Package 'dgumbel'

October 13, 2022

Type Package

Title The Gumbel Distribution Functions and Gradients

Version 1.0.1

Date 2020-04-07

Maintainer Berent Ånund Strømnes Lunde <lundeberent@gmail.com>

Description Gumbel distribution functions (De Haan L. (2007) <doi:10.1007/0-387-34471-3>) implemented with the techniques of automatic differentiation (Griewank A. (2008) <isbn:978-0-89871-659-7>). With this tool, a user should be able to quickly model extreme events for which the Gumbel distribution is the domain of attraction. The package makes available the density function, the distribution function the quantile function and a random generating function. In addition, it supports gradient functions. The package combines 'Adept' (C++ templated automatic differentiation) (Hogan R. (2017) <doi:10.5281/zenodo.1004730>) and 'Eigen' (templated matrix-vector library) for fast computations of both objective functions and exact gradients. It relies on 'RcppEigen' for easy access to 'Eigen' and bindings to R.

License GPL (>= 2)

URL https://github.com/blunde1/dgumbel

BugReports https://github.com/blunde1/dgumbel/issues

Encoding UTF-8

Imports Rcpp (>= 1.0.2)

LinkingTo Rcpp, RcppEigen

RoxygenNote 6.1.1

NeedsCompilation yes

Author Berent Ånund Strømnes Lunde [aut, cre, cph],
Robin Hogan [ctb] (Author of included Adept library),
The University of Reading [cph] (Copyright holder of included Adept library)

Repository CRAN

Date/Publication 2020-04-16 21:00:03 UTC

R topics documented:

`gumbel` *The Gumbel Distribution and Derivatives*

Description

Density function, distribution function, quantile function and random generation, and their gradient functions for the Gumbel distribution with location and scale parameters.

Usage

```
dgumbel(x, location=0, scale=1, log = FALSE, grad=FALSE)
pgumbel(q, location=0, scale=1, lower.tail = TRUE, log.p = FALSE, grad=FALSE)
qgumbel(p, location=0, scale=1, lower.tail = TRUE, grad=FALSE)
rgumbel(n, location=0, scale=1)
```

Arguments

`x, q`	Vector of quantiles.
`p`	Vector of probabilities.
`n`	Number of observations.
`location, scale`	Location and scale parameters.
`log, log.p`	Logical; if `TRUE`, probabilities p are given as log(p).
`lower.tail`	Logical; if `TRUE` (default), probabilities are P[X <= x], otherwise, P[X > x]
`grad`	Logical; if `TRUE`, the gradient w.r.t. parameters location and scale is given instead of function value.

Details

The Gumbel distribution function with parameters `location` $= a$ and `scale` $= b$ is

$$G(z) = \exp\left\{-\exp\left[-\left(\frac{z-a}{b}\right)\right]\right\}$$

for all real z, where $b > 0$. Gradients are exact numerical derivatives implemented using automatic differentiation. `dgumbel` builds on the `Eigen` linear algebra library, `Adept` for automatic differentiation and `RcppEigen` for bindings to `R` and loading `Eigen`.

Value

`dgumbel` gives the density function, `pgumbel` gives the distribution function, `qgumbel` gives the quantile function, and `rgumbel` generates random deviates. If `grad=TRUE` is supplied, then the gradient is returned instead of the objective function.

Examples

```
dgumbel(-1:2, -1, 0.5)
pgumbel(-1:2, -1, 0.5)
qgumbel(seq(0.9, 0.6, -0.1), 2, 0.5)
rgumbel(6, -1, 0.5)
p <- (1:9)/10
pgumbel(qgumbel(p, -1, 2), -1, 2)
## [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9

## Random number generation
loc = .5
scale = 3.2
n <- 1000
x <- rgumbel(n, loc, scale)

## The density
hist(x, freq=FALSE)
xs <- sort(x)
fx <- dgumbel(xs, loc, scale)
points(xs,fx, type="l", col=2, lwd=2)

## The distribution
edf <- sapply(xs, function(x){sum(xs<=x)/n})
plot(xs, edf)
Fx <- pgumbel(xs, loc, scale)
points(xs, Fx, type="l", col=2, lwd=2)

## The quantile function
q <- qgumbel(0.6, loc, scale)
polygon(c(xs[xs <= q], q), c(Fx[xs<=q], 0), col=3)

## Negative log likelihood: Objective and gradient
nll <- function(par, data) -sum(dgumbel(data, par[1], par[2], log=TRUE))
dnll <- function(par, data) -rowSums(dgumbel(data, par[1], par[2], log=TRUE, grad=TRUE))

## Parameter estimation
par_start <- c(3,1)
opt <- nlminb(par_start, objective=nll, gradient=dnll, data=x, control = list(trace=5))
opt$convergence
opt$par
```

Index