

Package: ExtremeCI (via r-universe)

May 13, 2026

Type Package

Title Realistic Confidence Intervals for Non-Stationary Extreme Value Statistics

Version 0.2.1

Description This framework provides versatile algorithms to efficiently infer confidence intervals for extreme value statistics, such as extreme quantiles and return levels, that are representative of the asymmetric uncertainty spread, using extreme value theory extrapolation and the profile likelihood (see e.g., Coles (2001) <doi:10.1007/978-1-4471-3675-0>). Unlike existing algorithms, the CI endpoints are found without the need for a strict prespecified range, can be covariate-dependent, and can be based on weighted samples. This package is motivated by Zeder et al. (2023) <doi:10.1029/2023GL104090> and by Pasche et al. (2026) <doi:10.1007/s10687-026-00536-9>.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

Imports doFuture, dplyr, evd, foreach, future, ggplot2, magrittr, rlang, stats, tibble, tidyr, tidyselect

URL <https://github.com/opasche/ExtremeCI>,
<https://opasche.github.io/ExtremeCI/>

BugReports <https://github.com/opasche/ExtremeCI/issues>

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libicu-dev

Repository <https://opasche.r-universe.dev>

Date/Publication 2026-05-13 15:34:04 UTC

RemoteUrl <https://github.com/opasche/extremeci>

RemoteRef HEAD

RemoteSha da8ef1edaeb9db211ce9aa2c19b0a72ff5e58e8

Contents

GEV_change_parametrization	2
GEV_endpoint	3
GEV_log_likelihood	4
GEV_maxlik	4
GEV_profile_CI	6
GEV_profile_CIs_multiple	8
GEV_profile_loglik	11
GEV_profile_loglik_curve	12
GEV_return_level	15
GPD_change_parametrization	15
GPD_endpoint	16
GPD_log_likelihood	17
GPD_maxlik	18
GPD_profile_CI	19
GPD_profile_CIs_multiple	22
GPD_profile_loglik	25
GPD_profile_loglik_curve	27
GPD_quantiles	29
plot_data_quantile_ci	30
plot_profile_loglik_curve	31
Index	32

GEV_change_parametrization
GEV parameter vector reparametrization

Description

GEV parameter vector reparametrization

Usage

```
GEV_change_parametrization(
  parameters,
  parametrization = c("classical", "return_level", "endpoint"),
  new_parametrization = c("classical", "return_level", "endpoint"),
  return_period = 100,
  nbloc = 1,
  nbsca = 1,
  nbsha = 1
)
```

Arguments

parameters	Vector of GEV parameters in the internal format.
parametrization	Current parametrization of parameters.
new_parametrization	Desired new parametrization.
return_period	Return period for the 'return_level' parameter.
nbloc	Number of location parameter coefficients (i.e. one plus the number of shape parameter covariates).
nbsca	Number of scale parameter coefficients (i.e. one plus the number of scale parameter covariates).
nbsha	Number of shape parameter coefficients (i.e. one plus the number of shape parameter covariates).

Value

The vector of GEV parameters reparametrized from parametrization to new_parametrization.

GEV_endpoint	<i>GEV_endpoint</i>
--------------	---------------------

Description

GEV endpoint

Usage

```

GEV_endpoint(
  loc = 0,
  scale = 1,
  shape,
  ...,
  endpoint_type = c("unrestricted", "upper", "lower")
)

```

Arguments

loc	Location parameter.
scale	Scale parameter.
shape	Shape parameter.
...	Other optional parameters for the internal endpoint function (Unused).
endpoint_type	Whether to compute the upper endpoint (returns Inf if shape>=0), or the unrestricted endpoint (returns the lower GEV endpoint if shape>=0).

Value

The endpoint of the specified GEV distribution.

Examples

```
GEV_endpoint(loc=0, scale=1, shape=-0.1, endpoint_type='upper')
```

GEV_log_likelihood	<i>GEV log likelihood</i>
--------------------	---------------------------

Description

GEV log likelihood

Usage

```
GEV_log_likelihood(Z, loc, scale, shape)
```

Arguments

Z	Block maxima observations.
loc	Location parameter.
scale	Scale parameter.
shape	Shape parameter.

Value

The GEV log-likelihood evaluated at the given parameters, given the data observations Y.

GEV_maxlik	<i>Maximum-likelihood GEV estimate</i>
------------	--

Description

Maximum-likelihood GEV estimate

Usage

```

GEV_maxlik(
  Z,
  parametrization = c("classical", "return_level", "endpoint"),
  return_period = 100,
  orthogonal = FALSE,
  X = NULL,
  x_rlvl = NULL,
  loc_cols = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  out_param = parametrization,
  hessian = TRUE,
  maxit = 1e+06,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  ...
)

```

Arguments

Z	Block maxima observations.
parametrization	Likelihood parametrization. Alternatives to classical substitute for the location parameter.
return_period	Return period for the 'return_level' parametrization.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits).
x_rlvl	Covariate vector at which to reparametrize for the 'return_level' or 'endpoint' parametrizations (for conditional/non-stationary fits).
loc_cols	Column indices of X to use as covariate for the (conditional) location parameter (for conditional/non-stationary fits).
scale_cols	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).
shape_cols	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).
out_param	Additional output parametrization (same as parametrization, by default). If out_param != parametrization, the parameters are reparametrized from parametrization to out_param after estimation, in a separate output.
hessian	Logical. Should a numerically differentiated Hessian matrix be returned? See stats::optim() for more details.
maxit	The maximum number of iterations. See stats::optim() for more details.
method	The optimisation method to be used. See stats::optim() for more details.
...	Other arguments passed to the control argument of stats::optim() .

Value

The fitted maximum-likelihood GEV as a GEV_ML object, containing:

mle	The estimated maximum likelihood GEV parameters, as a named vector (expressed in parametrization).
loglik	The log-likelihood of the estimated parameters, given the data
conv	Whether the optimisation procedure converged. See the convergence output of <code>stats::optim()</code> for more details.
hessian	The loglikelihood hessian evaluated at the estimated parameters, given the data.
parametrization	Likelihood parametrization.
out_mle	The estimated maximum likelihood GEV parameters, reparametrized in out_param.
out_parametrization	Additional output parametrization.

 GEV_profile_CI

GEV profile CI using binary search

Description

GEV profile CI using binary search

Usage

```

GEV_profile_CI(
  Z,
  parameter = c("shape", "location", "scale", "return_level", "endpoint"),
  subparam_id = 0,
  alpha = 0.05,
  return_period = 100,
  orthogonal = FALSE,
  X = NULL,
  x_rlvl = NULL,
  loc_cols = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  warmstart_table = NULL,
  init_step_pos = 100,
  init_step_neg = 10,
  tol = 0.01,
  steps_beyond_conf = 5,
  initial_MLE_para = c("classical", "same"),
  max_steps = 1000,
  hessian = TRUE,
  maxit = 1e+06,

```

```

method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
verbose = 1,
...
)

```

Arguments

Z	Block maxima observations.
parameter	Parameter for which to compute the profile log-likelihood.
subparam_id	Index of the parameter coefficient for which to compute the profile log-likelihood (for conditional/non-stationary fits).
alpha	Confidence alpha for the profile likelihood confidence interval (i.e. for the confidence line on the profile plot).
return_period	Return period for the 'return_level' parameter.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits). Columns should be variables, and rows should be observations matching Y.
x_rlvl	Covariate vector at which to reparametrize for the 'return_level' or 'endpoint' parametrizations (for conditional/non-stationary fits).
loc_cols	Column indices of X to use as covariate for the (conditional) location parameter (for conditional/non-stationary fits).
scale_cols	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).
shape_cols	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).
warmstart_table	Evaluation table from a previous run.
init_step_pos	Initial numerical size of each evaluation step to the right, in the profile parameter's scale.
init_step_neg	Initial numerical size of each evaluation step to the left, in the profile parameter's scale.
tol	Numerical tolerance for convergence, in the profile parameter's scale.
steps_beyond_conf	Number of additional steps to take (in each direction) after the profile log-likelihood values reach below the confidence line.
initial_MLE_para	Parametrization used for the initial maximum likelihood estimate (defaults to classical, for better stability).
max_steps	Maximum number of steps taken (in each direction). If the confidence line was not reached, the corresponding confidence interval endpoint will be infinite.
hessian	Logical. Should a numerically differentiated Hessian matrix be returned? See stats::optim() for more details.
maxit	The maximum number of iterations. See stats::optim() for more details.

method	The optimisation method to be used. See <code>stats::optim()</code> for more details.
verbose	Verbose level, as integer.
...	Other arguments passed to the control argument of <code>stats::optim()</code> .

Value

The GEV profile log-likelihood confidence interval for the desired parameter, with confidence line and resulting $(1-\alpha)$ confidence interval, as a `GEV_profileLogLik` object containing:

mle	The estimated maximum likelihood GEV parameters, as a named vector (expressed in the profile parametrization).
ci	Length-two vector containing the lower and upper endpoints of the desired profile likelihood confidence interval.
profile_loglik	Named matrix containing the profile loglikelihood value (Column 2) for each considered profile parameter value (Column 1).
conf_line	Confidence line for the desired profile likelihood confidence interval. See e.g. Coles (2001) for more details.
eval_table	Tibble (<code>tibble::tibble()</code>) containing the history of profile log-likelihood evaluation values, and related metadata.
param_name	Name of the profiled parameter (inferred, for debugging purposes).
parameter	Name of the profiled parameter (given).
parametrization	Parametrization used for the profile likelihood.
subparam_id	Index of the parameter coefficient for which the profile log-likelihood was computed.
id_param	Index of the profile parameter in the GEV parameter vector.

References

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

GEV_profile_CIs_multiple

Multi-value conditional GEV profile likelihood confidence intervals

Description

For non-stationary models, the return level reparametrization depends on covariate values. This function repeats the profile likelihood procedure for several covariate values. It enables obtaining a return-level (or endpoint) curve, with profile-likelihood confidence bands, as a function of the covariate values.

Usage

```

GEV_profile_CIs_multiple(
  Z,
  parameter = c("return_level", "endpoint"),
  alpha = 0.05,
  return_period = 100,
  orthogonal = FALSE,
  X = NULL,
  X_rlvl = NULL,
  loc_cols = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  init_step_pos = 100,
  init_step_neg = 10,
  tol = 0.01,
  steps_beyond_conf = 5,
  initial_MLE_para = c("classical", "same"),
  max_steps = 10000,
  hessian = TRUE,
  maxit = 1e+06,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  parallel_strat = c("none", "multisession", "sequential", "multicore"),
  n_workers = NULL,
  ...
)

```

Arguments

Z	Block maxima observations.
parameter	Parameter for which to compute the profile likelihood confidence intervals.
alpha	Confidence alpha for the profile likelihood confidence intervals.
return_period	Return period for the 'return_level' parameter.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits). Columns should be variables, and rows should be observations matching Y.
X_rlvl	Covariate matrix at which to reparametrize for the 'return_level' or 'endpoint' parametrizations (for conditional/non-stationary fits). Columns should be variables, and each row should give one covariate realization at which to reparametrize and obtain a CI.
loc_cols	Column indices of X to use as covariate for the (conditional) location parameter (for conditional/non-stationary fits).
scale_cols	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).
shape_cols	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).

<code>init_step_pos</code>	Initial numerical size of each evaluation step to the right, in the profile parameter's scale.
<code>init_step_neg</code>	Initial numerical size of each evaluation step to the left, in the profile parameter's scale.
<code>tol</code>	Numerical tolerance for convergence, in the profile parameter's scale.
<code>steps_beyond_conf</code>	Number of additional steps to take (in each direction) after the profile log-likelihood values reach below the confidence line.
<code>initial_MLE_para</code>	Parametrization used for the initial maximum likelihood estimate (defaults to classical, for better stability).
<code>max_steps</code>	Maximum number of steps taken (in each direction). If the confidence line was not reached, the corresponding confidence interval endpoint will be infinite.
<code>hessian</code>	Logical. Should a numerically differentiated Hessian matrix be returned? See stats::optim() for more details.
<code>maxit</code>	The maximum number of iterations. See stats::optim() for more details.
<code>method</code>	The optimisation method to be used. See stats::optim() for more details.
<code>parallel_strat</code>	Parallel strategy. One of "sequential" (default), "multisession", "multicore", or "mixed".
<code>n_workers</code>	A positive numeric scalar or a function specifying the maximum number of parallel futures that can be active at the same time before blocking. If a function, it is called without arguments when the future is created and its value is used to configure the workers. The function should return a numeric scalar. Defaults to future::availableCores() -1 if NULL (default), with "multicore" constraint in the relevant case. Ignored if <code>strategy=="sequential"</code> .
<code>...</code>	Other arguments passed to the <code>control</code> argument of stats::optim() .

Value

The GEV profile log-likelihood (1-alpha) confidence intervals for the desired parameter, for each desired covariate values, as a [tibble::tibble\(\)](#), with columns:

<code>obs</code>	Index of the observation (i.e. row) of <code>X_rlvl</code> for which the parameter estimate and CI was computed.
<code>\verb{<parameter name>}</code>	Conditional estimate of the parameter.
<code>ci_down</code>	Lower endpoint of the conditional (1-alpha) profile-likelihood confidence interval for parameter.
<code>ci_up</code>	Upper endpoint of the conditional (1-alpha) profile-likelihood confidence interval for parameter.
<code>parameter</code>	Name of the parameter for which the estimates and CIs were computed.
<code>alpha</code>	Confidence alpha for the profile likelihood confidence intervals.
<code>return_period</code>	Return period for the 'return_level' parameter (only if <code>parameter==return_level</code>).

GEV_profile_loglik *GEV profile log-likelihood*

Description

GEV profile log-likelihood

Usage

```
GEV_profile_loglik(
  val,
  Z,
  parameter = c("shape", "location", "scale", "return_level", "endpoint"),
  subparam_id = 0,
  return_period = 100,
  orthogonal = FALSE,
  X = NULL,
  x_rlvl = NULL,
  loc_cols = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  init = NULL,
  hessian = TRUE,
  maxit = 1e+06,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  ...
)
```

Arguments

val	Parameter value at which to evaluate the GEV profile log-likelihood.
Z	Block maxima observations.
parameter	Parameter for which to compute the profile log-likelihood.
subparam_id	Index of the parameter coefficient for which to compute the profile log-likelihood (for conditional/non-stationary fits).
return_period	Return period for the 'return_level' parameter.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits). Columns should be variables, and rows should be observations matching Y.
x_rlvl	Covariate vector at which to reparametrize for the 'return_level' or 'endpoint' parametrizations (for conditional/non-stationary fits).
loc_cols	Column indices of X to use as covariate for the (conditional) location parameter (for conditional/non-stationary fits).
scale_cols	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).

shape_cols	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).
init	Optional initial values for the remaining parameter's optimisation process, in the correct internal format.
hessian	Logical. Should a numerically differentiated Hessian matrix be returned? See stats::optim() for more details.
maxit	The maximum number of iterations. See stats::optim() for more details.
method	The optimisation method to be used. See stats::optim() for more details.
...	Other arguments passed to the control argument of stats::optim() .

Value

The GEV profile log-likelihood of parameter evaluated at val, given the data, as a GEV_profML object containing:

param_val	(Named) parameter value at which the GEV profile log-likelihood was evaluated.
param_name	Name of the evaluated profile likelihood parameter.
mle_other	Maximum-likelihood estimate of the other GEV parameters.
loglik	Profile GEV log-likelihood value of parameter evaluated at val, given the data.
conv	Whether the optimisation procedure converged. See the convergence output of stats::optim() for more details.
hessian	The loglikelihood hessian evaluated at the estimated parameters, given the data.
parameter	Name of the evaluated profile likelihood parameter given as argument (redundent).
parametrization	Likelihood parametrization.
subparam_id	Index of the parameter coefficient for which the profile log-likelihood was computed.
id_param	Index of the likelihood profile parameter, in the internal parameter vector format.

GEV_profile_loglik_curve

GEV profile log-likelihood curve

Description

GEV profile log-likelihood curve

Usage

```

GEV_profile_loglik_curve(
  Z,
  parameter = c("shape", "location", "scale", "return_level", "endpoint"),
  subparam_id = 0,
  alpha = 0.05,
  return_period = 100,
  orthogonal = FALSE,
  X = NULL,
  x_rlvl = NULL,
  loc_cols = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  warmstart_table = NULL,
  stepsize = 0.1,
  steps_beyond_conf = 5,
  initial_MLE_para = c("classical", "same"),
  max_steps = 10000,
  hessian = TRUE,
  maxit = 1e+06,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  ...
)

```

Arguments

Z	Block maxima observations.
parameter	Parameter for which to compute the profile log-likelihood.
subparam_id	Index of the parameter coefficient for which to compute the profile log-likelihood (for conditional/non-stationary fits).
alpha	Confidence alpha for the profile likelihood confidence interval (i.e. for the confidence line on the profile plot).
return_period	Return period for the 'return_level' parameter.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits). Columns should be variables, and rows should be observations matching Y.
x_rlvl	Covariate vector at which to reparametrize for the 'return_level' or 'endpoint' parametrizations (for conditional/non-stationary fits).
loc_cols	Column indices of X to use as covariate for the (conditional) location parameter (for conditional/non-stationary fits).
scale_cols	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).
shape_cols	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).

warmstart_table	Evaluation table from a previous run.
stepsize	Numerical size of each evaluation step, in the profile parameter's scale.
steps_beyond_conf	Number of additional steps to take (in each direction) after the profile log-likelihood values reach below the confidence line.
initial_MLE_para	Parametrization used for the initial maximum likelihood estimate (defaults to classical, for better stability).
max_steps	Maximum number of steps taken (in each direction). If the confidence line was not reached, the corresponding confidence interval endpoint will be infinite.
hessian	Logical. Should a numerically differentiated Hessian matrix be returned? See <code>stats::optim()</code> for more details.
maxit	The maximum number of iterations. See <code>stats::optim()</code> for more details.
method	The optimisation method to be used. See <code>stats::optim()</code> for more details.
...	Other arguments passed to the control argument of <code>stats::optim()</code> .

Value

The GEV profile log-likelihood curve for the desired parameter, with confidence line and resulting (1-alpha) confidence interval, as a `GEV_profileLogLik` object containing:

mle	The estimated maximum likelihood GEV parameters, as a named vector (expressed in the profile parametrization).
ci	Length-two vector containing the lower and upper endpoints of the desired profile likelihood confidence interval.
profile_loglik	Named matrix containing the profile loglikelihood value (Column 2) for each considered profile parameter value (Column 1).
conf_line	Confidence line for the desired profile likelihood confidence interval. See e.g. Coles (2001) for more details.
eval_table	Tibble (<code>tibble::tibble()</code>) containing the history of profile log-likelihood evaluation values, and related metadata.
param_name	Name of the profiled parameter (inferred, for debugging purposes).
parameter	Name of the profiled parameter (given).
parametrization	Parametrization used for the profile likelihood.
subparam_id	Index of the parameter coefficient for which the profile log-likelihood was computed.
id_param	Index of the profile parameter in the GEV parameter vector.

References

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

GEV_return_level	<i>Compute return level from GEV parameters</i>
------------------	---

Description

Compute return level from GEV parameters

Usage

```
GEV_return_level(loc = 0, scale = 1, shape, return_period)
```

Arguments

loc	Location parameter.
scale	Scale parameter.
shape	Shape parameter.
return_period	Return period for the desired return level.

Value

The return level of the specified GEV distribution with a return period of return_period. In other terms, the quantile of the GEV distribution at probability level $1 - 1/\text{return_period}$.

Examples

```
GEV_return_level(loc=0, scale=1, shape=0.1, return_period=100)
```

GPD_change_parametrization	<i>GPD parameter vector reparametrization</i>
----------------------------	---

Description

GPD parameter vector reparametrization

Usage

```
GPD_change_parametrization(
  parameters,
  threshold = 0,
  threshold_lvl = 0,
  parametrization = c("classical", "orthogonal", "quantile", "endpoint"),
  new_parametrization = c("classical", "orthogonal", "quantile", "endpoint"),
  quantile_lvl = 1 - (1/100),
  nbsca = 1,
  nbsha = 1
)
```

Arguments

parameters	Vector of GPD parameters in the internal format.
threshold	GPD threshold value.
threshold_lvl	Probability level of the threshold threshold.
parametrization	Current parametrization of parameters.
new_parametrization	Desired new parametrization.
quantile_lvl	Quantile probability level for the 'quantile' parameter.
nbsca	Number of scale parameter coefficients (i.e. one plus the number of scale parameter covariates).
nbsha	Number of shape parameter coefficients (i.e. one plus the number of shape parameter covariates).

Value

The vector of GPD parameters reparametrized from parametrization to new_parametrization.

GPD_endpoint	<i>GPD endpoint</i>
--------------	---------------------

Description

GPD endpoint

Usage

```
GPD_endpoint(
  threshold = 0,
  scale = 1,
  shape,
  ...,
  endpoint_type = c("unrestricted", "upper")
)
```

Arguments

threshold	GPD threshold value.
scale	Scale parameter.
shape	Shape parameter.
...	Other optional parameters for the internal endpoint function (Unused).
endpoint_type	Whether to compute the upper endpoint (returns Inf if shape>=0), or the unrestricted endpoint (returns the lower GPD endpoint if shape>=0).

Value

The endpoint of the specified GPD distribution.

Examples

```
GPD_endpoint(threshold=0, scale=1, shape=-0.1, endpoint_type='upper')
```

GPD_log_likelihood *GPD log likelihood*

Description

GPD log likelihood

Usage

```
GPD_log_likelihood(
  Y,
  threshold,
  scale,
  shape,
  obs_weights = NULL,
  ill_defined_value = -10^6
)
```

Arguments

Y	Data observations.
threshold	GPD threshold value.
scale	Scale parameter.
shape	Shape parameter.
obs_weights	Optional observation weights for weighted likelihood.
ill_defined_value	Value to return if the arguments are out of support (e.g. negative scale, or non-positive arguments to logarithms).

Value

The GPD log-likelihood evaluated at the given parameters, given the data observations Y.

GPD_maxlik

*Maximum-likelihood GPD estimate***Description**

Maximum-likelihood GPD estimate

Usage

```
GPD_maxlik(
  Y,
  threshold = 0,
  threshold_lvl = 0,
  parametrization = c("classical", "orthogonal", "quantile", "endpoint"),
  quantile_lvl = 1 - (1/100),
  orthogonal = FALSE,
  X = NULL,
  x_rlvl = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  out_param = parametrization,
  obs_weights = NULL,
  ill_defined_value = -10^6,
  hessian = TRUE,
  maxit = 1e+06,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  verbose = 1,
  ...
)
```

Arguments

Y	Data observations.
threshold	GPD threshold value.
threshold_lvl	Probability level of the threshold threshold.
parametrization	Likelihood parametrization. Alternatives to classical substitute for the scale parameter.
quantile_lvl	Quantile probability level for the 'quantile' parametrization.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits).
x_rlvl	Covariate vector at which to reparametrize for the 'quantile' or 'endpoint' parametrizations (for conditional/non-stationary fits).
scale_cols	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).

shape_cols	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).
out_param	Additional output parametrization (same as parametrization, by default). If out_param != parametrization, the parameters are reparametrized from parametrization to out_param after estimation, in a separate output.
obs_weights	Optional observation weights for weighted likelihood.
ill_defined_value	Value to return if the arguments are out of support (e.g. negative scale, or non-positive arguments to logarithms).
hessian	Logical. Should a numerically differentiated Hessian matrix be returned? See stats::optim() for more details.
maxit	The maximum number of iterations. See stats::optim() for more details.
method	The optimisation method to be used. See stats::optim() for more details.
verbose	Verbose level, as integer.
...	Other arguments passed to the control argument of stats::optim() .

Value

The fitted maximum-likelihood GPD as a GPD_ML object, containing:

mle	The estimated maximum likelihood GPD parameters, as a named vector (expressed in parametrization).
loglik	The log-likelihood of the estimated parameters, given the data
conv	Whether the optimisation procedure converged. See the convergence output of stats::optim() for more details.
hessian	The loglikelihood hessian evaluated at the estimated parameters, given the data.
parametrization	Likelihood parametrization.
out_mle	The estimated maximum likelihood GPD parameters, reparametrized in out_param.
out_parametrization	Additional output parametrization.

GPD_profile_CI

GPD profile CI using binary search

Description

GPD profile CI using binary search

Usage

```
GPD_profile_CI(
  Y,
  threshold = 0,
  threshold_lvl = 0,
  parameter = c("shape", "scale", "quantile", "endpoint"),
  subparam_id = 0,
  alpha = 0.05,
  quantile_lvl = 1 - (1/100),
  orthogonal = FALSE,
  X = NULL,
  x_rlvl = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  warmstart_table = NULL,
  init_step_pos = 100,
  init_step_neg = 10,
  tol = 0.01,
  steps_beyond_conf = 5,
  initial_MLE_para = c("classical", "same"),
  max_steps = 1000,
  obs_weights = NULL,
  ill_defined_value = -10^6,
  hessian = TRUE,
  maxit = 1e+06,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  method_prof = c("default", "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  verbose = 1,
  ...
)
```

Arguments

Y	Data observations.
threshold	GPD threshold value.
threshold_lvl	Probability level of the threshold threshold.
parameter	Parameter for which to compute the profile log-likelihood.
subparam_id	Index of the parameter coefficient for which to compute the profile log-likelihood (for conditional/non-stationary fits).
alpha	Confidence alpha for the profile log-likelihood confidence interval (i.e. for the confidence line on the profile plot).
quantile_lvl	Quantile probability level for the 'quantile' parameter.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits). Columns should be variables, and rows should be observations matching Y.

<code>x_rlvl</code>	Covariate vector at which to reparametrize for the 'quantile' or 'endpoint' parametrizations (for conditional/non-stationary fits).
<code>scale_cols</code>	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).
<code>shape_cols</code>	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).
<code>warmstart_table</code>	Evaluation table from a previous run.
<code>init_step_pos</code>	Initial numerical size of each evaluation step to the right, in the profile parameter's scale.
<code>init_step_neg</code>	Initial numerical size of each evaluation step to the left, in the profile parameter's scale.
<code>tol</code>	Numerical tolerance for convergence, in the profile parameter's scale.
<code>steps_beyond_conf</code>	Number of additional steps to take (in each direction) after the profile log-likelihood values reach below the confidence line.
<code>initial_MLE_para</code>	Parametrization used for the initial maximum likelihood estimate (defaults to classical, for better stability).
<code>max_steps</code>	Maximum number of steps taken (in each direction). If the confidence line was not reached, the corresponding confidence interval endpoint will be infinite.
<code>obs_weights</code>	Optional observation weights for weighted likelihood.
<code>ill_defined_value</code>	Value to return if the arguments are out of support (e.g. negative scale, or non-positive arguments to logarithms).
<code>hessian</code>	Logical. Should a numerically differentiated Hessian matrix be returned? See stats::optim() for more details.
<code>maxit</code>	The maximum number of iterations. See stats::optim() for more details.
<code>method</code>	The optimisation method to be used for the initial maximum likelihood optimisation. See stats::optim() for more details.
<code>method_prof</code>	The optimisation method to be used for the profile likelihood optimisation. See stats::optim() for more details.
<code>verbose</code>	Verbose level, as integer.
<code>...</code>	Other arguments passed to the <code>control</code> argument of stats::optim() .

Value

The GPD profile log-likelihood confidence interval for the desired parameter, with confidence line and resulting $(1-\alpha)$ confidence interval, as a `GPD_profileLogLik` object containing:

<code>mle</code>	The estimated maximum likelihood GPD parameters, as a named vector (expressed in the profile parametrization).
<code>ci</code>	Length-two vector containing the lower and upper endpoints of the desired profile likelihood confidence interval.

profile_loglik	Named matrix containing the profile loglikelihood value (Column 2) for each considered profile parameter value (Column 1).
conf_line	Confidence line for the desired profile likelihood confidence interval. See e.g. Coles (2001) for more details.
eval_table	Tibble (<code>tibble::tibble()</code>) containing the history of profile log-likelihood evaluation values, and related metadata.
param_name	Name of the profiled parameter (inferred, for debugging purposes).
parameter	Name of the profiled parameter (given).
parametrization	Parametrization used for the profile likelihood.
subparam_id	Index of the parameter coefficient for which the profile log-likelihood was computed.
id_param	Index of the profile parameter in the GPD parameter vector.

References

Coles, S. (2001). *An Introduction to Statistical Modeling of Extreme Values*. Springer. doi:[10.1007/9781447136750](https://doi.org/10.1007/9781447136750).

GPD_profile_CIs_multiple

Multi-value conditional GPD profile likelihood confidence intervals

Description

For non-stationary models, the quantile reparametrization depends on covariate values. This function repeats the profile likelihood procedure for several covariate values. It enables obtaining a quantile (or endpoint) curve, with profile-likelihood confidence bands, as a function of the covariate values.

Usage

```
GPD_profile_CIs_multiple(
  Y,
  threshold = 0,
  threshold_lv1 = 0,
  parameter = c("quantile", "endpoint"),
  alpha = 0.05,
  quantile_lv1 = 1 - (1/100),
  orthogonal = FALSE,
  X = NULL,
  X_rlv1 = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  init_step_pos = 100,
```

```

init_step_neg = 10,
tol = 0.01,
steps_beyond_conf = 5,
initial_MLE_para = c("classical", "same"),
max_steps = 10000,
obs_weights = NULL,
ill_defined_value = -10^6,
hessian = TRUE,
maxit = 1e+06,
method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
method_prof = c("default", "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
parallel_strat = c("none", "multisession", "sequential", "multicore"),
n_workers = NULL,
...
)

```

Arguments

Y	Data observations.
threshold	GPD threshold value.
threshold_lvl	Probability level of the threshold threshold.
parameter	Parameter for which to compute the profile likelihood confidence intervals.
alpha	Confidence alpha for the profile likelihood confidence intervals.
quantile_lvl	Quantile probability level for the 'quantile' parameter.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits). Columns should be variables, and rows should be observations matching Y.
X_rlvl	Covariate matrix at which to reparametrize for the 'quantile' or 'endpoint' parametrizations (for conditional/non-stationary fits). Columns should be variables, and each row should give one covariate realization at which to reparametrize and obtain a CI.
scale_cols	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).
shape_cols	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).
init_step_pos	Initial numerical size of each evaluation step to the right, in the profile parameter's scale.
init_step_neg	Initial numerical size of each evaluation step to the left, in the profile parameter's scale.
tol	Numerical tolerance for convergence, in the profile parameter's scale.
steps_beyond_conf	Number of additional steps to take (in each direction) after the profile log-likelihood values reach below the confidence line.

<code>initial_MLE_para</code>	Parametrization used for the initial maximum likelihood estimate (defaults to classical, for better stability).
<code>max_steps</code>	Maximum number of steps taken (in each direction). If the confidence line was not reached, the corresponding confidence interval endpoint will be infinite.
<code>obs_weights</code>	Optional observation weights for weighted likelihood.
<code>ill_defined_value</code>	Value to return if the arguments are out of support (e.g. negative scale, or non-positive arguments to logarithms).
<code>hessian</code>	Logical. Should a numerically differentiated Hessian matrix be returned? See stats::optim() for more details.
<code>maxit</code>	The maximum number of iterations. See stats::optim() for more details.
<code>method</code>	The optimisation method to be used for the initial maximum likelihood optimisation. See stats::optim() for more details.
<code>method_prof</code>	The optimisation method to be used for the profile likelihood optimisation. See stats::optim() for more details.
<code>parallel_strat</code>	Parallel strategy. One of "sequential" (default), "multisession", "multicore", or "mixed".
<code>n_workers</code>	A positive numeric scalar or a function specifying the maximum number of parallel futures that can be active at the same time before blocking. If a function, it is called without arguments when the future is created and its value is used to configure the workers. The function should return a numeric scalar. Defaults to future::availableCores() -1 if NULL (default), with "multicore" constraint in the relevant case. Ignored if <code>strategy=="sequential"</code> .
<code>...</code>	Other arguments passed to the control argument of stats::optim() .

Value

The GPD profile log-likelihood $(1-\alpha)$ confidence intervals for the desired parameter, for each desired covariate values, as a [tibble::tibble\(\)](#), with columns:

<code>obs</code>	Index of the observation (i.e. row) of <code>X_rlvl</code> for which the parameter estimate and CI was computed.
<code>\verb{<parameter name>}</code>	Conditional estimate of the parameter.
<code>ci_down</code>	Lower endpoint of the conditional $(1-\alpha)$ profile-likelihood confidence interval for parameter.
<code>ci_up</code>	Upper endpoint of the conditional $(1-\alpha)$ profile-likelihood confidence interval for parameter.
<code>parameter</code>	Name of the parameter for which the estimates and CIs were computed.
<code>alpha</code>	Confidence alpha for the profile likelihood confidence intervals.
<code>quantile_lvl</code>	Quantile probability level for 'quantile' parameter (only if <code>parameter==quantile</code>).

GPD_profile_loglik *GPD profile log-likelihood*

Description

GPD profile log-likelihood

Usage

```
GPD_profile_loglik(
  val,
  Y,
  threshold = 0,
  threshold_lvl = 0,
  parameter = c("shape", "scale", "quantile", "endpoint"),
  subparam_id = 0,
  quantile_lvl = 1 - (1/100),
  orthogonal = FALSE,
  X = NULL,
  x_rlvl = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  obs_weights = NULL,
  ill_defined_value = -10^6,
  init = NULL,
  hessian = TRUE,
  maxit = 1e+06,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  ...
)
```

Arguments

val	Parameter value at which to evaluate the GPD profile log-likelihood.
Y	Data observations.
threshold	GPD threshold value.
threshold_lvl	Probability level of the threshold threshold.
parameter	Parameter for which to compute the profile log-likelihood.
subparam_id	Index of the parameter coefficient for which to compute the profile log-likelihood (for conditional/non-stationary fits).
quantile_lvl	Quantile probability level for the 'quantile' parameter.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits). Columns should be variables, and rows should be observations matching Y.

<code>x_rlvl</code>	Covariate vector at which to reparametrize for the 'quantile' or 'endpoint' parametrizations (for conditional/non-stationary fits).
<code>scale_cols</code>	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).
<code>shape_cols</code>	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).
<code>obs_weights</code>	Optional observation weights for weighted likelihood.
<code>ill_defined_value</code>	Value to return if the arguments are out of support (e.g. negative scale, or non-positive arguments to logarithms).
<code>init</code>	Optional initial values for the remaining parameter's optimisation process, in the correct internal format.
<code>hessian</code>	Logical. Should a numerically differentiated Hessian matrix be returned? See <code>stats::optim()</code> for more details.
<code>maxit</code>	The maximum number of iterations. See <code>stats::optim()</code> for more details.
<code>method</code>	The optimisation method to be used. See <code>stats::optim()</code> for more details.
<code>...</code>	Other arguments passed to the <code>control</code> argument of <code>stats::optim()</code> .

Value

The GPD profile log-likelihood of parameter evaluated at `val`, given the data, as a `GPD_profML` object containing:

<code>param_val</code>	(Named) parameter value at which the GPD profile log-likelihood was evaluated.
<code>param_name</code>	Name of the evaluated profile likelihood parameter.
<code>mle_other</code>	Maximum-likelihood estimate of the other GPD parameters.
<code>loglik</code>	Profile GPD log-likelihood value of parameter evaluated at <code>val</code> , given the data.
<code>conv</code>	Whether the optimisation procedure converged. See the convergence output of <code>stats::optim()</code> for more details.
<code>hessian</code>	The loglikelihood hessian evaluated at the estimated parameters, given the data.
<code>parameter</code>	Name of the evaluated profile likelihood parameter given as argument (redundant).
<code>parametrization</code>	Likelihood parametrization.
<code>subparam_id</code>	Index of the parameter coefficient for which the profile log-likelihood was computed.
<code>id_param</code>	Index of the likelihood profile parameter, in the internal parameter vector format.

GPD_profile_loglik_curve
GPD profile log-likelihood curve

Description

GPD profile log-likelihood curve

Usage

```
GPD_profile_loglik_curve(
  Y,
  threshold = 0,
  threshold_lvl = 0,
  parameter = c("shape", "scale", "quantile", "endpoint"),
  subparam_id = 0,
  alpha = 0.05,
  quantile_lvl = 1 - (1/100),
  orthogonal = FALSE,
  X = NULL,
  x_rlvl = NULL,
  scale_cols = NULL,
  shape_cols = NULL,
  warmstart_table = NULL,
  stepsize = 0.1,
  steps_beyond_conf = 5,
  initial_MLE_para = c("classical", "same"),
  max_steps = 10000,
  obs_weights = NULL,
  ill_defined_value = -10^6,
  hessian = TRUE,
  maxit = 1e+06,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  method_prof = c("default", "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  ...
)
```

Arguments

Y	Data observations.
threshold	GPD threshold value.
threshold_lvl	Probability level of the threshold threshold.
parameter	Parameter for which to compute the profile log-likelihood.
subparam_id	Index of the parameter coefficient for which to compute the profile log-likelihood (for conditional/non-stationary fits).

alpha	Confidence alpha for the profile log-likelihood confidence interval (i.e. for the confidence line on the profile plot).
quantile_lvl	Quantile probability level for the 'quantile' parameter.
orthogonal	DEPRECATED.
X	Covariate matrix (for conditional/non-stationary fits). Columns should be variables, and rows should be observations matching Y.
x_rlvl	Covariate vector at which to reparametrize for the 'quantile' or 'endpoint' parametrizations (for conditional/non-stationary fits).
scale_cols	Column indices of X to use as covariate for the (conditional) scale parameter (for conditional/non-stationary fits).
shape_cols	Column indices of X to use as covariate for the (conditional) shape parameter (for conditional/non-stationary fits).
warmstart_table	Evaluation table from a previous run.
stepsize	Numerical size of each evaluation step, in the profile parameter's scale.
steps_beyond_conf	Number of additional steps to take (in each direction) after the profile log-likelihood values reach below the confidence line.
initial_MLE_para	Parametrization used for the initial maximum likelihood estimate (defaults to classical, for better stability).
max_steps	Maximum number of steps taken (in each direction). If the confidence line was not reached, the corresponding confidence interval endpoint will be infinite.
obs_weights	Optional observation weights for weighted likelihood.
ill_defined_value	Value to return if the arguments are out of support (e.g. negative scale, or non-positive arguments to logarithms).
hessian	Logical. Should a numerically differentiated Hessian matrix be returned? See stats::optim() for more details.
maxit	The maximum number of iterations. See stats::optim() for more details.
method	The optimisation method to be used for the initial maximum likelihood optimisation. See stats::optim() for more details.
method_prof	The optimisation method to be used for the profile likelihood optimisation. See stats::optim() for more details.
...	Other arguments passed to the control argument of stats::optim() .

Value

The GPD profile log-likelihood curve for the desired parameter, with confidence line and resulting $(1-\alpha)$ confidence interval, as a `GPD_profileLogLik` object containing:

mle	The estimated maximum likelihood GPD parameters, as a named vector (expressed in the profile parametrization).
-----	--

ci	Length-two vector containing the lower and upper endpoints of the desired profile likelihood confidence interval.
profile_loglik	Named matrix containing the profile loglikelihood value (Column 2) for each considered profile parameter value (Column 1).
conf_line	Confidence line for the desired profile likelihood confidence interval. See e.g. Coles (2001) for more details.
eval_table	Tibble (<code>tibble::tibble()</code>) containing the history of profile log-likelihood evaluation values, and related metadata.
param_name	Name of the profiled parameter (inferred, for debugging purposes).
parameter	Name of the profiled parameter (given).
parametrization	Parametrization used for the profile likelihood.
subparam_id	Index of the parameter coefficient for which the profile log-likelihood was computed.
id_param	Index of the profile parameter in the GPD parameter vector.

References

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

GPD_quantiles

Compute extreme quantile from GPD parameters

Description

Compute extreme quantile from GPD parameters

Usage

```
GPD_quantiles(quantile_lvl, threshold_lvl, threshold, scale, shape)
```

Arguments

quantile_lvl	Probability level of the desired extreme quantile.
threshold_lvl	Probability level of the GPD threshold.
threshold	GPD threshold value.
scale	Value(s) for the GPD scale parameter.
shape	Value(s) for the GPD shape parameter.

Value

The quantile value at probability level `quantile_lvl`.

Examples

```
GPD_quantiles(quantile_lvl=0.999, threshold_lvl=0.95, threshold=0, scale=1, shape=0.1)
```

plot_data_quantile_ci *Non-stationary confidence bands plot*

Description

Plot the estimated conditional quantile, return level, or endpoint, with confidence bands for non-stationary models.

Usage

```
plot_data_quantile_ci(
  quantiles,
  q_down,
  q_up,
  time_index = seq_along(quantiles),
  x_label = "X",
  y_label = "Y",
  Y = NULL,
  event_index = NULL,
  obs_label = "Fitting obs.",
  event_label = "Event",
  legend.position = "bottom"
)
```

Arguments

quantiles	Vector of estimated quantiles, return levels, or endpoints.
q_down	Vector of lower confidence band values for quantiles.
q_up	Vector of upper confidence band values for quantiles.
time_index	Vector of time indices corresponding to the quantiles. Defaults to <code>seq_along(quantiles)</code> .
x_label	Label for the x-axis. Defaults to "X".
y_label	Label for the y-axis. Defaults to "Y".
Y	(Optional) Vector of observations to add to the plot and compare to the quantiles.
event_index	(Optional) Index of an event (observation) to be highlighted.
obs_label	Label for the fitting observations. Defaults to "Fitting obs.".
event_label	(Optional) Label for the highlighted event observation. Defaults to "Event".
legend.position	Position of the legend to the side of the plot. Can be one of "bottom" (default), "right", "top", "left", or "none".

Value

A `ggplot2::ggplot()` object showing the estimated conditional quantile with confidence bands.

plot_profile_loglik_curve
Profile likelihood curve plot

Description

Plot the profile likelihood curve for the desired parameter with the confidence line.

Usage

```
plot_profile_loglik_curve(  
  profile_fct_object,  
  prop_below = NULL,  
  legend.position = "bottom"  
)
```

Arguments

`profile_fct_object` Object returned by either [GPD_profile_loglik_curve\(\)](#) or [GEV_profile_loglik_curve\(\)](#).

`prop_below` (Optional) Proportion of the distance below the confidence line to show in the plot, to crop y-axis.

`legend.position` Position of the legend to the side of the plot. Can be one of "bottom" (default), "right", "top", "left", or "none".

Value

A `ggplot2::ggplot()` object showing the profile likelihood curve for the desired parameter with the confidence line.

Index

`future::availableCores()`, [10](#), [24](#)

`GEV_change_parametrization`, [2](#)
`GEV_endpoint`, [3](#)
`GEV_log_likelihood`, [4](#)
`GEV_maxlik`, [4](#)
`GEV_profile_CI`, [6](#)
`GEV_profile_CIs_multiple`, [8](#)
`GEV_profile_loglik`, [11](#)
`GEV_profile_loglik_curve`, [12](#)
`GEV_profile_loglik_curve()`, [31](#)
`GEV_return_level`, [15](#)
`ggplot2::ggplot()`, [30](#), [31](#)
`GPD_change_parametrization`, [15](#)
`GPD_endpoint`, [16](#)
`GPD_log_likelihood`, [17](#)
`GPD_maxlik`, [18](#)
`GPD_profile_CI`, [19](#)
`GPD_profile_CIs_multiple`, [22](#)
`GPD_profile_loglik`, [25](#)
`GPD_profile_loglik_curve`, [27](#)
`GPD_profile_loglik_curve()`, [31](#)
`GPD_quantiles`, [29](#)

`plot_data_quantile_ci`, [30](#)
`plot_profile_loglik_curve`, [31](#)

`stats::optim()`, [5–8](#), [10](#), [12](#), [14](#), [19](#), [21](#), [24](#),
[26](#), [28](#)

`tibble::tibble()`, [8](#), [10](#), [14](#), [22](#), [24](#), [29](#)