Package 'evgam'

August 26, 2020

Type Package

Title Generalised Additive Extreme Value Models

Version 0.1.4

Date 2020-08-18

Author Ben Youngman

Maintainer Ben Youngman <b.youngman@exeter.ac.uk>

Description Methods for fitting various extreme value distributions with parameters of generalised additive model (GAM) form are provided. For details of distributions see Coles, S.G. (2001) <doi:10.1007/978-1-4471-3675-0>, GAMs see Wood, S.N. (2017) <doi:10.1201/9781315370279>, and the fitting approach see Wood, S.N., Pya, N. & Safken, B. (2016) <doi:10.1080/01621459.2016.1180986>.

License GPL-3

Imports Rcpp (>= 1.0.4.6), mgcv

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 3.5.0)

RoxygenNote 7.1.0

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-08-26 16:11:08 UTC

R topics documented:

`colplot` *Scatter plot, with variable-based point colours*

Description

Scatter plot, with variable-based point colours

Usage

```
colplot(
  x,
  y,
  z,
  n = 20,
  z.lim = NULL,
  breaks = NULL,
  palette = heat.colors,
  rev = TRUE,
  pch = 21,
  add = FALSE,
  ...,
  legend = FALSE,
  n.legend = 6,
  legend.pretty = TRUE,
  legend.plot = TRUE,
  legend.x,
  legend.y = NULL,
  legend.horiz = FALSE,
  legend.bg = par("bg")
)
```

Arguments

x	a vector of x coordinates
y	a vector of y coordinates
z	a variable for defining colours

n	an integer giving the number of colour levels, supplied to pretty
z.lim	xxx
breaks	a vector or breaks for defining color intervals; defaults to NULL, so pretty and n are used on z
palette	a function for the color palette, or colors between breaks; defaults to heat.colors
rev	logical: should the palette be reversed? Defaults to TRUE
pch	an integer giving the plotting character, supplied to plot
add	should this be added to an existing plot? Defaults to FALSE
...	other arguments passed to plot
legend	should a legend be added? Defaults to codeFALSE
n.legend	an integer giving the approximate number of legend entries; defaults to 6
legend.pretty	logical: should the legend values produced by \[base]pretty? Othewrwise they are exact. Defaults to TRUE
legend.plot	passed to legend's plot argument
legend.x	passed to legend's x argument
legend.y	passed to legend's y argument
legend.horiz	passed to legend's horiz argument
legend.bg	passed to legend's bg argument

Value

A plot

Examples

```
x <- runif(50)
y <- runif(50)
colplot(x, y, x * y)
colplot(x, y, x * y, legend=TRUE, legend.x="bottomleft")
colplot(x, y, x * y, legend=TRUE, legend.pretty=FALSE, n.legend=10,
  legend.x="bottomleft", legend.horiz=TRUE)
```

COprcp *Colorado daily precipitation accumulations*

Description

Three objects: 1) COprcp, a 404,326-row data frame with columns date, prcp and meta_row; 2) COprcp_meta, a 64-row data frame, with meta data for 64 stations. 3) COelev, a list of elevation for the domain at 0.02 x 0.02 degree resolution. Precipitation amounts are only given for April to October in the years 1990 - 2019. The domain has a longitude range of [-106, -104] and a latitude range [37, 41]. These choices reflect the analysis of Cooley et al. (2007).

Usage

```
data(COprcp) # loads all three objects
```

Format

A data frame with 2383452 rows and 8 variables

The variables are as follows:

date date of observation

prcp daily rainfall accumulation in mm

meta_row an identifier for the row in COprcp_meta; see 'Examples'

lon longitude of station

lat latitude of station

elev elevation of station in metres

id GHCDN identifier

References

Cooley, D., Nychka, D., & Naveau, P. (2007). Bayesian spatial modeling of extreme precipitation return levels. Journal of the American Statistical Association, 102(479), 824-840.

Examples

```
library(evgam)
data(COprcp)

brks <- pretty(COelev$z, 50)
image(COelev, breaks=brks, col=rev(heat.colors(length(brks[-1]))))
colplot(COprcp_meta$lon, COprcp_meta$lat, COprcp_meta$elev, breaks=brks, add=TRUE)
```

---

dfbind *Bind a list a data frames*

Description

Bind a list a data frames

Usage

```
dfbind(x)
```

Arguments

x	a list of data frames

Value

A data frame

See Also

rbind

Examples

```
z <- list(data.frame(x=1, y=1), data.frame(x=2, y=2))
dfbind(z)
```

---

evgam *Fitting generalised additive extreme-value family models*

Description

Function evgam fits generalised additive extreme-value models. It allows the fitting of various extreme-value models, including the generalised extreme value and Pareto distributions. It can also perform quantile regression via the asymmetric Laplace dsitribution.

Usage

```
evgam(
  formula,
  data,
  family = "gev",
  correctV = TRUE,
  rho0 = 0,
  inits = NULL,
  outer = "bfgs",
  control = NULL,
  removeData = FALSE,
  trace = 0,
  knots = NULL,
  maxdata = 1e+20,
  maxspline = 1e+20,
  compact = FALSE,
  ald.args = list(),
  exi.args = list(),
  pp.args = list(),
  sandwich.args = list()
)
```

Arguments

formula	a list of formulae for location, scale and shape parameters, as in gam
data	a data frame
family	a character string giving the type of family to be fitted; defaults to "gev"
correctV	logicial: should the variance-covariance matrix include smoothing parameter uncertainty? Defaults to TRUE
rho0	a scalar or vector of initial log smoothing parameter values; a scalar will be repeated if there are multiple smoothing terms
inits	a vector or list giving initial values for constant basis coefficients; if a list, a grid is formed using expand.grid, and the 'best' used; defaults to NULL, so initial values are automatically found
outer	a character string specifying the outer optimiser is full "Newton", "BFGS" or uses finite differences, "FD"; defaults to "BFGS"
control	a list of lists of control parameters to pass to inner and outer optimisers; defaults to evgam.control()
removeData	logical: should data be removed from evgam object? Defaults to FALSE
trace	an integer specifying the amount of information supplied about fitting, with -1 suppressing all output; defaults to 0
knots	passed to s; defaults to NULL
maxdata	an integer specifying the maximum number of data rows. data is sampled if its number of rows exceeds maxdata; defaults to 1e20
maxspline	an integer specifying the maximum number of data rows used for spline construction; defaults to 1e20
compact	logical: should duplicated data rows be compacted? Defaults to FALSE
ald.args	a list of arguments for family="ald"; see Details
exi.args	a list of arguments for family="exi"; see Details
pp.args	a list of arguments for family="pp"; see Details
sandwich.args	a list of arguments for sandwich adjustment; see Details

Details

The following families are currently available: "ald", the asymmetric Laplace distribution, primarily intended for quantile regression, as in Yu & Moyeed (2001); "gev" (default), the generalised extreme valued distribution; "exp", the exponential distribution; "gpd", the generalised Pareto distribution; "gauss", the Gaussian distribution; "pp", the point process model for extremes, implemented through r-largest order statistics; "weibull", the Weibull distribution; "exi", estimation if the extremal index, as in Schlather & Tawn (2003).

Arguments for the asymmetric Laplace distribution are given by ald.args. A scalar tau defines the quantile sought, which has no default. The scalar C specifies the curvature parameter of Oh et al. (2011).

Arguments for extremal index estimation are given by exi.args. A character string id specifies the variable in dataover which an nexi (default 2) running max. has been taken. The link is specified as a character string, which is one of "logistic", "probit", "cloglog"; defaults to "logistic".

Arguments for the point process model are given by `pp.args`. An integer `r` specifies the number of order statistics from which the model will be estimated. If `r = -1`, all `data` will be used. The character string `id` specifies the variable in `data` over which the point process isn't integrated; e.g. if a map of parameter estimates related to extremes over time is sought, integration isn't over locations. The scalar `nper` number of data per period of interest; scalar or integer vector `ny` specifies the number of periods; if `length(ny) > 1` then `names(ny)` must ne supplied and must match to every unique `id`. logical `correctny` specifies whether `ny` is corrected to adjust proportionally for data missingness.

Arguments for the sandwich adjustment are given by `sandwich.args`. A character string `id` can be supplied to the list, which identifies the name of the variable in `data` such that independence will be assumed between its values. The `method` for the adjustment is supplied as `"magnitude"` (default) or `"curvature"`; see Chandler & Bate (2007) for their definitions.

Value

An object of class `evgam`

References

Chandler, R. E., & Bate, S. (2007). Inference for clustered data using the independence loglikelihood. Biometrika, 94(1), 167-183.

Oh, H. S., Lee, T. C., & Nychka, D. W. (2011). Fast nonparametric quantile regression with arbitrary smoothing methods. Journal of Computational and Graphical Statistics, 20(2), 510-526.

Schlather, M., & Tawn, J. A. (2003). A dependence measure for multivariate and spatial extreme values: Properties and inference. Biometrika, 90(1), 139-156.

Wood, S. N., Pya, N., & Safken, B. (2016). Smoothing parameter and model selection for general smooth models. Journal of the American Statistical Association, 111(516), 1548-1563.

Yu, K., & Moyeed, R. A. (2001). Bayesian quantile regression. Statistics & Probability Letters, 54(4), 437-447.

See Also

predict.evgam

Examples

```
data(fremantle)
fmla_gev <- list(SeaLevel ~ s(Year, k=5, bs="cr"), ~ 1, ~ 1)
m_gev <- evgam(fmla_gev, fremantle, family = "gev")



data(COprcp)

## fit generalised Pareto distribution to excesses on 20mm

COprcp <- cbind(COprcp, COprcp_meta[COprcp$meta_row,])
threshold <- 20
```

```
COprcp$excess <- COprcp$prcp - threshold
COprcp_gpd <- subset(COprcp, excess > 0)
fmla_gpd <- list(excess ~ s(lon, lat, k=12) + s(elev, k=5, bs="cr"), ~ 1)
m_gpd <- evgam(fmla_gpd, data=COprcp_gpd, family="gpd")

## fit generalised extreme value distribution to annual maxima

COprcp$year <- format(COprcp$date, "%Y")
COprcp_gev <- aggregate(prcp ~ year + meta_row, COprcp, max)
COprcp_gev <- cbind(COprcp_gev, COprcp_meta[COprcp_gev$meta_row,])
fmla_gev2 <- list(prcp ~ s(lon, lat, k=30) + s(elev, bs="cr"), ~ s(lon, lat, k=20), ~ 1)
m_gev2 <- evgam(fmla_gev2, data=COprcp_gev, family="gev")
summary(m_gev2)
plot(m_gev2)
predict(m_gev2, newdata=COprcp_meta, type="response")

## fit point process model using r-largest order statistics

# we have `ny=30' years' data and use top 45 order statistics
pp_args <- list(id="id", ny=30, r=45)
m_pp <- evgam(fmla_gev2, COprcp, family="pp", pp.args=pp_args)

## estimate 0.98 quantile using asymmetric Laplace distribution

fmla_ald <- prcp ~ s(lon, lat, k=15) + s(elev, bs="cr")
m_ald <- evgam(fmla_ald, COprcp, family="ald", ald.args=list(tau=.98))
```

`extremal` *Estimate extremal index using 'intervals' method*

Description

Estimate extremal index using 'intervals' method

Usage

```
extremal(x, y = NULL)
```

Arguments

x	a logical vector or list of logical vectors
y	an integer vector the same length as x; see Details

Details

Intervals estimator of extremal index based on Ferro and Segers (2003)'s moment-based estimator.

If `x` is supplied and `y` is not, `x` is assumed to identify consecutive threshold exceedances. If `x` is supplied as a list, each list element is assumed to comprise identifiers of consecutive exceedances. If `y` is supplied, `x` must be a logical vector, and `y` gives positions of `x` in its original with-missing-values vector: so `y` identifies consecutive `x`.

Value

A scalar estimate of the extremal index

References

Ferro, C. A., & Segers, J. (2003). Inference for clusters of extreme values. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 65(2), 545-556.

Examples

```
n <- 1e2
x <- runif(n)
extremal(x > .9)

y <- sort(sample(n, n - 5))
x2 <- x[y]
extremal(x2 > .9, y)
```

`FCtmax` *Fort Collins, Colorado, US daily max. temperatures*

Description

Daily maximum temperatures at Fort Collins, Colorado, US from 1st January 1970 to 31st December 2019

Usage

```
data(FCtmax)
```

Format

A data frame with 18156 rows and 2 variables

The variables are as follows:

date date of observation

tmax daily maximum temperature in degrees Celcius

Examples

```
library(evgam)
data(FCtmax)
```

`fitted.evgam` *Extract Model Fitted Values*

Description

Extract Model Fitted Values

Usage

```
## S3 method for class 'evgam'
fitted(object, ...)
```

Arguments

`object`	a fitted `evgam` object
`...`	not used

Value

Fitted values extracted from the object 'object'.

Examples

```
data(fremantle)
fmla_gev <- list(SeaLevel ~ s(Year, k=5, bs="cr"), ~ 1, ~ 1)
m_gev <- evgam(fmla_gev, fremantle, family = "gev")
fitted(m_gev)
```

`fremantle` *Annual Maximum Sea Levels at Fremantle, Western Australia*

Description

The 'fremantle' data frame has 86 rows and 3 columns. The second column gives 86 annual maximimum sea levels recorded at Fremantle, Western Australia, within the period 1897 to 1989. The first column gives the corresponding years. The third column gives annual mean values of the Southern Oscillation Index (SOI), which is a proxy for meteorological volitility.

Usage

```
data(fremantle)
```

Format

A data frame with 86 rows and 3 variables

The variables are as follows:

Year a numeric vector of years

SeaLevel a numeric vector of annual sea level maxima

SOI A numeric vector of annual mean values of the Southern Oscillation Index

Source

Coles, S. G. (2001) _An Introduction to Statistical Modelling of Extreme Values. London: Springer.

Eric Gilleland's ismev R package.

Examples

```
library(evgam)
data(fremantle)
```

---

logLik.evgam *Log-likelihood, AIC and BIC from a fitted* evgam *object*

Description

Log-likelihood, AIC and BIC from a fitted evgam object

Usage

```
## S3 method for class 'evgam'
logLik(object, ...)
```

Arguments

object	a fitted evgam object
...	not used

Value

A scalar

Examples

```
data(fremantle)
fmla_gev <- list(SeaLevel ~ s(Year, k=5, bs="cr"), ~ 1, ~ 1)
m_gev <- evgam(fmla_gev, fremantle, family = "gev")
logLik(m_gev)
AIC(m_gev)
BIC(m_gev)
```

`pinv` *Moore-Penrose pseudo-inverse of a matrix*

Description

Moore-Penrose pseudo-inverse of a matrix

Usage

```
pinv(x, tol = -1)

ginv.evgam(x, tol = sqrt(.Machine$double.eps))
```

Arguments

`x`	a matrix
`tol`	a scalar

Details

This function is merely a wrapper for Armadillo's pinv function with its default settings, which, in particular uses the divide-and-conquer method. If `tol` isn't provided Armadillo's default for pinv is used. `ginv.evgam` mimics ginv using Armadillo's pinv.

Value

A matrix

References

http://arma.sourceforge.net/docs.html#pinv

See Also

ginv

plot.evgam *Plot a fitted* evgam *object*

Description

Plot a fitted evgam object

Usage

```
## S3 method for class 'evgam'
plot(x, onepage = TRUE, which = NULL, main, ask = !onepage, ...)
```

Arguments

x	a fitted evgam object
onepage	logical: should all plots be on one page, or on separate pages? Defaults to TRUE
which	a vector of integers identifying which smooths to plot. The default NULL plots all smooths
main	a character string or vector of plot titles for each plot. If not supplied default titles are used
ask	logical: ask to show next plots if too many figures for current device?
...	extra arguments to pass to plot.gam

Value

Plots representing all one- or two-dimensional smooths

Examples

```
data(fremantle)
fmla_gev <- list(SeaLevel ~ s(Year, k=5, bs="cr"), ~ 1, ~ 1)
m_gev <- evgam(fmla_gev, fremantle, family = "gev")
plot(m_gev)
```

predict.evgam *Predictions from a fitted* evgam *object*

Description

Predictions from a fitted evgam object

Usage

```
## S3 method for class 'evgam'
predict(
  object,
  newdata,
  type = "link",
  prob = NULL,
  se.fit = FALSE,
  marginal = TRUE,
  exi = FALSE,
  trace = 0,
  ...
)
```

Arguments

object	a fitted evgam object
newdata	a data frame
type	a character string giving the type of prediction sought; see Details. Defaults to "link"
prob	a scalar or vector of probabilities for quantiles to be estimated if type == "quantile"; defaults to 0.5
se.fit	a logical: should estimated standard errors be returned? Defaults to FALSE
marginal	a logical: should uncertainty estimates integrate out smoothing parameter uncertainty? Defaults to TRUE
exi	a logical: if a dependent GEV is fitted should the independent parameters be returned? Defaults to FALSE
trace	an integer where higher values give more output. -1 suppresses everything. Defaults to 0
...	unused

Details

There are five options for type: 1) "link" distribution parameters transformed to their model fitting scale; 2) "response" as 1), but on their original scale; 3) "lpmatrix" a list of design matrices; 4) "quantile" estimates of distribution quantile(s); and 5) "qqplot" a quantile-quantile plot.

Value

A data frame or list of predictions, or a plot if `type == "qqplot"`

Examples

```
data(fremantle)
fmla_gev <- list(SeaLevel ~ s(Year, k=5, bs="cr"), ~ 1, ~ 1)
m_gev <- evgam(fmla_gev, fremantle, family = "gev")
# prediction of link GEV parameter for fremantle data
predict(m_gev)
# predictions for Year 1989
y1989 <- data.frame(Year = 1989)
# link GEV parameter predictions
predict(m_gev, y1989)
# GEV parameter predictions
predict(m_gev, y1989, type= "response")
# 10-year return level predictions
predict(m_gev, y1989, type= "quantile", prob = .9)
# 10- and 100-year return level predictions
predict(m_gev, y1989, type= "quantile", prob = c(.9, .99))
```

`print.evgam` *Print a fitted* `evgam` *object*

Description

Print a fitted `evgam` object

Usage

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

Arguments

`x`	a fitted `evgam` object
`...`	not used

Value

The call of the `evgam` object

Examples

```
data(fremantle)
fmla_gev <- list(SeaLevel ~ s(Year, k=5, bs="cr"), ~ 1, ~ 1)
m_gev <- evgam(fmla_gev, fremantle, family = "gev")
print(m_gev)
```

---

qev *Quantile estimation of a composite extreme value distribution*

Description

Quantile estimation of a composite extreme value distribution

Usage

```
qev(
  p,
  loc,
  scale,
  shape,
  m = 1,
  alpha = 1,
  theta = 1,
  family,
  tau = 0,
  start = NULL
)
```

Arguments

p	a scalar giving the quantile of the distribution sought
loc	a scalar, vector or matrix giving the location parameter
scale	as above, but scale parameter
shape	as above, but shape parameter
m	a scalar giving the number of values per return period unit, e.g. 365 for daily data giving annual return levels
alpha	a scalar, vector or matrix of weights if within-block variables not identically distributed and of different frequencies
theta	a scalar, vector or matrix of extremal index values
family	a character string giving the family for which return levels sought
tau	a scalar, vector or matrix of values giving the threshold quantile for the GPD (i.e. 1 - probability of exceedance)
start	a 2-vector giving starting values that bound the return level

Details

If F is the generalised extreme value or generalised Pareto distribution, qev solves

$$\prod_{j=1}^{n} \{F(z)\}^{m\alpha_j\theta_j} = p.$$

For both distributions, location, scale and shape parameters are given by loc, scale and shape. The generalised Pareto distribution, for $\xi \neq 0$ and $z > u$, is parameterised as $1-(1-\tau)[1+\xi(z-u)/\psi_u]^{-1/\xi}$, where u, ψ_u and ξ are its location, scale and shape parameters, respectively, and τ corresponds to argument tau.

Value

A scalar or vector of estimates of p

Examples

```
qev(0.9, c(1, 2), c(1, 1.1), .1, family="gev")
qev(0.99, c(1, 2), c(1, 1.1), .1, family="gpd", tau=0.9)
```

---

runmax *Running maximum*

Description

Running n-value maximum and data frame with variable swapped for running maximum

Usage

```
runmax(y, n)

dfrunmax(data, cons, ynm, n = 2)
```

Arguments

y	a vector
n	an integer giving the number of observations to calculate running maxmimum over; defaults to 2
data	a data frame
cons	a character string for the variable in data that identifies consecutive observations
ynm	a character string for the variable in data that is the observations

Value

`runmax` returns a vector of the same dimension as `y`

`dfrunmax` returns a data frame with observations swapped for n-observation running maximum

Examples

```
runmax(runif(10), 5)
```

`seq_between` *More Sequence Generation*

Description

Generate a sequence of values between a range.

Usage

```
seq_between(x, length = NULL)
```

Arguments

`x`	a 2-vector
`length`	an integer

Value

A vector

See Also

seq, seq_len, seq_along

Examples

```
seq_between(c(1, 9))
seq_between(range(runif(10)), 5)
```

simulate.evgam *Simulations from a fitted* evgam *object*

Description

Simulations from a fitted evgam object

Usage

```
## S3 method for class 'evgam'
simulate(
  object,
  nsim = 1000,
  seed = NULL,
  newdata,
  type = "link",
  probs = NULL,
  threshold = 0,
  marginal = TRUE,
  ...
)
```

Arguments

object	a fitted evgam object
nsim	an integer giving the number of simulations
seed	an integer giving the seed for simulations
newdata	a data frame
type	a character string, as in predict.evgam; defaults to "quantile"
probs	a scalar or vector of probabilities for quantiles; defaults to NULL
threshold	a scalar, vector or matrix, which is added to each simulation if family == "gpd"; defaults to 0
marginal	a logical: should simulations integrate out smoothing parameter uncertainty? Defaults to TRUE
...	arguments to be passed to predict.evgam

Value

Simulations of parameters or quantiles

See Also

predict.evgam

Examples

```
data(fremantle)
fmla_gev <- list(SeaLevel ~ s(Year, k=5, bs="cr"), ~ 1, ~ 1)
m_gev <- evgam(fmla_gev, fremantle, family = "gev")
# simulations of link GEV parameters for fremantle data
simulate(m_gev, nsim=5)
# simulations for Year 1989
y1989 <- data.frame(Year = 1989)
# link GEV parameter simulations
simulate(m_gev, nsim=5, newdata = y1989)
# GEV parameter simulations
simulate(m_gev, nsim=5, newdata = y1989, type = "response")
# 10-year return level simulations
simulate(m_gev, nsim=5, newdata = y1989, type= "quantile", prob = .9)
# 10- and 100-year return level simulations
simulate(m_gev, nsim=5, newdata = y1989, type= "quantile", prob = c(.9, .99))
```

---

summary.evgam *Summary method for a fitted* evgam *object*

Description

Summary method for a fitted evgam object

Usage

```
## S3 method for class 'evgam'
summary(object, ...)

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

Arguments

object	a fitted evgam object
...	not used
x	a summary.evgam object

Details

The key part of summary.evgam is p-values for smooths. The tests use code directly taken from mgcv 1.8-14. This is to avoid use of mgcv:::... . Tests implement the method of Wood (2013).

Value

A summary.evgam object

References

Wood, S. N., (2013) On p-values for smooth components of an extended generalized additive model, Biometrika 100(1) 221–228

Examples

```
data(fremantle)
fmla_gev <- list(SeaLevel ~ s(Year, k=5, bs="cr"), ~ 1, ~ 1)
m_gev <- evgam(fmla_gev, fremantle, family = "gev")
summary(m_gev)
```

Index