Package ‘revdbayes’

January 11, 2026

Title Ratio-of-Uniforms Sampling for Bayesian Extreme Value Analysis

Version 1.5.6

Date 2026-01-11

Description Provides functions for the Bayesian analysis of extreme value models. The 'rust' package <https://cran.r-project.org/package=rust> is used to simulate a random sample from the required posterior distribution. The functionality of 'revdbayes' is similar to the 'evdbayes' package <https://cran.r-project.org/package=evdbayes>, which uses Markov Chain Monte Carlo ('MCMC') methods for posterior simulation. In addition, there are functions for making inferences about the extremal index, using the models for threshold inter-exceedance times of Suveges and Davison (2010) <doi:10.1214/09-AOAS292> and Holesovsky and Fusek (2020) <doi:10.1007/s10687-020-00374-3>. Also provided are d,p,q,r functions for the Generalised Extreme Value ('GEV') and Generalised Pareto ('GP') distributions that deal appropriately with cases where the shape parameter is very close to zero.

Imports bayesplot (>= 1.1.0), exdex, graphics, Rcpp, rust (>= 1.2.2), stats, utils

License GPL (>= 2)

LazyData TRUE

Encoding UTF-8

Depends R (>= 3.3.0),

RoxygenNote 7.2.3

Suggests ggplot2 (>= 2.2.1), knitr, microbenchmark, rmarkdown, testthat

VignetteBuilder knitr

URL https://paulnorthrop.github.io/revdbayes/, https://github.com/paulnorthrop/revdbayes

BugReports https://github.com/paulnorthrop/revdbayes/issues

LinkingTo Rcpp (>= 0.12.10), RcppArmadillo

Config/testthat/edition 3

NeedsCompilation yes

Author Paul J. Northrop [aut, cre, cph],
Scott D. Grimshaw [ctb]

Maintainer Paul J. Northrop <p.northrop@ucl.ac.uk>

Repository CRAN

Date/Publication 2026-01-11 16:50:02 UTC

Contents

`revdbayes-package` *revdbayes: Ratio-of-Uniforms Sampling for Bayesian Extreme Value Analysis*

Description

Uses the multivariate generalized ratio-of-uniforms method to simulate random samples from the posterior distributions commonly encountered in Bayesian extreme value analyses.

Details

The main functions in the revdbayes package are `rpost` and `rpost_rcpp`, which simulate random samples from the posterior distribution of extreme value model parameters using the functions `ru` and `ru_rcpp` from the rust package, respectively. The user chooses the extreme value model, the prior density for the parameters and provides the data. There are options to improve the probability of acceptance of the ratio-of-uniforms algorithm by working with transformation of the model parameters.

The functions `kgaps_post` and `dgaps_post` simulate from the posterior distribution of the extremal index θ based on the K-gaps model for threshold interexceedance times of Suveges and Davison (2010) and the similar D-gaps model of Holesovsky and Fusek (2020). See also Attalides (2015).

See `vignette("revdbayes-a-vignette", package = "revdbayes")` for an overview of the package and `vignette("revdbayes-b-using-rcpp-vignette", package = "revdbayes")` for an illustration of the improvements in efficiency produced using the Rcpp package. See `vignette("revdbayes-c-predictive-v`
`package = "revdbayes")` for an outline of how to use revdbayes to perform posterior predictive extreme value inference and `vignette("revdbayes-d-kgaps-vignette", package = "revdbayes")` considers Bayesian inference for the extremal index θ using threshold inter-exceedance times.

Author(s)

Maintainer: Paul J. Northrop <`p.northrop@ucl.ac.uk`> [copyright holder]

Other contributors:

- Scott D. Grimshaw [contributor]

References

Holesovsky, J. and Fusek, M. Estimation of the extremal index using censored distributions. Extremes 23, 197-213 (2020). doi:10.1007/s10687020003743

Northrop, P. J. (2016). rust: Ratio-of-Uniforms Simulation with Transformation. R package version 1.2.2. `https://cran.r-project.org/package=rust`.

Suveges, M. and Davison, A. C. (2010) Model misspecification in peaks over threshold analysis, *The Annals of Applied Statistics*, **4**(1), 203-221. doi:10.1214/09AOAS292

Attalides, N. (2015) Threshold-based extreme value modelling, PhD thesis, University College London. `https://discovery.ucl.ac.uk/1471121/1/Nicolas_Attalides_Thesis.pdf`

See Also

`set_prior` to set a prior density for extreme value parameters.

`rpost` and `rpost_rcpp` to perform ratio-of-uniforms sampling from an extreme value posterior distribution.

`kgaps_post` and `dgaps_post` to sample from a posterior distribution for the extremal index based on inter-exceedance times.

The `ru` and `ru_rcpp` functions in the `rust` package for details of the arguments that can be passed to `ru` via `rpost` and for the form of the object (of class `"evpost"`) returned from `rpost`, which has the same structure as an object (of class `"ru"`) returned by `ru` and `ru_rcpp`.

binpost *Random sampling from a binomial posterior distribution*

Description

Samples from the posterior distribution of the probability p of a binomial distribution.

Usage

```
binpost(n, prior, ds_bin, param = c("logit", "p"))
```

Arguments

`n` A numeric scalar. The size of posterior sample required.

`prior` A function to evaluate the prior, created by `set_bin_prior`.

`ds_bin` A numeric list. Sufficient statistics for inference about a binomial probability p. Contains

- `n_raw` : number of raw observations.
- `m` : number of threshold exceedances.

`param` A character scalar. Only relevant if `prior$prior` is a (user-supplied) R function. `param` specifies the parameterization of the posterior distribution that `ru` uses for sampling.

If `param = "p"` the original parameterization p is used.

If `param = "logit"` (the default) then `ru` samples from the posterior for the logit of p, before transforming back to the p-scale.

The latter tends to make the optimizations involved in the ratio-of-uniforms algorithm more stable and to increase the probability of acceptance, but at the expense of slower function evaluations.

Details

If `prior$prior == "bin_beta"` then the posterior for p is a beta distribution so `rbeta` is used to sample from the posterior.

If `prior$prior == "bin_mdi"` then rejection sampling is used to sample from the posterior with an envelope function equal to the density of a beta(`ds$m` + 1, `ds$n_raw` - `ds$m` + 1) density.

If `prior$prior == "bin_northrop"` then rejection sampling is used to sample from the posterior with an envelope function equal to the posterior density that results from using a Haldane prior.

If `prior$prior` is a (user-supplied) R function then `ru` is used to sample from the posterior using the generalised ratio-of-uniforms method.

Value

An object (list) of class `"binpost"` with components

`bin_sim_vals:` An `n` by 1 numeric matrix of values simulated from the posterior for the binomial probability p

`bin_logf:` A function returning the log-posterior for p.

`bin_logf_args:` A list of arguments to `bin_logf`.

If `prior$prior` is a (user-supplied) R function then this list also contains `ru_object` the object of class `"ru"` returned by `ru`.

See Also

`set_bin_prior` for setting a prior distribution for the binomial probability p.

Examples

```
u <- quantile(gom, probs = 0.65)
ds_bin <- list()
ds_bin$n_raw <- length(gom)
ds_bin$m <- sum(gom > u)
bp <- set_bin_prior(prior = "jeffreys")
temp <- binpost(n = 1000, prior = bp, ds_bin = ds_bin)
graphics::hist(temp$bin_sim_vals, prob = TRUE)

# Setting a beta prior (Jeffreys in this case) by hand
beta_prior_fn <- function(p, ab) {
  return(stats::dbeta(p, shape1 = ab[1], shape2 = ab[2], log = TRUE))
}
jeffreys <- set_bin_prior(beta_prior_fn, ab = c(1 / 2, 1 / 2))
temp <- binpost(n = 1000, prior = jeffreys, ds_bin = ds_bin)
```

create_prior_xptr *Create an external pointer to a C++ prior*

Description

This function provides an example of a way in which a user can specify their own prior density to rpost_rcpp. More specifically, a function like this (the user will need to create an edited version tailored to their own C++ function(s)) can be used to generate an external pointer to a compiled C++ function that evaluates the log-prior density. Please see the vignette "Faster simulation using revdbayes" for more information.

Usage

```
create_prior_xptr(fstr)
```

Arguments

fstr	A string indicating the C++ function required.

Details

Suppose that the user's C++ functions are in a file called "user_fns.cpp". These functions must be compiled and made available to R before the pointer is created. This can be achieved using the function sourceCpp in the **Rcpp** package or using RStudio's Source button on the editor toolbar.

For details see the examples in the documentation of the functions rpost_rcpp and set_prior, the vignette "Faster simulation using revdbayes" and the vignette "Rusting Faster: Simulation using Rcpp" in the package **rust**.

Value

An external pointer.

See Also

set_prior to specify a prior distribution using an external pointer returned by create_prior_xptr and for details of in-built named prior distributions.

The examples in the documentation of rpost_rcpp.

Examples

```
ptr_gp_flat <- create_prior_xptr("gp_flat")
prior_cfn <- set_prior(prior = ptr_gp_flat, model = "gp", min_xi = -1)

ptr_gev_flat <- create_prior_xptr("gev_flat")
prior_cfn <- set_prior(prior = ptr_gev_flat, model = "gev", min_xi = -1,
                       max_xi = Inf)

mat <- diag(c(10000, 10000, 100))
ptr_gev_norm <- create_prior_xptr("gev_norm")
pn_u <- set_prior(prior = ptr_gev_norm, model = "gev", mean = c(0,0,0),
                  icov = solve(mat))
```

---

dgaps_post — *Random sampling from D-gaps posterior distribution*

Description

Uses the `rust` package to simulate from the posterior distribution of the extremal index θ based on the D-gaps model for threshold interexceedance times of Holesovsky and Fusek (2020). We refer to this as the D-gaps model, because it uses a tuning parameter D, whereas the related K-gaps model of Suveges and Davison (2010) has a tuning parameter K.

Usage

```
dgaps_post(
  data,
  thresh,
  D = 1,
  n = 1000,
  inc_cens = TRUE,
  alpha = 1,
  beta = 1,
  param = c("logit", "theta"),
  use_rcpp = TRUE
)
```

Arguments

`data` A numeric vector or numeric matrix of raw data. If `data` is a matrix then the log-likelihood is constructed as the sum of (independent) contributions from different columns. A common situation is where each column relates to a different year.

If `data` contains missing values then `split_by_NAs` is used to divide the data further into sequences of non-missing values, stored in different columns in a matrix. Again, the log-likelihood is constructed as a sum of contributions from different columns.

thresh A numeric scalar. Extreme value threshold applied to data.

D A numeric scalar. The censoring parameter D, as defined in Holesovsky and Fusek (2020). Threshold inter-exceedances times that are not larger than D units are left-censored, occurring with probability $\log(1-\theta e^{-\theta d})$, where $d = qD$ and q is the probability with which the threshold u is exceeded.

n A numeric scalar. The size of posterior sample required.

inc_cens A logical scalar indicating whether or not to include contributions from right-censored inter-exceedance times, relating to the first and last observations. It is known that these times are greater than or equal to the time observed. If data has multiple columns then there will be right-censored first and last inter-exceedance times for each column. See also the **Details** section of dgaps.

alpha, beta Positive numeric scalars. Parameters of a beta(α, β) prior for θ.

param A character scalar. If param = "logit" (the default) then we simulate from the posterior distribution of $\phi = \log(\theta/(1-\theta))$ and then transform back to the θ-scale. If param = "theta" then we simulate directly from the posterior distribution of θ, unless the sample D-gaps are all equal to zero or all positive, when we revert to param = "logit". This is to avoid the possibility of sampling directly from a posterior with mode equal to 0 or 1.

use_rcpp A logical scalar. If TRUE (the default) the rust function ru_rcpp is used for posterior simulation. If FALSE the (slower) function ru is used.

Details

A beta(α, β) prior distribution is used for θ so that the posterior from which values are simulated is proportional to

$$\theta^{2N_1+\alpha-1}(1-\theta e^{-\theta d})^{N_0+\beta-1}\exp\{-\theta q(I_0T_0+\cdots+I_NT_N)\}.$$

See dgaps_stat for a description of the variables involved in the contribution of the likelihood to this expression.

The ru function in the rust package simulates from this posterior distribution using the generalised ratio-of-uniforms distribution. To improve the probability of acceptance, and to ensure that the simulation will work even in extreme cases where the posterior density of θ is unbounded as θ approaches 0 or 1, we simulate from the posterior distribution of $\phi = \log(\theta/(1-\theta))$ and then transform back to the θ-scale.

Value

An object (list) of class "evpost", which has the same structure as an object of class "ru" returned from ru. In addition this list contains

- call: The call to dgaps().
- model: The character scalar "dgaps".
- thresh: The argument thresh.
- ss: The sufficient statistics for the D-gaps likelihood, as calculated by dgaps_stat.

References

Holesovsky, J. and Fusek, M. Estimation of the extremal index using censored distributions. Extremes 23, 197-213 (2020). doi:10.1007/s10687020003743

Suveges, M. and Davison, A. C. (2010) Model misspecification in peaks over threshold analysis, *The Annals of Applied Statistics*, **4**(1), 203-221. doi:10.1214/09AOAS292

See Also

ru for the form of the object returned by dgaps_post.

kgaps_post for Bayesian inference about the extremal index θ using the K-gaps model.

Examples

```
# Newlyn sea surges

thresh <- quantile(newlyn, probs = 0.90)
d_postsim <- dgaps_post(newlyn, thresh)
plot(d_postsim)

### Cheeseboro wind gusts

d_postsim <- dgaps_post(exdex::cheeseboro, thresh = 45, D = 3)
plot(d_postsim)
```

---

gev *The Generalised Extreme Value Distribution*

Description

Density function, distribution function, quantile function and random generation for the generalised extreme value (GEV) distribution.

Usage

```
dgev(x, loc = 0, scale = 1, shape = 0, log = FALSE, m = 1)

pgev(q, loc = 0, scale = 1, shape = 0, lower.tail = TRUE, log.p = FALSE, m = 1)

qgev(p, loc = 0, scale = 1, shape = 0, lower.tail = TRUE, log.p = FALSE, m = 1)

rgev(n, loc = 0, scale = 1, shape = 0, m = 1)
```

Arguments

`x, q` Numeric vectors of quantiles.

`loc, scale, shape` Numeric vectors. Location, scale and shape parameters. All elements of `scale` must be positive.

`log, log.p` A logical scalar; if TRUE, probabilities p are given as log(p).

`m` A numeric scalar. The distribution is reparameterised by working with the GEV(`loc, scale, shape`) distribution function raised to the power `m`. See **Details**.

`lower.tail` A logical scalar. If TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

`p` A numeric vector of probabilities in [0,1].

`n` Numeric scalar. The number of observations to be simulated. If `length(n) > 1` then `length(n)` is taken to be the number required.

Details

The distribution function of a GEV distribution with parameters `loc` = μ, `scale` = $\sigma (> 0)$ and `shape` = ξ is

$$F(x) = \exp\{-[1 + \xi(x - \mu)/\sigma]^{-1/\xi}\}$$

for $1 + \xi(x - \mu)/\sigma > 0$. If $\xi = 0$ the distribution function is defined as the limit as ξ tends to zero. The support of the distribution depends on ξ: it is $x \leq \mu - \sigma/\xi$ for $\xi < 0$; $x \geq \mu - \sigma/\xi$ for $\xi > 0$; and x is unbounded for $\xi = 0$. Note that if $\xi < -1$ the GEV density function becomes infinite as x approaches $\mu - \sigma/\xi$ from below.

If `lower.tail = TRUE` then if `p = 0` (`p = 1`) then the lower (upper) limit of the distribution is returned, which is `-Inf` or `Inf` in some cases. Similarly, but reversed, if `lower.tail = FALSE`.

See https://en.wikipedia.org/wiki/Generalized_extreme_value_distribution for further information.

The effect of `m` is to change the location, scale and shape parameters to $(\mu + \sigma \log m, \sigma, \xi)$ if $\xi = 0$ and $(\mu + \sigma(m^\xi - 1)/\xi, \sigma m^\xi, \xi)$. For integer `m` we can think of this as working with the maximum of `m` independent copies of the original GEV(`loc, scale, shape`) variable.

Value

`dgev` gives the density function, `pgev` gives the distribution function, `qgev` gives the quantile function, and `rgev` generates random deviates.

The length of the result is determined by `n` for `rgev`, and is the maximum of the lengths of the numerical arguments for the other functions.

The numerical arguments other than `n` are recycled to the length of the result.

References

Jenkinson, A. F. (1955) The frequency distribution of the annual maximum (or minimum) of meteorological elements. *Quart. J. R. Met. Soc.*, **81**, 158-171. doi:10.1002/qj.49708134804

Coles, S. G. (2001) *An Introduction to Statistical Modeling of Extreme Values*, Springer-Verlag, London. Chapter 3: doi:10.1007/9781447136750_3

Examples

```
dgev(-1:4, 1, 0.5, 0.8)
dgev(1:6, 1, 0.5, -0.2, log = TRUE)
dgev(1, shape = c(-0.2, 0.4))

pgev(-1:4, 1, 0.5, 0.8)
pgev(1:6, 1, 0.5, -0.2)
pgev(1, c(1, 2), c(1, 2), c(-0.2, 0.4))
pgev(-3, c(1, 2), c(1, 2), c(-0.2, 0.4))
pgev(7, 1, 1, c(-0.2, 0.4))

qgev((1:9)/10, 2, 0.5, 0.8)
qgev(0.5, c(1,2), c(0.5, 1), c(-0.5, 0.5))

p <- (1:9)/10
pgev(qgev(p, 1, 2, 0.8), 1, 2, 0.8)

rgev(6, 1, 0.5, 0.8)
```

gev_beta — *Beta-type prior for GEV shape parameter ξ*

Description

For information about this and other priors see `set_prior`.

Usage

```
gev_beta(pars, min_xi = -1/2, max_xi = 1/2, pq = c(6, 9), trendsd = 0)
```

Arguments

`pars` A numeric vector of length 3. GEV parameters (μ, σ, ξ).

`min_xi` A numeric scalar. Prior lower bound on ξ.

`max_xi` A numeric scalar. Prior upper bound on ξ.

`pq` A numeric vector of length 2. See `set_prior` for details.

`trendsd` Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gev_flat *Flat prior for GEV parameters $(\mu, log\sigma, \xi)$*

Description

For information about this and other priors see set_prior.

Usage

```
gev_flat(pars, min_xi = -Inf, max_xi = Inf, trendsd = 0)
```

Arguments

pars	A numeric vector of length 3. GEV parameters (μ, σ, ξ).
min_xi	A numeric scalar. Prior lower bound on ξ. Must not be -Inf because this results in an improper posterior.
max_xi	A numeric scalar. Prior upper bound on ξ.
trendsd	Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gev_flatflat *Flat prior for GEV parameters (μ, σ, ξ)*

Description

For information about this and other priors see set_prior.

Usage

```
gev_flatflat(pars, min_xi = -Inf, max_xi = Inf, trendsd = 0)
```

Arguments

pars	A numeric vector of length 3. GEV parameters (μ, σ, ξ).
min_xi	A numeric scalar. Prior lower bound on ξ. Must not be -Inf because this results in an improper posterior.
max_xi	A numeric scalar. Prior upper bound on ξ.
trendsd	Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gev_loglognorm *Trivariate normal prior for GEV parameters ($log\mu, log\sigma, \xi$)*

Description

For information about this and other priors see set_prior.

Usage

```
gev_loglognorm(pars, mean, icov, min_xi = -Inf, max_xi = Inf, trendsd = 0)
```

Arguments

pars	A numeric vector of length 3. GEV parameters (μ, σ, ξ).
mean	A numeric vector of length 3. Prior mean.
icov	A 3x3 numeric matrix. The inverse of the prior covariance matrix.
min_xi	A numeric scalar. Prior lower bound on ξ.
max_xi	A numeric scalar. Prior upper bound on ξ.
trendsd	Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gev_mdi *Maximal data information (MDI) prior for GEV parameters (μ, σ, ξ)*

Description

For information about this and other priors see set_prior.

Usage

```
gev_mdi(pars, a = 0.577215664901532, min_xi = -1, max_xi = Inf, trendsd = 0)
```

Arguments

pars	A numeric vector of length 3. GEV parameters (μ, σ, ξ).
a	A numeric scalar. The default value, Euler's constant, gives the MDI prior.
min_xi	A numeric scalar. Prior lower bound on ξ. Must not be -Inf because this results in an improper posterior.
max_xi	A numeric scalar. Prior upper bound on ξ.
trendsd	Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gev_norm — *Trivariate normal prior for GEV parameters ($\mu, log\sigma, \xi$)*

Description

For information about this and other priors see set_prior.

Usage

```
gev_norm(pars, mean, icov, min_xi = -Inf, max_xi = Inf, trendsd = 0)
```

Arguments

`pars` A numeric vector of length 3. GEV parameters (μ, σ, ξ).

`mean` A numeric vector of length 3. Prior mean.

`icov` A 3x3 numeric matrix. The inverse of the prior covariance matrix.

`min_xi` A numeric scalar. Prior lower bound on ξ.

`max_xi` A numeric scalar. Prior upper bound on ξ.

`trendsd` Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gev_prob — *Informative GEV prior on a probability scale*

Description

Constructs an informative prior for GEV parameters (μ, σ, ξ), constructed on the probability scale. For information about how to set this prior see set_prior.

Usage

```
gev_prob(pars, quant, alpha, min_xi = -Inf, max_xi = Inf, trendsd = 0)
```

Arguments

`pars` A numeric vector of length 3. GEV parameters (μ, σ, ξ).

`quant` A numeric vector of length 3 containing quantiles (q_1, q_2, q_3) such that $q_1 < q_2 < q_3$. If the values in `quant` are not ordered from smallest to largest then they will be ordered inside `set_prior` without warning.

`alpha` A numeric vector of length 4. Parameters specifying a prior distribution for probabilities related to the quantiles in `quant`. See **Details** below.

`min_xi` A numeric scalar. Prior lower bound on ξ.

`max_xi` A numeric scalar. Prior upper bound on ξ.

`trendsd` Has no function other than to achieve compatibility with function in the evdbayes package.

Details

A prior for GEV parameters (μ, σ, ξ), based on Crowder (1992). This construction is typically used to set an informative prior, based on specified quantiles q_1, q_2, q_3. There are two interpretations of the parameter vector `alpha` = $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$: as the parameters of beta distributions for ratio of exceedance probabilities (Stephenson, 2016) and as the parameters of a Dirichlet distribution for differences between non-exceedance probabilities (Northrop et al., 2017). See these publications for details.

Value

The log of the prior density.

References

Crowder, M. (1992) Bayesian priors based on parameter transformation using the distribution function *Ann. Inst. Statist. Math.*, **44**, 405-416. `https://link.springer.com/article/10.1007/BF00050695`.

Northrop, P. J., Attalides, N. and Jonathan, P. (2017) Cross-validatory extreme value threshold selection and uncertainty with application to ocean storm severity. *Journal of the Royal Statistical Society Series C: Applied Statistics*, **66**(1), 93-120. doi:10.1111/rssc.12159

Stephenson, A. (2016) Bayesian inference for extreme value modelling. In *Extreme Value Modeling and Risk Analysis: Methods and Applications* (eds D. K. Dey and J. Yan), 257-280, Chapman and Hall, London. doi:10.1201/b19721.

See Also

`set_prior` for setting a prior distribution.

`rpost` and `rpost_rcpp` for sampling from an extreme value posterior distribution.

Sets the same prior as the function `prior.prob` in the evdbayes package.

gev_quant *Informative GEV prior on a quantile scale*

Description

Informative GEV prior for GEV parameters (μ, σ, ξ) constructed on the quantile scale. For information about how to set this prior see `set_prior`.

Usage

```
gev_quant(pars, prob, shape, scale, min_xi = -Inf, max_xi = Inf, trendsd = 0)
```

Arguments

`pars`	A numeric vector of length 3. GEV parameters (μ, σ, ξ).
`prob`	A numeric vector of length 3 containing exceedance probabilities (p_1, p_2, p_3) such that $p_1 > p_2 > p_3$. If the values in `quant` are not ordered from largest to smallest then they will be ordered inside `set_prior` without warning.
`shape, scale`	Numeric vectors of length 3. Shape and scale parameters specifying (independent) gamma prior distributions placed on the differences between the quantiles corresponding to the probabilities given in `prob`.
`min_xi`	A numeric scalar. Prior lower bound on ξ.
`max_xi`	A numeric scalar. Prior upper bound on ξ.
`trendsd`	Has no function other than to achieve compatibility with function in the evdbayes package.

Details

See Coles and Tawn (1996) and/or Stephenson (2016) for details.

Note that the lower end point of the distribution of the distribution of the variable in question is assumed to be equal to zero. If this is not the case then the user should shift the data to ensure that this is true.

Value

The log of the prior density.

References

Coles, S. G. and Tawn, J. A. (1996) A Bayesian analysis of extreme rainfall data. *Appl. Statist.*, **45**, 463-478.

Stephenson, A. (2016) Bayesian inference for extreme value modelling. In *Extreme Value Modeling and Risk Analysis: Methods and Applications* (eds D. K. Dey and J. Yan), 257-280, Chapman and Hall, London. doi:10.1201/b19721.

`gom` *Storm peak significant wave heights from the Gulf of Mexico*

Description

A numeric vector containing 315 hindcasts of storm peak significant wave heights, metres, from 1900 to 2005 at an unnamed location in the Gulf of Mexico.

Usage

```
gom
```

Format

A vector containing 315 observations.

Source

Oceanweather Inc. (2005) GOMOS – Gulf of Mexico hindcast study.

References

Northrop, P. J., Attalides, N. and Jonathan, P. (2017) Cross-validatory extreme value threshold selection and uncertainty with application to ocean storm severity. *Journal of the Royal Statistical Society Series C: Applied Statistics*, **66**(1), 93-120. doi:10.1111/rssc.12159

`gp` *The Generalised Pareto Distribution*

Description

Density function, distribution function, quantile function and random generation for the generalised Pareto (GP) distribution.

Usage

```
dgp(x, loc = 0, scale = 1, shape = 0, log = FALSE)

pgp(q, loc = 0, scale = 1, shape = 0, lower.tail = TRUE, log.p = FALSE)

qgp(p, loc = 0, scale = 1, shape = 0, lower.tail = TRUE, log.p = FALSE)

rgp(n, loc = 0, scale = 1, shape = 0)
```

Arguments

`x, q` Numeric vectors of quantiles. All elements of `x` and `q` must be non-negative.

`loc, scale, shape` Numeric vectors. Location, scale and shape parameters. All elements of `scale` must be positive.

`log, log.p` A logical scalar; if TRUE, probabilities p are given as log(p).

`lower.tail` A logical scalar. If TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

`p` A numeric vector of probabilities in [0,1].

`n` Numeric scalar. The number of observations to be simulated. If `length(n) > 1` then `length(n)` is taken to be the number required.

Details

The distribution function of a GP distribution with parameters `location` = μ, `scale` = $\sigma (> 0)$ and `shape` = ξ is

$$F(x) = 1 - [1 + \xi(x - \mu)/\sigma]^{-1/\xi}$$

for $1 + \xi(x - \mu)/\sigma > 0$. If $\xi = 0$ the distribution function is defined as the limit as ξ tends to zero. The support of the distribution depends on ξ: it is $x \geq \mu$ for $\xi \geq 0$; and $\mu \leq x \leq \mu - \sigma/\xi$ for $\xi < 0$. Note that if $\xi < -1$ the GP density function becomes infinite as x approaches $\mu - \sigma/\xi$.

If `lower.tail = TRUE` then if `p = 0` (`p = 1`) then the lower (upper) limit of the distribution is returned. The upper limit is `Inf` if `shape` is non-negative. Similarly, but reversed, if `lower.tail = FALSE`.

See `https://en.wikipedia.org/wiki/Generalized_Pareto_distribution` for further information.

Value

`dgp` gives the density function, `pgp` gives the distribution function, `qgp` gives the quantile function, and `rgp` generates random deviates.

References

Pickands, J. (1975) Statistical inference using extreme order statistics. *Annals of Statistics*, **3**, 119-131. doi:10.1214/aos/1176343003

Coles, S. G. (2001) *An Introduction to Statistical Modeling of Extreme Values*, Springer-Verlag, London. Chapter 4: doi:10.1007/9781447136750_4

Examples

```
dgp(0:4, scale = 0.5, shape = 0.8)
dgp(1:6, scale = 0.5, shape = -0.2, log = TRUE)
dgp(1, scale = 1, shape = c(-0.2, 0.4))

pgp(0:4, scale = 0.5, shape = 0.8)
pgp(1:6, scale = 0.5, shape = -0.2)
pgp(1, scale = c(1, 2), shape = c(-0.2, 0.4))
pgp(7, scale = 1, shape = c(-0.2, 0.4))
```

```
qgp((0:9)/10, scale = 0.5, shape = 0.8)
qgp(0.5, scale = c(0.5, 1), shape = c(-0.5, 0.5))

p <- (1:9)/10
pgp(qgp(p, scale = 2, shape = 0.8), scale = 2, shape = 0.8)

rgp(6, scale = 0.5, shape = 0.8)
```

gp_beta *Beta-type prior for GP shape parameter ξ*

Description

For information about this and other priors see set_prior.

Usage

```
gp_beta(pars, min_xi = -1/2, max_xi = 1/2, pq = c(6, 9), trendsd = 0)
```

Arguments

pars	A numeric vector of length 2. GP parameters (σ, ξ).
min_xi	A numeric scalar. Prior lower bound on ξ.
max_xi	A numeric scalar. Prior upper bound on ξ.
pq	A numeric vector of length 2. See set_prior for details.
trendsd	Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gp_flat *Flat prior for GP parameters $(\log\sigma, \xi)$*

Description

For information about this and other priors see set_prior.

Usage

```
gp_flat(pars, min_xi = -Inf, max_xi = Inf, trendsd = 0)
```

Arguments

`pars` A numeric vector of length 2. GP parameters (σ, ξ).

`min_xi` A numeric scalar. Prior lower bound on ξ. Must not be `-Inf` because this results in an improper posterior.

`max_xi` A numeric scalar. Prior upper bound on ξ.

`trendsd` Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gp_flatflat *Flat prior for GP parameters (σ, ξ)*

Description

For information about this and other priors see `set_prior`.

Usage

```
gp_flatflat(pars, min_xi = -Inf, max_xi = Inf, trendsd = 0, upper = NULL)
```

Arguments

`pars` A numeric vector of length 2. GP parameters (σ, ξ).

`min_xi` A numeric scalar. Prior lower bound on ξ. Must not be `-Inf` because this results in an improper posterior.

`max_xi` A numeric scalar. Prior upper bound on ξ.

`trendsd` Has no function other than to achieve compatibility with function in the evdbayes package.

`upper` A positive numeric scalar. The upper endpoint of the GP distribution.

Value

The log of the prior density.

gp_jeffreys *Jeffreys prior for GP parameters (σ, ξ)*

Description

For information about this and other priors see set_prior.

Usage

```
gp_jeffreys(pars, min_xi = -1/2, max_xi = Inf, trendsd = 0)
```

Arguments

pars	A numeric vector of length 2. GP parameters (σ, ξ).
min_xi	A numeric scalar. Prior lower bound on ξ. Must not be -Inf because this results in an improper posterior.
max_xi	A numeric scalar. Prior upper bound on ξ.
trendsd	Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gp_lrs *Linear Combinations of Ratios of Spacings estimation of generalised Pareto parameters*

Description

Uses the Linear Combinations of Ratios of Spacings (LRS) methodology of (Reiss and Thomas, 2007, page 134) to estimate the parameters of the generalised Pareto (GP) distribution, based on a sample of positive values.

Usage

```
gp_lrs(x)
```

Arguments

x	A numeric vector containing only **positive** values, assumed to be a random sample from a generalized Pareto distribution.

Value

A numeric vector of length 2. The estimates of the scale parameter σ and the shape parameter ξ.

References

Reiss, R.-D., Thomas, M. (2007) Statistical Analysis of Extreme Values with Applications to Insurance, Finance, Hydrology and Other Fields.Birkhauser. doi:10.1007/9783764373993.

See Also

`gp` for details of the parameterisation of the GP distribution.

Examples

```
u <- quantile(gom, probs = 0.65)
gp_lrs((gom - u)[gom > u])
```

---

gp_mdi *Maximal data information (MDI) prior for GP parameters (σ, ξ)*

Description

For information about this and other priors see `set_prior`.

Usage

```
gp_mdi(pars, a = 1, min_xi = -1, max_xi = Inf, trendsd = 0)
```

Arguments

`pars` A numeric vector of length 2. GP parameters (σ, ξ).

`a` A numeric scalar. The default value of 1 gives the MDI prior.

`min_xi` A numeric scalar. Prior lower bound on ξ. Must not be `-Inf` because this results in an improper posterior. See Northrop and Attalides (2016) for details.

`max_xi` A numeric scalar. Prior upper bound on ξ.

`trendsd` Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

References

Northrop, P.J. and Attalides, N. (2016) Posterior propriety in Bayesian extreme value analyses using reference priors *Statistica Sinica*, **26(2)**, 721–743 doi:10.5705/ss.2014.034.

gp_norm *Bivariate normal prior for GP parameters ($log\sigma, \xi$)*

Description

For information about this and other priors see `set_prior`.

Usage

```
gp_norm(pars, mean, icov, min_xi = -Inf, max_xi = Inf, trendsd = 0)
```

Arguments

`pars` A numeric vector of length 2. GP parameters (σ, ξ).

`mean` A numeric vector of length 2. Prior mean.

`icov` A 2x2 numeric matrix. The inverse of the prior covariance matrix.

`min_xi` A numeric scalar. Prior lower bound on ξ.

`max_xi` A numeric scalar. Prior upper bound on ξ.

`trendsd` Has no function other than to achieve compatibility with function in the evdbayes package.

Value

The log of the prior density.

gp_pwm *Probability-weighted moments estimation of generalised Pareto parameters*

Description

Uses the methodology of Hosking and Wallis (1987) to estimate the parameters of the generalised Pareto (GP) distribution.

Usage

```
gp_pwm(gp_data, u = 0)
```

Arguments

`gp_data` A numeric vector of raw data, assumed to be a random sample from a probability distribution.

`u` A numeric scalar. A threshold. The GP distribution is fitted to the excesses of `u`.

Value

A list with components

- est: A numeric vector. PWM estimates of GP parameters σ (scale) and ξ (shape).
- se: A numeric vector. Estimated standard errors of σ and ξ.
- cov: A numeric matrix. Estimate covariance matrix of the the PWM estimators of σ and ξ.

References

Hosking, J. R. M. and Wallis, J. R. (1987) Parameter and Quantile Estimation for the Generalized Pareto Distribution. Technometrics, 29(3), 339-349. doi:10.2307/1269343.

See Also

gp for details of the parameterisation of the GP distribution.

Examples

```
u <- quantile(gom, probs = 0.65)
gp_pwm(gom, u)
```

grimshaw_gp_mle *Maximum likelihood estimation of generalised Pareto parameters*

Description

Uses the methodology of Grimshaw (1993) to find the MLEs of the parameters of the generalised Pareto distribution, based on a sample of positive values. The function is essentially the same as that made available with Grimshaw (1993), with only minor modifications.

Usage

```
grimshaw_gp_mle(x)
```

Arguments

x	A numeric vector containing only **positive** values, assumed to be a random sample from a generalized Pareto distribution.

Value

A numeric vector of length 2. The estimates of the **negated** shape parameter $k (= -\xi)$ and the scale parameter $a (= \sigma)$.

References

Grimshaw, S. D. (1993) Computing Maximum Likelihood Estimates for the Generalized Pareto Distribution. Technometrics, 35(2), 185-191. and Computing (1991) 1, 129-133. doi:10.1080/00401706.1993.10485040.

See Also

gp for details of the parameterisation of the GP distribution, in terms of σ and ξ.

Examples

```
u <- quantile(gom, probs = 0.65)
grimshaw_gp_mle((gom - u)[gom > u])
```

---

kgaps_post *Random sampling from K-gaps posterior distribution*

Description

Uses the rust package to simulate from the posterior distribution of the extremal index θ based on the K-gaps model for threshold interexceedance times of Suveges and Davison (2010).

Usage

```
kgaps_post(
  data,
  thresh,
  k = 1,
  n = 1000,
  inc_cens = TRUE,
  alpha = 1,
  beta = 1,
  param = c("logit", "theta"),
  use_rcpp = TRUE
)
```

Arguments

`data` A numeric vector or numeric matrix of raw data. If `data` is a matrix then the log-likelihood is constructed as the sum of (independent) contributions from different columns. A common situation is where each column relates to a different year.

If `data` contains missing values then split_by_NAs is used to divide the data further into sequences of non-missing values, stored in different columns in a matrix. Again, the log-likelihood is constructed as a sum of contributions from different columns.

`thresh` A numeric scalar. Extreme value threshold applied to data.

`k` A numeric scalar. Run parameter K, as defined in Suveges and Davison (2010). Threshold inter-exceedances times that are not larger than k units are assigned to the same cluster, resulting in a K-gap equal to zero. Specifically, the K-gap S corresponding to an inter-exceedance time of T is given by $S = \max(T - K, 0)$.

`n` A numeric scalar. The size of posterior sample required.

`inc_cens` A logical scalar indicating whether or not to include contributions from right-censored inter-exceedance times, relating to the first and last observations. It is known that these times are greater than or equal to the time observed. If `data` has multiple columns then there will be right-censored first and last inter-exceedance times for each column. See also the **Details** section of `kgaps`.

`alpha, beta` Positive numeric scalars. Parameters of a beta(α, β) prior for θ.

`param` A character scalar. If `param = "logit"` (the default) then we simulate from the posterior distribution of $\phi = \log(\theta/(1-\theta))$ and then transform back to the θ-scale. If `param = "theta"` then we simulate directly from the posterior distribution of θ, unless the sample K-gaps are all equal to zero or all positive, when we revert to `param = "logit"`. This is to avoid sampling directly from a posterior with mode equal to 0 or 1.

`use_rcpp` A logical scalar. If `TRUE` (the default) the rust function `ru_rcpp` is used for posterior simulation. If `FALSE` the (slower) function `ru` is used.

Details

A beta(α, β) prior distribution is used for θ so that the posterior from which values are simulated is proportional to

$$\theta^{2N_1+\alpha-1}(1-\theta)^{N_0+\beta-1}\exp\{-\theta q(S_0+\cdots+S_N)\}.$$

See `kgaps_stat` for a description of the variables involved in the contribution of the likelihood to this expression.

The `ru` function in the `rust` package simulates from this posterior distribution using the generalised ratio-of-uniforms distribution. To improve the probability of acceptance, and to ensure that the simulation will work even in extreme cases where the posterior density of θ is unbounded as θ approaches 0 or 1, we simulate from the posterior distribution of $\phi = \log(\theta/(1-\theta))$ and then transform back to the θ-scale.

Value

An object (list) of class `"evpost"`, which has the same structure as an object of class `"ru"` returned from `ru`. In addition this list contains

- `call`: The call to `kgaps()`.
- `model`: The character scalar `"kgaps"`.
- `thresh`: The argument `thresh`.
- `ss`: The sufficient statistics for the K-gaps likelihood, as calculated by `kgaps_stat`.

References

Suveges, M. and Davison, A. C. (2010) Model misspecification in peaks over threshold analysis, *The Annals of Applied Statistics*, **4**(1), 203-221. doi:10.1214/09AOAS292

See Also

`ru` for the form of the object returned by `kgaps_post`.

`dgaps_post` for Bayesian inference about the extremal index θ using the D-gaps model.

Examples

```
# Newlyn sea surges

thresh <- quantile(newlyn, probs = 0.90)
k_postsim <- kgaps_post(newlyn, thresh)
plot(k_postsim)

### Cheeseboro wind gusts

k_postsim <- kgaps_post(exdex::cheeseboro, thresh = 45, k = 3)
plot(k_postsim)
```

---

newlyn *Newlyn sea surges*

Description

The vector newlyn contains 2894 maximum sea-surges measured at Newlyn, Cornwall, UK over the period 1971-1976. The observations are the maximum hourly sea-surge heights over contiguous 15-hour time periods.

Usage

```
newlyn
```

Format

A vector of length 2894.

Source

Coles, S.G. (1991) Modelling extreme multivariate events. PhD thesis, University of Sheffield, U.K.

References

Fawcett, L. and Walshaw, D. (2012) Estimating return levels from serially dependent extremes. *Environmetrics*, **23**(3), 272-283. doi:10.1002/env.2133

Northrop, P. J. (2015) An efficient semiparametric maxima estimator of the extremal index. *Extremes*, **18**, 585-603. doi:10.1007/s1068701502215

oxford *Annual Maximum Temperatures at Oxford*

Description

A numeric vector containing annual maximum temperatures, in degrees Fahrenheit, from 1901 to 1980 at Oxford, England.

Usage

```
oxford
```

Format

A vector containing 80 observations.

Source

Tabony, R. C. (1983) Extreme value analysis in meteorology. *The Meteorological Magazine*, **112**, 77-98.

plot.evpost *Plot diagnostics for an evpost object*

Description

plot method for class "evpost". For d = 1 a histogram of the simulated values is plotted with a the density function superimposed. The density is normalized crudely using the trapezium rule. For d = 2 a scatter plot of the simulated values is produced with density contours superimposed. For d > 2 pairwise plots of the simulated values are produced. An interface is also provided to the functions in the **bayesplot** package that produce plots of Markov chain Monte Carlo (MCMC) simulations. See MCMC-overview for details of these functions.

Usage

```
## S3 method for class 'evpost'
plot(
  x,
  y,
  ...,
  n = ifelse(x$d == 1, 1001, 101),
  prob = c(0.5, 0.1, 0.25, 0.75, 0.95, 0.99),
  ru_scale = FALSE,
  rows = NULL,
  xlabs = NULL,
  ylabs = NULL,
  points_par = list(col = 8),
  pu_only = FALSE,
  add_pu = FALSE,
  use_bayesplot = FALSE,
  fun_name = c("areas", "intervals", "dens", "hist", "scatter")
)
```

Arguments

`x` An object of class "evpost", a result of a call to `rpost` or `rpost_rcpp`.

`y` Not used.

`...` Additional arguments passed on to `hist`, `lines`, `contour`, `points` or functions from the **bayesplot** package.

`n` A numeric scalar. Only relevant if `x$d = 1` or `x$d = 2`. The meaning depends on the value of x$d.

- For d = 1 : n + 1 is the number of abscissae in the trapezium method used to normalize the density.
- For d = 2 : an n by n regular grid is used to contour the density.

`prob` Numeric vector. Only relevant for d = 2. The contour lines are drawn such that the respective probabilities that the variable lies within the contour are approximately prob.

`ru_scale` A logical scalar. Should we plot data and density on the scale used in the ratio-of-uniforms algorithm (TRUE) or on the original scale (FALSE)?

`rows` A numeric scalar. When `d` > 2 this sets the number of rows of plots. If the user doesn't provide this then it is set internally.

`xlabs, ylabs` Numeric vectors. When `d` > 2 these set the labels on the `x` and `y` axes respectively. If the user doesn't provide these then the column names of the simulated data matrix to be plotted are used.

`points_par` A list of arguments to pass to `points` to control the appearance of points depicting the simulated values. Only relevant when `d = 2`.

`pu_only` Only produce a plot relating to the posterior distribution for the threshold exceedance probability p. Only relevant when `model == "bingp"` was used in the call to `rpost` or `rpost_rcpp`.

`add_pu` Before producing the plots add the threshold exceedance probability p to the parameters of the extreme value model. Only relevant when `model == "bingp"` was used in the call to `rpost` or `rpost_rcpp`.

`use_bayesplot` A logical scalar. If `TRUE` the bayesplot function indicated by `fun_name` is called. In principle *any* bayesplot function (that starts with `mcmc_`) can be called but this may not always be successful because, for example, some of the bayesplot functions work only with multiple MCMC simulations.

`fun_name` A character scalar. The name of the bayesplot function, with the initial `mcmc_` part removed. See MCMC-overview and links therein for the names of these functions. Some examples are given below.

Details

For details of the **bayesplot** functions available when `use_bayesplot = TRUE` see MCMC-overview and the **bayesplot** vignette Plotting MCMC draws.

Value

Nothing is returned unless `use_bayesplot = TRUE` when a ggplot object, which can be further customized using the **ggplot2** package, is returned.

References

Jonah Gabry (2016). bayesplot: Plotting for Bayesian Models. R package version 1.1.0. https://CRAN.R-project.org/package=bayesplot

See Also

`summary.evpost` for summaries of the simulated values and properties of the ratio-of-uniforms algorithm.

`MCMC-overview`, `MCMC-intervals`, `MCMC-distributions`.

Examples

```
## GP posterior
u <- stats::quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
gpg <- rpost(n = 1000, model = "gp", prior = fp, thresh = u, data = gom)
plot(gpg)


# Using the bayesplot package
plot(gpg, use_bayesplot = TRUE)
plot(gpg, use_bayesplot = TRUE, pars = "xi", prob = 0.95)
plot(gpg, use_bayesplot = TRUE, fun_name = "intervals", pars = "xi")
plot(gpg, use_bayesplot = TRUE, fun_name = "hist")
plot(gpg, use_bayesplot = TRUE, fun_name = "dens")
plot(gpg, use_bayesplot = TRUE, fun_name = "scatter")


## bin-GP posterior
u <- quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
bp <- set_bin_prior(prior = "jeffreys")
npy_gom <- length(gom)/105
bgpg <- rpost(n = 1000, model = "bingp", prior = fp, thresh = u,
              data = gom, bin_prior = bp, npy = npy_gom)
plot(bgpg)
plot(bgpg, pu_only = TRUE)
plot(bgpg, add_pu = TRUE)


# Using the bayesplot package
```

```
dimnames(bgpg$bin_sim_vals)
plot(bgpg, use_bayesplot = TRUE)
plot(bgpg, use_bayesplot = TRUE, fun_name = "hist")
plot(bgpg, use_bayesplot = TRUE, pars = "p[u]")
```

plot.evpred *Plot diagnostics for an evpred object*

Description

plot method for class "evpred". Plots summarising the predictive distribution of the largest value to be observed in N years are produced. The plot produced depends on x$type. If x$type = "d", "p" or "q" then matplot is used to produce a line plot of the predictive density, distribution or quantile function, respectively, with a line for each value of N in x$n_years. If x$type = "r" then estimates of the predictive density (from density) are plotted with a line for each N. If x$type = "i" then lines representing estimated predictive intervals are plotted, with the level of the interval indicated next to the line.

Usage

```
## S3 method for class 'evpred'
plot(
  x,
  ...,
  leg_pos = NULL,
  leg_text = NULL,
  which_int = c("long", "short", "both")
)
```

Arguments

x	An object of class "evpost", a result of a call to rpost.
...	Additional arguments passed on to matplot.
leg_pos	A character scalar. Keyword for the position of legend. See legend.
leg_text	A character or expression vector. Text for legend. See legend.
which_int	A character scalar. If x$type = "i" which intervals should be plotted? "long" for equi-tailed intervals, "short" for the shortest possible intervals, "both" for both.

Value

Nothing is returned.

See Also

predict.evpost for the S3 predict method for objects of class evpost.

Examples

```
data(portpirie)
mat <- diag(c(10000, 10000, 100))
pn <- set_prior(prior = "norm", model = "gev", mean = c(0,0,0), cov = mat)
gevp  <- rpost(n = 1000, model = "gev", prior = pn, data = portpirie)

# Predictive density function
d_gevp <- predict(gevp, type = "d", n_years = c(100, 1000))
plot(d_gevp)

# Predictive distribution function
p_gevp <- predict(gevp, type = "p", n_years = c(100, 1000))
plot(p_gevp)

# Predictive quantiles
q_gevp <- predict(gevp, type = "q", n_years = c(100, 1000))
plot(q_gevp)

# Predictive intervals
i_gevp <- predict(gevp, type = "i", n_years = c(100, 1000), hpd = TRUE)
plot(i_gevp, which_int = "both")

# Sample from predictive distribution
r_gevp <- predict(gevp, type = "r", n_years = c(100, 1000))
plot(r_gevp)
plot(r_gevp, xlim = c(4, 10))
```

portpirie *Annual Maximum Sea Levels at Port Pirie, South Australia*

Description

A numeric vector of length 65 containing annual maximum sea levels, in metres, from 1923 to 1987 at Port Pirie, South Australia.

Usage

```
portpirie
```

Format

A numeric vector containing 65 observations.

Source

Coles, S. G. (2001) *An Introduction to Statistical Modelling of Extreme Values*. London: Springer. doi:10.1007/9781447136750

`pp_check.evpost` *Posterior predictive checks for an evpost object*

Description

pp_check method for class "evpost". This provides an interface to the functions that perform posterior predictive checks in the **bayesplot** package. See PPC-overview for details of these functions.

Usage

```
## S3 method for class 'evpost'
pp_check(
  object,
  ...,
  type = c("stat", "overlaid", "multiple", "intervals", "user"),
  subtype = NULL,
  stat = "median",
  nrep = 8,
  fun = NULL
)
```

Arguments

`object` An object of class "evpost", a result of a call to `rpost` or `rpost_rcpp`. Currently `object$model = "gev", "gp", "bingp"` and `"pp"` are supported.

`...` Additional arguments passed on to bayesplot functions.

`type` A character vector. The type of bayesplot plot required:

- "stat" for predictive test statistics (see PPC-test-statistics),
- "overlaid" for comparison of observed data to predictive simulated datasets using overlaid density function or distribution functions (see PPC-distributions),
- "multiple" for comparison of observed data to predictive simulated datasets using multiple summary plots (see PPC-distributions),
- "intervals" for comparison of observed data to predictive simulated datasets using sample medians and a predictive interval, (see PPC-intervals),
- "user" for direct access to the default bayesplot function `pp_check`. This requires the argument `fun` to be supplied (see pp_check).

`subtype` A character scalar. Specifies the form of the plot(s) produced. Could be one of `"dens"`, `"hist"`, `"boxplot"`, `"ribbon"` or `"intervals"`. If `subtype` is not supplied then the defaults are: `"ecdf"` if `type = overlaid`, `"dens"` if `type = multiple`, `"intervals"` if `type = intervals`. `subtype` is not relevant if `type = "stat"`.

`stat` See PPC-test-statistics.

`nrep` If `type = "multiple"` the maximum number of summary plots of the predictive simulated datasets to include. If `nrep` is greater than `nrow(object$data_rep)` then `nrep` is set equal to `nrow(object$data_rep)`.

fun The plotting function to call. Only relevant if type = "user". Can be any of the functions detailed at PPC-overview. The "ppc_" prefix can optionally be dropped if fun is specified as a string.

Details

For details of these functions see PPC-overview. See also the vignette Posterior Predictive Extreme Value Inference and the **bayesplot** vignette Graphical posterior predictive checks.

The general idea is to compare the observed data object$data with a matrix object$data_rep in which each row is a replication of the observed data simulated from the posterior predictive distribution. For greater detail see Chapter 6 of Gelman et al. (2013).

The format of object$data depends on the model:

- model = "gev". A vector of block maxima.
- model = "gp". Data that lie above the threshold, i.e. threshold exceedances.
- model = "bingp" or model = "pp". The input data are returned but any value lying below the threshold is set to object$thresh.

In all cases any missing values have been removed from the data.

If model = "bingp" or "pp" the rate of threshold exceedance is part of the inference. Therefore, the number of values in object$data_rep that lie above the threshold varies between predictive replications, with values below the threshold being left-censored at the threshold. This limits a little the posterior predictive checks that it is useful to perform. In the examples below we have compared object$data and object$data_rep using only their sample maxima.

Value

A ggplot object that can be further customized using the **ggplot2** package.

References

Jonah Gabry (2016). bayesplot: Plotting for Bayesian Models. R package version 1.1.0. https://CRAN.R-project.org/package=bayesplot

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian Data Analysis*. Chapman & Hall/CRC Press, London, third edition. (Chapter 6)

See Also

rpost and rpost_rcpp for sampling from an extreme value posterior distribution.

bayesplot functions PPC-overview, PPC-distributions, PPC-test-statistics, PPC-intervals, pp_check.

Examples

```
# GEV model
data(portpirie)
mat <- diag(c(10000, 10000, 100))
pn <- set_prior(prior = "norm", model = "gev", mean = c(0,0,0), cov = mat)
gevp  <- rpost(1000, model = "gev", prior = pn, data = portpirie,
               nrep = 50)
```

```
# Posterior predictive test statistics
pp_check(gevp)
pp_check(gevp, stat = "min")
pp_check(gevp, stat = c("min", "max"))
iqr <- function(y) diff(quantile(y, c(0.25, 0.75)))
pp_check(gevp, stat = "iqr")

# Overlaid density and distributions functions
pp_check(gevp, type = "overlaid")
pp_check(gevp, type = "overlaid", subtype = "dens")

# Multiple plots
pp_check(gevp, type = "multiple")
pp_check(gevp, type = "multiple", subtype = "hist")
pp_check(gevp, type = "multiple", subtype = "boxplot")

# Intervals
pp_check(gevp, type = "intervals")
pp_check(gevp, type = "intervals", subtype = "ribbon")

# User-supplied bayesplot function
# Equivalent to p_check(gevp, type = "overlaid")
pp_check(gevp, type = "user", fun = "dens_overlay")

# GP model
u <- quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
gpg <- rpost(n = 1000, model = "gp", prior = fp, thresh = u,
             data = gom, nrep = 50)
pp_check(gpg)
pp_check(gpg, type = "overlaid")

# bin-GP model
bp <- set_bin_prior(prior = "jeffreys")
bgpg <- rpost(n = 1000, model = "bingp", prior = fp, thresh = u,
              data = gom, bin_prior = bp, nrep = 50)
pp_check(bgpg, stat = "max")

# PP model
data(rainfall)
rthresh <- 40
pf <- set_prior(prior = "flat", model = "gev", min_xi = -1)
ppr <- rpost(n = 1000, model = "pp", prior = pf, data = rainfall,
             thresh = rthresh, noy = 54, nrep = 50)
pp_check(ppr, stat = "max")
```

predict.evpost *Predictive inference for the largest value observed in N years.*

Description

predict method for class "evpost". Performs predictive inference about the largest value to be observed over a future time period of N years. Predictive inferences accounts for uncertainty in model parameters and for uncertainty owing to the variability of future observations.

Usage

```
## S3 method for class 'evpost'
predict(
  object,
  type = c("i", "p", "d", "q", "r"),
  x = NULL,
  x_num = 100,
  n_years = 100,
  npy = NULL,
  level = 95,
  hpd = FALSE,
  lower_tail = TRUE,
  log = FALSE,
  big_q = 1000,
  ...
)
```

Arguments

object An object of class "evpost", a result of a call to rpost or rpost_rcpp with model = "gev", model = "os", model = "pp" or model == "bingp". Calling these functions after a call to rpost or rpost_rcpp with model == "gp" will produce an error, because inferences about the probability of threshold exceedance are required, in addition to the distribution of threshold excesses. The model is stored in object$model.

object may also be an object created within the function predict.blite in the lite package. In this case object$sim_vals has a column named "theta" containing a posterior sample of values of the extremal index.

type A character vector. Indicates which type of inference is required:

- "i" for predictive intervals,
- "p" for the predictive distribution function,
- "d" for the predictive density function,
- "q" for the predictive quantile function,
- "r" for random generation from the predictive distribution.

x A numeric vector or a matrix with n_years columns. The meaning of x depends on type.

- type = "p" or type = "d": x contains quantiles at which to evaluate the distribution or density function.
  If object$model == "bingp" then no element of x can be less than the threshold object$thresh.

If `x` is not supplied then `n_year`-specific defaults are set: vectors of length `x_num` from the 0.1% quantile to the 99% quantile, subject all values being greater than the threshold.

- `type = "q"`: `x` contains probabilities in (0,1) at which to evaluate the quantile function. Any values outside (0, 1) will be removed without warning.
  If `object$model == "bingp"` then no element of `p` can correspond to a predictive quantile that is below the threshold, `object$thresh`. That is, no element of `p` can be less than the value of `predict.evpost(object, type = "q", x = object$thresh)`.
  If `x` is not supplied then a default value of `c(0.025, 0.25, 0.5, 0.75, 0.975)` is used.
- `type = "i"` or `type = "r"`: `x` is not relevant.

`x_num` A numeric scalar. If `type = "p"` or `type = "d"` and `x` is not supplied then `x_num` gives the number of values in `x` for each value in `n_years`.

`n_years` A numeric vector. Values of N.

`npy` A numeric scalar. The mean number of observations per year of data, after excluding any missing values, i.e. the number of non-missing observations divided by total number of years' worth of non-missing data.

If `rpost` or `rpost_rcpp` was called with `model == "bingp"` then `npy` must either have been supplied in that call or be supplied here.

Otherwise, a default value will be assumed if `npy` is not supplied, based on the value of `model` in the call to `rpost` or `rpost_rcpp`:

- `model = "gev"`: `npy` = 1, i.e. the data were annual maxima so the block size is one year.
- `model = "os"`: `npy = 1`, i.e. the data were annual order statistics so the block size is one year.
- `model = "pp"`: `npy = length(x$data) / object$noy`, i.e. the value of `noy` used in the call to `rpost` or `rpost_rcpp` is equated to a block size of one year.

If `npy` is supplied twice then the value supplied here will be used and a warning given.

`level` A numeric vector of values in (0, 100). Only relevant when `type = "i"`. Levels of predictive intervals for the largest value observed in N years, i.e. level% predictive intervals are returned.

`hpd` A logical scalar. Only relevant when `type = "i"`.

If `hpd = FALSE` then the interval is equi-tailed, with its limits produced by `predict.evpost(object, type ="q", x = p)`, where `p = c((1-level/100)/2, (1+level/100)/2)`.

If `hpd = TRUE` then, in addition to the equi-tailed interval, the shortest possible level% interval is calculated. If the predictive distribution is unimodal then this is a highest predictive density (HPD) interval.

`lower_tail` A logical scalar. Only relevant when `type = "p"` or `type = "q"`. If TRUE (default), (output or input) probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

`log` A logical scalar. Only relevant when `type = "d"`. If TRUE the log-density is returned.

`big_q` A numeric scalar. Only relevant when `type = "q"`. An initial upper bound for the desired quantiles to be passed to `uniroot` (its argument `upper`) in the search for the predictive quantiles. If this is not sufficiently large then it is increased until it does provide an upper bound.

`...` Additional optional arguments. At present no optional arguments are used.

Details

Inferences about future extreme observations are integrated over the posterior distribution of the model parameters, thereby accounting for uncertainty in model parameters and uncertainty owing to the variability of future observations. In practice the integrals involved are estimated using an empirical mean over the posterior sample. See, for example, Coles (2001), Stephenson (2016) or Northrop et al. (2017) for details. See also the vignette Posterior Predictive Extreme Value Inference

GEV / OS / PP. If `model = "gev"`, `model = "os"` or `model = "pp"` in the call to `rpost` or `rpost_rcpp` we first calculate the number of blocks b in `n_years` years. To calculate the density function or distribution function of the maximum over `n_years` we call `dgev` or `pgev` with `m` = b.

- `type = "p"`. We calculate using `pgev` the GEV distribution function at `q` for each of the posterior samples of the location, scale and shape parameters. Then we take the mean of these values.
- `type = "d"`. We calculate using `dgev` the GEV density function at `x` for each of the posterior samples of the location, scale and shape parameters. Then we take the mean of these values.
- `type = "q"`. We solve numerically `predict.evpost(object, type = "p", x = q) = p[i]` numerically for `q` for each element `p[i]` of `p`.
- `type = "i"`. If `hpd = FALSE` then the interval is equi-tailed, equal to `predict.evpost() object, type ="q", x = p)`, where `p = c((1-level/100)/2, (1+level/100)/2)`. If `hpd = TRUE` then, in addition, we perform a numerical minimisation of the length of level% intervals, after approximating the predictive quantile function using monotonic cubic splines, to reduce computing time.
- `type = "r"`. For each simulated value of the GEV parameters at the `n_years` level of aggregation we simulate one value from this GEV distribution using `rgev`. Thus, each sample from the predictive distribution is of a size equal to the size of the posterior sample.

Binomial-GP. If `model = "bingp"` in the call to `rpost` or `rpost_rcpp` then we calculate the mean number of observations in `n_years` years, i.e. `npy * n_years`.

Following Northrop et al. (2017), let M_N be the largest value observed in N years, m = `npy * n_years` and u the threshold `object$thresh` used in the call to `rpost` or `rpost_rcpp`. For fixed values of $\theta = (p, \sigma, \xi)$ the distribution function of M_N is given by $F(z, \theta)^m$, for $z \geq u$, where

$$F(z, \theta) = 1 - p[1 + \xi(x - u)/\sigma]^{-1/\xi}.$$

The distribution function of M_N cannot be evaluated for $z < u$ because no model has been supposed for observations below the threshold.

- `type = "p"`. We calculate $F(z, \theta)^m$ at `q` for each of the posterior samples θ. Then we take the mean of these values.
- `type = "d"`. We calculate the density of of M_n, i.e. the derivative of $F(z, \theta)^m$ with respect to z at `x` for each of the posterior samples θ. Then we take the mean of these values.

- type = "q" and type = "i". We perform calculations that are analogous to the GEV case above. If n_years is very small and/or level is very close to 100 then a predictive interval may extend below the threshold. In such cases NAs are returned (see **Value** below).
- type = "r". For each simulated value of the bin-GP parameter we simulate from the distribution of M_N using the inversion method applied to the distribution function of M_N given above. Occasionally a value below the threshold would need to be simulated. If these instances a missing value code NA is returned. Thus, each sample from the predictive distribution is of a size equal to the size of the posterior sample, perhaps with a small number os NAs.

Value

An object of class "evpred", a list containing a subset of the following components:

type The argument type supplied to predict.evpost. Which of the following components are present depends type.

x A matrix containing the argument x supplied to predict.evpost, or set within predict.evpost if x was not supplied, replicated to have n_years columns if necessary. Only present if type is "p", "d" or "q".

y The content of y depends on type:

- type = "p", "d", "q": A matrix with the same dimensions as x. Contains distribution function values (type = "p"), predictive density (type = "d") or quantiles (type = "q").
- type = "r": A numeric matrix with length(n_years) columns and number of rows equal to the size of the posterior sample.
- type = "i": y is not present.

long A length(n_years)*length(level) by 4 numeric matrix containing the equi-tailed limits with columns: lower limit, upper limit, n_years, level. Only present if type = "i". If an interval extends below the threshold then NA is returned.

short A matrix with the same structure as long containing the HPD limits. Only present if type = "i". Columns 1 and 2 contain NAs if hpd = FALSE or if the corresponding equi-tailed interval extends below the threshold.

The arguments n_years, level, hpd, lower_tail, log supplied to predict.evpost are also included, as is the argument npy supplied to, or set within, predict.evpost and the arguments data and model from the original call to rpost or rpost_rcpp.

References

Coles, S. G. (2001) *An Introduction to Statistical Modeling of Extreme Values*, Springer-Verlag, London. Chapter 9: doi:10.1007/9781447136750_9

Northrop, P. J., Attalides, N. and Jonathan, P. (2017) Cross-validatory extreme value threshold selection and uncertainty with application to ocean storm severity. *Journal of the Royal Statistical Society Series C: Applied Statistics*, **66**(1), 93-120. doi:10.1111/rssc.12159

Stephenson, A. (2016). Bayesian Inference for Extreme Value Modelling. In *Extreme Value Modeling and Risk Analysis: Methods and Applications*, edited by D. K. Dey and J. Yan, 257-80. London: Chapman and Hall. doi:10.1201/b19721

See Also

plot.evpred for the S3 plot method for objects of class evpred.

rpost or rpost_rcpp for sampling from an extreme value posterior distribution.

Examples

```
### GEV
data(portpirie)
mat <- diag(c(10000, 10000, 100))
pn <- set_prior(prior = "norm", model = "gev", mean = c(0,0,0), cov = mat)
gevp  <- rpost_rcpp(n = 1000, model = "gev", prior = pn, data = portpirie)

# Interval estimation
predict(gevp)$long
predict(gevp, hpd = TRUE)$short
# Density function
x <- 4:7
predict(gevp, type = "d", x = x)$y
plot(predict(gevp, type = "d", n_years = c(100, 1000)))
# Distribution function
predict(gevp, type = "p", x = x)$y
plot(predict(gevp, type = "p", n_years = c(100, 1000)))
# Quantiles
predict(gevp, type = "q", n_years = c(100, 1000))$y
# Random generation
plot(predict(gevp, type = "r"))

### Binomial-GP
u <- quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
bp <- set_bin_prior(prior = "jeffreys")
npy_gom <- length(gom)/105
bgpg <- rpost_rcpp(n = 1000, model = "bingp", prior = fp, thresh = u,
                   data = gom, bin_prior = bp)

# Setting npy in call to predict.evpost()
predict(bgpg, npy = npy_gom)$long

# Setting npy in call to rpost() or rpost_rcpp()
bgpg <- rpost_rcpp(n = 1000, model = "bingp", prior = fp, thresh = u,
                   data = gom, bin_prior = bp, npy = npy_gom)

# Interval estimation
predict(bgpg)$long
predict(bgpg, hpd = TRUE)$short
# Density function
plot(predict(bgpg, type = "d", n_years = c(100, 1000)))
# Distribution function
plot(predict(bgpg, type = "p", n_years = c(100, 1000)))
# Quantiles
predict(bgpg, type = "q", n_years = c(100, 1000))$y
# Random generation
```

```
plot(predict(bgpg, type = "r"))
```

print.evpost *Print method for objects of class "evpost"*

Description

Print method for objects of class "evpost"

Usage

```
## S3 method for class 'evpost'
print(x, ...)
```

Arguments

x	An object of class "evpost", a result of a call to rpost, rpost_rcpp, kgaps_post or dgaps_post.
...	Further arguments. None are used.

Details

print.evpost just prints the original function call, to avoid printing a huge list.

Value

The argument x is returned, invisibly.

See Also

plot.evpost for a diagnostic plot.

Examples

```
# Newlyn sea surges

thresh <- quantile(newlyn, probs = 0.90)
k_postsim <- kgaps_post(newlyn, thresh)
k_postsim
```

print.summary.evpost *Print method for objects of class "summary.evpost"*

Description

print method for an object object of class "summary.evpost".

Usage

```
## S3 method for class 'summary.evpost'
print(x, ...)
```

Arguments

x	An object of class "summary.evpost", a result of a call to summary.evpost.
...	Additional arguments passed on to print.

Value

Prints

- information about the ratio-of-uniforms bounding box, i.e. object$box
- an estimate of the probability of acceptance, i.e. object$pa
- a summary of the simulated values, via summary(object$sim_vals)

See Also

ru or ru_rcpp for descriptions of object$sim_vals and $box.

plot.evpost for a diagnostic plot.

Examples

```
# GP posterior
u <- stats::quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
gpg <- rpost_rcpp(n = 1000, model = "gp", prior = fp, thresh = u,
                  data = gom)
summary(gpg)
```

`quantile_to_gev` *Converts quantiles to GEV parameters*

Description

Three quantiles, that is, the value of quantile and their respective exceedance probabilities, are provided. This function attempts to find the location, scale and shape parameters of a GEV distribution that has these quantiles.

Usage

```
quantile_to_gev(quant, prob)
```

Arguments

`quant` A numeric vector of length 3. Values of the quantiles. The values should *increase* with the index of the vector. If not, the values in `quant` will be sorted into increasing order without warning.

`prob` A numeric vector of length 3. Exceedance probabilities corresponding to the quantiles in `quant`. The values should *decrease* with the index of the vector. If not, the values in `prob` will be sorted into decreasing order without warning.

Details

Suppose that $G(x)$ is the distribution function of a GEV(μ, σ, ξ) distribution. This function attempts to solve numerically the set of three non-linear equations

$$G(q_i) = 1 - p_i, i = 1, 2, 3$$

where $q_i, i = 1, 2, 3$ are the quantiles in `quant` and $p_i, i = 1, 2, 3$ are the exceedance probabilities in `prob`. This is reduced to a one-dimensional optimisation over the GEV shape parameter.

Value

A numeric vector of length 3 containing the GEV location, scale and shape parameters.

See Also

`rprior_quant` for simulation of GEV parameters from a prior constructed on the quantile scale.

Examples

```
my_q <- c(15, 20, 22.5)
my_p <- 1-c(0.5, 0.9, 0.5^0.01)
x <- quantile_to_gev(quant = my_q, prob = my_p)
# Check
qgev(p = 1 - my_p, loc = x[1], scale = x[2], shape = x[3])
```

`rainfall` *Daily Aggregate Rainfall*

Description

A numeric vector of length 20820 containing daily aggregate rainfall observations, in millimetres, recorded at a rain gauge in England over a 57 year period, beginning on a leap year. Three of these years contain only missing values.

Usage

```
rainfall
```

Format

A vector containing 20820 observations.

Source

Unknown

`rDir` *Simulation from a Dirichlet distribution*

Description

Simulates from a Dirichlet distribution with concentration parameter vector α = (α_1, ..., α_K).

Usage

```
rDir(n = 1, alpha = c(1, 1))
```

Arguments

`n` A numeric scalar. The size of sample required.

`alpha` A numeric vector. Dirichlet concentration parameter.

Details

The simulation is based on the property that if $Y_1, \ldots, Y_K$ are independent, Y_i has a gamma(α_i, 1) distribution and $S = Y_1 + \cdots + Y_k$ then $(Y_1, \ldots, Y_K)/S$ has a Dirichlet(α_1, ..., α_K) distribution.

See https://en.wikipedia.org/wiki/Dirichlet_distribution#Gamma_distribution

Value

An `n` by `length(alpha)` numeric matrix.

References

Kotz, S., Balakrishnan, N. and Johnson, N. L. (2000) *Continuous Multivariate Distributions, vol. 1, Models and Applications, 2nd edn*, ch. 49. New York: Wiley. doi:10.1002/0471722065

See Also

`rprior_prob` for prior simulation of GEV parameters - prior on probability scale.

Examples

```
rDir(n = 10, alpha = 1:4)
```

---

rpost — *Random sampling from extreme value posterior distributions*

Description

Uses the `ru` function in the `rust` package to simulate from the posterior distribution of an extreme value model.

Usage

```
rpost(
  n,
  model = c("gev", "gp", "bingp", "pp", "os"),
  data,
  prior,
  ...,
  nrep = NULL,
  thresh = NULL,
  noy = NULL,
  use_noy = TRUE,
  npy = NULL,
  ros = NULL,
 bin_prior = structure(list(prior = "bin_beta", ab = c(1/2, 1/2), class = "binprior")),
  bin_param = "logit",
  init_ests = NULL,
  mult = 2,
  use_phi_map = FALSE,
  weights = NULL
)
```

Arguments

`n`	A numeric scalar. The size of posterior sample required.
`model`	A character string. Specifies the extreme value model.
`data`	Sample data, of a format appropriate to the value of `model`.

- `"gp"`. A numeric vector of threshold excesses or raw data.
- `"bingp"`. A numeric vector of raw data.
- `"gev"`. A numeric vector of block maxima.
- `"pp"`. A numeric vector of raw data.
- `"os"`. A numeric matrix or data frame. Each row should contain the largest order statistics for a block of data. These need not be ordered: they are sorted inside `rpost`. If a block contains fewer than `dim(as.matrix(data))[2]` order statistics then the corresponding row should be padded by `NA`s. If `ros` is supplied then only the largest `ros` values in each row are used. If a vector is supplied then this is converted to a matrix with one column. This is equivalent to using `model = "gev"`.

`prior` A list specifying the prior for the parameters of the extreme value model, created by `set_prior`.

`...` Further arguments to be passed to `ru`. Most importantly `trans` and `rotate` (see **Details**), and perhaps `r`, `ep`, `a_algor`, `b_algor`, `a_method`, `b_method`, `a_control`, `b_control`. May also be used to pass the arguments `n_grid` and/or `ep_bc` to `find_lambda`.

`nrep` A numeric scalar. If `nrep` is not `NULL` then `nrep` gives the number of replications of the original dataset simulated from the posterior predictive distribution. Each replication is based on one of the samples from the posterior distribution. Therefore, `nrep` must not be greater than `n`. In that event `nrep` is set equal to `n`. Currently only implemented if `model = "gev"` or `"gp"` or `"bingp"` or `"pp"`, i.e. *not* implemented if `model = "os"`.

`thresh` A numeric scalar. Extreme value threshold applied to data. Only relevant when `model = "gp"`, `"pp"` or `"bingp"`. Must be supplied when `model = "pp"` or `"bingp"`. If `model = "gp"` and `thresh` is not supplied then `thresh = 0` is used and `data` should contain threshold excesses.

`noy` A numeric scalar. The number of blocks of observations, excluding any missing values. A block is often a year. Only relevant, and must be supplied, if `model = "pp"`.

`use_noy` A logical scalar. Only relevant if model is "pp". If `use_noy = FALSE` then sampling is based on a likelihood in which the number of blocks (years) is set equal to the number of threshold excesses, to reduce posterior dependence between the parameters (Wadsworth *et al.*, 2010). The sampled values are transformed back to the required parameterisation before returning them to the user. If `use_noy = TRUE` then the user's value of `noy` is used in the likelihood.

`npy` A numeric scalar. The mean number of observations per year of data, after excluding any missing values, i.e. the number of non-missing observations divided by total number of years' worth of non-missing data.

The value of `npy` does not affect any calculation in `rpost`, it only affects subsequent extreme value inferences using `predict.evpost`. However, setting `npy` in the call to `rpost` avoids the need to supply `npy` when calling `predict.evpost`. This is likely to be useful only when `model = bingp`. See the documentation of `predict.evpost` for further details.

`ros` A numeric scalar. Only relevant when `model = "os"`. The largest `ros` values in each row of the matrix `data` are used in the analysis.

bin_prior A list specifying the prior for a binomial probability p, created by set_bin_prior. Only relevant if model = "bingp". If this is not supplied then the Jeffreys beta(1/2, 1/2) prior is used.

bin_param A character scalar. The argument param passed to binpost. Only relevant if a user-supplied prior function is set using set_bin_prior.

init_ests A numeric vector. Initial parameter estimates for search for the mode of the posterior distribution.

mult A numeric scalar. The grid of values used to choose the Box-Cox transformation parameter lambda is based on the maximum a posteriori (MAP) estimate +/- mult x estimated posterior standard deviation.

use_phi_map A logical scalar. If trans = "BC" then use_phi_map determines whether the grid of values for phi used to set lambda is centred on the maximum a posterior (MAP) estimate of phi (use_phi_map = TRUE), or on the initial estimate of phi (use_phi_map = FALSE).

weights An optional numeric vector of weights by which to multiply the observations when constructing the log-likelihood. Currently only implemented for model = "gp" or model = "bingp". In the latter case bin_prior$prior must be "bin_beta". weights must have the same length as data.

Details

Generalised Pareto (GP): model = "gp". A model for threshold excesses. Required arguments: n, data and prior. If thresh is supplied then only the values in data that exceed thresh are used and the GP distribution is fitted to the amounts by which those values exceed thresh. If thresh is not supplied then the GP distribution is fitted to all values in data, in effect thresh = 0. See also gp.

Binomial-GP: model = "bingp". The GP model for threshold excesses supplemented by a binomial(length(data), p) model for the number of threshold excesses. See Northrop et al. (2017) for details. Currently, the GP and binomial parameters are assumed to be independent *a priori*.

Generalised extreme value (GEV) model: model = "gev". A model for block maxima. Required arguments: n, data, prior. See also gev.

Point process (PP) model: model = "pp". A model for occurrences of threshold exceedances and threshold excesses. Required arguments: n, data, prior, thresh and noy.

r-largest order statistics (OS) model: model = "os". A model for the largest order statistics within blocks of data. Required arguments: n, data, prior. All the values in data are used unless ros is supplied.

Parameter transformation. The scalar logical arguments (to the function ru) trans and rotate determine, respectively, whether or not Box-Cox transformation is used to reduce asymmetry in the posterior distribution and rotation of parameter axes is used to reduce posterior parameter dependence. The default is trans = "none" and rotate = TRUE.

See the Introducing revdbayes vignette for further details and examples.

Value

An object (list) of class "evpost", which has the same structure as an object of class "ru" returned from ru. In addition this list contains model: The argument model to rpost detailed above.

data: The content depends on model: if model = "gev" then this is the argument data to rpost detailed above, with missing values removed; if model = "gp" then only the values that lie above the threshold are included; if model = "bingp" or model = "pp" then the input data are returned but any value lying below the threshold is set to thresh; if model = "os" then the order statistics used are returned as a single vector.

prior: The argument prior to rpost detailed above.

If nrep is not NULL then this list also contains data_rep, a numerical matrix with nrep rows. Each row contains a replication of the original data data simulated from the posterior predictive distribution. If model = "bingp" or "pp" then the rate of threshold exceedance is part of the inference. Therefore, the number of values in data_rep that lie above the threshold varies between predictive replications (different rows of data_rep). Values below the threshold are left-censored at the threshold, i.e. they are set at the threshold.

If model == "pp" then this list also contains the argument noy to rpost detailed above. If the argument npy was supplied then this list also contains npy.

If model == "gp" or model == "bingp" then this list also contains the argument thresh to rpost detailed above.

If model == "bingp" then this list also contains

bin_sim_vals: An n by 1 numeric matrix of values simulated from the posterior for the binomial probability p

bin_logf: A function returning the log-posterior for p.

bin_logf_args: A list of arguments to bin_logf.

References

Coles, S. G. and Powell, E. A. (1996) Bayesian methods in extreme value modelling: a review and new developments. *Int. Statist. Rev.*, **64**, 119-136.

Northrop, P. J., Attalides, N. and Jonathan, P. (2017) Cross-validatory extreme value threshold selection and uncertainty with application to ocean storm severity. *Journal of the Royal Statistical Society Series C: Applied Statistics*, **66**(1), 93-120. doi:10.1111/rssc.12159

Stephenson, A. (2016) Bayesian Inference for Extreme Value Modelling. In *Extreme Value Modeling and Risk Analysis: Methods and Applications*, edited by D. K. Dey and J. Yan, 257-80. London: Chapman and Hall. doi:10.1201/b19721 value posterior using the evdbayes package.

Wadsworth, J. L., Tawn, J. A. and Jonathan, P. (2010) Accounting for choice of measurement scale in extreme value modeling. *The Annals of Applied Statistics*, **4**(3), 1558-1578. doi:10.1214/10-AOAS333

See Also

set_prior for setting a prior distribution.

rpost_rcpp for faster posterior simulation using the Rcpp package.

plot.evpost, summary.evpost and predict.evpost for the S3 plot, summary and predict methods for objects of class evpost.

ru and ru_rcpp in the rust package for details of the arguments that can be passed to ru and the form of the object returned by rpost.

find_lambda and find_lambda_rcpp in the rust package is used inside rpost to set the Box-Cox transformation parameter lambda when the trans = "BC" argument is given.

Examples

```
# GP model
u <- quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
gpg <- rpost(n = 1000, model = "gp", prior = fp, thresh = u, data = gom)
plot(gpg)

# Binomial-GP model
u <- quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
bp <- set_bin_prior(prior = "jeffreys")
bgpg <- rpost(n = 1000, model = "bingp", prior = fp, thresh = u, data = gom,
              bin_prior = bp)
plot(bgpg, pu_only = TRUE)
plot(bgpg, add_pu = TRUE)

# Setting the same binomial (Jeffreys) prior by hand
beta_prior_fn <- function(p, ab) {
  return(stats::dbeta(p, shape1 = ab[1], shape2 = ab[2], log = TRUE))
}
jeffreys <- set_bin_prior(beta_prior_fn, ab = c(1 / 2, 1 / 2))
bgpg <- rpost(n = 1000, model = "bingp", prior = fp, thresh = u, data = gom,
              bin_prior = jeffreys)
plot(bgpg, pu_only = TRUE)
plot(bgpg, add_pu = TRUE)

# GEV model
mat <- diag(c(10000, 10000, 100))
pn <- set_prior(prior = "norm", model = "gev", mean = c(0, 0, 0), cov = mat)
gevp  <- rpost(n = 1000, model = "gev", prior = pn, data = portpirie)
plot(gevp)

# GEV model, informative prior constructed on the probability scale
pip  <- set_prior(quant = c(85, 88, 95), alpha = c(4, 2.5, 2.25, 0.25),
                  model = "gev", prior = "prob")
ox_post <- rpost(n = 1000, model = "gev", prior = pip, data = oxford)
plot(ox_post)

# PP model
pf <- set_prior(prior = "flat", model = "gev", min_xi = -1)
ppr <- rpost(n = 1000, model = "pp", prior = pf, data = rainfall,
             thresh = 40, noy = 54)
plot(ppr)

# PP model, informative prior constructed on the quantile scale
piq <- set_prior(prob = 10^-(1:3), shape = c(38.9, 7.1, 47),
                 scale = c(1.5, 6.3, 2.6), model = "gev", prior = "quant")
rn_post <- rpost(n = 1000, model = "pp", prior = piq, data = rainfall,
                 thresh = 40, noy = 54)
plot(rn_post)

# OS model
mat <- diag(c(10000, 10000, 100))
pv <- set_prior(prior = "norm", model = "gev", mean = c(0, 0, 0), cov = mat)
osv <- rpost(n = 1000, model = "os", prior = pv, data = venice)
plot(osv)
```

---

rpost_rcpp *Random sampling from extreme value posterior distributions*

Description

Uses the ru_rcpp function in the rust package to simulate from the posterior distribution of an extreme value model.

Usage

```
rpost_rcpp(
  n,
  model = c("gev", "gp", "bingp", "pp", "os"),
  data,
  prior,
  ...,
  nrep = NULL,
  thresh = NULL,
  noy = NULL,
  use_noy = TRUE,
  npy = NULL,
  ros = NULL,
 bin_prior = structure(list(prior = "bin_beta", ab = c(1/2, 1/2), class = "binprior")),
  init_ests = NULL,
  mult = 2,
  use_phi_map = FALSE
)
```

Arguments

n	A numeric scalar. The size of posterior sample required.
model	A character string. Specifies the extreme value model.
data	Sample data, of a format appropriate to the value of model. • "gp". A numeric vector of threshold excesses or raw data. • "bingp". A numeric vector of raw data.

- "gev". A numeric vector of block maxima.
- "pp". A numeric vector of raw data.
- "os". A numeric matrix or data frame. Each row should contain the largest order statistics for a block of data. These need not be ordered: they are sorted inside rpost. If a block contains fewer than dim(as.matrix(data))[2] order statistics then the corresponding row should be padded by NAs. If ros is supplied then only the largest ros values in each row are used. If a vector is supplied then this is converted to a matrix with one column. This is equivalent to using model = "gev".

prior — A list specifying the prior for the parameters of the extreme value model, created by set_prior.

... — Further arguments to be passed to ru_rcpp. Most importantly trans and rotate (see **Details**), and perhaps r, ep, a_algor, b_algor, a_method, b_method, a_control, b_control. May also be used to pass the arguments n_grid and/or ep_bc to find_lambda.

nrep — A numeric scalar. If nrep is not NULL then nrep gives the number of replications of the original dataset simulated from the posterior predictive distribution. Each replication is based on one of the samples from the posterior distribution. Therefore, nrep must not be greater than n. In that event nrep is set equal to n. Currently only implemented if model = "gev" or "gp" or "bingp" or "pp", i.e. *not* implemented if model = "os".

thresh — A numeric scalar. Extreme value threshold applied to data. Only relevant when model = "gp", "pp" or "bingp". Must be supplied when model = "pp" or "bingp". If model = "gp" and thresh is not supplied then thresh = 0 is used and data should contain threshold excesses.

noy — A numeric scalar. The number of blocks of observations, excluding any missing values. A block is often a year. Only relevant, and must be supplied, if model = "pp".

use_noy — A logical scalar. Only relevant if model is "pp". If use_noy = FALSE then sampling is based on a likelihood in which the number of blocks (years) is set equal to the number of threshold excesses, to reduce posterior dependence between the parameters (Wadsworth *et al*., 2010). The sampled values are transformed back to the required parameterisation before returning them to the user. If use_noy = TRUE then the user's value of noy is used in the likelihood.

npy — A numeric scalar. The mean number of observations per year of data, after excluding any missing values, i.e. the number of non-missing observations divided by total number of years' worth of non-missing data.

The value of npy does not affect any calculation in rpost, it only affects subsequent extreme value inferences using predict.evpost. However, setting npy in the call to rpost avoids the need to supply npy when calling predict.evpost. This is likely to be useful only when model = bingp. See the documentation of predict.evpost for further details.

ros — A numeric scalar. Only relevant when model = "os". The largest ros values in each row of the matrix data are used in the analysis.

bin_prior — A list specifying the prior for a binomial probability p, created by set_bin_prior. Only relevant if model = "bingp". If this is not supplied then the Jeffreys beta(1/2, 1/2) prior is used.

init_ests	A numeric vector. Initial parameter estimates for search for the mode of the posterior distribution.
mult	A numeric scalar. The grid of values used to choose the Box-Cox transformation parameter lambda is based on the maximum a posteriori (MAP) estimate +/- mult x estimated posterior standard deviation.
use_phi_map	A logical scalar. If trans = "BC" then use_phi_map determines whether the grid of values for phi used to set lambda is centred on the maximum a posterior (MAP) estimate of phi (use_phi_map = TRUE), or on the initial estimate of phi (use_phi_map = FALSE).

Details

Generalised Pareto (GP): model = "gp". A model for threshold excesses. Required arguments: n, data and prior. If thresh is supplied then only the values in data that exceed thresh are used and the GP distribution is fitted to the amounts by which those values exceed thresh. If thresh is not supplied then the GP distribution is fitted to all values in data, in effect thresh = 0. See also gp.

Binomial-GP: model = "bingp". The GP model for threshold excesses supplemented by a binomial(length(data), p) model for the number of threshold excesses. See Northrop et al. (2017) for details. Currently, the GP and binomial parameters are assumed to be independent *a priori*.

Generalised extreme value (GEV) model: model = "gev". A model for block maxima. Required arguments: n, data, prior. See also gev.

Point process (PP) model: model = "pp". A model for occurrences of threshold exceedances and threshold excesses. Required arguments: n, data, prior, thresh and noy.

r-largest order statistics (OS) model: model = "os". A model for the largest order statistics within blocks of data. Required arguments: n, data, prior. All the values in data are used unless ros is supplied.

Parameter transformation. The scalar logical arguments (to the function ru) trans and rotate determine, respectively, whether or not Box-Cox transformation is used to reduce asymmetry in the posterior distribution and rotation of parameter axes is used to reduce posterior parameter dependence. The default is trans = "none" and rotate = TRUE.

See the Introducing revdbayes vignette for further details and examples.

Value

An object (list) of class "evpost", which has the same structure as an object of class "ru" returned from ru_rcpp. In addition this list contains

model:	The argument model to rpost detailed above.
data:	The content depends on model: if model = "gev" then this is the argument data to rpost detailed above, with missing values removed; if model = "gp" then only the values that lie above the threshold are included; if model = "bingp" or model = "pp" then the input data are returned but any value lying below the threshold is set to thresh; if model = "os" then the order statistics used are returned as a single vector.
prior:	The argument prior to rpost detailed above.

logf_rho_args: A list of arguments to the (transformed) target log-density.

If nrep is not NULL then this list also contains data_rep, a numerical matrix with nrep rows. Each row contains a replication of the original data data simulated from the posterior predictive distribution. If model = "bingp" or "pp" then the rate of threshold exceedance is part of the inference. Therefore, the number of values in data_rep that lie above the threshold varies between predictive replications (different rows of data_rep). Values below the threshold are left-censored at the threshold, i.e. they are set at the threshold.

If model == "pp" then this list also contains the argument noy to rpost detailed above. If the argument npy was supplied then this list also contains npy.

If model == "gp" or model == "bingp" then this list also contains the argument thresh to rpost detailed above.

If model == "bingp" then this list also contains

bin_sim_vals: An n by 1 numeric matrix of values simulated from the posterior for the binomial probability p

bin_logf: A function returning the log-posterior for p.

bin_logf_args: A list of arguments to bin_logf.

References

Coles, S. G. and Powell, E. A. (1996) Bayesian methods in extreme value modelling: a review and new developments. *Int. Statist. Rev.*, **64**, 119-136.

Northrop, P. J., Attalides, N. and Jonathan, P. (2017) Cross-validatory extreme value threshold selection and uncertainty with application to ocean storm severity. *Journal of the Royal Statistical Society Series C: Applied Statistics*, **66**(1), 93-120. doi:10.1111/rssc.12159

Stephenson, A. (2016) Bayesian Inference for Extreme Value Modelling. In *Extreme Value Modeling and Risk Analysis: Methods and Applications*, edited by D. K. Dey and J. Yan, 257-80. London: Chapman and Hall. doi:10.1201/b19721 value posterior using the evdbayes package.

Wadsworth, J. L., Tawn, J. A. and Jonathan, P. (2010) Accounting for choice of measurement scale in extreme value modeling. *The Annals of Applied Statistics*, **4**(3), 1558-1578. doi:10.1214/10-AOAS333

See Also

set_prior for setting a prior distribution.

rpost for posterior simulation without using the Rcpp package.

plot.evpost, summary.evpost and predict.evpost for the S3 plot, summary and predict methods for objects of class evpost.

ru_rcpp in the rust package for details of the arguments that can be passed to ru_rcpp and the form of the object returned by rpost_rcpp.

find_lambda in the rust package is used inside rpost to set the Box-Cox transformation parameter lambda when the trans = "BC" argument is given.

Examples

```
# GP model
u <- quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
gpg <- rpost_rcpp(n = 1000, model = "gp", prior = fp, thresh = u,
                  data = gom)
plot(gpg)

# GP model, user-defined prior (same prior as the previous example)
ptr_gp_flat <- create_prior_xptr("gp_flat")
p_user <- set_prior(prior = ptr_gp_flat, model = "gp", min_xi = -1)
gpg <- rpost_rcpp(n = 1000, model = "gp", prior = p_user, thresh = u,
                  data = gom)
plot(gpg)

# Binomial-GP model
u <- quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
bp <- set_bin_prior(prior = "jeffreys")
bgpg <- rpost_rcpp(n = 1000, model = "bingp", prior = fp, thresh = u,
                   data = gom, bin_prior = bp)
plot(bgpg, pu_only = TRUE)
plot(bgpg, add_pu = TRUE)

# Setting the same binomial (Jeffreys) prior by hand
beta_prior_fn <- function(p, ab) {
  return(stats::dbeta(p, shape1 = ab[1], shape2 = ab[2], log = TRUE))
}
jeffreys <- set_bin_prior(beta_prior_fn, ab = c(1 / 2, 1 / 2))
bgpg <- rpost_rcpp(n = 1000, model = "bingp", prior = fp, thresh = u,
                   data = gom, bin_prior = jeffreys)
plot(bgpg, pu_only = TRUE)
plot(bgpg, add_pu = TRUE)

# GEV model
mat <- diag(c(10000, 10000, 100))
pn <- set_prior(prior = "norm", model = "gev", mean = c(0, 0, 0), cov = mat)
gevp  <- rpost_rcpp(n = 1000, model = "gev", prior = pn, data = portpirie)
plot(gevp)

# GEV model, user-defined prior (same prior as the previous example)
mat <- diag(c(10000, 10000, 100))
ptr_gev_norm <- create_prior_xptr("gev_norm")
pn_u <- set_prior(prior = ptr_gev_norm, model = "gev", mean = c(0, 0, 0),
                  icov = solve(mat))
gevu <- rpost_rcpp(n = 1000, model = "gev", prior = pn_u, data = portpirie)
plot(gevu)

# GEV model, informative prior constructed on the probability scale
pip  <- set_prior(quant = c(85, 88, 95), alpha = c(4, 2.5, 2.25, 0.25),
  model = "gev", prior = "prob")
ox_post <- rpost_rcpp(n = 1000, model = "gev", prior = pip, data = oxford)
```

```
plot(ox_post)

# PP model
pf <- set_prior(prior = "flat", model = "gev", min_xi = -1)
ppr <- rpost_rcpp(n = 1000, model = "pp", prior = pf, data = rainfall,
                  thresh = 40, noy = 54)
plot(ppr)

# PP model, user-defined prior (same prior as the previous example)
ptr_gev_flat <- create_prior_xptr("gev_flat")
pf_u <- set_prior(prior = ptr_gev_flat, model = "gev", min_xi = -1,
                  max_xi = Inf)
ppru <- rpost_rcpp(n = 1000, model = "pp", prior = pf_u, data = rainfall,
                   thresh = 40, noy = 54)
plot(ppru)

# PP model, informative prior constructed on the quantile scale
piq <- set_prior(prob = 10^-(1:3), shape = c(38.9, 7.1, 47),
                 scale = c(1.5, 6.3, 2.6), model = "gev", prior = "quant")
rn_post <- rpost_rcpp(n = 1000, model = "pp", prior = piq, data = rainfall,
                      thresh = 40, noy = 54)
plot(rn_post)

# OS model
mat <- diag(c(10000, 10000, 100))
pv <- set_prior(prior = "norm", model = "gev", mean = c(0, 0, 0), cov = mat)
osv <- rpost_rcpp(n = 1000, model = "os", prior = pv, data = venice)
plot(osv)
```

rprior_prob *Prior simulation of GEV parameters - prior on probability scale*

Description

Simulates from the prior distribution for GEV parameters based on Crowder (1992), in which independent beta priors are specified for ratios of probabilities (which is equivalent to a Dirichlet prior on differences between these probabilities).

Usage

```
rprior_prob(n, quant, alpha, exc = FALSE, lb = NULL, lb_prob = 0.001)
```

Arguments

n	A numeric scalar. The size of sample required.
quant	A numeric vector of length 3. Contains quantiles q_1, q_2, q_3. A prior distribution is placed on the non-exceedance (exc = FALSE) or exceedance (exc = TRUE) probabilities corresponding to these quantiles. The values should *increase* with the index of the vector. If not, the values in quant will be sorted into increasing order without warning.

`alpha` A numeric vector of length 4. Parameters of the Dirichlet distribution for the exceedance probabilities.

`exc` A logical scalar. Let M be the GEV variable, $r_q = P(M \leq q)$, $p_q = P(M > q) = 1 - r_q$ and `quant` = (q_1, q_2, q_3). If `exc` = `FALSE` then a Dirichlet(`alpha`) distribution is placed on $(r_{q_1}, r_{q_2} - r_{q_1}, r_{q_3} - r_{q_2}, 1 - r_{q_3})$, as in Northrop et al. (2017). If `exc` = `TRUE` then a Dirichlet(`alpha`) distribution is placed on $(1 - p_{q_1}, p_{q_1} - p_{q_2}, p_{q_2} - p_{q_3}, p_{q_3})$, where $p_q = P(M > q)$, as in Stephenson (2016).

`lb` A numeric scalar. If this is not `NULL` then the simulation is constrained so that `lb` is an approximate lower bound on the GEV variable. Specifically, only simulated GEV parameter values for which the 100`lb_prob`% quantile is greater than `lb` are retained.

`lb_prob` A numeric scalar. The non-exceedance probability involved in the specification of `lb`. Must be in (0,1). If `lb=NULL` then `lb_prob` is not used.

Details

The simulation is based on the way that the prior is constructed. See Stephenson (1996) the evdbayes user guide or Northrop et al. (2017) Northrop et al. (2017) for details of the construction of the prior. First, differences between probabilities are simulated from a Dirichlet distribution. Then the GEV location, scale and shape parameters that correspond to these quantile values are found, by solving numerically a set of three non-linear equations in which the GEV quantile function evaluated at the simulated probabilities is equated to the quantiles in `quant`. This is reduced to a one-dimensional optimisation over the GEV shape parameter.

Value

An `n` by 3 numeric matrix.

References

Crowder, M. (1992) Bayesian priors based on parameter transformation using the distribution function. *Ann. Inst. Statist. Math.*, **44**(3), 405-416. https://link.springer.com/article/10.1007/BF00050695

Stephenson, A. 2016. Bayesian Inference for Extreme Value Modelling. In *Extreme Value Modeling and Risk Analysis: Methods and Applications*, edited by D. K. Dey and J. Yan, 257-80. London: Chapman and Hall. doi:10.1201/b19721

Northrop, P. J., Attalides, N. and Jonathan, P. (2017) Cross-validatory extreme value threshold selection and uncertainty with application to ocean storm severity. *Journal of the Royal Statistical Society Series C: Applied Statistics*, **66**(1), 93-120. doi:10.1111/rssc.12159

See Also

`rpost` and `rpost_rcpp` for sampling from an extreme value posterior distribution.

Examples

```
quant <- c(85, 88, 95)
alpha <- c(4, 2.5, 2.25, 0.25)
```

```
x <- rprior_prob(n = 1000, quant = quant, alpha = alpha, exc = TRUE)
x <- rprior_prob(n = 1000, quant = quant, alpha = alpha, exc = TRUE, lb = 0)
```

rprior_quant *Prior simulation of GEV parameters - prior on quantile scale*

Description

Simulates from the prior distribution for GEV parameters proposed in Coles and Tawn (1996), based on independent gamma priors for differences between quantiles.

Usage

```
rprior_quant(n, prob, shape, scale, lb = NULL, lb_prob = 0.001)
```

Arguments

n A numeric scalar. The size of sample required.

prob A numeric vector of length 3. Exceedance probabilities corresponding to the quantiles used to specify the prior distribution. The values should *decrease* with the index of the vector. If not, the values in prob will be sorted into decreasing order without warning.

shape A numeric vector of length 3. Respective shape parameters of the gamma priors for the quantile differences.

scale A numeric vector of length 3. Respective scale parameters of the gamma priors for the quantile differences.

lb A numeric scalar. If this is not NULL then the simulation is constrained so that lb is an approximate lower bound on the GEV variable. Specifically, only simulated GEV parameter values for which the 100lb_prob% quantile is greater than lb are retained.

lb_prob A numeric scalar. The non-exceedance probability involved in the specification of lb. Must be in (0,1). If lb=NULL then lb_prob is not used.

Details

The simulation is based on the way that the prior is constructed. See Coles and Tawn (1996), Stephenson (2016) or the evdbayes user guide for details of the construction of the prior. First, the quantile differences are simulated from the specified gamma distributions. Then the simulated quantiles are calculated. Then the GEV location, scale and shape parameters that give these quantile values are found, by solving numerically a set of three non-linear equations in which the GEV quantile function evaluated at the values in prob is equated to the simulated quantiles. This is reduced to a one-dimensional optimisation over the GEV shape parameter.

Value

An n by 3 numeric matrix.

References

Coles, S. G. and Tawn, J. A. (1996) A Bayesian analysis of extreme rainfall data. *Appl. Statist.*, **45**, 463-478.

Stephenson, A. 2016. Bayesian Inference for Extreme Value Modelling. In *Extreme Value Modeling and Risk Analysis: Methods and Applications*, edited by D. K. Dey and J. Yan, 257-80. London: Chapman and Hall. doi:10.1201/b19721

See Also

`rpost` and `rpost_rcpp` for sampling from an extreme value posterior distribution.

Examples

```
pr <- 10 ^ -(1:3)
sh <- c(38.9, 7.1, 47)
sc <- c(1.5, 6.3, 2.6)
x <- rprior_quant(n = 1000, prob = pr, shape = sh, scale = sc)
x <- rprior_quant(n = 1000, prob = pr, shape = sh, scale = sc, lb = 0)
```

`set_bin_prior` *Construction of a prior distribution for a binomial probability p*

Description

Constructs a prior distribution for use as the argument `bin_prior` in `rpost` or in `binpost`. The user can choose from a list of in-built priors or specify their own prior function, returning the **log** of the prior density, using an R function and arguments for hyperparameters.

Usage

```
set_bin_prior(
  prior = c("jeffreys", "laplace", "haldane", "beta", "mdi", "northrop"),
  ...
)
```

Arguments

`prior` Either

- An R function that returns the value of the log of the prior density (see **Examples**), or
- A character string giving the name of the prior for p. See **Details** for a list of priors available.

`...` Further arguments to be passed to the user-supplied or in-built prior function. For the latter this is only relevant if `prior = "beta"`, when `ab` can be passed. See **Details**.

Details

Binomial priors. The names of the binomial priors set using `bin_prior` are:

- `"jeffreys"`: the *Jeffreys* beta(1/2, 1/2) prior.
- `"laplace"`: the *Bayes-Laplace* beta(1, 1) prior.
- `"haldane"`: the *Haldane* beta(0, 0) prior.
- `"beta"`: a beta(α, β) prior. The argument `ab` is a vector containing `c`(α, β). The default is `ab = c(1, 1)`.
- `"mdi"`: the MDI prior $\pi(p) = 1.6186 p^p (1-p)^{1-p}$, for $0 < p < 1$.
- `"northrop"`: the improper prior $\pi(p) = \{-\ln(1-p)\}^{-1}(1-p)^{-1}$, for $0 < p < 1$.

Apart from the last two priors these are all beta distributions.

Value

A list of class `"binprior"`. The first component is the name of the input prior. Apart from the MDI prior this will be "beta", in which case the other component of the list is a vector of length two giving the corresponding values of the beta parameters.

See Also

`binpost` for sampling from a binomial posterior distribution.

Examples

```
bp <- set_bin_prior(prior = "jeffreys")

# Setting the Jeffreys prior by hand
beta_prior_fn <- function(p, ab) {
  return(stats::dbeta(p, shape1 = ab[1], shape2 = ab[2], log = TRUE))
}
jeffreys <- set_bin_prior(beta_prior_fn, ab = c(1 / 2, 1 / 2))
```

`set_prior` *Construction of prior distributions for extreme value model parameters*

Description

Constructs a prior distribution for use as the argument `prior` in `rpost` and `rpost_rcpp`. The user can either specify their own prior function, returning the **log** of the prior density, (using an R function or an external pointer to a compiled C++ function) and arguments for hyperparameters or choose from a list of in-built model-specific priors. Note that the arguments `model = "gev"`, `model = "pp"` and `model =="os"` are equivalent because a prior is specified is the GEV parameterisation in each of these cases. Note also that for `model = "pp"` the prior GEV parameterisation relates to the value of noy subsequently supplied to `rpost` or `rpost_rcpp`. The argument `model` is used for consistency with `rpost`.

Usage

```
set_prior(
  prior = c("norm", "loglognorm", "mdi", "flat", "flatflat", "jeffreys", "beta", "prob",
    "quant"),
  model = c("gev", "gp", "pp", "os"),
  ...
)
```

Arguments

`prior` Either

- An R function, or a pointer to a user-supplied compiled C++ function, that returns the value of the log of the prior density (see **Examples**), or
- A character string giving the name of the prior. See **Details** for a list of priors available for each model.

`model` A character string. If `prior` is a character string then `model` gives the extreme value model to be used. Using either `model = "gev"`, `model = "pp"` or `model = "os"` will result in the same (GEV) parameterisation. If `prior` is a function then the value of `model` is stored so that in the subsequent call to `rpost`, consistency of the prior and extreme value model parameterisations can be checked.

`...` Further arguments to be passed to the user-supplied or in-built prior function. For details of the latter see **Details** and/or the relevant underlying function: `gp_norm`, `gp_mdi`, `gp_flat`, `gp_flatflat`, `gp_jeffreys`, `gp_beta`, `gev_norm`, `gev_loglognorm`, `gev_mdi`, `gev_flat`, `gev_flatflat`, `gev_beta`, `gev_prob`, `gev_quant`. All these priors have the arguments `min_xi` (prior lower bound on ξ) and `max_xi` (prior upper bound on ξ).

Details

Of the in-built named priors available in revdbayes only those specified using `prior = "prob"` (`gev_prob`), `prior = "quant"` (`gev_quant`) `prior = "norm"` (`gev_norm`) or `prior = "loglognorm"` (`gev_loglognorm`) are proper. If `model = "gev"` these priors are equivalent to priors available in the evdbayes package, namely `prior.prob`, `prior.quant`, `prior.norm` and `prior.loglognorm`.

The other in-built priors are improper, that is, the integral of the prior function over its support is not finite. Such priors do not necessarily result in a proper posterior distribution. Northrop and Attalides (2016) consider the issue of posterior propriety in Bayesian extreme value analyses. In most of improper priors below the prior for the scale parameter σ is taken to be $1/\sigma$, i.e. a flat prior for $\log \sigma$. Here we denote the scale parameter of the GP distribution by σ, whereas we use σ_u in the revdbayes vignette.

For all in-built priors the arguments `min_xi` and `max_xi` may be supplied by the user. The prior density is set to zero for any value of the shape parameter ξ that is outside (`min_xi`, `max_xi`). This will override the default values of `min_xi` and `max_xi` in the named priors detailed above.

Extreme value priors. It is typical to use either `prior = "prob"` (`gev_prob`) or `prior = "quant"` (`gev_quant`) to set an informative prior and one of the other prior (or a user-supplied function) otherwise. The names of the in-built extreme value priors set using `prior` and details of hyperparameters are:

- "prob". A prior for GEV parameters (μ, σ, ξ) based on Crowder (1992). See gev_prob for details. See also Northrop et al. (2017) and Stephenson (2016).
- "quant". A prior for GEV parameters (μ, σ, ξ) based on Coles and Tawn (1996). See gev_quant for details.
- "norm".

  For model = "gp": $(\log \sigma, \xi)$, is bivariate normal with mean mean (a numeric vector of length 2) and covariance matrix cov (a symmetric positive definite 2 by 2 matrix).

  For model = "gev": $(\mu, \log \sigma, \xi)$, is trivariate normal with mean mean (a numeric vector of length 3) and covariance matrix cov (a symmetric positive definite 3 by 3 matrix).
- "loglognorm". For model = "gev" only: $(\log \mu, \log \sigma, \xi)$, is trivariate normal with mean mean (a numeric vector of length 3) and covariance matrix cov (a symmetric positive definite 3 by 3 matrix).
- "mdi".

  For model = "gp": (an extended version of) the maximal data information (MDI) prior, that is,

  $$\pi(\sigma, \xi) = \sigma^{-1} \exp[-a(\xi + 1)], \text{ for } \sigma > 0, \xi \geq -1, a \geq 0.$$

  The value of a is set using the argument a. The default value is $a = 1$, which gives the MDI prior.

  For model = "gev": (an extended version of) the maximal data information (MDI) prior, that is,

  $$\pi(\mu, \sigma, \xi) = \sigma^{-1} \exp[-a(\xi + 1)], \text{ for } \sigma > 0, \xi \geq -1, a \geq 0.$$

  The value of a is set using the argument a. The default value is $a = \gamma$, where $\gamma = 0.57721$ is Euler's constant, which gives the MDI prior.

  For each of these cases ξ must be is bounded below *a priori* for the posterior to be proper (Northrop and Attalides, 2016). An argument for the bound $\xi \geq -1$ is that for $\xi < -1$ the GP (GEV) likelihood is unbounded above as $-\sigma/\xi$ $(\mu - \sigma/\xi)$) approaches the sample maximum. In maximum likelihood estimation of GP parameters (Grimshaw, 1993) and GEV parameters a local maximum of the likelihood is sought on the region $\sigma > 0, \xi \geq -1$.
- "flat".

  For model = "gp": a flat prior for ξ (and for $\log \sigma$):

  $$\pi(\sigma, \xi) = \sigma^{-1}, \text{ for } \sigma > 0.$$

  For model = "gev": a flat prior for ξ (and for μ and $\log \sigma$):

  $$\pi(\mu, \sigma, \xi) = \sigma^{-1}, \text{ for } \sigma > 0.$$
- "flatflat".

  For model = "gp": flat priors for σ and ξ:

  $$\pi(\sigma, \xi) = 1, \text{ for } \sigma > 0.$$

  For model = "gev": flat priors for μ, σ and ξ:

  $$\pi(\mu, \sigma, \xi) = 1, \text{ for } \sigma > 0.$$

  Therefore, the posterior is proportional to the likelihood.

- "jeffreys". For model = "gp" only: the Jeffreys prior (Castellanos and Cabras, 2007):

  $$\pi(\sigma, \xi) = \sigma^{-1}(1+\xi)^{-1}(1+2\xi)^{-1/2}, \text{ for } \sigma > 0, \xi > -1/2.$$

  In the GEV case the Jeffreys prior doesn't yield a proper posterior for any sample size. See Northrop and Attalides (2016) for details.

- "beta". For model = "gp": a beta-type(p, q) prior is used for xi on the interval (min_xi, max_xi):

  $$\pi(\sigma, \xi) = \sigma^{-1}(\xi - \min_\xi)^{p-1}(\max_\xi - \xi)^{q-1}, \text{ for } \min_\xi < \xi < \max_\xi.$$

  For model = "gev": similarly ...

  $$\pi(\mu, \sigma, \xi) = \sigma^{-1}(\xi - \min_\xi)^{p-1}(\max_\xi - \xi)^{q-1}, \text{ for } \min_\xi < \xi < \max_\xi.$$

  The argument pq is a vector containing c(p,q). The default settings for this prior are p = 6, q = 9 and min_xi = -1/2, max_xi = 1/2, which corresponds to the prior for ξ proposed in Martins and Stedinger (2000, 2001).

Value

A list with class "evprior". The first component is the input prior, i.e. either the name of the prior or a user-supplied function. The remaining components contain the numeric values of any hyperparameters in the prior.

References

Castellanos, E. M. and Cabras, S. (2007) A default Bayesian procedure for the generalized Pareto distribution. *Journal of Statistical Planning and Inference* **137(2)**, 473-483. doi:10.1016/j.jspi.2006.01.006.

Coles, S. G. and Tawn, J. A. (1996) A Bayesian analysis of extreme rainfall data. *Appl. Statist.*, **45**, 463-478.

Crowder, M. (1992) Bayesian priors based on parameter transformation using the distribution function *Ann. Inst. Statist. Math.*, **44**, 405-416. https://link.springer.com/article/10.1007/BF00050695.

Grimshaw, S. D. (1993) Computing Maximum Likelihood Estimates for the Generalized Pareto Distribution. *Technometrics*, **35(2)**, 185-191. doi:10.1080/00401706.1993.10485040.

Hosking, J. R. M. and Wallis, J. R. (1987) Parameter and Quantile Estimation for the Generalized Pareto Distribution. *Technometrics*, **29(3)**, 339-349. doi:10.2307/1269343.

Martins, E. S. and Stedinger, J. R. (2000) Generalized maximum likelihood generalized extreme value quantile estimators for hydrologic data, *Water Resources Research*, **36(3)**, 737-744. doi:10.1029/1999WR900330.

Martins, E. S. and Stedinger, J. R. (2001) Generalized maximum likelihood Pareto-Poisson estimators for partial duration series, *Water Resources Research*, **37(10)**, 2551-2557. doi:10.1029/2001WR000367.

Northrop, P.J. and Attalides, N. (2016) Posterior propriety in Bayesian extreme value analyses using reference priors *Statistica Sinica*, **26**(2), 721–743 doi:10.5705/ss.2014.034.

Northrop, P. J., Attalides, N. and Jonathan, P. (2017) Cross-validatory extreme value threshold selection and uncertainty with application to ocean storm severity. *Journal of the Royal Statistical Society Series C: Applied Statistics*, **66**(1), 93-120. doi:10.1111/rssc.12159

Stephenson, A. (2016) Bayesian inference for extreme value modelling. In *Extreme Value Modeling and Risk Analysis: Methods and Applications* (eds D. K. Dey and J. Yan), 257-280, Chapman and Hall, London. doi:10.1201/b19721.

See Also

rpost and rpost_rcpp for sampling from an extreme value posterior distribution.

create_prior_xptr for creating an external pointer to a C++ function to evaluate the log-prior density.

rprior_prob and rprior_quant for sampling from informative prior distributions for GEV parameters.

gp_norm, gp_mdi, gp_flat, gp_flatflat, gp_jeffreys, gp_beta to see the arguments for priors for GP parameters.

gev_norm, gev_loglognorm, gev_mdi, gev_flat, gev_flatflat, gev_beta, gev_prob, gev_quant to see the arguments for priors for GEV parameters.

Examples

```
# Normal prior for GEV parameters (mu, log(sigma), xi).
mat <- diag(c(10000, 10000, 100))
pn <- set_prior(prior = "norm", model = "gev", mean = c(0,0,0), cov = mat)
pn

# Prior for GP parameters with flat prior for xi on (-1, infinity).
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
fp

# A user-defined prior (see the vignette for details).
u_prior_fn <- function(x, ab) {
  if (x[1] <= 0 | x[2] <= -1 | x[2] >= 1) {
    return(-Inf)
  }
  return(-log(x[1]) + (ab[1] - 1) * log(1 + x[2]) +
         (ab[2] - 1) * log(1 - x[2]))
}
up <- set_prior(prior = u_prior_fn, ab = c(2, 2), model = "gp")

# A user-defined prior using a pointer to a C++ function
ptr_gp_flat <- create_prior_xptr("gp_flat")
u_prior_ptr <- set_prior(prior = ptr_gp_flat, model = "gp")
```

summary.evpost *Summarizing an evpost object*

Description

summary method for class "evpost"

Usage

```
## S3 method for class 'evpost'
summary(object, add_pu = FALSE, ...)
```

Arguments

object	An object of class "evpost", a result of a call to rpost or rpost_rcpp.
add_pu	Includes in the summary of the simulated values the threshold exceedance probability p. Only relevant when model == "bingp" was used in the call to rpost or rpost_rcpp.
...	Additional arguments passed on to print.

Value

Prints

- information about the ratio-of-uniforms bounding box, i.e. object$box
- an estimate of the probability of acceptance, i.e. object$pa
- a summary of the simulated values, via summary(object$sim_vals)

See Also

ru or ru_rcpp for descriptions of object$sim_vals and object$box.

plot.evpost for a diagnostic plot.

Examples

```
# GP posterior
u <- stats::quantile(gom, probs = 0.65)
fp <- set_prior(prior = "flat", model = "gp", min_xi = -1)
gpg <- rpost_rcpp(n = 1000, model = "gp", prior = fp, thresh = u,
                  data = gom)
summary(gpg)
```

venice *Largest Sea Levels in Venice*

Description

The `venice` data frame has 51 rows and 10 columns. The jth column contains the jth largest sea levels in Venice, for the years 1931-1981. Only the largest six measurements are available for the year 1935; the corresponding row contains four missing values. The years for each set of measurements are given as row names.

Usage

```
venice
```

Format

A data frame with 51 rows and 10 columns.

Source

Smith, R. L. (1986) Extreme value theory based on the *r* largest annual events. *Journal of Hydrology*, **86**, 27-43. doi:10.1016/00221694(86)900041

References

Coles, S. G. (2001) *An Introduction to Statistical Modelling of Extreme Values*. London: Springer. doi:10.1007/9781447136750

wbinpost *Random sampling from a binomial posterior distribution, using weights*

Description

Samples from the posterior distribution of the probability p of a binomial distribution. User-supplied weights are applied to each observation when constructing the log-likelihood.

Usage

```
wbinpost(n, prior, ds_bin)
```

Arguments

`n` A numeric scalar. The size of posterior sample required.

`prior` A function to evaluate the prior, created by `set_bin_prior`. `prior$prior` must be `"bin_beta"`.

`ds_bin` A numeric list. Sufficient statistics for inference about the binomial probability p. Contains

- `sf` : a logical vector of success (`TRUE`) and failure (`FALSE`) indicators.
- `w` : a numeric vector of length `length(sf)` containing the values by which to multiply the observations when constructing the log-likelihood.

Details

For `prior$prior == "bin_beta"` the posterior for p is a beta distribution so `rbeta` is used to sample from the posterior.

Value

An object (list) of class `"binpost"` with components

`bin_sim_vals:` An `n` by 1 numeric matrix of values simulated from the posterior for the binomial probability p

`bin_logf:` A function returning the log-posterior for p.

`bin_logf_args:` A list of arguments to `bin_logf`.

See Also

`set_bin_prior` for setting a prior distribution for the binomial probability p.

Examples

```
u <- quantile(gom, probs = 0.65)
ds_bin <- list(sf = gom > u, w = rep(1, length(gom)))
bp <- set_bin_prior(prior = "jeffreys")
temp <- wbinpost(n = 1000, prior = bp, ds_bin = ds_bin)
graphics::hist(temp$bin_sim_vals, prob = TRUE)
```

Index