Dra. María Angélica Gutiérrez Nava

Dr. Ernesto Sánchez Mendoza

“INTRODUCCIÓN A LA ECOLOGÍA DE POBLACIONES CON EL LENGUAJE R Y EL PAQUETE POPBIO”

Primera edición: 2015

ISBN: 978-607-28-0492-0

D.R. © UNIVERSIDAD AUTÓNOMA METROPOLITANA

Unidad Xochimilco

Calzada Del Hueso 1100, Col. Villa Quietud, Del. Coyoacán, C.P. 04960, México, D.F.,

Tel.: 5483 7000 ext. 3783.

Hecho en México

Introducción a la Ecología de poblaciones con el lenguaje R y el paquete Popbio

División de Ciencias Biológicas

José A. Arévalo R.¹, Christopher Stubben², Brook Milligan³,
María del C. Mandujano S.⁴, Jordan K. Golubov F.¹

¹Lab. Ecología y Sistemática Vegetal, Depto. El Hombre y su Ambiente, CBS, Universidad Autónoma Metropolitana
Unidad Xochimilco, México

²División de Biociencias, Los Alamos National Laboratory, USA.

³Departamento de Biología, New Mexico State University, USA.

⁴Departamento de Ecología de la Biodiversidad, Instituto de Ecología, Universidad Nacional Autónoma de México.



UNIVERSIDAD AUTÓNOMA METROPOLITANA

Índice

1. Introducción y conceptos sobre R	1
Introducción	2
 2. Paquete popbio	 41
popbio	42
Caswell	46
Morris	48
aq.census	49
aq.matrix	51
aq.trans	53
betaval	55
boot.transitions	57
calathea	60
colorguide	62
countCDFxt	64
damping.ratio	66
eigen.analysis	68
elasticity	71
extCDF	73
fundamental.matrix	75
generation.time	77
grizzly	79
head2	81
hudcorrs	82
hudmxdef	84
hudsonia	86
hudvrs	87
image2	88
Kendall	91
lambda	94
lnorms	96
logi.hist.plot	98
LTRE	100
matplot2	104
mean.list	106
monkeyflower	108
multiresultm	110
nematode	112
net.reproductive.rate	113
pfister.plot	115
pop.projection	117

projection.matrix	119
QPmat	123
reproductive.value	125
resample	126
sensitivity	129
splitA	131
stable.stage	133
stage.vector.plot	134
stoch.growth.rate	136
stoch.projection	138
stoch.quasi.ext	140
stretchbetaval	142
teasel	144
test.census	145
tortoise	147
var2	148
varEst	149
vitalsens	151
vitalsim	154
whale	158
woodpecker	159

Índice de figuras

1. Página Web principal del proyecto R	7
2. Página Web de R para descargar archivos de instalación de Linux, Windows y MacOS X	7
3. Instalación de R utilizando Synaptic package manager	9
4. Cita del paquete R	11
5. Gráficas generadas utilizando la función plot()	28
6. Gráfica generada utilizando la función plot(x,y)	29
7. Gráfica de la especie Nisp1 utilizando la función hist(x)	29
8. Gráfica generada utilizando la función plot(x,y) con mas de dos variables y leyenda	30
9. Gráfica de puntos y líneas utilizando la función plot(x,y) con mas de dos variables y leyenda	31
10. Boxplot o gráfica de caja	32

Índice de cuadros

1. Funciones algebraicas en R	16
2. Principales funciones específicas para los vectores en R	24
3. Funciones de estadística descriptiva más utilizadas de R	25
4. Algunas funciones de las salidas de los archivos gráficos	34

1. Introducción y conceptos sobre R

"When you make the finding yourself
- even if you're the last person on
Earth to see the light
- you'll never forget it"

Carl Sagan

Introducción

El presente trabajo surge de la necesidad de contar con un manual del paquete **popbio** para usuarios de **R** en habla hispana. Es así como en coautoría con C. Stubben y B. Milligan, autores de la versión en Inglés, desarrollamos este volumen, que tiene como primer objetivo hacer accesibles las herramientas que brinda **popbio** para el estudio de la ecología de poblaciones¹. El manual de usuario del paquete **popbio** en lenguaje **R**, tiene como propósito que el usuario entienda la forma de organizar la información (datos) obtenidos normalmente durante los muestreos demográficos de plantas o animales para ser analizados. El paquete **popbio** consta de una serie de rutinas que son fácilmente ajustables a bases de datos de cualquier especie para estimar atributos y dinámicas poblacionales, así como comparar las tasas vitales entre sitios, años o especies, utilizando modelos matriciales de poblacionales. Cuando se tienen datos poblacionales, usualmente el problema al analizarlos radica en la organización de las bases de datos para poder manipularlas en **R** o en cualquier otro programa u hoja de cálculo. El paquete **popbio** como muchos de los que están en la página de archivos de **R CRAN** (por sus siglas en inglés, Comprehensive R Archive Network), está diseñado para un usuario con conocimientos básicos del funcionamiento de **R**, por lo tanto éste manual tiene la función de ser una introducción al lenguaje **R** y aplicar los conocimientos al utilizar el paquete **popbio**.

No cabe duda que las aplicaciones de este programa son muy diversas puesto que los comandos que se usan pueden ser utilizados en un sinnúmero de estudios. Las instrucciones generales del uso de **R** las encontramos en muchos documentos accesibles de manera gratuita en la red especialmente en **CRAN**. También existe una serie de libros extraordinarios, como la serie *USE R!* de la editorial Springer <http://www.springer.com/series/6991>, que sin duda son una excelente guía para el uso de **R**. Entre ellos hemos encontrado particularmente útiles los libros de Zuur *et al.* (2009) y el de Crawley (2013). El primero es muy claro para principiantes y el segundo es más complicado por lo que se recomienda para usuarios más avanzados.

Este manual no es un texto de estadística, ni un libro para aprender **R**, es un manual que trata de ser una primera referencia para iniciar el uso de **R** y aplicar todos los comandos de **popbio**. El documento se centra en las herramientas para

1 El manual original de **popbio** se traduce al español y la aplicación del lenguaje **R** es desarrollado por José A. Arévalo R., María del C. Mandujano S. y Jordan K. Golubov F.

estructuración de una base de datos para utilizar el programa **popbio**, por lo que esperamos cumplir con el objetivo.

¿Por qué **popbio**?

Los ecólogos de poblaciones hemos tenido que realizar estudios con computadoras y diversos paquetes estadísticos. Inicialmente muchos análisis requerían un tiempo relativamente largo, el conocimiento de múltiples programas, pocas veces accesibles. Posteriormente, los problemas de muestreo en ecología de poblaciones se han complejizado, ya que necesariamente un conjunto de datos o una matriz tiene que muestrearse repetidas veces con el objetivo de determinar los errores de los estimadores poblacionales. A la luz de dicha problemática, se empezó a utilizar el lenguaje C como respuesta a mejorar el análisis de datos, sin embargo la falta de biólogos preparados en este lenguaje limitó sus aplicaciones y usos. Más aún la dificultad de preparar a ecólogos poblacionales con un recurso limitado de tiempo y profesores, nos hizo buscar nuevas y mejores herramientas. Es entonces que a raíz de la experiencia doctoral de uno de los autores, C. Stubben, en el laboratorio de B. Milligan se llegó a la conclusión de que R proveía la plataforma necesaria para realizar distintos análisis en términos de ecología de poblaciones. Pocos años después se generó el paquete **popbio** en CRAN (Comprehensive R Archive Network) que daba la posibilidad de explotar todas las herramientas estadísticas que se encuentran en R y aplicarlas al análisis de las poblaciones. Por ejemplo, ya se podían generar los modelos de sobrevivencia, obtener los parámetros e incluirlos dentro de los modelos matriciales de población, manipular los datos de entrada con facilidad y automatizarlo todo dentro de un mismo ambiente y totalmente accesible mediante la licencia de libre acceso, Free and open-source software (FOSS). No sólo esto, sino que cada usuario ahora también tiene la posibilidad de manipular el código para ajustarlo a sus requerimientos particulares o añadir funciones a las ya existentes.

Recientemente se desarrolló la paquetería (IPMPack, <http://ipmpack.r-forge.r-project.org/>) para la siguiente generación de modelos integrales de población, en el lenguaje de R, creando así un ambiente común para un número más grande de posibles modelos. De allí nació la idea de generar documentos de la paquetería de R en español. La paquetería original de **popbio** se puede encontrar en CRAN y los autores hicimos un esfuerzo por traducirlos lo mejor posible al español. Sin embargo, la gente de habla hispana seguirá teniendo que utilizar los comandos internos de R, en inglés, por el simple hecho de que la paquetería se desarrolló en este idioma. Tanto R como **popbio** están en evolución continua y es especialmente importante la contribución de los usuarios que usan los paquetes. Todos los ejemplos han sido ejecutados en diversos sistemas y computadoras, y en todos los casos han funcionado. Creemos que el valor de los paquetes dependen en gran medida de los comentarios y la experiencia del usuario el cual enriquece tanto la discusión teórica como las posibles aplicaciones.

¿Qué es R?

Como en el caso de muchos biólogos, la primera aproximación a R inspira temor y desconfianza, la segunda, genera frustración. Por ello, este manual brinda las herramientas para despejar dichas dudas y eliminar la desconfianza a usar el lenguaje R con todas sus bondades y retos. Usar R, necesariamente conlleva una curva de aprendizaje inicial pronunciada. Básicamente estamos hablando de muchas horas frente a una pantalla en la interfase de R probando como hacer uno u otro análisis o en algunos casos sintiendo frustración al obtener respuesta de R con mensajes de error que no tienen mucho sentido inicialmente. En muchas ocasiones es simplemente un error tipográfico, otras veces son errores un poco más complicados, los cuales no sabemos como solucionar y se reducen a nuestra falta de conocimientos teóricos o se solucionan fácilmente si consultamos un buen libro de estadística. No debemos sentirnos frustrados ya que existe mucha ayuda disponible de un número cada vez más grande de usuarios de R (foros, blogs, artículos, manuales o libros disponibles en línea) y basta una búsqueda en internet para encontrar la solución. En conclusión, creemos que los beneficios que se obtendrán al aprender R son mucho mayores al tiempo invertido. R es una plataforma para entender lo que queremos hacer, sin ser una caja negra que genera valores o información con poco sentido para el usuario como muchos programas estadísticos actuales. Por lo tanto, R es un lenguaje de programación en el que podemos realizar cálculos estadísticos básicos y sentar las bases para un análisis más complejo. El programa R se puede encontrar con cualquier buscador utilizando nombres como: **R project** o **R statistical software** siendo la página principal: <http://www.r-project.org/>

Al realizar cualquier trabajo o actividad siempre nos preguntamos, en algún momento, si la herramienta usada es la apropiada, es la mejor para el trabajo y/o es la más eficiente. Esta disyuntiva se presenta también al utilizar un paquete estadístico para analizar nuestros datos ya que existe la posibilidad de que los paquetes estadísticos puedan tener algún efecto en nuestros razonamientos e inferencias. Por lo tanto, la búsqueda, prueba y rechazo de herramientas estadísticas es un proceso normal en el quehacer científico. Los autores de este manual concluimos que bajo nuestra experiencia el lenguaje de programación R y sus características nos ayudan mantener un pensamiento organizado y estructurado con un objetivo bien definido lo cual nos hace más grato y divertido el análisis e interpretación de la información y que a su vez nos permite descubrir, cuestionar crear conocimiento sobre las dinámicas y los procesos ecológicos de nuestro interés.

R es un lenguaje de programación diseñado para realizar manipulación de datos, cálculos y desplegar gráficos. Tiene la capacidad de manejar de forma efectiva y fácil información de diferentes formatos, por ejemplo podemos tener en una misma estructura datos numéricos, booleanos y caracteres. Se pueden realizar cálculos con 'arrays' (arreglos sistemático de objetos en una lista o columna) lo cual es muy útil para trabajar con matrices. Las características gráficas que presenta son útiles para el análisis y se pueden observar en la computadora o imprimir. Además, las funciones que tiene se encuentran escritas en el lenguaje de programación lla-

mado 'S', diseñado durante la década de 1980, el cual ha sido altamente utilizado entre la comunidad científica y tiene muchas funciones y facilidades².

Preguntas básicas sobre el uso de R

El nuevo usuario de R seguramente se hace la pregunta del ¿por qué usar R? Como usuarios de R y lectores de varios manuales podemos responder en forma general, que el uso de R nos abre la posibilidad de aprender los procedimientos estadísticos utilizados, es decir, nos ayuda a entender y apreciar el tipo de análisis que queremos hacer y el tipo de resultados que estamos esperando. Esto es una gran ventaja sobre algunos paquetes comerciales los cuales tienen como salidas, después de realizar una prueba estadística, una gran cantidad de información la cual no ocupamos o no conocemos su significado. Los usuarios expertos pueden argumentar que precisamente eso es lo que se espera de un paquete, sin embargo, como nuevos usuarios y expertos en otros campos del conocimiento es deseable conocer, analizar y entender la información generada del muestreo o en el laboratorio de forma precisa y clara.

Una segunda razón por la cual es importante usar R, se encuentra relacionada con la ventaja de utilizar un software libre. Sabemos que tanto el presupuesto de un estudiante de licenciatura o posgrado y el de los proyectos de investigación es limitado. Por lo tanto el uso de software libre constituye una ventaja económica. Actualmente el software comercial exige el pago de una licencia, anual, el cual generalmente sólo se puede instalar en una sola computadora y no se puede utilizar en forma remota a menos que se encuentre disponible en un servidor dentro de la universidad o un centro de investigación. Por lo tanto el software libre es una opción para los estudiantes, profesores e investigadores. Además, puede ser instalado en las máquinas personales o de los laboratorios y ser utilizado en cualquier momento por el usuario sin ninguna restricción de licencia. Otra ventaja del software libre, como en el caso de R, es la existencia de una gran comunidad que se encuentra desarrollando aplicaciones para resolver los problemas dentro de sus investigaciones y éstas son evaluadas, probadas y aplicadas por diferentes usuarios. Así mismo, tenemos la posibilidad de generar aplicaciones propias para resolver algún problema específico al cual nos enfrentamos dentro de una investigación, opciones como ésta quedan fuera de la posibilidad de los desarrolladores de software de las compañías comerciales. La tercera razón y la más importante para usar R es nuestro propio deseo de aprender y la posibilidad de participar en la generación de conocimiento el cual podrá ser consultado, utilizado, evaluado y modificado por una gran comunidad de usuarios para satisfacer necesidades específicas del análisis de datos.

2 Para mayor información sobre la historia de S y R recomendamos consultar Hornik (2013).

El lenguaje **R** presenta algunas desventajas frente a sus competidores comerciales. La primera es el hecho que **R** no tiene una interfase gráfica para el usuario, sin embargo esta aparente carencia, puede aprovecharse ya que al ser eliminados los distractores para el usuario aumenta la estabilidad del software al no utilizar las localidades de memoria en recursos gráficos decorativos, que pueden generar conflictos de funcionamiento y pérdida de tiempo e información generada. Aunque es importante aclarar que existen interfases como **Rcmdr** y **R studio** con características similares a los paquetes comerciales donde es posible ejecutar los comandos de **R** utilizando mayores recursos gráficos. La imposibilidad de realizar un copiado y pegado, “copypaste”, como se realiza de manera rutinaria en el uso de los procesadores de texto. En el caso de los usuarios Linux, Windows y Mac OS X (Unix) con la interfase **R studio** y L^AT_EX existe la opción, `sweave()`³, la cual puede ser utilizada para incorporar texto y los resultados de alta calidad a un artículo, reporte o libro.

El uso de **Rcmdr** y de **R studio** puede tener varias ventajas para los usuarios que inician el aprendizaje del uso de **R** por las características de la interfase. El uso de **R** con la interfase **Rcmdr** o de **R studio** es recomendado para todos los usuarios en general, pero sugerimos que los principiantes utilicen la interfase de comandos de **R** en la consola.

Podemos resumir de forma general las razones en:

- Es un software de dominio público el cual junto con **S** es el software estándar utilizado por los profesionales dedicados a su desarrollo (R core group).
- Es comparable y/o superior a cualquier software comercial (SAS, SPSS, entre otros).
- Disponible para ambiente Windows, Mac OS X y Linux.
- Al ser un lenguaje de programación se pueden automatizar las tareas de análisis, además de crear nuevas funciones para resolver las necesidades y problemas específicos a las que se enfrenta cada usuario.
- La ayuda que se puede tener es amplia ya que el grupo de usuarios de **R** se encuentra formado por una gran cantidad de usuarios de diferentes disciplinas científicas.

Instalación de R

R se encuentra disponible en la página web <http://www.r-project.org/>, ver fig. 1. Recomendamos la exploración de la página web y se invierta todo el tiempo posible revisando los diferentes archivos, pantallas y manuales disponibles en el sitio. Esto podrá ser de gran utilidad al usuario ya que se presenta mucha información la cual es muy útil para entender la estructura lógica de **R**.

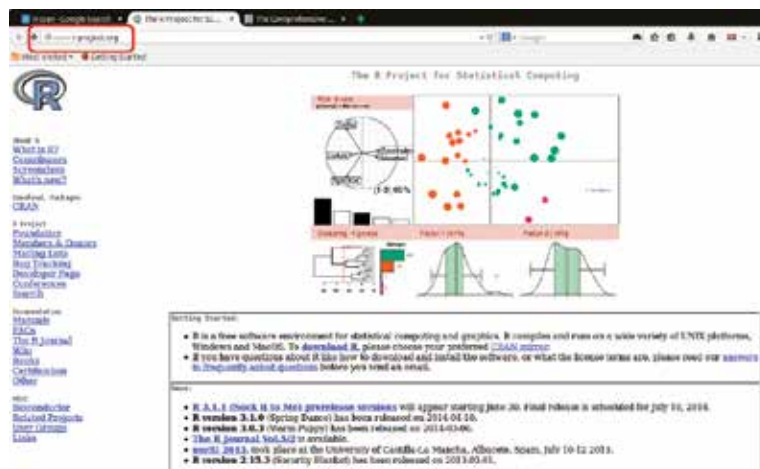
3 La función `sweave()` no será tratada en éste manual, recomendamos consultar Leisch y R-core (2014), Rudloff (2005), Morales (S/F).

En la figura 2 se presentan las diferentes opciones para descargar e instalar R para los diferentes sistemas operativos, Linux, Windows y MacOS X. Seleccione la opción apropiada para su sistema operativo y siga las instrucciones de la página web.

WINDOWS

La instalación de R en el sistema operativo Windows se puede realizar descargando el archivo **R-3.1.0-win.exe** que es la última versión disponible (Abril, 2014). Recomendamos se invierta un poco de tiempo en la lectura de las intrucciones, recomendaciones y características de la intalación de R para las diferentes versiones deWindows. Información sobre la instalación se encuentra en <http://cran.r-project.org/bin/windows/base/rw-FAQ.html>

Figura 1



Página Web del proyecto R, <http://www.r-project.org/>

Figura 2



PáginaWeb de R con las diferentes opciones de descarga de los archivos de instalación para los sistemas operativos Linux, Windows y MacOS X

MACOS X

La instalación en el sistema operativo MacOS X se puede realizar desde la página web presentada en la figura 2. Lea las instrucciones y recomendaciones que se presentan en la página web. Se tiene que seleccionar el archivo binario precompilado para su sistema operativo, no es recomendable la descarga del código fuente de R. Una excelente ayuda para la instalación la puede encontrar en <http://cran.r-project.org/bin/macosx/RMacOSX-FAQ.html>

LINUX

La siguiente sección describe de manera general la instalación de R en linux, en este ejemplo se utilizó Linux Ubuntu-Kubuntu ver. 14.04 LTS.

¿Cómo saber se tiene instalado R?

Abra la consola del “shell” y escribe después del prompt “R” (sin comillas), si aparece el mensaje:

```
desktop:~$ R
The program 'R' is currently not installed. You can install it by typing:
sudo apt-get install r-base-core
Make sure you have the 'universe' component enabled
bash: R: command not found
```

no se encuentra instalado R y se tiene que instalar, escriba en la misma consola, el comando:

```
sudo apt-get install r-base-core
```

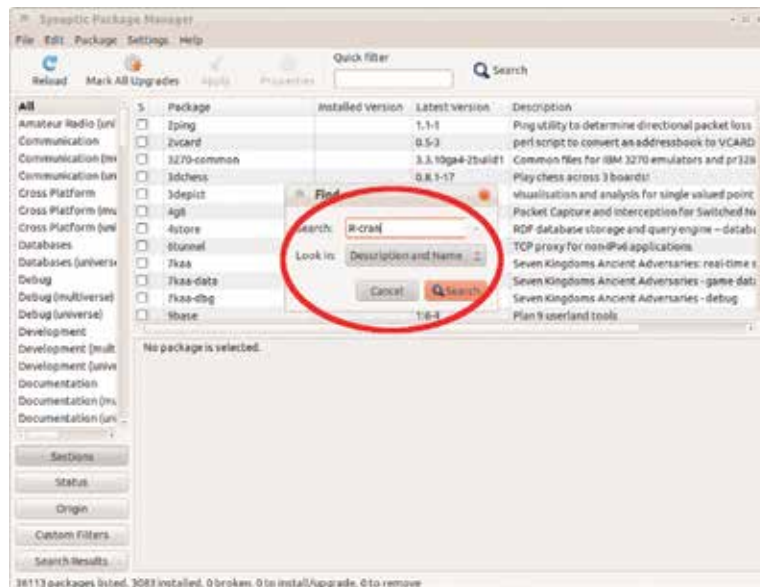
La instalación de esta forma requiere de los privilegios de super usuario o root, sólo éste puede instalar paquetes en Linux. Al terminar la instalación tiene que correr el comando para actualizar su sistema y las dependencias de los archivos.

```
sudo apt-get update && apt-get upgrade
```

Otra forma es el utilizar el programa de instalación de software Synaptic package manager (ver fig. 3), al igual se requieren los privilegios de super usuario para instalar programas. Se realiza la búsqueda del paquete escribiendo, **r-cran** e instalar **r-base-core**, con esto se tiene el sistema y con algunas de las funciones básicas

para iniciar el aprendizaje del uso de R. Si requiere de algún otro programa o rutinas para realizar un tipo específico de análisis se puede utilizar éste programa o se puede utilizar la consola de R, el ejemplo se encuentra donde se explica la instalación de **popbio** [pag. 30].

Figura 3:



Instalación de R utilizando Synaptic package manager

Es recomendable leer todas y cada una de las opciones que se despliegan en la pantalla para asegurarse de que se realiza una instalación adecuada. Al terminar la instalación de R se prueba la instalación del programa escribiendo:

```
desktop@computer:~$ R
```

desde su consola de comandos en línea. Se desplegará el siguiente mensaje:

```
R version 3.1.0 (2014-04-10) -- "Spring Dance"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: i686-pc-linux-gnu (32-bit)
```

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
```

```
Natural language support but running in an English locale
R is a collaborative project with many contributors.
```

```
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publica-
tions.
Type 'demo()' for some demos, 'help()' for on-line help,
or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
>
```

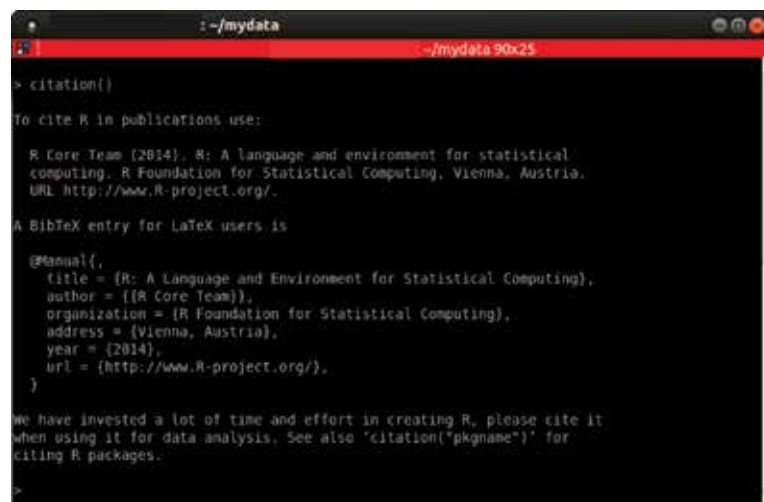
Si se despliega el mensaje anterior ya se tiene instalado el paquete R y se puede iniciar la sesión. En la ventana se despliegan algunas de las funciones que pueden ser de utilidad, es recomendable leer éstas y memorizarlas o apuntarlas en algún lugar si se inicia por primera vez la sesión.

Comandos a recordar:

- El comando `help()` ayuda en línea de las funciones. Por ejemplo, probar `help(mean)` o `?mean` nos dan información sobre el comando de la media aritmética. Al terminar la consulta preseione “q” (sin comillas).
- Si se escribe `help.start()`, el comando se ejecuta en el navegador y despliega la página principal del manual en la red.
- La función `citation()` despliega la forma de citar a R y/o los paquetes en una publicación, ver fig. 4.
- La función `demo()` despliega los diferentes ejemplos que han sido instalados y prueba el sistema. Son algunos ejemplo del uso de las funciones más generales. Se recomienda utilizar:

- `demo(graphics)` para demostración gráfica
- `demo(persp)` para graficos en perepectiva (3D)
- `demo(lm.glm)` para un ejemplo de modelo lineal
- `demo(plotmath)` despliega una tabla de notación matemática

Figura 4:



```
R Core Team (2014). R: A language and environment for statistical
computing. R Foundation for Statistical Computing, Vienna, Austria.
URL http://www.R-project.org/.

A BibTeX entry for LaTeX users is

@Manual{,
  title = {R: A Language and Environment for Statistical Computing},
  author = {{R Core Team}},
  organization = {R Foundation for Statistical Computing},
  address = {Vienna, Austria},
  year = {2014},
  url = {http://www.R-project.org/},
}

We have invested a lot of time and effort in creating R, please cite it
when using it for data analysis. See also 'citation('pkgname')' for
citing R packages.
```

Cita del paquete R, utilizando la función `citation()`, en el formato BibTeX para T_EX y L^AT_EX.

- Use `q()` o `Ctrl d` si se usa Linux, y reinicie la sesión de **R**, recuerde en contestar a la pregunta, `Save workspace image? [y/n/c]`: **R** guarda en un archivo `.Rhistory` los comandos de la sesión.
- Se puede utilizar las opciones `q("yes")` o `q("no")`

Primeros comandos

La primera actividad en una sesión de **R** es definir el directorio de trabajo. Es recomendable crear un subdirectorio (carpeta) de trabajo previamente para las bases de datos. Es importante la definición del directorio ya que se puede tener un seguimiento histórico de las diferentes actividades y análisis realizados a nuestra base de datos. La declaración de directorio de trabajo se hace mediante un comando de **R** o función `setwd()`. Los comandos de **R** tienen dos elementos: el nombre de la función y los argumentos de la función entre paréntesis `()`. Ya se han probado algunos de los comandos los cuales presentan o no argumentos de la función, `demo(graphics)` y `help.start()`, respectivamente.

Después de creado el subdirectorio de trabajo, se inicia la sesión de **R** en el mismo subdirectorio. El subdirectorio de trabajo contiene la(s) base(s) de datos, previamente organizada en columnas. Los comandos que se utilizan son `setwd()` para declarar al directorio de trabajo y `getwd()` para consultar el subdirectorio.

1. Iniciar la sesión de **R** en el subdirectorio creado para la(s) base(s) de datos.
2. Utilizar el comando: `setwd(dir="/path/")`, ejemplo:
`setwd(dir="~/Datos/P-Antilocapra_americana")`,
 para el usuario de Windows recuerde que tiene que especificar el disco donde se encuentra trabajando, es decir: C:, D:, etc.
3. Para comprobar la dirección del subdirectorio de trabajo se utiliza el comando `getwd()` y el cual despliega, por ejempl.:

```
[1] "/Datos/P-Antilocapra_americana"
```

 confirmandose así que se está trabajando en ese subdirectorio, para el caso del usuario Windows se despliega el disco donde se encuentra trabajando, (C:, D:, etc.).
4. Para generar una lista de todos los archivos en el subdirectorio de trabajo se utiliza el comando `list.files()`

En este momento ya se tiene declarado el subdirectorio de trabajo, se puede iniciar en trabajo con **R** y consultarse la base de datos por analizar.

Consulta y conocimiento de la base de datos

Después de ser creado un archivo con cualquier editor de texto u hoja de cálculo, se tiene que exportar con extensión `.txt` (delimitado por tabuladores o espacios) o `.csv` (separado por comas), el archivo se copia en el subdirectorio de trabajo y

se inicia la sesión de R. Un archivo .txt se incorpora a la sesión de R utilizando el comando `read.table()`:

```
ejemplo1 <- read.table("~/mydata/ejemplo1.txt")
```

o utilizar, si se tienen encabezados en las columnas del archivo,

```
ejemplo1 <- read.table("ejemplo1.txt", header = TRUE)
```

se puede utilizar `"header=T"` en lugar de `"header=TRUE"` pero no es recomendable utilizarlo si se está iniciando con el aprendizaje de R.

Para asegurarse de que se ha importado a la memoria de trabajo la base de datos se pueden listar los encabezados de las columnas con el comando, `names()`, por ejemplo: `names(ejemplo1)`.

Si el archivo contiene para cada columna un encabezado se tienen que utilizar la función `attach()` para llamar directamente a cada una de las columnas de forma independiente, por ejemplo: `attach(ejemplo1)`.

Para verificar si se tiene la base de datos en la memoria de R se escribe:

```
ejemplo1
```

Una segunda opción es el desplegar solamente una parte de la base de datos. El inicio de ésta con el comando:

```
head(ejemplo1)
```

y/o el final de la misma con:

```
tail(ejemplo1)
```

Cuando se ha generado la base de datos en una hoja de cálculo, se requiere que se encuentre ésta en un formato separado por comas "," en un archivo del tipo .csv, para tenerlo en la memoria se requiere del comando:

```
ejemplo <- read.csv("ejemplo1.csv", header=TRUE)
```

La opción `header=TRUE` se utiliza si las columnas de la base de datos se encuentran con identificador.

La base de datos se puede guardar en formato de R con el comando:

```
save(ejemplo1, file="ejemplo1.Rdata")
```

Es recomendable utilizar la extensión .Rdata para reconocer los archivos que puede utilizar R y mantener a los archivos originales como respaldo. Recuerde que **No es recomendable trabajar en la base de datos original.**

Al iniciar una nueva sesión en R se puede continuar trabajando en la base de datos utilizando el comando:

```
load(file="ejemplo1.Rdata")
```

La función `ls()` u `objects()` se pueden utilizar para listar los nombres de los objetos que se encuentran en la memoria o ambiente. La liberación de la memoria del ambiente de trabajo se realiza con la función `rm()` o `remove()` las cuales eliminan los objetos del ambiente de trabajo, puede ser utilizada en conjunto con otras funciones como `ls()`.

Ejemplos:

```
rm(ejemplo1)
```

```
remove(ejemplo1)
```

La opción conjunta de `rm()` y `ls()` elimina toda la información del ambiente de trabajo, es recomendable no usar esta opción si no se está completamente seguro de lo que se obtendrá de su uso. Se recomienda su uso en el caso de ser necesario iniciar el ambiente de trabajo con una nueva base de datos.

```
rm(list = ls())
```

Se puede importar datos de una hoja de cálculo mediante el uso de la memoria portapapeles o clipboard resultado del comando, `Ctrl c`, copiar. Se utiliza la función `read.delim` para incorporar al ambiente de **R** una base de datos. Primero se copia (`Ctrl c`) el área de la hoja de cálculo que se quiere incorporar a **R**, sin cerrar la hoja de cálculo ya que sólo funciona mientras la aplicación se encuentra utilizando la memoria, con el comando:

```
ejemplo2 <- read.delim("clipboard")
```

después se consulta si se ha incorporado al ambiente la base de datos escribiendo `ejemplo2`. La base de datos se salva en formato **R** con la función `save()`

```
save(ejemplo2, file="ejemplo2.Rdata")
```

Una calculadora gigante

Prácticamente se puede pensar que **R** es una calculadora gigante, a continuación se presentan diferentes funciones para los cálculos matemáticos, por ejemplo:

La suma de $2+3$, escriba:

```
2+3
```

y despliega

```
[1] 5
```

Se puede hacer una serie de operaciones sencillas, como por ejemplo:

Dividir $2=2$:

```
2/2
```

```
[1] 1
```

la raíz cuadrada de 9:

```
sqrt(9)
```

```
[1] 3
```

Para obtener la parte entera de la división:

```
7/%3
```

```
[1] 2
```

y el resto de la división se obtiene escribiendo

```
7%%3  
[1] 1
```

Los logaritmos y las diferentes bases, el logaritmo por defecto es el base e, se puede calcular diferentes logaritmos utilizando el comando `log(#, base= #)`, por ejemplo:

```
log(2.718282)  
[1] 1
```

```
log(10)  
[1] 2.302585
```

```
log(100)  
[1] 4.60517
```

```
log(10, base = 10)  
[1] 1
```

```
log(10, base= 2)  
[1] 3.321928
```

La Tabla 1 presenta las funciones algebraicas más comunes utilizadas en R.

Calcular el área de un círculo de 10 cm de radio, $(\pi * r^2)$.

```
pi*10^2  
[1] 314.1593
```

Calcular el volumen de un paracono $\frac{(3\pi)D^2A}{28}$ (sólido que se encuentra entre un cono y paraboloide) de 7 cm de diámetro, D y 20 cm de altura, A.

```
((3*pi)*(7^2)*20)/28  
[1] 329.8672
```

Tabla 1: Funciones algebraicas en R

Función	Operación
+, -, *, /	Suma, Resta, Multiplicación, División
/%/%	División entera
%%	Resto de la división
sin(), cos(), tan()	Funciones trigonométricas
asin(), acos(), atan()	Inversas de las funciones trigonométricas
exp(), log()	Exponencial y logaritmo natural
sqrt(), ^	Raíz cuadrada, Potencia
abs()	Valor absoluto
round()	Redondeo

Primer ejemplo en R

La primera forma de ingresar un grupo pequeño de datos en **R** es utilizando la función concatenación, `c()`. Al utilizar la función se combinan varios terminos. Suponga que en un muestreo preliminar ha contado el número de individuos en 9 cuadrantes (10x10 m), de la orquídea endémica *Mexicoa ghiesbreghtiana* (Rich. & Galeotti) Garay:

2 4 6 5 3 5 1 5 1

Para introducir estos datos en la sesión de **R** escriba:

```
x <- c(2,4,6,5,3,5,1,5,1)
```

Escriba, x obteniendo:

```
[1] 2 4 6 5 3 5 1 5 1
```

Es la manipulación más simple que realiza **R**, que opera con estructuras de datos. La estructura se llama vector numérico, la cual es una entidad ordenada de números. Lo que se ha hecho es crear un vector llamado x consistente en la serie de números, llamados argumentos, 2 4 6 5 3 5 1 5 1.

Al vector x se le ha asignado mediante la función `c()` una serie de elementos. El vector toma el valor de la concatenación de los argumentos. La asignación es realizada con el operador (`'<-'`) el cual esta formado por los caracteres `'<'` "menor que" y `'-'` signo "menos" que se tiene que escribir sin espacio y en el que señala la dirección de ésta. El operador `"="` puede ser utilizado en forma alternativa. La asignación equivalente puede ser hecha por medio de la función `assign()`.

```
assign("x", c(2,4,6,5,3,5,1,5,1))
```

Se pueden realizar nuevas asignaciones utilizando los vectores ya creados, como por ejemplo:

```
y <- c(x,0,x)
```

El cual crea un vector, y con los valores de x repetidos con un cero entre ellos.

```
y  
[1] 2 4 6 5 3 5 1 5 1 0 2 4 6 5 3 5 1 5 1
```

De vectores a base de datos

Se puede incorporar la información directamente en R con la función `c()` por ejemplo:

```
numero.de.planta<-c(1,1,2,2)  
anio<-c(2009,2010,2009,2010)  
diametro<-c(12.5,15,15,0)  
frutos<-c(3,4,2,0)
```

se pueden listar las variables de cada uno de los vectores:

```
numero.de.planta  
[1] 1 1 2 2  
anio  
[1] 2009 2010 2009 2010  
diametro  
[1] 12.5 15.0 15.0 0.0  
frutos  
[1] 3 4 2 0
```

Para crear una base de datos con 4 variables necesitamos unir los vectores utilizando la función `cbind()`, la cual basicamente es "une por columnas", en R no existen los acentos ni la letra ñ.

```
datos<-cbind(numero.de.planta,anio,diametro,frutos)  
datos  
  numero.de.planta anio    diametro  frutos  
[1,]             1 2009     12.5      3  
[2,]             1 2010     15.0      4  
[3,]             2 2009     15.0      2  
[4,]             2 2010      0.0      0
```

Se salva la base de datos en el formato de R y se puede utilizar para su análisis

```
write.table(datos,file="datos.Rdata")
```

y en una nueva sesión se utiliza el comando:

```
datos <- read.table("datos.Rdata")
```

o importarlos con el comando:

```
datos <- read.table("datos.Rdata", header=TRUE, quote="\")
```

MATRICES

Las matrices son simplemente un arreglo multidimensional de vectores. En las matrices los vectores que se juntan necesitan ser del mismo modo. Las matrices en R internamente se guardan como vectores que se sobrelapan. La función `matrix` convierte un vector en una matriz. Los argumentos `nrow=` y `ncol=` especifican el número de renglones y columnas. Si sólo uno de los argumentos se ingresa, el otro es asumido basándose en el número de elementos del vector. Como las matrices se guardan en columnas la función `matrix` asume que el vector de entrada se va a convertir en una matriz por columnas. Si se quiere ingresar la matriz por renglones entonces debes de usar el argumento `byrow=TRUE`. Una matriz no solo puede ser numérica sino que también puede ser de caracteres, sin embargo, las matrices solo pueden ser formadas por un tipo de elemento, numérico o carácter. Las matrices tienen el atributo `dim` el cual especifica la dimensión de la matriz y al ser ejecutado regresa un vector cuyos elementos pueden ser seleccionados con la función `nrow=` y `ncol=`. También se pueden asignar nombres a las columnas y a los renglones con el argumento `dimnames=` o posteriormente asignarlos a la matriz. Cuando queremos asignarle nombres a las columnas es importante tener en cuenta que las matrices pueden tener diferentes dimensiones y contener nombres diferentes de tal forma que el valor de `dimnames` debe ser una lista `list`, en donde el primer elemento es un vector de los nombres de los renglones y el segundo elemento de la lista es un vector de las columnas. Si no se requiere nombrar las columnas se puede usar el valor `NULL`. Por ejemplo, se quiere generar una matriz de 4x4

```
> matriz4x4 <-matrix(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,
15,16), ncol=4)
> matriz4x4
      [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
```

o se puede hacer una asignación por renglones

```
> matriz4x4 <-matrix(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,
15,16),
byrow=TRUE,nrow=4)
> matriz4x4
```

	[,1]	[,2]	[,3]	[,4]
[1,]	1	2	3	4
[2,]	5	6	7	8
[3,]	9	10	11	12
[4,]	13	14	15	16

o cambiar las dimensiones

```
> matriz8x2 <- matrix(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,
  15,16),
  byrow=TRUE,nrow=8)
> matriz8x2
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
[5,]    9   10
[6,]   11   12
[7,]   13   14
[8,]   15   16dimna
```

Se pueden asignar nombres de dos formas: de un vector para las columnas:

```
> dimnames(matriz8x2)<-list(c(1,2,3,4,5,6,7,8),c(1994,1995))
> matriz8x2
  1994 1995
1      1    2
2      3    4
3      5    6
4      7    8
5      9   10
6     11   12
7     13   14
8     15   16
```

o directamente al gener la matriz:

```
matriz8x2 <-matrix(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16),
  byrow=TRUE,nrow=8,dimnames=list(c(1,2,3,4,5,6,7,8),
  c(1994,1995)))
matriz8x2
```

Se puede acceder a partes de la matriz, por ejemplo, si quiero obtener el quinto dato del año 1994:


```
> matriz8x2[5,1]
[1] 9
```

en donde el corchete cuadrados contiene el número de renglón y después de la columna. Es importante recordar que el orden en el cual se asigna la posición (primero renglón y después columna). Por ejemplo si le hacemos el revés me dice que esa posición de la matriz no existe o está fuera de las dimensiones de la matriz.

```
> matriz8x2[1,5]
Error: subscript out of bounds

> dim(matriz8x2)
[1] 8 2
```

Se pueden usar columnas o renglones enteros de una matriz. En estos casos, se deja sin índice los renglones (se quieren todos los renglones) y solo se define la columna, en este caso, la segunda [, 2]. Por ejemplo, seleccionar únicamente los valores asociados a 1995

```
> solo.1995 <- matriz8x2[,2]
> solo.1995
1  2  3  4  5  6  7  8
2  4  6  8 10 12 14 16
```

MANIPULACION DE MATRICES

Existen otras formas de ingresar matrices en R. Por ejemplo, se declaran una serie de vectores, se unen y se convierten esos objetos en una matriz.

```
> renglon1 <- c(1,2,3)
> renglon2 <- c(4,5,6)
> renglon3 <- c(7,8,9)
>
> union_renglones <- rbind(renglon1, renglon2, renglon3)
> como_matriz <- as.matrix(union_renglones)
> como_matriz
      [,1] [,2] [,3]
renglon1    1    2    3
renglon2    4    5    6
renglon3    7    8    9
> is.matrix(como_matriz)
[1] TRUE
```

Se pueden incluir en la matriz los nombres de los objetos creados. Por ejemplo, cada renglón significa el estado de un ciclo de vida. Primero creamos el objeto de los nombres que se quieren asignar y se añaden a la matriz con el comando `row.names`

```
> estados <- c("huevo", "larva", "adulto")
```

```
> rownames(como_matriz) <- estados
> como_matriz
      [,1] [,2] [,3]
huevo     1     2     3
larva     4     5     6
adulto    7     8     9
```

Para las columnas existe la función `colnames` que asigna a columnas los nombres:

```
> colnames(como_matriz) <- estados
> como_matriz
      huevo larva adulto
huevo     1     2     3
larva     4     5     6
adulto    7     8     9
```

La manipulación de los datos de una matriz se hacen basicamente con valores en blanco lo cual significa “todos los renglones” o “todas las columnas”. Se refiere a las matrices por renglones y columnas. Una matriz de 31 quiere decir que tiene tres renglones y una columna. La matriz es referida por el número de renglones *primero*. Las matrices se componen de elementos, donde cada elemento se define por sus “coordenadas” en renglones y columnas, por ejemplo `[renglones, columnas]`. Se requiere calcular el promedio de los valores de la tercera columna de la matriz `como_matrix`.

```
> mean(como_matriz[,3])
[1] 6
```

Se requiere el promedio de la categoria de larva y adulto (los renglones 2 al 3 de la tercera columna.

```
> mean(como_matriz[2:3,3])
> sum(como_matriz[3,])
[1] 24
```

o la suma del último renglón

```
> sum(como_matriz[3,])
[1] 24
```

Para multiplicar matrices es necesario multiplicar y después sumar cada renglón por cada columna. De manera mas especifica, se multiplica cada elemento del renglón de la matriz **A** por cada elemento de la columna de la matriz **B**, se suman y se coloca esta suma en la matriz resultante **C**. Si consideramos la matriz:

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix}$$

y se multiplica por el vector

$$B = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

se obtiene una matriz C de la forma:

$$C = \begin{pmatrix} 1a+2b+3c+4d \\ 5a+6b+7c+8d \\ 9a+10b+11c+12d \end{pmatrix}$$

Se requiere que el número de columnas de la primer matriz tenga las mismas dimensiones que el número de renglones del vector (también puede ser una matriz). Así el vector resultante tendrá la misma dimensión que el vector inicial.

En R la multiplicación de matrices es muy sencilla, si se quiere multiplicar una par de matrices se usa el comando `%*%`.

```
> como_matriz%%como_matriz
      huevo  larva  adulto
huevo     30     36     42
larva     66     81     96
adulto    102    126    150
```

El eigen análisis es una técnica matemática que resume datos y básicamente transforma una matriz cuadrada en elementos independientes y ortogonales llamados eigenvectores y sus eigenvalores. El análisis encuentra de manera analítica las soluciones para λ y w de

$$Aw = \lambda w$$

donde A es una matriz, λ es un eigenvalor y w es una eigenvector. si se tiene:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} w_{11} \\ w_{21} \\ w_{31} \end{pmatrix} = \lambda \begin{pmatrix} w_{11} \\ w_{21} \\ w_{31} \end{pmatrix}$$

Tipicamente existe un número infinito de soluciones a esta ecuación y lo que hace el eigenanálisis es encontrar las soluciones independientes una de otra. La primera solución captura los atributos más importantes. En poblaciones esto correspondería al eigenvalor dominante y su eigenvector correspondiente w_1 .

Para la transposición de matrices solo se antepone una `t` para multiplicar la matriz contra su transpuesta:

```
> como_matriz%*%t(como_matriz)
      huevo larva adulto
huevo    14    32    50
larva     32    77   122
adulto    50   122   194
```

Las sumas y restas de matrices se hacen con las operaciones de R:

```
> suma_matrices <- como_matriz+como_matriz
> suma_matrices
      huevo larva adulto
huevo     2     4     6
larva     8    10    12
adulto    14    16    18
> resta_matrices <- suma_matrices-como_matriz
> resta_matrices
      huevo larva adulto
huevo     1     2     3
larva     4     5     6
adulto     7     8     9
```

Tabla 2: Principales funciones específicas para los vectores en R

Función	Significado
length()	Longitud del vector
sum()	Suma de los elementos del vector
prod()	Producto de los elementos del vector
max()	Máximo valor en el vector
min()	Mínimo valor en el vector
cumsum()	Suma acumulada de los elementos de vector
cummax()	Máximos acumulados del vector
cummin()	Mínimos acumulados del vector
cumprod()	Producto acumulado de los elementos del vector
diff()	Diferencia entre elementos del vector
unique()	Vector de valores únicos
duplicated()	Duplicación de los elementos en un vector lógico

Estadística descriptiva en R

Existen varias funciones específicas de R, Tabla 2 y las funciones estadísticas, Tabla 3, las cuales pueden ser aplicadas a los vectores numéricos.

Utilizando el vector `x`, `x <- c(2, 4, 6, 5, 3, 5, 1, 5, 1)`, podemos utilizar las funciones de los vectores.

```
length(x)
[1] 9
```

```

sum(x)
[1] 32
prod(x)
[1] 18000
max(x)
[1] 6
max(y)
[1] 6
min(x)
[1] 1
cumsum(x)
[1] 2 6 12 17 20 25 26 31 32
cummax(x)
[1] 2 4 6 6 6 6 6 6 6
cummin(x)
[1] 2 2 2 2 2 2 1 1 1
cumprod(x)
[1] 2 8 48 240 720 3600 3600 18000 18000
diff(x)
[1] 2 2 -1 -2 2 -4 4 -4
unique(x)
[1] 2 4 6 5 3 1
duplicated(x)
[1] FALSE FALSE FALSE FALSE FALSE TRUE FALSE TRUE TRUE
x
[1] 2 4 6 5 3 5 1 5 1

```

Las funciones de estadística descriptiva que pueden ser utilizadas con los vectores en R, se pueden ver en la Tabla 3.

Tabla 3: Funciones de estadística descriptiva más utilizadas en R

Nombre	Significado
median ()	Mediana del vector
mean()	Media del vector
sd ()	Desviación estándar
var()	Varianza de los elementos del vector
quantile()	Cuartil (Cuantil)
IQR()	Rango intercuartil
range ()	Rango del vector
summary()	Resumen estadísticos descriptivos

Ejemplos del uso de las funciones de estadística descriptiva:

```

median(x)
[1] 4
mean(x)
[1] 3.555556

```

```

sd(x)
[1] 1.878238
var(x)
[1] 3.527778
quantile(x)
0%    25%    50%    75%   100%
1      2      4      5      6
IQR(x)
[1] 3
range(x)
[1] 1 6
summary(x)
Min. 1st Qu. Median  Mean 3rd Qu.  Max.
1.000  2.000  4.000  3.556  5.000  6.000

```

La información de todas las funciones estadísticas presentes en R se obtiene escribiendo `library(help="stats")` desplegándose así todas las opciones del paquete 'stats', uno de los paquetes básicos de R.

```

library(help="stats")

Information on package 'stats'

Description:

Package:      stats
Version:      3.1.0
Priority:      base
Title:        The R Stats Package
Author:       R Core Team and contributors worldwide
Maintainer:   R Core Team <R-core@r-project.org>
Description:  R statistical functions
License:      Part of R 3.1.0
Built:        R3.1.0;i686-pc-linux-gnu;2014-04-10 20:21:40UTC;
              unix

```

Se despliega la información del paquete y el índice con las diferentes funciones, las cuales se pueden consultar, `help(arima)`, modelado de series de tiempo ARIMA, por ejemplo.

Gráficas en R

Las gráficas en R se pueden desplegar en alguno de los dispositivos de salida, en la pantalla o en un archivo. Los archivos gráficos pueden tener varios formatos según el sistema operativo que el usuario tenga instalado. Los dispositivos gráficos de pantalla son de tres tipos, Unix/Linux, MacOS X y Windows, dependiendo del

sistema operativo utilizado la función para declarar la salida gráfica en la pantalla es: `X11()`, `quartz` o `windows()`, respectivamente.

Las funciones que controlan las salidas gráficas, ya sea en la pantalla o archivo (texto o imagen), se encuentran divididas en tres grupos de funciones. Funciones de alto nivel, funciones utilizadas para generar un nuevo gráfico; de bajo nivel, las cuales se utilizan para añadir información al gráfico ya generado como tipos de líneas y etiquetas; y las funciones interactivas las cuales se utilizan para añadir o eliminar información al gráfico. Existe también una gran variedad de parámetros los cuales pueden ser utilizados dentro del gráfico.

Las funciones para generar gráficos, las cuales son parte de paquete de instalación de R y más comunes en su uso, son: `plot()`, `boxplot()`, `barplot()`, `stars()`, `pairs()`, `matplot()`, `hist()`, `image()`, `contour()`. Se presentan algunos ejemplos básicos del uso de algunas funciones, si se requiere más información sobre el comportamiento y características utilice en comando `help(nombre-funcion)`, por ejemplo, `help(plot)`. La función `plot()` es la más básica, al usarse con una variable se genera una gráfica con el eje de las abscisas con el número de renglón (usualmente es el orden de colecta) y en las ordenadas el valor de la variable, llamada gráfica indexada. Al utilizar dos variables para generar la gráfica, una independiente (x) y otra dependiente (y), se genera una gráfica donde la primera variable es asignada al eje de las abscisas (eje x) y la segunda variable al eje de las ordenadas (eje y).

Para los ejemplos se utilizaran tres variables generadas con la función `runif()`, distribución uniforme aleatoria, a la cual se le pueden asignar valores mínimos y máximos, `runif(n, min=0, max=1)`. Utilizando la función `runif()` se generaran las densidades poblacionales de dos especies en 50 parcelas de 1x1 m de nuestro muestreo. Se utiliza la función `runif()` para generar el número de individuos de las dos especies y la función `round()` para redondear el valor a enteros sin decimales, la *sp1* será una especie rara que se encuentra agrupada, `Nisp1`, y la *sp2* será un especie abundante muy dispersa `Nisp2`. Se puede escribir las dos funciones en la misma línea sólo se separan por una ";":

```
Nisp1 <- round(runif(50, min= 0, max = 12)) ;  
Nisp2 <- round(runif(50, min= 0, max = 35))
```

y la secuencia del número de parcelas que se muestrearon, 50.

```
cuadrante <- 1:50
```

Esto genera el vector (cuadrante) con números que van del 1 al 50.

También se puede usar la función `seq()`:

```
prueba<-seq(from=3,to=21,by=1)
```

quiere decir, prueba es un vector generado por la función `(seq)` desde el número 3 (from =) al número 21 (to =) y de uno en uno (by =):

```
[1] 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
```

Al cambiar el valor de `by=1` por `by=2` el nuevo vector será de cada segundo número:

```
[1] 3 5 7 9 11 13 15 17 19 21
```

Las primeras gráficas se generan escribiendo `plot(Nisp1)` y `plot(Nisp2)`, si se escribe `plot(cuadrante,Nisp1)` y `cuadrante,Nisp2` se obtendrá el mismo resultado con la diferencia de que en el eje de las abscisas se despliega las leyendas **index** y **cuadrante** para cada tipo de gráfica respectivamente, ver el ejemplo en la fig. 5.

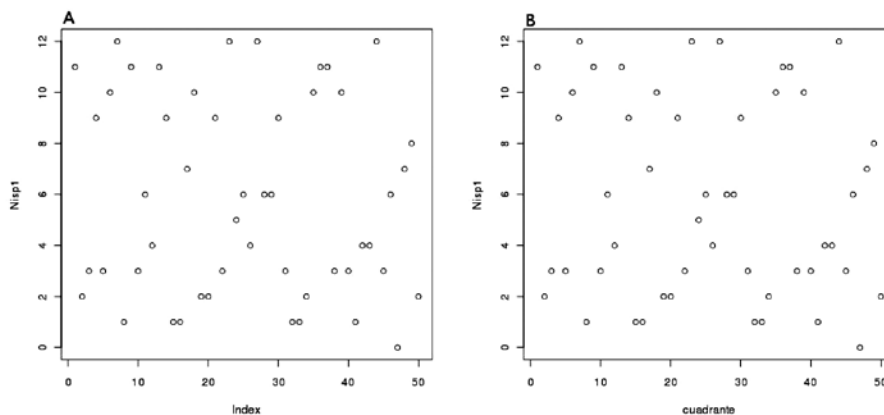


Figura 5: Gráficas generadas utilizando la función `plot()`; A: `plot(Nisp1)` y B: `plot(cuadrante,Nisp1)`

La gráfica del número de individuos por especie, `Nisp1` y `Nisp2` utiliza el mismo comando `plot(Nisp1,Nisp2)`, ver ejemplo en la figura 6.

El histograma de frecuencias de `Nisp1` produce mediante la función `hist()`, la figura 7 es el resultado de la aplicación de la función: `hist(Nisp1)`.

Si se tiene más de dos especies, que es el caso siempre que se realiza un muestreo, se pueden agregar en una gráfica de dispersión, fig. 6. Para generar una nueva variable, utilizar el mismo comando de `Nisp1` y añadir `Nisp3` y `Nisp4`, además de los nombres a los ejes y la leyenda de cada una de las variables, especies, por ejemplo. Usualmente la simbología de la gráfica se pone dentro de ésta cuando se genera como borrador pero en cualquier reporte o artículo se escribe en el pie de figura. Se puede graficar el número de individuos por cada especie en los diferentes cuadrantes. Esta no es la manera más apropiada de reportar el número de individuos pero para fines prácticos, es un tipo de gráfica muy utilizada. Los comandos para generar la gráfica son los siguientes:

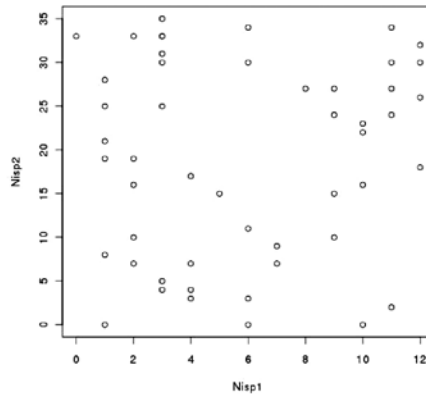


Figura 6: Gráfica generada utilizando la función `plot(x,y)` para las dos especies Nisp1 y Nisp2

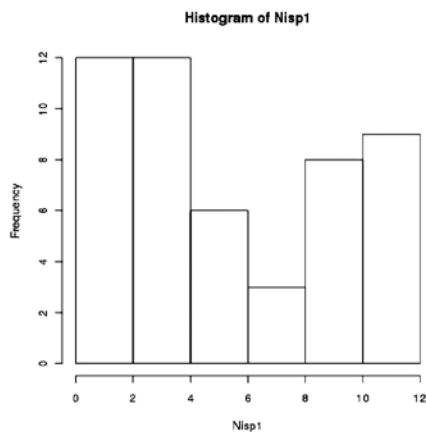


Figura 7: Gráfica de especie Nisp1 utilizando la función `hist()`

```
plot(cuadrante, Nisp2, xlab="Cuadrante", ylab="Especies 1,2,3 y 4")
points(cuadrante, Nisp1, pch = 2)
points(cuadrante, Nisp3, pch = 3)
points(cuadrante, Nisp4, pch = 4)
legend(1, 21, c("Nisp1", "Nisp2", "Nisp3", "Nisp4"), pch= c(2,1,3,4))
```

La primera función tiene los argumentos para asignar los nombres de los ejes, `xlab=""` e `ylab=""`. Las variables (Nispn) son incorporadas con la función `points()` y el argumento `pch=` es parte de la función `plot()`. Los números del argumento `pch=` corresponden a los diferentes símbolos disponibles para las gráficas, el lector puede explorar utilizando diferentes valores y comparar los resultados. La leyenda se encuentra colocada en un lugar donde es visible dentro de la gráfica con la función `legend()`, con la posición en x y y dentro de la gráfica, además se utiliza la función de concatenación `c()` para los nombres de las variables y los símbolos correspondientes, ver fig.8.

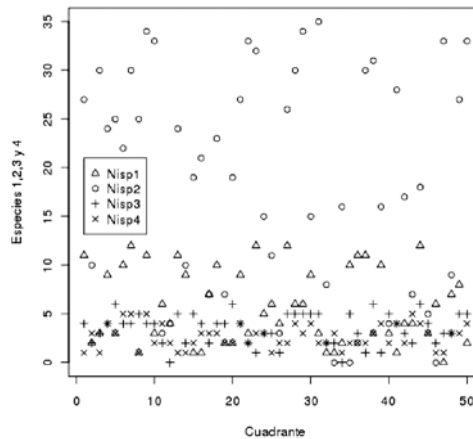


Figura 8: Gráfica generada utilizando la función plot(x,y) con más de dos variables y leyendas.

Una gráfica muy utilizada es la de puntos y líneas, se genera con las siguientes funciones y argumentos, ver fig. 9.

```
plot(cuadrante, Nisp2, type="o", pch =2, lty =2, col="blue",
     ylab="Numero de individuos" )
lines(cuadrante, Nisp1, type="o", pch =25, lty =2 , col="red")
lines(cuadrante, Nisp3, type="o", pch = 21, lty =4, col="green")
lines(cuadrante, Nisp4, type="o", pch = 18, lty = 6, col="black")
legend(3, 21, c("Nisp1","Nisp2","Nisp3","Nisp4"), pch= c(2,25,21,18))
```

Una segunda opción se puede generar mediante los comandos:

```
plot(cuadrante,Nisp2,xlab ="Cuadrantes",ylab="Especies",type ="l")
lines(cuadrante, Nisp1, lty = 2 )
lines(cuadrante, Nisp3, lty = 4 )
lines(cuadrante, Nisp4, lty = 6 )
```

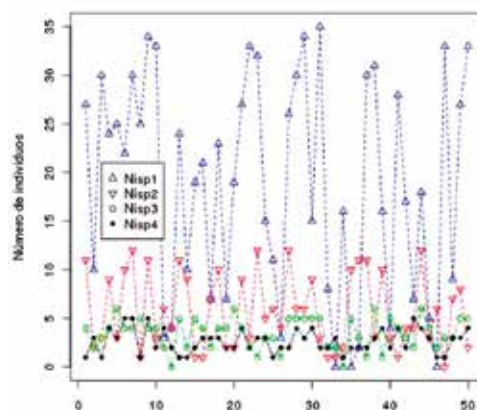


Figura 9: Gráfica de puntos y líneas utilizando la función plot(x,y) con más de dos variables y leyenda

R tiene la opción de gráficos de caja (Box and whisker plot, en inglés) también llamados de “caja y bigotes”, los cuales fueron propuestos por J.W. Tukey (1915-2000) en 1977. Las gráficas de cajas son generadas utilizando la función `boxplot()`.

```
boxplot(Nisp2)
```

La función `boxplot()` puede tener los siguientes argumentos.

```
boxplot( Nisp2 , horizontal =T, col="lightblue",
boxwex = 0.5, xlab = "N. de ind." ,
ylab = "Sp. 2", varwidth = TRUE, boxfill = "lightblue",
border = "red", notch = TRUE, font = 4, main = "Muestreo")
```

La gráfica generada mediante la función `boxplot` presenta una gran cantidad de los argumentos disponibles. La figura 10 muestra dos gráficas de caja, vertical y horizontal, además una gráfica con las cuatro especies en un solo archivo de salida, utilizando la función `par()`. La función, `par()`, es usada para fijar los parámetros de las gráficas de salida. Las gráficas pueden variar en tamaño, por lo tanto se tiene que definir los parámetros para cada gráfica. Los parámetros más utilizados son los de las figuras múltiples por fila y columna, `mfrow()` (multiple figures by row) y `mfcoll()` (multiple figures by column), respectivamente, los cuales determinan la posición de las gráficas en una sola salida, pantalla o archivo. Si se utiliza `mfrow=c(n,m)` el resultado será una matriz $n*m$ de gráficas ordenadas en renglones y `mfcoll=c(n,m)` será una matriz de gráficas ordenadas por columnas. Existen una gran cantidad de opciones las cuales pueden ser consultadas mediante la ayuda de R en línea.

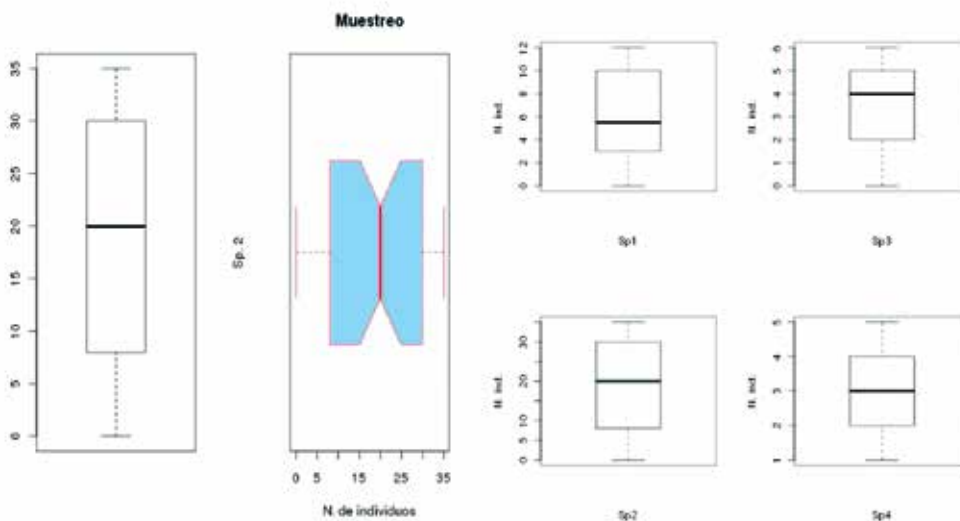


Figura 10: Boxplot A: gráfica de caja vertical y horizontal y B: salida de varias gráficas en una sola pantalla utilizando la función `par()`

Salvando los gráficos

Al generar un gráfico se despliega una ventana en la pantalla, la cual es llamada despliegue gráfico. **R** tiene tres diferentes tipos de despliegue dependiendo del sistema operativo donde se encuentra instalado. Las funciones utilizadas para cada sistema operativo son: Unix y Linux, `x11()`; MacOS X, `quartz()` y Windows, `windows()`. Las funciones gráficas en **R** para el manejo de ventanas generadas por las funciones gráficas son: función de listado del número de ventanas activas, `dev.set()`; cierre de ventanas una por una, `dev.off()` y cierre de todas las ventanas gráficas, `graphics.off()`.

Las gráficas pueden ser incorporadas en cualquier editor o procesador de textos de cualquier sistema operativo. **R** salva los archivos en una gran variedad de formatos, ver Tabla 4. Entre los principales se encuentran: **PNG** (Portable Network Graphics), **BMP** (Windows Bitmap), **ps** (postscript), **PDF** (Portable Document Format) y **JPEG** (Joint Photographic Experts Group) los cuales son los más utilizados. Las funciones tienen como salida un formato específico de archivo, la ayuda en **R** se obtiene utilizando el comando `?device`. A la función de tipo de archivo como salida se le asigna un nombre y extensión, después se utiliza las funciones para generar la gráfica y finalmente utilizar la función `dev.off` para cerrar y grabar la pantalla gráfica o `graphics.off` para cerrar y grabar todas las ventanas con las gráficas desplegadas. Es recomendable usar las funciones anteriores para cerrar las las ventanas gráficas para evitar conflictos. Algunos ejemplos para diferentes formatos son:

```
png("grafica-c-spl.png")
plot(cuadrante,Nisp1)
dev.off()

xfig("histosp1.fig", onefile = T)
hist(Nisp2)
dev.off

pdf("boxplotsp3.pdf")
boxplot(Nisp3)
dev.off()

postscript("plotesp-1-2.ps")
plot(cuadrante,Nisp2,xlab="Cuadrantes",ylab="Especies",-
      type="l")lines(cuadrante, Nisp1, lty = 2)
dev.off()
```

Scripts, archivos de comandos

El registro sobre el uso de las funciones en **R** se guarda en el archivo `.Rhistory`, conteniendo la información de los diferentes comandos utilizados en las sesiones,

siempre y cuando al finalizar la sesión, `q()` se responda y (yes) a la pregunta `Save workspace image? [y/n/c]:`.

Tabla 4: Algunas funciones de las salidas de los archivos gráficos

Función	Tipo de archivo
<code>postscript()</code>	ps
<code>pdf()</code>	pdf
<code>png()</code>	png
<code>xfig()</code>	fig
<code>bitmap()</code>	BMP
<code>pictex()</code>	tex
<code>jpeg()</code>	jpg
<code>bmp()</code>	bmp
<code>win.metafile()</code>	wmf
<code>tiff()</code>	tif

Así, se tiene acceso a los diferentes procedimientos realizados durante la sesión(es) anterior(es) de R los cuales se pueden copiar y repetir cada vez que sea necesario.

En R se tiene la posibilidad de evaluar secuencialmente las funciones y expresiones en un archivo de texto, llamado script. Los archivos script son secuencias de comandos generadas durante el análisis de la información o generación de diferentes gráficos. Los comandos utilizados se guardan en un archivo ASCII. El archivo script se ejecuta por medio de la función `source()`. Los archivos script se utilizan para bases de datos que tienen la misma estructura y el análisis se hace en forma secuencial. Los comentarios se escriben después de signo `#` los cuales no son interpretados por R. El archivo script tiene la extensión `.R` o `.r` usualmente.

Ejemplo de un archivo script:

```
## ejemplo archivo script, ejemploA.R
Nisp1 <- round(runif(50, min=0,max = 12))
Nisp2 <- round(runif(50, min=0,max = 35))

cuadrante <- 1:50

boxplot(Nisp2,xlab="Cuadrante",ylab="No. de individuos
Nips2",notch=T)
```

El archivo se ejecuta desde R por medio del comando:

```
source("ejemploA.R")
```

La función `sink()` se utiliza para almacenar los resultados en un archivo de texto, por ejemplo; `sink("salidaStat.out")`, en el subdirectorío de trabajo. La desactivación se realiza utilizando las funciones `sink()`, de esta forma el archivo

de salida puede ser incorporado en un editor o al procesador de textos. La función `unlink("salidaStat.out")` borra desde el ambiente de **R** el archivo en el subdirectorio de trabajo y dependiendo del sistema operativo se puede o no recuperar.

Un ejemplo de un archivo script el cual utiliza la base de datos de las especies y tiene como salida gráficas de caja o de Tukey.

```
jpeg("boxplotexampl1.jpg")

### ocho graficas de caja en un solo archivo organizadas
par(mfrow = c(2,4))

boxplot(Nisp1, xlab = "Sp1" , ylab ="N. ind.")
boxplot(Nisp2, xlab = "Sp2" , ylab ="N. ind.")
boxplot(Nisp3, xlab = "Sp3" , ylab ="N. ind.")
boxplot(Nisp4, xlab = "Sp4" , ylab ="N. ind.")

boxplot(Nisp1, horizontal =T, xlab = "N. ind.",
ylab ="Sp1", border = "red",notch = TRUE)
boxplot(Nisp2, horizontal =T, xlab = "N. ind." ,
ylab ="Sp2", border = "red",notch = TRUE)
boxplot(Nisp3, horizontal =T, xlab = "N. ind." ,
ylab ="Sp3", border = "red",notch = TRUE)
boxplot(Nisp4, horizontal =T, xlab = "N. ind." ,
ylab ="Sp4", border = "red",notch = TRUE)

dev.off()
```

Instalación de popbio

La instalación de cualquier paquete en **R** se realiza por medio de la función `install.packages()`, ésta simplemente instala en el subdirectorio de **R** el paquete a utilizar se incorpora a la memoria utilizando el la función `library()`.

El paquete `popbio` se instala y se ejecuta utilizando las dos funciones:

```
install.packages("popbio")
library("popbio")
```

El paquete queda así disponible para la sesión de trabajo. Al iniciar una nueva sesión se requiere llamar nuevamente al paquete, instalado previamente. Se puede utilizar la función `demo()` para verificar la ejecución del paquete. Los ejemplos se incian escribiendo: `demo(Caswell)` por ejemplo.

Demos in package "popbio":

Caswell	Create plots from matrix population models book
fillmore	Fillmore Canyon demography plots
stage.classify	Methods for selecting stage classes

La ayuda en línea se obtiene con el comando `?popbio`, el cual despliega la información en la consola de trabajo.

01.Introduction package:popbio R Documentation

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 [URL:http://www.jstatsoft.org/v22/i11](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

Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation, Second edition. Sinauer, Sunderland:, Massachusetts, USA.

Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

La cita de popbio se puede obter con la función `citation("popbio")`, se despliega en formato texto y en BibTeX para L^AT_EX.

To cite popbio in publications use:

Stubben, C.J. and Milligan, B.G. 2007. Estimating and Analyzing Demographic Models Using the popbio Package in R. Journal of Statistical Software 22:11.

A BibTeX entry for LaTeX users is

```
@Article{,
  author = {Chris J. Stubben and Brook G. Milligan},
  title = {Estimating and Analyzing Demographic Models
           Using the popbio Package in R},
  journal = {Journal of Statistical Software},
  volume = {22},
  number = {11},
  year = {2007},
}
```

The paper is available at <http://www.jstatsoft.org/v22/i11>

La función `library(help="popbio")` presenta la información del paquete **popbio**, la cual puede ser consultada desde la consola. Desplegando a continuación el índice de las funciones de paquete **popbio**.

Information on package 'popbio'

Description:

```
Package:      popbio
Version:      2.4
Author:       Chris Stubben, Brook Milligan, Patrick Nantel
Maintainer:   Chris Stubben <stubben@lanl.gov>
Date:         2012-02-05
Title:        Construction and analysis of matrix population models
License:      GPL
Depends:      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).
Packaged:     2012-03-20 17:38:54 UTC; stubben
Repository:   CRAN
Date/Publication: 2012-03-20 21:30:23
Built:        R 3.1.0; ; 2014-06-24 03:58:19 UTC; unix
```

MANTENIMIENTO AL DÍA DE R Y POPBIO

El mantenimiento de los paquetes instalados en R se realiza por medio de la función:

- `update.packages()`, escribiendo `update.packages("popbio")` se instalará la última versión del paquete, en este caso popbio.

RECOMENDACIONES PARA EL USO DE R

Recomendamos usar la consola del sistema para iniciar el entrenamiento en el uso de R. Existen algunos editores que pueden funcionar como terminales interactivas, como por ejemplo:

- emacs, instalando el paquete ESS (Emacs Speak Statistics), siendo una de las opciones recomendadas para Linux, encontrara información y los manuales en: <http://ess.r-project.org/index.php?Section=home>
- R studio representa la opción más interesante y más estable de los paquetes gráficos para R, existen programas para Os X, Linux y Windows, se encuentra en <http://www.rstudio.com/>
- Una buena elección: Tinn-R, para ser usado en el sistema operativo Windows, se encontra en: <https://sourceforge.net/projects/tinn-r>
- Para MacOS X una buena opción es <http://www.peterborgapps.com/smultron/> smultron, el cual tiene un costo mínimo; otra opción es Aqua-

text que también tiene una versión de emacs, el manual se encuentra en:
<http://aquamacs.org/>

REFERENCIAS

- Crawley, M. J. 2013 The R book. 2a. ed. John Wiley & Sons Ltd, England, U.K.
- Hornik, K. 2013. The R FAQ, URL <http://CRAN.R-project.org/doc/FAQ/R-FAQ.html>
- Leisch, F. y R-core. 2014. Sweave User Manual.
URL <http://stat.ethz.ch/R-manual/R-patched/library/utils/doc/Sweave.pdf>
- Morales Rivera, M. A. S/F. Generación automática de reportes con R y L^AT_EX.
URL http://cran.r-project.org/doc/contrib/Rivera-Tutorial_Sweave.pdf
- Rudloff, P. 2005. An Introduction to Sweave. The Political Methodologist 13(1):8-13
- Tukey, J. W. 1977. Exploratory Data Analysis. Addison-Wesley MA. USA.
- Zuur, A. F., Ieno, E. N., Meesters, E. H. W. G. 2009. A Beginner's Guide to R. Springer. N.Y. U.S.A.

2. Paquete popbio

popbio Introducción al paquete popbio

DESCRIPCIÓN

En demografía de plantas siempre se cuenta el número de individuos de la misma especie que están presentes en un sitio o tiempo dado, a lo que denominamos población. La población tiene atributos diferentes a los individuos pero representan lo que en conjunto todos ellos están haciendo, generando un esquema promedio. Podemos hacernos exactamente la misma pregunta en cualquier tipo de estudio de ecología de poblaciones, ¿qué determina el crecimiento (o disminución) de una población? Los procesos demográficos que determinan el número de individuos de una población son la natalidad o número de individuos que nacen en una población (B), la mortalidad o número de individuos que mueren en la población (D) y la migración (individuos que inmigran (I) y los individuos que emigran (E)). Sin embargo, los procesos migratorios generalmente son ignorados en los modelos, dado que en la mayoría de los casos son muy difíciles de medir. El modelo básico que determina el número de individuos (N) en cualquier intervalo de tiempo (t) y que originalmente fue desarrollado es:

$$N_{t+1} = N_t + B + I - D - E \quad (1)$$

Si se supone que no hay migración, el crecimiento de la población va a depender de B y de D de tal forma que si se denoma $B - D$ como λ se obtiene:

$$N_{t+1} = \lambda N_t \quad (2)$$

Para cualquier tiempo:

$$N_t = \lambda^t N_{\text{inicial}} \quad (3)$$

$$N_t = N_0 e^{rt} \quad (4)$$

A partir del modelo sencillo expresado en (1) se han desarrollado todos los modelos que permiten medir la tasa de crecimiento de cualquier población, generalmente haciendo la elección del modelo de acuerdo al ciclo de vida de la especie y a la duración del mismo, al momento en el que ocurre la reproducción, o si las generaciones son discretas o sobrepuestas. Por ejemplo, las expresiones más elementales para describir el número de individuos de cualquier población con crecimiento poblacional discreto son las ecuaciones 1 y 2 o con crecimiento poblacional continuo (ecuaciones 3 y 4), además los modelos 2 y 4 se usan para la proyección del tamaño poblacional en el tiempo. Ambos modelos pueden posteriormente ser utilizados de manera similar en poblaciones que se estructuran o se subdividen por edades, estadios de desarrollo o tamaño de los individuos que son elaborados en forma de tablas de vida y en particular de matrices poblacionales (ver Caswell 2001). La tasa de crecimiento poblacional se puede calcular directamente a partir de la estimación de las tasas instantáneas de natalidad (b) y de mortalidad (d) de individuos $B = b/N$ y $D = d/N$, respectivamente. El parámetro maltusiano (r o tasa intrínseca de crecimiento poblacional) puede estimarse directamente para pobla-

ciones no estructuradas como $r=(b-d)$ individuos/ind/tiempo. Por otro lado, la tasa finita de crecimiento poblacional denominada lambda (λ) puede estimarse al dividir el número de individuos en la población de un tiempo al otro, $\lambda=N_{t+1}/N_t$ y fácilmente se puede ir de un modelo continuo a un modelo discreto con la relación entre las tasas de crecimiento de $\lambda=e^r$ o $\ln\lambda=r$ (Caswell 2001). Ambos estimadores del crecimiento nos indican si la población está creciendo; la r , en términos de la ganancia o pérdida de individuos, donde, si $r=0$, la población se mantiene con el mismo número de individuos (población en equilibrio), si $r>0$, es decir positiva, se produce el número de individuos que nos indique su valor por cada individuo en una unidad de tiempo, y si $r<0$, es decir negativa, entonces la población pierde los individuos que indica su valor, por cada individuo presente en la población, por eso es una tasa per cápita. λ es una tasa adimensional (no tiene unidades), siempre es positiva y nos indica en qué proporción está cambiando la población, si es menor a uno la población disminuye y si es mayor a uno la población crece.

En poblaciones estructuradas se puede trabajar con tablas de vida (poblaciones estructuradas por edad) o con modelos matriciales poblacionales (para cualquier tipo de estructura; por ejemplo diferente sexo, estadio de desarrollo; Caswell, 2001). Es importante señalar que para las plantas (y todos los organismos clonales) generalmente no se conoce la edad y pueden permanecer en la misma categoría de estado o de tamaño de un tiempo a otro, por lo que el modelo adecuado es el de crecimiento discreto, matricial y de tipo Lefkovitch (Caswell, 2001). Este modelo matricial permite que el crecimiento de la población se estime por la iteración de un vector de densidad poblacional por una matriz de transiciones que resumen los atributos demográficos, "método de potencias" (Caswell, 2001):

$$n_{t+1}=An_t \quad (5)$$

en donde n_t son los vectores poblacionales en t y $t+1$, A es una matriz de transiciones demográficas (resumen demográfico de los procesos de natalidad, fecundidad y supervivencia; esta matriz se calcula con los datos del destino de individuos, etiquetados e identificables claramente, de una temporada de crecimiento a otra, generalmente anual, y sus fertilidades). La construcción de la matriz puede hacerse en **popbio**. Sin embargo, resolver una ecuación de este tipo con el método de potencias no siempre es posible, por lo que los métodos numéricos, como el que usa **R** o **Matlab** son más recomendables. Además, si se trabajan varios años, diferentes sitios, etc., resolver las matrices con el método de potencias es muy impreciso. Se ha corroborado que los resultados son idénticos si se usa **Matlab** (un programa excelente para resolver álgebra lineal, pero de muy alto costo económico) o **R**.

Las tasas vitales se resumen en formato matricial de tal manera que se obtiene una estructura matricial:

$$\begin{pmatrix} 0 & 2 & 0 & 55 \\ 0,15 & 0,89 & 0 & 0 \\ 0 & 0,05 & 0,87 & 0,1 \\ 0 & 0 & 0,12 & 0,96 \end{pmatrix} \begin{pmatrix} 10 \\ 24 \\ 35 \\ 52 \end{pmatrix}$$

La matriz muestra las regiones y entradas (a_{ij}) de una matriz **A** de transiciones hipotética. El número de categorías (columnas j y renglones i , siempre simétrico) depende de la estructura poblacional de la especie de estudio. Las entradas o elementos a_{ij} indican a las variables demográficas como probabilidades o como número de vástagos y los subíndices dan la referencia a la columna y la fila a la que corresponden (contrario a como se asignan renglones y columnas en **R**). Las regiones en la matriz representan los procesos demográficos y generalmente se conservan en las zonas que a continuación se describen. En este ejemplo, los individuos de la población están agrupados en 4 categorías de estado (o en 4 estadios). En el primer renglón las entradas o elementos de la matriz (a_{1j}) representan la fecundidad por vía sexual en número de vástagos de la categoría reproductiva de Adultos I a la categoría de semillas; los elementos diagonales ($a_{i=j}$) representan las probabilidades de quedarse del tiempo t a $t+1$ en el mismo estado de desarrollo, no crecer o mantenerse igual. Los elementos subdiagonales ($a_{i>j}$) representan el crecimiento y los elementos supradiagonales ($a_{i<j}$) pueden representar el encogimiento o retroceso a estados anteriores (p. ej. $a_{34}=0,1$, indica que el 10% de los individuos adultos regresan a la categoría de Juveniles II de t a $t+1$) y la fecundidad por vía clonal (p. ej. $a_{24}=2$, indica que los adultos producen en promedio 2 vástagos clonales de tipo Juveniles I de t a $t+1$). La columna n_t del cuadro representa el vector de densidades poblacionales, que indica el número de individuos presentes en cada categoría de estado en un tiempo t . El vector **w** (estructura estable de edades) representa la proporción de individuos que se encuentra en cada categoría de estado en una población que ha alcanzado la estructura estable y **v** (valor reproductivo) representa el número de hijos que vale cada individuo en la categoría de estado i .

Popbio es un paquete para la construcción de modelos matriciales. El paquete consiste en primer lugar en una traducción al **R** del código de **Matlab** que se encuentra en Caswell (2001) o en Morris y Doak (2002). La lista de funciones que se convirtieron de cada libro se puede desplegar usando la función `help(Caswell)` y `help(Morris)` dentro de **R**, o en las páginas de ayuda. En segundo lugar el paquete **popbio** incluye funciones para estimar las tasas vitales y construye matrices de población a partir de datos crudos que comunmente se toman en los estudios demográficos en plantas. En estos estudios, las tasas vitales por lo general son estimadas directamente de los censos anuales de individuos marcados utilizando tablas de frecuencias de transición. Para estimar las tasas vitales en demografía de animales usando métodos de captura-recaptura use preferencialmente el paquete **Rcapture**, **mra**, o la interfase de **RMark** en lugar de éste. Finalmente, el paquete incluye métodos para graficar y datos de muestra los cuales consisten en matrices de proyección o datos de censos anuales de estudios demográficos publicados. Tres demostraciones ilustran algunas de las capacidades del paquete (`Caswell`, `fillmore` y `stage.classify`). Una descripción del paquete se puede consultar en la revista *Journal of Statistical Software* que se encuentra en <http://www.jstatsoft.org/v22/i11>.

REFERENCIAS

- Para citar el paquete popbio en las publicaciones teclee `citation('popbio')` o ver subsección Instalación de popbio. Para los detalles de los modelos matriciales de población vea:
- Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.
- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

Caswe11 Funciones de Caswell (2001) convertidas de Matlab

DESCRIPCIÓN

Capítulo 2. Modelos clasificados por edad

`pop.projection` Sección 2.2. Proyección de tasas de crecimiento poblacional.

Capítulo 4. Modelos matriciales clasificados por estado

`lambda` sección 4.4. Regresa el eigenvalor dominante

`stable.stage` sección 4.5. Regresa la estructura estable de edades (eigen-vector derecho)

`reproductive.value` sección 4.6. Regresa el valor reproductivo (eigen-vector izquierdo)

`damping.ratio` sección 4.7. Regresa la tasa de amortiguamiento

`eigen.analysis` sección 4.8. Calcula eigenvalores y vectores, incluyendo el eigenvalor dominante, estructura estable de edades, valor reproductivo, tasa de amortiguamiento, sensibilidades y elasticidades. Desde la versión 2.0, también se incluyen como funciones independientes

Capítulo 5. Eventos en el ciclo de vida

`fundamental.matrix` sección 5.3.1. Calcula la sobrevivencia específica por edad en una matriz clasificada por estados usando la matriz fundamental **N**

`net.reproductive.rate` sección 5.3.4. Calcula la tasa neta reproductiva de una matriz clasificada por estados usando el eigenvalor dominante de la matriz **R**

`generation.time` sección 5.3.5. Calcula el tiempo generacional de una matriz clasificada por estados

Las curvas de sobrevivencia y fecundidad específicas en la Figura 5.1 y 5.2 ahora se incluyen en `demo(Caswell)`.

Capítulo 6. Estimación de los parámetros

`projection.matrix` sección 6.1.1. Estima las tasas vitales y construye la matriz de proyección usando tablas de frecuencias de transición

`QPmat` sección 6.2.2. Construye la matriz de proyección de una serie de tiempo de individuos por estado usando el método de programación cuadrática de Wood. Requiere la librería `quadprog`

Capítulo 9. Análisis de sensibilidad

`sensitivity` sección 9.1. Calcula sensibilidades

`elasticity` sección 9.2. Calcula elasticidades.

Vea la función `secder` en el paquete `demogR` package para las segundas derivadas de los eigenvalores descritas en la sección 9.7

Capítulo 10. Experimentos de Respuesta de Tabla de Vida (LTRE)

`LTRE` sección 10.1 and 10.2. Diseños fijos en LTREs. Vea `demo(Caswell)` para descomponer la varianza en un diseño aleatorio (Fig 10.10)

Capítulo 12. Inferencia estadística

`boot.transitions` sección 12.1.4. Remuestrea los datos observados de las transiciones del censo en una de estado-destino (state-fate).

`resample` sección 12.1.5.2. Remuestrea transiciones de una matriz de proyección a partir de una distribución multinomial (y las fecundidades de una distribución log-normal)

Capítulo 14. Estocasticidad ambiental

`stoch.growth.rate` sección 14.3. Calcula la tasa logarítmica de crecimiento por simulación usando la aproximación de Tuljapukar

`stoch.projection` sección 14.5.3. Proyecta la tasa estocástica de crecimiento de una secuencia de matrices en un ambiente uniforme y no uniforme

Vea la función `stoch.sens` del paquete `demogR` para la sensibilidad y elasticidad de la tasa estocástica logarítmica descrita en la sección 14.4.

Capítulo 15. Estocasticidad demográfica

`multiresultm` sección 15.1.3. Incorpora la estocasticidad demográfica a las proyecciones poblacionales. El ejemplo usa el juego de datos `whale dataset` para crear el gráfico como el de la figura 15.3.

Autor(es)

Chris Stubben

Morris Funciones de Matlab convertidas a partir de Morris and Doak (2002)

DESCRIPCIÓN

Capítulo 3

`grizzly` Tabla 3.1. Conteos de población de oso grizzly (Grizzly). El ejemplo incluye código para calcular el promedio, varianza e intervalos de confianza usando regresión y otros procedimientos

`extCDF` Recuadro 3.3. Función de distribución acumulativa del tiempo de extinción basada en conteos

`countCDFxt` Recuadro 3.4. Probabilidades de extinción e intervalos de confianza para conteos

Capítulo 7

`stoch.projection` Recuadro 7.3. Proyecta la tasa de crecimiento estocástica a partir de una secuencia de matrices

`stoch.growth.rate` Recuadro 7.4. Calcula la tasa de crecimiento log-estocástica por la aproximación de Tujapurkar y por simulación

`stoch.quasi.ext` Recuadro 7.5. Estima el umbral de cuasiextinción

Capítulo 8

`Kendall` Recuadro 8.2. Método de Kendall para corregir la variación por muestreo

`betaval` Recuadro 8.3. Genera números al azar con distribución beta

`lnorms` Recuadro 8.4. Genera números al azar con distribución log-normal

`stretchbetaval` Recuadro 8.5. Genera números al azar con distribución beta estirados

`vitalsim` Recuadro 8.10. Calcula la frecuencia de densidad acumulada (CDF) de la tasa estocástica de crecimiento y extinción usando tasas vitales

`multiresultm` Recuadro 8.11. Incorpora estocasticidad demográfica en las proyecciones poblacionales

Capítulo 9

`vitalsens` Recuadro 9.1. Elasticidad y sensibilidad de tasas vitales

aq.census Datos de censo anual para *Aquilegia* en el suroeste de los EUA

DESCRIPCIÓN

Datos de censos demográficos de *Aquilegia chrysantha* en Fillmore Canyon, Organ Mountains, Nuevo México, 1996-2003.

Uso

```
data(aq.census)
```

FORMATO

Un data frame con 2,853 observaciones con las siguientes 8 variables.

```
plot  Número de parcela
year  Año del censo
plant  Identificador de la planta
status Estado de la planta como se registra en campo: muerto (dead), latente (dormant), recluta 0 unicamente con cotiledones (recruit0), recluta1 de tipo 1 sin cotiledones (recruit1), en floración (flowering) o vegetativo (vegetative)
rose  Número total de rosetas
leaf  Número total de hojas
infl  Número total de inflorescencias o escapos
fruits Número total de frutos maduros
```

DETALLES

Estos datos incluyen censos de 10 de las 15 parcelas demográficas establecidas en 1995. Si tiene preguntas contacte los autores de los datos para tener el juego completo de datos de 1995-2006.

FUENTE

- Propietario del juego de datos: Brook Milligan, Chris Stubben, Allan Strand

Ver también

```
aq.trans para transiciones anuales con estado y destino en el mismo renglón
```

Ejemplos

```
data(aq.census)
sv<-table(aq.census$status, aq.census$year)
sv
stage.vector.plot(sv[-1,], prop=FALSE)
```

aq.matrix Crea una matriz de proyección para Aquilegia

DESCRIPCIÓN

Crea una matriz de proyección para *Aquilegia* a partir de datos de transiciones anuales, asumiendo que las semillas nuevas y el banco de semillas tienen la misma probabilidad de éxito en la germinación y tasas iguales de sobrevivencia.

Uso

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

ARGUMENTOS

trans	Un data frame con transiciones enlistando estados y destinos ordenados (<code>ordered</code>) y conteos de frutos maduros.
recruits	Número de reclutas observados en el año t_{+1} .
summary	Salida de la matriz de proyección y los resúmenes. Si no es una tabla de transiciones con las fecundidades añadidas.
seed.survival	Tasa de sobrevivencia de semillas para semillas nuevas y del banco de semillas. El default es 12.6% de sobrevivencia.
seed.bank.size	Tamaño estimado del banco de semillas. El banco de semillas y de semillas nuevas contribuyen a un pool común que tiene una misma probabilidad de germinar. El default es de 10,000 semillas en el banco de semillas.
seeds.per.fruit	Número de semillas producida por un fruto maduro. Default es de 120 semillas.
...	argumentos adicionales pasados a <code>projection.matrix</code>

DETALLES

Añade fecundidades individuales a las transiciones anuales usando un censo prereproductivo.

VALOR

Si el resumen (`summary`) es `TRUE`, genera una lista con:

recruits	número total de reclutas
seed.survival	tasa de sobrevivencia de semillas
seed.bank	número total de semillas en el banco de semillas
seeds.from.plants	número de semillas nuevas liberadas recientemente de los frutos
recruitment.rate	tasa de reclutamiento calculado como $\text{reclutas} / (\text{tamaño.banco.semillas} + \text{semillas.de.plantas})$
A	matriz de proyección
lambda	tasa de crecimiento poblacional
n	vector inicial de la población
n1	vector final de la población

Si el resumen (summary) es FALSE, regresa un data frame con fecundidades individuales añadidas únicamente al data frame de transiciones.

AUTOR(ES)

Chris Stubben

VER TAMBIÉN

`projection.matrix`

EJEMPLOS

```
data(aq.trans)

x<-subset(aq.trans, year==1996)

## numero de reclutas en 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 Datos de transiciones para *Aquilegia* en el soroeste de los EUA

DESCRIPCIÓN

Datos de transiciones enlistando estados y destinos de *Aquilegia chrysantha* en Fillmore Canyon, Organ Mountains, Nuevo México, 1996-2003.

Uso

```
data(aq.trans)
```

FORMATO

Un data frame con 1,637 observaciones de las siguientes 9 variables.

```
plot  Número de parcela (plot)
year  Año del inicio del censo (year)
plant  Etiqueta de la planta (plant)
stage  Estado inicial (state) con niveles del factor ordenados seed <recruit
       <small <large <flower.
leaf  Número total de hojas (leaf)
rose  Número total de rosetas (rose)
fruits  Número total de frutos maduros (fruit)
fate  Estado final (fate) o destino con niveles seed <recruit <small
       <large <flower <dead
rose2  Número final de rosetas (rose2)
```

DETALLES

Los primeros 5 estados incluyen semillas en el banco de semillas, nuevos reclutas, plántulas plantas vegetativas con una roseta, plantas grandes vegetativas con 2 o más rosetas y plantas en flor. Las clases de estado se asignaron a cada planta usando una combinación de estado y tamaño registrado en campo. Vea `demo(stage.classify)` para detalles de como categorizar.

FUENTE

- Propietario del juego de datos: Brook Milligan, Chris Stubben, Allan Strand

VER TAMBIÉN

```
aq.census
```

EJEMPLOS

```
data(aq.trans)
head(aq.trans, 3)
sv<-table(aq.trans$stage, aq.trans$year)
```

```

addmargins(sv)
stage.vector.plot(sv[-1,], prop=FALSE, main="Aquilegia stage
vectors")

## grafica de porporciones con barras usando barplot
## use xpd para dibujar la leyenda por fuera de los margenes del
grafico
op<-par(mar=c(5,4,4,1), xpd=TRUE)
x<-barplot(prop.table(sv[-1,],2), las=1,
xlab="A~no", ylab="Proporcion en clase de estado",
col=rainbow(4), ylim=c(0,1), xaxt='n', space=.5)
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 Genera números al azar de una distribución beta

DESCRIPCIÓN

Esta función calcula un número al azar de una distribución beta y usa la función `pbeta(x, vv, ww)` de R.

Uso

```
betaval(mn, sdev, fx=runif(1))
```

ARGUMENTOS

<code>mn</code>	Tasa promedio entre 0 y 1
<code>sdev</code>	Desviación estandar
<code>fx</code>	Función distribución acumulada, el default es un número al azar entre 0 y 1

DETALLES

Esta función es utilizada por `vitalsim`.

VALOR

Regresa un valor de beta al azar

AUTOR(ES)

Código original de MATLAB por Morris y Doak (2002: 277- 278), adaptado a R por Patrick Nantel, 20 Junio 2005.

FUENTE

- Código de Matlab convertido del Recuadro 8.3 en Morris and Doak (2002)

REFERENCIAS

- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

Beta Distribution `rbeta`

EJEMPLOS

```
betaval(.5, sd=.05)
betaval(.5, sd=.05)
## histograma con media=0.5 y ds=0.05
x<-numeric(100)
```



```

for (i in 1:100)
{
  x[i]<-betaval(.5,.05)
}
hist(x, seq(0,1,.025), col="green", ylim=c(0,25), xlab="Valor",
main="Distribucion beta con media=0.5 y ds=0.05")

# genera una grafica similar a la Figura 8.2 en (2002:264)
# una version mucho mas sencilla de BetaDemo esta en el Recuadro 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(0,0.4), type='n', ylab='Frecuencia',
xlab='Valor', main="Ejemplos de distribuciones beta")
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(.6,.4, c("(0.50, 0.05)", "(0.50, 0.25)", "(0.50, 0.45)"),
lty=1:3, lwd=2, col=2:4, title="media y ds")

```

boot.transitions Remuestrea transiciones observadas de los censos

DESCRIPCIÓN

Calcula la distribución de un remuestreo de las tasas de crecimiento (λ), vectores de estado, y elementos de la matriz de proyección muestreando transiciones observadas de un data frame de estado-destino con reemplazo

Uso

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

ARGUMENTOS

<code>transitions</code>	Un data frame de estado-destino con el estado o la categoría de edad en el censo actual, destino en los censos subsecuentes y uno o más columnas de fecundidad
<code>iterations</code>	Número de iteraciones para remuestrear
<code>by.stage.counts</code>	Remuestrea transiciones con igual probabilidad (default) o con subconjuntos de los conteos iniciales
<code>...</code>	Opciones adicionales pasadas a <code>projection.matrix</code>

VALOR

Una lista con 3 elementos

<code>lambda</code>	Un vector que contiene los valores de λ remuestreados
<code>matrix</code>	Una matriz que contiene las matrices de transición remuestreadas, siendo una matriz de transición por renglón
<code>vector</code>	Una matriz que contiene los vectores de estado, siendo un vector de estado por renglón

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Ver Morris y Doak (2005) en <http://esapubs.org/Archive/mono/M075/004/appendix-A.htm> para remuestrear conteos por estado

VER TAMBIÉN

`projection.matrix`

EJEMPLOS

```
data(test.census)

## crea un data frame de estado-destino usando las funciones merge y
##subset
trans01 <- subset(
  merge(test.census, test.census, by="plant", sort=FALSE),
  year.x==2001 & year.y==2002)

## Pone formato y nombre a las columnas y renglones
trans01<-trans01[,c(1:4,6)]
colnames(trans01)[2:5] <- c("Year", "State", "fruits", "Fate")
rownames(trans01) <- 1:nrow(trans01)
# order stage columns corresponding to matrix
trans01$stage <- ordered(trans01$stage,
  levels = c("seedling", "vegetative", "reproductive"))

## añade fecundidades individuales usando un censo prereproductivo
## sin banco de semillas basada en la produccion reproductiva
## proporcional de plantas en floracion y el numero total de
## semillas al final del intervalo de proyeccion

seedlings<-nrow(subset(test.census, year==2002 & stage=="seedling"))
trans01$seedling<-trans01$fruits/sum(trans01$fruits) * seedlings
trans01

## Instrucciones paso a paso para remuestrear el data frame
n<-nrow(trans01)
n
set.seed(77)
x <- sample(n, replace=TRUE)
x
bt<-trans01[x,]
bt
projection.matrix(bt)

## o resmuestraa por el conteo de clases de estado
lapply(split(trans01, trans01$stage, drop=TRUE),
  function(x) x[sample(nrow(x), replace=TRUE),])

## usando boot.transitions
boot.transitions(trans01, 5)
boot.transitions(trans01, 5, by.stage=TRUE)

## Ejemplo de Aquilegia
data(aq.trans)
x<-subset(aq.trans, year==1996)
# calcula lambda, sobrevivencia de semilla y tasa de reclutamiento
# usando aq.matrix
```

```

rec<-nrow(subset(aq.trans, year==1997 & stage == "recruit"))
aq.96<- aq.matrix(x, rec)
# a~nade fecundidades individuales unicamente al data frame
aq.96.trans<-aq.matrix(x, rec, summary=FALSE)
# pasa las transiciones estimadas en aq.96 a la matriz de proyeccion
aq.96.boot<-boot.transitions(aq.96.trans, 200,
    add=c(1,1, aq.96$seed.survival, 2,1, aq.96$recruitment.rate) )
# calcula percentiles de los valores de lambda remuestras
    dos usando quantile()
ci<- quantile(aq.96.boot$lambda, c(0.025,0.975) )
aq.96$lambda
ci
# grafica el histograma de frecuencias de lambda
hist(aq.96.boot$lambda, col="green", xlab="Lambda",
main=paste('Estimadores de bootstrap de las tasas de crecimiento
pobacional de 1996-1997'))
abline(v=ci, lty=3)

```

calathea **Matrices de proyección de una hierba tropical del sotobosque**

DESCRIPCIÓN

Matrices de proyección para la hierba tropical (*Calathea ovandensis*) de parcelas 1-4 y años 1982-1985 y la matriz agrupada

Uso

```
data(calathea)
```

FORMATO

Una lista de 17 matrices ordenada por parcela y despues por año, con la matriz agrupada al final.

DETALLES

Una matriz de proyección construida usando censo post-reproductivo con 8 categorías de tamaño: semilla (seed), plántula (seedling), juvenil (juvenile), pre-reproductivo (pre-reproductive), y 4 clases reproductivas divididas por cobertura.

FUENTE

- Tabla 7 en Horvitz y Schemske (1995). La matriz agrupada es de la Tabla 8.

REFERENCIAS

- Horvitz, C.C. and D.W. Schemske. 1995. Spatiotemporal variation in demographic transitions of a tropical understory herb: Projection matrix analysis. Ecological Monographs 65:155-192.

EJEMPLOS

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

## Matriz promedio (excluye la matriz agrupada)
mean(calathea[-17])

## Toda la parcela 1
calathea[1:4]
## Todas las matrices de 1982
calathea[ grep("1982", names(calathea)) ]
```

```

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

## Tasas de crecimiento -Vea Figura 7
x<-sapply(calathea[-17], lambda)
x<-matrix(x, nrow=4, byrow=TRUE,
dimnames= list(paste("Parcela", 1:4), 1982:1985))
x
matplot2(x, type='b', ylab='tasa de Crecimiento',
main='Tasa de Crecimiento de Calathea')

```

colorguide Grafica una simple guía de colores

DESCRIPCIÓN

Gráfica una guía de colores con bloques de colores y nombres arreglados verticalmente

Uso

```
colorguide(col, main = "", border = FALSE)
```

ARGUMENTOS

`col` Un vector de colores
`main` Título
`border` Tipo de borde alrededor de los rectángulos

VALOR

Un gráfico de colores y nombres

AUTOR(ES)

Chris Stubben

EJEMPLOS

```
op<-par(mfrow=c(2,2))
colorguide(palette(), "Palette colors")
## 657 colores incluidos
## ROJO
reds<-grep("red", colors(), value=TRUE)
## sorteados alfabeticamente
colorguide(reds, "Reds sorted alphabetically")
# VERDE
greens<-grep("green", colors(), value=TRUE)
RGBColors <- col2rgb(greens)
RGBOrder <- order( RGBColors[2,], RGBColors[3,], RGBColors[1,] )
colorguide(greens[RGBOrder][1:30], "Verdes sorteados por RGB")
## AZULES CLARO
colorguide(grep("light.*blue", colors(), value=TRUE) , "Azul claro")

## Funciones
colorguide(rainbow(16, end=.7) , "Colores arcoirisRainbow colors")
colorguide(heat.colors(16) , "Heat.colors")
```

```
## colorRampPalette
jet.colors = colorRampPalette(c("#00007F", "blue", "#007FFF", "cyan",
"#7FFF7F", "yellow", "#FF7F00", "red", "#7F0000"))
colorguide(rev(jet.colors(16)), "Jet colors from Matlab")
blue2red<-colorRampPalette(c('blue','lightyellow','red'))
colorguide(blue2red(16), "Blue to Red colors")

par(op)
```


countCDFxt Probabilidades de extinción basadas en conteos e intervalos de confianza por remuestreo

DESCRIPCIÓN

Esta función toma los parámetros de los conteos poblacionales y calcula la probabilidad de extinción con intervalos de confianza por remuestreo para un modelo denso-independiente usando una aproximación de difusión.

Uso

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

ARGUMENTOS

<code>mu</code>	Valor estimado del promedio μ
<code>sig2</code>	Valor estimado de la varianza de la muestra
<code>nt</code>	Número de transiciones en el juego de datos
<code>Nc</code>	Tamaño poblacional actual
<code>Ne</code>	Umbral de cuasi-extinción
<code>tq</code>	Duración del censo (en años), por default es el número de transiciones
<code>tmax</code>	Tiempo para calcular la probabilidad de extinción (default=50)
<code>Nboot</code>	Número de muestreos con reemplazo para calcular los intervalos de confianza para las probabilidades de extinción (default=500)
<code>plot</code>	Dibuja el gráfico de tiempo de extinción (CDF) con una escala logarítmica en el eje las ordenadas (y)

VALOR

La función grafica la probabilidad acumulada de cuasi-extinción en el tiempo con sus intervalos de confianza al 95%. También da un data frame con el tiempo de extinción (CDF) para los mejores parámetros (Gbest), al igual que los intervalos de confianza superior e inferior para las probabilidades de extinción (Glo, Gup).

AUTOR(ES)

Adaptado a R por Patrick Nantel, 4 Mayo 2005, del programa 'extprob' de Morris y Doak (2002: 79-86)

FUENTE

- Código de Matlab convertido del Recuadro 3.4 en Morris y Doak (2002)

REFERENCIAS

- Dennis *et al.* 1991, Ecological Monographs 61: 115-143.
- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

`extCDF`

EJEMPLOS

```
## Grafica como la Figura 3.8 en Morris y Doak (2002).  
data(grizzly)  
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 Razón de Amortiguamiento

DESCRIPCIÓN

Calcula la razón de amortiguamiento de una matriz de proyección

Uso

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

ARGUMENTOS

A Una matriz de proyección

DETALLES

Vea la seccion 4.7 en Caswell (2001).

VALOR

Razón de amortiguamiento (Damping Ratio)

NOTAS

La razón de amortiguamiento se calcula dividiendo el eigenvalor dominante entre el segundo eigenvalor de mayor magnitud.

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

lambda

EJEMPLOS

```
## las ballenas convergen lentamente a una estructura estable
data(whale)
matplot2(pop.projection(whale, c(1,1,1,1), 60)$stage.vectors,
prop=TRUE, legend=NA,
main=paste("razon de amortiguamiento de la ballena = ",
           round(damping.ratio(whale),3) ) )

# Calathea - compare con Tabla 12 en Horvitz y Schemske (1995)
data(calathea)
```

```
x<-sapply(calathea[-17], damping.ratio)
x<-matrix(x, nrow=4, byrow=TRUE, dimnames= list(paste("parcela", 1:4),
1982:1985))
x
matplot2(x, type='b', ylab="razon de amortiguamiento", main="Calathea")
```

eigen.analysis Análisis de eigenvalores y eigenvectores de una matriz de proyección

DESCRIPCIÓN

Calcula la tasa de crecimiento y otros parámetros demográficos a partir de una matriz de proyección utilizando álgebra de matrices

Uso

```
eigen.analysis(A, zero=TRUE)
```

ARGUMENTOS

A	Una Matriz de proyección
zero	Ajusta las sensibilidades de transiciones no observadas a cero

DETALLES

El cálculo de eigenvalores y eigenvectores parcialmente sigue el código de Matlab en la sección 4.8.1 (p. 107) en Caswell (2001). A partir de popbio versión 2.0, cada parte de eigen.analysis se incluye ahora como una función separada.

VALOR

Una lista con 6 elementos

lambda1	Eigenvalor dominante con el valor real más alto
stable.stage	Estructura estable proporcional
sensitivities	Matriz de sensibilidades
elasticities	Matriz de elasticidades
repro.value	Valor reproductivo escalado, de tal manera que el valor de la primera categoría $v[1]=1$
damping.ratio	razón de amortiguamiento

NOTAS

Si la matriz A es singular, entonces eigen.analysis regresa elasticidades, sensibilidades y valores reproductivos con NA's.

Esta función está incluida en el paquete demogR.

AUTOR(ES)

Código original por James Holland Jones, Stanford University, Department of Anthropological Sciences, 12 agosto 2005 en http://popstudies.stanford.edu/summer_course/

REFERENCIAS

- Caswell, H. 2001. Matrix population models: construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

eigen and pop.projection

EJEMPLOS

```
## Matriz no primitiva
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]
## razon de amortiguamiento es NA
eigen.analysis(A)
## Ciclo de cada 3 a~nos
stage.vector.plot(pop.projection(A, c(1,1,1), 10)$stage.vectors)

### Teasel
data(teasel)
a<-eigen.analysis(teasel)
a
barplot(a$stable.stage, col="green", ylim=c(0,1),
ylab="Porpocion en estructura estable", xlab="Clase de estado",
      main="Teasel")
box()

op<-par(mfrow=c(2,2))
image2(teasel, cex=.8, mar=c(0.5,3,4,1) )
title("Matriz de proyeccion de Teasel", line=3)

image2(a$elasticities, cex=.8, mar=c(0.5,3,4,1) )
title("Matriz de elasticidad", line=3)

## default es la sensibilidad para los elementos no cero de la matriz
image2(a$sensitivities, cex=.8, mar=c(0.5,3,4,1) )
title("Matriz de sensibilidad 1", line=3)

## use zero=FALSE para tener las sensibilidades de todos los elementos
## de la matriz
image2(eigen.analysis(teasel, zero=FALSE)$sensitivities,
cex=.8, mar=c(0.5,3,4,1) )
title("Matriz de sensibilidad 2", line=3)
par(op)
```

elasticity **Análisis de elasticidad de una matriz de proyección**

DESCRIPCIÓN

Calcula las elasticidades de los eigenvalores a cambios en los elementos de la matriz de proyección

Uso

```
elasticity(A)
```

ARGUMENTOS

A Una matriz de proyección

DETALLES

Vea seccion 9.2 en Caswell (2001).

VALOR

Una matriz de elasticidad

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

```
sensitivity
```

EJEMPLOS

```
data(teasel)
elas<-elasticity(teasel)
image2(elas, mar=c(1,3.5,5,1) )
  title("Matriz de elasticidad de Teasel", line=2.5)

## Elasticidades sumadas para teasel.
## Fertilidad en la ultima columna, estasis P en la diagonal y
## crecimiento en la subdiagonal inferior
c(F=sum(elas[,6]), P=sum(diag(elas)), G=sum(elas[row(elas)>col(elas)]))
data(tortoise)
elas<-elasticity(tortoise[["med.high"]])
image2(elas, mar=c(1,3.5,5,1), log=FALSE)
```

```

    title("Matriz de elasticidad de tortuga", line=2.5)
## Suma las elasticidades para la tortuga (Vea el ejemplo 9.4)
## Fecundidad en el primer renglon, estasis o sobrevivencia en la
## diagonal principal y crecimiento en la subdiagonal

c(F=sum(elas[1,]), P=sum(diag(elas)), G=sum(elas[row(elas)=col(elas)+1]))

```


extCDF Función de distribución acumulada del tiempo de extinción para conteos

DESCRIPCIÓN

Regresa la función de distribución acumulada del tiempo de extinción usando los parametros derivados de los conteos poblacionales.

Uso

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

ARGUMENTOS

mu	Valor estimado del promedio mu
sig2	Valor estimado de la varianza de la muestra
Nc	Tamaño de población actual
Ne	Umbral de cuasi-extinción
tmax	Tiempo para calcular la probabilidad de extinción, (default=50)

VALOR

Un vector con las probabilidades acumuladas de cuasi-extinción de t=0 a t=t-max.

AUTOR(ES)

Chris Stubben

FUENTE

- Código de Matlab convertido del Recuadro 3.3 y ecuación 3.5 en Morris y Doak (2002)

REFERENCIAS

- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

countCDFxt para intervalos de confianza por remuestreo

EJEMPLOS

```
data(grizzly)
logN<-log(grizzly$N[-1]/grizzly$N[-39])
mu<-mean(logN)
sig2<-var(logN)
## grizzly cdf (escala log)
```

```

ex<-extCDF(mu, sig2, Nc=99, Ne=20)
plot(ex, log='y', type='l', pch=16, col="blue", yaxt='n',
xlab="A~nos", ylab="Probabilidad de Cuasi-extincion",
main="Osos grizzly de Yellowstone")
pwrs<-seq(-15,-5,5)
axis(2, at = 10^pwrs, labels=parse(text=paste("10^", pwrs,
sep = "")),las=1)
##grafica como Figura 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="Tama~no poblacional actual",
ylab="Probabilidad de Cuasi-extincion para el a~no 50")

```

fundamental.matrix Matriz fundamental y sobrevivencia edad específica

DESCRIPCIÓN

Cálculos de sobrevivencia de edad específica de matrices clasificadas por estados. Incluye el promedio, varianza y coeficiente de variación (cv) del tiempo de permanencia en cada categoría, y el promedio y varianza del tiempo que tarda en morir

Uso

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

ARGUMENTOS

A	Una matriz de proyección
...	Elementos adicionales pasados a <code>splitA</code> que son usados para dividir A en las matrices T de transiciones y F de fecundidades

DETALLES

Vea sección 5.3.1 en Caswell (2001).

VALOR

Una lista con 5 elementos

N	Matriz fundamental o tiempo promedio de permanencia en cada clase de estado
var	Varianza del tiempo de permanencia en cada categoría de estado
cv	Coeficiente de variación (sd/promedio)
meaneta	Tiempo promedio a la muerte
vareta	Varianza del tiempo a la muerte

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

Vease `generation.time` y `net.reproductive.rate` para otros atributos específicos de edad

EJEMPLOS

```
data(whale)  
fundamental.matrix(whale)
```

generation.time Tiempo Generacional

DESCRIPCIÓN

Calcula el tiempo generacional de una matriz clasificada por estados

Uso

```
generation.time(A, ...)
```

ARGUMENTOS

A	Una matriz de proyección
...	Elementos adicionales pasados a <code>splitA</code> que son utilizados para separar A en la matriz de transiciones T y de fecundidades F

DETALLES

Vease seccion 5.3.5 en Caswell (2001).

VALOR

Tiempo generacional. Si la matriz de transición es singular, entonces regresa un valor de NA

NOTAS

Las versiones anteriores de `popbio` requerian de matrices T y F separadas como insumo

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

Vease `fundamental.matrix` y `net.reproductive.rate` para otros atributos edad-específicos

EJEMPLOS

```
data(whale)
generation.time(whale)
## fecundidades en la ultima columna
data(teasel)
generation.time(teasel, r=1:6, c=6)
## Parcela 3 de Calathea
data(calathea)
sapply(calathea[9:12], generation.time)
```

grizzly Tamaños de población de oso grizzly en el Parque Yellowstone entre 1959-1997

DESCRIPCIÓN

Número estimado de hembras adultas de oso grizzly en la población de Yellowstone entre 1959-1997.

Uso

```
data(grizzly)
```

FORMATO

Un data frame con 39 observaciones con las siguientes 2 variables.

year	Año del censo (year)
N	Número estimado de hembras de oso grizzly (N)

DETALLES

Este conjunto de datos de oso grizzly es usado para generar análisis de viabilidad poblacional (PVA) de conteos en el capítulo 3 de Morris y Doak (2002).

FUENTE

- Tabla 3.1 en Morris y Doak (2002). El conjunto de datos original es de Eberhardt *et al.* (1986) y Haroldson (1999). Detalles adicionales pueden consultarse en Interagency Grizzly Bear Study Team <http://nrmssc.usgs.gov/research/igbst-home.htm>.

REFERENCIAS

- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
data(grizzly)
attach(grizzly)
## Grafica como Figura 3.6 (pagina 66)
plot(year, N, type='o', pch=16, las=1, xlab="a~no",
      ylab="Hembras adultas", main="Osos grizzly de Yellowstone")
## calcula log(Nt+1/Nt)
nt<-length(N) ## numero de transiciones
logN<-log(N[-1]/N[-nt])
## Media y varianza
c(mean=mean(logN), var=var(logN))
## o usando regresion lineal
## transformacion para varianzas diferentes (p. 68)
```

```

x<-sqrt(year[-1]-year[-length(year)])
y<-logN/x
mod<-lm(y~0 + x )
## Grafica como Figura 3.7
plot(x,y, xlim=c(0,1.2), ylim=c(-.3,.3), pch=16, las=1,
xlab=expression((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("Estimando ", mu, "y", sigma^2,
"usando regresion")))
abline(mod)
## Promedio (pendiente)
mu<- coef(mod)
## Varianza (Cuadrado medio de la tabla del Analisis de varianza)
sig2<-anova(mod)[["Mean Sq"]][2]
c(mean= mu , var= sig2)
## Intervalo de confianza para el promedio (pagina 72)
confint(mod,1)
## Intervalo de confianza para sigma 2 (ecuacion 3.13)
df1<-length(logN)-1
df1*sig2 /qchisq(c(.975, .025), df= df1)
## prueba para valores extremos usando dffits (p.74)
dffits(mod)[dffits(mod)> 2*sqrt(1/38) ]
## Grafica como Figura 3.11
plot(N[-nt], logN, pch=16, xlim=c(20,100), ylim=c(-.3, .3),las=1,
xlab="Numero de hembras en T",
ylab=expression(log(N[t+1]/N[t])),
main="Log tasa de crecimiento poblacional de oso Grizzly")
cor(N[-nt], logN)
abline(lm(logN ~ N[-nt]), lty=3 )
detach(grizzly)

```


head2 Regresa la primera y última parte de una matriz o data frame

DESCRIPCIÓN

Regresa el primer y último renglón de la salida de `head` y `tail` y separa los dos pedazos. Es muy útil cuando se tienen juegos de datos muy grandes o cuando quieres conocer como es la estructura de tus datos.

Uso

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

ARGUMENTOS

<code>x</code>	Una matriz o data frame
<code>head</code>	El número de primeros renglones
<code>tail</code>	El número de últimos renglones
<code>dotrows</code>	El número de renglones de puntos

VALOR

Un objeto más pequeño que `x` únicamente con el primer y último renglón

AUTOR(ES)

Chris Stubben

EJEMPLOS

```
data(aq.trans)
head2(aq.trans)
```

hudcorrs

Matrices de correlación para las tasas vitales de Hudsonia

DESCRIPCIÓN

Matrices de correlación intra e inter anual de las tasas vitales de Hudsonia montana. Las correlaciones se calcularon únicamente usando las primeras 13 tasas de crecimiento y sobrevivencia ya que las tasas de fecundidad varían poco.

Uso

```
data(hudcorrs)
```

FORMATO

Una lista con 2 matrices de correlación, `corrin` (correlación intra anual) y `corrout` (correlación inter anual).

AUTOR(ES)

Conjunto de datos original de Morris y Doak (2002)

FUENTE

- Las matrices de correlación en <http://www.sinauer.com/PVA/hudcorrs.mat> incluyen algunas correlaciones >1. Un juego corregido de correlaciones fue enviado por D. Doak el 8/4/2007.

REFERENCIAS

- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

`vitalsim`

EJEMPLOS

```
data(hudcorrs)
hudcorrs$corrin
```

DESCRIPCIÓN

Crea una matriz de proyección a partir de las tasas vitales (sobrevivencia, crecimiento y reproducción) de *Hudsonia*. Las tasas de crecimiento se definen como un juego de opciones binomiales como lo que se describe en la Tabla 8.4 B de Morris y Doak (2002).

Uso

```
hudmxdef(vrs)
```

ARGUMENTOS

`vrs` Media de las tasas vitales en `hudvrs`

VALOR

Una matriz de proyección

AUTOR(ES)

Código original en Matlab por Morris y Doak (2002)

FUENTE

- <http://www.sinauer.com/PVA/hudmxdef.m>

REFERENCIAS

- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

`vitalsim`

EJEMPLOS

```
data(hudvrs)
hudmxdef(hudvrs$mean)
```

hudsonia Matrices de proyección para una planta de montaña

DESCRIPCIÓN

Matrices de proyección de la planta (*Hudsonia montana*) para los años 1985 a 1988

Uso

```
data(hudsonia)
```

FORMATO

Una lista de 4 matrices de 1985-1988

DETALLES

Una matriz de proyección con 6 categorías de tamaño: semillas (seeds), plántula (seedlings), y 4 categorías de tamaño divididas por cobertura.

FUENTE

- Table 6.7 in Morris and Doak (2002). Los datos originales son de Frost (1990).
- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
data(hudsonia)
sapply(hudsonia, lambda)

## matriz promedio
x<-mean(hudsonia)
image2(x, mar=c(1,4,5.5,1))
title("Hudsonia matriz promedio", line=2.5)
lambda(x)
# varianza
var2(hudsonia)
```

hudvrs Estimadores de Kendall para la media y varianza de las tasas vitales de Hudsonia

DESCRIPCIÓN

Estimadores de Kendall para las tasas vitales (9 de crecimiento, 4 de supervivencia, y 11 de fertilidad) para *Hudsonia montana*.

Uso

```
data(hudvrs)
```

FORMATO

Un data frame con 24 observaciones de las siguientes 2 variables.

<code>mean</code>	Promedio de las tasas vitales
<code>var</code>	Varianza de las tasas vitales

FUENTE

Datos enlistados en el Recuadro 8.10 para el código de la función `vitalsim`. Vease también la Tabla 8.5 en Morris y Doak (2002).

REFERENCIAS

- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
data(hudvrs)
hudvrs

hudmxdef(hudvrs$mean)
```

image2 Despliega una imagen de una matriz

DESCRIPCIÓN

Crea un cuadrícula de rectángulos para desplegar la proyección, elasticidad, sensibilidad u otra matriz.

Uso

```
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)
```

ARGUMENTOS

x	Una matriz numérica con renglones y columnas
col	Un vector de colores para las cajas
breaks	Un vector numérico de los puntos límites o el número de intervalos en el cual x va a ser subdividido (<code>cut</code>). El default es de longitud <code>col</code>
log	Corta las variables en x usando una escala logarítmica, default= TRUE
border	El color del borde de las cajas, el default es sin bordes
box.offset	Porcentaje de reducción del tamaño de la caja (un número entre 0 y 1), default es 10% de reducción
round	Número de puntos decimales de los valores de x que se van a desplegar en cada caja
cex	Tamaño de la ampliación del texto y las etiquetas, si se especifica esto reemplaza los valores de <code>text.cex</code> y <code>label.cex</code>
text.cex	Tamaño de la ampliación del texto únicamente en las celdas
text.col	Color del texto en las celdas, use NA para evitar las etiquetas del texto
mar	Márgenes en los cuatro lados del gráfico
labels	Un vector dandolos lados del gráfico (1=inferior, 2=izquierda, 3=superior, 4=derecha) para las etiquetas de los renglones y las columnas
label.offset	Cantidad de espacio entre etiquetas y cajas
label.ce	Aumento en el tamaño de las etiquetas

DETALLES

El valor mínimo de x usualmente es asignado a la primera categoria de color y el resto de los valores son asignadas en intervalos iguales. Esto se añadió para mostrar transiciones con valores de probabilidad bajos con otro color, por ejemplo, 2e-06 por lo general se agruparía con el 0 cuando `image`. Nota, si todos los elementos son > 0 , entonces el primer color no sera utilizado.

VALOR

Una imagen de la matriz en x

AUTOR(ES)

Chris Stubben

VER TAMBIÉN

[image](#)

EJEMPLOS

```
data(calathea)

A<-calathea[[11]]

op<-par(mfrow=c(2,2))
image2(A, text.cex=.8)
## con borde gris y etiquetas en el margen inferior izquierdo
image2(A, text.cex=.8, border="gray70", labels=c(1,4), mar=c(3,1,1,3))
## sin texto ni texto
image2( A, box.offset=0, text.col=NA)
# asigna ceros a NA para imprimir todo menos cero
A[A==0]<-NA
image2( A, box.offset=0 , text.cex=.8)

## si se comparan 2 o mas matrices, se obtiene el rango base log10
## de valores (sin incluir el cero) y los pasa a secciones
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 Encuentra la mejor estimación de Kendall de la media y varianza ambiental para tasas vitales con distribución beta-binomial.

DESCRIPCIÓN

Esta función encuentra la mejor estimación de la media y varianza ambiental para tasas vitales con distribución beta-binomial, usando una búsqueda de fuerza bruta para el mejor ajuste de los estimados a partir de un número muy grande de combinaciones de posibles valores de media y varianza.

Uso

```
Kendall(rates, grades=1000, maxvar=0.2, minvar=0.00001, maxmean=1, minmean=0.01)
```

ARGUMENTOS

rates	Una matriz o data frame con cuatro columnas: Identificador de la tasa (Rate identifier), Año (Year), Número total de individuos al empezar, Número de individuos que crecen o se mueren.
grades	Número de diferentes niveles del promedio y varianzas que se van a probar (default es de 1000)
maxvar	Varianza máxima a buscar, el default es de 0.20. El máximo posible es de 0.25, y si se busca en un rango más pequeño mejora la exactitud de la respuesta
minvar	Varianza mínima a buscar, el valor de default es 0.00001
maxmean	Límite superior de los valores promedio a buscar, el default es de 1
minmean	Límite mínimo de los valores promedio a buscar, el valor de default es de 0.01

VALOR

Una lista de los estimados y los intervalos de confianza

est	Una matriz de 5 columnas: (1) Promedio estimado (estimated mean), (2) Promedio por máxima verosimilitud de Kendall (Kendall's MLE mean), (3) Varianza estimada (estimated variance), (4) Varianza por máxima verosimilitud de Kendall (Kendall's MLE variance), (5) Varianza no sesgada por máxima verosimilitud de Kendall (Kendall's unbiased MLE variance)
ci	Una matriz con intervalos de confianza del 95% (confidence limits) para el promedio y varianza estimada sin sesgo con 4 columnas: (1) límite de la media inferior (low) y (3) superior (high), (3) límite de la varianza inferior (low) y (4) superior (high)

NOTAS

Nota: Puede presentarse el mensaje: 'no finite arguments to min; returning Inf', lo cual indica el uso de valores muy bajos de varianza. No es un error del programa.

AUTOR(ES)

Adaptado a R de Morris y Doak (2002: 267-270) por Patrick Nantel.

FUENTE

- Código de Matlab convertido del Recuadro 8.2 en Morris y Doak (2002)

REFERENCIAS

- Kendall, B. E. 1998. Estimating the magnitude of environmental stochasticity in survivorship data. *Ecological Applications* 8(1): 184-193.
- Morris, W. F., and D. F. Doak. 2002. *Quantitative conservation biology: Theory and practice of population viability analysis*. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

varEst

EJEMPLOS

```
## Tortuga del desierto del Recuadro 8.2 - compare el resultado con
## la Tabla 8.3
tor<-data.frame(rate=rep(c("g4","g5","g6"), each=3),
year=rep(1:3,3), ## representa los 70, principio de los 80,
## y finales de los 80
start=c(17,15,7,22,19,4,32,31,10),
grow=c(8,1,0,5,5,0,2,1,0))
## use menor graduacion para incrementar la velocidad del loop
tor.est<-Kendall(tor, grades=200)
tor.est

data(woodpecker)
wp.est <- Kendall(woodpecker, grades=200)
wp.est
```

lambda Tasa finita de crecimiento poblacional

DESCRIPCIÓN

Calcula la tasa finita de crecimiento de la población a partir de una matriz de proyección

Uso

`lambda (A)`

ARGUMENTOS

`A` Una matriz de proyección

DETALLES

Vease seccion 4.4 en Caswell (2001)

VALOR

El eigenvalor dominante

NOTAS

El código en la función `eigen` regresa eigenvalores en orden decreciente o el módulo. El eigenvalor dominante de la matriz no primitiva con `d` eigenvalores de módulo igual es el del valor real mayor (`which.max(Re(eigen(A)$values))`).

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

`eigen` y `pop.projection`

EJEMPLOS

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

data(hudsonia)
sapply(hudsonia, lambda)
```

lnorms Genera números al azar de una distribución lognormal para las tasas de fertilidad

DESCRIPCIÓN

Convierte valores de una distribución normal a valores de una distribución log normal con media y varianza definida

Uso

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

ARGUMENTOS

n	Número de observaciones
mean	Valor promedio de la tasa de fertilidad
var	Varianza de la tasa vital (no la desviación estandar)

VALOR

Un vector de valores al azar log-normales

NOTAS

Esta función probablemente pueda ser reemplazada con funciones internas para la distribución log normal `rlnorm`

AUTOR(ES)

Código original de Matlab por Morris y Doak (2002: 281). Adaptado a R por Patrick Nantel, 20 Junio 2005.

FUENTE

- Código de Matlab convertido del Recuadro 8.4 en Morris y Doak (2002)

REFERENCIAS

- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

`stretchbetaval`

EJEMPLOS

```
lnorms(1)
```

```
## Genera fecundidades al azar con distribucion log-normal para
## una poblacion de 1,000 individuos maduros con fecundidad promedio
## de 3 y varianza en la fecundidad inter individual de 1.5.

rndfert <- lnorms(1000, 3,1.5)
summary(rndfert)
hist(rndfert,40, main="Fertilidades al azar log-norma-
les",
xlab="Tasa de Fertilidad", col="blue")
```

logi.hist.plot Gráfica la regresión logística

DESCRIPCIÓN

Gráfica la combinación de gráficos para regresiones logísticas

Uso

```
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, ...)
```

ARGUMENTOS

independ	Variable independiente o explicatoria
depend	Variable dependiente, normalmente es un vector lógico
logi.mod	Tipo de ajuste, 1=logística; 2=logística "gaussiana"
type	Tipo de representación, "dit"=dit plot; "hist"=histogram
boxp	TRUE=con diagrama de caja, FALSE=sin diagrama de caja
rug	TRUE=con gráficos rug, FALSE=sin gráficos rug
ylabel	Etiquetas del eje de las y
ylabel2	Etiqueta del segundo eje de las y
xlabel	Etiqueta del eje de las x
mainlabel	Título general del gráfico
las.h	Orientación de las etiquetas de los ejes (0=vertical, 1=horizontal)
counts	Añade conteos arriba de las barras del histograma
...	opciones adicionales pasados a logi.hist

VALOR

Gráfico de una regresión logística

NOTAS

Se añadieron opciones para las etiquetas de los ejes

AUTOR(ES)

M. de la Cruz Rot

REFERENCIAS

- de la Cruz Rot, M. 2005. Improving the Presentation of Results of Logistic Regression with R. ESA Bulletin 86:41-48.
- <http://esapubs.org/bulletin/backissues/086-1/bulletinjan2005.htm>

EJEMPLOS

```
data(aq.trans)

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="Probabilidad de sobrevivencia", ylabel2="Numero de plantas",
xlab="Numero de hojas" )

b<-glm(survived ~ leaf, binomial, data=a)
summary(b)
```

LTRE Experimentos de Respuesta de Tabla de Vida (LTRE)

DESCRIPCIÓN

Función para evaluar las sensibilidades de un Experimento de Respuesta de Tabla de Vida con factores fijos (LTRE).

Uso

```
LTRE(trts, ref)
```

ARGUMENTOS

<code>trts</code>	Una matriz del tratamiento o una lista de dos o más matrices de tratamiento
<code>ref</code>	Una matriz de referencia

DETALLES

Las sensibilidad es evaluada entre el tratamiento y la matriz de referencia como se describe en la sección 10.1.1 en Caswell (2001).

VALOR

Una matriz de contribuciones (ecuación 10.4 en Caswell) o una lista de matrices con una matriz de contribuciones por tratamiento

NOTAS

Los ejemplos de un Experimento de Respuesta de Tabla de Vida con factores fijos son de:

- Horvitz, C. C., D. W. Schemske, y H. Caswell. 1997. The relative importance of life-history stages to population growth: prospective and retrospective analyses. Pag. 247-271 en: S. Tuljapurkar y H. Caswell, editores. Structured population models in marine, terrestrial and freshwater systems. Chapman and Hall, New York.
- Angert, A.L. 2006. Demography of central and marginal populations of monkeyflowers (*Mimulus cardinalis* and *M. lewisii*). Ecology 87:2014-2025.
- `Revise demo (Caswell)` para de descomposición de la varianza en un diseño aleatorio usando la ballena asesina.

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
##### Calathea ovandensis
data(calathea)
calathea_pool<-calathea[['pooled']]

## Crea los graficos como los de la FIGURA 7 en Horvitz et al 1997
##Graficos
plots<- split(calathea[-17], rep(1:4,each=4))
## Use la matriz promedio ya que la agrupada no puede utilizarse
## por la funcion plot
plots<- lapply(plots, mean)
Cm<-LTRE(plots, calathea_pool)
pe<-sapply(Cm, sum)
barplot(pe, xlab="Parcela", ylab="Efecto de parcela", ylim=c(-.25, .25),
col="blue", las=1)
abline(h=0)
box()
title(expression(italic("Calathea ovandensis"))))

## Alos -- separa el vector que se recicla
yrs<-split(calathea[-17], 1:4)
yrs <- lapply(yrs, mean)
names(yrs)<-1982:1985
Cm<-LTRE(yrs, calathea_pool)
ye<-sapply(Cm, sum)
barplot(ye, xlab="A~no", ylab="Efecto de A~no", ylim=c(-.25, .25),
col="blue", las=1)
abline(h=0)
box()
title(expression(italic("Calathea ovandensis"))))

## Interaccion
Cm<-LTRE(calathea[-17], calathea_pool)
ie<-sapply(Cm, sum)
## grafico de negativos, efectos de a~no
ie<- ie - rep(pe, each=4) - rep(ye, 4)
names(ie)<-NULL
names(ie)[seq(1,16,4)]<-1:4
barplot(ie, xlab="Parcelas (A~nos 82-83 to 85-86)", ylab="Efecto de la \\
Interaccion", ylim=c(-.25, .25), col="blue", las=1)
abline(h=0)
box()
title(expression(italic("Calathea ovandensis"))))

##### Mimulus
## Matriz de referencia de M. cardinalis dada por 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)

## Efectos poblaciones usando las matrices de poblacion agrupadas
data(monkeyflower)
card<-subset(monkeyflower, species=="cardinalis" & year=="pooled")
## separa los renglones en una lista de 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")

## Grafico como en la Figura 2A en Angert (2006)
op<-par(mar=c(5,5,4,1))
barplot(x, xlab="Poblacion", ylab="", ylim=c(-.4, .4),
xlim=c(0,6.5), las=1, space=.5, col="blue")
abline(h=0)
mtext(expression(paste(sum(a[ij]), " contribuciones")), 2, 3.5)
title(expression(paste(italic("M. cardinalis"), " Efectos pobla-
  cionales"))))
box()

## y grafico como en la Figura 3A en Angert (2006)
x<-matrix(unlist(Cm), nrow=4, byrow=TRUE)
colnames(x)<-paste("a", rep(1:4, each=4), 1:4, sep="")
bp<-barplot(x[1:2,], beside=TRUE, ylim=c(-.2,.2), las=1,
xlab="Transicion", ylab="", xaxt='n')
mtext(expression(paste("Contribucion de", a[ij], "a la variacion
  en", lambda)), 2, 3.5)
## rota las etiqueta
text(bp[1,]-0.5, -.22, labels=colnames(x), srt=45, xpd=TRUE)
title(expression(paste(italic("M. cardinalis"), "Centro del rango"))))
box()
par(op)

```

matplot2 Graficar una matriz

DESCRIPCIÓN

Grafica los renglones de la matriz. Muy útil para desplegar una matriz de vectores de estado, tasas de sobrevivencia, sensibilidades, etc.

Uso

```
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, ...)
```

ARGUMENTOS

x	Una matriz
proportions	Si TRUE, entonces grafica los cambios proporcionales
legend	Una palabra clave o vector de legend con coordenadas x,y; las coordenadas por default son la esquina superior derecha
xlab	Una etiqueta para el eje x
ylab	Una etiqueta para el eje y
type	Tipo de gráfico, default es línea
las	Estilo de las etiquetas del eje, el default es horizontal
pch	Tipos de punto
lwd	Ancho de la línea
lty	Tipo de línea
col	Color
lcex	Aumento de la leyenda
lbty	Tipo de caja para la leyenda
lcol	Número de columnas para la leyenda
ltitle	Título de la leyenda
lsort	Sortea las leyendas en orden descendiente
...	Opciones adicionales se encuentran en la función plot

DETALLES

Sólo algunas posibilidades de leyenda se pueden utilizar. Para más control, ponga legend=NA y corra de manera separada.

VALOR

Un gráfico de matriz

AUTOR(ES)

Chris Stubben

VER TAMBIÉN

matplot y stage.vector.plot

EJEMPLOS

```
data(calathea)
# Tasas de sobrevivencia
x<-calathea[9:12]
x<-sapply(x, function(x) colSums(splitA(x, r=1:2)$T))
matplot2(t(x), legend="bottomright", ylab="Sobrevivencia",
main="Curvas de sobrevivencia de Calathea")

# Tasas de crecimiento - sin ordenar la leyenda
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="Tasa de Crecimiento",
main="Tasa de crecimiento de Calathea ")

# Convergencia a la estructura estable (excluyendo semillas)
x<-pop.projection(calathea[[7]], rep(1,8), 10)
matplot2(x$stage.vectors[-1,], prop=TRUE,
main="Vectores de estado de Calathea", lcex=.7)
```

mean.list **Calcula la matriz promedio**

DESCRIPCIÓN

Calcula la matriz promedio de una lista de matrices

Uso

```
## Metodo S3 ara la clase 'list':  
mean(x, ...)
```

ARGUMENTOS

x	Una lista de dos o más matrices
...	Argumentos adicionales pasadas a rowMeans

DETALLES

Regresa la matriz promedio de una lista de matrices usando la combinación de `unlist` y `rowMeans`. Vease ejemplo para mas detalles.

VALOR

La matriz promedio

NOTAS

Método S3 para `mean` de una lista de matrices.

AUTOR(ES)

Chris Stubben

VER TAMBIÉN

`var2`

EJEMPLOS

```
data(hudsonia)  
mean(hudsonia)  
## O  
x <- matrix(unlist(hudsonia), ncol=length(hudsonia) )  
matrix(rowMeans(x), 6, 6)
```

monkeyflower **Matrices de proyección para Mimulus**

DESCRIPCIÓN

Matrices agrupadas y anuales de poblaciones centrales y marginales de (*Mimulus cardinalis* y *M. lewisii*)

Uso

`data(monkeyflower)`

FORMATO

Un data frame con 32 matrices de proyección, arregladas con una matriz por renglón. Inicio del año de proyección o el agrupado de los tres años

a11	Elemento de matriz a11; transición de semilla a semilla o supervivencia del banco de semillas
a12	Elemento de matriz a12; pequeño no rep a semilla-fecundidad
a13	Elemento de matriz a13; grande no rep a semilla-fecundidad
a14	Elemento de matriz a14; reprod a semilla-fecundidad
a21	Elemento de matriz a21; semilla a pequeño no rep-crecimiento
a22	Elemento de matriz a22; pequeño no rep a pequeño no rep-stasis
a23	Elemento de matriz a23; grande no rep a pequeño no rep-regression
a24	Elemento de matriz a24; reprod a pequeño no rep-regression
a31	Elemento de matriz a31; semilla a grande no rep-crecimiento
a32	Elemento de matriz a32; pequeño no rep a grande no rep-crecimiento
a33	Elemento de matriz a33; grande no rep a grande no rep-stasis
a34	Elemento de matriz a34; reprod a grande no rep-regression
a41	Elemento de matriz a41; semilla a reprod-crecimiento
a42	Elemento de matriz a42; pequeño no rep a reprod-crecimiento
a43	Elemento de matriz a43; grande no rep a reprod-crecimiento
a44	Elemento de matriz a44; reprod to reprod-stasis

DETALLES

Matriz construida de un censo post reproductivo con cuatro categorías de estado: Semillas, pequeño no reproductivo, grande no-reproductivo, y reproductivo.

FUENTE

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

REFERENCIAS

- Amy Lauren Angert. 2006. Demography of central and marginal populations of monkeyflowers (*Mimulus cardinalis* and *M. lewisii*). *Ecology* 87:2014-2025.

EJEMPLOS

```
data(monkeyflower)
## Convierte los renglones de M. cardinalis a una lista de 16 matrices
A <- subset(monkeyflower, species=="cardinalis")
# usa as.matrix para convertir el data.frame a una matriz numerica
A<-split( as.matrix(A[, 4:19]), paste(A$site, A$year))
stages<-c("semilla", "sm.no rep", "lg.no rep", "repro")
## convierte a una lista de 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)

## grafica como la Figura 1A
x<- matrix(sapply(A, lambda), ncol=4)
colnames(x)<-c("BU", "CA", "RP", "WA")
rownames(x)<-c(2000:2002, "Agrupado")
x<-x[,c(1,3,4,2)]
colrs<-gray(0:3 / 3)[c(1,3,2,4)]
barplot(x, beside=TRUE, las=1, col=colrs, ylim=c(0,2),
ylab="Tasa de crecimiento poblacional", main="Mimulus cardinalis")
box()
abline(h=1, lwd=.5)
legend(1,1.95, rownames(x), fill=colrs, bty='n')
```

multiresultm Incorpora estocasticidad demográfica a las proyecciones poblacionales

DESCRIPCIÓN

Esta función genera números al azar para estados de transición a partir de una distribución multinomial y valores al azar con distribución lognormales o binomiales (para camadas=1) para fecundidades y regresa un vector del número de individuos por categoría de estado en el tiempo $t+1$.

Uso

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

ARGUMENTOS

<code>n</code>	Un vector de individuos por clase en el tiempo t
<code>T</code>	Una matriz de transición T
<code>F</code>	Una matriz de fecundidad F
<code>varF</code>	Una matriz de varianzas en fecundidad inter individual, el default es NULL para simular una población con tamaños de camada=1, para que las fecundidades den las probabilidades de nacimiento.

VALOR

La función regresa un vector del número de individuos por clase en el tiempo $t+1$.

AUTOR(ES)

Adaptado a R por Patrick Nantel.

FUENTE

- Adaptado del código de Matlab en el Recuadro 8.11 en Morris y Doak (2002) y sección 15.1.3 en Caswell (2001).

REFERENCIAS

- Caswell, H. 2001. Matrix population models. Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts.
- Morris, W. F., and D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
data(whale)
x<-splitA(whale)
whaleT<-x$T
whaleF<-x$F
```

```

multiresultm(c(1,9,9,9),whaleT, whaleF)
multiresultm(c(1,9,9,9),whaleT, whaleF)

## crea una grafica similar a la Figura 15.3a
reps <- 10 # numero de trayectorias
tmax <- 200 # longitud de las trayectorias
# inicializa la matriz totalpop para guardar las trayectorias
totalpop <- matrix(0,tmax,reps)
nzero <- c(1,1,1,1) # tama~no inicial de la poblacion
for (j in 1:reps)
{
  n <- nzero
  for (i in 1:tmax)
  {
    n <- multiresultm(n,whaleT,whaleF)
    totalpop[i,j] <- sum(n)
  }
}
matplot(totalpop, type = 'l', log="y",
        xlab = 'Tiempo (A~nos)', ylab = 'Poblacion Total')

```


nematode

Densidad poblacional del nemátodo del betabel

DESCRIPCIÓN

Una serie de tiempo de vectores poblacionales para el nemátodo del betabel *Heterodera schachtii*. Los individuos se clasificaron en tres categorías de estado (J2, J3+J4, y adulto) y las densidades (por 60 cc de suelo) fueron promediadas de cuatro replicas, medidas cada dos días durante diez días.

Uso

```
data(nematode)
```

FORMATO

Una matriz de densidades a partir de 3 clases de densidad a lo largo de 6 periodos de tiempo

FUENTE

- Usado en el ejemplo 6.3 de Caswell (2001).

REFERENCIAS

- Caswell, H. 2001. Matrix population models. Construction, Analysis, and interpretation. Second edition, Sinauer, Sunderland, Massachusetts.

VER TAMBIÉN

QPmat

EJEMPLOS

```
data(nematode)
stage.vector.plot(nematode, prop=FALSE, log='y', ylim=c(.3,200),
  xlab="Tiempo" , ylab="Densidad de Nematodos")
```

net.reproductive.rate **Tasa neta reproductiva**

DESCRIPCIÓN

Calcula la tasa neta reproductiva de una matriz clasificada por estados usando el eigenvalor dominante de la matriz **R**.

Uso

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

ARGUMENTOS

A	Una matriz de proyección
...	Elementos adicionales pasados a <code>splitA</code> y que son usados para separar A en matrices de transición T y de fecundidad F

DETALLES

`see seccion 5.3.4 in Caswell (2001).`

VALOR

Tasa neta reproductiva. Si la matriz de transición es singular, entonces se regresa NA.

NOTAS

Las versiones anteriores requerian a la matriz **T** y **F** como insumo

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

Vease `fundamental.matrix` y `generation.time` para otros atributos edad específicos

EJEMPLOS

```
data(whale)
net.reproductive.rate(whale)
## Fecundidades en la ultima columna
data(teasel)
net.reproductive.rate(teasel, r=1:6, c=6)
## Parcela 3 de Calathea - valores no son los mismos que en
  Caswell p. 105.
data(calathea)
sapply(calathea[9:12], net.reproductive.rate)
```

pfister.plot

Crea gráficas log-log de la varianza vs sensibilidad y el Coeficiente de variación vs elasticidad

DESCRIPCIÓN

Crea gráficas log-log de la varianza vs. elasticidad y Coeficiente de variación vs. elasticidad en elementos de la matriz. Las gráficas se basan en la Figura 2 de Pfister(1998).

Uso

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

ARGUMENTOS

A Una lista de dos o mas matrices anuales de proyección

DETALLES

Calcula el promedio, varianza y coeficiente de variación (CV) de los elementos de la matriz de una lista de dos o más matrices de proyección. Las matrices de sensibilidad y elasticidad son calculadas para elementos de la matriz usando la matriz promedio con `eigen.analysis`

VALOR

Crea dos gráficas log-log plots similares a la Figura 2 en Pfister (1998) y saca un data.frame con 5 columnas enlistando el promedio, varianza, CV, sensibilidad y elasticidad para los elementos de la matriz con un promedio y varianza > 0.

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Pfister, CA. 1998. Patterns of variance in stage-structured populations: Evolutionary predictions and ecological implications. PNAS 95:213-218.

EJEMPLOS

```
## 4 marices de Hudsonia
data(hudsonia)
pfister.plot(hudsonia)
```

```
## Matrices de Mimulus cardinalis en Carlon
data(monkeyflower)
mim<-subset(monkeyflower, species == "cardinalis" &
```

```

        site == "Carlon" & year != "pooled", select = c(4:19))
## Convierte el data frame a una lista de matrices usando la
  funcion split
mim1<-split(mim, 2000:2002)
mim2<-lapply(mim1, matrix, nrow=4, byrow=TRUE)
vr1<- pfister.plot(mim2)
vr1

## Grafica usando etiquetas
plot(vr1$cv, vr1$elas, xlab="CV", ylab="Elasticidad", log="xy", type='n')

# Separa los elementos de la matriz en transiciones que representan F
# (fecundidad), S (sobrevivencia), G (crecimiento), y R (regresos).
# Fecundidad en el primer renglon, sobrevivencia en la diagonal
# principal, crecimiento en la diagonal inferior y regresos en la
# diagonal inferior.

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)

### añade la linea de tendencia
abline(lm(log10(vr1$elas)~log10(vr1$cv)), col="red")

## incluye correlacion de rangos de Spearman
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 **Calcula las tasas de crecimiento por proyección**

DESCRIPCIÓN

Calcula la tasa de crecimiento poblacional y la estructura estable de tamaños por iteración de la ecuación $n(t+1) = An(t)$.

Uso

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

ARGUMENTOS

A	Una matriz de proyección
n	Vector inicial de estado o edad
iterations	Número de iteraciones

DETALLES

Eventualmente las poblaciones estructuradas convergen a una estructura estable y no hay cambio en la proporción de individuos entre categorías.

VALOR

Una lista con 5 elementos

lambda	Estimación de lambda usando los cambios entre el primer y segundo conteo poblacional
stable.stage	Estimado de la estructura estable usando proporciones en el vector de estados anterior
stage.vector	Una matriz con el número proyectado de individuos en cada categoría de edad o estado
pop.sizes	Número total de individuos proyectados
pop.changes	Cambios proporcionales en el tamaño de la población

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Vease sección 2.2 in Caswell (2001)

VER TAMBIÉN

`stage.vector.plot` para graficar los vectores de estado

EJEMPLOS

```
## matriz promedio de Freville et al 2004
stages<-c("plantula", "vegetativo", "floreando")
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)

####

data(whale)
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$stage.vectors, col=2:4, ylim=c(0, 0.6))
## convergencia es lenta con una razon de amortiguamiento cercano a 1
damping.ratio(A)
pop.projection(A,n, 100)$pop.changes
```

projection.matrix Construye una matriz de proyección usando tablas de frecuencias de transición

DESCRIPCIÓN

Construye un modelo de proyección por edad o estado de una tabla de frecuencias de transición que enlista el estado en el tiempo t , destino en el tiempo $t+1$, y uno o más columnas de fecundidad individual.

Uso

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

ARGUMENTOS

transitions	Un data frame de estado y destino con clase de edad en el censo actual, destino en los censos subsecuentes y uno o más columnas de fertilidad
stage	Nombre de columna o posición de la columna de estado en el data frame de estado-destino. El default es "stage".
fate	Nombre de la columna de destino en el data frame de estado-destino. El default es "fate".
fertility	Una o más columnas de fertilidad en el data frame de estado-destino. Por default, cualquier columna cuyo nombre se parezca a los nombres de los estados asume que contienen fecundidades individuales.
sort	Un vector enlistando clases de estado que corresponden a los renglones y columnas de la matriz de proyección deseada. Actualmente, los nombres de este vector deben de coincidir con los nombres en la columna de estado. Además, esta opción sólo debe de ser utilizada si los estados no están ordenados (<code>ordered</code>), ya que el default es que sean organizados por los niveles (<code>levels</code>) de la columna de estado.
add	Un vector enlistando renglón, columna y valor usado para añadir transiciones <i>estimadas</i> a la matriz de transición (por ejemplo, a una transición del banco de semillas a plántula) puede ser representada.
TF	Salida separada por matrices de transición (T) y fecundidad (F) matrices. El default es FALSE y sólo genera una sola matriz de proyección A

DETALLES

Las tasas de transición de cada estado se estiman utilizando la tabla de frecuencias de transiciones (Vease sección 6.1.1, Caswell 2001), de tal manera que esta técnica puede ser ampliamente utilizada en estudios demográficos de plantas u

otros organismos sésiles en donde los organismos son marcados o etiquetados y consistentemente reencontrados en censos anuales. Las tasas de fecundidad se calculan promediando las fecundidades individuales por estado, de tal manera que se debe de tener algo de cuidado para poder estimar correctamente las fecundidades individuales basadas en cuando se realiza el censo. Genera un error cuando no existe una transición. Esto se puede arreglar si se agrega artificialmente la transición y posteriormente se cambia con el comando add. La otra posibilidad es que genere una matriz vacía inicial. Creemos que la segunda es más elegante y versátil que la primera.

VALOR

El valor de default de la salida es una matriz única de proyección A. Si la bandera TF flag es TRUE, entonces se generan dos matrices, una de las transiciones T y una de las fecundidades F en donde $A=T+F$

T	Matriz de transición
F	Matriz de Fecundidad

NOTAS

Las fecundidades individuales deben de ser el número total de vástagos al final del intervalo del censo. Por lo tanto, las fecundidades deben de incluir la sobrevivencia de los vástagos en un censo prereproductivo (y más de un tipo de vástago puede ser representado). En un censo postreproductivo, los nuevos vástagos nacieron antes del censo, así que la fecundidad es simplemente el número de vástagos en esa clase.

AUTOR(ES)

Chris Stubben

EJEMPLOS

```
data(test.census)

trans01 <- subset(merge(test.census, test.census, by = "plant",
  sort =FALSE),
  year.x==2001 & year.y==2002 )
## A~nade fecundidades individuales usando "reproduccion anonima"
  basada
## en el potencial reproductivo proporcional de plantas en flor
  y el numero
## total de plantulas al final del intervalo de proyeccion
trans01$seedferts <- trans01$fruits.x/sum(trans01$fruits.x) * 5
trans01

stages<-c("seedling", "vegetative", "reproductive")
```

```

## tres maneras de especificar columnas
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)

## La MEJOR manera es usar la columna default (columna de
## fecundidad (seedling) ahora tenemos que hacer iguales
## el nombre de la clase)
names(trans01)[c(3, 6, 8)] <- c("stage", "fate", "seedling")
# Y ordena los estados en el dataframe
trans01$stage<-ordered(trans01$stage, stages)

projection.matrix(trans01)
projection.matrix(trans01, TF=TRUE)

## Ejemplo usando datos de Aquilegia
data(aq.trans)
sf<- subset(aq.trans, year==1998 & plot==909,
            c(year, plant, stage, fruits, fate))
## renglones y columnas de la matriz final
levels(sf$stage)

## plantulas el a~no que entra
seedlings<-nrow(subset(aq.trans, plot==909 & year==1999 &
                      stage=="recruit"))

## Añade estimados de fecundidad individual para reclutas y semillas
## asumiendo un banco de semillas y las nuevas semillas contribuyen a
## un pool comun de plantulas germinadas con una probabilidad
## igual de reclutamiento

seed.survival<-0.4
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)

## añade dos columnas de fecundidad
sf$recruit<- sf$fruits/sum(sf$fruits) * seeds.from.plants *
  recruitment.rate
sf$seed<-sf$fruits * seeds.per.fruit * seed.survival

## añade sobrevivencia de banco de semillas y reclutamiento de la tasa
## de reclutamiento del banco de semillas a la matriz de transicion

A<-projection.matrix(sf, add=c(1,1, seed.survival, 2,1, recruitment.rate))
A
max(Re(eigen(A)$values))

```

QPmat Construye una matriz de proyección de una serie de tiempo de individuos (o densidades) por estado.

DESCRIPCIÓN

Esta función construye una matriz de proyección de una serie de tiempo de individuos (o densidades) por estado (clases de tamaño o estado) usando el método de programación cuadrático de Wood. El modelo matricial también requiere de una matriz acotada *C*, un vector *b*, y un vector que enlista elementos que no son cero de la matriz de población deseada.

Uso

```
QPmat(nout, C, b, nonzero)
```

ARGUMENTOS

<i>nout</i>	Una serie de tiempo de vectores de población
<i>C</i>	Matriz <i>C</i> de restricción
<i>b</i>	Vector <i>b</i>
<i>nonzero</i>	Índices de lo elementos no-cero de la matriz de transición (contando por columna)

VALOR

Una matriz de proyección.

NOTAS

Esta función utiliza el paquete 'quadprog', que debe de ser instalado y cargado en el sistema.

AUTOR(ES)

Código original de Matlab code en Caswell (2001:148). Adaptado a R por Patrick Nantel

FUENTE

- Código de Matlab convertido del ejemplo 6.3 en Caswell (2001)

REFERENCIAS

- Caswell, H. 2001. Matrix population models. Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts.

EJEMPLOS

```
data(nematode)
## enlista elementos que no son cero
```

```
nonzero<- c( 1, 2, 5, 6, 7, 9)
## crea la matriz C
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))
## calcula b (no es necesario transponer - funciona de ambas maneras)
b<-apply(C, 1, max)
QPmat(nematode, C,b,nonzero)
```

reproductive.value Valor reproductivo

DESCRIPCIÓN

Calcula el valor reproductivo de una matriz de proyección

Uso

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

ARGUMENTOS

A Una matriz de proyección

DETALLES

Vease sección 4.5 en Caswell (2001).

VALOR

Un vector que contiene los valores reproductivos escalados para que $v[1]=1$

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
data(teasel)
v<-reproductive.value(teasel)
v
dotchart(log10(v), pch=16, xlab="Valor reproductivo (log10)")
```

resample Remuestreo

DESCRIPCIÓN

Remuestrea una matriz de proyección usando una distribución multinomial para transiciones y una log-normal para fertilidades.

Uso

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

ARGUMENTOS

A	Una matriz de proyección
n	Ya sea un vector de estados con el número de transiciones a muestrear en cada columna o un valor único que se aplica a todas las columnas
fvar	Ya sea un vector de diferentes varianzas de la fecundidad o una varianza única (default=1.5) de fecundidad que se aplica a todas las tasas
...	Elementos adicionales son pasados a splitA y que son usados para separar la matriz de transiciones A en una de transiciones T y una de Fecundidad F

DETALLES

La matriz de proyección A es primero separada en matrices de transición y fecundidad. La mortalidad se añade a la matriz de transición y las columnas son remuestreadas de una distribución `Multinomial` basada en el tamaño correspondiente en cada clase de estado n.

Las tasas de fecundidad son muestreadas de una distribución log-normal usando la función `lnorms`. La varianza puede ser un valor único que se aplica a todas las tasas o un vector con diferentes valores que se aplica a cada tasa. En este caso, los valores se reciclan para ajustar el número de fecundidades que no son cero.

VALOR

Una matriz remuestreada

NOTAS

Vease sección 12.1.5.2 sobre el remuestreo en Caswell (2001)

AUTOR(ES)

Chris Stubben

VER TAMBIÉN

boot.transitions

EJEMPLOS

```
data(hudsonia)
A<-hudsonia[[1]]
lambda(A)
## NOTA: Las fecundidades estan en los primeros dos renglones por lo que
## se debe de usar r=1:2 para partir esta matriz. Remuestrea las
## transiciones 100 veces cada una resample(A, 100, r=1:2). Ajusta un
## valor superior de fvar para los estados 4 y 6 porque hay dos
## fecundidades por estado (8 en total), se necesitan repetir
## los valores de
## remuestreo resample(A,1000, fvar=c(1.5, 1.5, 3, 3), r=1:2)
## o remuestrea basandose en el numero de plantas censadas
# datos de la Tabla 6.4 y Recuadro 7.3)
n<-c(4264,3, 30, 16, 24,5)
## crea una lista con 1000 matrices
x<-lapply(1:1000, function(x) resample(A,n, r=1:2))
mean(x)
## usa var2 para revisar las varianzas especialmente si se usan
## diferentes valores
var2(x)
## tasas de crecimiento
y<-sapply(x, lambda)
quantile( y, c(0.025, .975) )

hist(y, br=30, col="palegreen", xlab="Lambda",
main="Tasas de Crecimiento de Hudsonia 1985")
abline(v=quantile(y, c(0.025, .975)), lty=3)

## Duplica el tamaño de muestra (y cuadruplica plantulas) y es posible
## que se pueda detectar una disminucion
n<-n*2
n[2]<-n[2]*2
x<-lapply(1:1000, function(x) resample(A, n*2, r=1:2))
quantile( sapply(x, lambda), c(0.025, .975) )
```

sensitivity

Análisis de sensibilidad de una matriz

DESCRIPCIÓN

Calcula las sensibilidades de los eigenvalores a cambios en los elementos de la matriz de proyección

Uso

```
sensitivity(A, zero=FALSE)
```

ARGUMENTOS

A	Una matriz de proyección
zero	Ajusta las sensibilidades para transiciones no observadas a cero, el default es FALSE

DETALLES

Vease seccion 9.1 en Caswell (2001).

VALOR

Una matriz de sensibilidad

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

`elasticity`

EJEMPLOS

```
data(teasel)

sens<-sensitivity(teasel)
## Grafica de la IMAGEN con cajas peque~nas
image2(sens, mar=c(1,3.5,5,1), box.offset=.1)
  title("Matriz de sensibilidad usando image2", line=2.5)
## MATPLOT
matplot2(sens, log='y', type='b', yaxt='n', ltitle="Destino",
  ylab=expression(paste("Sensibilidad de ",lambda)),
  main="Matriz de sensibilidad usando matplot2")
pwrs<- -4:1
axis(2, 10^pwrs, parse(text=paste("10^", pwrs, sep = "")), las=1)
```


splitA Separa una matriz de proyección en matrices separadas de T y F

DESCRIPCIÓN

Separa una matriz de proyección en una matriz de transiciones y una matriz de fecundidades donde $A=T+F$.

Uso

```
splitA(A, r = 1, c = -1)
```

ARGUMENTOS

A	Una matriz de proyección
r	Renglones contienen fecundidades, default es el primer renglón
c	Columnas que contienen fecundidades, el default es todas las columnas excepto la primera

DETALLES

Vease sección 5.1 en Caswell (2001)

VALOR

Una lista con matrices T y F

NOTAS

Por default, la matriz de fecundidad incluye elementos del primer renglón (excepto el primer elemento). En un censo prereproductivo, es posible tener elementos de la matriz que incluyen tanto la fertilidad como tasas de transición y por lo tanto esta función no va a funcionar.

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

funciones como `generation.time` y `net.reproductive.rate` usan `splitA` internamente para separar la matriz

EJEMPLOS

```
data(whale)
splitA(whale)
# teasel -fecundidades en la ultima columna
data(teasel)
splitA(teasel, r=1:6, c=6)
# hudsonia - fecundidades en las primeras dos columnas
data(hudsonia)
splitA(hudsonia[[1]], r=1:2)
# curvas de sobrevivencia
x<-sapply(hudsonia, function(x) colSums(splitA(x, r=1:2)$T))
matplot2(t(x), legend="bottomright", ylab="Sobrevivencia",
main="Curvas de sobrevivencia de Hudsonia")
```

stable.stage Estructura Estable

DESCRIPCIÓN

Calcula la estructura estable de una matriz de proyección

Uso

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

ARGUMENTOS

A Una matriz de proyección

DETALLES

Vease sección 4.5 en Caswell (2001).

VALOR

Un vector que contiene la estructura estable

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
data(teasel)
w<-stable.stage(teasel)
w
barplot(w, col="green", ylim=c(0,1), las=1,
        ylab="Proporciondeestadoestable", xlab="Clasedeestado",
        main="Teasel")
box()
```

stage.vector.plot

Grafica las trayectorias del vector de estado

DESCRIPCIÓN

Grafica la dinámica en el corto plazo y convergencia a la estructura estable usando proyecciones del vector de estado.

Uso

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

ARGUMENTOS

stage.vectors	Una matriz enlistando los vectores de estado en columnas
proportions	Grafica cambios proporcionales o números totales. El default son proporciones
legend.coords	A legend palabra clave del vector de coordenadas x, y. El default es la esquina superior derecha
ylim	Los límites del gráfico, default son los valores max y min en stage.vectors
xlab	Etiqueta para el eje x
ylab	Etiqueta para el eje y
col	Vector de colores de línea, el default es rainbow(8)
...	Opciones adicionales que son pasadas a la función plot

DETALLES

Un gráfico de proyecciones de estado o clase

AUTOR(ES)

Chris Stubben

REFERENCIAS

- Vease sección 2.2 en Caswell 2001

VER TAMBIÉN

- Vease pop.projection

EJEMPLOS

```
## matriz del ejemplo 2.1 en Caswell  
A<-matrix(c(  
  0,    0.3,    0,  
  1,    0,    0.5,  
  5,    0,    0),
```

```

nrow=3, dimnames=list(1:3,1:3))
n<-c(1,0,0)
p<-pop.projection(A,n,60)

## Graficas de la Figura 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)

## Escala logartmica de la y
stage.vector.plot(p$stage.vector, col=2:4, prop=FALSE,
ylim=c(.01, 10), log='y', legend="bottomright", yaxt='n')
pwrs<- -2:1

# Marcas mayores
axis(2, at = 10^pwrs, labels=parse(text=paste("10^", pwrs, sep = "")),
las=1, tcl= -.6)

# Marcas menores
axis(2, at = 1:9 * rep(10^pwrs[-1] / 10, each = 9),
tcl = -0.3, labels = FALSE)

```

stoch.growth.rate **Calcula la tasa de crecimiento log estocástica**

DESCRIPCIÓN

Calcula la tasa de crecimiento log estocástica por la aproximación de Tuljapurkar y por simulación.

Uso

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

ARGUMENTOS

<code>matrices</code>	Una lista (list) con dos o mas matrices de proyección, o una matriz con una proyección por columna con los elementos en columnas
<code>prob</code>	Un vector de probabilidades de la muestra (<code>sample</code>) que seleccione diferencialmente las matrices de proyección, el default son probabilidades iguales para cada matriz
<code>maxt</code>	Número de intervalos de tiempo, default=50000
<code>verbose</code>	Imprime el comentario al inicio del tiempo 1, 10000, 20000, etc.

VALOR

Una lista con tres elementos

<code>approx</code>	Tasa de crecimiento log estocástica con la aproximación de Tuljapurkar
<code>sim</code>	Tasa de crecimiento log estocástica por simulación
<code>sim.CI</code>	Intervalo de confianza de la simulación

AUTOR(ES)

Chris Stubben

FUENTE

- Código de Matlab convertido del Recuadro 7.4 en Morris y Doak (2002)

REFERENCIAS

- Morris, W. F., y D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

`stoch.projection` para obtener los tamaños poblacionales de la simulación

EJEMPLOS

```
data(hudsonia)
sgr<-stoch.growth.rate(hudsonia)
sgr

exp(sgr$approx)
```

stoch.projection **Simula la tasa de crecimiento e stocástica de una serie de matrices**

DESCRIPCIÓN

Simula la tasa de crecimiento estocástica por proyección usando técnicas de selección de matrices completas en un ambiente independiente e idénticamente distribuido (iid) de una serie de 2 o más matrices de proyección

Uso

```
stoch.projection(matrices, n0, tmax = 50, nreps = 5000,  
prob = NULL, nmax = NULL, sumweight = NULL, verbose=TRUE)
```

ARGUMENTOS

<code>matrices</code>	Una lista (<code>list</code>) con dos o más matrices de proyección, o una matriz con una proyección por columna, con elementos en las columnas
<code>n0</code>	Vector inicial de la población
<code>tmax</code>	Número de intervalos de tiempo o de proyección para predecir el tamaño de la población a futuro
<code>nreps</code>	Número de iteraciones
<code>prob</code>	Un vector de probabilidades de la muestrea (<code>sample</code>) que seleccione diferencialmente las matrices de proyección, el default son probabilidades iguales para cada matriz
<code>nmax</code>	Un número máximo de individuos que la población no puede exceder. Default es la denso independiente
<code>sumweight</code>	Un vector de unos y ceros usados para omitir los estados cuando se revisan los umbrales de densidad. Default es la suma en todas las clases de estado
<code>verbose</code>	Imprime comentarios el inicio de la iteración 1, 100, 200, 300, etc.

VALOR

Una matriz enlistando tamaños de población finales por estado o clase con una iteración por renglón.

AUTOR(ES)

Chris Stubben

FUENTE

- Código de Matlab convertido del Recuadro 7.3 en Morris y Doak (2002) con Nmax option añadido para introducir denso dependencia sencilla

REFERENCIAS

- Morris, W. F., y D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
data(hudsonia)
n<-c(4264, 3,30,16,25,5)
names(n)<-c("semilla", "plantula", "mini", "pequena", "mediana", "grande")

### uso de probabilidades iguales y desiguales para la seleccion
### de las matrices
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), xlim=c(0,5000), ylim=c(0,200), col="green",
breaks=seq(0,5000, 100), xlab="Tamano poblacional final a t=50", main='')

par(new=TRUE)

## use transparent para sobrelapar las distribuciones - puede no
## funcionar en algunos sistemas
hist(apply(x.uneq, 1, sum), xlim=c(0,5000), ylim=c(0,200),
col = rgb(0, 0, 1, 0.2), xaxt='n', yaxt='n', ylab=' ', xlab=' ',
breaks=seq(0,10000, 100), main=' ')

legend(2500,200, c("igual", "desigual"),fill=c("green", rgb(0,0,1,0.2)))
title(paste("Proyeccion de crecimiento estocastico para Hudsonia
usando probabilidades iguales y desiguales"), cex.main=1)

## tama~no inicial de la poblacion
sum(n)
abline(v=sum(n), lty=3)
```

DESCRIPCIÓN

Estima la probabilidad de cuasi-extinción por simulación para una población estructurada en una ambiente distribuido estocásticamente, independiente e idéntico

Uso

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

ARGUMENTOS

<code>matrices</code>	Una lista (<i>list</i>) con dos o más matrices de proyección, o una matriz con una proyección por columna, con elementos en las columnas
<code>n0</code>	Vector inicial de la población
<code>Nx</code>	Umbral de cuasi extinción
<code>tmax</code>	Número de intervalos de tiempo o de proyección
<code>maxruns</code>	Número de veces para simular la función de distribución acumulada
<code>nreps</code>	Número de iteraciones
<code>prob</code>	Un vector de probabilidades de la muestra (<i>sample</i>) que seleccione diferencialmente las matrices de proyección
<code>sumweight</code>	Un vector de unos y ceros usados para omitir los estados cuando se revisan los umbrales de cuasi-extinción. Default es la suma de todas las clases.
<code>verbose</code>	Imprimir comentarios al inicio de la corrida 1, 2, 3, etc.

VALOR

Una matriz con probabilidades de cuasi-extinción por cada una de las corridas en columnas

AUTOR(ES)

Chris Stubben

FUENTE

- Código de Matlab convertido del Recuadro 7.5 en Morris y Doak (2002).

REFERENCIAS

- Morris, W. F., y D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

- `stoch.projection`

EJEMPLOS

```
data(hudsonia)
n<-c(4264, 3,30,16,25,5)
names(n)<-c("semilla", "plantula", "mini", "pequena", "mediana", "grande")
## excluye las semillas usando sumweight
x<-stoch.quasi.ext(hudsonia,n,Nx=10,nreps=500,sumweight=c(0,1,1,1,1,1))
matplot(x, xlab="A~nos", ylab="Probabilidad de Cuasi-extincion",
        type='l', lty=1, col=rainbow(10), las=1,
        main="Tiempo para llegar al umbral de cuasi extincion de
        10 individuos")
```

stretchbetaval Genera números al azar con distribución beta estirada

DESCRIPCIÓN

Genera un número de beta estirada con promedio, desviación estándar, valores mínimo y máximo y valores de CDF para estimaciones de fecundidad acotadas

Uso

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

ARGUMENTOS

mn	Promedio de la tasa de fecundidad
std	Desviación estándar
minb	Valor mínimo
maxb	Valor máximo
fx	Valor de la función de distribución acumulada

DETALLES

Esta función llama a la función 'betaval'.

VALOR

Regresa un número distribución beta estirado con promedio, mn, desviación estándar std, valores mínimo y máximo (minb,maxb) valor de CDF de Fx.

AUTOR(ES)

Script original de MATLAB por Morris y Doak (2002:283). Adaptado a R por Patrick Nantel, 11 Julio 2005.

FUENTE

- Código de Matlab convertido del Recuadro 8.5 en Morris y Doak (2002).

REFERENCIAS

- Morris, W. F., y D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

`betaval`

EJEMPLOS

```
stretchbetaval(3, 1.2, 1, 20, runif(1))
## Genera al azar valor de beta estirado
## Fecundidades para una poblacion de 1000 individuos maduros
## (ni) con fecundidad promedio (f) de 3.0 y varianza inter-individual
## en fecundidad (varF) de 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="Fertilidades al azar beta estiradas",
xlab="Tasa de fertilidad", , col="blue")
```

teasel **Matriz de proyección de teasel**

DESCRIPCIÓN

Matriz de proyección de la planta teasel

Uso

```
data(teasel)
```

FORMATO

Una matriz de proyección

FUENTE

- Example 5.2

REFERENCIAS

- Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
data(teasel)
image2(teasel, mar=c(1,3.5,5,1) , box.offset=.1)
  title("Matriz de proyeccion de teasel", line=2.5)
# # fecundidades para una planta monocarpica en un censo
## prereproductivo en la ultima columna
splitA(teasel, r=1:6, c=6)
lambda(teasel)
```

test.census Datos de censo de una planta hipotética

DESCRIPCIÓN

Tres años de datos de censos de una planta hipotética con tres clases de estado.

Uso

```
data(test.census)
```

FORMATO

Un data frame con 41 observaciones de censo con las siguientes variables

plant	Número de la planta
year	Año del censo
stage	Clases de estado: plántula (seedling), vegetativo (vegetative), o reproductivo (reproductive)
fruits	Número total de frutos

EJEMPLOS

```
data(test.census)
stages <- c("seedling", "vegetative", "reproductive")

## Tabulación de los vectores de estado y ordena los renglones por
## estado
sv<- table(test.census$stage, test.census$year)[stages,]
sv
stage.vector.plot(sv)

## Asigna xaxt='n' para evitar fracciones de año (2002.5)
stage.vector.plot(sv, prop=FALSE, xaxt="n", las=1)
axis(1, 2001:2003, c(2001, 2002, 2003))

## Convierte datos de censo a una tabla de transiciones de
## estado-destino usando reshape
reshape(test.census, direction="wide", idvar="plant", timevar="year")
## Convierte datos de censo a una tabla de transiciones de
## estado-destino usando merge
trans <- subset(merge(test.census, test.census, by="plant", sort=FALSE),
               year.x==year.y-1)
trans

## Formato de nombres en columnas y renglones
trans<-trans[,c(1:4,6)]
colnames(trans)[2:5] <- c("year", "stage", "fruits", "fate")
rownames(trans) <- 1:nrow(trans)
## Ordena las columnas de estado y destino
trans$stage <- ordered(trans$stage, levels = stages)
trans$fate <- ordered(trans$fate, levels = c(stages,"dead"))
```

```

## Selecciona transiciones para 2001-2002 y cuenta vastagos (plantulas)
trans01 <- subset(trans, year==2001)
seedlings<-nrow(subset(test.census, year==2002 & stage=="seedling"))

## Añade fecundidades individuales usando una "reproduccion anonima"
## basada en reproduccion proporcional de plantas en flor y el total
## de plantulas al final del intervalo de proyeccion
trans01$seedling<-trans01$fruits/sum(trans01$fruits) * seedlings
trans01
## Crea la tabla de frecuencias de transicion y construye la matriz T
tf<-table( trans01$fate, trans01$stage )
tf
## quita el destino "dead" de la matriz T.mat<-prop.table(tf,2)[-4,]
T.mat<-prop.table(tf,2)[stages,]
T.mat
## Resume las tasa de fecundidad especificas por estado y construye la
## matriz F
fert<-tapply(trans01$seedling, trans01$stage, mean)
fert
F.mat<-T.mat*fert
F.mat[1,]<- fert
F.mat
## La matriz de proyeccion final es simplemente T.mat+F.mat
## O usa la funcion de proyeccion de la matriz -
projection.matrix(trans01)

```


tortoise **Matrices de proyección de la tortuga de desierto**

DESCRIPCIÓN

Matrices de proyección para la tortuga de desierto *Gopherus agassizii*

Uso

```
data(tortoise)
```

FORMATO

Una lista de 4 matrices de proyección con 4 diferentes estimaciones de fecundidad (bajo, medio bajo, medio alto, y alto)

FUENTE

- Tabla 5 en Doak *et al* (1994). Usado por Caswell (2001) en el Capítulo sobre análisis de sensibilidad.

REFERENCIAS

- Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.
- Doak, D., P. Kareiva, and B. Kleptetka. 1994. Modeling population viability for the desert tortoise in the Western Mojave Desert. *Ecological Applications* 4:446-460.

EJEMPLOS

```
data(tortoise)
A<-tortoise[["med.high"]]
# No se necesita la escala de color log
image2(A, mar=c(1,3.5, 5,1), log=FALSE, box.off=.1)
title("Matriz de proteccion de tortuga", line=3)

splitA(A)
lambda(A)
sapply(tortoise, lambda)
```

var2 **Calcula la varianza de una matriz**

DESCRIPCIÓN

Calcula la varianza de una lista de matrices

Uso

```
var2(x)
```

ARGUMENTOS

`x` Una lista de dos o más matrices

DETALLES

Regresa una matriz que contiene las varianzas de una lista de matrices usando una combinación de `unlist` y `apply`.

VALOR

Una matriz que contiene varianzas

AUTOR(ES)

Chris Stubben

EJEMPLOS

```
data(hudsonia)
var2(hudsonia)
```

varEst Estima la varianza de una tasa vital beta-binomial usando el método de aproximación de Akcakaya

DESCRIPCIÓN

Esta función encuentra el mejor estimador de la media y varianza ambiental de tasas vitales beta-binomiales, usando el método de proximación de Akcakaya (2002).

Uso

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

ARGUMENTOS

<code>rates</code>	Una matriz o data frame con 4 columnas: Identificador de tasa, Año, Número total de individuos al inicio, Número que sobreviven (o crecen)
<code>weighted</code>	Ya sea 1 para la varianza demográfica promedio ponderada o 0 para el promedio no ponderado, el default es 1

VALOR

Una matriz con 3 columnas: (1) Varianza observada total, (2) estimado de la varianza por estocasticidad demográfica y (3) estimado de la varianza debida a estocasticidad ambiental

AUTOR(ES)

Patrick Nantel, 20 Junio 2005. Modificado el 1 Mayo 2007.

REFERENCIAS

- Akcakaya, H. R. 2002. Estimating the variance of survival rates and fecundities. *Animal Conservation* 5:333-336.
- Kendall, B. E. 1998. Estimating the magnitude of environmental stochasticity in survivorship data. *Ecological Applications* 8(1):184-193.

VER TAMBIÉN

`Kendall`

EJEMPLOS

```
data(woodpecker)
varEst(woodpecker)
```

vitalsens Elasticidades y sensibilidades de las tasas vitales

DESCRIPCIÓN

Calcula las sensibilidades y elasticidades determinísticas de lambda para las tasas vitales usando derivadas parciales

Uso

```
vitalsens(elements, vitalrates)
```

ARGUMENTOS

<code>elements</code>	Un objeto de modo <code>expression</code> con todos los elementos de la matriz representados por ceros o valores simbólicos de las tasas vitales
<code>vitalrates</code>	Una lista de tasa vitales con nombres (<code>names</code>) igualando las expresiones de los elementos de arriba

DETALLES

Las sensibilidades y elasticidades de las tasa vitales se ven en el ejemplo 9.3 y 9.6 en Caswell (2001). Vease también el Capítulo 9 y Recuadro 9.1 para el código en Matlab en Morris y Doak (2002).

VALOR

Un data frame con estimados de las tasas vitales, sensibilidades y elasticidades.

NOTAS

Las expresiones de los elementos deberían de regresar valores de los elementos estimados de la matriz después de evaluar las variables usando la función `eval` abajo.

```
A<-sapply(elements, eval, vitalrates, NULL)
```

Además, estas expresiones deberían de ser arregladas en renglones para que lo siguiente regrese una matriz de proyección.

```
matrix(A, nrow=sqrt(length(elements)), byrow=TRUE)
```

AUTOR(ES)

Chris Stubben. Basado en código de Simon Blomberg al R-help mailing list.

REFERENCIAS

- Caswell, H. 2001. Matrix population models. Construction, Analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts.
- Morris, W. F., y D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts.

EJEMPLOS

```
## ganso emperador en Morris y 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,      0,      Ss1,    Ss1)

## primero grafica el efecto de cambiar las tasas vitales --Figura 9.1
n<-length(goose.vr)
vr<-seq(0,1,.1)
vrсен<-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)
    vrсен[i,h] <- max(Re(eigen(A)$values))
  }
}
matplot(rownames(vrсен), vrсен, type='l', lwd=2, las=1,
ylab="Creciento poblacional de ganso", xlab="valor de la tasa vital",
main="Efecto de cambiar las tasas vitales del ganso")
vrн<-expression(s[0], s[""]>=1], f[2], f[""]>=3])
legend(.8, .4, vrн, lty=1:4, lwd=2, col=1:4, cex=1.2)
## despues calcula las sensibilidades -- Tabla 9.1
x<-vitalsens(goose.el, goose.vr)
x
sum(x$elasticity)

barplot(t(x[,2:3]), beside=TRUE, legend=TRUE, las=1, xlab="Tasa vital",
main="Elasticidad y sensibilidad de las tasas vitales del ganso")
abline(h=0)

## Tabla 7 cerncalo en peligro de 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="Tasa vital",
main="Sensibilidad y elasticidad de la tasa vital del cerncalo")
legend(1,1, rownames(t(x[,2:3])), fill=grey.colors(2))
abline(h=0)

```

vitalsim **Calcula la tasa de crecimiento estocástica y tiempo a la extinción CDF usando tasas vitales con correlación intra anual, auto y inter**

DESCRIPCIÓN

Esta función genera una serie de proyecciones de viabilidad poblacional (PVA) estocásticas remuestreando las tasas vitales de una distribución beta, beta estirada o log normal dependiendo de la variable e incluye auto correlaciones.

Uso

```
vitalsim(vrmeans, vrvars, corrin, corROUT, makemx, n0,  
yrspan, Ne=500, tmax=50, runs=500, vrtypes=NULL,  
vrmins=NULL, vrmaxs=NULL, sumweight=NULL)
```

ARGUMENTOS

vrmeans	Promedios de las tasas vitales
vrvars	Varianza de las tasas vitales
corrin	Correlaciones intra anuales
corROUT	Correlaciones inter anuales
makemx	Una función que crea una matriz de proyección de un vector de vrmeans
n0	Vector población inicial
yrspan	El número de años para construir las correlaciones para construir la matriz M12
Ne	Umbral de cuasi-extinción
tmax	Tiempo final para calcular la probabilidad de cuasi-extinción, default es de 50 intervalos de tiempo
runs	Número de trayectorias, el default es de 500, pero se recomiendan 1000
vrtypes	Identifica la distribución para cada tasa vital en vrmeans donde 1=beta, 2=beta estirada, 3=lognormal, default son todas
vrmins	Valor mínimo para cada tasa vital; use ceros para tasas que no son betas estiradas, el default es todos ceros
vrmaxs	Valor máximo, valor para cada tasa vital, use ceros para tasas que no son betas estiradas, el default es todo ceros
sumweight	Un vector de ponderación con 0 para omitir una clase y 1 para incluirla cuando calcula la suma de la densidad y compararla con el umbral de cuasiextinción, el default incluye todas las clases

DETALLES

Las tasas vitales usadas deben de tener valores de fecundidad o probabilidades binomiales, esto es, probabilidades para eventos con únicamente dos posibles resultados (como la sobrevivencia). Promedios y varianza de las tasas vitales de preferencia deben de ser corregidas para eliminar errores de muestreo y estocasticidad

demográfica. Nota: esta versión de la función no simula estocasticidad demográfica y es denso independiente.

VALOR

La función grafica un histograma de las tasas vitales estocásticas y la probabilidad acumulada de cuasi-extinción y regresa una lista con 4 elementos:

<code>detLambda</code>	La tasa de crecimiento determinística calculada de la matriz promedio
<code>stochlambda</code>	La tasa promedio de crecimiento estocástica con intervalos de confianza al 95%.
<code>logLambdas</code>	Un vector del logaritmo de todas las tasas de crecimiento estocásticas en la primera gráfica
<code>CDFExt</code>	Un vector de probabilidades acumuladas de cuasi extinción en una segunda gráfica

NOTAS

Las matrices de correlación para Hudsonia en <http://www.sinauer.com/PVA/hudcorrs.mat> incluyen algunas correlaciones. Una versión corregida de las correlaciones fue enviada por D. Doak el 8/4/2007. Por lo tanto los resultados de la simulación que se encuentra abajo son diferentes a los presentados en el libro.

AUTOR(ES)

Programa en Matlab original por Morris y Doak (2002:301-304). Adaptado a R por Patrick Nantel, 12 Julio 2005.

FUENTE

- Código de Matlab convertido del Recuadro 8.10 en Morris y Doak (2002).

REFERENCIAS

- Morris, W. F., y D. F. Doak. 2002. Quantitative conservation biology: Theory and practice of population viability analysis. Sinauer, Sunderland, Massachusetts, USA.

VER TAMBIÉN

`hudmxdef`, `hudvrs` y `hudcorrs`

EJEMPLOS

```
## carga las tasas vitales y las matrices de correlacion
data(hudvrs)
data(hudcorrs)
## ajusta vrtypes
hudvrtypes<-c(rep(1,13), rep(3,5), rep(1,6))
```



```

## corre el modelo completo- usando 100 corridas
## en este ejemplo para que sea mas rapido
#aunque deben de ser arriba de 100,000
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)
## lambda deterministica y estocastica
full[1:2]
## log de lambda estocastica
log(full$stochLambda)
sd(full$logLambdas)

## Brinca las siguientes dos simulaciones, sin embargo
## la salida de muestra se incluye para graficar sin correlaciones
## entre años SIN correlaciones entre años de tal manera que
#corrout = diag(0,13)- all zeros
# no.between<-vitalsim(hudvrs$mean, hudvrs$var, hudcorrs$corrin,
# diag(0,13), hudmxdef, vrtypes=hudvrtypes,
# n0=c(4264,3,30,16,25,5), yrspan=20 )
no.between<-list(CDFExt=c(rep(0,40),0.01,0.04,0.12,0.15,
0.20,0.31,0.49,0.58,0.72,0.78))
#SIN correlaciones as que corrout = diag(0,13) y corrin=diag(13)
##- unos en la diagonal
# no.corr<-vitalsim(hudvrs$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))

## Figura 8.3 con correccion de las matrices de correlacion
## para el modelo completo
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="Años al futuro",
ylab="Probabilidad acumulada de cuasi extincion")
legend(2,1,c("Modelo completo","Sin correlacion interanual",
  "Sin correlacion"),
lty=1:3, col=2:4, lwd=2)

```

whale **Matriz de proyección para la ballena asesina**

DESCRIPCIÓN

Matriz de proyección para la ballena asesina

Uso

```
data(whale)
```

FORMATO

Una matriz de proyección

FUENTE

Matriz de proyección del ejemplo 5.1 en Caswell (2001)

REFERENCIAS

- Caswell, H. 2001. Matrix population models: Construction, analysis, and interpretation. Second edition. Sinauer, Sunderland, Massachusetts, USA.

EJEMPLOS

```
data(whale)
whale
splitA(whale)
lambda(whale)
sensitivity(whale)
# grafica la sensibilidad
matplot2(sensitivity(whale), type='b', legend='topleft',
         ltitle='Destino', main='Sensibilidad de la ballena asesina')
```

woodpecker Datos de sobrevivencia para adultos y juveniles de pájaro carpintero

DESCRIPCIÓN

Número de pájaros carpinteros juveniles y adultos y sobrevivencia en la población de Water Canyon, New Mexico, reconstruida de Stacey y Taper (1992).

Uso

```
data(woodpecker)
```

FORMATO

Un data frame con 18 observaciones de las siguientes 4 variables

rate	Estado adulto o juvenil
year	Año
start	Número total de individuos al inicio
surv	Número que sobreviven a la primavera

FUENTE

Stacey, P.B., y M. Taper. 1992. Environmental variation and the persistence of small populations. *Ecological Applications* 2:18-29.

REFERENCIAS

- Akcakaya, H. R. 2002. Estimating the variance of survival rates and fecundities. *Animal Conservation* 5:333-336.
- Kendall, B. E. 1998. Estimating the magnitude of environmental stochasticity in survivorship data. *Ecological Applications* 8(1):184-193.

VER TAMBIÉN

Kendall and varEst

EJEMPLOS

```
data(woodpecker)
woodpecker
with(subset(woodpecker, rate=='adult'),
  plot(year, start, type='o', pch=16,
    ylab="Numero de adultos", xlab="Año",
    main="Pajaros carpinteros en Water Canyon"))
## tasa de sobrevivencia especifica por estado
x<-aggregate(list(Nstart=woodpecker$start, Nsurv=woodpecker$surv),
  list(stage=woodpecker$rate), sum)
x$survival<-x[,3]/x[,2]
x
```

Índice alfabético

TEMA COLOR		multiresultm,	110
colorguide,	62	net.reproductive.rate,	113
image2,	88	pfister.plot,	115
		pop.projection,	117
TEMA DATASETS		projection.matrix,	119
aq.census,	49	QPmat,	123
aq.trans,	53	reproductive.value,	125
calathea,	60	resample,	126
grizzly,	79	sensitivity,	129
hudcorrs,	82	splitA,	131
hudsonia,	86	stable.stage,	133
hudvrs,	87	stage.vector.plot,	134
monkeyflower,	108	stoch.growth.rate,	136
nematode,	112	stoch.projection,	138
teasel,	144	stoch.quasi.ext,	140
test.census,	145	stretchbetaval,	142
tortoise,	147	var2,	148
whale,	158	varEst,	149
woodpecker,	159	vitalsens,	151
		vitalsim,	154
		apply,	148
TEMA DOCUMENTATION		aq.census,	49, 54
Caswell,	46	aq.matrix,	51
Morris,	48	aq.trans,	50, 53
popbio,	42		
R,	2	betaval,	48, 55, 143
Tema hplot		boot.transitions,	47, 57, 127
matplot2,	104		
Tema manip		calathea,	60
head2,	81	Caswell,	46
		colorguide,	62
TEMA SURVEY		Conceptos sobre (R),	2
aq.matrix,	51	countCDFxt,	48, 64, 74
betaval,	55	cut,	88
boot.transitions,	57		
countCDFxt,	64	damping.ratio,	46, 66
damping.ratio,	66	eigen,	69, 94, 95
eigen.analysis,	68	eigen.analysis,	46, 68, 115
elasticity,	71	elasticity,	47, 71, 129
extCDF,	73	eval,	151
fundamental.matrix,	75	expression,	151
generation.time,	77	extCDF,	48, 65, 73
hudmxdef,	84		
Kendall,	91	Funciones de Caswell (<i>Caswell</i>),	46
lambda,	94	fundamental.matrix,	46, 75, 78, 114
lnorms,	96		
logi.hist.plot,	98	generation.time,	46, 76, 77, 114, 132
LTRE,	100	grizzly,	48, 79
mean.list,	106		

head,	81	rbeta,	56
head2,	81	reproductive.value,	46, 125
hudcorrs,	82, 156	resample,	47, 126
hudmxdef,	84, 156	rlnorm,	96
hudsonia,	86	rowMeans,	106
hudvrs,	84, 87, 156	sample,	136, 138, 140
image,	89	sensitivity,	47, 71, 129
image2,	88	splitA,	75, 77, 113, 126, 131
Introduccion,	2	stable.stage,	46, 133
Kendall,	48, 91, 149, 159	stage.vector.plot,	105, 118, 134
lambda,	46, 66, 94	stoch.growth.rate,	47, 48, 136
legend,	104, 134	stoch.projection,	47, 48, 137, 138, 141
levels,	119	stoch.quasi.ext,	48, 140
list,	136, 138, 140	stretchbetaval,	48, 97, 142
lnorms,	48, 96, 126	tail,	81
logi.hist.plot,	98	teasel,	144
LTRE,	47, 100	test.census,	145
matplot,	105	tortoise,	147
matplot2,	104	unlist,	106, 148
mean,	106	var2,	106, 148
mean.list,	106	varEst,	92, 149, 159
monkeyflower,	108	vitalsens,	48, 151
Morris,	48	vitalsim,	48, 55, 82, 84, 87, 154
Morris (<i>Morris</i>),	48	whale,	47, 158
Multinomial,	126	woodpecker,	159
multiresultm,	47, 48, 110		
names,	151		
nematode,	112		
net.reproductive.rate,	46, 76, 78, 113, 132		
ordered,	51, 119		
pfister.plot,	115		
plot,	104, 134		
pop.projection,	46, 69, 95, 117, 135		
popbio,	42		
popbio (Introduction),	42		
projection.matrix,	46, 51, 52, 57, 58, 119		
QPmat,	46, 112, 123		

