Package 'emdbook'

February 19, 2020

Type Package

Title Support Functions and Data for ``Ecological Models and Data''

Version 1.3.12

LazyData yes

Description Auxiliary functions and data sets for ``Ecological Models and Data'', a book presenting maximum likelihood estimation and related topics for ecologists (ISBN 978-0-691-12522-0).

Suggests R2jags, ellipse, SuppDists, numDeriv, testthat, rgl

Imports MASS, lattice, plyr, coda, bbmle

License GPL

URL http://www.math.mcmaster.ca/bolker/emdbook

NeedsCompilation no

Author Ben Bolker [aut, cre],
Sang Woo Park [ctb],
James Vonesh [dtc],
Jacqueline Wilson [dtc],
Russ Schmitt [dtc],
Sally Holbrook [dtc],
James D. Thomson [dtc],
R. Scot Duncan [dtc]

Maintainer Ben Bolker <bolker@mcmaster.ca>

Repository CRAN

Date/Publication 2020-02-19 16:50:02 UTC

R topics documented:

`emdbook-package` *Support Functions and Data for "Ecological Models and Data"*

Description

Auxiliary functions and data sets for "Ecological Models and Data", a book presenting maximum likelihood estimation and related topics for ecologists (ISBN 978-0-691-12522-0).

References

Bolker, Benjamin M. Ecological Models and Data in R. Princeton University Press, 2008

apply2d *Apply a function to a combination of vectors*

Description

applies a (non-vectorized) function to a combination of vectors; substitute for `outer`

Usage

```
apply2d(fun, x, y, ..., use_plyr = TRUE, .progress="none")
```

Arguments

`fun`	a function of two arguments (or a character string such as `"*"`)
`x`	first vector
`y`	second vector
`...`	additional arguments to `fun`
`use_plyr`	use methods from the `plyr` package?
`.progress`	progress bar type (`"none"`, `"text"`, `"tk"`,`"win"`: see `create_progress_bar`)

Value

a matrix of the function applied to the combinations of the vector values

Author(s)

Ben Bolker

See Also

outer

Examples

```
outer(1:3,1:3)
## this example would work with outer() too
apply2d("*",1:3,1:3)
```

as.mcmc.bugs *Convert WinBUGS output to CODA format*

Description

Converts results of a bugs run (class "bugs") to a form that can be used by CODA (class "mcmc")

Usage

```
as.mcmc.bugs(x)
```

Arguments

x	an object of class bugs (output from bugs())

Value

an object of class mcmc

Author(s)

Ben Bolker

calcslice *Negative log-likelihood slice*

Description

Calculate the negative log-likelihood along a line connecting two mle fits

Usage

```
calcslice(fit1, fit2, fn = fit1@minuslogl, range = c(-0.1, 1.1), n = 400)
```

Arguments

fit1	An mle object
fit2	Another mle object
fn	Negative log-likelihood function
range	Numeric vector: range of parameters to try, where 0 corresponds to coef(fit1) and 1 corresponds to coef(fit2)
n	Number of points to evaluate

Details

Calculates the negative log-likelihood (not a profile, just a "slice") along the line connecting the two sets of coefficients. Intended for diagnosing and visualizing multiple minima in a likelihood surface, especially in higher-dimensional models.

Value

x	Parameter values, along the 0-1 scale described above
y	Negative log-likelihood values

Author(s)

Ben Bolker

`contour3d` *Superimpose contour lines on a 3D plot*

Description

Plot contour lines computed from data in 3D, or add them to an existing 3D (RGL) surface

Usage

```
contour3d(x, y, z, contourArgs=NULL, ...)
```

Arguments

x	numeric vector of x values (as in `contour`), or a list with components x, y and z
y	numeric vector of y values (as in `contour`)
z	numeric z matrix (as in `contour`)
contourArgs	list of arguments to `contourLines`
...	other arguments to `lines3d`

Value

Returns a list of contour lines (as in `contourLines`), invisibly.

Note

You have to install the `rgl` package before you can use this function.

Note

If you are superimposing the contour lines on a surface, it helps to draw the surface with some level of transparency (`alpha` parameter: see `material3d`) so the contour lines are not obscured by the surface.

Author(s)

Ben Bolker

`credint` *Calculate Bayesian credible intervals*

Description

Calculate Bayesian credible intervals based on various types of information about the posterior distribution

Usage

```
tcredint(dist, parlist, ranges, level = 0.95, eps = 1e-05,verbose=FALSE)
ncredint(pvec,npost,level=0.95,tol=0.01,verbose=FALSE)
```

Arguments

`dist`	character string giving the name of a distribution for which "d", "q", and "p" function exist, e.g. "beta"
`parlist`	list of parameters to pass to distribution functions
`ranges`	lower, middle, and upper values to bracket lower and upper boundaries of the credible interval
`level`	confidence level
`eps`	if `ranges` is missing, set lower and upper brackets to the `eps` and `1-eps` quantiles of the distribution
`tol`	tolerance on credible interval
`verbose`	if TRUE, return detailed information on the probability cutoff and realized area of the credible interval; if FALSE, just lower and upper bounds of the credible region
`pvec`	numeric vector of parameter values
`npost`	numeric vector of posterior density values corresponding to `pvec`

Details

`tcredint` gives credible intervals for a theoretical posterior density with defined density, cumulative density, and quantile functions; `ncredint` gives credible intervals for a numerical posterior density.

Value

A numeric vector giving the credible interval. If `verbose=FALSE`, gives just lower and upper bounds; if `verbose=TRUE`, also gives information on the probability cutoff and realized area of the credible interval

Note

For credible intervals from a sample (e.g. from an MCMC run), see `HPDinterval` in the `coda` package.

Author(s)

Ben Bolker

Examples

```
tcredint("beta",list(shape1=5,shape2=10),verbose=TRUE)
pvec = seq(0,1,length=100)
postvec = dbeta(pvec,shape1=5,shape2=10)
ncredint(pvec,postvec,verbose=TRUE)
set.seed(1001)
```

---

curve3d *Plot a 3D surface representing a 2D curve*

Description

Two-dimensional analogue of `curve`: generates a surface and plots it

Usage

```
curve3d(expr, from = c(0, 0), to = c(1, 1), n = c(41, 41),
xlim, ylim, add = FALSE,
xlab=varnames[1],
ylab=varnames[2],
zlab = NULL, log = NULL, sys3d = c("persp", "wireframe", "rgl",
"contour", "image", "none"),
varnames=c("x","y"),use_plyr=TRUE,.progress="none",...)
```

Arguments

`expr`	a mathematical expression using `x` and `y` as the independent variables
`from`	minimum values for `x` and `y`
`to`	maximum values for `x` and `y`
`xlim`	range of values for `x`
`ylim`	range of values for `y`
`n`	number of grid points in each direction
`add`	(logical) add to an existing plot? (only possible for contour plots or rgl)
`xlab`	x label
`ylab`	y label
`zlab`	z label

log	(character): "x", "y", or "xy" for logarithmic axes
sys3d	3D plotting system to use: one of "persp", "wireframe", "rgl", "contour", "image", or "none"
varnames	names of variables to substitute
use_plyr	use methods from the plyr package?
.progress	progress bar type ("none", "text", "tk","win": see create_progress_bar)
...	additional arguments to the plotting functions

Value

invisibly, a list of

x	x values
y	y values
z	z matrix

Note

- You must explicitly install the rgl package (via install.packages("rgl")) before using sys3d="persp".
- if you encounter the error 'Results must have one or more dimensions', try use_plyr=FALSE or use c() to remove attributes from the result of your expression

See Also

outer, curve

Examples

```
curve3d(cos(2*pi*x)+sin(2*pi*y/3),
 from=c(0,0),to=c(1,1))
x <- 1
y <- 3
curve3d(cos(2*pi*x)+sin(2*pi*y/3),
 from=c(0,0),to=c(1,1),sys3d="wireframe")
curve3d(x*cos(2*pi*a/x)+sin(2*pi*b/y),
 from=c(0,0),to=c(1,1),sys3d="wireframe",
varnames=c("a","b"))  ## identical
op <- par(mfrow=c(2,2))
curve3d(cos(2*pi*x)+sin(2*pi*y/3),
 from=c(0,0),to=c(1,1),sys3d="image")
curve3d(x*cos(2*pi*a/x)+sin(2*pi*b/y),
 from=c(0,0),to=c(1,1),sys3d="image",
varnames=c("a","b"))  ## identical
x <- 4
curve3d(cos(2*pi*a/x)+y*sin(2*pi*b/y),
 from=c(0,0),to=c(1,1),sys3d="image",
varnames=c("a","b"))
curve3d(cos(2*pi*x)+sin(2*pi*y/3),
```

```
 from=c(0,0),to=c(1,1),sys3d="image")
curve3d(cos(2*pi*x)+sin(2*pi*y/3),
        sys3d="contour",add=TRUE)
par(op)
```

`Damselfish` *Reef fish (damselfish) data*

Description

Two data sets on *Dascyllus trimaculatus* (three-spot damselfish), one on the distribution of settlement densities to empty anemones across time and space, the other on survival (recruitment) of arriving settlers as a function of experimentally manipulated densities from Schmitt et al. (1999).

Usage

```
data(DamselSettlement)
data(DamselRecruitment)
data(DamselRecruitment_sum)
```

Format

Three data frames:

`site` settlement site (location)

`pulse` monthly settlement pulse

`obs` observation within pulse

`density` density of settlers per 0.1 m2 anemone

`area` anemone area in cm2

`init` initial settler density

`surv` surviving density after 6 months

`settler.den` target experimental density of settlers on experimental anemones

`surv.den` mean surviving density after 6 months, by target density

`SE` standard error of survivor density, by target density

Source

Schmitt et al. (1999), "Quantifying the effects of multiple processes on local abundance", Ecology Letters 2:294-303. DOI: 10.1046/j.1461-0248.1999.00086.x (Original data kindly provided by Schmitt and Holbrook.) The original version of this data set is available from the Moorea Coral Reef LTER data repository.

dbetabinom *Beta-binomial distribution*

Description

Density function and random variate generator for the beta-binomial function, parameterized in terms of probability and overdispersion

Usage

```
dbetabinom(x, prob, size,  theta, shape1, shape2, log = FALSE)
rbetabinom(n, prob, size, theta, shape1, shape2)
```

Arguments

x	a numeric vector of integer values
prob	numeric vector: mean probability of underlying beta distribution
size	integer: number of samples
theta	overdispersion parameter
shape1	shape parameter of per-trial probability distribution
shape2	shape parameter of per-trial probability distribution
log	(logical) return log probability density?
n	integer number of random variates to return

Details

The beta-binomial distribution is the result of compounding a beta distribution of probabilities with a binomial sampling process. The density function is

$$p(x) = \frac{C(N,x)\text{Beta}(x+\theta p, N-x+\theta(1-p))}{\text{Beta}(\theta p, \theta(1-p))}$$

The parameters shape1 and shape2 are the more traditional parameterization in terms of the parameters of the per-trial probability distribution.

Value

A vector of probability densities or random deviates. If x is non-integer, the result is zero (and a warning is given).

Note

Although the quantile (qbetabinom) and cumulative distribution (pbetabinom) functions are not available, in a pinch they could be computed from the pghyper and qghyper functions in the SuppDists package – provided that shape2>1. As described in ?pghyper, pghyper(q,a=-shape1,N=-shape1-shape2,k=s should give the cumulative distribution for the beta-binomial distribution with parameters (shape1,shape2,size), and similarly for qghyper. (Translation to the (theta,size,prob) parameterization is left as an exercise.)

Author(s)

Ben Bolker

References

Morris (1997), American Naturalist 150:299-327; https://en.wikipedia.org/wiki/Beta-binomial_distribution

See Also

dbeta, dbinom

Examples

```
set.seed(100)
n <- 9
z <- rbetabinom(1000, 0.5, size=n, theta=4)
par(las=1,bty="l")
plot(table(z)/length(z),ylim=c(0,0.34),col="gray",lwd=4,
     ylab="probability")
points(0:n,dbinom(0:n,size=n,prob=0.5),col=2,pch=16,type="b")
points(0:n,dbetabinom(0:n,size=n,theta=4,
         prob=0.5),col=4,pch=17,type="b")
## correspondence with SuppDists
if (require(SuppDists)) {
  d1a <- dghyper(0:5,a=-5,N=-10,k=5)
  d1b <- dbetabinom(0:5,shape1=5,shape2=5,size=5)
  max(abs(d1a-d1b))
  p1a <- pghyper(0:5,a=-5,N=-10,k=5,lower.tail=TRUE)
  p1b <- cumsum(d1b)
  max(abs(p1a-p1b))
}
```

---

dchibarsq *Mixed chi-squared distributions*

Description

Calculates "mixed" chi-squared distributions (mixtures of chi-square(n) and chi-square(n-1)); useful for Likelihood Ratio Tests when parameters are on the boundary

Usage

```
dchibarsq(x, df = 1, mix = 0.5, log = FALSE)
pchibarsq(p, df = 1, mix = 0.5, lower.tail=TRUE, log.p = FALSE)
qchibarsq(q, df = 1, mix = 0.5)
rchibarsq(n, df = 1, mix = 0.5)
```

Arguments

x	numeric vector of positive values
p	numeric vector of positive values
q	numeric vector of quantiles (0-1)
n	integer: number of random deviates to pick
df	degrees of freedom (positive integer)
mix	mixture parameter: fraction of distribution that is chi-square(n-1) distributed
log	return log densities?
log.p	return log probabilities?
lower.tail	return lower tail values?

Value

Vectors of probability densities (dchibarsq), cumulative probabilities (pchibarsq), quantiles (qchibarsq), or random deviates (rchibarsq) from Goldman and Whelan's "chi-bar-squared" distribution. qchibarsq uses simple algebra for df=1 and uniroot for df>1.

Author(s)

Ben Bolker

References

N. Goldman and S. Whelan (2000) "Statistical Tests of Gamma-Distributed Rate Heterogeneity in Models of Sequence Evolution in Phylogenetics", Mol. Biol. Evol. 17:975-978. D. O. Stram and J. W. Lee (1994) "Variance Components Testing in the Longitudinal Fixed Effects Model", Biometrics 50:1171-1177.

Examples

```
x <- rchibarsq(100)
plot(density(x,from=0))
curve(dchibarsq(x),add=TRUE,col=2,from=0)
## Not run:
library(lattice)
print(qqmath(~ simdist,
       distribution=qchibarsq,
       panel = function(x, ...) {
         panel.qqmathline(x, ...)
         panel.qqmath(x, ...)
            }))


## End(Not run)
## create first line of table in Goldman and Whelan 2000
round(qchibarsq(c(0.01,0.05,0.9,0.95,0.975,0.99,0.995),df=1),2)
## check second line of table
round(pchibarsq(c(3.81,5.14,6.48,8.27,9.63),df=2),3)
```

```
## create middle column
round(qchibarsq(0.95,df=1:10))
```

---

deltamethod *Delta method functions*

Description

Delta-method implementations for Jensen's inequality and prediction uncertainty

Usage

```
deltamethod(fun, z, var = "x", params = NULL, max.order = 2)
deltavar(fun,meanval=NULL,vars,Sigma,verbose=FALSE)
```

Arguments

fun	Function of one (deltamethod) or more arguments, expressed in raw form (e.g. a*x/(b+x))
z	numeric vector of values
var	variable name
vars	list of variable names: needed if params does not have names, or if some of the values specified in params should be treated as constant
params	list or numeric vector of parameter values to substitute
meanval	possibly named vector of mean values of parameters
Sigma	numeric vector of variances or variance-covariance matrix
max.order	maximum order of delta method to compute
verbose	print details?

Details

deltamethod() is for computing delta-method approximations of the mean of a function of data; deltavar() is for estimating variances of a function based on the mean values and variance-covariance matrix of the parameters. If Sigma is a vector rather than a matrix, the parameters are assumed to be independently estimated.

Value

For deltavar(), a vector of predicted variances; for deltamethod() a vector containing the observed value of the function average, the function applied to the average, and a series of delta-method approximations

Author(s)

Ben Bolker

References

Lyons (1991), "A practical guide to data analysis for physical science students", Cambridge University Press

Examples

```
deltamethod(a*x/(b+x),runif(50),params=list(a=1,b=1),max.order=9)
deltavar(scale*gamma(1+1/shape),meanval=c(scale=0.8,shape=12),
   Sigma=matrix(c(0.015,0.125,0.125,8.97),nrow=2))
## more complex deltavar example
xvec = seq(-4,4,length=101)
x1 = xvec
x2 = xvec
v = matrix(0.2,nrow=3,ncol=3)
diag(v) = 1
m = c(b0=1,b1=1.5,b2=1)
v3  = deltavar(1/(1+exp(-(b0+b1*x1+b2*x2))),meanval=m,Sigma=v)
plot(xvec,v3)
```

---

deprecated *Deprecated (obsolete) functions*

Description

Functions that are obsolete for one reason or another

Author(s)

Ben Bolker

---

dmvnorm *Multivariate normal distribution density function*

Description

Calculates the probability density function of the multivariate normal distribution

Usage

```
dmvnorm(x, mu, Sigma, log = FALSE, tol = 1e-06)
```

Arguments

x	a vector or matrix of multivariate observations
mu	a vector or matrix of mean values
Sigma	a square variance-covariance matrix
log	(logical) return log-likelihood?
tol	tolerance for positive definiteness

Details

uses naive linear algebra – could probably use QR decomposition and/or crossprod.

Value

vector of log-likelihoods

Author(s)

Ben Bolker

See Also

mvrnorm (in MASS package), dmvnorm (in mvtnorm package)

Examples

```
M = matrix(c(1,0.5,0.5,0.5,1,0.5,0.5,0.5,1),nrow=3)
dmvnorm(1:3,mu=1:3,Sigma=M,log=TRUE)
dmvnorm(matrix(1:6,nrow=2),mu=1:3,Sigma=M,log=TRUE)
dmvnorm(matrix(1:6,nrow=2),mu=matrix(1:6,nrow=2),Sigma=M,log=TRUE)
```

---

dzinbinom *Zero-inflated negative binomial distribution*

Description

Probability distribution function and random variate generation for the zero-inflated negative binomial distribution

Usage

```
dzinbinom(x, mu, size, zprob, log=FALSE)
rzinbinom(n, mu, size, zprob)
```

Arguments

x	vector of integer values
n	number of values to draw
mu	mean parameter (or vector of parameters) of negative binomial
size	number of trials/overdispersion parameter (or vector of parameters) of negative binomial
zprob	probability of structural zeros
log	return log probability?

Details

The zero-inflated negative binomial distribution is widely used to model extra zero counts in count data that otherwise follows a negative binomial distribution. The probability distribution is

$$p(0) = p_z + (1 - p_z)NB(0, mu, k)$$

and

$$p(x) = (1 - p_z)NB(x, mu, k)$$

for $x > 0$.

Value

Probabilities of x or random deviates.

Note

Only the "ecological" parameterization is included here (must specify `mu`, not `prob`)

Author(s)

Ben Bolker

References

Tyre et al., "Improving precision and reducing bias in biological surveys: estimating false-negative error rates", Ecological Applications 13:1790-1801 (2003)

See Also

`dnbinom`, Simon Jackman's pscl package

Examples

```
dzinbinom(0:9,mu=2,zprob=0.3,size=0.9)
dnbinom(0:9,mu=2,size=0.9)
rzinbinom(10,mu=2,zprob=0.3,size=0.9)
```

Fir *Data on fir (Abies) life history*

Description

Data on various aspects of life history (diameter at breast height, onset of reproduction, crowding, fecundity) from subalpine *Abies balsamea*, from Dodd and Silvertown

Usage

```
data(FirDBHFec)
data(FirDBHFec_sum)
```

Format

`DBH` diameter in m at breast height (1.4 m)

`fecundity` number of cone rachises [per year?]

`pop` which population (wave, nonwave) an individual was sampled from

`VAR1` location

`WAVE_NON` non-wave (n) or wave (w)

`TREE_NO` tree number

`C1991` 1991 cones

`C1992` 1992 cones

`C1993` 1993 cones

`C1994` 1994 cones

`C1995` 1995 cones

`C1996` 1996 cones

`C1997` 1997 cones

`C1998` 1998 cones

`C1999` 1999 cones

`NOTES_IN` notes

`G1990` 1990 growth

`G1991` 1991 growth

`G1992` 1992 growth

`G1993` 1993 growth

`G1994` 1994 growth

`G1995` 1995 growth

`G1996` 1996 growth

`G1997` 1997 growth

`G1998` 1998 growth

`DBH` diameter at breast height

`DBH_MM` dbh in mm

`DBH_2` ?

`DBH_2MM` ?

`AGE` ?

`GOOD_OR` ?

`PC1998` ?

`AC1998` ?

`PC1994` ?

`AC1994` ?

`R3PC1998` ?

RPC1994 ?

RAC1994 ?

RLOOKOUT ?

RSHREWYS ?

RWILLS a factor with levels 0 1 Fajita

C8TOT ?

G8TOT ?

RAC1994I ?

RPC1994I ?

R3PC1998.1 ?

AC1998I ?

TOTCONES total cones

Source

J. Silvertown and M. Dodd, Evolution of life history in balsam fir (*Abies balsamea*) in subalpine forests, Proc. Roy. Soc. Lond. B (1999) 266, 729-733.

References

M. Dodd and J. Silvertown, Size-specific fecundity and the influence of lifetime size variation upon effective population size in *Abies balsamea*

Examples

```
data(FirDBHFec_sum)
attach(FirDBHFec_sum)
plot(DBH,fecundity,col=as.numeric(pop),pch=as.numeric(pop))
lms = lapply(split(FirDBHFec_sum,pop),lm,formula=fecundity~DBH)
for (i in 1:2) abline(lms[[i]],col=i)
detach(FirDBHFec_sum)
```

---

get.emdbook.packages *install and update auxiliary packages*

Description

convenience function for downloading and installing all the packages needed for the book (just a list and a wrapper around `install.packages`)

Usage

```
get.emdbook.packages()
```

Value

none: installs packages as a side effect

Author(s)

Ben Bolker

See Also

`install.packages`

`GobySurvival` *Goby (reef fish) survivorship data*

Description

Survivorship data from experimental manipulations on gobies *Elacatinus evelynae* and *E. prochilos* in the US Virgin Islands, 2000-2002

Format

`exper` experiment

`year` year

`site` site (factor: backreef, patchreef)

`head` coral head (factor)

`density` treatment "density" (number of "target" fish)

`qual` treatment "quality"; background settlement rate

`d1` last day observed (starting at 1)

`d2` first day not observed

Details

These data have been made available by the author for pedagogical use; out of courtesy, please don't redistribute (outside of the context of this package) or use in an academic publication without requesting permission (via the package maintainer).

Source

J. Wilson, pers. comm.; "Habitat quality, competition and recruitment processes in two marine gobies", Ph.D. thesis, University of Florida (2004); http://uf.catalog.fcla.edu/permalink.jsp?20UF022544524

Examples

```
## midpoint of survival times
gg <- transform(GobySurvival,mid=(d1+d2)/2)
plot(table(gg$mid))
```

`gridsearch2d` *Graphical grid search in 2D*

Description

Given an objective function and starting ranges, computes the values over the ranges and displays them in the graphics window. User can then interactively zoom in to view interesting parts of the surface.

Usage

```
gridsearch2d(fun, v1min, v2min, v1max, v2max,
n1 = 20, n2 = 20, logz = FALSE,
sys3d = c("both", "contour", "image"), ...)
```

Arguments

`fun`	Objective function to be minimized: function of two arguments
`v1min`	Minimum starting value of variable 1
`v2min`	Minimum starting value of variable 2
`v1max`	Maximum starting value of variable 1
`v2max`	Maximum starting value of variable 2
`n1`	Number of grid points for variable 1
`n2`	Number of grid points for variable 2
`logz`	Display image or contour on log scale?
`sys3d`	Display surface as an image, contour, or both?
`...`	Other arguments to `fun`

Details

If `log=TRUE`, the value of the surface is rescaled to `log10(m-min(m)+mindm)`, where `mindm` is the difference between the minimum and the next-largest value (or 1e-10 if this difference is zero).

At each iteration, the user is prompted to select two corners of the new range with the mouse; if this choice is confirmed then the view zooms in. When the user chooses to quit, they are asked whether they want to choose a final point (e.g. an estimate of the minimum) with the mouse.

Value

If a final point is chosen, a list with elements `x` and `y`, otherwise NULL.

Author(s)

Ben Bolker

See Also

`curve3d`

HPDregionplot *Plot highest posterior density region*

Description

Given a sample from a posterior distribution (an mcmc object from the coda package), plot the bivariate region of highest marginal posterior density for two variables, using kde2d from MASS to calculate a bivariate density.

Usage

```
HPDregionplot(x, vars = 1:2, h, n = 50, lump = TRUE, prob = 0.95, xlab =
NULL, ylab = NULL, lims=NULL, ...)
```

Arguments

x	an mcmc or mcmc.list object
vars	which variables to plot: numeric or character vector
h	bandwidth of 2D kernel smoother (previous default value was c(1,1), which worked poorly with some plots with very small scales; if not specified, defaults to values in kde2d)
n	number of points at which to evaluate the density grid
lump	if x is an mcmc.list object, lump the chains together for plotting?
prob	probability level
xlab	x axis label
ylab	y axis label
lims	limits, specified as (x.lower,x.upper,y.lower,y.upper) (passed to kde2d)
...	other arguments to contour

Details

Uses kde2d to calculate a bivariate density, then normalizes the plot and calculates the contour corresponding to a contained volume of prob of the total volume under the surface (a two-dimensional Bayesian credible region).

Value

Draws a plot on the current device, and invisibly returns a list of contour lines (contourLines).

Note

Accuracy may be limited by density estimation; you may need to tinker with h and n (see kde2d in the MASS package).

Author(s)

Ben Bolker

See Also

`HPDinterval` in the coda package, `ellipse` package

Examples

```
library(MASS)
library(coda)
z <- mvrnorm(1000,mu=c(0,0),Sigma=matrix(c(2,1,1,2),nrow=2))
z2 <- mvrnorm(1000,mu=c(0,0),Sigma=matrix(c(2,1,1,2),nrow=2))
HPDregionplot(mcmc(z))
HPDregionplot(mcmc.list(mcmc(z),mcmc(z2)))
```

`lambertW` *Lambert W function*

Description

Computes the Lambert W function, giving efficient solutions to the equation x*exp(x)==z

Usage

```
lambertW_base(z, b = 0, maxiter = 10, eps = .Machine$double.eps, min.imag =
1e-09)
lWasymp(z,logz)
lambertW(z,...)
```

Arguments

`z`	(complex) vector of values for which to compute the function
`logz`	(complex (?)) vector of $log(z)$ values (to be specified by name instead of `z`)
`b`	(integer) b=0 specifies the principal branch, 0 and -1 are the ones that can take non-complex values
`maxiter`	maximum numbers of iterations for convergence
`eps`	convergence tolerance
`min.imag`	maximum magnitude of imaginary part to chop when returning solutions
`...`	arguments to pass to `lambertW_base`

Details

Compute the Lambert W function of z. This function satisfies $W(z)\exp(W(z)) = z$, and can thus be used to express solutions of transcendental equations involving exponentials or logarithms. For $z > 10^3 07$, an asymptotic formula (from Corless et al by way of http://mathworld.wolfram.com/LambertW-Function.html) is used: `lambertW` is a wrapper that automatically selects the asymptotic formula where appropriate.

- In ecology, the Lambert W can be used to solve the so-called "Rogers equation" for predator functional response with depletion.
- In epidemiology, the Lambert W function solves the final-size equation of a simple SIR epidemic model.

Value

Complex or real vector of solutions.

Note

This implementation should return values within 2.5*eps of its counterpart in Maple V, release 3 or later. Please report any discrepancies to the author or translator.

The derivative of the `lambertW` function is `plogis(-lambertW)`.

Author(s)

Nici Schraudolph <schraudo@inf.ethz.ch> (original version (c) 1998), Ben Bolker (R translation)

References

Corless, Gonnet, Hare, Jeffrey, and Knuth (1996), "On the Lambert W Function", Advances in Computational Mathematics 5(4):329-359

See Also

`?Lambert` in the `gsl` package by Robin Hankin, which uses Gnu Scientific Library code; also `?lambertW` in the `VGAM` and `pracma` packages, and the `lambertW` package

Examples

```
curve(lambertW(x),from=0,to=10)
pvec <- seq(-1,1,length=40)
m <- outer(pvec,pvec,function(x,y)Re(lambertW(x+y*1i)))
persp(pvec,pvec,m,
      theta=290,shade=0.5,zlab="lambertW")
num1 <- uniroot(function(x) {x*exp(x)-1},lower=0,upper=1,tol=1e-9)
abs(lambertW(1)-num1$root)<1e-9
###
## Rogers random predator equation:
rogers.pred <- function(N0,a,h,T) {
   N0 - lambertW(a*h*N0*exp(-a*(T-h*N0)))/(a*h)
}
```

```
holling2.pred <- function(N0,a,h) {
  a*N0/(1+a*h*N0)
}
curve(rogers.pred(x,a=1,h=0.2,T=1),from=0,to=60,
  ylab="Number eaten/unit time",xlab="Initial number",ylim=c(0,5),
  main="Predation: a=1, h=0.2")
curve(rogers.pred(x,a=1,h=0.2,T=5)/5,add=TRUE,lty=2,from=0)
curve(rogers.pred(x,a=1,h=0.2,T=0.2)*5,add=TRUE,lty=3,from=0)
curve(rogers.pred(x,a=1,h=0.2,T=10)/10,add=TRUE,lty=4,from=0)
curve(holling2.pred(x,a=1,h=0.2),add=TRUE,lty=1,lwd=2,from=0)
abline(h=5)
legend(30,2,
   c(paste("Rogers, T=",c(0.2,1,5,10),sep=""),
    "Holling type II"),lwd=c(rep(1,4),2),lty=c(3,1,2,4,1))
## final size of an epidemic
finalsize <- function(R0) {
   1+1/R0*lambertW(-R0*exp(-R0))
}
curve(finalsize,from=1,to=10,xlab=expression(R[0]),ylab="Final size")
## comparison of asymptotic results
tmpf <- function(x) {
  L0 <- lambertW_base(10^x)
  L1 <- lWasymp(logz=x*log(10))
  (L1-L0)/L0
}
curve(tmpf,from=1,to=307,log="y")

## derivative
## don't run (avoid numDeriv dependency)
## require(numDeriv)
##   grad(lambertW(1))
##   plogis(-lambertW(1))
```

`Lily` *Glacier lily occurrence and fecundity data*

Description

Data on sample quadrats of the glacier lily, *Erythronium grandiflorum*, from Thomson et al 1996

Usage

```
data(Lily_sum)
```

Format

`x` location of quadrat

`y` location of quadrat

`flowers` number of flowers

`seedlings` number of seedlings

`vegetative` number of vegetative plants

`gopher` index of gopher activity

`rockiness` rockiness index

`moisture` moisture index

`flowcol` inverse quintile of flowering plants

`seedcol` inverse quintile of seedlings

`vegcol` inverse quintile of number of vegetative plants for image plots

`gophcol` inverse quintile of gopher activity

`rockcol` inverse quintile of rockiness

`moiscol` inverse quintile of moisture

Details

16x16 grid of 2x2m quadrats in Washington Gulch, sampled 1992

Source

Thomson et al 1996, "Untangling multiple factors in spatial distributions", Ecology 77:1698-1715. Data from James D. Thomson, with file format conversion help from Jennifer Schmidt

Examples

```
data(Lily_sum)
par(mfrow=c(3,2))
for (i in 9:14) {
  image(matrix(Lily_sum[,i],nrow=16),main=names(Lily_sum)[i])
}
```

`lseq` *Log-spaced sequence*

Description

Generates a logarithmically spaced sequence

Usage

```
lseq(from, to, length.out)
```

Arguments

`from`	starting value
`to`	ending value
`length.out`	number of intervening values

Details

`lseq()` is just a wrapper for `exp(seq(log(from),log(to),length.out = length.out))`

Value

logarithmically spaced sequence

Author(s)

Ben Bolker

See Also

`seq`

Examples

```
lseq(10,1000,9)
```

---

lump.mcmc.list *Utility functions for mcmc objects*

Description

Creates traceplots or combine mcmc list into mcmc objects

Usage

```
lump.mcmc.list(x)
```

Arguments

`x`	an `mcmc.list` object

Value

a single `mcmc` object with the chains lumped together

Author(s)

Ben Bolker

See Also

`coda` package

metropSB *Metropolis-Szymura-Barton algorithm*

Description

Stochastic global optimization using the Metropolis-Szymura-Barton algorithm. New parameters are chosen from a uniform candidate distribution with an adaptively tuned scale, and accepted or rejected according to a Metropolis rule.

Usage

```
metropSB(fn, start, deltap = NULL, scale = 1, rptfreq = -1, acceptscale
= 1.01, rejectscale = 0.99, nmax = 10000,
retvals = FALSE, retfreq = 100, verbose = FALSE, ...)
```

Arguments

fn	Objective function, taking a vector of parameters as its first argument. The function is minimized, so it should be a negative log-likelihood or a negative log-posterior density.
start	Vector of starting values
deltap	Starting jump size; half-width of uniform distribution
scale	Scaling factor for acceptance
rptfreq	Frequency for reporting interim results (<0 means no reporting)
acceptscale	Amount to inflate candidate distribution if last jump was accepted
rejectscale	Amount to shrink candidate distribution if last jump was rejected
nmax	Number of iterations
retvals	Return detailed statistics?
retfreq	Sampling frequency for detailed statistics
verbose	Print status?
...	Other arguments to fn

Details

Metropolis-Szymura-Barton algorithm: given function and starting value, try to find parameters that minimize the function Algorithm: at a given step, 1. pick a new set of parameters, each of which is uniformly distributed in (p[i]-deltap[i],p[i]+deltap[i]) 2. calculate function value at new parameter values 3. if f(new)<f(old), accept 4. if f(new)>f(old), accept with probability (exp(-scale*(f(new)-f(old))) 5. if accept, increase all deltap values by acceptscale; if reject, decrease by rejectscale 6. if better than min so far, save function and parameter values 7. if reject, restore old values

Value

minimum	minimum value achieved
estimate	parameters corresponding to minimum
funcalls	number of function evaluations

If retvals=TRUE:

retvals	matrix of periodic samples including parameters, jump scale, current value, and minimum achieved value

Note

If scale=1 the algorithm satisfies MCMC rules, provided that the other properties of the MC (irreducibility and aperiodicity) are satisfied.

Author(s)

Ben Bolker

References

Szymura and Barton (1986) Genetic analysis of a hybrid zone between the fire-bellied toads,Bombina bombina and B. variegata, near Cracow in southern Poland. Evolution 40(6):1141-1159.

See Also

optim, MCMCmetrop1R (MCMCpack package)

Myxo *Myxomatosis titer data*

Description

Myxomatosis viral titer in blood samples from European rabbits, as a function of day-of-infection and virus grade, from Dwyer et al. 1990, ultimately from Fenner et al. 1956

Usage

```
data(MyxoTiter_sum)
```

Format

grade virus grade (1, least virulent; 5, most virulent)

day day of infection

titer blood virus titer (in log10 rabbit infectious doses)

Note

Pulled graphically from figure in Dwyer et al.; to be replaced (eventually) with original tabular data in Fenner et al.

Source

Dwyer, Levin and Buttel, "A Simulation Model of the Population Dynamics and Evolution of Myxomatosis", Ecological Monographs 60(4):423-447 (1990). Original source: Fenner et al. 1956

Examples

```
data(MyxoTiter_sum)
library(lattice)
xyplot(titer~day|factor(grade),data=MyxoTiter_sum,xlim=c(0,30))
```

---

perturb.params *Create a list of perturbed parameters*

Description

Takes a baseline set of parameters and perturbs it to create a variety of starting points for maximum likelihood estimation or MCMC

Usage

```
perturb.params(base, alt, which, mult = FALSE, use.base = TRUE)
```

Arguments

base	a named list (or vector) of parameters
alt	a list of lists (or vectors) of alternative parameter values or multipliers
which	which parameters to perturb (currently unused)
mult	(logical) multiply baseline values rather than replacing them?
use.base	(logical) include baseline parameters in the list?

Details

Takes the baseline parameter list and substitutes alternative values.

Value

A list of named lists of parameters.

Note

To be extended.

Author(s)

Ben Bolker

Examples

```
perturb.params(list(x=1,y=2,z=3),alt=list(x=c(2,4),z=5))
```

Reedfrog *Data on reed frog predation experiments*

Description

Data on lab experiments on the density- and size-dependent predation rate of an African reed frog, *Hyperolius spinigularis*, from Vonesh and Bolker 2005

Usage

```
data(ReedfrogPred)
data(ReedfrogSizepred)
data(ReedfrogFuncresp)
```

Format

Various data with variables:

`density` initial tadpole density (number of tadpoles in a 1.2 x 0.8 x 0.4 m tank) [experiment 1]

`pred` factor: predators present or absent [experiment 1]

`size` factor: big or small tadpoles [experiment 1]

`surv` number surviving

`propsurv` proportion surviving (=surv/density) [experiment 1]

`TBL` tadpole body length in mm [size-predation experiment]

`Kill` number killed out of 10, in 3 days [size-predation]

`Initial` initial number/density (300 L tank) [functional response]

`Killed` number killed by 3 dragonfly larvae in 14 days [functional response]

Source

Vonesh and Bolker (2005) Compensatory larval responses shift trade-offs associated with predator-induced hatching plasticity. Ecology 86:1580-1591

Examples

```
data(ReedfrogPred)
boxplot(propsurv~size*density*pred,data=ReedfrogPred)
data(ReedfrogSizepred)
data(ReedfrogFuncresp)
```

scinot *Scientific notation as LaTeX/expression()*

Description

Takes a number and returns a version formatted in LaTeX (suitable for use with `Sexpr()` in an Sweave document) or in `expression()` (suitable for plotting), or plots an axis with labels in scientific notation

Usage

```
scinot(x, format = c("latex", "expression"), delim="$",
pref="", ...)
axis.scinot(side,at)
```

Arguments

`x`	a numeric vector (of length 1)
`format`	produce LaTeX or expression() format?
`delim`	delimiter to add at beginning and end (latex only)
`pref`	text to put before expression (expression only)
`side`	side on which to plot axis
`at`	list of locations/labels
`...`	additional arguments to `formatC`

Value

a character vector (if `latex`) or expression (if `expression`); `axis.scinot` draws an axis on the current plot

Author(s)

Ben Bolker

See Also

`formatC`, `expression`, `plotmath`, `axis`, `axTicks`, `latexSN` in the `Hmisc` package, `eaxis` in the `sfsmisc` package

Examples

```
scinot(1e-5)
scinot(1e-5,digits=0)
scinot(1e-5,"expression")
scinot(1e-5,"expression",pref="p=")
set.seed(1001)
```

```
plot(1:100,rlnorm(100,0,2),log="y",axes=FALSE)
axis(side=1)
axis.scinot(side=2)  ## fix bug!
```

SeedPred *Seed predation data set from Duncan and Duncan 2000*

Description

Data on seed predation over time from Duncan and Duncan (2000)

Usage

```
data(SeedPred)
data(SeedPred_wide)
data(SeedPred_mass)
```

Format

`station` a factor specifying the station number

`species` a factor with levels `abz` `cd` `cor` `dio` `mmu` `pol` `psd` `uva`

`date` sample date

`seeds` number of seeds present

`tcum` cumulative time elapsed

`tint` time since last sample

`taken` seeds removed since last sample

`dist` distance from forest edge (m)

Details

`SeedPred` is in long format, `SeedPred_wide` is in wide format; `SeedPred_wide` has lots of `NA` values because stations at 10 and 25 m from the forest were sampled on different days. `SeedPred_mass` is a numeric vector containing the approximate seed masses for each species.

Source

R. Scot Duncan and Virginia E. Duncan (2000) Forest Succession and Distance from Forest Edge in an Afro-Tropical Grassland, Biotropica 32(1):33-41. (Data from Scot Duncan.)

Examples

```
data(SeedPred)
```

`trcoef` *Transform coefficients*

Description

Perform standard transformations of coefficients based on information encoded in the names or the `transf` attribute of the vector or list

Usage

```
trcoef(x, inverse = FALSE)
```

Arguments

`x`	A numeric vector of coefficients with names and/or a `transf` attribute
`inverse`	(logical) Perform inverse transform?

Details

If `inverse=FALSE` and coefficient names begin with "logit", "log", or "sqrt" the function will back-transform them (using `plogis`, `exp`, or squaring), strip the descriptor from the names, and set the `transf` attribute. Naturally, `inverse=TRUE` will do the opposite. If the `transf` attribute is all empty strings after an inverse transformation, it will be deleted.

Value

A vector of transformed variables with modified names and `transf` attributes.

Author(s)

Ben Bolker

Examples

```
x = c(loga=1,logitb=2,sqrtc=2)
trx = trcoef(x); trx
trcoef(trx,inverse=TRUE)
```

Index