Package 'ExtremalDep'

December 18, 2024

Version 0.0.4-4

Date 2024-12-17

Title Extremal Dependence Models

Author Boris Beranger [aut],
Simone Padoan [cre, aut],
Giulia Marcon [aut],
Steven G. Johnson [ctb] (Author of included cubature fragments),
Rudolf Schuerer [ctb] (Author of included cubature fragments),
Brian Gough [ctb] (Author of included cubature fragments),
Alec G. Stephenson [ctb],
Anne Sabourin [ctb] (Author of included BMAmevt fragments),
Philippe Naveau [ctb] (Author of PAMfmado function)

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

Imports numDeriv, evd, sn, quadprog, copula, nloptr, gtools, mvtnorm,
fda, parallel, doParallel, foreach, cluster

Suggests fields, extraDistr

BugReports https://github.com/borisberanger/ExtremalDep/issues

Description
A set of procedures for parametric and non-parametric modelling of the dependence structure of multivariate extreme-values is provided. The statistical inference is performed with non-parametric estimators, likelihood-based estimators and Bayesian techniques. It adapts the methodologies of Beranger and Padoan (2015) <doi:10.48550/arXiv.1508.05561>, Marcon et al. (2016) <doi:10.1214/16-EJS1162>, Marcon et al. (2017) <doi:10.1002/sta4.145>, Marcon et al. (2017) <doi:10.1016/j.jspi.2016.10.004> and Beranger et al. (2021) <doi:10.1007/s10687-019-00364-0>. This package also allows for the modelling of spatial extremes using flexible max-stable processes. It provides simulation algorithms and fitting procedures relying on the Stephenson-Tawn likelihood as per Beranger at al. (2021) <doi:10.1007/s10687-020-00376-1>.

License GPL (>= 2)

LazyData yes

NeedsCompilation yes

Repository CRAN

Repository/R-Forge/Project extremaldep

Repository/R-Forge/Revision 107

Repository/R-Forge/DateTimeStamp 2019-08-12 22:22:37

Date/Publication 2024-12-18 16:30:06 UTC

Depends R (>= 3.5.0)

URL https://faculty.unibocconi.it/simonepadoan/

Contents

`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>, https://faculty.unibocconi.it/simonepadoan/; Boris Beranger, <borisberanger@gmail.com> https://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)
```

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>`, https://faculty.unibocconi.it/simonepadoan/; Boris Beranger, `<borisberanger@gmail.com>` https://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 <- 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 <- 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>, https://faculty.unibocconi.it/simonepadoan/; Boris Beranger, <borisberanger@gmail.com> https://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 <- simplex(2)
data <- evd::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>, https://faculty.unibocconi.it/simonepadoan/; Boris Beranger, <borisberanger@gmail.com> https://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 <- simplex(2)
data <- evd::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)
```

---

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>, https://faculty.unibocconi.it/simonepadoan/;
Boris Beranger, <borisberanger@gmail.com> https://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>, https://faculty.unibocconi.it/simonepadoan/;
Boris Beranger, <borisberanger@gmail.com> https://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)
```

---

dExtDep *Parametric and non-parametric density of Extremal Dependence*

Description

This function calculates the density of parametric multivariate extreme distributions and corresponding angular density, or the non-parametric angular density represented through Bernstein polynomials.

Usage

```
dExtDep(x, method="Parametric", model, par, angular=TRUE, log=FALSE,
        c=NULL, vectorial=TRUE, mixture=FALSE)
```

Arguments

`x`	A vector or a matrix. The value at which the density is evaluated.
`method`	A character string taking value `"Parametric"` or `"NonParametric"`
`model`	A string with the name of the model: `"PB"` (Pairwise Beta), `"HR"` (Husler-Reiss), `"ET"` (Extremal-t), `"EST"` (Extremal Skew-t), `TD` (Tilted Dirichlet) or `AL` (Asymmetric Logistic) when evaluating the angular density. Restricted to `"HR"`, `"ET"` and `"EST"` for multivariate extreme value densities. Required when `method="Parametric"`.
`par`	A vector representing the parameters of the (parametric or non-parametric) model.
`angular`	A logical value specifying if the angular density is computed.
`log`	A logical value specifying if the log density is computed.

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 parametric angular density of the Extremal-t, Extremal Skew-t and Asymmetric Logistic models.

vectorial
: A logical value; if `TRUE` a vector of `nrow(x)` densities is returned. If `FALSE` the likelihood function is returned (product of densities or sum if `log=TRUE`. `TRUE` is the default.

mixture
: A logical value specifying if the Bernstein polynomial representation of distribution should be expressed as a mixture. If `mixture=TRUE` the density integrates to 1. Required when `method="NonParametric"`.

Details

Note that when `method="Parametric"` and `angular=FALSE`, the density is only available in 2 dimensions. When `method="Parametric"` and `angular=TRUE`, the models `"AL"`, `"ET"` and `"EST"` are limited to 3 dimensions. This is because of the existence of mass on the subspaces on the simplex (and therefore the need to specify `c`).

For the parametric models, the appropriate length of the parameter vector can be obtained from the `dim_ExtDep` function and are summarized as follows. When `model="HR"`, the parameter vector is of length `choose(dim,2)`. When `model="PB"` or `model="Extremalt"`, the parameter vector is of length `choose(dim,2) + 1`. When `model="EST"`, the parameter vector is of length `choose(dim,2) + dim + 1`. When `model="TD"`, the parameter vector is of length `dim`. When `model="AL"`, the parameter vector is of length `2^(dim-1)*(dim+2) - (2*dim+1)`.

Value

If `x` is a matrix and `vectorial=TRUE`, a vector of length `nrow(x)`, otherwise a scalar.

Author(s)

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

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

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.

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.

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

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.

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.

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

See Also

pExtDep, rExtDep, fExtDep, fExtDep.np

Examples

```
# Example of PB on the 4-dimensional simplex
dExtDep(x=rbind(c(0.1,0.3,0.3,0.3),c(0.1,0.2,0.3,0.4)), method="Parametric",
        model="PB", par=c(2,2,2,1,0.5,3,4), log=FALSE)

# Example of EST in 2 dimensions
dExtDep(x=c(1.2,2.3), method="Parametric", model="EST", par=c(0.6,1,2,3), angular=FALSE, log=TRUE)

# Example of non-parametric angular density
beta <- c(1.0000000, 0.8714286, 0.7671560, 0.7569398,
          0.7771908, 0.8031573, 0.8857143, 1.0000000)
dExtDep(x=rbind(c(0.1,0.9),c(0.2,0.8)), method="NonParametric", par=beta)
```

---

dGEV *The Generalized Extreme Value Distribution*

Description

Density, distribution and quantile function for the Generalized Extreme Value (GEV) distribution.

Usage

```
dGEV(x, loc, scale, shape, log=FALSE)
pGEV(q, loc, scale, shape, lower.tail=TRUE)
qGEV(p, loc, scale, shape)
```

Arguments

x, q	vector of quantiles.
p	vector of probabilities.
loc	vector of locations.
scale	vector of scales.

shape	vector of shapes.
log	A logical value; if TRUE returns the log density.
lower.tail	A logical value; if TRUE probabilities are $P(X \leq x)$, otherwise $P(X > x)$.

Details

The GEV distribution has density

$$f(x;\mu,\sigma,\xi) = \exp\left\{ -\left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]_+^{-1/\xi} \right\}$$

Value

density (dGEV), distribution function (pGEV) and quantile function (qGEV) from the Generalized Extreme Value distirbution with given location, scale and shape.

Author(s)

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

See Also

fGEV

Examples

```
# Densities
dGEV(x=1, loc=1, scale=1, shape=1)
dGEV(x=c(0.2, 0.5), loc=1, scale=1, shape=c(0,0.3))

# Probabilities
pGEV(q=1, loc=1, scale=1, shape=1, lower.tail=FALSE)
pGEV(q=c(0.2, 0.5), loc=1, scale=1, shape=c(0,0.3))

# Quantiles
qGEV(p=0.5, loc=1, scale=1, shape=1)
qGEV(p=c(0.2, 0.5), loc=1, scale=1, shape=c(0,0.3))
```

---

diagnostics *Diagnostics plots for MCMC algorithm.*

Description

This function displays traceplots of the scaling parameter from the proposal distribution of the adaptive MCMC scheme and the associated acceptance probability.

Usage

```
diagnostics(mcmc)
```

Arguments

mcmc	An output of the fGEV or fExtDep.np function with method="Bayesian".

Details

When mcmc is the output of fGEV then this corresponds to a marginal estimation and therefore diagnostics will display in a first plot the value of τ the scaling parameter in the multivariate normal proposal which directly affects the acceptance rate of the proposal parameter values that are displayed in the second plot.

When mcmc is the output of fExtDep.np, then this corresponds to an estimation of the dependence structure following the procedure given in Algorithm 1 of Beranger et al. (2021). If the margins are jointly estimated with the dependence (step 1 and 2 of the algorithm) then diagnostics provides trace plots of the corresponding scaling parameters (τ_1, τ_2) and acceptance probabilities. For the dependence structure (step 3 of the algorithm), a trace plot of the polynomial order κ is given with the associated acceptance probability.

Value

a graph of traceplots of the scaling parameter from the proposal distribution of the adaptive MCMC scheme and the associated acceptance probability.

Author(s)

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

References

Beranger, B., Padoan, S. A. and Sisson, S. A. (2021). Estimation and uncertainty quantification for extreme quantile regions. *Extremes*, **24**, 349-375.

See Also

fExtDep.np.

Examples

```
###################################################
### Example - Pollution levels in Milan, Italy ###
###################################################

## Not run:

### Here we will only model the dependence structure
data(MilanPollution)
```

```
data <- Milan.winter[,c("NO2","SO2")]
data <- as.matrix(data[complete.cases(data),])

# Thereshold
u <- apply(data, 2, function(x) quantile(x, prob=0.9, type=3))

# Hyperparameters
hyperparam <- list(mu.nbinom = 6, var.nbinom = 8, a.unif=0, b.unif=0.2)

### Standardise data to univariate Frechet margins

f1 <- fGEV(data=data[,1], method="Bayesian", sig0 = 0.1, nsim = 5e+4)
diagnostics(f1)
burn1 <- 1:30000
gev.pars1 <- apply(f1$param_post[-burn1,],2,mean)
sdata1 <- trans2UFrechet(data=data[,1], pars=gev.pars1, type="GEV")

f2 <- fGEV(data=data[,2], method="Bayesian", sig0 = 0.1, nsim = 5e+4)
diagnostics(f2)
burn2 <- 1:30000
gev.pars2 <- apply(f2$param_post[-burn2,],2,mean)
sdata2 <- trans2UFrechet(data=data[,2], pars=gev.pars2, type="GEV")

sdata <- cbind(sdata1,sdata2)

### Bayesian estimation using Bernstein polynomials

pollut1 <- fExtDep.np(method="Bayesian", data=sdata, u=TRUE,
                      mar.fit=FALSE, k0=5, hyperparam = hyperparam, nsim=5e+4)

diagnostics(pollut1)


## End(Not run)
```

---

dim_ExtDep *Dimensions calculations for parametric extremal dependence models*

Description

This function calculates the dimensions of an extremal dependence model for a given set of parameters, the dimension of the parameter vector for a given dimension and verifies the adequacy between model dimension and length of parameter vector when both are provided.

Usage

```
dim_ExtDep(model, par=NULL, dim=NULL)
```

Arguments

`model`	A string with the name of the model: `"PB"` (Pairwise Beta), `"HR"` (Husler-Reiss), `"ET"` (Extremal-t), `"EST"` (Extremal Skew-t), `TD` (Tilted Dirichlet) or `AL` (Asymmetric Logistic).
`par`	A vector representing the parameters of the model.
`dim`	An integer representing the dimension of the model.

Details

One of `par` or `dim` need to be provided. If `par` is provided, the dimension of the model is calculated. If `dim` is provided, the length of the parameter vector is calculated. If both `par` and `dim` are provided, the adequacy between the dimension of the model and the length of the parameter vector is checked.

For `model="HR"`, the parameter vector is of length `choose(dim,2)`. For `model="PB"` or `model="Extremalt"`, the parameter vector is of length `choose(dim,2) + 1`. For `model="EST"`, the parameter vector is of length `choose(dim,2) + dim + 1`. For `model="TD"`, the parameter vector is of length `dim`. For `model="AL"`, the parameter vector is of length `2^(dim-1)*(dim+2) - (2*dim+1)`.

Value

If `par` is not provided and `dim` is provided: returns an integer indicating the length of the parameter vector. If `par` is provided and `dim` is not provided: returns an integer indicating the dimension of the model. If `par` and `dim` are provided: returns a `TRUE/FALSE`statement indicating whether the length of the parameter and the dimension match.

Author(s)

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

Examples

```
dim_ExtDep(model="EST", dim=3)
dim_ExtDep(model="AL", dim=3)

dim_ExtDep(model="PB", par=rep(0.5,choose(4,2)+1) )
dim_ExtDep(model="TD", par=rep(1,5) )

dim_ExtDep(model="EST", dim=2, par=c(0.5,1,1,1) )
dim_ExtDep(model="PB", dim=4, par=rep(0.5,choose(4,2)+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>, https://faculty.unibocconi.it/simonepadoan/;
Boris Beranger, <borisberanger@gmail.com> https://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>, https://faculty.unibocconi.it/simonepadoan/;
Boris Beranger, <borisberanger@gmail.com> https://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>, https://faculty.unibocconi.it/simonepadoan/; Boris Beranger, <borisberanger@gmail.com> https://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)
```

ExtQ *Univariate Extreme Quantile*

Description

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

Usage

```
ExtQ(P=NULL, method="Frequentist", pU=NULL,
     cov=NULL, param=NULL,  param_post=NULL)
```

Arguments

P	A vector with values in $[0,1]$ indicating the probabilities of the quantiles to be computed.
method	A character string indicating the estimation method. Takes value "bayesian" or "frequentist".
pU	A value in $[0,1]$ indicating the probability of exceeding a high threshold. In the estimation procedure, observations below the threshold are censored.
cov	A $q \times c$ matrix indicating q observations of $c-1$ covariates for the location parameter.
param	A $(c+2)$ vector indicating the estimated parameters. Required when method="Frequentist".
param_post	A $n \times (c+2)$ matrix indicating the posterior sample for the parameters, where n is the number of MCMC replicates after removal of the burn-in period. Required when method="Bayesian".

Details

The first column of cov is a vector of 1s corresponding to the intercept.

When pU is NULL (default), then it is assumed that a block maxima approach was taken and quantiles are computed using the qGEV function. When pU is provided, the it is assumed that a threshold exceedances approach is taken and the quantiles are computed as

$$\mu + \sigma * \left(\left(\frac{pU}{P} \right)^{\xi} - 1 \right) \frac{1}{\xi}.$$

Value

When method=="frequentist", the function returns a vector of length length(P) if ncol(cov)=1 (constant mean) or a (length(P) x nrow(cov)) matrix if ncol(cov)>1.

When method=="bayesian", the function returs a (length(param_post) x length(P)) matrix if ncol(cov)=1 or a list of ncol(cov) elements each taking a (length(param_post) x length(P)) matrix if ncol(cov)>1.

Author(s)

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

References

Beranger, B., Padoan, S. A. and Sisson, S. A. (2021). Estimation and uncertainty quantification for extreme quantile regions. *Extremes*, **24**, 349-375.

See Also

fGEV, qGEV

Examples

```
###################################################
### Example - Pollution levels in Milan, Italy ###
###################################################

## Not run:

data(MilanPollution)

# Frequentist estimation
fit <- fGEV(Milan.winter$PM10)
fit$est

q1 <- ExtQ(P=1/c(600,1200,2400), method="Frequentist", param=fit$est)
q1

# Bayesian estimation with high threshold
cov <- cbind(rep(1,nrow(Milan.winter)), Milan.winter$MaxTemp,
             Milan.winter$MaxTemp^2)
u <- quantile(Milan.winter$PM10, prob=0.9, type=3, na.rm=TRUE)

fit2 <- fGEV(data=Milan.winter$PM10, par.start=c(50,0,0,20,1),
               method="Bayesian", u=u, cov=cov, sig0=0.1, nsim=5e+4)

r <- range(Milan.winter$MaxTemp, na.rm=TRUE)
t <- seq(from=r[1], to=r[2], length=50)
pU <- mean(Milan.winter$PM10>u, na.rm=TRUE)
q2 <- ExtQ(P=1/c(600,1200,2400), method="Bayesian", pU=pU,
           cov=cbind(rep(1,50), t, t^2),
           param_post=fit2$param_post[-c(1:3e+4),])

R <- c(min(unlist(q2)), 800)
qseq <- seq(from=R[1],to=R[2], length=512)
Xl <- "Max Temperature"
Yl <- expression(PM[10])

for(i in 1:length(q2)){
  K_q2 <- apply(q2[[i]],2, function(x) density(x, from=R[1], to=R[2])$y)
```

```
  D <- cbind(expand.grid(t, qseq), as.vector(t(K_q2)) )
  colnames(D) <- c("x","y","z")
  fields::image.plot(x=t, y=qseq, z=matrix(D$z, 50, 512), xlim=r,
                         ylim=R, xlab=Xl, ylab=Yl)
}


## End(Not run)

##########################################################
### Example - Simulated data from Frechet distirbution ###
##########################################################

if(interactive()){

set.seed(999)
data <- extraDistr::rfrechet(n=1500, mu=3, sigma=1, lambda=1/3)

u <- quantile(data, probs=0.9, type=3)
fit3 <- fGEV(data=data, par.start=c(1,2,1), method="Bayesian",
             u=u, sig0=1, nsim=5e+4)

pU <- mean(data>u)
P <- 1/c(750,1500,3000)
q3 <- ExtQ(P=P, method="Bayesian", pU=pU,
           param_post=fit3$param_post[-c(1:3e+4),])

### Illustration

# Tail index estimation

ti_true <- 3
ti_ps <- fit3$param_post[-c(1:3e+4),3]

K_ti <- density(ti_ps) # KDE of the tail index
H_ti <- hist(ti_ps, prob=TRUE, col="lightgrey",
       ylim=range(K_ti$y), main="", xlab="Tail Index",
       cex.lab=1.8, cex.axis=1.8, lwd=2)
ti_ic <- quantile(ti_ps, probs=c(0.025, 0.975))

points(x=ti_ic, y=c(0,0), pch=4, lwd=4)
lines(K_ti, lwd = 2, col = "dimgrey")
abline(v=ti_true, lwd=2)
abline(v=mean(ti_ps), lwd=2, lty=2)

# Quantile estimation

q3_true <- extraDistr::qfrechet(p=P, mu=3, sigma=1, lambda=1/3, lower.tail=FALSE)

ci <- apply(log(q3), 2, function(x) quantile(x, probs=c(0.025, 0.975)))
K_q3 <- apply(log(q3), 2, density)

R <- range(log(c(q3_true, q3, data)))
```

```
Xlim <- c(log(quantile(data, 0.95)), R[2])
Ylim <- c(0, max(K_q3[[1]]$y, K_q3[[2]]$y, K_q3[[3]]$y))

plot(0, main="", xlim=Xlim, ylim=Ylim, xlab=expression(log(x)),
     ylab="Density", cex.lab=1.8, cex.axis=1.8, lwd=2)
cval <- c(211, 169, 105)
for(j in 1:length(P)){
  col <- rgb(cval[j], cval[j], cval[j], 0.8*255, maxColorValue=255)
  col2 <- rgb(cval[j], cval[j], cval[j],  255, maxColorValue=255)
  polygon(K_q3[[j]], col=col, border=col2, lwd=4)
}
points(log(data), rep(0,n), pch=16)
# add posterior means
abline(v=apply(log(q3),2,mean), lwd=2, col=2:4)
# add credible intervals
abline(v=ci[1,], lwd=2, lty=3, col=2:4)
abline(v=ci[2,], lwd=2, lty=3, col=2:4)

}
```

`fExtDep` *Extremal dependence estimation*

Description

This function estimates the parameters of extremal dependence models.

Usage

```
fExtDep(method="PPP", data, model, par.start = NULL,
        c = 0, optim.method = "BFGS", trace = 0, sig = 3,
        Nsim, Nbin = 0, Hpar, MCpar, seed = NULL)
```

Arguments

`method`	A character string indicating the estimation method inlcuding `"PPP"`, `"BayesianPPP"` and `"Composite"`.
`data`	A matrix containing the data.
`model`	A character string with the name of the model. When `method="PPP"` or `"BayesianPPP"`, this includes `"PB"`, `"HR"`, `"ET"`, `"EST"`, `TD` and `AL` whereas when `method="composite"` it is restricted to `"HR"`, `"ET"` and `"EST"`.
`par.start`	A vector representing the initial parameters values for the optimization algorithm.
`c`	A real value in $[0, 1]$ required when `method="PPP"` or `"BayesianPPP"` and `model="ET"`, `"EST"` and `"AL"`. See `dExtDep` for more details.
`optim.method`	A character string indicating the optimization algorithm. Required when `method="PPP"` or `"Composite"`. See `optim` for more details.

trace — A non-negative integer, tracing the progress of the optimization. Required when method="PPP" or "Composite". See optim for more details.

sig — An integer indicating the number of significant digits when reporting outputs.

Nsim — An integer indicating the number of MCMC simulations. Required when method="BayesianPPP".

Nbin — An integer indicating the length of the burn-in period. Required when method="BayesianPPP".

Hpar — A list of hyper-parameters. See 'details'. Required when method="BayesianPPP".

MCpar — A positive real representing the variance of the proposal distirbution. See 'details'. Required when method="BayesianPPP".

seed — An integer indicating the seed to be set for reproducibility, via the routine set.seed.

Details

When method="PPP" the approximate likelihood is used to estimate the model parameters. It relies on the dExtDep function with argument method="Parametric" and angular=TRUE.

When method="BayesianPPP" a Bayesian estimation procedure of the spatral measure is considered, following Sabourin et al. (2013) and Sabourin & Naveau (2014). The argument Hpar is required to specify the hyper-parameters of the prior distributions, taking the following into consideration:

- 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 are independent, log-normally distributed. 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 are independent, log-normally distributed. 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 are independent, logit-squared for rho and log-normal for mu. 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 **Extremal skewt-t model**, the parameters are independent, logit-squared for rho, normal for alpha and log-normal for mu. The first elements correspond to the correlation parameters rho, then the skewness parameters alpha and the last parameter is the global dependence parameter mu. The list of hyper-parameters should be of the form mean.rho=, mean.alpha=, mean.mu=, sd.rho=, sd.alpha=, sd.mu=;
- For the **Asymmetric Logistic model**, the parameters' components are independent, log-normal for alpha and logit for beta. The list of hyper-parameters should be of the form mean.alpha=, mean.beta=, sd.alpha=, sd.beta=.

The proposal distribution for each (transformed) parameter is a normal distribution centred on the (transformed) current parameter value, with variance MCpar.

When method="Composite", the pairwise composite likelihood is applied, based on the dExtDep function with argument method="Parametric" and angular=FALSE.

Value

When `method` == `"PPP"` or `"Composite"`, a list is returned including

par: The estimated parameters.

LL: The maximised log-likelihood.

SE: The standard errors.

TIC: The Takeuchi Information Criterion.

When `method` == `"BayesianPPP"`, a list is returned including

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.

arguments: The specifics of the algorithm.

elapsed: The time elapsed, as given by `proc.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>, https://faculty.unibocconi.it/simonepadoan/;
Boris Beranger, <borisberanger@gmail.com> https://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**.

Sabourin, A., Naveau, P. and Fougeres, A-L (2013) Bayesian model averaging for multivariate extremes *Extremes*, **16**, 325-350.

Sabourin, A. and Naveau, P. (2014) Bayesian Dirichlet mixture model for multivariate extremes: A re-parametrization *Computational Statistics & Data Analysis*, **71**, 542-567.

See Also

`dExtDep`, `pExtDep`, `rExtDep`, `fExtDep.np`

Examples

```
# Example using the Poisson Point Proce Process appraoch
data(pollution)

  f.hr <- fExtDep(method="PPP", data=PNS, model="HR",
                  par.start = rep(0.5, 3), trace=2)


# Example using the pairwise composite (full) likelihood

  set.seed(1)
  data <- rExtDep(n=300, model="ET", par=c(0.6,3))
  f.et <- fExtDep(method="Composite", data=data, model="ET",
                  par.start = c(0.5, 1), trace=2)
```

`fExtDep.np` *Non-parametric extremal dependence estimation*

Description

This function estimates the bivariate extremal dependence structure using a non-parametric approach based on Bernstein Polynomials.

Usage

```
fExtDep.np(method, data, cov1=NULL, cov2=NULL, u, mar.fit=TRUE,
            mar.prelim=TRUE, par10, par20, sig10, sig20, param0=NULL,
            k0=NULL, pm0=NULL, prior.k="nbinom", prior.pm="unif",
            nk=70, lik=TRUE,
            hyperparam = list(mu.nbinom=3.2, var.nbinom=4.48),
            nsim, warn=FALSE, type="rawdata")
```

Arguments

`method`	A character string indicating the estimation method inlcuding `"Bayesian"`, `"Frequentist"` and `"Empirical"`.
`data`	A matrix containing the data.
`cov1, cov2`	A covariate vector/matrix for linear model on the location parameter of the marginal distributions. `length(cov1)/nrow(cov1)` needs to match `nrow(data)`. If `NULL` it is assumed to be constant. Required when `method="Bayesian"`.
`u`	When `method="Bayesian"`: a bivariate indicating the threshold for the exceedance approach. If logical `TRUE` the threshold are calculated by default as the 90% quantiles. When missing, a block maxima approach is considered. When `method="Frequentist"`: the associated quantile is used to select observations with the largest radial components. If missing, the 90% quantile is used.

- `mar.fit` A logical value indicated whether the marginal distributions should be fitted. When `method="Frequentist"`, data are empirically transformed to unit Frechet scale if `mar.fit=TRUE`. Not required when `method="Empirical"`.
- `rawdata` A character string specifying if the data is `"rawdata"` or `"maxima"`. Required when `method="Frequentist"`.
- `mar.prelim` A logical value indicated whether a preliminary fit of marginal distributions should be done prior to estimating the margins and dependence. Required when `method="Bayesian"`.
- `par10, par20` Vectors of starting values for the marginal parameter estimation. Required when `method="Bayesian"` and `mar.fit=TRUE`
- `sig10, sig20` Positive reals representing the initial value for the scaling parameter of the multivariate normal proposal distribution for both margins. Required when `method="Bayesian"` and `mar.fit=TRUE`
- `param0` A vector of initial value for the Bernstein polynomial coefficients. It should be a list with elements `$eta` and `$beta` which can be generated by the internal function `rcoef` if missing. Required when `method="Bayesian"`.
- `k0` An integer indicating the initial value of the polynomial order. Required when `method="Bayesian"`.
- `pm0` A list of initial values for the probability masses at the boundaries of the simplex. It should be a list with two elements `$p0` and `$p1`. Randomly generated if missing. Required when `method="Bayesian"`.
- `prior.k` A character string indicating the prior distribution on the polynomial order. By default `prior.k="nbinom"` (negative binomial) but can also be `"pois"` (Poisson). Required when `method="Bayesian"`.
- `prior.pm` A character string indicating the prior on the probability masses at the endpoints of the simplex. By default `prior.pm="unif"` (uniform) but can also be `"beta"` (beta). Required when `method="Bayesian"`.
- `nk` An integer indicating the maximum polynomial order. Required when `method="Bayesian"`.
- `lik` A logical value; if `FALSE`, an approximation of the likelihood, by means of the angular measure in Bernstein form is used (`TRUE` by default). Required when `method="Bayesian"`.
- `hyperparam` A list of the hyper-parameters depending on the choice of `prior.k` and `prior.pm`. See **details**. Required when `method="Bayesian"`.
- `nsim` An integer indicating the number of iterations in the Metropolis-Hastings algorithm. Required when `method="Bayesian"`.
- `warn` A logical value. If `TRUE` warnings are printed when some specifics (e.g., `param0`, `k0`, `pm0` and `hyperparam`) are not provided and set by default. Required when `method="Bayesian"`.
- `type` A character string indicating whther the data are `"rawdata"` or `"maxima"`. Required when `method="Bayesian"`.

Details

When `method="Bayesian"`, the vector of hyper-parameters is provided in the argument `hyperparam`. It should include:

If `prior.pm="unif"` requires `hyperparam$a.unif` and `hyperparam$b.unif`.

If `prior.pm="beta"` requires `hyperparam$a.beta` and `hyperparam$b.beta`.

If `prior.k="pois"` requires `hyperparam$mu.pois`.

If `prior.k="nbinom"` requires `hyperparam$mu.nbinom` and `hyperparam$var.nbinom` or `hyperparam$pnb` and `hyperparam$rnb`. The relationship is `pnb = mu.nbinom/var.nbinom` and `rnb = mu.nbinom^2 / (var.nbinom-mu.nbinom)`.

When `u` is specified Algorithm 1 of Beranger et al. (2021) is applied whereas when it is not specified Algorithm 3.5 of Marcon et al. (2016) is considered.

When `method="Frequentist"`, if `type="rawdata"` then pseudo-polar coordinates are extracted and only observations with a radial component above some high threshold (the quantile equivalent of `u` for the raw data) are retained. The inferential approach proposed in Marcon et al. (2017) based on the approximate likelihood is applied.

When `method="Empirical"`, the empirical estimation procedure presented in Einmahl et al. (2013) is applied.

Value

Outputs take the form of list including:

method: The argument.

type: whether it is `"maxima"` or `"rawdata"` (in the broader sense that a threshold exceedance model was taken).

If `method="Bayesian"` the list also includes:

mar.fit: The argument.

pm: The posterior sample of probability masses.

eta: The posterior sample for the coeficients of the Bernstein polynomial.

k: The posterior sample for the Bernstein polynoial order.

accepted: A binary vector indicating if the proposal was accepted.

acc.vec: A vector containing the acceptance probabilities for the dependence parameters at each iteration.

prior: A list containing `hyperparam`, `prior.pm` and `prior.k`.

nsim: The argument.

data: The argument.

In addition if the marginal parameters are estimated (`mar.fit=TRUE`):

cov1, cov2: The arguments.

accepted.mar, accepted.mar2: Binary vectors indicating if the marginal proposals were accepted.

straight.reject1, straight.reject2: Binary vectors indicating if the marginal proposals were rejected straight away as not respecting existence conditions (proposal is multivariate normal).

acc.vec1, acc.vec2: Vectors containing the acceptance probabilities for the marginal parameters at each iteration.

sig1.vec, sig2.vec: Vectors containing the values of the scaling parameter in the marginal proposal distributions.

Finally, if the argument `u` is provided, the list also contains:

threshold: A bivariate vector indicating the threshold level for both margins.

kn: The empirical estimate of the probability of being greater than the threshold.

When `method="Frequentist"`, the list includes:

When `method="Empirical"`, the list includes:

hhat: An estimate of the angular density.

Hhat: An estimate of the angular measure.

p0, p1: The estimates of the probability mass at 0 and 1.

Ahat: An estimate of the PIckands dependence function.

w: A sequence of value on the bivariate unit simplex.

q: A real in $[0, 1]$ indicating the quantile associated with the threshold `u`. Takes value `NULL` if `type="maxima"`.

data: The data on the unit Frechet scale (empirical transformation) if `type="rawdata"` and `mar.fit=TRUE`. Data on the original scale otherwise.

Author(s)

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

References

Beranger, B., Padoan, S. A. and Sisson, S. A. (2021). Estimation and uncertainty quantification for extreme quantile regions. *Extremes*, **24**, 349-375.

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

Marcon, G., Padoan, S. A. and Antoniano-Villalobos, I. (2016). Bayesian inference for the extremal dependence. *Electronic Journal of Statistics*, **10**, 3310-3337.

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

`dExtDep`, `pExtDep`, `rExtDep`, `fExtDep`

Examples

```
# Example Bayesian estimation,
# Threshold exceedances approach, threshold set by default
# Joint estimation margins + dependence
# Default uniform prior on pm
# Default negative binomial prior on polynomial order
# Quadratic relationship between location and max temperature

## Not run:
  data(MilanPollution)
  data <- Milan.winter[,c("NO2", "SO2", "MaxTemp")]
  data <- data[complete.cases(data),]

  covar <- cbind(rep(1,nrow(data)), data[,3], data[,3]^2)
  hyperparam <- list(mu.binom=6, var.binom=8, a.unif=0, b.unif=0.2)
  pollut <- fExtDep.np(method="Bayesian", data = data[,-3], u=TRUE,
                       cov1 = covar, cov2 = covar, mar.prelim=FALSE,
                       par10 = c(100,0,0,35,1), par20 = c(20,0,0,20,1),
                       sig10 = 0.1, sig20 = 0.1, k0 = 5,
                       hyperparam = hyperparam, nsim = 5e+4)
  # Warning: This is slow!

## End(Not run)

# Example Frequentist estimation
# Data are maxima

data(WindSpeedGust)
years <- format(ParcayMeslay$time, format="%Y")
attach(ParcayMeslay[which(years %in% c(2004:2013)),])

WS_th <- quantile(WS,.9)
DP_th <- quantile(DP,.9)

pars.WS <- evd::fpot(WS, WS_th, model="pp")$estimate
pars.DP <- evd::fpot(DP, DP_th, model="pp")$estimate

data_uf <- trans2UFrechet(cbind(WS,DP), type="Empirical")

rdata <- rowSums(data_uf)
r0 <- quantile(rdata, probs=.90)
extdata <- data_uf[rdata>=r0,]

SP_mle <- fExtDep.np(method="Frequentist", data=extdata, k0=10,
                     type="maxima")
```

---

fExtDepSpat *Fitting of a max-stable process*

Description

This function uses the Stephenson-Tawn likelihood to estimate parameters of max-stable models.

Usage

```
fExtDepSpat(model, z, sites, hit, jw, thresh, DoF, range, smooth,
            alpha, par0, acov1, acov2, parallel, ncores, args1, args2,
            seed=123, method = "BFGS", sandwich=TRUE,
            control = list(trace=1, maxit=50, REPORT=1, reltol=0.0001))
```

Arguments

`model`	A character string indicating the max-stable model, currently extremal-t (`"ET"`) and extremal skew-t (`"EST"`) only available. Note that the Schlather model can be obtained as a special case by specifying the extremal-t model with `DoF=1`
`z`	A $(n \times d)$ matrix containing n observations at d locations.
`sites`	A $(d \times 2)$ matrix corresponding to the coordinates of locations where the processes is simulated. Each row corresponds to a location.
`hit`	A $(n \times d)$ matrix containing the hitting scenarios for each observations.
`jw`	An integer between 2 and d, indicating the tuples considered in the composite likelihood. If `jw=d` the full likelihood is considered.
`thresh`	A positive real indicating the threshold value for pairwise distances. See `details`.
`DoF`	A positive real indicating a fixed value of the degree of freedom of the extremal-t and extremal skew-t models.
`range`	A positive real indicating a fixed value of the range parameter for the power exponential correlation function (only correlation function currently available).
`smooth`	A positive real in $(0, 2]$ indicating a fixed value of the smoothness parameter for the power exponential correlation function (only correlation function currently available).
`alpha`	A vector of length 3 indicating fixed values of the skewness parameters α_0, α_1 and α_2 for the extremal skew-t model. If some components are `NA`, then the corresponding parameter will be estimated.
`par0`	A vector of initial value of the parameter vector, in order the degree of freedom ν, the range r, the smoothness η and the skewness parameters α_0, α_1. Its length depends on the model and the number of fixed parameters.
`acov1, acov2`	Vectors of length d representing covariates to model the skewness parameter of the extremal skew-t model.
`parallel`	A logical value; if `TRUE` parallel computing is done.
`ncores`	An integer indicating the number of cores considered in the parallel socket cluster of type `'PSOCK'`, based on the `makeCluster` routine from the `parallel` package. Required if `parallel=TRUE`.
`args1, args2`	Lists specifying details about the Monte Carlo simulation schereme to compute multivariate CDFs. See `details`.
`seed`	An integer for reproduciblity in the CDF computations.

method	A character string indicating the optimisation method to be used. See optim for more details.
sandwich	A logical value; if TRUE the standard errors of the estimates are computed from the Sandwich (Godambe) information matrix.
control	A list of control parameter for the optimisation. See optim for more details.

Details

This routine follows the methodology developped by Beranger et al. (2021). It uses on the Stephenson-Tawn which relies on the knowledge of time occurrences of each block maxima. Rather than considering all partitions of the set $\{1,\ldots,d\}$, the likelihood is computed using the observed partition. Let $\Pi = (\pi_1,\ldots,\pi_K)$ denote the observed partition, then the Stephenson-Tawn likelihood is given by

$$L(\theta; z) = \exp\{-V(z;\theta)\} \times \prod_{k=1}^{K} -V_{\pi_k}(z;\theta),$$

where V_π represents the partial derivative(s) of $V(z;\theta)$ with respect to π.

When jw=d the full Stephenson-Tawn likelihood is considered whereas for values lower than d a composite likelihood approach is taken.

The argument thresh is required when the composite likelihood is used. A tuple of size jw, is assigned a weight of one if the maximum pairwise distance between corresponding locations is less that thresh and a weight of zero otherwise.

Arguments args1 and args2 relate to specifications of the Monte Carlo simulation scheme to compute multivariate CDF evaluations. These should take the form of lists including the minimum and maximum number of simulations used (Nmin and Nmax), the absolute error (eps) and whether the error should be controlled on the log-scale (logeps).

Value

A list comprising of the vector of estimated parameters (est), the composite likelihood order (jw), the maximised log-likelihood value (LL). In addition, if sandwich=TRUE the the standard errors from the sandwich information matrix are reported via stderr.sand as well as the TIC for model selection (TIC). Finally, if the composite likelihood is considered, a matrix with all tuples considered with a weight of 1 are reported in cmat.

Author(s)

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

References

Beranger, B., Stephenson, A. G. and Sisson, S.A. (2021) High-dimensional inference using the extremal skew-t process *Extremes*, **24**, 653-685.

See Also

fExtDepSpat

Examples

```
set.seed(14342)

# Simulation of 20 locations
Ns <- 20
sites <- matrix(runif(Ns*2)*10-5,nrow=Ns,ncol=2)
for(i in 1:2) sites[,i] <- sites[,i] - mean(sites[,i])

# Simulation of 50 replicates from the Extremal-t model
Ny <- 50
z <- rExtDepSpat(Ny, sites, model="ET", cov.mod="powexp", DoF=1,
                 range=3, nugget=0, smooth=1.5,
                 control=list(method="exact"))

# Fit the extremal-t using the full Stephenson-Tawn likelihood
args1 <- list(Nmax=50L, Nmin=5L, eps=0.001, logeps=FALSE)
args2 <- list(Nmax=500L, Nmin=50L, eps=0.001, logeps=TRUE)
## Not run:
fit1 <- fExtDepSpat(model="ET", z=z$vals, sites=sites, hit=z$hits,
                    par0=c(3,1,1), parallel=TRUE, ncores=6,
                    args1=args1, args2=args2, control = list(trace=0))
fit1$est

## End(Not run)
```

---

fGEV *Fitting of the Generalized Extreme Value Distribution*

Description

Maximum-likelihood and Metropolis-Hastings algorithm for the estimation of the generalized extreme value distribution.

Usage

```
fGEV(data, par.start, method="Frequentist", u, cov,
     optim.method="BFGS", optim.trace=0, sig0, nsim)
```

Arguments

`data`	A vector representing the data, which may contain missing values.
`par.start`	A vector of length 3 giving the starting values for the parameters to be estimated. If missing, moment estimates are used.
`method`	A character string indicating whether the estimation is done following a `"Frequentist"` or `"Bayesian"` approach.
`u`	A real indicating a high threshold. If supplied a threshld exceedance approach is taken and computations use the censored likelihood. If missing, a block maxima approach is taken and the regular GEV likelihood is used.

`cov` A matrix of covariates to define a linear model for the location parameter.

`optim.method` The optimization method to be used. Required when `method="Frequentist"`. See `optim` for more details.

`optim.trace` A non-negative interger tracing the progress of the optimization. Required when `method="Frequentist"`. See `optim` for more details.

`sig0` Positive reals representing the initial value for the scaling parameter of the multivariate normal proposal distribution for both margins. Required when `method="Bayesian"`.

`nsim` An integer indicating the number of iterations in the Metropolis-Hastings algorithm. Required when `method="Bayesian"`.

Details

When `cov` is a vector of ones then the location parameter μ is constant. On the contrary, when `cov` is provided, it represents the design matrix for the linear model on μ (the number of columns in the matrix indicates the number of linear predictors). When `u=NULL` or missing, the likelihood function is given by

$$\prod_{i=1}^{n} g(x_i; \mu, \sigma, \xi)$$

where $g(\cdot; \mu, \sigma, \xi)$ represents the GEV pdf, whereas when a threshold value is set the likelihood is given by

$$k_n \log \left(G(u; \mu, \sigma, \xi) \right) \times \prod_{i=1}^{n} \frac{\partial}{\partial x} G(x; \mu, \sigma, \xi)|_{x=x_i}$$

where $G(\cdot; \mu, \sigma, \xi)$ is the GEV cdf and k_n is the empirical estimate of the probability of being greater than the threshold `u`.

Note that the case $\xi <= 0$ is not yet considered when `u` is considered.

The choice `method="Bayesian"` applies a random walk Metropolis-Hastings algorithm as described in Section 3.1 and Step 1 and 2 of Algorithm 1 from Beranger et al. (2021). The algorithm may restart for several reasons including if the proposed value of the parameters changes too much from the current value (see Garthwaite et al. (2016) for more details.)

The choice `method="Frequentist"` uses the `optim` function to find the maximum likelihood estimator.

Value

When `method="Frequentist"` the routine returns a list including the parameter estimates (`est`) and associated variance-covariance matrix (`varcov`) and standard errors (`stderr`). When `method="Bayesian"` the routine returns a list including

param_post: the parameter posterior sample;

accepted: a binary vector indicating which proposal was accepted;

straight.reject: a binary vector indicating which proposal were rejected straight away given that the proposal is a multivariate normal and there are conditions on the parameter values;

nsim: the number of simulations in the algorithm;

sig.vec: the vector of updated scaling parameter in the multivariate normal porposal distribution at each iteration;

sig.restart: the value of the scaling parameter in the multivariate normal porposal distribution when the algorithm needs to restart;

acc.vec: a vector of acceptance probabilities at each iteration.

Author(s)

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

References

Beranger, B., Padoan, S. A. and Sisson, S. A. (2021). Estimation and uncertainty quantification for extreme quantile regions. *Extremes*, **24**, 349-375.

Garthwaite, P. H., Fan, Y. and Sisson S. A. (2016). Adaptive optimal scaling of Metropolis-Hastings algorithms using the Robbins-Monro process. *Communications in Statistics - Theory and Methods*, **45**(17), 5098-5111.

See Also

dGEV

Examples

```
###################################################
### Example - Pollution levels in Milan, Italy ###
###################################################

data(MilanPollution)

# Frequentist estimation
fit <- fGEV(Milan.winter$PM10)
fit$est

# Bayesian estimation with high threshold
cov <- cbind(rep(1,nrow(Milan.winter)), Milan.winter$MaxTemp,
             Milan.winter$MaxTemp^2)
u <- quantile(Milan.winter$PM10, prob=0.9, type=3, na.rm=TRUE)

  fit2 <- fGEV(data=Milan.winter$PM10, par.start=c(50,0,0,20,1),
               method="Bayesian", u=u, cov=cov, sig0=0.1, nsim=5e+4)
```

---

heat *Summer temperature maxima in Melbourne, Australia between 1961 and 2010.*

Description

The dataset corresponds to the summer maxima taken over the period from August to April inclusive, recorded between 1961 and 2010 at 90 stations on a 0.15 degree grid in a 9 by 10 formation.

Details

The first maximum is taken over the August 1961 to April 1962 period, and the last maximum is taken over the August 2010 to April 2011 period. The object `heatdata` contains the core of the data:

vals: A $50 * 90$ matrix containing the 50 summer maxima at the 90 locations.

sitesLL: A $90 * 2$ matrix containing the sites locations in Latitude-Longitude, recentred (means have been substracted).

sitesEN: A $90 * 2$ matrix containing the sites locations in Eastings-Northings, recentred (means have been substracted).

vals: A $50 * 90$ matrix containing integers indicating the "heatwave" number of each of the 50 summer maxima at all 90 locations. Locations on the same row with the same integer indicates that they were obtained from the same heatwave. Heatwaves are defined over a three day window.

sitesLLO: A $90 * 2$ matrix containing the sites locations in Latitude-Longitude, on the original scale.

sitesENO: A $90 * 2$ matrix containing the sites locations in Eastings-Northings, on the original scale.

ufvals: A $50 * 90$ matrix containing the 50 summer maxima at the 90 locations, on the unit Frechet scale.

Standardisation to unit Frechet is performed as in Beranger et al. (2021) by fitting the GEV distribution marginally using unconstrained location and shape parameters and the shape parameter to be a linear function of eastings and northings in 100 kilometre units. The resulting estimates are given in the objects `locgrid`, `scalegrid` and `shapegrid`, which are $10 * 9$ matrices.

Details about the study region are given in `mellat` and `mellon`, vectors of length 10 and 11 which give the latitude and longitude coordinates of the grid.

References

Beranger, B., Stephenson, A. G. and Sisson, S.A. (2021) High-dimensional inference using the extremal skew-t process *Extremes*, **24**, 653-685.

Examples

```
image(x=mellon, y=mellat, z=locgrid)
points(heatdata$sitesLLO, pch=16)
```

`index.ExtDep` *Index of extremal dependence*

Description

This function computes the extremal coefficient, Pickands dependence function and the coefficients of upper and lower tail dependence for several parametric models and the lower tail dependence function for the bivairate skew-normal distribution.

Usage

```
index.ExtDep(object, model, par, x, u)
```

Arguments

`object`	A character string indicating the index of extremal dependence to compute, including the extremal coefficient `"extremal"`, the Pickands dependence function `"pickands"`, the coefficient of upper tail dependence `"upper.tail"` and the coefficient of lower tail dependence `"lower.tail"`.
`model`	A character string indicating the model/distribution. When `object="extremal"`, `"pickands"` or `"upper.tail"` corresponding quantities can be calculated for the Husler-Reiss (`"HR"`), extremal-t (`"ET"`) and extremal skew-t (`"EST"`) are available. When `object="lower.tail"` then the extremal-t (`"ET"`) and extremal skew-t (`"EST"`) models are available as well as the skew-normal distribution (`"SN"`).
`par`	A vector indicating the parameter values of the corresponding model/distribution.
`x`	A vector on the bivariate or trivariate unit simplex indicating where to evaluate the Pickands dependence function.
`u`	A real in $[0,1]$ indicating the value at which to evaluate the lower tail dependence function of the bivariate skew-normal distribution.

Details

The extremal coefficient is defined as

$$\theta = V(1,\dots,1) = d\int_W \max_{j\in\{1,\dots,d\}}(w_j)dH(w) = -\log G(1,\dots,1),$$

where W represents the unit simplex, V is the exponent function and $G(\cdot)$ the distribution function of a multivariate extreme value model. Bivariate and trivariate versions are available.

The Pickands dependence function is defined as $A(x) = -\log G(1/x)$ for x in the bivariate/trivariate simplex (W).

The coefficient of upper tail dependence is defined as

$$\vartheta = R(1,\dots,1) = d\int_W \min_{j\in\{1,\dots,d\}}(w_j)dH(w).$$

In the bivariate case, using the inclusion-exclusion principle this reduces to $\vartheta = 2 + \log G(1,1) = 2 - V(1,1)$.

For the skew-normal distribution, the lower tail dependence function is defined as in Bortot (2010). This is an approximation where the tail dependence is obtained in the limiting case where u goes to 1. The par argument should be a vector of length 3 comprising of the correlation parameter, between -1 and 1 and two real-valued skewness parameters.

Value

When object="extremal", returns a value between 1 and d ($d = 2, 3$).
When object="pickands", returns a value between $\max(x)$ and 1.
When object="upper.tail", returns a value between 0 and 1.
When object="lower.tail", returns a value between -1 and 1.

Author(s)

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

References

Bortot, P. (2010) Tail dependence in bivariate skew-normal and skew-t distributions. *Unpublished.*

Examples

```
##############################
### Extremal skew-t model ###
##############################

### Extremal coefficient
index.ExtDep(object="extremal", model="EST", par=c(0.5,1,-2,2))

### Pickands dependence function
w <- seq(0.00001, .99999, length=100)
pick <- vector(length=100)
for(i in 1:100){
  pick[i] <- index.ExtDep(object="pickands", model="EST", par=c(0.5,1,-2,2),
                          x=c(w[i],1-w[i]))
}

plot(w, pick, type="l", ylim=c(0.5, 1), ylab="A(t)", xlab="t")
polygon(c(0, 0.5, 1), c(1, 0.5, 1), lwd=2, border = 'grey')

### Upper tail dependence coefficient
index.ExtDep(object="upper.tail", model="EST", par=c(0.5,1,-2,2))

### Lower tail dependence coefficient
index.ExtDep(object="lower.tail", model="EST", par=c(0.5,1,-2,2))

#################################
### Skew-normal distribution ###
```

```
################################

### Lower tail dependence function
index.ExtDep(object="lower.tail", model="SN", par=c(0.5,1,-2), u=0.5)
```

`logReturns` *Monthly maxima of log-return exchange rates of the Pound Sterling (GBP) against the US dollar (USD) and the Japanese yen (JPY), between March 1991 and December 2014.*

Description

The dataset `logReturns` contains 4 columns: `date_USD` and `USD` give the date of the monthly maxima of the log-return exchange rate GBP/USD and its value while `date_JPY` and `JPY` give the date of the monthly maxima of the log-return exchange rate GBP/JPY and its value.

Format

A $286 * 4$ matrix. The first and third columns are objects of type `"character"` while the second and fourth columns are of type `"numeric"`.

`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>, https://faculty.unibocconi.it/simonepadoan/; Boris Beranger, <borisberanger@gmail.com> https://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 <- 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.

PAMfmado *Clustering of maxima*

Description

Clustering of times series of maxima based on the pam package tailored for the F-madogram distance

Usage

```
PAMfmado(x, K, J = 0, threshold = 0.99, max.min = 0)
```

Arguments

x	x a matrix of maxima. For example, number of stations = ncol(x) and time series length = nrow(x) for weekly maxima of precipitation.
K	number of clusters
J	number of resampling for which the stations are randomly moved to break the dependence. By default, `J=0` means no resampling.
threshold	Threshold corresponding to the quantile level for the resampling. The resulting quantile is printed (when `J` does not take value `0`).
max.min	A threshold to remove very small values. For example, some raingauges cannot go below 2 mm. By default, `max.min=0`.

Value

an object of class "pam" representing the clustering. See ?pam.object for details.

Author(s)

Philippe Naveau

References

Bernard E., Naveau P., Vrac M. and Mestre O. (2013). Clustering of maxima: Spatial dependencies among heavy rainfall in France. *Journal of Climate 26*, 7929–7937.

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

Cooley, D., P. Naveau, and P. Poncet (2006). Variograms for spatial max-stable random fields. In: Bertail, P., Soulier, P., Doukhan, P. (eds) *Dependence in Probability and Statistics. Lecture Notes in Statistics*, vol 187. Springer, New York, NY .

Reynolds, A., Richards, G., de la Iglesia, B. and Rayward-Smith, V. (1992). Clustering rules: A comparison of partitioning and hierarchical clustering algorithms. *Journal of Mathematical Modelling and Algorithms 5*, 475-504.

See Also

See the function as pam in the package cluster

Examples

```
data(PrecipFrance)
attach(PrecipFrance)
PAMmado <- PAMfmado(precip,7)
```

---

pExtDep *Parametric and non-parametric distribution function of Extremal Dependence*

Description

This function evaluates the distribution function of parametric multivariate extreme distributions and the angular probability distribution represented through Bernstein polynomials.

Usage

```
pExtDep(q, type, method="Parametric", model, par, plot=TRUE,
        main, xlab, cex.lab, cex.axis, lwd,...)
```

Arguments

`q`	A vector or matrix of quantiles.
`type`	A character string taking value `"lower"`, `"inv.lower"` or `"upper"`. Required when `method="Parametric"`.
`method`	A character string taking value `"Parametric"` or `"NonParametric"`
`model`	A character string with the name of the model: `"HR"` (Husler-Reiss), `"ET"` (Extremal-t) or `"EST"` (Extremal Skew-t). Required when `method="Parametric"`.
`par`	A vector or a matrix representing the parameters of the (parametric or non-parametric) model. When in matrix format, rows indicate different sets of parameter values.
`plot`	A logical value; if `TRUE` (default) a plot is displayed. See **details**.
`main`, `xlab`, `cex.lab`, `cex.axis`, `lwd`	Arguments of the `hist()` function.
`...`	Additional graphical parameter when `plot=TRUE`.

Details

Note that when `method="Parametric"`, the distribution function is only available in 2 and 3 dimensions. Refer to the `dim_ExtDep` function for the appropriate length of the parameter vector.
When `type="lower"`, the cumulative distribution function is computed, i.e.,

$$G(\boldsymbol{x}) = P\left(\boldsymbol{X} \leq \boldsymbol{x}\right), x \in R^d, d = 2, 3.$$

When `type="inv.lower"`, the survival function is computed, i.e.,

$$1 - G(\boldsymbol{x}) = 1 - P\left(\boldsymbol{X} \leq \boldsymbol{x}\right).$$

This corresponds to the probability of at least one component of X is greater than its corresponding element in x.
When `type="upper"`, the joint probability of exceedance is computed, i.e.,

$$P\left(\boldsymbol{X} \geq \boldsymbol{x}\right).$$

Finally, when `method="NonParametric"`, the distribution function is only available in 2 dimensions.

The argument `plot` is only applicable when `par` is a matrix. Typically its main use should be when `par` corresponds to some posterior sample (e.g. from `fExtDep` with `moethod="BayesianPPP"`). A histogram of the probabilities evaluated at each set of parameters is displayed, as well as a kernel density estimator, $2.5\%, 50\%, 97.5\%$ quantiles (crosses) and mean (dot). The argument `...` is used to specify additional parameters in the `hist()` function.

Value

When `par` is a vector: if `q` is a matrix the function returns a vector of length `nrow(q)`, otherwise a scalar.
When `par` is a matrix: if `q` is a matrix the function returns a matrix with `nrow(par)` rows and `nrow(q)` columns, otherwise a vector of length `nrow(par)`.

Author(s)

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

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

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

dExtDep, rExtDep, fExtDep, fExtDep.np

Examples

```
# Example using the trivariate Extremal Skew-t
pExtDep(q=c(1,1.2, 0.6), type="lower", method="Parametric",
        model="EST", par=c(0.2, 0.4, 0.6,2,2,2,1))

# Example using the bivariate Extremal-t
pExtDep(q=rbind(c(1.2, 0.6), c(1.1, 1.3)), type="inv.lower",
        method="Parametric", model="ET", par=c(0.2, 1))
pExtDep(q=rbind(c(1.2, 0.6), c(1.1, 1.3)), type="inv.lower",
        method="Parametric", model="EST", par=c(0.2, 0, 0, 1))

# Example of non-parametric angular density
beta <- c(1.0000000, 0.8714286, 0.7671560, 0.7569398,
          0.7771908, 0.8031573, 0.8857143, 1.0000000)
pExtDep(q=rbind(c(0.1,0.9),c(0.2,0.8)), method="NonParametric", par=beta)
```

---

pFailure *Probability of falling into a failure region*

Description

This function computes the empirical estimate of the probability of falling into two types of failure regions.

Usage

```
pFailure(n, beta, u1, u2, mar1, mar2, type, plot, xlab, ylab,
          nlevels=10)
```

Arguments

`n`	An integer indicating the number of points generated to compute the empirical probability.
`beta`	A vector representing the Bernstein polynomial coefficients.
`u1`, `u2`	Vectors of positive reals representing some high thresholds.
`mar1`, `mar2`	Vectors of marginal (GEV) parameters
`type`	A character string indicating if the failure region includes at least one exceedance (`"or"`), or both exceednaces (`"and"`). If `"both"` then probabilities to fall in both regions are computed.
`plot`	A logical value; if `TRUE` contour plots of the probalities are displayed.
`xlab`, `ylab`	A character string equivalent representing the graphical parameters as in `par`.
`nlevels`	The number of contour levels to be displayed.

Details

The two failure regions are:

$$A_u = \{(v_1, v_2) : v_1 > u_1 \text{ or } v_2 > u_2\}$$

and

$$B_u = \{(v_1, v_2) : v_1 > u_1 \text{ and } v_2 > u_2\}$$

where $(v_1, v_2) \in R_+^2$ and $u_1, u_2 > 0$.

Exceedances samples $y_{1,1}, \dots, y_{n_1}$ and $y_{1,2}, \dots, y_{n_2}$ are generating according to Algorithm 3 of Marcon et al. (2017) and the the estimates of the probability of falling in A_u and B_u are given by

$$\hat{P}_{A_u} = \frac{1}{n} \sum i = 1^n I \{y_{i,1} > u_1^* \text{ or } y_{i,2} > u_2^*\}$$

and

$$\hat{P}_{B_u} = \frac{1}{n} \sum i = 1^n I \{y_{i,1} > u_1^* \text{ and } y_{i,2} > u_2^*\}$$

Value

A list containing `AND` and/or `OR`, depending on the `type` argument. Each element is a `length(u1) x length(u2)` matrix.

Author(s)

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

References

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

See Also

dExtDep, rExtDep, fExtDep, fExtDep.np

Examples

```
# Example wind speed and wind gust

data(WindSpeedGust)
years <- format(ParcayMeslay$time, format="%Y")
attach(ParcayMeslay[which(years %in% c(2004:2013)),])

WS_th <- quantile(WS,.9)
DP_th <- quantile(DP,.9)

pars.WS <- evd::fpot(WS, WS_th, model="pp")$estimate
pars.DP <- evd::fpot(DP, DP_th, model="pp")$estimate

data_uf <- trans2UFrechet(cbind(WS,DP), type="Empirical")

rdata <- rowSums(data_uf)
r0 <- quantile(rdata, probs=.90)
extdata <- data_uf[rdata>=r0,]

SP_mle <- fExtDep.np(method="Frequentist", data=extdata, k0=10,
                     type="maxima")


  pF <- pFailure(n=50000, beta=SP_mle$Ahat$beta,
                u1=seq(from=19, to=28, length=200), mar1=pars.WS,
                u2=seq(from=40, to=60, length=200), mar2=pars.DP,
                type="both", plot=TRUE,
                xlab="Daily-maximum Wind Speed (m/s)",
                ylab="Differential of Pressure (mbar)", nlevels=15)
```

`plot_ExtDep` *Graphical summaries of parametric representations of extremal dependence.*

Description

This function displays the angular density, Pickands dependence function and return levels for bivariate and trivariate extreme values models.

Usage

```
plot_ExtDep(object="angular", model, par, log=TRUE, data=NULL, contour=TRUE,
            style, labels, cex.dat=1, cex.lab=1, cex.cont=1,
            Q.fix, Q.range, Q.range0, cond=FALSE, ...)
```

Arguments

`object`	A character string indicating which graphical summary to plot. Takes value `"angular"` (default) `"pickands"` or`"returns"`.
`model`	A string with the name of the model considered. Takes value `"PB"` (Pairwise Beta), `"HR"` (Husler-Reiss), `"ET"` (Extremal-t), `"EST"` (Extremal Skew-t), `TD` (Tilted Dirichlet) or `AL` (Asymmetric Logistic) when evaluating the angular density. Restricted to `"HR"`, `"ET"` and `"EST"` for the Pickands dependence function.
`par`	A vector representing the parameters of the model.
`log`	A logical value specifying if the log density is computed. Required when `object="angular"`.
`data`	A matrix representing angular data to be added to the density plot. Required when `object="angular"`.
`contour`	A logical value; if `TRUE` the contour levels are displayed. Required for trivariate models only.
`style`	A character string indicating the plotting style of the data. Takes value `"hist"` or `"ticks"`. See **details**.
`labels`	A vector of character strings indicating the labels. Must be of length 1 for bivariate models and 3 for trivariate models.
`cex.dat`	A positive real indicating the size of the data points. Required for the trivariate angular density.
`cex.lab`	A positive real indicating the size of the labels.
`cex.cont`	A positive real indicating the size of the contour labels.
`Q.fix`	A vector of length the dimension of the model, indicating some fixed quantiles to compute joint return levels. Must contain `NA` (maximum 2) for components whose quantiles are allowed to vary. Required when `object="returns"`.
`Q.range`	A vector or matrix indicating quantile values on the unit Frechet scale, for the components that are allowed to vary. Must be a vector or a one-column matrix if there is one `NA` in `Q.fix`. Must be a two-column matrix if there are two `NA`s in `Q.fix`. Required when `object="returns"`.
`Q.range0`	A object of the same format as `Q.range`, corresponding to the quantiles on the original scale. Required when `object="returns"`.
`cond`	A logical value; if `TRUE` then conditional return levels are computed where the conditional event is the fixed event. Default if `FALSE`. Required when `object="returns"`.
`...`	Additional graphical arguments for the `hist()` and `plot()` functions respectively used to compute the bivariate angular density and Pickands dependence function as well as for the `plot()` and `contour()` functionswhen `object="returns"`.

Details

The angular density is computed using the function dExtDep with arguments method="Parametric" and angular=TRUE. The Pickands dependence function is computed using the function index.ExtDep with argument object="pickands".

When displaying the bivariate angular density and some data are provided (a 2-column matrix is specified for data), there is the choice to summarise the data using a histogram (style="hist") or to display the observations using tick marks (style="ticks").

When displaying return levels, there are two possibilities: univariate and bivariate return levels. Since the model dimensions are restricted to a maximum of three, in that case, aunivariate return level corresponds to fixing two components while a bivariate return level fixes only one component. The choice of the fixed component is decided by the position of the NA value(s) in the Q.fix argument. If par is a vector then the corresponding return level(s) are printed. However if par is a matrix then the return level(s) are evaluated for each value of the parameter vector and the mean, and empirical 95% empirical interval are displayed. Typically this is used when posterior samples are available. When par is a matrix with only two rows, resulting plots may not provide much information.

When contours are displayed, levels are chosen to be the deciles.

Value

A graph depending on argument object.

Author(s)

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

See Also

dExtDep, index.ExtDep.

Examples

```
data(pollution)

################################
### Trivariate Husler-Reiss ###
################################


  f.hr <- fExtDep(method="PPP", data=PNS, model="HR", par.start=rep(1,3))

  plot_ExtDep(object="angular", model="HR", par=f.hr$par, data=PNS,
              labels=c(expression(PM[10]), expression(NO), expression(SO[2])),
              cex.lab=2)

  plot_ExtDep(object="pickands", model="HR", par=f.hr$par, data=PNS,
              labels=c(expression(PM[10]), expression(NO), expression(SO[2])),
              cex.lab=2) # Takes time!
```

```
################################
### Bivariate Husler-Reiss ###
################################


  PN <- na.omit(Leeds.frechet[,1:2])
  PN <- cbind(PN, rowSums(PN))
  PN <- PN[order(PN[,3], decreasing = TRUE)[1:100],]
  PN <- PN[,1:2]/PN[,3]

  f.hr2 <- fExtDep(method="PPP", data=PN, model = "HR", par.start = 1)
  plot_ExtDep(model="HR", par=f.hr2$par, log=FALSE, data=PN, style="hist")
  plot_ExtDep(model="HR", par=f.hr2$par, log=FALSE, data=PN, style="ticks")
  plot_ExtDep(object="pickands", model="HR", par=f.hr2$par)
```

`plot_ExtDep.np` *Graphical summaries of non-parametric representations of extremal dependence.*

Description

This function displays several summaries of extremal dependence represented through Bernstein polynomials.

Usage

```
plot_ExtDep.np(out, type, summary.mcmc, burn, y, probs,
                 A_true, h_true, est.out, mar1, mar2, dep,
                 QatCov1=NULL, QatCov2=QatCov1, P,
                 labels=c(expression(y[1]),expression(y[2])),
                 CEX=1.5, xlim, ylim, col.data,
                 col.Qfull, col.Qfade, data=NULL, ...)
```

Arguments

`out`	An output of the `fExtDep.np` function.
`type`	A character string indicating the type of graphical summary to be plotted. Takes values `"summary"`, `"returns"`, `"A"`, `"h"`, `"pm"`, `"k"` or `"Qsets"`.
`summary.mcmc`	The output of the `summary_ExtDep` function. Only required when `out` is obtained using a Bayesian estimation method (`out$est=="Bayesian"`).
`burn`	The burn-in period. Only required when `out` is obtained using a Bayesian estimation method (`out$est=="Bayesian"`).

y A 2-column matrix of unobserved thresholds at which the returns are calculated. Required when type="returns".

probs The probability of joint exceedances, the output of the return function.

A_true A vector representing the true pickands dependence function evaluated at the grid points on the simplex given by summary.mcmc$w.

h_true A vector representing the true angular density function evaluated at the grid points on the simplex given by summary.mcmc$w.

est.out A list containing:

ghat: a 3-row matrix giving the posterior summary for the estimate of the bound;

Shat and Shat_post: a matrix of the posterior and a 3-row matrix giving the posterior summary for the estimate of the basic set S;

nuShat and nuShat_post: a matrix of the posterior and a 3-row matrix giving the posterior summary for the estimate of the measure $\nu(S)$;

Note that a posterior summary is made of its mean and 90%credibility interval.

Only required when using a Bayesian estimation method (out$est=="Bayesian") and type="Qsets".

mar1, mar2 Vectors of marginal GEV parameters. Required when type="Qsets" and either out$method=="Bayesian" if the marginal parameter weren't fitted or "empirical".

dep A logical value; if TRUE the estimate of the dependence is plotted when computing extreme quantile regions (type="Qsets").

QatCov1, QatCov2

Matrices representing the value of the covariates at which extreme quantile regions should be computed. Required when type="Qsets". See arguments cov1 and cov2 from fExtDep.

P A vector indicating the probabilities associated with the quantiles to be computed. Required when type="Qsets".

labels A bivariate vector of character strings providing labels for extreme quantile regions. Required when type="Qsets".

CEX Label and axis sizes.

xlim, ylim Limits of the x and y axis when computing extreme quantile regions. Required when type="Qsets".

col.data, col.Qfull, col.Qfade

Colors for data, estimate of extreme quantile regions and its credible interval (when applicable). Required when type="Qsets".

data A 2-column matrix providing the original data to be plotted when type="Qsets".

... Additional graphical parameters

Details

If type="returns", a (contour) plot of the probabilities of exceedances for some threshold is returned. This corresponds to the output of the returns function.
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"`, when the estimation was performed in a Bayesian framework then a 2 by 2 plot with types `"A"`, `"h"`, `"pm"` and `"k"` is returned. Otherwise a 1 by 2 plot with types `"A"` and `"h"` is returned. If `type="Qsets"`, extreme quantile regions are computed according to the methodology developped in Beranger et al. (2021).

Value

a graph depending on argument `type`.

Author(s)

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

References

Beranger, B., Padoan, S. A. and Sisson, S. A. (2021). Estimation and uncertainty quantification for extreme quantile regions. *Extremes*, **24**, 349-375.

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

`fExtDep.np`.

Examples

```
###########################################################
### Example 1 - Wind Speed and Differential of pressure ###
###########################################################

data(WindSpeedGust)

years <- format(ParcayMeslay$time, format="%Y")
attach(ParcayMeslay[which(years %in% c(2004:2013)),])

# Marginal quantiles
WS_th <- quantile(WS,.9)
DP_th <- quantile(DP,.9)

# Standardisation to unit Frechet (requires evd package)
pars.WS <- evd::fpot(WS, WS_th, model="pp")$estimate
pars.DP <- evd::fpot(DP, DP_th, model="pp")$estimate

# transform the marginal distribution to common unit Frechet:
```

```
data_uf <- trans2UFrechet(cbind(WS,DP), type="Empirical")

# compute exceedances
rdata <- rowSums(data_uf)
r0 <- quantile(rdata, probs=.90)
extdata_WSDP <- data_uf[rdata>=r0,]

# Fit
SP_mle <- fExtDep.np(method="Frequentist", data=extdata_WSDP, k0=10, type="maxima")

# Plot
plot_ExtDep.np(out=SP_mle, type="summary")

####################################################
### Example 2 - Pollution levels in Milan, Italy ###
####################################################

## Not run:

### Here we will only model the dependence structure
data(MilanPollution)

data <- Milan.winter[,c("NO2","SO2")]
data <- as.matrix(data[complete.cases(data),])

# Thereshold
u <- apply(data, 2, function(x) quantile(x, prob=0.9, type=3))

# Hyperparameters
hyperparam <- list(mu.nbinom = 6, var.nbinom = 8, a.unif=0, b.unif=0.2)

### Standardise data to univariate Frechet margins

f1 <- fGEV(data=data[,1], method="Bayesian", sig0 = 0.0001, nsim = 5e+4)
diagnostics(f1)
burn1 <- 1:30000
gev.pars1 <- apply(f1$param_post[-burn1,],2,mean)
sdata1 <- trans2UFrechet(data=data[,1], pars=gev.pars1, type="GEV")

f2 <- fGEV(data=data[,2], method="Bayesian", sig0 = 0.0001, nsim = 5e+4)
diagnostics(f2)
burn2 <- 1:30000
gev.pars2 <- apply(f2$param_post[-burn2,],2,mean)
sdata2 <- trans2UFrechet(data=data[,2], pars=gev.pars2, type="GEV")

sdata <- cbind(sdata1,sdata2)

### Bayesian estimation using Bernstein polynomials

pollut1 <- fExtDep.np(method="Bayesian", data=sdata, u=TRUE,
                      mar.fit=FALSE, k0=5, hyperparam = hyperparam, nsim=5e+4)

diagnostics(pollut1)
```

```
pollut1_sum <- summary_ExtDep(mcmc=pollut1, burn=3e+4, plot=TRUE)
pl1 <- plot_ExtDep.np(out=pollut1, type="Qsets", summary.mcmc=pollut1_sum,
                      mar1=gev.pars1, mar2=gev.pars2, P = 1/c(600, 1200, 2400),
                      dep=TRUE, data=data, xlim=c(0,400), ylim=c(0,400))

pl1b <- plot_ExtDep.np(out=pollut1, type="Qsets", summary.mcmc=pollut1_sum, est.out=pl1$est.out,
                       mar1=gev.pars1, mar2=gev.pars2, P = 1/c(1200),
                       dep=FALSE, data=data, xlim=c(0,400), ylim=c(0,400))

### Frequentist estimation using Bernstein polynomials

pollut2 <- fExtDep.np(method="Frequentist", data=sdata, mar.fit=FALSE, type="rawdata", k0=8)
plot_ExtDep.np(out=pollut2, type = c("summary"), CEX=1.5)

pl2 <- plot_ExtDep.np(out=pollut2, type="Qsets", mar1=gev.pars1, mar2=gev.pars2,
                      P = 1/c(600, 1200, 2400),
                      dep=TRUE, data=data, xlim=c(0,400), ylim=c(0,400),
                      labels=c(expression(NO[2]),expression(SO[2])),
                      col.Qfull = c("red", "green", "blue"))

### Frequentist estimation using EKdH estimator

pollut3 <- fExtDep.np(method="Empirical", data=data)
plot_ExtDep.np(out=pollut3, type = c("summary"), CEX=1.5)

pl3 <- plot_ExtDep.np(out=pollut3, type="Qsets", mar1=gev.pars1, mar2=gev.pars2,
                      P = 1/c(600, 1200, 2400),
                      dep=TRUE, data=data, xlim=c(0,400), ylim=c(0,400),
                      labels=c(expression(NO[2]),expression(SO[2])),
                      col.Qfull = c("red", "green", "blue"))


## End(Not run)
```

`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 `NA`s. 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

https://uk-air.defra.gov.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

PrecipFrance *Weekly maxima of hourly rainfall in France*

Description

List containing the weekly maxima of hourly rainfall in the Fall season from 1993 to 2011 recorded at 92 stations across France (precip). Coordinates of the monitoring stations are given in lat and lon.

Format

A list made of a $228 * 92$ matrix (precip) and two vectors of length 228 (lat and lon).

Details

The fall season corresponds to the September-November (SON) period. The data thus cover a 12-week period over 19 years, yielding a sample of $n = 228$ observations (rows) and $p = 92$ stations (columns).

returns *Compute return values*

Description

Predicts the probability of future simultaneous exceedances

Usage

```
returns(out, summary.mcmc, y, plot=FALSE, labels=NULL,
        data=NULL)
```

Arguments

out	The output of the fExtDep.np function with method="Bayesian".
summary.mcmc	The output of the summary_ExtDep function.
y	A 2-column matrix of unobserved thresholds.
plot	A logical value; if TRUE, then the returns are plotted using plot_ExtDep.np.
labels	As in plot_ExtDep.np. Required if plot=TRUE.
data	As in plot_ExtDep.np. Required if plot=TRUE.

Details

Computes for a range of unobserved extremes (larger than those observed in a sample), the point-wise 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>, https://faculty.unibocconi.it/simonepadoan/; Boris Beranger, <borisberanger@gmail.com> https://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**, 3310-3337.

Examples

```
########################################################
### Example 1 - daily log-returns between the GBP/USD ###
###             and GBP/JPY exchange rates           ###
########################################################

if(interactive()){

data(logReturns)

mm_gbp_usd <- ts(logReturns$USD, start=c(1991,3), end=c(2014,12), frequency=12)
mm_gbp_jpy <- ts(logReturns$JPY, start=c(1991,3), end=c(2014,12), frequency=12)

### Detect seasonality and trend in the time series of maxima:
seas_usd <- stl(mm_gbp_usd, s.window="period")
seas_jpy <- stl(mm_gbp_jpy, s.window="period")

### remove the seasonality and trend from the two series:
mm_gbp_usd_filt <- mm_gbp_usd - rowSums(seas_usd$time.series[,-3])
mm_gbp_jpy_filt <- mm_gbp_jpy - rowSums(seas_jpy$time.series[,-3])

### Estimation of margins and dependence

mm_gbp <- cbind(as.vector(mm_gbp_usd_filt), as.vector(mm_gbp_jpy_filt))

hyperparam <- list(mu.nbinom = 3.2, var.nbinom = 4.48)
gbp_mar <- fExtDep.np(method="Bayesian", data=mm_gbp, par10=rep(0.1, 3),
                      par20=rep(0.1,3), sig10=0.0001, sig20=0.0001, k0=5,
                      hyperparam = hyperparam, nsim=5e+4)

gbp_mar_sum <- summary_ExtDep(mcmc=gbp_mar, burn=3e+4, plot=TRUE)

mm_gbp_range <- apply(mm_gbp,2,quantile,c(0.9,0.995))

y_gbp_usd <- seq(from=mm_gbp_range[1,1], to=mm_gbp_range[2,1], length=20)
y_gbp_jpy <- seq(from=mm_gbp_range[1,2], to=mm_gbp_range[2,2], length=20)
y <- as.matrix(expand.grid(y_gbp_usd, y_gbp_jpy, KEEP.OUT.ATTRS = FALSE))

ret_marg <- returns(out=gbp_mar, summary.mcmc=gbp_mar_sum, y=y, plot=TRUE,
                 data=mm_gbp, labels=c("GBP/USD exchange rate", "GBP/JPY exchange rate"))

}

########################################################
### Example 2 - Reproducing some of the results shown ###
###             in Marcon et al. (2016, Figure 1)     ###
########################################################

## Not run:

set.seed(1890)
data <- evd::rbvevd(n=100, dep=0.6, asy=c(0.8,0.3), model="alog", mar1=c(1,1,1))
```

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

mcmc <- fExtDep.np(method="Bayesian", data=data, mar.fit=FALSE, k0=5,
                   pm0=pm0, prior.k = "nbinom", prior.pm = "unif",
                   hyperparam=hyperparam, nsim=5e+5)

w <- seq(0.001, 0.999, length=100)
summary.mcmc <- summary_ExtDep(w, mcmc, burn=4e+5, plot=TRUE)

plot_ExtDep.np(out=mcmc, type = "A", summary.mcmc=summary.mcmc)
plot_ExtDep.np(out=mcmc, type = "h", summary.mcmc=summary.mcmc)
plot_ExtDep.np(out=mcmc, type = "pm", summary.mcmc=summary.mcmc)
plot_ExtDep.np(out=mcmc, type = "k", summary.mcmc=summary.mcmc)

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


## End(Not run)
```

`rExtDep` *Parametric and semi-parametric random generator of extreme events*

Description

This function generates random samples of iid observations from extremal dependence models and semi-parametric stochastic generators.

Usage

```
rExtDep(n, model, par, angular=FALSE, mar=c(1,1,1), num, threshold,
        exceed.type)
```

Arguments

`n`	An integer indictaing the number of observations.
`model`	A character string with the name of the model. Parametric model include `"HR"` (Husler-Reiss), `"ET"` (Extremal-t), `"EST"` (Extremal Skew-t). Semi-parametric generators include `"semi.bvevd"` and `"semi.bvexceed"`.
`par`	A vector representing the parameters of the (parametric or non-parametric) model.
`angular`	A logical value; `TRUE` for angular outputs.
`mar`	A vector or matrix of marginal parameters.
`num`	An integer indicating the number of observations the componentwise maxima is computed over. Required when `model="HR"`, `"ET"` or `"EST"`. Set to `5e+5` unless specified.

`threshold` A bivariate vector indicating the level of exceedances. Required when `model="semi.bvexceed"`.

`exceed.type` A character string taking value "and" or "or" indicating the type of exceednaces. Required when `model="semi.bvexceed"`.

Details

There is no limit of the dimensionality when `model="HR"`, `"ET"` or `"EST"` while `model="semi.bvevd"` and `"semi.bvexceed"` can only generate bivariate observations. When `angular=TRUE` and `model="semi.bvevd"` or `"semi.bvexceed"` the simulation of pseudo-angles follows Algorithm 1 of Marcon et al. (2017). When `model="semi.bvevd"` and `angular=FALSE`, maxima samples are generated according to Algorithm 2 of Marcon et al. (2017). When `model="semi.bvexceed"` and `angular=FALSE`, exceedance samples are generated above value specified by `threshold`, according to Algorithm 3 of Marcon et al. (2017). `exceed.type="and"` generates samples that exceed both thresholds while `exceed.type="or"` generates samples exceeding at least one threshold.

When the argument `mar` is a vector, the marginal distrutions are identical. When a matrix is provided each row corresponds to a set of marginal parameters. No marginal transformation is applied when `mar=c(1,1,1)`.

Value

A matrix with n rows and $p >= 2$ columns. $p = 2$ when `model="semi.bvevd"` or `"semi.bvexceed"`.

Author(s)

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

References

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

See Also

`dExtDep`, `pExtDep`, `fExtDep`, `fExtDep.np`

Examples

```
# Example using the trivariate Husler-Reiss
set.seed(1)
data <- rExtDep(n=10, model="HR", par=c(2,3,3))

# Example using the semi-parammetric generator of maxima
set.seed(2)
beta <- c(1.0000000, 0.8714286, 0.7671560, 0.7569398,
          0.7771908, 0.8031573, 0.8857143, 1.0000000)
data <- rExtDep(n=10, model="semi.bvevd", par=beta,
                mar=rbind(c(0.2, 1.5, 0.6),c(-0.5, 0.4, 0.9)))
```

```
# Example using the semi-parammetric generator of maxima
set.seed(3)
data <- rExtDep(n=10, model="semi.bvexceed", par=beta,
                threshold=c(0.2, 0.4), exceed.type="and")
```

`rExtDepSpat` *Random generation of max-stable processes*

Description

This function generates realisations from a max-stable process.

Usage

```
rExtDepSpat(n, coord, model="SCH", cov.mod = "whitmat", grid = FALSE,
            control = list(), cholsky = TRUE, ...)
```

Arguments

`n`	An integer indictaing the number of observations.
`coord`	A vector or matrix corresponding to the coordinates of locations where the processes is simulated. Each row corresponds to a location.
`model`	A character string indicating the max-stable model. See `details`.
`cov.mod`	A character string indicating the correlation function function. See `details`.
`grid`	A logical value; `TRUE` for coordinates on a grid.
`control`	A named list with arguments `nlines` giving the number of lines of the TBM simulation, `method` a character string specifying the name of the simulation method and `uBound` the uniform upper bound. Note that `method` must take value `'exact'`, `'tbm'` or `'circ'`. See `details`.
`cholsky`	A logical value; if `TRUE` a Cholesky decomposition is performed. Used for the extremal-t and extremal skew-t models when `control=list(method='exact')`.
`...`	The parameters of the max-stable model. See `details`.

Details

This function extends the `rmaxstab` function from the `SpatialExtremes` package in two ways:

1. The **extremal skew-t** model is included.
2. The function returns the hitting scenarios, i.e. the index of which 'storm' (or process) led to the maximum value for each location and observation.

The max-stable models available in this procedure and the specifics are:

Smith model: when `model='SMI'`, does not require `cov.mod`. If `coord` is univariate then `var` needs to be specified and for higher dimensions covariance parameters should be provided such as `cov11`, `cov12`, `cov22`, etc.

Schlather model: when `model='SCH'`, requires `cov.mod='whitmat'`, `'cauchy'`, `'powexp'` or `'bessel'` depending on the correlation family. Parameters `'nugget'`, `'range'` and `'smooth'` should be specified.

Extremal-t model: when `model='ET'`, requires `cov.mod='whitmat'`, `'cauchy'`, `'powexp'` or `'bessel'` depending on the correlation family. Parameters `'nugget'`, `'range'`, `'smooth'` and `'DoF'` should be specified.

Extremal skew-t model: when `model='EST'`, requires `cov.mod='whitmat'`, `'cauchy'`, `'powexp'` or `'bessel'` depending on the correlation family. Parameters `'nugget'`, `'range'`, `'smooth'`, `'DoF'`, `'alpha'` (a vector of length 3) and `'acov1'` and `'acov2'` (both vector of length the number of locations) should be specified. The skewness vector is defined as $\alpha = \alpha_0 + \alpha_1 \mathrm{acov1} + \alpha_2 \mathrm{acov2}$.

Geometric gaussian model: when `model='GG'`, requires `cov.mod='whitmat'`, `'cauchy'`, `'powexp'` or `'bessel'` depending on the correlation family. Parameters `'sig2'`, `'nugget'`, `'range'` and `'smooth'` should be specified.

Brown-Resnick model: when `model='BR'`, does not require `cov.mod`. Parameters `'range'` and `'smooth'` should be specified.

For the argument `control`, details of the list components are as follows:

method is `NULL` by default, meaning that the function tries to find the most appropriate simulation technique. Current simulation techniques are a direct approach, i.e. Cholesky decomposition of the covariance matrix, the turning bands and the circular embedding methods. Note that for the extremal skew-t model it can only take value `'exact'` or `'direct'`;

nlines if `NULL` then it is set to 1000;

uBound if `NULL` then it is set to reasonable values - for example 3.5 for the Schlather model.

Value

A list made of

vals: A $(n \times d)$ matrix containing n observations at d locations, from the specified max-stable model.

hits: A $(n \times d)$ matrix containing the hitting scenarios for each observations. On each row, elements with the same integer value indicate that the maxima at these two locations is coming from the same 'storm' or process.

Author(s)

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

References

Beranger, B., Stephenson, A. G. and Sisson, S.A. (2021) High-dimensional inference using the extremal skew-t process *Extremes*, **24**, 653-685.

See Also

fExtDepSpat

Examples

```
# Generate some locations
set.seed(1)
lat <- lon <- seq(from=-5, to=5, length=20)
sites <- as.matrix(expand.grid(lat,lon))

# Example using the extremal-t
set.seed(2)
z <- rExtDepSpat(1, sites, model="ET", cov.mod="powexp", DoF=1,
                 nugget=0, range=3, smooth=1.5,
                 control=list(method="exact"))
fields::image.plot(lat, lon, matrix(z$vals,ncol=20) )

# Example using the extremal skew-t
set.seed(3)
z2 <- rExtDepSpat(1, sites, model="EST", cov.mod="powexp", DoF=5,
                  nugget=0, range=3, smooth=1.5, alpha=c(0,5,5),
                  acov1=sites[,1], acov2=sites[,2],
                  control=list(method="exact"))
fields::image.plot(lat, lon, matrix(z2$vals,ncol=20) )
```

---

simplex *Definition of a multivariate simplex*

Description

Generation of grid points over the multivariate simplex

Usage

```
simplex(d, n=50, a=0, b=1)
```

Arguments

d	A positive integer indicating the dimension of the simplex.
n	A positive integer indicating the number of grid points to be generated on the univariate components of the simplex.
a, b	Two numeric values indicating the lower and upper bound of the simplex. By default a=0 and b=0, indicating the unit-simplex.

Details

A d-dimensional simplex is defined by

$$S = \{(\omega_1, \dots, \omega_d) \in R_+^d : \sum_{i=1}^{d} \omega_i = 1\}.$$

Here the function defines the simplex as

$$S = \{(\omega_1, \dots, \omega_d) \in [a, b]^d : \sum_{i=1}^{d} \omega_i = 1\}.$$

When d=2 and $[a, b] = [0, 1]$, a grid of points of the form $\{(\omega_1, \omega_2) \in [0, 1] : \omega_1 + \omega_2 = 1\}$.

Value

Returns a matrix with d columns. When d=2, the number of rows is n. When d>2, the number of rows is equal to

$$\sum_{i_{d-1}=0}^{n-1} \sum_{i_{d-2}=0}^{n-i_{d-1}} \cdots \sum_{i_1=1}^{n-i_{d-1}-\cdots-i_2} i_1$$

Author(s)

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

Examples

```
### 3-dimensional unit simplex

W <- simplex(d=3, n=10)
plot(W[,-3], pch=16)
```

---

summary_ExtDep *Summary of MCMC algorithm.*

Description

This function computes summaries on the posterior sample obtained from the adaptive MCMC scheme for the non-parametric estimation of a bivariate dependence structure.

Usage

```
summary_ExtDep(object, mcmc, burn, cred=0.95, plot=FALSE, ...)
```

Arguments

`object`	A vector of values on $[0, 1]$. If missing, a regular grid of length 100 is considered.
`mcmc`	An output of the `fExtDep.np` function with `method="Bayesian"`.
`burn`	A positive integer indicating the burn-in period.
`cred`	A value in $[0, 1]$ indicating the level of the credibility intervals to be computed.
`plot`	A logical value; if `TRUE` a summary of the estimated dependence is displayed by calling the `plot_ExtDep.np` function with `type="summary"`.
`...`	Additional graphical parameters for `plot_ExtDep.np` when `plot=TRUE`.

Details

For each value say $\omega \in [0, 1]$ given, the complement $1 - \omega$ is automatically computed to define the observation $(\omega, 1 - \omega)$ on the bivariate unit simplex.

It is obvious that the value of `burn` must be greater than the number of iterations in the mcmc algorithm. This can be found in `mcmc`.

Value

The function returns a list with the following objects:

k.median, k.up, k.low: Posterior median, upper and lower bounds of the CI for the estimated Bernstein polynomial degree κ;

h.mean, h.up, h.low: Posterior mean, upper and lower bounds of the CI for the estimated angular density h;

A.mean, A.up, A.low: Posterior mean, upper and lower bounds of the CI for the estimated Pickands dependence function A;

p0.mean, p0.up, p0.low: Posterior mean, upper and lower bounds of the CI for the estimated point mass p_0;

p1.mean, p1.up, p1.low: Posterior mean, upper and lower bounds of the CI for the estimated point mass p_1;

A_post: Posterior sample for Pickands dependence function;

h_post: Posterior sample for angular density;

eta.diff_post: Posterior sample for the Bernstein polynomial coefficients (η parametrisation);

beta_post: Posterior sample for the Bernstein polynomial coefficients (β parametrisation);

p0_post, p1_post: Posterior sample for point masses p_0 and p_1;

w: A vector of values on the bivariate simplex where the angular density and Pickands dependence function were evaluated;

burn: The argument provided;

If the margins were also fitted, the list given as `object` would contain `mar1` and `mar2` and the function would also output:

mar1.mean, mar1.up, mar1.low: Posterior mean, upper and lower bounds of the CI for the estimated marginal parameter on the first component;

mar2.mean, mar2.up, mar2.low: Posterior mean, upper and lower bounds of the CI for the estimated marginal parameter on the second component;

mar1_post: Posterior sample for the estimated marginal parameter on the first component;

mar2_post: Posterior sample for the estimated marginal parameter on the second component;

Author(s)

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

See Also

fExtDep.np.

Examples

```
#####################################################
### Example - Pollution levels in Milan, Italy ###
#####################################################

## Not run:

### Here we will only model the dependence structure
data(MilanPollution)

data <- Milan.winter[,c("NO2","SO2")]
data <- as.matrix(data[complete.cases(data),])

# Thereshold
u <- apply(data, 2, function(x) quantile(x, prob=0.9, type=3))

# Hyperparameters
hyperparam <- list(mu.nbinom = 6, var.nbinom = 8, a.unif=0, b.unif=0.2)

### Standardise data to univariate Frechet margins

f1 <- fGEV(data=data[,1], method="Bayesian", sig0 = 0.0001, nsim = 5e+4)
diagnostics(f1)
burn1 <- 1:30000
gev.pars1 <- apply(f1$param_post[-burn1,],2,mean)
sdata1 <- trans2UFrechet(data=data[,1], pars=gev.pars1, type="GEV")

f2 <- fGEV(data=data[,2], method="Bayesian", sig0 = 0.0001, nsim = 5e+4)
diagnostics(f2)
burn2 <- 1:30000
gev.pars2 <- apply(f2$param_post[-burn2,],2,mean)
sdata2 <- trans2UFrechet(data=data[,2], pars=gev.pars2, type="GEV")

sdata <- cbind(sdata1,sdata2)

### Bayesian estimation using Bernstein polynomials
```

```
pollut1 <- fExtDep.np(method="Bayesian", data=sdata, u=TRUE,
                      mar.fit=FALSE, k0=5, hyperparam = hyperparam, nsim=5e+4)

diagnostics(pollut1)
pollut1_sum <- summary_ExtDep(mcmc=pollut1, burn=3e+4, plot=TRUE)

## End(Not run)
```

---

trans2GEV *Transformation to GEV distribution*

Description

Transformation of marginal distribution from unit Frechet to GEV

Usage

```
trans2GEV(data, pars)
```

Arguments

data	A vector of length n or a $(n \times p)$ matrix representing the data on its original scale.
pars	A (1×3) vector or a $(p \times 3)$ matrix of marginal GEV parameters.

Details

The transformation function is $\left(x^{\xi} - 1\right) \frac{\sigma}{\xi} + \mu$ if $\xi \neq 0$, and $x^{-1}\sigma + \mu$ if $\xi = 0$.

Value

An object of the same format and dimensions as `data`.

Author(s)

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

See Also

`trans2UFrechet`

Examples

```
data(pollution)
pars <- fGEV(Leeds.frechet[,1])$est

par_new <- c(2, 1.5, 0.5)
data_new <- trans2GEV(Leeds.frechet[,1], pars=par_new)

fGEV(data_new)
```

`trans2UFrechet` *Transformation to unit Frechet distribution*

Description

Empirical and parametric transformation of a dataset to unit Frechet marginal distribution

Usage

```
trans2UFrechet(data, pars, type="Empirical")
```

Arguments

`data`	A vector of length n or a $(n \times p)$ matrix representing the data on its original scale.
`pars`	A (1×3) vector or a $(p \times 3)$ matrix of marginal GEV parameters. Required when `type="GEV"`.
`type`	A character string indicating the type of transformation. Can take value `"Empirical"` or `"GEV"`.

Details

When `type="Empirical"`, the transformation function is $t(x) = -1/\log(F_{\text{emp}}(x))$ where $F_{\text{emp}}(x)$ denotes the empirical cumulative distribution.

When `type="GEV"`, the transformation function is $\left(1+\xi\frac{x-\mu}{\sigma}\right)^{1/\xi}$ if $\xi \neq 0$, $\left(\frac{x-\mu}{\sigma}\right)^{-1}$ if $\xi = 0$. If the argument `pars` is missing then a GEV is fitted on the columns of `data` using the `fGEV` function.

Value

An object of the same format and dimensions as `data`.

Author(s)

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

See Also

`trans2GEV`, `fGEV`

Examples

```
data(MilanPollution)

pars <- fGEV(Milan.winter$PM10)$est
pars

data_uf <- trans2UFrechet(data=Milan.winter$PM10, pars=pars,
                          type="GEV")
fGEV(data_uf)$est

data_uf2 <- trans2UFrechet(data=Milan.winter$PM10,
                           type="Empirical")
fGEV(data_uf2)$est
```

`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.

`WindSpeedGust` *Hourly wind gust, wind speed and air pressure at Lingen (GER), Ossendorf (GER) and Parcay-Meslay (FRA).*

Description

There are three objects of type `data.frame`, one for each location.

Details

Each object has the following columns:

WS: the hourly wind speed in metres per second (m/s);

WG: the hourly wind gust in metres per second (m/s);

DP: the hourly air pressure at sea level in millibars.

Specifics about each object is given below:

Lingen: is a `data.frame` object with 1083 rows and 4 columns. Measurements are recorded between January 1982 and June 2003;

Ossendorf: is a `data.frame` object with 676 rows and 4 columns. Measurements are recorded between March 1982 and August 1995;

ParcayMeslay: is a `data.frame` object with 2140 rows and 4 columns. Measurements are recorded between November 1984 and July 2013.

References

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

Index