Package 'BMAmevt'

October 12, 2022

Type Package

Title Multivariate Extremes: Bayesian Estimation of the Spectral Measure

Version 1.0.4

Date 2019-12-17

Depends stats, utils, coda

Description Toolkit for Bayesian estimation of the dependence structure in multivariate extreme value parametric models, following Sabourin and Naveau (2014) <doi:10.1016/j.csda.2013.04.021> and Sabourin, Naveau and Fougeres (2013) <doi:10.100
012-0163-0>.

License GPL (>= 2)

LazyData no

NeedsCompilation yes

Repository CRAN

Encoding UTF-8

Collate 'BMAmevt-package.R' 'Laplace.evt.r' 'MCpriorIntFun.nl.r' 'MCpriorIntFun.pb.r' 'MCpriorIntFun.r' 'add.frame.r' 'transf.to.equi.r' 'cons.angular.dat.r' 'ddirimix.grid.r' 'ddirimix.grid1D.r' 'ddirimix.r' 'dgridplot.R' 'diagnose.r' 'discretize.R' 'dnestlog.grid.r' 'dnestlog.r' 'dpairbeta.grid.r' 'dpairbeta.r' 'emp_density.r' 'excessProb.condit.dm.r' 'excessProb.nl.r' 'excessProb.pb.r' 'lAccept.ratio.r' 'marginal.lkl.nl.r' 'marginal.lkl.pb.r' 'marginal.lkl.r' 'maxLikelihood.r' 'posterior.predictive.nl.r' 'posterior.predictive.pb.r' 'posterior.predictive3D.r' 'posteriorDistr.bma.r' 'posteriorMCMC.nl.r' 'posteriorMCMC.pb.r' 'posteriorMCMC.r' 'posteriorMean.r' 'posteriorWeights.r' 'prior.nl.r' 'prior.pb.r' 'proposal.nl.r' 'proposal.pb.r' 'rdirimix.r' 'rect.integrate.r' 'rnestlog.r' 'rpairbeta.r' 'rstable.posit.R' 'scores3D.r' 'transf.to.rect.r' 'wrapper.R' 'zzz.r'

RoxygenNote 7.0.2

Author Leo Belzile [cre] (<https://orcid.org/0000-0002-9135-014X>),
Anne Sabourin [aut] (<https://orcid.org/0000-0002-5096-9157>)

Maintainer Leo Belzile <belzilel@gmail.com>

Date/Publication 2019-12-18 22:40:10 UTC

R topics documented:

`BMAmevt-package` *Bayesian Model Averaging for Multivariate Extremes*

Description

Toolkit for Bayesian estimation of the dependence structure in Multivariate Extreme Value parametric models, with possible use of Bayesian model Averaging techniques

Details

Package:	BMAmevt
Type:	Package
Version:	1.0
Date:	2012-03-1
License:	GPL-2
LazyData:	no

Includes a Generic MCMC sampler. Estimation of the marginal distributions is a prerequisite, *e.g.* using one of the packages `ismev`, `evd`, `evdbayes` or `POT`. This package handles data sets which are assumed to be marginally unit-Frechet distributed.

Author(s)

Anne Sabourin <sabourin@math.univ-lyon1.fr>

See Also

`evdbayes`

add.frame *Adds graphical elements to a plot of the two dimensional simplex.*

Description

Adds graphical elements to the current plot (on the two-dimensional simplex).

Usage

```
add.frame(
  equi = FALSE,
  lab1 = "w1",
  lab2 = "w2",
  lab3 = "w3",
  npoints = 60,
  col.polygon = "black",
  axes = TRUE
)
```

Arguments

equi	logical. Is the simplex represented as an equilateral triangle (if TRUE) or a right triangle (if FALSE) ?
lab1	Character string: label for first component.
lab2	Character string: label for second component.
lab3	Character string: label for third component.
npoints	The number of grid nodes on the squared grid containing the desired triangle.
col.polygon	The background color outside the simplex.
axes	logical. Should axes be added ?

Details

Generic graphical tool for obtaining nice plots of the two-dimensional simplex

Examples

```
plot.new()
add.frame()
plot.new()
mult.x=sqrt(2); mult.y=sqrt(3/2)
plot.window( xlim=c(0,mult.x),ylim=c(0,mult.y), asp=1,bty ="n")
add.frame(equi=TRUE)
```

cons.angular.dat *Angular data set generation from unit Frechet data.*

Description

Builds an angular data set, retaining the points with largest radial component.

Usage

```
cons.angular.dat(
  coordinates = c(1, 2, 3),
  frechetDat = get("frechetdat"),
  n = 100,
  displ = TRUE,
  invisible = TRUE,
  add = FALSE,
  lab1 = "w1",
  lab2 = "w2",
  lab3 = "w3",
  npoints = 60,
  col.polygon = "white",
  ...
)
```

Arguments

coordinates	Index vector of the columns in frechetDat to be retained to construct the angular data set.
frechetDat	The data set. A matrix: each row is a multivariate record. May contain *NA*'s.
n	The number of desired observations in the final angular data set. Should be less than nrow(frechetDat)
displ	logical. Should the angular data set be plotted ?
invisible	logical. Should the result be returned as invisible ?
add	logical. Only used when displ==TRUE. Should the points be added to the current plot ?
lab1	Character string: label for first component.
lab2	Character string: label for second component.
lab3	Character string: label for third component.
npoints	The number of grid nodes on the squared grid containing the desired triangle.
col.polygon	The background color outside the simplex.
...	Additional graphical parameters and arguments to be passed to function plot.window and points.

Details

The data set `frechetDat` is assumed to be marginally unit Frechet distributed.

Value

The angular data set: A `n*length(coordinates)` matrix, containing values between zero and one, which rows sum to one: Each row is thus a point on the unit simplex of dimension `length(coordinates)-1`. Returned as invisible if `invisible==TRUE`.

Examples

```
## Not run: cons.angular.dat()
```

`ddirimix` *Angular density/likelihood function in the Dirichlet Mixture model.*

Description

Likelihood function (spectral density on the simplex) and angular data sampler in the Dirichlet mixture model.

Usage

```
ddirimix(
  x = c(0.1, 0.2, 0.7),
  par,
  wei = par$wei,
  Mu = par$Mu,
  lnu = par$lnu,
  log = FALSE,
  vectorial = FALSE
)

rdirimix(
  n = 10,
  par = get("dm.expar.D3k3"),
  wei = par$wei,
  Mu = par$Mu,
  lnu = par$lnu
)
```

Arguments

`x`	An angular data set which may be reduced to a single point: A $n * p$ matrix or a vector of length `p`, where p is the dimension of the sample space and n is the sample size. Each row is a point on the simplex, so that each row sum to one. The error tolerance is set to `1e-8` in this package.

par	The parameter list for the Dirichlet mixture model.
wei	Optional. If present, overrides the value of par$wei.
Mu	Optional. If present, overrides the value of par$Mu.
lnu	Optional. If present, overrides the value of par$lnu.
log	Logical: should the density or the likelihood be returned on the log-scale ?
vectorial	Logical: Should a vector of size n or a single value be returned ?
n	The number of angular points to be generated

Details

The spectral probability measure defined on the simplex characterizes the dependence structure of multivariate extreme value models. The parameter list for a mixture with k components, is made of

Mu The density kernel centers $\mu_{i,m}, 1 \leq i \leq p, 1 \leq m \leq k$: A $p * k$ matrix, which columns sum to one, and such that Mu %*% wei=1, for the moments constraint to be satisfied. Each column is a Dirichlet kernel center.

wei The weights vector for the kernel densities: A vector of k positive numbers summing to one.

lnu The logarithms of the shape parameters $nu_m, 1 \leq m \leq k$ for the density kernels: a vector of size k.

The moments constraint imposes that the barycenter of the columns in Mu, with weights wei, be the center of the simplex.

Value

ddirimix returns the likelihood as a single number if vectorial ==FALSE, or as a vector of size nrow(x) containing the likelihood of each angular data point. If log == TRUE, the log-likelihood is returned instead. rdirimix returns a matrix with n points and p=nrow(Mu) columns.

---

ddirimix.grid *Plots the Dirichlet mixture density on a discretization grid*

Description

Only valid in the tri-variate case

Usage

```
ddirimix.grid(
  par = get("dm.expar.D3k3"),
  wei = par$wei,
  Mu = par$Mu,
  lnu = par$lnu,
  npoints = 30,
  eps = 10^(-3),
  equi = TRUE,
```

```
  marginal = TRUE,
  coord = c(1, 2, 3),
  invisible = TRUE,
  displ = TRUE,
  ...
)
```

Arguments

par	The parameter list for the Dirichlet mixture model.
wei	Optional. If present, overrides the value of par$wei.
Mu	Optional. If present, overrides the value of par$Mu.
lnu	Optional. If present, overrides the value of par$lnu.
npoints	The number of grid nodes on the squared grid containing the desired triangle.
eps	Positive number: minimum distance from any node inside the simplex to the simplex boundary
equi	logical. Is the simplex represented as an equilateral triangle (if TRUE) or a right triangle (if FALSE) ?
marginal	logical. If TRUE, the angular density corresponds to the marginal intensity measure, over coordinates coord. Otherwise, it is only the projection of the full dimensional angular measure (hence the moments constraints is not satisfied anymore).
coord	A vector of size 3: the indices of the coordinates upon which the marginalization is to be done.
invisible	Logical: should the result be returned as invisible ?
displ	Logical: should a plot be issued ?
...	Additional arguments to be passed to dgridplot.

Value

The discretized density

---

ddirimix.grid1D *Univariate projection or marginalization of a Dirichlet mixture density on on* [0,1]

Description

Plots a univariate Dirichlet mixture (in other words, a Beta mixture) angular density for extreme bi-variate data.

Usage

```
ddirimix.grid1D(
  par = get("dm.expar.D2k4"),
  wei = par$wei,
  Mu = par$Mu,
  lnu = par$lnu,
  npoints = 30,
  eps = 10^(-3),
  coord = c(1, 2),
  marginal = TRUE,
  invisible = TRUE,
  displ = TRUE,
  add = FALSE,
  ...
)
```

Arguments

par	The parameter list for the Dirichlet mixture model.
wei	Optional. If present, overrides the value of par$wei.
Mu	Optional. If present, overrides the value of par$Mu.
lnu	Optional. If present, overrides the value of par$lnu.
npoints	number of points on the 1D discretization grid.
eps	the minimum value (= 1- the maximum value) of the grid points.
coord	A vector of size 2: the indices of the coordinates upon which the marginalization or projection is to be done if the dimension of the sample space is greater than two.
marginal	logical. If TRUE, the angular density corresponds to the marginal intensity measure of the extreme Poisson process, over coordinates coord. Otherwise, it is only the projection of the full dimensional angular measure (hence the moments constraints is not satisfied anymore).
invisible	Logical: should the result be returned as invisible ?
displ	Logical: should a plot be issued ?
add	Logical: should the density be added to the currently active plot ?
...	Additional arguments to be passed to plot

Value

The discretized density on [eps, 1-eps] (included in [0,1])

dgridplot *Image and/or Contour plots of spectral densities in trivariate extreme value models*

Description

Plots contours or gray-scale level sets of a spectral density on the two-dimensional simplex.

Usage

```
dgridplot(
  density = matrix(5 * sin(1/73 * (1:(40 * 40)))^2, ncol = 40, nrow = 40),
  eps = 10^(-3),
  equi = TRUE,
  add = FALSE,
  breaks = seq(-0.01, 5.1, length.out = 1000),
  levels = seq(0, 6, length.out = 13),
  col.lines = "black",
  labcex = 0.8,
  background = FALSE,
  col.polygon = gray(0.5),
  lab1 = "w1",
  lab2 = "w2",
  lab3 = "w3",
  ...
)
```

Arguments

density	A npoints*npoints matrix containing the density's values scattered on the discretization grid defined by npoints, equi, eps (see discretize).
eps	Positive number: minimum distance from any node inside the simplex to the simplex boundary
equi	logical. Is the simplex represented as an equilateral triangle (if TRUE) or a right triangle (if FALSE) ?
add	Logical. Should the contours be added to a currently active plot ?
breaks	Set of breakpoints for the gray scale colors. See image
levels	Levels to which plot the contour lines. See contour
col.lines	The color to be used for the contour lines.
labcex	cex for contour labeling. See contour.
background	Logical. Should a the background be filled inside the simplex *via* a call to image ?
col.polygon	The background color outside the simplex.
lab1	Character string: label for first component.

lab2	Character string: label for second component.
lab3	Character string: label for third component.
...	Additional graphical parameters and arguments to be passed to contour and image.

Details

The function interprets the density matrix as contour does, *i.e.* as a table of f(X[i], Y[j]) values, with column 1 at the bottom, where X and Y are returned by discretize and f is the density function.

Examples

```
wrapper <- function(x, y, my.fun,...)
      {
       sapply(seq_along(x), FUN = function(i) my.fun(x[i], y[i],...))
      }

grid <- discretize(npoints=40,eps=1e-3,equi=FALSE)

Density <- outer(grid$X,grid$Y,FUN=wrapper,
                 my.fun=function(x,y){10*((x/2)^2+y^2)*((x+y)<1)})

dgridplot(density= Density,npoints=40, equi=FALSE)
```

---

diagnose *Diagnostics for the MCMC output in the PB and NL models.*

Description

The method issues several convergence diagnostics, in the particular case when the PB or the NL model is used. The code may be easily modified for other angular models.

Usage

```
diagnose(obj, ...)

## S3 method for class 'PBNLpostsample'
diagnose(
  obj,
  true.par = NULL,
  from = NULL,
  to = NULL,
  autocor.max = 0.2,
  default.thin = 50,
  xlim.density = c(-4, 4),
  ylim.density = NULL,
  plot = TRUE,
```

```
  predictive = FALSE,
  save = TRUE,
  ...
)
```

Arguments

obj	an object of class postsample: posterior sample, as produced by posteriorMCMC.pb or posteriorMCMC.nl
...	Additional parameters to be passed to the functions posterior.predictive.pb or posterior.predictive.nl.
true.par	The true parameter. If NULL, it is considered as unknown.
from	Integer or NULL. If NULL, the default value is used. Otherwise, should be greater than post.sample$Nbin. Indicates the index where the averaging process should start. Default to post.sample$Nbin +1
to	Integer or NULL. If NULL, the default value is used. Otherwise, must be lower than Nsim+1. Indicates where the averaging process should stop. Default to post.sample$Nsim.
autocor.max	The maximum accepted auto-correlation for two successive parameters in the thinned sample.
default.thin	The default thinning interval if the above condition cannot be satisfied.
xlim.density	The xlim interval for the density plots, on the transformed scale.
ylim.density	the ylim intervals for the density plots.
plot	Logical. Should plots be issued ?
predictive	Logical. Should the predictive density be plotted ?
save	Logical: should the result be saved ? Only used if the posterior sample has been saved itself (*i.e.* if it contains save=TRUE in its arguments list)

Value

A list made of

predictive The posterior predictive, or 0 if predictive=FALSE

effective.size the effective sample size of each component

heidelTest The first part of the Heidelberger and Welch test (stationarity test). The first row indicates "success" (1) or rejection(0), the second line shows the number of iterations to be discarded, the third line is the p-value of the test statistic.

gewekeTest The test statistics from the Geweke stationarity test.

gewekeScore The p-values for the above test statistics

thin The thinning interval retained

correl.max.thin The maximum auto-correlation for a lag equal to thin

linked.est.mean The posterior mean of the transformed parameter (on the real line)

linked.est.sd The standard deviation of the transformed parameters

est.mean The posterior mean of the original parameters, as they appears in the expression of the likelihood

sample.sd the posterior standard deviation of the original parameters

---

`discretize` *Discretization grid builder.*

Description

Builds a discretization grid covering the two-dimensional unit simplex, with specified number of points and minimal distance from the boundary.

Usage

```
discretize(npoints = 40, eps = 0.001, equi = FALSE)
```

Arguments

`npoints`	The number of grid nodes on the squared grid containing the desired triangle.
`eps`	Positive number: minimum distance from any node inside the simplex to the simplex boundary
`equi`	logical. Is the simplex represented as an equilateral triangle (if `TRUE`) or a right triangle (if `FALSE`) ?

Details

The `npoints*npoints` grid covers either the equilateral representation of the simplex, or the right angled one. In any case, the grid is *rectangular*: some nodes lie outside the triangle. Density computations on such a grid should handle the case when the point passed as argument is outside the simplex (typically, the function should return zero in such a case).

Value

A list containing two elements: `X` and `Y`, vectors of size `npoints`, the Cartesian coordinates of the grid nodes.

Note

In case `equi==TRUE`, `epsilon` is the minimum distance from any node inside the simplex to the simplex boundary, *after transformation* to the right-angled representation.

dm.expar.D3k3 *Example of valid Dirichlet mixture parameter for tri-variate extremes.*

Description

The Dirichlet mixture density has three components, the center of mass of the three columns of `Mu`, with weights `wei` is $(1/3, 1/3, 1/3)$: the centroid of the two dimensional unit simplex.

Format

A list made of

Mu A $3*3$ matrix, which rows sum to one, such that the center of mass of the three column vectors (weighted with `wei`) is the centroid of the simplex: each column is the center of a Dirichlet mixture component.

wei A vector of length three, summing to one: the mixture weights

lnu A vector of length three: the logarithm of the concentration parameters.

dnestlog *Pairwise Beta (PB) and Nested Asymmetric Logistic (NL) distributions*

Description

Likelihood function (spectral density) and random generator in the Pairwise Beta and NL models.

Usage

```
dnestlog(
  x = rbind(c(0.1, 0.3, 0.6), c(0.3, 0.3, 0.4)),
  par = c(0.5, 0.5, 0.2, 0.3),
  log = FALSE,
  vectorial = TRUE
)

dpairbeta(
  x,
  par = c(1, rep(2, choose(4, 2) + 1)),
  log = FALSE,
  vectorial = TRUE
)

rnestlog(
  n = 5,
  par = c(0.2, 0.3, 0.4, 0.5),
  threshold = 1000,
```

```
  return.points = FALSE
)

rpairbeta(n = 1, dimData = 3, par = c(1, rep(1, 3)))
```

Arguments

x	An angular data set (may be reduced to a single point). A npoints*dimData matrix (or a vector of length(dimData). For the NL model, dimData is always 3. Each row is a point on the simplex, so that the sum of each rows should equal 1 (the error tolerance is set to 1e-8 in this package).
par	The parameter for the Pairwise Beta or the Nested Logistic density. • In the Pairwise Beta model, par is of length choose(p,2)+1. The first element is the global dependence parameter, the subsequent ones are the pairwise dependence parameters, in lexicographic order (*e.g.* $\beta_{12}, \beta_{13}, \beta_{23}$). • In the NL model, par is a vector of length four with components between zero and one. The first one is the global dependence parameter, the three subsequent ones are the pairwise dependence parameters, again in lexicographic order.
log	Logical. Should the density be returned on the log scale ?
vectorial	Logical. Should a vector or a single value be returned ?
n	The number of points on the simplex to be generated.
threshold	The radial threshold r above which the simulated points should be kept to build the angular dataset. Should be set to a high value, for the asymptotic approximation $$P(W \in B \mid \|\|X\|\| > r) \simeq H(B)$$ to hold.
return.points	logical: should the censored vectorial dataset corresponding to the angular one be returned ?
dimData	the dimension of the sample space, which is $1+$ the dimension of the simplex.

Details

Applies to angular data sets. The density is given with respect to the Lebesgue measure on R^{p-1}, where p is the number of columns in x (or the length of x, if the latter is a single point).

Value

The value returned by the likelihood function is imposed (see *e.g.* posteriorMCMC. In contrast, the random variable have unconstrained output format.

- dpairbeta returns the likelihood as a single number if vectorial ==FALSE, or as a vector of size nrow(x) containing the likelihood of each angular data point. If log == TRUE, the log-likelihood is returned instead. rpairbeta returns a matrix with n rows and dimData columns.

- dnestlog returns the likelihood as a single number if vectorial ==FALSE, or as a vector of size nrow(x) containing the likelihood of each angular data point. If log == TRUE, the log-likelihood is returned instead. rnestlog returns a matrix with n rows and dimData columns if return.points==FALSE (the default). Otherwise, a list is returned, with two elements:
  - Angles: The angular data set
  - Points: The full tri-variate data set above threshold (*i.e.* Angles multiplied by the radial components)

---

dnestlog.grid *PB and NL spectral densities on the two-dimensional simplex*

Description

The two functions compute respectively the NL and PB spectral densities, in the three-dimensional case, on a discretization grid. A plot is issued (optional).

Usage

```
dnestlog.grid(
  par,
  npoints = 50,
  eps = 0.001,
  equi = TRUE,
  displ = TRUE,
  invisible = TRUE,
  ...
)

dpairbeta.grid(
  par,
  npoints = 50,
  eps = 0.001,
  equi = TRUE,
  displ = TRUE,
  invisible = TRUE,
  ...
)
```

Arguments

par	The parameter for the Pairwise Beta or the Nested Logistic density.

- In the Pairwise Beta model, par is of length choose(p,2)+1. The first element is the global dependence parameter, the subsequent ones are the pairwise dependence parameters, in lexicographic order (*e.g.* $\beta_{12}, \beta_{13}, \beta_{23}$).

- In the NL model, par is a vector of length four with components between zero and one. The first one is the global dependence parameter, the three subsequent ones are the pairwise dependence parameters, again in lexicographic order.

npoints	The number of grid nodes on the squared grid containing the desired triangle.
eps	Positive number: minimum distance from any node inside the simplex to the simplex boundary
equi	logical. Is the simplex represented as an equilateral triangle (if TRUE) or a right triangle (if FALSE) ?
displ	logical. Should a plot be produced ?
invisible	logical. If TRUE, the result is returned as invisible.
...	Additional arguments to be passed to dgridplot

Value

A npoints*npoints matrix containing the considered density's values on the grid. The row (resp. column) indices increase with the first (resp. second) coordinate on the simplex.

Note

If equi==TRUE, the density is relative to the Hausdorff measure on the simplex itself: the values obtained with equi = FALSE are thus divided by $\sqrt{3}$.

Examples

```
dpairbeta.grid(par=c( 0.8, 8, 5, 2),
npoints=70, eps = 1e-3, equi = TRUE, displ = TRUE, invisible=TRUE)

##  or ...

Dens <- dpairbeta.grid(par=c(0.8, 8, 5, 2),
npoints=70, eps = 1e-3, equi = TRUE, displ = FALSE)
Grid=discretize(npoints=70,eps=1e-3,equi=TRUE)
dev.new()
image(Grid$X, Grid$Y, Dens)
contour(Grid$X, Grid$Y, Dens, add=TRUE)
add.frame(equi=TRUE, npoints=70, axes=FALSE)
```

excessProb.condit.dm *Probability of joint threshold exceedance, in the Dirichlet Mixture model, given a DM parameter.*

Description

simple MC integration on the simplex.

Usage

```
excessProb.condit.dm(
  N = 100,
  par = get("dm.expar.D3k3"),
  thres = rep(100, 3),
  plot = FALSE,
  add = FALSE
)
```

Arguments

N	The number of MC iterations to be performed
par	the DM parameter, as a list
thres	the multivariate threshold
plot	logical: should convergence diagnostic plots be issued ?
add	logical: should the plot be added to a current one ?

Value

a list made of

mean the mean estimate from the MC sample

esterr the estimated standard deviation of the estimator

estsd The estimated standard deviation of the MC sample

excessProb.condit.nl *Probability of joint threshold excess in the NL model*

Description

Probability of joint threshold excess in the NL model

Usage

```
excessProb.condit.nl(par = c(0.3, 0.4, 0.5, 0.6), thres = rep(100, 3))
```

Arguments

par	The Nested logistic parameter: of length four.
thres	a positive vector of size three.

Value

The approximate probability of joint excess, valid when at least one coordinate of thres is large

---

excessProb.nl *Posterior distribution the probability of joint threshold excess, in the NL model.*

Description

Posterior distribution the probability of joint threshold excess, in the NL model.

Usage

```
excessProb.nl(
  post.sample,
  from = NULL,
  to = NULL,
  thin = 100,
  thres = rep(100, 3),
  known.par = FALSE,
  true.par,
  displ = FALSE
)
```

Arguments

post.sample	The posterior sample, as returned by posteriorMCMC
from	Integer or NULL. If NULL, the default value is used. Otherwise, should be greater than post.sample$Nbin. Indicates the index where the averaging process should start. Default to post.sample$Nbin +1
to	Integer or NULL. If NULL, the default value is used. Otherwise, must be lower than Nsim+1. Indicates where the averaging process should stop. Default to post.sample$Nsim.
thin	Thinning interval.
thres	a positive vector of size three.
known.par	logical. Is the true parameter known ?
true.par	The true parameter, only used if known.par=TRUE
displ	logical. Should a plot be produced ?

Value

A list made of

whole The output of `posteriorMean` called with `FUN=excessProb.condit.nl`.

mean The posterior mean of the excess probability

esterr The standard deviation of the mean estimator

estsd The standard deviation of the excess probability, in the posterior sample.

lowquant The lower 0.1 quantile of the empirical posterior distribution of the excess probability

upquant The upper 0.1 quantile of the empirical posterior distribution of the excess probability

true `NULL` if `known.par=FALSE`, otherwise the excess probability in the true model.

`excessProb.pb` *Estimates the probability of joint excess (Frechet margins)*

Description

Double Monte-Carlo integration.

Usage

```
excessProb.pb(
  post.sample,
  Nmin.intern = 100,
  precision = 0.05,
  from = NULL,
  to = NULL,
  thin = 100,
  displ = FALSE,
  thres = rep(500, 5),
  known.par = FALSE,
  true.par
)
```

Arguments

`post.sample`	The posterior sample.
`Nmin.intern`	The minimum number of MC iteration in the internal loop (excess probability, conditional to a parameter).
`precision`	The desired precision for the internal MC estimate
`from`	Integer or `NULL`. If `NULL`, the default value is used. Otherwise, should be greater than `post.sample$Nbin`. Indicates the index where the averaging process should start. Default to `post.sample$Nbin +1`
`to`	Integer or `NULL`. If `NULL`, the default value is used. Otherwise, must be lower than `Nsim+1`. Indicates where the averaging process should stop. Default to `post.sample$Nsim`.

thin	Thinning interval.
displ	logical. Should a plot be produced ?
thres	A multivariate threshold
known.par	Logical
true.par	The true parameter from which the data are issued.

Value

A list made of

whole A vector of estimated excess probabilities, one for each element of the thinned posterior sample.

mean the estimated threshold excess probability: mean estimate.

esterr The estimated standard deviation of the mean estimate (where the Monte-Carlo error is neglected)

estsd The estimated standard deviation of the posterior sample (where the Monte-Carlo error is neglected)

lowquants The three lower 0.1 quantiles of, respectively, the conditional mean estimates and of the upper and lower bounds of the Gaussian (centered) 80 % confidence intervals around the conditional estimates.

upquants The three upper 0.9 quantiles

true.est the mean estimate conditional to the true parameter: a vector of size three: the mean estimate , and the latter +/- the standard deviation of the estimate

---

expfunction.nl *Exponent function in the NL model.*

Description

The exponent function V for a max-stable variable M is such that $P(M < x) = exp(-V(x))$

Usage

```
expfunction.nl(par = c(0.3, 0.4, 0.5, 0.6), x = 10 * rep(1, 3))
```

Arguments

par	The parameter for the NL distribution, respectively of length two or four.
x	A vector of three extended positive real numbers

Value

the value of $V(x)$ for $x = thres$.

frechetdat *Multivariate data set with margins following unit Frechet distribution.*

Description

Five-variate dataset which margins follow unit-Frechet distributions, obtained from `winterdat` by probability integral transform. Marginal estimation was performed by maximum likelihood estimation of a Generalized Pareto distribution over marginal thresholds corresponding to 0.7 quantiles, following Cooley *et.al.* (see reference below). The "non extreme" part of the marginal distributions was approximated by the empirical distribution function.

Format

A $601 * 5$ - matrix:

References

COOLEY, D., DAVIS, R. and NAVEAU, P. (2010). The pairwise beta distribution: A flexible parametric multivariate model for extremes. *Journal of Multivariate Analysis 101, 2103-2117*.

invlogit *Inverse logit transformation*

Description

Inverse transformation of the `logit` function.

Usage

```
invlogit(x)
```

Arguments

x	A real number

Value

A number between 0 and 1.

`laplace.evt` *Laplace approximation of a model marginal likelihood by Laplace approximation.*

Description

Approximation of a model marginal likelihood by Laplace method.

Usage

```
laplace.evt(
  mode = NULL,
  npar = 4,
  likelihood,
  prior,
  Hpar,
  data,
  link,
  unlink,
  method = "L-BFGS-B"
)
```

Arguments

`mode`	The parameter vector (on the "unlinked" scale, *i.e.* before transformation to the real line) which maximizes the posterior density, or `NULL`.
`npar`	The size of the parameter vector. Default to four.
`likelihood`	The likelihood function, *e.g.* `dpairbeta` or `dnestlog`
`prior`	The prior density (takes an "unlinked" parameter as argument and returns the density of the `linked` parameter)
`Hpar`	The prior hyper parameter list.
`data`	The angular dataset
`link`	The link function, from the "classical" or "unlinked" parametrization onto the real line. (*e.g.* `log` for the PB model, an `logit` for the NL model)
`unlink`	The inverse link function (*e.g.* `exp` for the PB model and `invlogit` for the NL model)
`method`	The optimization method to be used. Default to `"L-BFGS-B"`.

Details

The posterior mode is either supplied, or approximated by numerical optimization. For an introduction about Laplace's method, see *e.g.* Kass and Raftery, 1995 and the references therein.

Value

A list made of

mode the parameter (on the unlinked scale) deemed to maximize the posterior density. This is equal to the argument if the latter is not null.

value The value of the posterior, evaluated at `mode`.

laplace.llh The logarithm of the estimated marginal likelihood

invHess The inverse of the estimated hessian matrix at `mode`

References

KASS, R.E. and RAFTERY, A.E. (1995). Bayes Factors. *Journal of the American Statistical Association, Vol. 90, No.430*

`Leeds` *Tri-variate 'angular' data set approximately distributed according to a multivariate extremes angular distribution*

Description

The data set is constructed from coordinates (columns) $1, 2, 3$ of `frechetdat`. It contains 100 angular points corresponding to the tri-variate vectors $V = (X, Y, Z)$ with largest L^1 norm ($||V|| = X + Y + Z$). The angular points are obtained by 'normalizing': *e.g.*, $x = X/||V||$. Thus, each row in `Leeds` is a point on the two-dimensional simplex : $x + y + z = 1$.

Format

A $100 * 3$ - matrix.

References

COOLEY, D., DAVIS, R. and NAVEAU, P. (2010). The pairwise beta distribution: A flexible parametric multivariate model for extremes. *Journal of Multivariate Analysis 101, 2103-2117*

RESNICK, S. (1987). Extreme values, regular variation, and point processes, *Applied Probability. A, vol. 4, Series of the Applied Probability Trust. Springer-Verlag, New York*.

Leeds.frechet *Multivariate data set with margins following unit Frechet distribution.*

Description

The data set contains 590 (transformed) daily maxima of five air pollutants recorded in Leeds (U.K.) during five winter seasons (1994-1998). Contains NA's. Marginal transformation to unit Frechet was performed by Cooley *et.al.* (see reference below).## $x = X/||V||$. Thus,

Format

A $590 * 5$ - matrix:

References

COOLEY, D., DAVIS, R. and NAVEAU, P. (2010). The pairwise beta distribution: A flexible parametric multivariate model for extremes. *Journal of Multivariate Analysis 101, 2103-2117*

logit *Logit transformation*

Description

Bijective Transformation from $(0, 1)$ to the real line, defined by $logit(p) = log(p/(1-p))$.

Usage

```
logit(p)
```

Arguments

p	A real number in $[0, 1]$

Value

A real number

`marginal.lkl` *Marginal model likelihood*

Description

Estimates the marginal likelihood of a model, proceeding by simple Monte-Carlo integration under the prior distribution.

Usage

```
marginal.lkl(
  dat,
  likelihood,
  prior,
  Nsim = 300,
  displ = TRUE,
  Hpar,
  Nsim.min = Nsim,
  precision = 0,
  show.progress = floor(seq(1, Nsim, length.out = 20))
)
```

Arguments

`dat`	The angular data set relative to which the marginal model likelihood is to be computed
`likelihood`	The likelihood function of the model. See `posteriorMCMC` for the required format.
`prior`	The prior distribution: of type `function(type=c("r","d"),n ,par, Hpar, log, dimData )`, where `dimData` is the dimension of the sample space (*e.g.*, for the two-dimensional simplex (triangle), `dimData=3`. Should return either a matrix with `n` rows containing a random parameter sample generated under the prior (if `type == "d"`), or the density of the parameter `par` (the logarithm of the density if `log==TRUE`. See `prior.pb` and `prior.nl` for templates.
`Nsim`	Total number of iterations to perform.
`displ`	logical. If `TRUE`, a plot is produced, showing the temporal evolution of the cumulative mean, with approximate confidence intervals of $+/-2$ estimated standard errors.
`Hpar`	A list containing Hyper-parameters to be passed to `prior`.
`Nsim.min`	The minimum number of iterations to be performed.
`precision`	the desired relative precision. See `MCpriorIntFun`.
`show.progress`	An vector of integers containing the times (iteration numbers) at which a message showing progression will be printed on the standard output.

Details

The function is a wrapper calling MCpriorIntFun with parameter FUN set to likelihood.

Value

The list returned by MCpriorIntFun. The estimate is the list's element named emp.mean.

Note

The estimated standard deviations of the estimates produced by this function should be handled with care:For "larger" models than the Pairwise Beta or the NL models, the likelihood may have infinite second moment under the prior distribution. In such a case, it is recommended to resort to more sophisticated integration methods, *e.g.* by sampling from a mixture of the prior and the posterior distributions. See the reference below for more details.

References

KASS, R. and RAFTERY, A. (1995). Bayes factors. *Journal of the american statistical association , 773-795*.

See Also

marginal.lkl.pb, marginal.lkl.nl for direct use with the implemented models.

Examples

```
## Not run:
  lklNL=  marginal.lkl(dat=Leeds,
                likelihood=dnestlog,
                prior=prior.nl,
                Nsim=20e+3,
                displ=TRUE,
                Hpar=nl.Hpar,
               )

## End(Not run)
```

---

marginal.lkl.nl *Marginal likelihoods of the PB and NL models.*

Description

Wrappers for marginal.lkl, in the specific cases of the PB and NL models, with parameter likelihood set to dpairbeta or dnestlog, and prior set to prior.pb or prior.nl. See MCpriorIntFun for more details.

Usage

```
marginal.lkl.nl(
  dat,
  Nsim = 10000,
  displ = TRUE,
  Hpar = get("nl.Hpar"),
  Nsim.min = Nsim,
  precision = 0,
  show.progress = floor(seq(1, Nsim, length.out = 20))
)

marginal.lkl.pb(
  dat,
  Nsim = 10000,
  displ = TRUE,
  Hpar = get("pb.Hpar"),
  Nsim.min = Nsim,
  precision = 0,
  show.progress = floor(seq(1, Nsim, length.out = 20))
)
```

Arguments

dat	The angular data set relative to which the marginal model likelihood is to be computed
Nsim	Total number of iterations to perform.
displ	logical. If TRUE, a plot is produced, showing the temporal evolution of the cumulative mean, with approximate confidence intervals of $+/-2$ estimated standard errors.
Hpar	A list containing Hyper-parameters to be passed to prior.
Nsim.min	The minimum number of iterations to be performed.
precision	the desired relative precision. See MCpriorIntFun.
show.progress	An vector of integers containing the times (iteration numbers) at which a message showing progression will be printed on the standard output.

Value

The list returned by marginal.lkl, *i.e.*, the one returned by MCpriorIntFun

See Also

marginal.lkl, MCpriorIntFun .

Examples

```
## Not run:
```

```
marginal.lkl.pb(dat=Leeds ,
         Nsim=20e+3 ,
         displ=TRUE, Hpar = get("pb.Hpar") ,
          )

marginal.lkl.nl(dat=Leeds ,
         Nsim=10e+3 ,
         displ=TRUE, Hpar = get("nl.Hpar") ,
          )

## End(Not run)
```

`maxLikelihood` *Maximum likelihood optimization*

Description

Maximum likelihood optimization

Usage

```
maxLikelihood(
  data,
  model,
  init = NULL,
  maxit = 500,
  method = "L-BFGS-B",
  hess = T,
  link,
  unlink
)
```

Arguments

`data`	The angular data to be used for inference
`model`	A list made of **likelihood** The likelihood function, see `dpairbeta` for a template. **npar** The length of the parameter vector
`init`	NULL or a real vector of size `model$npar` giving the initial values for `link{par}`.
`maxit`	maximum number of iterations to be performed by function `optim`
`method`	The method to be used by `optim`
`hess`	logical: should an approximation of the hessian be performed ?
`link`	the link function from the natural marginal parameter spaces to the real line.
`unlink`	the inverse link function. If `x` is any real number, then `unlink(x)` should be in the admissible range for the likelihood function and the prior function.

Value

The list returned by optim and the AIC and BIC criteria

MCpriorIntFun *Generic Monte-Carlo integration of a function under the prior distribution*

Description

Simple Monte-Carlo sampler approximating the integral of FUN with respect to the prior distribution.

Usage

```
MCpriorIntFun(
  Nsim = 200,
  prior,
  Hpar,
  dimData,
  FUN = function(par, ...) {     as.vector(par) },
  store = TRUE,
  show.progress = floor(seq(1, Nsim, length.out = 20)),
  Nsim.min = Nsim,
  precision = 0,
  ...
)
```

Arguments

Nsim	Maximum number of iterations
prior	The prior distribution: of type function(type=c("r","d"),n ,par, Hpar, log, dimData), where dimData is the dimension of the sample space (*e.g.*, for the two-dimensional simplex (triangle), dimData=3. Should return either a matrix with n rows containing a random parameter sample generated under the prior (if type == "d"), or the density of the parameter par (the logarithm of the density if log==TRUE. See prior.pb and prior.nl for templates.
Hpar	A list containing Hyper-parameters to be passed to prior.
dimData	The dimension of the model's *sample* space, on which the parameter's dimension may depend. Passed to prior inside MCintegrateFun
FUN	A function to be integrated. It may return a vector or an array.
store	Should the successive evaluations of FUN be stored ?
show.progress	same as in posteriorMCMC
Nsim.min	The minimum number of iterations to be performed.
precision	The desired relative precision ϵ. See **Details** below.
...	Additional arguments to be passed to FUN.

Details

The algorithm exits after n iterations, based on the following stopping rule : n is the minimum number of iteration, greater than `Nsim.min`, such that the relative error is less than the specified `precision`.

$$max(est.esterr(n)/|est.mean(n)|) \leq \epsilon,$$

where $est.mean(n)$ is the estimated mean of `FUN` at time n, $est.err(n)$ is the estimated standard deviation of the estimate: $est.err(n) = \sqrt{est.var(n)/(nsim-1)}$. The empirical variance is computed component-wise and the maximum over the parameters' components is considered.

The algorithm exits in any case after `Nsim` iterations, if the above condition is not fulfilled before this time.

Value

A list made of

- `stored.vals` : A matrix with `nsim` rows and `length(FUN(par))` columns.
- `elapsed` : The time elapsed during the computation.
- `nsim` : The number of iterations performed
- `emp.mean` : The desired integral estimate: the empirical mean.
- `emp.stdev` : The empirical standard deviation of the sample.
- `est.error` : The estimated standard deviation of the estimate (*i.e.* $emp.stdev/\sqrt{(nsim)}$).
- `not.finite` : The number of non-finite values obtained (and discarded) when evaluating `FUN(par,...)`

Author(s)

Anne Sabourin

`MCpriorIntFun.nl` *Generic Monte-Carlo integration under the prior distribution in the PB and NL models.*

Description

Wrappers for `MCpriorIntFun` with argument `prior=prior.pb` or `prior=prior.nl`

Usage

```
MCpriorIntFun.nl(
  Nsim = 200,
  FUN = function(par, ...) {     par },
  store = TRUE,
  Hpar = get("nl.Hpar"),
  show.progress = floor(seq(1, Nsim, length.out = 20)),
  Nsim.min = Nsim,
```

```
  precision = 0,
  ...
)

MCpriorIntFun.pb(
  Nsim = 200,
  Hpar = get("pb.Hpar"),
  dimData = 3,
  FUN = function(par, ...) {     as.vector(par) },
  store = TRUE,
  show.progress = floor(seq(1, Nsim, length.out = 20)),
  Nsim.min = Nsim,
  precision = 0,
  ...
)
```

Arguments

Nsim	Maximum number of iterations
FUN	A function to be integrated. It may return a vector or an array.
store	Should the successive evaluations of FUN be stored ?
Hpar	Hyper-parameters for the PB prior (in MCpriorIntFun.pb) or the NL prior (MCpriorIntFun.nl). See pb.Hpar and nl.Hpar for the required formats.
show.progress	same as in posteriorMCMC
Nsim.min	The minimum number of iterations to be performed.
precision	The desired relative precision ϵ. See **Details** below.
...	Additional arguments to be passed to FUN.
dimData	Only for the PB model: The dimension of model's *sample* space. The PB parameter space is of dimension choose(dimData,2)+1. The NL model implemented here is restricted to three-dimensional sample spaces.

Value

The list returned by function MCpriorIntFun.

See Also

MCpriorIntFun

nl.Hpar — *Default hyper-parameters for the NL model.*

Description

The logit-transformed parameters for the NL model are *a priori* Gaussian. The list has the same format as pb.Hpar.

Format

A list of four parameters:

mean.alpha, sd.alpha Mean and standard deviation of the normal prior distribution for the logit-transformed global dependence parameter $alpha$. Default to $0, 3$.

mean.beta, sd.beta Idem for the pairwise dependence parameters.

nl.MCpar — *Default MCMC tuning parameter for the Nested Asymmetric logistic model.*

Description

The proposals (on the logit-scale) are Gaussian, centered aroud the current value.

Format

A list made of a single element: sd. The standard deviation of the normal proposition kernel centered at the (logit-transformed) current state. Default to 0.35.

pb.Hpar — *Default hyper-parameters for the Pairwise Beta model.*

Description

The log-transformed dependence parameters are a priori independent, Gausian. This list contains the means and standard deviation for the prior distributions.

Format

A list of four parameters:

mean.alpha Mean of the log-transformed global dependence parameter. Default to 0)

sd.alpha Standard deviation of the log-transformed global dependence parameter. Default to 3.

mean.beta Mean of each of the log-transformed pairwise dependence parameters. Default to 0)

sd.beta Standard deviation of each of the log-transformed pairwise dependence parameters. Default to 3.

pb.MCpar *Default MCMC tuning parameter for the Pairwise Beta model.*

Description

The proposal for the log-transformed parameters are Gaussian, centered at the current value.

Format

A list made of a single element: `sd`, the standard deviation of the normal proposition kernel (on the log-transformed parameter). Default to 0.35.

posterior.predictive.nl *Posterior predictive densities in the three dimensional PB, NL and NL3 models*

Description

Wrappers for `posterior.predictive3D` in the PB and NL models.

Usage

```
posterior.predictive.nl(
  post.sample,
  from = post.sample$Nbin + 1,
  to = post.sample$Nsim,
  thin = 50,
  npoints = 40,
  eps = 0.001,
  equi = T,
  displ = T,
  ...
)

posterior.predictive.pb(
  post.sample,
  from = post.sample$Nbin + 1,
  to = post.sample$Nsim,
  thin = 50,
  npoints = 40,
  eps = 10^(-3),
  equi = T,
  displ = T,
  ...
)
```

Arguments

post.sample	A posterior sample as returned by posteriorMCMC
from	Integer or NULL. If NULL, the default value is used. Otherwise, should be greater than post.sample$Nbin. Indicates the index where the averaging process should start. Default to post.sample$Nbin +1
to	Integer or NULL. If NULL, the default value is used. Otherwise, must be lower than Nsim+1. Indicates where the averaging process should stop. Default to post.sample$Nsim.
thin	Thinning interval.
npoints	The number of grid nodes on the squared grid containing the desired triangle.
eps	Positive number: minimum distance from any node inside the simplex to the simplex boundary
equi	logical. Is the simplex represented as an equilateral triangle (if TRUE) or a right triangle (if FALSE) ?
displ	logical. Should a plot be produced ?
...	Additional graphical parameters and arguments to be passed to contour and image.

Details

The posterior predictive density is approximated by averaging the densities corresponding to the parameters stored in post.sample. See posterior.predictive3D for details.

Value

A npoints*npoints matrix: the posterior predictive density.

See Also

posterior.predictive3D, posteriorMCMC.pb.

---

posterior.predictive3D

Posterior predictive density on the simplex, for three-dimensional extreme value models.

Description

Computes an approximation of the predictive density based on a posterior parameters sample. Only allowed in the three-dimensional case.

Usage

```
posterior.predictive3D(
  post.sample,
  densityGrid,
  from = post.sample$Nbin + 1,
  to = post.sample$Nsim,
  thin = 40,
  npoints = 40,
  eps = 10^(-3),
  equi = T,
  displ = T,
  ...
)
```

Arguments

post.sample	A posterior sample as returned by posteriorMCMC
densityGrid	A function returning a npoints*npoints matrix, representing a discretized version of the spectral density on the two dimensional simplex. The function should be compatible with dgridplot. In particular, it must use discretize to produce the discretization grid. It must be of type function(par, npoints, eps, equi, displ,invisible,...). See **Details** below.
from	Integer or NULL. If NULL, the default value is used. Otherwise, should be greater than post.sample$Nbin. Indicates the index where the averaging process should start. Default to post.sample$Nbin +1
to	Integer or NULL. If NULL, the default value is used. Otherwise, must be lower than Nsim+1. Indicates where the averaging process should stop. Default to post.sample$Nsim.
thin	Thinning interval.
npoints	The number of grid nodes on the squared grid containing the desired triangle.
eps	Positive number: minimum distance from any node inside the simplex to the simplex boundary
equi	logical. Is the simplex represented as an equilateral triangle (if TRUE) or a right triangle (if FALSE) ?
displ	logical. Should a plot be produced ?
...	Additional graphical parameters and arguments to be passed to contour and image.

Details

The posterior predictive density is approximated by averaging the densities produced by the function densityGrid(par, npoints, eps, equi, displ,invisible, ...) for par in a subset of the parameters sample stored in post.sample. The arguments of densityGrid must be

- par: A vector containing the parameters.

- npoints, eps, equi: Discretization parameters to be passed to dgridplot.
- displ: logical. Should a plot be produced ?
- invisible: logical. Should the result be returned as invisible ?
- ... additional arguments to be passed to dgridplot

Only a sub-sample is used: one out of thin parameters is used (thinning). Further, only the parameters produced between time from and time to (included) are kept.

Value

A npoints*npoints matrix: the posterior predictive density.

Note

The computational burden may be high: it is proportional to npoints^2. Therefore, the function assigned to densityGridplot should be optimized, typically by calling .C with an internal, user defined C function.

Author(s)

Anne Sabourin

See Also

dgridplot, posteriorMCMC.

---

posteriorDistr.bma *Posterior distribution in the average model*

Description

Builds an empirical distribution defined as a sum of weighted Dirac masses from posterior samples in individual models.

Usage

```
posteriorDistr.bma(postweights = c(0.5, 0.5), post.distrs = list())
```

Arguments

postweights	a vector of positive real numbers, summing to one: the posterior weights (in the same order as the elements of post.distrs) of the individual models.
post.distrs	A list of same length as postweights. Each element must be a vector which will be used as a posterior sample.

Value

A matrix with two rows and as many columns as the sum of the lengths of the elements of `post.distrs`. The second line contains the weighted posterior sample in the BMA; the first line contains the weights to be assigned to each corresponding value on the second one.

`posteriorMCMC` *MC MC sampler for parametric spectral measures*

Description

Generates a posterior parameters sample, and computes the posterior mean and component-wise variance on-line.

Usage

```
posteriorMCMC(
  prior = function(type = c("r", "d"), n, par, Hpar, log, dimData) {     NULL },
 proposal = function(type = c("r", "d"), cur.par, prop.par, MCpar, log) {    NULL },
  likelihood = function(x, par, log, vectorial) {     NULL },
  Nsim,
  dat,
  Hpar,
  MCpar,
  Nbin = 0,
  par.start = NULL,
  show.progress = floor(seq(1, Nsim, length.out = 20)),
  seed = NULL,
  kind = "Mersenne-Twister",
  save = FALSE,
  class = NULL,
  name.save = NULL,
  save.directory = "~",
  name.dat = "",
  name.model = ""
)
```

Arguments

`prior`	The prior distribution: of type `function(type=c("r","d"),n ,par, Hpar, log, dimData )`, where `dimData` is the dimension of the sample space (*e.g.*, for the two-dimensional simplex (triangle), `dimData=3`. Should return either a matrix with `n` rows containing a random parameter sample generated under the prior (if `type == "d"`), or the density of the parameter `par` (the logarithm of the density if `log==TRUE`. See `prior.pb` and `prior.nl` for templates.

proposal	The proposal function: of type function(type = c("r","d"),cur.par, prop.par, MCpar, log) . Should return the (logarithm of) the proposal density for the move cur.par --> prop.par if type == "d". If type =="r", proposal must return a candidate parameter, depending on cur.par, as a vector. See proposal.pb or proposal.nl for templates.
likelihood	The likelihood function. Should be of type function(x, par, log, vectorial), where log and vectorial are logical flags indicating respectively if the result is to be returned on the log-scale, and if the value is a vector of length nrow(x) or a single number (the likelihood, or the log-likelihood, for the data set x). See dpairbeta or dnestlog for templates.
Nsim	Total number of iterations to perform.
dat	An angular data set, *e.g.* constructed by cons.angular.dat: A matrix which rows are the Cartesian coordinates of points on the unit simplex (summing to one).
Hpar	A list containing Hyper-parameters to be passed to prior.
MCpar	A list containing MC MC tuning parameters to be passed to proposal.
Nbin	Length of the burn-in period.
par.start	Starting point for the MC MC sampler.
show.progress	An vector of integers containing the times (iteration numbers) at which a message showing progression will be printed on the standard output.
seed	The seed to be set *via* set.seed.
kind	The kind of random numbers generator. Default to "Mersenne-Twister". See set.seed for details.
save	Logical. Should the result be saved ?
class	Optional character string: additional class attribute to be assigned to the result. A predefined class "PBNLpostsample" exists, for which a method performing convergence diagnostics is defined (see diagnose)
name.save	A character string giving the name under which the result is to be saved. If NULL (default), nothing is saved. Otherwise, the result is saved in file paste(save.directory,"/",name.save,".rda",sep=""). A "log" list is also saved, named paste(name.save, ".log", sep=""), in file paste(save.directory,"/", name.log,".rda",sep="").
save.directory	A character string giving the directory where the result is to be saved (without trailing slash).
name.dat	A character string naming the data set used for inference. Default to "".
name.model	A character string naming the model. Default to "".

Value

A list made of

- stored.vals: A (Nsim-Nbin)*d matrix, where d is the dimension of the parameter space.

- `llh` A vector of size `(Nsim-Nbin)` containing the loglikelihoods evaluated at each parameter of the posterior sample.
- `lprior` A vector of size `(Nsim-Nbin)` containing the logarithm of the prior densities evaluated at each parameter of the posterior sample.
- `elapsed`: The time elapsed, as given by `proc.time` between the start and the end of the run.
- `Nsim`: The same as the passed argument
- `Nbin`: idem.
- `n.accept`: The total number of accepted proposals.
- `n.accept.kept`: The number of accepted proposals after the burn-in period.
- `emp.mean` The estimated posterior parameters mean
- `emp.sd` The empirical posterior sample standard deviation.

See Also

`posteriorMCMC.pb`, `posteriorMCMC.pb` for specific uses in the PB and the NL models.

Examples

```
data(Leeds)
data(pb.Hpar)
data(pb.MCpar)
postsample1 <- posteriorMCMC(Nsim=1e+3,Nbin=500,
         dat= Leeds,
         prior = prior.pb,
         proposal = proposal.pb,
         likelihood = dpairbeta,
         Hpar=pb.Hpar,
         MCpar=pb.MCpar)

dim(postsample1[[1]])
postsample1[-1]

## Not run:
## a more realistic one:

postsample2 <- posteriorMCMC(Nsim=50e+3,Nbin=15e+3,
         dat= Leeds,
         prior = prior.pb,
         proposal = proposal.pb,
         likelihood = dpairbeta,
         Hpar=pb.Hpar,
         MCpar=pb.MCpar)
dim(postsample2[[1]])
postsample2[-1]

## End(Not run)
```

posteriorMCMC.nl *MC MC posterior samplers for the the PB and the NL model.*

Description

The functions generate parameters samples approximating the posterior distribution in the PB model or the NL model.

Usage

```
posteriorMCMC.nl(Nsim, dat, Hpar, MCpar, ...)

posteriorMCMC.pb(Nsim, dat, Hpar, MCpar, ...)
```

Arguments

Nsim	Total number of iterations to perform.
dat	An angular data set, *e.g.* constructed by cons.angular.dat: A matrix which rows are the Cartesian coordinates of points on the unit simplex (summing to one).
Hpar	A list containing Hyper-parameters to be passed to prior.
MCpar	A list containing MC MC tuning parameters to be passed to proposal.
...	Additional arguments to be passed to posteriorMCMC instead of their default values (must not contain any of "prior","likelihood", "proposal","name.model" or "class").

Details

The two functions are wrappers simplifying the use of posteriorMCMC for the two models implemented in this package.

Value

an object with class attributes "postsample" and "PBNLpostsample": The posterior sample and some statistics as returned by function posteriorMCMC

Note

For the Leeds data set, and for simulated data sets with similar features, setting Nsim=50e+3 and Nbin=15e+3 is enough (possibly too much), with respect to the Heidelberger and Welch tests implemented in heidel.diag.

See Also

posteriorMCMC

Examples

```
## Not run:
data(Leeds)
data(pb.Hpar)
data(pb.MCpar)
data(nl.Hpar)
data(nl.MCpar)
pPB <- posteriorMCMC.pb(Nsim=5e+3, dat=Leeds, Hpar=pb.Hpar,
MCpar=pb.MCpar)

dim(pPB[1])
pPB[-(1:3)]

pNL <- posteriorMCMC.nl(Nsim=5e+3, dat=Leeds, Hpar=nl.Hpar,
MCpar=nl.MCpar)

dim(pNL[1])
pNL[-(1:3)]

## End(Not run)
```

`posteriorMean` *Posterior predictive density on the simplex, for three-dimensional extreme value models.*

Description

Computes an approximation of the posterior mean of a parameter functional, based on a posterior parameters sample.

Usage

```
posteriorMean(
  post.sample,
  FUN = function(par, ...) {     par },
  from = NULL,
  to = NULL,
  thin = 50,
  displ = TRUE,
  ...
)
```

Arguments

`post.sample`	A posterior sample as returned by `posteriorMCMC`
`FUN`	a parameter functional returning a vector.

from	Integer or NULL. If NULL, the default value is used. Otherwise, should be greater than post.sample$Nbin. Indicates the index where the averaging process should start. Default to post.sample$Nbin +1
to	Integer or NULL. If NULL, the default value is used. Otherwise, must be lower than Nsim+1. Indicates where the averaging process should stop. Default to post.sample$Nsim.
thin	Thinning interval.
displ	logical. Should a plot be produced ?
...	Additional parameters to be passed to FUN.

Details

Only a sub-sample is used: one out of thin parameters is used (thinning). Further, only the parameters produced between time from and time to (included) are kept.

Value

A list made of

values A matrix : each column is the result of FUN applied to a parameter from the posterior sample.

est.mean The posterior mean

est.sd The posterior standard deviation

See Also

posteriorMCMC.

---

posteriorWeights *Posterior model weights*

Description

Approximates the models' posterior weights by simple Monte Carlo integration

Usage

```
posteriorWeights(
  dat,
  HparList = list(get("pb.Hpar"), get("nl.Hpar")),
  lklList = list(get("dpairbeta"), get("dnestlog")),
  priorList = list(get("prior.pb"), get("prior.nl")),
  priorweights = c(0.5, 0.5),
  Nsim = 20000,
  Nsim.min = 10000,
  precision = 0.05,
  seed = 1,
```

```
  kind = "Mersenne-Twister",
  show.progress = floor(seq(1, Nsim, length.out = 10)),
  displ = FALSE
)
```

Arguments

dat	The angular data set relative to which the marginal model likelihood is to be computed
HparList	A list containing the hyper Parameter for the priors in each model. (list of lists).
lklList	A list containing the likelihood functions of each model
priorList	A list containing the prior definitions of each model.
priorweights	A vector of positive weights, summing to one: the prior marginal weights of each model.
Nsim	The maximum number of iterations to be performed.
Nsim.min	The minimum number of iterations to be performed.
precision	the desired relative precision. See MCpriorIntFun.
seed	The seed to be set *via* set.seed.
kind	The kind of random numbers generator. Default to "Mersenne-Twister". See set.seed for details.
show.progress	An vector of integers containing the times (iteration numbers) at which a message showing progression will be printed on the standard output.
displ	Logical. Should convergence monitoring plots be issued ?

Details

if J is the number of models, the posterior weights are given by

$$postW(i) = priorW(i) * lkl(i)/(\sum_{j=1,\ldots,J} priorW(j) * lkl(j)),$$

where $lkl(i)$ stands for the Monte-Carlo estimate of the marginal likelihood of model i and $priorW(i)$ is the prior weight defined in priorweights[i]. For more explanations, see the reference below The confidence intervals are obtained by adding/subtracting two times the estimated standard errors of the marginal likelihood estimates. The latter are only estimates, which interpretation may be misleading: See the note in section marginal.lkl

Value

A matrix of 6 columns and length(priorweights) rows. The columns contain respectively the posterior model weights (in the same order as in priorweights), the lower and the upper bound of the confidence interval (see **Details**), the marginal model weights, the estimated standard error of the marginal likelihood estimators, and the number of simulations performed.

References

HOETING, J., MADIGAN, D., RAFTERY, A. and VOLINSKY, C. (1999). Bayesian model averaging: A tutorial. *Statistical science 14, 382-401*.

Examples

```
data(pb.Hpar)
data(nl.Hpar)
set.seed(5)
mixDat=rbind(rpairbeta(n=10,dimData=3, par=c(0.68,3.1,0.5,0.5)),
  rnestlog(n=10,par=c(0.68,0.78, 0.3,0.5)))
posteriorWeights (dat=mixDat,
                  HparList=list(get("pb.Hpar"),get("nl.Hpar")),
                  lklList=list(get("dpairbeta"), get("dnestlog")),
                  priorList=list(get("prior.pb"), get("prior.nl")),
                  priorweights=c(0.5,0.5),
                  Nsim=1e+3,
                  Nsim.min=5e+2, precision=0.1,
                  displ=FALSE)
## Not run: posteriorWeights (dat=mixDat,
                  HparList=list(get("pb.Hpar"),get("nl.Hpar")),
                  lklList=list(get("dpairbeta"), get("dnestlog")),
                  priorList=list(get("prior.pb"), get("prior.nl")),
                  priorweights=c(0.5,0.5),
                  Nsim=20e+3,
                  Nsim.min=10e+3, precision=0.05,
                  displ=TRUE)
## End(Not run)
```

---

prior.nl *Prior parameter distribution for the NL model*

Description

Density and generating function of the prior distribution.

Usage

```
prior.nl(type = c("r", "d"), n, par, Hpar, log, dimData = 3)
```

Arguments

type	One of the character strings "r", "d"
n	The number of parameters to be generated. Only used if type == "r".
par	A vector of length four, with component comprised between 0 and 1 (both end points excluded for the first element and 1 included for the others): The parameter where the density is to be taken. Only used if type=="d". In the NL model, par is of length 4. The first element is the global dependence parameter, the others are partial dependence parameter between pairs (12), (13), (23) respectively. In the NL model, par is of length 4. The first element has the same interpretation as in the NL model, the subsequent ones are dependence parameters between

Hpar	list of Hyper-parameters : see nl.Hpar for a template.
log	logical. Should the density be returned on the log scale ? Only used if type=="d"
dimData	The dimension of the sample space, equal to 3. Only for compatibility with *e.g.* posteriorMCMC.

Details

The four parameters are independent, the logit-transformed parameters follow a normal distribution.

Value

Either a matrix with n rows containing a random parameter sample generated under the prior (if type == "d"), or the (log)-density of the parameter par.

Author(s)

Anne Sabourin

Examples

```
## Not run: prior.nl(type="r", n=5 ,Hpar=get("nl.Hpar"))

## Not run: prior.trinl(type="r", n=5 ,Hpar=get("nl.Hpar"))
## Not run: prior.pb(type="d", par=rep(0.5,2), Hpar=get("nl.Hpar"))
```

---

prior.pb *Prior parameter distribution for the Pairwise Beta model*

Description

Density and generating function of the prior distribution.

Usage

```
prior.pb(type = c("r", "d"), n, par, Hpar, log, dimData)
```

Arguments

type	One of the character strings "r", "d"
n	The number of parameters to be generated. Only used if type == "r".
par	A vector with positive components: The parameter where the density is to be taken. Only used if type=="d". In the Pairwise Beta model, par is of length choose(p,2)+1. The first element is the global dependence parameter, the subsequent ones are the pairwise dependence parameters, in lexicographic order (*e.g.* $\beta_{1,2}, \beta_{1,3}, \beta_{2,3}$.
Hpar	list of Hyper-parameters : see pb.Hpar for a template.
log	logical. Should the density be returned on the log scale ? Only used if type=="d"
dimData	The dimension of the sample space. (one more than the dimension of the simplex)

Details

The parameters components are independent, log-normal.

Value

Either a matrix with n rows containing a random parameter sample generated under the prior (if type == "d"), or the (log)-density of the parameter par.

Author(s)

Anne Sabourin

Examples

```
## Not run: prior.pb(type="r", n=5 ,Hpar=get("pb.Hpar"), dimData=3 )
## Not run: prior.pb(type="d", par=rep(1,choose(4,2), Hpar=get("pb.Hpar"), dimData=4 )
```

---

proposal.nl *NL3 model: proposal distribution.*

Description

Density of the proposal distribution q(cur.par,prop.par) and random generator for MC MC algorithm in the NL3 model.

Usage

```
proposal.nl(
  type = c("r", "d"),
  cur.par,
  prop.par,
  MCpar = get("nl.MCpar"),
  log = TRUE
)
```

Arguments

type	One of the character strings "r" or "d".
cur.par	Current state of the chain.
prop.par	Candidate parameter.
MCpar	A list made of a single element: MC MC parameter. Re-centering parameter for the proposal distribution.
log	Logical. Only used when type =="d". Should the result be returned on the log-scale ?

Details

The two components of proposal parameter (`alpha*`, `beta12*`, `beta13*`, `beta23*`) are generated independently, under a beta distribution with mode at the current parameter's value.

Let $\epsilon =$ `MCpar$eps.recentre`. To generate `alpha*`, given the current state `alpha(t)`, let $m(t) = \epsilon/2 + (1-\epsilon) * \alpha(t)$ be the mean of the Beta proposal distribution and $\lambda = 2/\epsilon$ (a scaling constant). Then

$$\alpha^* \sim \text{Beta}(\lambda m(t), (1-\lambda)m(t))$$

The `betaij*`'s are generated similarly.

Value

Either the (log-)density of the proposal parameter `prop.par`, given `cur.par` (if `type == "d"`), or a proposal parameter (a vector), if `type =="r"`.

`proposal.pb` *PB model: proposal distribution*

Description

Density of the proposal distribution `q(cur.par,prop.par)` and random generator for MC MC algorithm in the PB model.

Usage

```
proposal.pb(type = c("r", "d"), cur.par, prop.par, MCpar, log = TRUE)
```

Arguments

`type`	One of the character strings `"r"`, `"d"`
`cur.par`	Current state of the chain
`prop.par`	Candidate parameter
`MCpar`	A list made of a single element: MC MC parameter for the standard deviation of the log-normal proposition, on the log scale. See `pb.MCpar` for the default value
`log`	Logical. Only used when `type =="d"`. Should the result be returned on the log-scale ?

Details

The components `prop.par[i]` of the proposal parameter are generated independently, from the lognormal distribution:

`prop.par = rlnorm(length(cur.par), meanlog=log(cur.par),sdlog=rep(MCpar$sdlog,length(cur.par)))`

Value

Either the (log-)density of the proposal `prop.par`, given `cur.par` (if `type == "d"`), or a proposal parameter (a vector), if `type =="r"`.

Examples

```
## Not run:  proposal.pb(type = "r",
cur.par = rep(1,4), MCpar=get("pb.MCpar"))

## End(Not run)
## Not run:  proposal.pb(type = "d", cur.par = rep(1,4),
prop.par=rep(1.5,4), MCpar=get("pb.MCpar"))

## End(Not run)
```

---

rdirichlet *Dirichlet distribution: random generator*

Description

Dirichlet distribution: random generator

Usage

```
rdirichlet(n = 1, alpha)
```

Arguments

n	Number of draws
alpha	Dirichlet parameter: a vector of positive number

Value

A matrix with n rows and length(alpha) columns

---

rect.integrate *Density integration on the two-dimensional simplex*

Description

The integral is approximated by a rectangular method, using the values stored in matrix density.

Usage

```
rect.integrate(density, npoints, eps)
```

Arguments

density	A npoints*npoints matrix containing the density's values scattered on the discretization grid defined by npoints, equi, eps (see discretize).
npoints	The number of grid nodes on the squared grid containing the desired triangle.
eps	Positive number: minimum distance from any node inside the simplex to the simplex boundary

Details

Integration is made with respect to the Lebesgue measure on the projection of the simplex onto the plane $(x, y) : x > 0, y > 0, x + y < 1$. It is assumed that `density` has been constructed on a grid obtained *via* function `discretize`, with argument `equi` set to `FALSE` and `npoints` and `eps` equal to those passed to `rect.integrate`.

Value

The value of the estimated integral of `density`.

Examples

```
wrapper <- function(x, y, my.fun,...)
  {
    sapply(seq_along(x), FUN = function(i) my.fun(x[i], y[i],...))
  }

grid <- discretize(npoints=40,eps=1e-3,equi=FALSE)

Density <- outer(grid$X,grid$Y,FUN=wrapper,
                                  my.fun=function(x,y){10*((x/2)^2+y^2)})

rect.integrate(Density,npoints=40,eps=1e-3)
```

---

rstable.posit *Positive alpha-stable distribution.*

Description

Random variable generator

Usage

```
rstable.posit(alpha = 0.5)
```

Arguments

`alpha`	The parameter of the alpha-stable random variable

Details

An alpha-stable random variable S with index α is defined by its Laplace transform $E(exp(tS))) = exp(-t^{\alpha})$. The algorithm used here is directly derived from Stephenson (2003).

Value

A realization of the alpha-stable random variable.

References

STEPHENSON, A. (2003). Simulating multivariate extreme value distributions of logistic type. *Extremes 6, 49-59.*

`scores3D` *Logarithmic score and L^2 distance between two densities on the simplex (trivariate case).*

Description

Computes the Kullback-Leibler divergence and the L^2 distance between the "true" density (`true.dens`) and an estimated density (`est.dens`).

Usage

```
scores3D(true.dens, est.dens, npoints, eps)
```

Arguments

`true.dens`	A `npoints*npoints` matrix: The reference density, typically the distribution from which data was simulated. Must be a valid `density` argument to be passed to `dgridplot`, with `equi=FALSE`.
`est.dens`	The estimated density: of the same type as `true.dens`.
`npoints`	Number of grid points used to construct the density matrices (see `discretize`).
`eps`	Minimum distance from a grid point to the simplex boundary (see `discretize`).

Details

The integration is made *via* `rect.integrate`: The discretization grid corresponding to the two matrices must be constructed with `discretize(npoints, eps, equi=FALSE)`.

Value

A list made of

- `check.true`: The result of the rectangular integration of `true.dens`. It should be equal to one. If not, re size the grid.
- `check.true`: Idem, replacing `true.dens` with `est.dens`.
- `L2score`: The estimated L^2 distance.
- `KLscore`: The estimated Kullback-Leibler divergence between the two re-normalized densities, using `check.true` and `check.est` as normalizing constants (this ensures that the divergence is always positive).

Examples

```
dens1=dpairbeta.grid(par=c(0.8,2,5,8),npoints=150,eps=1e-3,
                     equi=FALSE)
dens2=dnestlog.grid(par=c(0.5,0.8,0.4,0.6),npoints=150,eps=1e-3, equi=FALSE)

scores3D(true.dens=dens1,
  est.dens=dens2,
  npoints=150, eps=1e-4)
```

`transf.to.equi` *Linear coordinate transformations*

Description

Switching coordinates system between equilateral and right-angled representation of the two-dimensional simplex.

Usage

```
transf.to.equi(vect)

transf.to.rect(vect)
```

Arguments

`vect`	a bi-variate vector, giving the first two coordinates of the angular point to be transformed.

Details

If `transf.to.rect`, is called, `vect` must belongs to the triangle $[(0,0),(\sqrt(2),0),(\sqrt(2)/2,\sqrt(3/2))]$ and the result lies in $([(0,0),(1,0),(0,1)]$. `transf.to.equi` is the reciprocal.

Value

The vector obtained by linear transformation.

Author(s)

Anne Sabourin

Examples

```
## Not run:  transf.to.equi(c(sqrt(2)/2, sqrt(3/8) ) )
```

`winterdat` *Five-dimensional air quality dataset recorded in Leeds(U.K.), during five winter seasons.*

Description

Contains 590 daily maxima of five air pollutants (respectively PM10, N0, NO2, 03, S02) recorded in Leeds (U.K.) during five winter seasons (1994-1998, November-February included). Contains NA's.

Format

A $590 * 5$ matrix.

Source

http://www.airquality.co.uk

Index