Package 'ExtremalDep'

August 29, 2019

Version 0.0.3-1

Date 2019-08-08

Title Extremal Dependence Models

Author Boris Beranger [aut],
Simone Padoan [cre, aut],
Giulia Marcon [aut],
Steven G. Johnson [ctb] (Author of included cubature fragments)

Maintainer Simone Padoan <simone.padoan@unibocconi.it>

Imports numDeriv, evd, sn, CompRandFld, quadprog, copula, nloptr, gtools, mvtnorm, rlist, fda

Description A set of procedures for modelling parametrically and non-parametrically the dependence structure of multivariate extreme-values is provided. The statistical inference is performed with non-parametric estimators, likelihood-based estimators and Bayesian techniques. Adapts the methodologies derived in Beranger et al. (2019) <arxiv:1904.08251>, Beranger et al. (2017) <doi:10.1111/sjos.12240>, Beranger and Padoan (2015) <arxiv:1508.05561>, Marcon et al. (2017) <doi:10.1002/sta4.145>, Marcon et al. (2017) <doi:10.1016/j.jspi.2016.10.004> and Marcon et al. (2016) <doi:10.1214/16-EJS1162>. It also refers to the works of Bortot (2010) <https://pdfs.semanticscholar.org/b0dc/1cb608d35bf515c76e39aacc14b4de82e281.pdf>, Padoan (2011) <doi:10
ley et al. (2010) <doi:10.1016/j.jmva.2010.04.007>, Husler and Reiss (1989) <doi:10.1016/0167-7152(89)90106-5>, Engelke et al. (2015) <doi:10.1111/rssb.12074>, Coles and Tawn (1991) <doi:10.1111/j.2517-6161.1991.tb01830.x>, Nikoloulopoulos et al. (2011) <doi:10.1007/s10687-008-0072-4>, Opitz (2013) <doi:10.1016/j.jmva.2013.08.008>, Tawn (1990) <doi:10.2307/2336802>, Azzalini (1985) <https://www.jstor.org/stable/pdf/4615982.pdf>, Azzalini and Capitanio (2014) <doi:10.1017/CBO9781139248891>, Azzalini (2003) <doi:10.1111/1467-9469.00322>, Azzalini and Capitanio (1999) <doi:10.1111/1467-9868.00194>, Azzalini and Dalla Valle (1996) <doi:10.1093/biomet/83.4.715>, Einmahl et al. (2013) <doi:10.1007/s10687-012-0156-z>, Naveau et al (2009) <doi:10.1093/biomet/asp001> and Heffernan and Tawn (2004) <doi:10.1111/j.1467-9868.2004.02050.x>.

License GPL (>= 2)

LazyData yes

NeedsCompilation yes

Repository CRAN

Repository/R-Forge/Project extremaldep

Repository/R-Forge/Revision 122

Repository/R-Forge/DateTimeStamp 2019-08-28 15:43:09

Date/Publication 2019-08-29 07:40:02 UTC

Depends R (>= 2.10)

R topics documented:

alik . . . 3
AngDensPlot . . . 5
angular . . . 8
bbeed . . . 10
beed . . . 13
beed.boot . . . 15
beed.confband . . . 17
chi.bsn . . . 19
chi.extst . . . 20
dens . . . 21
dens_extr_mod . . . 26
desn . . . 27
dest . . . 28
dmesn . . . 29
dmest . . . 31
ellipse . . . 32
excess_pr_extr_mod . . . 34
exponent_extr_mod . . . 35
ExtQset . . . 37
fit_pclik_extr_mod . . . 41
madogram . . . 44
MilanPollution . . . 45
pk.extst . . . 46
plot.bbeed . . . 47
pollution . . . 49
posteriorMCMC . . . 50
prior . . . 53
proposal . . . 55
returns . . . 57
r_extr_mod . . . 58
summary.bbeed . . . 60
UniExtQ . . . 61
Wind . . . 65

Index **66**

alik *Approximate likelihood estimation of extremal dependence models.*

Description

Estimates the parameters of extremal dependence models. It also provides standard errors and TIC.

Usage

```
alik(data, model, parastart, c=NULL, trace=0, sig=3)
```

Arguments

data	A $(n \times d)$ matrix of angular components, where the rows represent n independent points in the d-dimensional unit simplex.
model	A string with the name of the parametric model to be estimated. See **Details**.
parastart	A vector containing the starting values of the model's parameters for the maximisation of the log-approximate likelihood. See **Details**.
c	A real value in $[0, 1]$, providing the decision rule to allocate a data point to a subset of the simplex. Only required for the Extremal-t, Extremal Skew-t and Asymmetric Logistic models.
trace	Non-negative integer. See the options of the routine **optim** in **R** for details. `trace=0` is the default.
sig	Non-negative integer. Provides the number of decimal places for the returned object. `sig=3` is the default.

Details

The available parametric extremal dependence models are:

- The **Pairwise Beta**, called with `model="Pairwise"`. The number of parameters is `choose(d,2)+1`;
- The **Husler-Reiss**, called with `model="Husler"`. The number of parameters is `choose(d,2)`;
- The **Tilted Dirichlet**, called with `model="Dirichlet"`. The number of parameters is d;
- The **Extremal-t**, called with `model="Extremalt"`. The number of parameters is `choose(d,2)+1`;
- The **Extremal Skew-t**, called with `model="Skewt"`. The number of parameters is `choose(d,2)+d+1`;
- The **Asymmetric Logistic**, that can be called with `model="Asymmetric"`. The number of dependence parameters is $2^{d-1}(d+2) - (2d+1)$.

See **References** and the references therein.

Standard errors are calculated using the sandwich (Godambe) information matrix.

Value

Returns a list where `par` are the estimated parameters, `LL` is the value of the maximized log-likelihood, `TIC` is the Takeuchi Information Criterion and `SE` are the standard errors.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Beranger, B. and Padoan, S. A. (2015). Extreme dependence models, chapater of the book *Extreme Value Modeling and Risk Analysis: Methods and Applications*, **Chapman Hall/CRC**.

Beranger, B., Padoan, S. A. and Sisson, S. A. (2017). Models for extremal dependence derived from skew-symmetric families. *Scandinavian Journal of Statistics*, **44**(1), 21-45.

Examples

```
################################################
# The following examples provide the fitting
# results of the air quality data recorded in
# the city center of Leeds, UK, analysed in
# Beranger and Padoan (2015).
################################################

## Load datsets
data(pollution)


## Dataset PM10-NO-SO2 (PNS)


alik(PNS,model="Pairwise",c(1,1,1,1),trace=2,sig=2)
alik(PNS,model="Husler",rep(1,3),trace=2,sig=2)
alik(PNS,model="Dirichlet",rep(0.1,3),trace=2,sig=2)
alik(PNS,model="Extremalt",c(-0.5,-0.4,-0.5,1),c=0.01,trace=2,sig=2)
alik(PNS,model="Asymmetric",c(rep(1.1,4),rep(0.1,9)),c=0.01,trace=2,sig=2)


## Dataset NO2-SO2-NO (NSN)


alik(NSN,model="Pairwise",c(1,1,1,1),trace=2,sig=2)
alik(NSN,model="Husler",rep(1,3),trace=2,sig=2)
alik(NSN,model="Dirichlet",rep(0.1,3),trace=2,sig=2)
alik(NSN,model="Extremalt",c(-0.5,-0.4,-0.5,1),c=0.01,trace=2,sig=2)
alik(NSN,model="Asymmetric",c(rep(1.1,4),rep(0.1,9)),c=0.01,trace=2,sig=2)


## Dataset PM10-NO-NO2 (PNN)


alik(PNN,model="Pairwise",c(1,1,1,1),trace=2,sig=2)
alik(PNN,model="Husler",rep(1,3),trace=2,sig=2)
alik(PNN,model="Dirichlet",rep(0.1,3),trace=2,sig=2)
alik(PNN,model="Extremalt",c(-0.5,-0.4,-0.5,1),c=0.01,trace=2,sig=2)
```

```
alik(PNN,model="Asymmetric",c(rep(1.1,4),rep(0.1,9)),c=0.01,trace=2,sig=2)


## Dataset PM10-NO-NO2-SO2 (PNNS)


alik(PNNS,model="Pairwise",rep(1,choose(ncol(PNNS),2)+1),trace=2,sig=2)
alik(PNNS,model="Husler",rep(1,choose(ncol(PNNS),2)),trace=2,sig=2)
alik(PNNS,model="Dirichlet",rep(1,ncol(PNNS)),trace=2,sig=2)
```

AngDensPlot *3-D plot of parametric angular densities.*

Description

Plots (log)-angular densities on the three-dimensional simplex. Contour levels and data points (optional) are represented.

Usage

```
AngDensPlot(model='Pairwise', para=c(2,4,15,1), log=TRUE, data=NULL,
            contour=TRUE, labels=c(expression(w[1]),expression(w[3]),
            expression(w[2])), cex.dat=1, cex.lab=1, cex.cont=1)
```

Arguments

model	A string with the name of the parametric model for the angular density. "Pairiwse" is the default, see alik.
para	A numeric vector with the parameters of the parameteric model. Default is para=c(2,4,15,1), parameters for the Pairwise Beta model.
log	Logical; if log=TRUE (default) then the log-density is plotted.
data	If a (three-dimensional) dataset if provided then the data points are added to the density plot.
contour	Logical; if contour=TRUE (default) then the contour levels are plotted.
labels	Labels for the three corners of the simplex. Default is labels=c(expression(w[1]),expression(w[3]) See **details**.
cex.dat	Magnification of data points. Only if data != NULL.
cex.lab	Magnification of the labels.
cex.cont	Magnification of the contour labels.

Details

Contour levels are given for the deciles. If data != NULL then the deciles are calculated using the density values of each data point. If data = NULL then the deciles are calcuated using all the points of the grid.

labels are given in an anticlockwise order: bottom right, top middle and bottom left.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Beranger, B. and Padoan, S. A. (2015). Extreme dependence models, chapater of the book *Extreme Value Modeling and Risk Analysis: Methods and Applications*, **Chapman Hall/CRC**.

Beranger, B., Padoan, S. A. and Sisson, S. A. (2017). Models for extremal dependence derived from skew-symmetric families. *Scandinavian Journal of Statistics*, **44**(1), 21-45.

Examples

```
################################################
# The following examples provide the plots of
# Figure 1.2 of the paper Beranger and Padoan (2015)
################################################

# The code has been frozen to speed up the package check.
# Please remove the hash symbol to test the code.

# Asymmetric Logistic


AngDensPlot('Asymmetric', c(rep(1,3),5.75, rep(0,6), 0.5,0.5,0.5), FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot('Asymmetric', c(rep(1,3),1.01, rep(0,6), 0.9,0.9,0.9), FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot('Asymmetric', c(rep(1,3),1.25, rep(0,6), 0.5,0.5,0.5), FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot('Asymmetric', c(rep(1,3),1.4, rep(0,6),0.7,0.15,0.15), FALSE, cex.lab=1.5, cex.cont=1.3)


# Tilted Dirichlet


AngDensPlot(model='Dirichlet', para=c(2,2,2), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Dirichlet', para=c(0.5,0.5,0.5), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Dirichlet', para=c(2,2.5,30), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Dirichlet', para=c(0.1,0.25,0.95), log=FALSE, cex.lab=1.5, cex.cont=1.3)


# Pairwise Beta


AngDensPlot(model='Pairwise', para=c(2,2,2,4), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Pairwise', para=c(1,1,1,0.5), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Pairwise', para=c(2,4,15,1), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Pairwise', para=c(10,10,10,1), log=FALSE, cex.lab=1.5, cex.cont=1.3)


# Husler-Reiss
```

```
AngDensPlot(model='Husler', para=c(0.3,0.3,0.3), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Husler', para=c(1.4,1.4,1.4), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Husler', para=c(1.7,0.7,1.1), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Husler', para=c(0.52,0.71,0.52), log=FALSE, cex.lab=1.5, cex.cont=1.3)


# Extremal-t


AngDensPlot(model='Extremalt', para=c(0.95,0.95,0.95,2), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Extremalt', para=c(-0.3,-0.3,-0.3,5), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Extremalt', para=c(0.52,0.71,0.52,3), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Extremalt', para=c(0.52,0.71,0.52,2), log=FALSE, cex.lab=1.5, cex.cont=1.3)


################################################
# The following examples provide
# the plots of Figure 1.3 of the paper
# Beranger and Padoan (2015)
################################################

## Load datasets
data(pollution)

Nsim <- 50e+4
Nbin <- 30e+4
MCpar <- 0.35
Hpar.pb <-  list(mean.alpha=0, mean.beta=3,sd.alpha=3, sd.beta=3)
Hpar.hr <- list(mean.lambda=0, sd.lambda=3)
Hpar.di <- list(mean.alpha=0, sd.alpha=3)
Hpar.et <- list(mean.rho=0, mean.mu=3,sd.rho=3, sd.mu=3)

## PNS Data


est.pb.PNS <- posteriorMCMC(Nsim, Nbin, Hpar.pb, MCpar, PNS, seed=14342, model='Pairwise')
est.hr.PNS <- posteriorMCMC(Nsim, Nbin, Hpar.hr, MCpar, PNS, seed=14342, model='Husler')
est.di.PNS <- posteriorMCMC(Nsim, Nbin, Hpar.di, MCpar, PNS, seed=14342, model='Dirichlet')

lab1 <- c("PM10","NO","SO2")
AngDensPlot("Pairwise", est.pb.PNS$emp.mean, data=PNS, labels=lab1, cex.dat=0.8)
AngDensPlot("Husler", est.hr.PNS$emp.mean, data=PNS, labels=lab1, cex.dat=0.8)
AngDensPlot("Dirichlet", est.di.PNS$emp.mean, data=PNS, labels=lab1, cex.dat=0.8)


## NSN data


est.pb.NSN <- posteriorMCMC(Nsim, Nbin, Hpar.pb, MCpar, NSN, seed=14342, model='Pairwise')
est.hr.NSN <- posteriorMCMC(Nsim, Nbin, Hpar.hr, MCpar, NSN, seed=14342, model='Husler')
est.di.NSN <- posteriorMCMC(Nsim, Nbin, Hpar.di, MCpar, NSN, seed=14342, model='Dirichlet')

lab2 <- c("NO2","NO","SO2")
```

```
AngDensPlot("Pairwise", est.pb.NSN$emp.mean, data=NSN, labels=lab2, cex.dat=0.8)
AngDensPlot("Husler", est.hr.NSN$emp.mean, data=NSN, labels=lab2, cex.dat=0.8)
AngDensPlot("Dirichlet", est.di.NSN$emp.mean, data=NSN, labels=lab2, cex.dat=0.8)


## PNN data


est.pb.PNN <- posteriorMCMC(Nsim, Nbin, Hpar.pb, MCpar, PNN, seed=14342, model='Pairwise')
est.hr.PNN <- posteriorMCMC(Nsim, Nbin, Hpar.hr, MCpar, PNN, seed=14342, model='Husler')
est.di.PNN <- posteriorMCMC(Nsim, Nbin, Hpar.di, MCpar, PNN, seed=14342, model='Dirichlet')

lab3 <- c("PM10","NO","NO2")
AngDensPlot("Pairwise", est.pb.PNN$emp.mean, data=PNN, labels=lab3, cex.dat=0.8)
AngDensPlot("Husler", est.hr.PNN$emp.mean, data=PNN, labels=lab3, cex.dat=0.8)
AngDensPlot("Dirichlet", est.di.PNN$emp.mean, data=PNN, labels=lab3, cex.dat=0.8)


################################################
# The following examples provide the plots of
# the supplementary material for
# Beranger, Padoan and Sisson (2016)
################################################


AngDensPlot(model='Skewt', para=c(0.6,0.8,0.7,0,0,0,3), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Skewt', para=c(0.6,0.8,0.7,-3,-3,7,3), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Skewt', para=c(0.6,0.8,0.7,7,-10,3,3), log=FALSE, cex.lab=1.5, cex.cont=1.3)

AngDensPlot(model='Skewt', para=c(0.7,0.7,0.7,0,0,0,3), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Skewt', para=c(0.7,0.7,0.7,-3,-3,7,3), log=FALSE, cex.lab=1.5, cex.cont=1.3)
AngDensPlot(model='Skewt', para=c(0.7,0.7,0.7,7,-10,3,3), log=FALSE, cex.lab=1.5, cex.cont=1.3)
```

`angular` *Estimation of the angular density, angular measure and random generation from the angular distribution.*

Description

Empirical estimation to the Pickands dependence function, the angular density, the angular measure and random generation of samples from the estimated angular density.

Usage

```
angular(data, model, n, dep, asy, alpha, beta, df, seed, k, nsim, plot=TRUE, nw=100)
```

Arguments

data	The dataset in vector form
model	The specified model; a character string. Must be either "log", "alog", "hr", "neglog", "aneglog", "bilog", "negbilog", "ct", "amix" or "Extremalt" for the logistic, asymmetric logistic, Husler-Reiss, negative logistic, asymmetric negative logistic, bilogistic, negative bilogistic, Coles-Tawn, asymetric mixed and Extremal-t models respectively.
n	The number of random generations from the model. Required if data=NULL.
dep	The dependence parameter for the model.
asy	A vector of length two, containing the two asymmetry parameters for the asymmetric logistic (model='alog') and asymmetric negative logistic models (model='aneglog').
alpha,beta	Alpha and beta parameters for the bilogistic, negative logistic, Coles-Tawn and asymmetric mixed models.
df	The degree of freedom for the extremal-t model.
seed	The seed for the data generation. Required if data=NULL.
k	The polynomial order.
nsim	The number of generations from the estimated angular density.
plot	If TRUE, the fitted angular density, histogram of the generated observations from the angular density and the true angular density (if model is specified) are displayed.
nw	The number of points at which the estimated functions are evaluated

Details

See Marcon et al. (2017).

Value

Returns a list which contains model, n, dep, data, Aest the estimated pickands dependence function, hest the estimated angular density, Hest the estimated angular measure, p0 and p1 the point masses at the edge of the simplex, wsim the simulated sample from the angular density and Atrue and htrue the true Pickand dependence function and angular density (if model is specified).

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Giulia Marcon, <giuliamarcongm@gmail.com>

References

Marcon, G., Naveau, P. and Padoan, S. A. (2017). A semi-parametric stochastic generator for bivariate extreme events, *Stat 6*(1), 184–201.

Examples

```
################################################
# The following examples provide the left panels
# of Figure 1, 2 & 3 of Marcon et al. (2017).
################################################

## Figure 1 - symmetric logistic


# Strong dependence
a <- angular(model='log', n=50, dep=0.3, seed=4321, k=20, nsim=10000)
# Mild dependence
b <- angular(model='log', n=50, dep=0.6, seed=212, k=10, nsim=10000)
# Weak dependence
c <- angular(model='log', n=50, dep=0.9, seed=4334, k=6, nsim=10000)


## Figure 2 - Asymmetric logistic


# Strong dependence
d <- angular(model='alog', n=25, dep=0.3, asy=c(.3,.8), seed=43121465, k=20, nsim=10000)
# Mild dependence
e <- angular(model='alog', n=25, dep=0.6, asy=c(.3,.8), seed=1890, k=10, nsim=10000)
# Weak dependence
f <- angular(model='alog', n=25, dep=0.9, asy=c(.3,.8), seed=2043, k=5, nsim=10000)
```

---

bbeed *Bayesian Estimation of Extremal Dependence*

Description

Nonparametric Bayesian estimation of the extremal dependence function (angular measure and Pickands dependence function) for two-dimensional data on the basis of Bernstein polynomials representations.

Usage

```
bbeed(data, pm0, param0, k0, hyperparam, nsim, nk = 70,
      prior.k = c('pois','nbinom'), prior.pm = c('unif','beta', 'null'),
      lik = TRUE)
```

Arguments

data	$(n \times d)$ matrix of component-wise maxima (n is the number of replications).
pm0	list; the initial state of the Markov chain of point masses p_0 and p_1. Randomly generated if missing.

param0	real vector in (0,1) of length k0+1; the initial state of the Markov chain of η coefficients (see **Details**). Randomly generated if missing.
k0	postive integer indicating the initial state of the Markov chain of the Bernstein polynomial's degree. Randomly generated if missing.
hyperparam	list containing the hyperparameters (see **Details**).
nsim	postive integer indicating the number of the iterations of the chain.
nk	positive integer, maximum value of k that can be reached (nk = 70 by default).
prior.k	string, indicating the prior distribution on the degree of the Bernstein polynomial, k. If missing, prior.k = "pois" by default.
prior.pm	string, indicating the prior distribution on the point masses p_0 and p_1. If missing, prior.pm = "unif" by default.
lik	logical; if FALSE, an approximation of the likelihood, by means of the angular measure in Bernstein form is used (TRUE by default).

Details

The hyperparameters and starting values may not be specified and in this case a warning message will be printed and default values are set. In particular if hyperparam is missing we have

```
hyperparam <-NULL hyperparam$a.unif <-0 hyperparam$b.unif <-.5 hyperparam$a.beta <-c(.8,.8)
hyperparam$b.beta <-c(5,5) mu.pois <-hyperparam$mu.pois <-4 mu.nbinom <-hyperparam$mu.nbinom
<-4 var.nbinom <-8 pnb <-hyperparam$pnb <-mu.nbinom/var.nbinom rnb <-hyperparam$rnb
<-mu.nbinom^2 / (var.nbinom-mu.nbinom)
```

The routine returns an estimate of the Pickands dependence function using the Bernstein polynomials approximation proposed in Marcon et al. (2016). The method is based on a preliminary empirical estimate of the Pickands dependence function. If you do not provide such an estimate, this is computed by the routine. In this case, you can select one of the empirical methods available. est = "ht" refers to the Hall-Tajvidi estimator (Hall and Tajvidi 2000). With est = "cfg" the method proposed by Caperaa et al. (1997) is considered. Note that in the multivariate case the adjusted version of Gudendorf and Segers (2011) is used. Finally, with est = "md" the estimate is based on the madogram defined in Marcon et al. (2016).

For each row of the $(m \times d)$ design matrix x is a point in the unit d-dimensional simplex,

$S_d := \left\{(w_1, \ldots, w_d) \in [0,1]^d : \sum_{i=1}^d w_i = 1\right\}.$

With this "regularization" method, the final estimate satisfies the neccessary conditions in order to be a Pickands dependence function.

$A(\boldsymbol{w}) = \sum_{\boldsymbol{\alpha} \in \Gamma_k} \beta_{\boldsymbol{\alpha}} b_{\boldsymbol{\alpha}}(\boldsymbol{w}; k)$.

The estimates are obtained by solving an optimization quadratic problem subject to the constraints. The latter are represented by the following conditions: $A(e_i) = 1; \max(w_i) \leq A(w) \leq 1; \forall i = 1, \ldots, d$; (convexity).

The order of polynomial k controls the smoothness of the estimate. The higher k is, the smoother the final estimate is. Higher values are better with strong dependence (e. g. k=23), whereas small values (e.g. k=6 or k=10) are enough with mild or weak dependence.

An empirical transformation of the marginals is performed when margin="emp". A max-likelihood fitting of the GEV distributions is implemented when margin="est". Otherwise it refers to marginal parametric GEV theorethical distributions (margin = "exp","frechet","gumbel").

If bounds = TRUE, a modification can be implemented to satisfy $\max(w, 1-w) \leq A_n(w) \leq 1 \forall 0 \leq w \leq 1$:

$A_n(w) = \min(1, \max A_n(w), w, 1-w)$.

Value

return(list(pm=spm, eta=seta, k=sk, accepted=accepted/nsim, prior = list(hyperparam=hyperparam, k=prior.k, pm=prior.pm)))

beta	vector of polynomial coefficients
A	numeric vector of the estimated Pickands dependence function A
Anonconvex	preliminary non-convex function
extind	extremal index

Note

The number of coefficients depends on both the order of polynomial k and the dimension d. The number of parameters is $\binom{d-1}{k+d-1}$.

The size of the vector beta must be compatible with the polynomial order k chosen.

With the estimated polynomial coefficients, the extremal coefficient, i.e. $d * A(1/d, \ldots, 1/d)$ is computed.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Giulia Marcon, <giuliamarcongm@gmail.com>

References

Marcon G., Padoan, S.A. and Antoniano-Villalobos I. (2016) Bayesian Inference for the Extremal Dependence. *Electronic Journal of Statistics*, **10**(2), 3310-3337.

See Also

summary.bbeed

Examples

```
# This reproduces some of the results showed in Fig. 1 (Marcon, 2016).
set.seed(1890)
data <- evd::rbvevd(n=100, dep=.6, asy=c(0.8,0.3), model="alog", mar1=c(1,1,1))

nsim = 500000
burn = 400000

mu.nbinom = 3.2
```

```
var.nbinom = 4.48
hyperparam <- list(a.unif=0, b.unif=.5, mu.nbinom=mu.nbinom, var.nbinom=var.nbinom)
k0 = 5
pm0 = list(p0=0.06573614, p1=0.3752118)
eta0 = ExtremalDep:::rcoef(k0, pm0)

mcmc <- bbeed(data, pm0, eta0, k0, hyperparam, nsim,
              prior.k = "nbinom", prior.pm = "unif")

w <- seq(0.001, .999, length=100)
summary.mcmc <- summary.bbeed(w, mcmc, burn, nsim, plot=TRUE)
```

---

beed *Bernstein Polynomials Based Estimation of Extremal Dependence*

Description

Estimates the Pickands dependence function corresponding to multivariate data on the basis of a Bernstein polynomials approximation.

Usage

```
beed(data, x, d = 3, est = c("ht","cfg","md","pick"),
     margin = c("emp","est","exp","frechet","gumbel"),
     k = 13, y = NULL, beta = NULL, plot = FALSE)
```

Arguments

data	$(n \times d)$ matrix of component-wise maxima. d is the number of variables and n is the number of replications.
x	$(m \times d)$ design matrix where the dependence function is evaluated (see **Details**).
d	positive integer greater than or equal to two indicating the number of variables (d = 3 by default).
est	string, indicating the estimation method (est="md" by default, see **Details**).
margin	string, denoting the type marginal distributions (margin="emp" by default, see **Details**).
k	postive integer, indicating the order of Bernstein polynomials (k=13 by default).
y	numeric vector (of size m) with an initial estimate of the Pickands function. If NULL, the initial estimate is computed using the estimation method denoted in est.
beta	vector of polynomial coefficients (see **Details**).
plot	logical; if TRUE and d<=3, the estimated Pickands dependence function is plotted (FALSE by default).

Details

The routine returns an estimate of the Pickands dependence function using the Bernstein polynomials approximation proposed in Marcon et al. (2017). The method is based on a preliminary empirical estimate of the Pickands dependence function. If you do not provide such an estimate, this is computed by the routine. In this case, you can select one of the empirical methods available. `est = 'ht'` refers to the Hall-Tajvidi estimator (Hall and Tajvidi 2000). With `est = 'cfg'` the method proposed by Caperaa et al. (1997) is considered. Note that in the multivariate case the adjusted version of Gudendorf and Segers (2011) is used. Finally, with `est = 'md'` the estimate is based on the madogram defined in Marcon et al. (2017).

Each row of the $(m \times d)$ design matrix `x` is a point in the unit `d`-dimensional simplex,

$S_d := \left\{(w_1, \ldots, w_d) \in [0,1]^d : \sum_{i=1}^{d} w_i = 1\right\}.$

With this "regularization"" method, the final estimate satisfies the neccessary conditions in order to be a Pickands dependence function.

$A(\boldsymbol{w}) = \sum_{\boldsymbol{\alpha} \in \Gamma_k} \beta_{\boldsymbol{\alpha}} b_{\boldsymbol{\alpha}}(\boldsymbol{w}; k)$.

The estimates are obtained by solving an optimization quadratic problem subject to the constraints. The latter are represented by the following conditions: $A(e_i) = 1; \max(w_i) \leq A(w) \leq 1; \forall i = 1, \ldots, d$; (convexity).

The order of polynomial `k` controls the smoothness of the estimate. The higher `k` is, the smoother the final estimate is. Higher values are better with strong dependence (e. g. `k = 23`), whereas small values (e.g. `k = 6` or `k = 10`) are enough with mild or weak dependence.

An empirical transformation of the marginals is performed when `margin="emp"`. A max-likelihood fitting of the GEV distributions is implemented when `margin="est"`. Otherwise it refers to marginal parametric GEV theorethical distributions (`margin = "exp","frechet","gumbel"`).

Value

`beta`	vector of polynomial coefficients
`A`	numeric vector of the estimated Pickands dependence function A
`Anonconvex`	preliminary non-convex function
`extind`	extremal index

Note

The number of coefficients depends on both the order of polynomial `k` and the dimension `d`. The number of parameters is explained in Marcon et al. (2017).

The size of the vector `beta` must be compatible with the polynomial order `k` chosen.

With the estimated polynomial coefficients, the extremal coefficient, i.e. $d * A(1/d, \ldots, 1/d)$ is computed.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Giulia Marcon, <giuliamarcongm@gmail.com>

References

Marcon, G., Padoan, S.A., Naveau, P., Muliere, P. and Segers, J. (2017) Multivariate Nonparametric Estimation of the Pickands Dependence Function using Bernstein Polynomials. *Journal of Statistical Planning and Inference*, **183**, 1-17.

See Also

beed.confband.

Examples

```
x <- ExtremalDep:::simplex(2)
data <- evd::rbvevd(50, dep = 0.4, model = "log", mar1 = c(1,1,1))

Amd <- beed(data, x, 2, "md", "emp", 20, plot=TRUE)
Acfg <- beed(data, x, 2, "cfg", "emp", 20)
Aht <- beed(data, x, 2, "ht", "emp", 20)

lines(x[,1], Aht$A, lty = 1, col = 3)
lines(x[,1], Acfg$A, lty = 1, col = 2)

#################################
# Trivariate case
#################################



x <- ExtremalDep:::simplex(3)

data <- evd::rmvevd(50, dep = 0.8, model = "log", d = 3, mar = c(1,1,1))

op <- par(mfrow=c(1,3))
Amd <- beed(data, x, 3, "md", "emp", 18, plot=TRUE)
Acfg <- beed(data, x, 3, "cfg", "emp", 18, plot=TRUE)
Aht <- beed(data, x, 3, "ht", "emp", 18, plot=TRUE)

par(op)
```

---

beed.boot *Bootstrap Resampling and Bernstein Estimation of Extremal Dependence*

Description

Computes nboot estimates of the Pickands dependence function for multivariate data (using the Bernstein polynomials approximation method) on the basis of the bootstrap resampling of the data.

Usage

```
beed.boot(data, x, d = 3, est = c("ht","md","cfg","pick"),
   margin=c("emp", "est", "exp", "frechet", "gumbel"), k = 13,
   nboot = 500, y = NULL, print = FALSE)
```

Arguments

data	$n \times d$ matrix of component-wise maxima.
x	$m \times d$ design matrix where the dependence function is evaluated, see **Details**.
d	postive integer (greater than or equal to two) indicating the number of variables (d=3 by default).
est	string denoting the preliminary estimation method (see **Details**).
margin	string denoting the type marginal distributions (see **Details**).
k	postive integer denoting the order of the Bernstein polynomial (k=13 by default).
nboot	postive integer indicating the number of bootstrap replicates (nboot=500 by default).
y	numeric vector (of size m) with an initial estimate of the Pickands function. If NULL, The initial estimation is performed by using the estimation method chosen in est.
print	logical; FALSE by default. If TRUE the number of the iteration is printed.

Details

Standard bootstrap is performed, in particular estimates of the Pickands dependence function are provided for each data resampling.

Most of the settings are the same as in the function beed.

An empirical transformation of the marginals is performed when margin="emp". A max-likelihood fitting of the GEV distributions is implemented when margin="est". Otherwise it refers to marginal parametric GEV theorethical distributions (margin = "exp","frechet","gumbel").

Value

A	numeric vector of the estimated Pickands dependence function.
bootA	matrix with nboot columns that reports the estimates of the Pickands function for each data resampling.
beta	matrix of estimated polynomial coefficients. Each column corresponds to a data resampling.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Giulia Marcon, <giuliamarcongm@gmail.com>

References

Marcon, G., Padoan, S.A., Naveau, P., Muliere, P. and Segers, J. (2017) Multivariate Nonparametric Estimation of the Pickands Dependence Function using Bernstein Polynomials. *Journal of Statistical Planning and Inference*, 183, 1-17.

See Also

beed, beed.confband.

Examples

```
x <- ExtremalDep:::simplex(2)
data <- rbvevd(50, dep = 0.4, model = "log", mar1 = c(1,1,1))

boot <- beed.boot(data, x, 2, "md", "emp", 20, 500)
```

---

beed.confband *Nonparametric Bootstrap Confidence Intervals*

Description

Computes nonparametric bootstrap $(1-\alpha)\%$ confidence bands for the Pickands dependence function.

Usage

```
beed.confband(data, x, d = 3, est = c("ht","md","cfg","pick"),
   margin=c("emp", "est", "exp", "frechet", "gumbel"), k = 13,
   nboot = 500, y = NULL, conf = 0.95, plot = FALSE, print = FALSE)
```

Arguments

data	$(n \times d)$ matrix of component-wise maxima.
x	$(m \times d)$ design matrix (see **Details**).
d	postive integer (greater than or equal to two) indicating the number of variables (d=3 by default).
est	string denoting the estimation method (see **Details**).
margin	string denoting the type marginal distributions (see **Details**).
k	postive integer denoting the order of the Bernstein polynomial (k=13 by default).
nboot	postive integer indicating the number of bootstrap replicates.
y	numeric vector (of size m) with an initial estimate of the Pickands function. If NULL, the initial estimation is performed by using the estimation method chosen in est.

conf	real value in $(0, 1)$ denoting the confidence level of the interval. The value conf=0.95 is the default.
plot	logical; FALSE by default. If TRUE, the confidence bands are plotted.
print	logical; FALSE by default. If TRUE, the number of the iteration is printed.

Details

Two methods for computing bootstrap $(1-\alpha)\%$ point-wise and simultaneous confidence bands for the Pickands dependence function are used.

The first method derives the confidence bands computing the point-wise $\alpha/2$ and $1-\alpha/2$ quantiles of the bootstrap sample distribution of the Pickands dependence Bernstein based estimator.

The second method derives the confidence bands, first computing the point-wise $\alpha/2$ and $1-\alpha/2$ quantiles of the bootstrap sample distribution of polynomial coefficient estimators, and then the Pickands dependence is computed using the Bernstein polynomial representation. See Marcon et al. (2017) for details.

Most of the settings are the same as in the function beed.

Value

A	numeric vector of the Pickands dependence function estimated.
bootA	matrix with nboot columns that reports the estimates of the Pickands function for each data resampling.
A.up.beta/A.low.beta	vectors of upper and lower bands of the Pickands dependence function obtained using the bootstrap sampling distribution of the polynomial coefficients estimator.
A.up.pointwise/A.low.pointwise	vectors of upper and lower bands of the Pickands dependence function obtained using the bootstrap sampling distribution of the Pickands dependence function estimator.
up.beta/low.beta	vectors of upper and lower bounds of the bootstrap sampling distribution of the polynomial coefficients estimator.

Note

This routine relies on the bootstrap routine (see beed.boot).

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Giulia Marcon, <giuliamarcongm@gmail.com>

References

Marcon, G., Padoan, S.A., Naveau, P., Muliere, P. and Segers, J. (2017) Multivariate Nonparametric Estimation of the Pickands Dependence Function using Bernstein Polynomials. *Journal of Statistical Planning and Inference*, 183, 1-17.

See Also

beed, beed.boot.

Examples

```
x <- ExtremalDep:::simplex(2)
data <- rbvevd(50, dep = 0.4, model = "log", mar1 = c(1,1,1))

# Note you should consider 500 bootstrap replications.
# In order to obtain fastest results we used 50!
cb <- beed.confband(data, x, 2, "md", "emp", 20, 50, plot=TRUE)
```

---

chi.bsn *Tail dependence coefficient for the skew-normal distirbution*

Description

Evaluates the upper and lower tail dependence coefficients for the bivariate skew-normal.

Usage

```
chi.bsn(u, corr=0, shape=rep(0,2), tail="upper")
```

Arguments

u	a real value in $[0, 1]$.
corr	the correlation parameter, between -1 and 1.
shape	a numeric vector of real values of length 2 with the skewness parameters.
tail	the string "upper" or "lower".

Details

Approximation, the tail dependence is obtained in the limiting case where uu goes to eqn11.

Value

Returns a value that is strictly greater than 0 and less than 1 for the upper coefficient, and between -1 and 1 for the lower coefficient.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Bortot, P. Tail dependence in bivariate skew-normal and skew-t distributions. *Unpublished*.

Examples

```
### Upper tail dependence

chi.bsn(u=0.9,corr=0.5, shape=c(1,-2), tail="upper")

### Lower tail dependence

chi.bsn(u=0.9, corr=0.5, shape=c(1,-2), tail="lower")
```

`chi.extst` *Tail dependence coefficient for the Extremal Skew-t model*

Description

Evaluates the upper and lower tail dependence coefficients for the bivariate Extremal Skew-t model.

Usage

```
chi.extst(corr=0, shape=rep(0,2), df=1, tail="upper")
```

Arguments

`corr`	the correlation parameter, between -1 and 1.
`shape`	a numeric skewness vector of length 2.
`df`	a single positive value representing the degree of freedom.
`tail`	the string `"upper"` or `"lower"`.

Value

Returns a value that is strictly greater than 0 and less than 1.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Padoan, S. A. (2011). Multivariate extreme models based on underlying skew-t and skew-normal distributions. *Journal of Multivariate Analysis*, **102**(5), 977-991.

Examples

```
### Upper tail dependence

chi.extst(corr=0.5, shape=c(1,-2), df=2, tail="upper")

### Lower tail dependence

chi.extst(corr=0.5, shape=c(1,-2), df=2, tail="lower")
```

---

dens *Angular density and likelihood function for some extremal dependence models*

Description

Evaluates the angular density or calculates the likelihood function of the Pairwise Beta, Husler-Reiss, Dirichlet, Extremal-t, Extremal Skew-t and Asymmetric Logistic models at one or more locations on the unit simplex.

Usage

```
dens(x=rbind(c(0.1,0.3,0.6),c(0.1,0.2,0.7)), model="Pairwise", par=c(2,2,2,4), c,
        log=FALSE, vectorial=TRUE)
```

Arguments

x	A $(n \times d)$ matrix of angular components, where the rows represent n independent points in the d-dimensional unit simplex. See **Details**. The default is `rbind(c(0.1,0.3,0.6),c(0.1,0.2,0.7))`, two points in the 3-dimensional simplex.
model	A string with the name of the parametric model to be estimated. Models are Pairwise Beta (`"Pairwise"`), Husler-Reiss (`"Husler"`), Dirichlet (`"Dirichlet"`), Extremal-t (`"Extremalt"`), Extremal Skew-t (`"Skewt"`) and Asymmetric Logistic (`"Asymmetric"`).
par	A vector containing the parameters of the model. See **Details**.
c	A real value in $[0, 1]$, providing the decision rule to allocate a data point to a subset of the simplex. Only required for the Extremal-t, Extremal Skew-t and Asymmetric Logistic models.
log	Logical; if `TRUE` the log-density is returned. `FALSE` is the default.
vectorial	Logical; if `TRUE` when $n > 1$ the different densities are returned as a vector of length n. If `FALSE` the likelihood function is returned. `TRUE` is the default.

Details

The Extremal-t and Asymmetric Logistic models are available up to 3 dimensions; mass on all the subsets of the simplex is included.

For the Pairwise Beta model, the parameter vector is decomposed as:

b A vector of size `choose(d,2)`. Controls the dependence between pairs. The default is `b=c(2,2,2)`.

alpha A positive real that controls the general dependence between all the variables. The default is 4.

For the Husler-Reiss model, the parameter vector is of size `choose(d,2)`.

For the Dirichlet model, the parameter vector is decomposed a vector of size d which controls the dependence between pairs.

For the Extremal-t model, the parameter vector is decomposed as:

rho A vector of size `choose(d,2)` representing the corrleation parameters.

mu A positive integer, $\mu \geq 1$, representing the degree of freedom.

For the Extremal Skew-t model, the parameter vector is decomposed as:

rho A vector of size `choose(d,2)` representing the corrleation parameters.

alpha A vector of size `d` representing the shape parameters.

mu A positive integer, $\mu \geq 1$, representing the degree of freedom.

For the Asymmetric Logistic model, the parameter vector is decomposed as:

alpha A vector of size 1 or 4 depending on whether $d = 2$ or 3.

beta A vector of size 2 or 9 depending on whether $d = 2$ or 3.

If `log=TRUE` and `vectorial=FALSE` then the log-likelihood function is calculated.

Value

Returns a n-dimensional vector if `vectorial=TRUE` or a single value if `vectorial=FALSE`.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com>

References

Cooley, D.,Davis, R. A., and Naveau, P. (2010). The pairwise beta distribution: a flexible parametric multivariate model for extremes. *Journal of Multivariate Analysis*, **101**, 2103–2117.

Husler, J. and Reiss, R.-D. (1989), Maxima of normal random vectors: between independence and complete dependence, *Statistics and Probability Letters*, **7**, 283–286.

Engelke, S., Malinowski, A., Kabluchko, Z., and Schlather, M. (2015), Estimation of Husler-Reiss distributions and Brown-Resnick processes, *Journal of the Royal Statistical Society, Series B (Methodological)*, **77**, 239–265.

Coles, S. G., and Tawn, J. A. (1991), Modelling Extreme Multivariate Events, *Journal of the Royal Statistical Society, Series B (Methodological)*, **53**, 377–392.

Nikoloulopoulos, A. K., Joe, H., and Li, H. (2009) Extreme value properties of t copulas. *Extremes*, **12**, 129–148.

Opitz, T. (2013) Extremal t processes: Elliptical domain of attraction and a spectral representation. *Jounal of Multivariate Analysis*, **122**, 409–413.

Beranger, B. and Padoan, S. A. (2015). Extreme dependence models, chapater of the book *Extreme Value Modeling and Risk Analysis: Methods and Applications*, **Chapman Hall/CRC**.

Beranger, B., Padoan, S. A. and Sisson, S. A. (2017). Models for extremal dependence derived from skew-symmetric families. *Scandinavian Journal of Statistics*, **44**(1), 21-45.

Tawn, J. A. (1990), Modelling Multivariate Extreme Value Distributions, *Biometrika*, **77**, 245–253.

Examples

```
### Pairwise Beta :


# Examples on the 3-dimensional simplex
# Returns the bivariate angular density at two locations

dens(x=rbind(c(0.1,0.3,0.6),c(0.1,0.2,0.7)), model="Pairwise", par=c(2,2,2,4),
log=FALSE, vectorial=TRUE)

# returns the likelihood function at two locations

dens(x=rbind(c(0.1,0.3,0.6),c(0.1,0.2,0.7)), model="Pairwise", par=c(2,2,2,4),
log=FALSE, vectorial=FALSE)

# returns the log-likelihood function

dens(x=rbind(c(0.1,0.3,0.6),c(0.1,0.2,0.7)), model="Pairwise", par=c(2,2,2,4),
log=TRUE, vectorial=FALSE)

# Examples on the 4-dimensional simplex
# returns the bivariate angular density at two locations

dens(x=rbind(c(0.1,0.3,0.3,0.3),c(0.1,0.2,0.3,0.4)), model="Pairwise", par=c(2,2,2,1,0.5,3,4),
log=FALSE, vectorial=TRUE)

# returns the likelihood function at two locations
dens(x=rbind(c(0.1,0.3,0.3,0.3),c(0.1,0.2,0.3,0.4)), model="Pairwise", par=c(2,2,2,1,0.5,3,4),
log=FALSE, vectorial=FALSE)

# returns the log-likelihood function

dens(x=rbind(c(0.1,0.3,0.3,0.3),c(0.1,0.2,0.3,0.4)), model="Pairwise", par=c(2,2,2,1,0.5,3,4),
log=TRUE, vectorial=FALSE)
```

```
### Husler-Reiss


# Example on the 2-dimensional simplex
# returns the log-likelihood at two locations

dens(x=rbind(c(0.1,0.9),c(0.3,0.7)), model="Husler", par=1.7,
log=TRUE, vectorial=FALSE)

# Example on the 3-dimensional simplex
# returns the likelihood function at two locations

dens(x=rbind(c(0.1,0.3,0.6),c(0.1,0.2,0.7)), model="Husler", par=c(1.7,0.7,1.1),
log=FALSE, vectorial=FALSE)

# Example on the 4-dimensional simplex
# returns the bivariate angular density at two locations

dens(x=rbind(c(0.1,0.1,0.4,0.4),c(0.1,0.2,0.3,0.4)), model="Husler", par=rep(1,6),
log=FALSE, vectorial=TRUE)


### Dirichlet


# Example on the 2-dimensional simplex
# returns the log-likelihood at two points

dens(x=rbind(c(0.1,0.9),c(0.3,0.7)), model="Dirichlet", par=c(1.7,0.7),
log=TRUE, vectorial=FALSE)


# Example on the 3-dimensional simplex
# returns the likelihood function at three locations

dens(x=rbind(c(0.1,0.3,0.6),c(0.1,0.2,0.7)), model="Dirichlet", par=c(1.7,0.7,1.1),
log=FALSE, vectorial=FALSE)

# Example on the 4-dimensional simplex
# returns the bivariate angular density at two locations

dens(x=rbind(c(0.1,0.1,0.4,0.4),c(0.1,0.2,0.3,0.4)), model="Dirichlet", par=c(1.7,0.7,1.1,0.1),
log=FALSE, vectorial=TRUE)


### Extremal-t


# Example on the 2-dimensional simplex
# Returns the log-likelihood

dens(x=rbind(c(0.4,0.6),c(0.3,0.7)), model="Extremalt", par=c(0.7,2), c=0.1,
log=TRUE, vectorial=FALSE)
```

```
# Density in the corner

dens(x=c(0.08,0.92), model="Extremalt", par=c(0.7,2), c=0.1,
log=FALSE, vectorial=FALSE)


# Example on the 3-dimensional simplex
# Returns the log-likelihood

dens(x=rbind(c(0.1,0.3,0.6),c(0.1,0.2,0.7)), model="Extremalt", par=c(rep(0.1,3),2), c=0.03,
log=FALSE, vectorial=FALSE)

# Returns the evalutaion of the angular density at three locations:
# The first one is set to be on the edge linking the second and third components
# The second one is set to be on the interior of the simplex
# The third one is set to be on the corner near the third component


dens(x=rbind(c(0.001,0.3,0.699),c(0.1,0.2,0.7),c(0.001,0.001,0.998)),
model="Extremalt", par=c(rep(0.1,3),2), c=0.01, log=FALSE, vectorial=TRUE)


### Extremal Skew-t


# Example on the 2-dimensional simplex
# Returns the log-likelihood

dens(x=rbind(c(0.4,0.6),c(0.3,0.7)), model="Skewt", par=c(0.7,0,0,2), c=0.1,
log=TRUE, vectorial=FALSE)

dens(x=rbind(c(0.4,0.6),c(0.3,0.7)), model="Skewt", par=c(0.7,2,-1,2), c=0.1,
log=TRUE, vectorial=FALSE)

# Density in the corner

dens(x=c(0.08,0.92), model="Skewt", par=c(0.7,0,0,2), c=0.1,
log=FALSE, vectorial=FALSE)

dens(x=c(0.08,0.92), model="Skewt", par=c(0.7,-1,2,2), c=0.1,
log=FALSE, vectorial=FALSE)

# Example on the 3-dimensional simplex
# Returns the log-likelihood

dens(x=rbind(c(0.1,0.3,0.6),c(0.1,0.2,0.7)), model="Skewt", par=c(rep(0.1,3),rep(0,3),2), c=0.03,
log=FALSE, vectorial=FALSE)

# Returns the evalutaion of the angular density at three locations:
# The first one is set to be on the edge linking the second and third components
# The second one is set to be on the interior of the simplex
# The third one is set to be on the corner near the third component
```

```
dens(x=rbind(c(0.001,0.3,0.699),c(0.1,0.2,0.7),c(0.001,0.001,0.998)),
model="Skewt", par=c(rep(0.1,3),rep(0,3),2), c=0.01, log=FALSE, vectorial=TRUE)



### Asymmetric Logistic

# Example on the 3-dimensional simplex
# Returns the angular density at three points:
# The first one is set to be on the edge linking the second and third components
# The second one is set to be on the interior of the simplex
# The third one is set to be on the corner near the third component

dens(x=rbind(c(0.001,0.3,0.699),c(0.1,0.2,0.7),c(0.001,0.001,0.998)), c=0.05,
model="Asymmetric", par=c(1.2,1.8,4,2,rep(0.3,9)), log=FALSE, vectorial=TRUE)
```

`dens_extr_mod` *Density of extremal dependence models*

Description

Evaluates the bivariate density of the Extremal-t and Extremal Skew-t models.

Usage

```
dens_extr_mod(model, z, param, log=TRUE)
```

Arguments

`model`	A string with the name of the model: `"Extremalt"` or `"Skewt"`.
`z`	A vector of length 2, containing strictly positive reals.
`param`	A vector containing the parameters of the model. See **Details**.
`log`	Logical; if `TRUE` the log-density is returned. `FALSE` is the default.

Details

If `model="Extremalt"` then the parameter vector is made of one dependence parameter and a degree of freedom. If `model="Skewt"` then the parameter vector is made of one dependence parameter, two shape (or skewness) parameters and a degree of freedom.

Value

Returns a single value corresponding to the density or the log-density.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Beranger, B., Padoan, S. A. and Sisson, S. A. (2017). Models for extremal dependence derived from skew-symmetric families. *Scandinavian Journal of Statistics*, **44**(1), 21-45.

Examples

```
### Extremal-t

dens_extr_mod(model="Extremalt", z=c(0.1,0.3), param=c(0.6,1),log=FALSE)

### Extremal Skew-t

dens_extr_mod(model="Skewt", z=c(0.1,0.3), param=c(0.6,0,0,1),log=FALSE)
dens_extr_mod(model="Skewt", z=c(0.1,0.3), param=c(0.6,-3,5,1),log=FALSE)
```

---

desn *Univariate extended skew-normal distribution*

Description

Density function, distribution function for the univariate extended skew-normal (ESN) distribution.

Usage

```
desn(x, location=0, scale=1, shape=0, extended=0)
pesn(x, location=0, scale=1, shape=0, extended=0)
```

Arguments

x	quantile.
location	location parameter. 0 is the default.
scale	scale parameter; must be positive. 1 is the default.
shape	skewness parameter. 0 is the default.
extended	extension parameter. 0 is the default.

Value

density (desn), probability (pesn) from the extended skew-normal distribution with given location, scale, shape and extended parameters or from the skew-normal if extended=0. If shape=0 and extended=0 then the normal distribution is recovered.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Azzalini, A. (1985). A class of distributions which includes the normal ones. *Scand. J. Statist.* **12**, 171-178.

Azzalini, A. with the collaboration of Capitanio, A. (2014). *The Skew-Normal and Related Families*. Cambridge University Press, IMS Monographs series.

Examples

```
dens1 <- desn(x=1, shape=3, extended=2)
dens2 <- desn(x=1, shape=3)
dens3 <- desn(x=1)
dens4 <- dnorm(x=1)
prob1 <- pesn(x=1, shape=3, extended=2)
prob2 <- pesn(x=1, shape=3)
prob3 <- pesn(x=1)
prob4 <- pnorm(q=1)
```

---

dest *Univariate extended skew-t distribution*

Description

Density function, distribution function for the univariate extended skew-t (EST) distribution.

Usage

```
dest(x, location=0, scale=1, shape=0, extended=0, df=Inf)
pest(x, location=0, scale=1, shape=0, extended=0, df=Inf)
```

Arguments

x	quantile.
location	location parameter. 0 is the default.
scale	scale parameter; must be positive. 1 is the default.
shape	skewness parameter. 0 is the default.
extended	extension parameter. 0 is the default.
df	a single positive value representing the degrees of freedom; it can be non-integer. Default value is nu=Inf which corresponds to the skew-normal distribution.

Value

density (dest), probability (pest) from the extended skew-t distribution with given location, scale, shape, extended and df parameters or from the skew-t if extended=0. If shape=0 and extended=0 then the t distribution is recovered.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Azzalini, A. and Capitanio, A. (2003). Distributions generated by perturbation of symmetry with emphasis on a multivariate skew-*t* distribution. *J.Roy. Statist. Soc. B* **65**, 367–389.

Azzalini, A. with the collaboration of Capitanio, A. (2014). *The Skew-normal and Related Families*. Cambridge University Press, IMS Monographs series.

Examples

```
dens1 <- dest(x=1, shape=3, extended=2, df=1)
dens2 <- dest(x=1, shape=3, df=1)
dens3 <- dest(x=1, df=1)
dens4 <- dt(x=1, df=1)
prob1 <- pest(x=1, shape=3, extended=2, df=1)
prob2 <- pest(x=1, shape=3, df=1)
prob3 <- pest(x=1, df=1)
prob4 <- pt(q=1, df=1)
```

---

dmesn *Bivariate and trivariate extended skew-normal distribution*

Description

Density function, distribution function for the bivariate and trivariate extended skew-normal (ESN) distribution.

Usage

```
dmesn(x=c(0,0), location=rep(0, length(x)), scale=diag(length(x)),
      shape=rep(0,length(x)), extended=0)
pmesn(x=c(0,0), location=rep(0, length(x)), scale=diag(length(x)),
      shape=rep(0,length(x)), extended=0)
```

Arguments

x	quantile vector of length d=2 or d=3.
location	a numeric location vector of length d. 0 is the default.
scale	a symmetric positive-definite scale matrix of dimension (d,d). diag(d) is the default.
shape	a numeric skewness vector of length d. 0 is the default.
extended	a single value extension parameter. 0 is the default.

Value

density (dmesn), probability (pmesn) from the bivariate or trivariate extended skew-normal distribution with given location, scale, shape and extended parameters or from the skew-normal distribution if extended=0. If shape=0 and extended=0 then the normal distribution is recovered.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Azzalini, A. and Capitanio, A. (1999). Statistical applications of the multivariate skew normal distribution. *J.Roy.Statist.Soc. B* **61**, 579–602.

Azzalini, A. with the collaboration of Capitanio, A. (2014). *The Skew-Normal and Related Families*. Cambridge University Press, IMS Monographs series.

Azzalini, A. and Dalla Valle, A. (1996). The multivariate skew-normal distribution. *Biometrika* **83**, 715–726.

Examples

```
sigma1 <- matrix(c(2,1.5,1.5,3),ncol=2)
sigma2 <- matrix(c(2,1.5,1.8,1.5,3,2.2,1.8,2.2,3.5),ncol=3)
shape1 <- c(1,2)
shape2 <- c(1,2,1.5)

dens1 <- dmesn(x=c(1,1), scale=sigma1, shape=shape1, extended=2)
dens2 <- dmesn(x=c(1,1), scale=sigma1)
dens3 <- dmesn(x=c(1,1,1), scale=sigma2, shape=shape2, extended=2)
dens4 <- dmesn(x=c(1,1,1), scale=sigma2)

prob1 <- pmesn(x=c(1,1), scale=sigma1, shape=shape1, extended=2)
prob2 <- pmesn(x=c(1,1), scale=sigma1)


prob3 <- pmesn(x=c(1,1,1), scale=sigma2, shape=shape2, extended=2)
prob4 <- pmesn(x=c(1,1,1), scale=sigma2)
```

dmest *Bivariate and trivariate extended skew-t distribution*

Description

Density function, distribution function for the bivariate and trivariate extended skew-t (EST) distribution.

Usage

```
dmest(x=c(0,0), location=rep(0, length(x)), scale=diag(length(x)),
      shape=rep(0,length(x)), extended=0, df=Inf)
pmest(x=c(0,0), location=rep(0, length(x)), scale=diag(length(x)),
      shape=rep(0,length(x)), extended=0, df=Inf)
```

Arguments

x	quantile vector of length d=2 or d=3.
location	a numeric location vector of length d. 0 is the default.
scale	a symmetric positive-definite scale matrix of dimension (d,d). diag(d) is the default.
shape	a numeric skewness vector of length d. 0 is the default.
extended	a single value extension parameter. 0 is the default.
df	a single positive value representing the degree of freedom; it can be non-integer. Default value is nu=Inf which corresponds to the skew-normal distribution.

Value

density (dmest), probability (pmest) from the bivariate or trivariate extended skew-t distribution with given location, scale, shape, extended and df parameters or from the skew-t distribution if extended=0. If shape=0 and extended=0 then the t distribution is recovered.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Azzalini, A. and Capitanio, A. (2003). Distributions generated by perturbation of symmetry with emphasis on a multivariate skew *t* distribution. *J.Roy. Statist. Soc. B* **65**, 367–389.

Azzalini, A. with the collaboration of Capitanio, A. (2014). *The Skew-Normal and Related Families*. Cambridge University Press, IMS Monograph series.

Examples

```
sigma1 <- matrix(c(2,1.5,1.5,3),ncol=2)
sigma2 <- matrix(c(2,1.5,1.8,1.5,3,2.2,1.8,2.2,3.5),ncol=3)
shape1 <- c(1,2)
shape2 <- c(1,2,1.5)

dens1 <- dmest(x=c(1,1), scale=sigma1, shape=shape1, extended=2, df=1)
dens2 <- dmest(x=c(1,1), scale=sigma1, df=1)
dens3 <- dmest(x=c(1,1,1), scale=sigma2, shape=shape2, extended=2, df=1)
dens4 <- dmest(x=c(1,1,1), scale=sigma2, df=1)

prob1 <- pmest(x=c(1,1), scale=sigma1, shape=shape1, extended=2, df=1)
prob2 <- pmest(x=c(1,1), scale=sigma1, df=1)


prob3 <- pmest(x=c(1,1,1), scale=sigma2, shape=shape2, extended=2, df=1)
prob4 <- pmest(x=c(1,1,1), scale=sigma2, df=1)
```

`ellipse` *Level sets for bivariate normal, student-t and skew-normal distributions probability densities.*

Description

Level sets of the bivariate normal, student-t and skew-normal distributions probability densities for a given probability.

Usage

```
ellipse(center=c(0,0), alpha=c(0,0), sigma=diag(2), df=1,
prob=0.01, npoints=250, pos=FALSE)
```

Arguments

`center`	A vector of length 2 corresponding to the location of the distribution.
`alpha`	A vector of length 2 corresponding to the skewness of the skew-normal distribution.
`sigma`	A 2 by 2 variance-covariance matrix.
`df`	An integer corresponding to the degree of freedom of the student-t distribution.
`prob`	The probability level. See `details`
`npoints`	The maximum number of points at which it is evaluated.
`pos`	If `pos=TRUE` only the region on the positive quadrant is kept.

Details

The Level sets are defined as

$R(f_\alpha) = \{x : f(x) \geq f_\alpha\}$

where f_α is the largest constant such that

$P(X \in R(f_\alpha)) \geq 1 - \alpha$. Here we consider $f(x)$ to be the bivariate normal, student-t or skew-normal density.

Value

Returns a bivariate vector of 250 rows if `pos=FALSE`, and half otherwise.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

Examples

```
library(mvtnorm)

# Data simulation (Bivariate-t on positive quadrant)
rho <- 0.5
sigma <- matrix(c(1,rho,rho,1), ncol=2)
df <- 2

set.seed(101)
n <- 1500
data <- rmvt(5*n, sigma=sigma, df=df)
data <- data[data[,1]>0 & data[,2]>0, ]
data <- data[1:n, ]

P <- c(1/750, 1/1500, 1/3000)

ell1 <- ellipse(prob=1-P[1], sigma=sigma, df=df, pos=TRUE)
ell2 <- ellipse(prob=1-P[2], sigma=sigma, df=df, pos=TRUE)
ell3 <- ellipse(prob=1-P[3], sigma=sigma, df=df, pos=TRUE)

plot(data, xlim=c(0,max(data[,1],ell1[,1],ell2[,1],ell3[,1])),
     ylim=c(0,max(data[,2],ell1[,2],ell2[,2],ell3[,2])), pch=19)
points(ell1, type="l", lwd=2, lty=1)
points(ell2, type="l", lwd=2, lty=1)
points(ell3, type="l", lwd=2, lty=1)
```

`excess_pr_extr_mod` *Exceedance Probability for extremal dependence models*

Description

Exceedance Probability for bivariate or trivariate Husler-Reiss, Extremal-t and Extremal Skew-t models.

Usage

```
excess_pr_extr_mod(model, z, param)
```

Arguments

`model`	A string with the name of the model: `"hr"`, `"Extremalt"` or `"Skewt"`.
`z`	A vector of length 2 or 3, containing strictly positive reals.
`param`	A vector containing the parameters of the model. See **Details**.

Details

If `model="hr"` then the parameter vector is made of `choose(d,2)` positive parameters, `d=2,3`. If `model="Extremalt"` then the parameter vector is made of `choose(d,2)` dependence parameters and a degree of freedom, `d=2,3`. If `model="Skewt"` then the parameter vector is made of `choose(d,2)` dependence parameters, `d` shape (or skewness) parameters and a degree of freedom, `d=2,3`.

Value

Returns a probability.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Beranger, B. and Padoan, S. A. (2015). Extreme dependence models, chapater of the book *Extreme Value Modeling and Risk Analysis: Methods and Applications*, **Chapman Hall/CRC**.

Beranger, B., Padoan, S. A. and Sisson, S. A. (2017). Models for extremal dependence derived from skew-symmetric families. *Scandinavian Journal of Statistics*, **44**(1), 21-45.

Examples

```
### Husler-Reiss


excess_pr_extr_mod(model="hr", z=c(1,3), param=0.5)
excess_pr_extr_mod(model="hr", z=c(1,3,5), param=c(5,4,2))
excess_pr_extr_mod(model="hr", z=c(0.001,0.001,0.001), param=c(5,4,2))


### Extremal-t


excess_pr_extr_mod(model="Extremalt", z=c(0.1,0.3), param=c(0.5,2))
excess_pr_extr_mod(model="Extremalt", z=c(1,3,5), param=c(0.5,0.4,0.8,2))
excess_pr_extr_mod(model="Extremalt", z=c(0.001,0.001,0.001), param=c(0.5,0.4,0.8,2))


### Extremal Skew-t


excess_pr_extr_mod(model="Skewt", z=c(0.1,0.3), param=c(0.5,0,0,2))
excess_pr_extr_mod(model="Skewt", z=c(0.1,0.3), param=c(0.5,-10,-4,2))
excess_pr_extr_mod(model="Skewt", z=c(1,3,5), param=c(0.5,0.4,0.8,0,0,0,2))
excess_pr_extr_mod(model="Skewt", z=c(1,3,5), param=c(0.5,0.4,0.8,1,5,10,2))
excess_pr_extr_mod(model="Skewt", z=c(0.001,0.001,0.001), param=c(0.5,0.4,0.8,1,5,10,2))
```

---

exponent_extr_mod *Exponent function of extremal dependence models*

Description

Evaluates the bivariate or trivariate exponent function of the Husler-Reiss, Extremal-t and Extremal Skew-t models.

Usage

```
exponent_extr_mod(model, z, param, dist)
```

Arguments

model	A string with the name of the model: "hr", "Extremalt" or "Skewt".
z	A vector of length 2 or 3, containing strictly positive reals.
param	A vector containing the parameters of the model. See **Details**.
dist	Logical; if TRUE the distribution of the model is returned.

Details

If `model="hr"` then the parameter vector is made of `choose(d,2)` positive parameters, `d=2,3`. If `model="Extremalt"` then the parameter vector is made of `choose(d,2)` dependence parameters and a degree of freedom, `d=2,3`. If `model="Skewt"` then the parameter vector is made of `choose(d,2)` dependence parameters, `d` shape (or skewness) parameters and a degree of freedom, `d=2,3`.

Value

Returns a single value corresponding to the exponent function or value of the distribution.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Beranger, B. and Padoan, S. A. (2015). Extreme dependence models, chapater of the book *Extreme Value Modeling and Risk Analysis: Methods and Applications*, **Chapman Hall/CRC**.

Beranger, B., Padoan, S. A. and Sisson, S. A. (2017). Models for extremal dependence derived from skew-symmetric families. *Scandinavian Journal of Statistics*, **44**(1), 21-45.

Examples

```
### Husler-Reiss


exponent_extr_mod(model="hr", z=c(2,3), param=1.2, dist=FALSE)
exponent_extr_mod(model="hr", z=c(2,3,1), param=c(1.2,1,1.4), dist=TRUE)


### Extremal-t


exponent_extr_mod(model="Extremalt", z=c(0.1,2), param=c(0.5,2), dist=FALSE)
exponent_extr_mod(model="Extremalt", z=c(0.1,2,3), param=c(0.5,0.4,0.9,2), dist=TRUE)


### Extremal Skew-t


exponent_extr_mod(model="Skewt", z=c(0.1,2), param=c(0.5,0,0,2), dist=FALSE)
exponent_extr_mod(model="Skewt", z=c(0.1,2,3), param=c(0.5,0.4,0.9,0,0,0,2), dist=TRUE)
exponent_extr_mod(model="Skewt", z=c(0.1,2,3), param=c(0.5,0.4,0.9,1,10,-5,2), dist=TRUE)
```

`ExtQset` *Bivariate Extreme Quantile Sets*

Description

Computes extreme-quantiles regions of a bivariate random variable correspoding to some exceedance probabilities.

Usage

```
ExtQset(data, P=NULL, method="bayesian", U=NULL,
        cov1=as.matrix(rep(1,nrow(data))), cov2=as.matrix(rep(1,nrow(data))),
        QatCov1=NULL, QatCov2=NULL, mar=TRUE, par10=c(1,2,1), par20=c(1,2,1),
        sig10=1, sig20=1, param0=NULL, k0=NULL, pm0=NULL, prior.k="nbinom",
        prior.pm="unif", hyperparam = list(mu.nbinom = 3.2, var.nbinom = 4.48),
        nsim=NULL, lo=NULL, up=NULL, d=5)
```

Arguments

`data`	A matrix of $n \times 2$ observations.
`P`	The vector of probabilities associated to the quantiles.
`method`	The estimation method can be "bayesian", "EdHK" or "frequentist".
`U`	The bivariate threshold value under which the observations are marginally censored.
`cov1`	A $n \times c1$ matrix of covariates for the location parameter of the first margin.
`QatCov1`	q $n \times c1$ matrix with the value of the first margin covariates at which the quantiles should be computed.
`cov2`	A $n \times c2$ matrix of covariates for the location parameter of the second margin.
`QatCov2`	q $n \times c2$ matrix with the value of the second margin covariates at which the quantiles should be computed.
`mar`	Only required when `method="bayesian"`. If `mar=TRUE` then a first estimation of the margins is done.
`par10, par20`	Only required when `method="bayesian"`. The vector of initial value for the parameters.
`sig10, sig20`	Only required when `method="bayesian"`. Initial value for the standard deviations of the multivariate normal proposal distribution for both margins.
`param0`	Only required when `method="bayesian"`. The vector of initial value for the Bernstein polynomial coefficients. It should be a list with elements `$eta` and `$beta`.
`k0`	Only required when `method="bayesian"`. The initial value of the polynomial order.
`pm0`	Only required when `method="bayesian"`. The list of initial values of the probability masses at the boundaries of the simplex. It should be a list with two elements `$p0` and `$p1`, see bbeed.

prior.k	Only required when method="bayesian". The prior on the polynomial order, see bbeed.
prior.pm	Only required when method="bayesian". The prior on the probability masses at the endpoints of the simplex, see bbeed.
hyperparam	Only required when method="bayesian". A list of the hyper-parameters, see bbeed.
nsim	Only required when method="bayesian". Number of iterations in the Metropolis-Hastings algorithm.
lo	Only required when method="EdHK","frequentist". Lower value of k in Hill estimator for shape parameter.
up	Only required when method="EdHK","frequentist". Upper value of k in Hill estimator for shape parameter.
d	postive integer, indicating the order of Bernstein polynomials

Details

For some dataset given by data, the extreme-quantiles corresponding to some exceedance probability(ies) given in P are computed. The observations below the threshold U are considered censored.

- If method="bayesian", the methodologies given by bbeed and UniExtQ are combined. The algorithm is a Trans-dimensional MCMC scheme as described in Algorithm 1 of Beranger et al. (2019). If mar=TRUE then the function UniExtQ is preliminarily applied to the margins to select some starting values updating par10,par20,sig10 and sig20. After running for nsim iteration the algorithm is paused and some diagnostics plots (on the margins and polynomial degree) are printed. The user then needs to enter the value of the burn-in period. See bbeed for details about the prior and hyperparameters for the dependence structure.
- If method="EdHK", then the methodology developped by Einmahl et al. (2013) is applied. This is a non-parametric approach where the marginal parameters are estimated first.The Hill estimator is used to estimate the marginal tail indexes.
- If method="frequentist", then similar to the "EdHK" estimator, the marginal parameters are estimated first (in the same way) and then the beed function is used to estimate the dependence structure.

Value

If method=="bayesian", a list with elements:

- Qset_P1,...: length(P) lists of 100 2 by 3 matrices corresponding to the extreme quantile regions associated with probability P, using 100 equidistant points in the unit simplex, for 3 different levels: 0.05-quantile, mean and 0.95-quantile;
- Qset_P1_post,...: length(P) lists of 100 2 by $npost$ matrices corresponding to the posterior samples of size npost of the extreme quantile regions associated with probability P;
- ghat: a 3 by 100 matrix giving the 0.05-quantile, mean and 0.95-quantile of the inverse angular density q^* obtained from the posterior sample;
- Shat: a 3 by 100 matrix giving the 0.05-quantile, mean and 0.95-quantile of the basic set $\mathcal{B}$ obtained from the posterior sample;

- `nuShat`: the 0.05-quantile, mean and 0.95-quantile of the estimate of the basic set size $\nu(\mathcal{B})$;
- `burn`: the burn-in period, specified by the user after observing the diagnostic plots;

If `method=="EDhK"`, a list with elements:

- `xn_hat1,...`: `length(P)` vectors of lentgh 101giving the x-axis values of the estimated quantiles associated with probability `P`.
- `yn_hat1,...`: `length(P)` vectors of lentgh 101giving the y-axis values of the estimated quantiles associated with probability `P`.

If `method=="frequentist"`, a list with elements:

- `hhat`: a vector of length 100 giving the estimated angular density at 100 equally spaced points on the unit simplex;
- `ghat`: a vector of length 100 giving the corresponding estimated $1/q^*$ function;
- `Shat`: a 100 by 2 matrix of the corresponding estimated basic set $\mathcal{B}$;
- `nuShat`: a real giving the estimate of the basic set size $\nu(\mathcal{B})$
- `Qhat`: a $100 \times 2 \times$ `length(P)` list corresponding to `length(P)` 100×2 matrices representing the estimated extreme quantile regions associated with probability `P`;
- `gamhat`: a bivariate vector of the estimated marginal tail indices;
- `uhat`: a bivariate vector of the estimated location parameters.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Andrea Krajina, <akrajina@gmail.com>

References

Beranger, B., Padoan S. A. and Sisson, S. A. (2019). Estimation and uncertainty quantification for extreme quantile regions. *arXiv e-prints* arXiv:1904:08251.

Einmahl, J. H. J., de Haan, L. and Krajina, A. (2013). Estimating extreme bivariate quantile regions. *Extremes*, **16**, 121-145.

See Also

UniExtQ, bbeed, beed

Examples

```
library(mvtnorm)

distribution <- "Cauchy"

par10 <- par20 <- c(1,2,1) # Initial marginal parameter values
sig10 <- sig20 <- 1 # Initial scale values in MVN proposal
prior.k <- "nbinom" # Prior distribution for polynomial degree
```

```
k0 <- 5 # Degree of the polynomial
prior.pm <- "unif" # Prior distribution for the point masses
pm0 <- list(p0=0, p1=0)
# Vector of hyperparameters for prior distribution:
hyperparam <- list(mu.nbinom = 3.2, var.nbinom = 4.48, a.unif = 0, b.unif = 0.1)

###
### Data simulation
###

n <- 1500 # Sample size
P <- c(1/750, 1/1500, 1/3000) # Vector of probabilities for extreme quantiles
prob <- c(0.9, 0.9) # To use to evaluate thresholds

# Dependence structure;
rho <- 0
sigma <- matrix(c(1,rho,rho,1), ncol=2)
df <- 1

# Compute quantiles for the Cauchy:
ell1 <- ellipse(prob=1-P[1], pos=TRUE)
ell2 <- ellipse(prob=1-P[2], pos=TRUE)
ell3 <- ellipse(prob=1-P[3], pos=TRUE)

realx1 <- ell1[,1]; realy1 <- ell1[,2]
realx2 <- ell2[,1]; realy2 <- ell2[,2]
realx3 <- ell3[,1]; realy3 <- ell3[,2]

# Data simulation (Cauchy)
set.seed(999)
data <- rmvt(5*n, sigma=sigma, df=df)
data <- data[data[,1]>0 & data[,2]>0, ]
data <- data[1:n, ]

# Threshold
U <- c(quantile(data[,1], probs = prob[1], type=3), quantile(data[,2], probs = prob[2], type=3))

###
### Estimation
###

Q <- ExtQset(data=data, P=P, U=U, par10=par10, par20=par20, sig10=sig10, sig20=sig20, pm0=pm0,
           k0=k0, prior.k=prior.k, prior.pm=prior.pm, hyperparam=hyperparam, nsim=50000)

Q.EDhK <- ExtQset(data=data, P=P, method="EDhK", lo=50, up=300)

w <- seq(0.00001, .99999, length=100) # define grid
gfun <- ((w^2+(1-w)^2)^(-3/2))^(1/3) # Compute the true g function
xT <- gfun*w # x-axis of Basic set
yT <- gfun*(1-w) # y-axis of Basic set

###
### Graphical representation
```

```
###

op <- par(mfrow=c(2,3), mar=c(3, 3, 0.5, 0.2), mgp=c(2,.8,0))

# Plot 1: Density of Exponent measure

ylim.pl1 <- c(0,1.7)
plot(w, gfun, type="l", xlab="w", ylab=expression(1/q[symbol("\052")](w)), ylim=ylim.pl1)
polygon(c(w, rev(w)), c(Q$ghat[3,], rev(Q$ghat[1,])), col="gray")
lines(w, Q$ghat[2,],col="gray0", lwd=2, lty=3)
lines(w, gfun, lwd=2)

# Plot 2: Basic-set S

xlim.pl2 <-c(0,1.5); ylim.pl2 <- c(0,1.5)
plot(xT,yT, pch=19, col=1, type="l", xlim=xlim.pl2, ylim=ylim.pl2,
 xlab=expression(x[1]), ylab=expression(x[2]))
polygon(c(Q$Shat[,1,3], rev(Q$Shat[,1,1])), c(Q$Shat[,2,3], rev(Q$Shat[,2,1])), col="gray")
points(Q$Shat[,,2], type="l", col="gray0", lwd=2, lty=3)
lines(xT,yT,lwd=2)

# Plot 3: Data + quantile regions

xlim.pl3 <- c(0, 3500); ylim.pl3 <- c(0, 3500)
plot(data, xlim=xlim.pl3, ylim=ylim.pl3, pch=19, xlab=expression(x[1]), ylab=expression(x[2]))
points(realx1,realy1, type="l", lwd=2, lty=1)
points(realx2,realy2, type="l", lwd=2, lty=1)
points(realx3,realy3, type="l", lwd=2, lty=1)
lines(Q$Qset_P1_CovNum_1[,,2], lty=3, col="gray0", lwd=2)
lines(Q$Qset_P2_CovNum_1[,,2], lty=3, col="gray0", lwd=2)
lines(Q$Qset_P3_CovNum_1[,,2], lty=3, col="gray0", lwd=2)

# Plot 4,5,6: Quantile region with probability 1/750, 1/1500, 1/3000

xlim.pl46 <- c(0,7400); ylim.pl46 <- c(0,7400)
for(j in 1:3){
  tmp.name <- paste("Qset_P",j,"_CovNum_1",sep="")
  tmp.quant <- Q[[tmp.name]]

  plot(data, xlim=xlim.pl46, ylim=ylim.pl46, type="n", pch=19,
   xlab=expression(x[1]), ylab=expression(x[2]))
  polygon(c(tmp.quant[,1,3], rev(tmp.quant[,1,1])),
   c(tmp.quant[,2,3], rev(tmp.quant[,2,1])), col="gray")
 points(get(paste("realx",j,sep="")), get(paste("realy",j,sep="")), type="l", lty=1, lwd=2)
  lines(tmp.quant[,,2], lty=3, col="gray0", lwd=2)
 lines(Q.EDhK[[paste("xn_hat",j,sep="")]], Q.EDhK[[paste("yn_hat",j,sep="")]], lty=2, lwd=2)
}
par(op)
```

`fit_pclik_extr_mod` *Fit extremal dependence models using pairwise composite likelihood*

Description

Estimates the parameters of the Husler-Reiss, Extremal-t and Extremal Skew-t models using pairwise composite likelihood, for up to 4 dimensional datasets.

Usage

```
fit_pclik_extr_mod(model, data, parastart, trace)
```

Arguments

`model`	A string with the name of the model: `"hr"`, `"Extremalt"` or `"Skewt"`.
`data`	A `data.frame` or `matrix` obejct with up to 4 columns.
`parastart`	A vector containing the initial parameter values. See **Details**.
`trace`	A non-negative integer. If positive, tracing information on the progress of the optimization is produced. See the options of the routine **optim** in **R** for details.

Details

Data must be marginally on unit Frechet scale.

If `model="hr"` then the vector of initial values is made of `choose(d,2)` positive parameters, `d=2,3`. If `model="Extremalt"` then the vector of initial values is made of `choose(d,2)` dependence parameters and a degree of freedom, `d=2,3`. If `model="Skewt"` then the vector of initial values is made of `choose(d,2)` dependence parameters, `d` shape (or skewness) parameters and a degree of freedom, `d=2,3`.

In the case of bivariate data the regular likelihood estimation is performed.

Value

Returns the vector of estimated parameters and the value of the pairwise composite log-likelihood.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Beranger, B. and Padoan, S. A. (2015). Extreme dependence models, chapter of the book *Extreme Value Modeling and Risk Analysis: Methods and Applications*, **Chapman Hall/CRC**.

Beranger, B., Padoan, S. A. and Sisson, S. A. (2017). Models for extremal dependence derived from skew-symmetric families. *Scandinavian Journal of Statistics*, **44**(1), 21-45.

Examples

```
## Reproduce the real data analysis from
## Beranger et al. (2016), Section 5.

data(Wind)

## Vector of starting values
p0 <- c(rep(0.5,3),1)

### CLOU CLAY SALL


ext1 <- fit_pclik_extr_mod("Extremalt", CLOU.CLAY.SALL, p0, 2)
est.ext1 <- round(ext1$par,2)
p01 <- c(ext1$par[1:3],rep(0,3),ext1$par[4])
skewt1 <- fit_pclik_extr_mod("Skewt", CLOU.CLAY.SALL, p01, 2)
est.skewt1 <- round(skewt1$par,2)


### CLOU CLAY PAUL


ext2 <- fit_pclik_extr_mod("Extremalt", CLOU.CLAY.PAUL, p0, 2)
est.ext2 <- round(ext2$par,2)
p02 <- c(ext2$par[1:3],rep(0,3),ext2$par[4])
skewt2 <- fit_pclik_extr_mod("Skewt", CLOU.CLAY.PAUL, p02, 2)
est.skewt2 <- round(skewt2$par,2)


### CLAY SALL PAUL


ext3 <- fit_pclik_extr_mod("Extremalt", CLAY.SALL.PAUL, p0, 2)
est.ext3 <- round(ext3$par,2)
p03 <- c(ext3$par[1:3],rep(0,3),ext3$par[4])
skewt3 <- fit_pclik_extr_mod("Skewt", CLAY.SALL.PAUL, p03, 2)
est.skewt3 <- round(skewt3$par,2)


### CLAY SALL PAUL


ext4 <- fit_pclik_extr_mod("Extremalt", CLOU.SALL.PAUL, p0, 2)
est.ext4 <- round(ext4$par,2)
p04 <- c(ext4$par[1:3],rep(0,3),ext4$par[4])
skewt4 <- fit_pclik_extr_mod("Skewt", CLOU.SALL.PAUL, p04, 2)
est.skewt4 <- round(skewt4$par,2)
```

`madogram` *Madogram-based estimation of the Pickands Dependence Function*

Description

Computes a non-parametric estimate Pickands dependence function, $A(w)$ for multivariate data, based on the madogram estimator.

Usage

```
madogram(w, data, margin = c("emp","est","exp","frechet","gumbel"))
```

Arguments

`w`	$(m \times d)$ design matrix (see **Details**).
`data`	$(n \times d)$ matrix of data or data frame with `d` columns. `d` is the numer of variables and `n` is the number of replications.
`margin`	string, denoting the type marginal distributions (`margin="emp"` by default, see **Details**).

Details

The estimation procedure is based on the madogram as proposed in Marcon et al. (2017). The madogram is defined by

$\nu(\boldsymbol{w}) = \mathrm{E}\left(\bigvee_{i=1,\ldots,d} \left\{ F_i^{1/w_i}(X_i) \right\} - \frac{1}{d} \sum_{i=1,\ldots,d} F_i^{1/w_i}(X_i) \,. \right),$

where $0 < w_i < 1$ and $w_d = 1 - (w_1 + \ldots + w_{d-1})$.

Each row of the design matrix `w` is a point in the unit `d`-dimensional simplex.

If X is a `d`-dimensional max-stable distributed random vector, with exponent measure function $V(\boldsymbol{x})$ and Pickands dependence function $A(\boldsymbol{w})$, then

$\nu(\boldsymbol{w}) = V(1/w_1, \ldots, 1/w_d)/(1+V(1/w_1, \ldots, 1/w_d)) - c(\boldsymbol{w})$, where $c(\boldsymbol{w}) = d^{-1} \sum_{i=1}^{d} w_i/(1+w_i)$.

From this, it follows that

$V(1/w_1, \ldots, 1/w_d) = \frac{\nu(\boldsymbol{w})+c(\boldsymbol{w})}{1-\nu(\boldsymbol{w})-c(\boldsymbol{w})}$,

and

$A(\boldsymbol{w}) = \frac{\nu(\boldsymbol{w})+c(\boldsymbol{w})}{1-\nu(\boldsymbol{w})-c(\boldsymbol{w})}$.

An empirical transformation of the marginals is performed when `margin="emp"`. A max-likelihood fitting of the GEV distributions is implemented when `margin="est"`. Otherwise it refers to marginal parametric GEV theorethical distributions (`margin="exp"`,`"frechet"`,`"gumbel"`).

Value

A numeric vector of estimates.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Giulia Marcon, <giuliamarcongm@gmail.com>

References

Marcon, G., Padoan, S.A., Naveau, P., Muliere, P. and Segers, J. (2017) Multivariate Nonparametric Estimation of the Pickands Dependence Function using Bernstein Polynomials. *Journal of Statistical Planning and Inference*, **183**, 1-17.

Naveau, P., Guillou, A., Cooley, D., Diebolt, J. (2009) Modelling pairwise dependence of maxima in space, Biometrika, **96**(1), 1-17.

See Also

beed, beed.confband

Examples

```
x <- ExtremalDep:::simplex(2)
data <- evd::rbvevd(50, dep = 0.4, model = "log", mar1 = c(1,1,1))

Amd <- madogram(x, data, "emp")
Amd.bp <- beed(data, x, 2, "md", "emp", 20, plot=TRUE)

lines(x[,1], Amd, lty = 1, col = 2)
```

MilanPollution *Pollution data for summer and winter months in Milan, Italy.*

Description

Two datasets Milan.summer and Milan.winter, each containing 5 air pollutants: daily maximum of NO2, NO, O3 and SO2, daily mean of PM10; and 6 meteorological covariates: maximum precipitation, maximum temperature, maximum humidity, mean precipitation, mean temperature and mean humidity.

Format

A $1968 * 12$ data frame and a $1924 * 12$ data frame.

Details

The summer period corresponds to the period 30 April - 30 August between 2003 and 2017 and thus the dataset contains 1968 observations. The winter period corresponds to the period 32 November - 27(28) February. The records start from 31 December 2001 until 30 December 2017 and thus the dataset contains 1924 observations.

pk.extst *Pickands dependence function for the Extremal Skew-t model.*

Description

Evaluates the bivariate and trivariate Pickands dependence function for the extremal skew-t model.

Usage

```
pk.extst(x, param)
```

Arguments

x	a vector of length 2 or 3 that belongs to the corresponding simplex.
param	the parameter vector, containing the dependence, shape and df parameters.

Details

In the bivariate case, there is 1 dependence parameter, 2 shape parameters and a degree of freedom. In the trivariate case, there is 3 dependence parameter, 3 shape parameters and a degree of freedom. Dependence parameters must be between -1 and 1, the degree of freedom must be positive.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

Examples

```
### Pickands dependence function for two-dimensional distribution

pk.extst(x=c(0.5,0.5), param=c(0.4,-2,4,3))

### Pickands dependence function for three-dimensional distribution

pk.extst(x=c(0.2,0.4,0.4), param=c(0.4,0.3,0.7,3,-1,0,2))
```

plot.bbeed *Plot of Extremal Dependence*

Description

Produces one or more plots of the extremal dependence.

Usage

```
## S3 method for class 'bbeed'
plot(x, type = c("summary", "returns", "A", "h", "pm", "k"),
          mcmc, summary.mcmc, nsim, burn, y, probs, CEX=1.5, A_true, h_true,
          labels=c(expression(y[1]),expression(y[2])), ...)
```

Arguments

x	Vector on the unit simplex where the dependence function is evaluated.
type	String, denoting the type of function to plot (see **Details**).
mcmc	The output of the bbeed function.
summary.mcmc	The output of the summary.bbeed function.
nsim	The number of simulation in the mcmc algorithm.
burn	The burn-in period.
y	A 2-column matrix of unobserved thresholds at which the returns are calculated. Required when type="y".
probs	The probability of joint exceedances, the output of the return function.
A_true	The true pickands dependence function (evaluated at x).
h_true	The true angular density function (evaluated at x).
CEX	Label and axis sizes.
labels	Labels.
...	Additional graphical parameters. See plot function for details.

Details

If type="returns", a (contour) plot of the probabilities of exceedances for some threshold y is returned.
If type="A", a plot of the estimated Pickands dependence function is drawn. If A_true is specified the plot includes the true Pickands dependence function and a functional boxplot for the estimated function. If type="h", a plot of the estimated angular density function is drawn. If h_true is specified the plot includes the true angular density and a functional boxplot for the estimated function. If type="pm", a plot of the prior against the posterior for the mass at $\{0\}$ is drawn. If type="k", a plot of the prior against the posterior for the polynomial degree k is drawn. If type="summary", a 2 by 2 plot with types "A", "h", "pm" and "k" is returned.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Giulia Marcon, <giuliamarcongm@gmail.com>

References

Marcon, G., Padoan, S.A., Naveau, P., Muliere, P., Segers, J. (2017) Multivariate Nonparametric Estimation of the Pickands Dependence Function using Bernstein Polynomials. *Journal of Statistical Planning and Inference*, **183**, 1-17.

See Also

beed.confband.

Examples

```
# This reproduces some of the results showed in Fig. 1 (Marcon, 2016).
set.seed(1890)
data <- evd::rbvevd(n=100, dep=.6, asy=c(0.8,0.3), model="alog", mar1=c(1,1,1))

nsim = 500000
burn = 400000

mu.nbinom = 3.2
var.nbinom = 4.48
hyperparam <- list(a.unif=0, b.unif=.5, mu.nbinom=mu.nbinom, var.nbinom=var.nbinom)
k0 = 5
pm0 = list(p0=0.06573614, p1=0.3752118)
eta0 = ExtremalDep:::rcoef(k0, pm0)

mcmc <- bbeed(data, pm0, eta0, k0, hyperparam, nsim,
              prior.k = "nbinom", prior.pm = "unif")

w <- seq(0.001, .999, length=100)
summary.mcmc <- summary.bbeed(w, mcmc, burn, nsim, plot=TRUE)

plot.bbeed(type = "A", x=w, mcmc=mcmc, summary.mcmc, nsim=nsim, burn=burn)
plot.bbeed(type = "h", x=w, mcmc=mcmc, summary.mcmc, nsim=nsim, burn=burn)
plot.bbeed(type = "pm", x=w, mcmc=mcmc, summary.mcmc, nsim=nsim, burn=burn)
plot.bbeed(type = "k", x=w, mcmc=mcmc, summary.mcmc, nsim=nsim, burn=burn)

Atrue <- evd::abvevd(w, dep=0.6, asy=c(0.3,0.8), model='alog')
htrue <- evd::hbvevd(w, dep=0.6, asy=c(0.8,0.3), model='alog',half=TRUE)

plot.bbeed(type = "A", summary.mcmc=summary.mcmc, A_true=Atrue)
plot.bbeed(type = "h", summary.mcmc=summary.mcmc, h_true=htrue)
```

pollution	*Air quality datasets containing daily maxima of air pollutants (PM10, NO, NO2, 03 and S02) recorded in Leeds (U.K.), during five winter seasons (November-Februrary) between 1994 and 1998.*

Description

Contains 6 datasets: PNS, PNN, NSN, PNNS, winterdat and Leeds.frechet.

Details

The dataset winterdat contains 590 (transformed) observations for each of the five pollutants. Contains NAs. Outliers have been removed according to Heffernan and Tawn (2004). The following datasets have been obtained by applying transformations to winterdat.

Leeds.frechet contains 590 observations corresponding to the daily maxima of five air pollutants transformed to unit Frechet scale.

NSN contains 100 observations in the 3-dimensional unit simplex for the daily maxima of nitrogen dioxide (NO2), sulfur dioxide (SO2) and nitrogen oxide (NO).

PNN contains 100 observations in the 3-dimensional unit simplex for the daily maxima of particulate matter (PM10), nitrogen oxide (NO) and nitrogen dioxide (NO2).

PNS contains 100 observations in the 3-dimensional unit simplex for the daily maxima of particulate matter (PM10), nitrogen oxide (NO) and sulfur dioxide (SO2).

PNNS contains 100 observations in the 4-dimensional unit simplex for the daily maxima of particulate matter (PM10), nitrogen oxide (NO), nitrogen dioxide (NO2) and sulfur dioxide (S02).

The transformation to unit Frechet margins of the raw data has been considered by Cooley et al (2010). Only the 100 data points with the largest radial components were kept.

Source

http://airquality.co.uk

References

Cooley, D.,Davis, R. A., and Naveau, P. (2010). The pairwise beta distribution: a flexible parametric multivariate model for extremes. *Journal of Multivariate Analysis*, **101**, 2103–2117.

Heffernan, J. E., and Tawn, J. A. (2004). A conditional approach for multivariate extreme values. *Journal of the Royal Statistical Society, Series B, Methodology*, **66**, 497–546

posteriorMCMC *MCMC sampler for parametric spectral measures*

Description

Generates a sample from the posterior distribution for the parameters and computes the posterior mean, component-wise variance and BIC.

Usage

```
posteriorMCMC(Nsim, Nbin=0, Hpar, MCpar, dat, par.start=NULL,
              show.progress=floor(seq(1,Nsim, length.out=20)),
            seed=NULL, kind="Mersenne-Twister", save=FALSE,
              name.save=NULL, save.directory = "~",
              name.dat="", model, c=NULL)
```

Arguments

Nsim	Total number of iterations to perform.
Nbin	Length of the burn-in period.
Hpar	A vector of hyper-parameters. See prior.
MCpar	MC MC parameter.See proposal.
dat	Angular dataset. Each row corresponds to coordinates in the simplex.
par.start	Starting point for the MC MC sample.
show.progress	A 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 the routine **set.seed**, see help of **R** for details.
kind	The kind of random number generator. Default is "Mersenne-Twister". See **set.seed** for details.
save	Logical; if save=TRUE then the result is saved
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, A "log" is also saved, named paste(name.save,".log",sep=""), in file paste(save.directory,"/",
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 dataset used for inference. Default is "".
model	A character string. Possible models are "Pairiwse", "Husler", "Dirichlet", "Extremalt" or "Asymmetric". See details.
c	A real value in $[0, 1]$, providing the decision rule to allocate a data point to a subset of the simplex. Only required for the Extremal-t and Asymmetric Logistic models.

Details

When `model="Pairwise"` the Pairiwse Beta model is selected and `prior.pb`, `proposal.pb`, `pb.Hpar` , `pb.MCpar` are considered. Similarly `model="Husler"` selects the Husler-Reiss model, `model="Dirichlet"` the Tilted Dirichlet model, `model="Extremalt"` the Extremal-t and `model="Asymmetric"` the Asymmetric Logistic model and the functions associated to these models.

Value

A list made of

`stored.vales`	A $(Nsim - Nbin) * d$ matrix, where d is the dimension of the parameter space
`llh`	A vector of size $(Nsim - Nbin)$ containing the log-likelihoods evaluadted 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 `porc.time` between the start and 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.
`BIC`	The Bayesian Information Criteria.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

Examples

```
################################################
# The following examples provide the results of
# the approximate bayesian analysis in Table 1.1
# of the paper Beranger and Padoan (2015)
################################################

## Load datasets :
data(pollution)
Nsim <- 50e+4
Nbin <- 30e+4
MCpar <- 0.35
Hpar.pb <-  list(mean.alpha=0, mean.beta=3,sd.alpha=3, sd.beta=3)
Hpar.hr <- list(mean.lambda=0, sd.lambda=3)
Hpar.di <- list(mean.alpha=0, sd.alpha=3)
Hpar.et <- list(mean.rho=0, mean.mu=3,sd.rho=3, sd.mu=3)
```

```
## Using the PNS dataset

est.pb.PNS <- posteriorMCMC(Nsim, Nbin, Hpar.pb, MCpar, PNS, seed=14342, model='Pairwise')
est.pb.PNS$emp.mean
est.pb.PNS$emp.sd
est.pb.PNS$BIC

est.hr.PNS <- posteriorMCMC(Nsim, Nbin, Hpar.hr, MCpar, PNS, seed=14342, model='Husler')
est.hr.PNS$emp.mean
est.hr.PNS$emp.sd
est.hr.PNS$BIC

est.di.PNS <- posteriorMCMC(Nsim, Nbin, Hpar.di, MCpar, PNS, seed=14342, model='Dirichlet')
est.di.PNS$emp.mean
est.di.PNS$emp.sd
est.di.PNS$BIC

est.et.PNS <- posteriorMCMC(Nsim, Nbin, Hpar.et, MCpar, PNS, seed=14342, model='Extremalt',c=0.1)
est.et.PNS$emp.mean
est.et.PNS$emp.sd
est.et.PNS$BIC


## Using the NSN dataset


est.pb.NSN <- posteriorMCMC(Nsim, Nbin, Hpar.pb, MCpar, NSN, seed=14342, model='Pairwise')
est.pb.NSN$emp.mean
est.pb.NSN$emp.sd
est.pb.NSN$BIC

est.hr.NSN <- posteriorMCMC(Nsim, Nbin, Hpar.hr, MCpar, NSN, seed=14342, model='Husler')
est.hr.NSN$emp.mean
est.hr.NSN$emp.sd
est.hr.NSN$BIC

est.di.NSN <- posteriorMCMC(Nsim, Nbin, Hpar.di, MCpar, NSN, seed=14342, model='Dirichlet')
est.di.NSN$emp.mean
est.di.NSN$emp.sd
est.di.NSN$BIC

est.et.NSN <- posteriorMCMC(Nsim, Nbin, Hpar.et, MCpar, NSN, seed=14342, model='Extremalt',c=0.1)
est.et.NSN$emp.mean
est.et.NSN$emp.sd
est.et.NSN$BIC


## Using the PNN dataset



est.pb.PNN <- posteriorMCMC(Nsim, Nbin, Hpar.pb, MCpar, PNN, seed=14342, model='Pairwise')
```

```
est.pb.PNN$emp.mean
est.pb.PNN$emp.sd
est.pb.PNN$BIC

est.hr.PNN <- posteriorMCMC(Nsim, Nbin, Hpar.hr, MCpar, PNN, seed=14342, model='Husler')
est.hr.PNN$emp.mean
est.hr.PNN$emp.sd
est.hr.PNN$BIC

est.di.PNN <- posteriorMCMC(Nsim, Nbin, Hpar.di, MCpar, PNN, seed=14342, model='Dirichlet')
est.di.PNN$emp.mean
est.di.PNN$emp.sd
est.di.PNN$BIC

est.et.PNN <- posteriorMCMC(Nsim, Nbin, Hpar.et, MCpar, PNN, seed=14342, model='Extremalt',c=0.1)
est.et.PNN$emp.mean
est.et.PNN$emp.sd
est.et.PNN$BIC


################################################
# The following examples provide the results of
# the approximate bayesian analysis in Table 1.2
# of the paper Beranger and Padoan (2015)
################################################


# Using the PNNS dataset



est.pb.PNNS <- posteriorMCMC(Nsim, Nbin, Hpar.pb, MCpar, PNNS, seed=14342, model='Pairwise')
est.pb.PNNS$BIC

est.hr.PNNS <- posteriorMCMC(Nsim, Nbin, Hpar.hr, MCpar, PNNS, seed=14342, model='Husler')
est.hr.PNNS$BIC

est.di.PNNS <- posteriorMCMC(Nsim, Nbin, Hpar.di, MCpar, PNNS, seed=14342, model='Dirichlet')
est.di.PNNS$BIC
```

`prior` *Prior parameter distribution for parametric models.*

Description

Random generation from the prior distribution for extremal parametric models or density evaluation of the extremal parametric models.

Usage

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

Arguments

model	The parametric model considered. Values can be model="Pairwise", model="Husler", model="Dirichlet", model="Extremalt" and model="Asymmetric" respectively for the Pairwise Beta, Hulser-Reiss, Tilted Dirichlet, Extremal-t and Asymmetric Logistic.
type	One of the character strings "r" or "d" representing random generation and density of the model considered.
n	The number of parameters to be generated. Only used if type=="r".
par	The values of the parameters. Only used if type=="d". The first elements correspond to the parameters alpha and the last parameters are the beta parameters. See **Details**.
Hpar	A list of hyper-parameters. See **Details**.
log	Logical; Only used if type=="d" in order to obtain the log-density. TRUE is the default.
dimData	The dimension of the simplex.

Details

- For the **Pairwise Beta model**, the parameters components are independent, log-normal. The vector of parameters is of size choose(dim,2)+1 with positive components. The first elements are the pairiwse dependence parameters b and the last one is the global dependence parameter alpha. The list of hyper-parameters should be of the form mean.alpha=,mean.beta=,sd.alpha=,sd.beta=;
- For the **Husler-Reiss model**, the parameters components are independent, log-normal. The vector of parameters is of size choose(dim,2)+1 with positive components. The elements correspond to the lambda parameter. The list of hyper-parameters should be of the form mean.lambda=,sd.lambda=;
- For the **Dirichlet model**, the parameters' components are independent, log-normal. The vector of parameters is of size dimData with positive components. The elements correspond to the alpha parameter. The list of hyper-parameters should be of the form mean.alpha=,sd.alpha=;
- For the **Extremal-t model**, the parameters' components are independent, logit-squared for rho and log-normal for mu. The vector of parameters is of size dimData with positive components. The first elements correspond to the correlation parameters rho and the last parameter is the global dependence parameter mu. The list of hyper-parameters should be of the form mean.rho=,mean.mu=,sd.rho=,sd.mu=;
- For the **Asymmetric Logistic model**, the parameters' components are independent, log(+1)-normal for alpha and logit for beta. The vector of parameters is of size 2^{dimData-1}(dimData+2)-(2 dimData+1)2^dimData-1(dimData+2)-(2 dimData+1) with positive components. The list of hyper-parameters should be of the form mean.alpha=,mean.beta=,sd.alpha=,sd.beta=.

Value

If type=="r", a matrix with n rows containing a random parameter sample generated under the prior is returned, the (log)-density is returned if type=="d".

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

Examples

```
MCpar <- 0.35
Hpar.pb <-  list(mean.alpha=0, mean.beta=3,sd.alpha=3, sd.beta=3)
Hpar.hr <- list(mean.lambda=0, sd.lambda=3)
Hpar.di <- list(mean.alpha=0, sd.alpha=3)
Hpar.et <- list(mean.rho=0, mean.mu=3,sd.rho=3, sd.mu=3)
Hpar.alm <- list(mean.alpha=0, mean.beta=0, sd.alpha=3, sd.beta=3)

prior(model="Pairwise", type="r", n=5, Hpar=Hpar.pb, dimData=3)
prior(model="Pairwise", type="d", par=rep(1,choose(4,2)+1), Hpar=Hpar.pb, log=TRUE, dimData=3)

prior(model="Husler", type="r", n=5, Hpar=Hpar.hr, dimData=3)
prior(model="Husler", type="d", par=rep(1,choose(4,2)), Hpar=Hpar.hr, log=TRUE, dimData=3)

prior(model="Dirichlet", type="r", n=5, Hpar=Hpar.di, dimData=3)
prior(model="Dirichlet", type="d", par=rep(1,3), Hpar=Hpar.di, log=TRUE, dimData=3)

prior(model="Extremalt", type="r", n=5, Hpar=Hpar.et, dimData=3)
prior(model="Extremalt", type="d", par=c(rep(0.1,3),4), Hpar=Hpar.et, log=TRUE, dimData=3)

prior(model="Asymmetric", type="r", n=5, Hpar=Hpar.alm, dimData=3)
prior(model="Asymmetric", type="d", par=c(rep(2,4),rep(0.7,9)), Hpar=Hpar.alm, log=TRUE, dimData=3)
```

`proposal` *Proposal distribution for parametric models*

Description

Density of the proposal distribution `q(cur.par,prop.par)` and random generator of the MCMC algorithm for parametric models.

Usage

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

Arguments

`model`	The parametric model considered. Values can be `model="Pairwise"`, `model="Husler"`, `model="Dirichlet"`, `model="Extremalt"` and `model="Asymmetric"` respectively for the Pairwise Beta, Hulser-Reiss, Tilted Dirichlet, Extremal-t and Asymmetric Logistic.
`type`	One of the character strings "r" or "d" representing random generation and density for the Asymmetric Logistic model.

cur.par	Vector representing the current state of the chain. See **Details**.
prop.par	Vector representing the candidate parameters. See **Details**.
MCpar	A list made of a single element: MC MC parameter. Re-centering parameters for the proposal distribution.
log	Logical; Only used if type=="d" in order to obtain the log-density. TRUE is the default.

Details

- For the Pairwise Beta model, cur.par and prop.par are of size choose(dim,2)+1. The components prop.par[i] of the proposal distribution are generated independently from the log-normal distribution. prop.par = rlnorm(length(cur.par),meanlog=log(cur.par),sdlog = rep(MCpar$sdlog,length(cur.par)));
- For the Husler-Reiss model, cur.par and prop.par are of size choose(dim,2). The components prop.par[i] of the proposal distribution are generated independently from the log-normal distribution. prop.par = rlnorm(length(cur.par),meanlog=log(cur.par),sdlog = rep(MCpar$sdlog,length(cur.par)));
- For the **Tilted Dirichlet model**, cur.par and prop.par are of size dim. The components prop.par[i] of the proposal distribution are generated independently from the log-normal distribution. prop.par = rlnorm(length(cur.par),meanlog=log(cur.par),sdlog = rep(MCpar$sdlog,length(c
- For the **Extremal-t model**, cur.par and prop.par are of size choose(dim,2)+1. The components prop.par[i] of the proposal distribution are generated independently from the square root of an inverse logit transformation of the normal distribution for the correlation parameters rho and from the log transformation of the normal distribution for the global dependence parameter (the degreee of freedom mu);
- For the **Asymmetric Logistic model**, cur.par and prop.par are of size $2^{d-1}(d+2)-(2d-1)$. The components prop.par[i] of the proposal distribution are generated independently from the log transformation of the normal distribution (minus 1) for the parameters alpha and from the inverse logit transformation of the parameters beta;

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".

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

Examples

```
proposal("Pairwise", "r", rep(1,4), MCpar=0.35)
proposal("Pairwise", "d", rep(1,4), rep(1.2,4), MCpar=0.35)

proposal("Husler", "r", rep(1,4), MCpar=0.35)
proposal("Husler", "d", rep(1,4), c(1.2,4), MCpar=0.35)
```

```
proposal("Dirichlet", "r", rep(1,3), MCpar=0.35)
proposal("Dirichlet", "d", rep(1,3), c(1.2,3), MCpar=0.35)

proposal("Extremalt", "r", rep(0.5,3), MCpar=0.35)
proposal("Extremalt", "d", c(rep(0.9,3),3), c(rep(0.8,3),2), MCpar=0.35)

proposal("Asymmetric", "r", c(rep(1.1,4),rep(0.1,9)), MCpar=0.35)
proposal("Asymmetric", "d", c(rep(1.1,4),rep(0.1,9)), c(rep(1.2,4),rep(0.1,9)), 0.35)
```

`returns` *Compute return values*

Description

Predicts the probability of future simultaneous exceedances

Usage

```
returns(mcmc, summary.mcmc, y, plot=FALSE)
```

Arguments

`mcmc`	The output of the `bbeed` function.
`summary.mcmc`	The output of the `summary.bbeed` function.
`y`	A 2-column matrix of unobserved thresholds.
`plot`	If `plot=TRUE`, then the `plot.bbeed` function is used.

Details

Computes for a range of unobserved extremes (larger than those observed in a sample), the pointwise mean from the posterior predictive distribution of such predictive values. The probabilities are calculated through

$$P(Y_1 > y_1, Y_2 > y_2) = \frac{2}{k}\sum_{j=0}^{k-2}(\eta_{j+1}-\eta_j)\times\left(\frac{(j+1)B(y_1/(y_1+y_2)|j+2,k-j-1)}{y_1} - \frac{(k-j-1)B(y_2/(y_1+y_2)|k-j,j+1)}{y_2}\right),$$

where $B(x|a,b)$ denotes the cumulative distribution function of a Beta random variable with shape $a, b > 0$. See Marcon et al. (2016, p.3323) for details.

Value

Returns a vector whose length is equal to the number of rows of the input value `y`.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Giulia Marcon, <giuliamarcongm@gmail.com>

References

Marcon G., Padoan, S.A. and Antoniano-Villalobos I. (2016) Bayesian Inference for the Extremal Dependence. *Electronic Journal of Statistics*, 10.2, 3310-3337.

Examples

```
# This reproduces some of the results shown in Fig. 1 (Marcon, 2016).
set.seed(1890)
data <- evd::rbvevd(n=100, dep=.6, asy=c(0.8,0.3), model="alog", mar1=c(1,1,1))

nsim = 500000
burn = 400000

mu.nbinom = 3.2
var.nbinom = 4.48
hyperparam <- list(a.unif=0, b.unif=.5, mu.nbinom=mu.nbinom, var.nbinom=var.nbinom)
k0 = 5
pm0 = list(p0=0.06573614, p1=0.3752118)
eta0 = ExtremalDep:::rcoef(k0, pm0)

mcmc <- bbeed(data, pm0, eta0, k0, hyperparam, nsim,
              prior.k = "nbinom", prior.pm = "unif")

w <- seq(0.001, .999, length=100)
summary.mcmc <- summary.bbeed(w, mcmc, burn, nsim, plot=TRUE)

plot.bbeed(type = "A", x=w, mcmc=mcmc, summary.mcmc, nsim=nsim, burn=burn)
plot.bbeed(type = "h", x=w, mcmc=mcmc, summary.mcmc, nsim=nsim, burn=burn)
plot.bbeed(type = "pm", x=w, mcmc=mcmc, summary.mcmc, nsim=nsim, burn=burn)
plot.bbeed(type = "k", x=w, mcmc=mcmc, summary.mcmc, nsim=nsim, burn=burn)

y <- seq(10,100,2)
y <- as.matrix(expand.grid(y,y))
probs <- returns(mcmc = mcmc, summary.mcmc = summary.mcmc, y = y, plot = TRUE)
```

`r_extr_mod` *Random sample generation from extremal dependence models*

Description

Generates random samples of iid observations from the Extremal-t and Extremal Skew-t models.

Usage

```
r_extr_mod(model, n, param)
```

Arguments

`model`	A string with the name of the model: `"Extremalt"` or `"Skewt"`.
`n`	An integer indicating the number of numbers to be generated.
`param`	A vector containing the parameters of the model. See **Details**.

Details

If `model="Extremalt"` then the parameter vector is made of a dependence parameter vector of size $choose(dim, 2)$ and a degree of freedom. If `model="Skewt"` then the parameter vector is made of a dependence parameter vector of size $choose(dim, 2)$, a vector of shape (or skewness) parameters of size dim and a degree of freedom.

Value

Returns a matrix with dim columns and n columns.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan;
Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com;

References

Beranger, B., Padoan, S. A. and Sisson, S. A. (2017). Models for extremal dependence derived from skew-symmetric families. *Scandinavian Journal of Statistics*, **44**(1), 21-45.

Examples

```
### Extremal-t


## Bivariate
r_extr_mod("Extremalt", n=5, par=c(0.5,2))



## Trivariate
r_extr_mod("Extremalt", n=5, par=c(0.5,0.6,0.4,2))


### Extremal Skew-t


## Bivariate
r_extr_mod("Skewt", n=5, par=c(0.5,-1,1,2))



## Trivariate
r_extr_mod("Skewt", n=5, par=c(0.5,0.6,0.4,-2,-2,5,2))
```

summary.bbeed *Compute summary statistics from the MCMC output.*

Description

Summary statistics of the MCMC output obtained from the Bayesian method based on the Bernstein polynomials for inferring the angular measure and Pickands dependence functions.

Usage

```
  ## S3 method for class 'bbeed'
summary(object, mcmc, burn, conf=0.95, plot=FALSE, ...)
```

Arguments

object	The values (on the simplex) at which the dependence is evaluated
mcmc	The output of an MCMC algorithm given by beed
burn	The burn-in period
conf	The confidence region
plot	If plot=TRUE then the function plot.beed is used.
...	Arguments to be passed for the graphical parameters

Value

Returns a list that contains:
* the conf-, 0.5- and 1-conf-quantiles and posterior sample for k (the polynomial order),
* the conf- and 1-conf-quantiles, mean and posterior sample for h (the angular density), A (the Pickands dependence function), p0 and p1 (the point masses at the endpoints of the simplex), mar1 and mar2 (the marginal parameters, if they exist). To access them, the names are for example k.low, k.median, k.up and k_post.
* w and burn which are the inputs object and burn.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com; Giulia Marcon, <giuliamarcongm@gmail.com>

References

Marcon G., Padoan, S.A. and Antoniano-Villalobos I. (2016) Bayesian Inference for the Extremal Dependence. *Electronic Journal of Statistics*, 10.2, 3310-3337.

See Also

plot.bbeed.

Examples

```
# This reproduces some of the results showed in Fig. 1 (Marcon, 2016).
set.seed(1890)
data <- evd::rbvevd(n=100, dep=.6, asy=c(0.8,0.3), model="alog", mar1=c(1,1,1))

nsim = 500000
burn = 400000

mu.nbinom = 3.2
var.nbinom = 4.48
hyperparam <- list(a.unif=0, b.unif=.5, mu.nbinom=mu.nbinom, var.nbinom=var.nbinom)
k0 = 5
pm0 = list(p0=0.06573614, p1=0.3752118)
eta0 = ExtremalDep:::rcoef(k0, pm0)

mcmc <- bbeed(data, pm0, eta0, k0, hyperparam, nsim,
              prior.k = "nbinom", prior.pm = "unif")

w <- seq(0.001, .999, length=100)
summary.mcmc <- summary.bbeed(w, mcmc, burn, nsim, plot=TRUE)
```

UniExtQ *Univariate Extreme Quantile*

Description

Computes the extreme-quantiles of a univariate random variable corresponding to some exceedance probabilities.

Usage

```
UniExtQ(data, P=NULL, method="bayesian", U=NULL,
cov=as.matrix(rep(1,length(data))), QatCov=NULL,
param0=NULL, sig0=NULL, nsim=NULL, p=0.234,
optim.meth="BFGS", control=NULL, ...)
```

Arguments

data	A vector of n observations.
P	The vector of probabilities associated to the quantiles.
method	The estimation method can be "bayesian" or "frequentist".

U	The threshold value under which the observations are censored.
cov	A $n \times c$ matrix of covariates for the location parameter.
QatCov	q $n \times c$ matrix with the value of the covariates at which the quantiles should be computed.
param0	The vector of initial value for the parameters. It should be of length $c + 2$
sig0	Only required when method="bayesian". Initial value for the standard deviation of the multivarial normal proposal distribution.
nsim	Only required when method="bayesian". Number of iterations in the Metropolis-Hastings algorithm.
p	Only required when method="bayesian". Targeted acceptance ratio.
optim.meth	Only required when method="frequentist". Optimisation algorithm as defined in optim function.
control	Only required when method="frequentist". See details of optim function.
...	Only required when method="frequentist". See details of optim function.

Details

For some dataset given by data, the extreme-quantiles corresponding to some exceedance probability(ies) given in P are computed. The observations below the threshold U are considered censored.

- If cov is specified then a linear model for the location parameter is fitted and QatCov must be specified. It sets the value of the covariates at which the extreme quantiles are evaluated;
- If method=="frequentist" then the GEV parameters are estimated via optimisation of the censored likelihood;
- If method=="bayesian" the the GEV parameters are estimated via a Metropolis-Hastings algorithm, where the proposal distirbution is a multivariate normal. After running for nsim iteration the algorithm is paused and some diagnostics plots are printed. The user then needs to enter the value of the burn-in period.

Refer to Beranger et al. (2019) for more details about the estimation procedures.

Value

If method=="frequentist", a list with elements:

- Q.est: a matrix of extreme quantiles where each of the length(P) rows represents an associated probability P and each of the nrow(QatCov) columns represents a level of the covariates;
- kn: the proportion of observation above the threshold U, corresponds to $\frac{k}{n}$;
- est: the vector of estimated parameters from the maximisation of the censored likelihood (on the log scale);
- VarCov: the variance-covariance matrix of the parameter estimates. Available if the hessian=TRUE is specified.

If method=="bayesian", a list with elements:

- Q.est: a list of extreme quantiles where each element is a vector and refers to a probability specified by P;

- `post_sample`: a matrix containing the posterior sample of each parameter. The number of columns should be equal to `ncol(cov)+2`;
- `straight.reject`: the number of straight rejections from the proposal distribution;
- `sig.vec`: the vector of updated scale parameter in the multivariate normal proposal.

Author(s)

Simone Padoan, <simone.padoan@unibocconi.it>, http://faculty.unibocconi.it/simonepadoan; Boris Beranger, <borisberanger@gmail.com> http://www.borisberanger.com

References

Beranger, B., Padoan S. A. and Sisson, S. A. (2019). Estimation and uncertainty quantification for extreme quantile regions. *arXiv e-prints* arXiv:1904:08251.

See Also

ExtQset

Examples

```
distribution <- "FRE" # MDA(FRE), Tail index "xi"

cat("\n Distribution:", distribution, "\n")
set.seed(999)
n <- 1500
loc <- 3
scale <- 1
shape <- 1/3
prob <- 0.9
cov <- as.matrix(rep(1,n)) # No covariates

data <- rfrechet(n = n, mu = loc, sigma = scale, lambda =shape)
pars <- c(loc, scale, shape)
pars.name <- paste("loc", loc*10, "_scale", scale*10, "_shape", shape*10, sep="")

### Required for Bayesian Estimation
nsim <- 5e+4
param0 <- c(1, 2, 1)
P <-c(1/750, 1/1500, 1/3000)
U <- quantile(data, probs=prob, type=3)
sig0 <- 1

### Estimation

# Bayesian approach:
mcmc.name <- paste("mcmc",distribution,"_n", n, "_prob", prob, "_", pars.name,sep="")
op <- par(mar=c(4.2,4.6,0.3,0.1))
```

```
assign(mcmc.name, UniExtQ(data=data, P=P, U=U, cov=cov, param0=param0, sig0=sig0,
   nsim=nsim))
### RUN UNTIL HERE AND SPECIFY BURN-IN PERIOD

# Frequentist approach:
q <- UniExtQ(data=data, method="frequentist", P=P, param0=param0,
 control = list(maxit = 5e+5, trace = 2))

### Illustration

ti <- 1/shape
mcmc <- get(mcmc.name)
Kern <- density(mcmc$post_sample[,ncol(cov)+2]) # KDE of the tail index

Hist <- hist(mcmc$post_sample[,ncol(cov)+2], prob=TRUE, col="lightgrey",
 ylim=range(Kern$y), main="", xlab="Tail Index",
 cex.lab=1.8, cex.axis=1.8, lwd=2)
ti_ic <- quantile(mcmc$post_sample[,ncol(cov)+2], probs=c(0.025, 0.975))
points(x=ti_ic, y=c(0,0), pch=4, lwd=4)
lines(Kern, lwd = 2, col = "dimgrey")
abline(v=ti, lwd=2)
abline(v=mean(mcmc$post_sample[,ncol(cov)+2]), lwd=2, lty=2)

#### Check the ability to estimate quantile regions
Q <- qfrechet(p = P, mu = loc, sigma = scale, lambda = shape, lower.tail=FALSE)

ci <- apply(log(mcmc$Q.est),2,function(x) quantile(x, probs=c(0.025, 0.975)))
for(i in 1:length(P)){
  cat("\n Confidence interval extreme quantile with Probability ", P[i],
  " : (", ci[1,i],",",ci[2,i],") \n")
}

Kern.est <- apply(log(mcmc$Q.est),2,density)

R <- range(log(c(Q,mcmc$Q.est,data)))
Xlim <- c(log(quantile(data,0.95)), R[2])
Ylim <- c(0, max(Kern.est[[1]]$y, Kern.est[[2]]$y, Kern.est[[3]]$y))

plot(log(data), rep(0,n), pch=16, main="", xlim=Xlim, ylim=Ylim,
 xlab=expression(log(x)), ylab="Density", cex.lab=1.8, cex.axis=1.8, lwd=2)
polygon(Kern.est[[1]], col= rgb(211,211,211, 0.8*255, maxColorValue = 255),
border=rgb(211,211,211, 255, maxColorValue = 255), lwd=4)
polygon(Kern.est[[2]], col= rgb(169,169,169, 0.8*255, maxColorValue = 255),
border=rgb(169,169,169, maxColorValue = 255), lwd=4)
polygon(Kern.est[[3]], col= rgb(105,105,105, 0.8*255, maxColorValue = 255),
border=rgb(105,105,105, maxColorValue = 255), lwd=4)
points(log(data), rep(0,n), pch=16, lwd=2)
abline(v=log(Q), lwd=2, lty=1)
for(j in 1:length(P)){abline(v=mean(log(mcmc$Q.est[,j])), lwd=2, lty=2)}
abline(v=log(q$Q.est), lwd=2, lty=3)

par(op)
```

`Wind` *Weekly maximum wind speed data collected over 4 stations across Oklahoma, USA, over the March-May preiod between 1996 and 2012.*

Description

There are four datasets of weekly maximum wind speed data, for each triplet of locations: `CLOU.CLAY.SALL`, `CLOU.CLAY.PAUL`, `CLAY.SALL.PAUL` and `CLOU.SALL.PAUL`.

Details

`CLOU.CLAY.SALL` is a `data.frame` object with 3 columns and 212 rows. `CLOU.CLAY.PAUL` is a `data.frame` object with 3 columns and 217 rows. `CLAY.SALL.PAUL` is a `data.frame` object with 3 columns and 211 rows. `CLOU.SALL.PAUL` is a `data.frame` object with 3 columns and 217 rows. Missing observations have been discarded for each triplet.

References

Beranger, B., Padoan, S. A. and Sisson, S. A. (2017). Models for extremal dependence derived from skew-symmetric families. *Scandinavian Journal of Statistics*, **44**(1), 21-45.

Index

∗Topic **Bayesian Nonparametric**
bbeed, 10
∗Topic **Bootstrap confidence bands**
beed.confband, 17
∗Topic **Nonparametric**
beed, 13
plot.bbeed, 47
∗Topic **aplot**
summary.bbeed, 60
∗Topic **bootstrap**
beed.boot, 15
∗Topic **datasets**
MilanPollution, 45
pollution, 49
Wind, 65
∗Topic **distribution**
desn, 27
dest, 28
dmesn, 29
dmest, 31
∗Topic **hplot**
AngDensPlot, 5
∗Topic **htest**
ExtQset, 37
UniExtQ, 61
∗Topic **models**
chi.bsn, 19
chi.extst, 20
dens, 21
dens_extr_mod, 26
ellipse, 32
excess_pr_extr_mod, 34
exponent_extr_mod, 35
fit_pclik_extr_mod, 42
madogram, 44
pk.extst, 46
posteriorMCMC, 50
prior, 53
proposal, 55
r_extr_mod, 58
returns, 57
∗Topic **optimize**
alik, 3

alik, 3, *5*
AngDensPlot, 5
angular, 8

bbeed, 10, *37–39*
beed, 13, *16–19*, *38, 39*, *45*
beed.boot, 15, *18, 19*
beed.confband, *15*, *17*, 17, *45*, *48*

chi.bsn, 19
chi.extst, 20
CLAY.SALL.PAUL *(*Wind*)*, 65
CLOU.CLAY.PAUL *(*Wind*)*, 65
CLOU.CLAY.SALL *(*Wind*)*, 65
CLOU.SALL.PAUL *(*Wind*)*, 65

dens, 21
dens_extr_mod, 26
desn, 27
dest, 28
dmesn, 29
dmest, 31

ellipse, 32
excess_pr_extr_mod, 34
exponent_extr_mod, 35
ExtQset, 37, *63*

fit_pclik_extr_mod, 41

Leeds.frechet *(*pollution*)*, 49

madogram, 44
Milan.summer *(*MilanPollution*)*, 45
Milan.winter *(*MilanPollution*)*, 45
MilanPollution, 45

NSN *(pollution)*, 49

pesn *(desn)*, 27
pest *(dest)*, 28
pk.extst, 46
plot.bbeed, 47, *61*
pmesn *(dmesn)*, 29
pmest *(dmest)*, 31
PNN *(pollution)*, 49
PNNS *(pollution)*, 49
PNS *(pollution)*, 49
pollution, 49
posteriorMCMC, 50
prior, *50*, 53
proposal, *50*, 55

r_extr_mod, 58
returns, 57

summary.bbeed, *12*, 60

UniExtQ, *38, 39*, 61

Wind, 65
winterdat *(pollution)*, 49