Package 'fitdistcp'

April 23, 2025

Type Package

Title Distribution Fitting with Calibrating Priors for Commonly Used Distributions

Version 0.1.1

Maintainer Stephen Jewson <stephen.jewson@gmail.com>

Imports stats, mev, extraDistr, gnorm, fdrtool, pracma, rust, actuar, fExtremes

Depends R (>= 3.5.0)

Description Generates predictive distributions based on calibrating priors for various commonly used statistical models, including models with predictors. Routines for densities, probabilities, quantiles, random deviates and the parameter posterior are provided. The predictions are generated from the Bayesian prediction integral, with priors chosen to give good reliability (also known as calibration). For homogeneous models, the prior is set to the right Haar prior, giving predictions which are exactly reliable. As a result, in repeated testing, the frequencies of out-of-sample outcomes and the probabilities from the predictions agree. For other models, the prior is chosen to give good reliability. Where possible, the Bayesian prediction integral is solved exactly. Where exact solutions are not possible, the Bayesian prediction integral is solved using the Datta-Mukerjee-Ghosh-Sweeting (DMGS) asymptotic expansion. Optionally, the prediction integral can also be solved using posterior samples generated using Paul Northrop's ratio of uniforms sampling package ('rust'). Results are also generated based on maximum likelihood, for comparison purposes. Various model selection diagnostics and testing routines are included. Based on ``Reducing reliability bias in assessments of extreme weather risk using calibrating priors'', Jewson, S., Sweeting, T. and Jewson, L. (2024); <doi:10.5194/ascmo-11-1-2025>.

License MIT + file LICENSE

BugReports https://github.com/stephenjewson/fitdistcp/issues

URL https://fitdistcp.info

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Suggests knitr, rmarkdown

NeedsCompilation no

Author Stephen Jewson [aut, cre] (<https://orcid.org/0000-0002-6011-6262>)

Repository CRAN

Date/Publication 2025-04-23 09:40:02 UTC

Contents

`adhoc_dmgs_cpmethod` *Generates a comment about the method*

Description

Generates a comment about the method

Usage

```
adhoc_dmgs_cpmethod()
```

Value

String

`analytic_cpmethod` *Generates a comment about the method*

Description

Generates a comment about the method

Usage

```
analytic_cpmethod()
```

Value

String

bayesian_dq_4terms_v1 *Evaluate DMGS equation 3.3*

Description

Evaluate DMGS equation 3.3

Usage

```
bayesian_dq_4terms_v1(lddi, lddd, mu1, pidopi1, pidopi2, mu2, dim)
```

Arguments

lddi	inverse of second derivative of observed log-likelihood
lddd	third derivative of observed log-likelihood
mu1	DMGS mu1 vector
pidopi1	first part of the prior term
pidopi2	second part of the prior term
mu2	DMGS mu2 matrix
dim	number of parameters

Value

Vector

calc_revert2ml *determine revert2ml or not*

Description

determine revert2ml or not

Usage

```
calc_revert2ml(v5h, v6h, t3)
```

Arguments

v5h	fifth parameter
v6h	sixth parameter
t3	a vector of predictors for the shape

Value

Logical

cauchy_cp *Cauchy Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qcauchy_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rcauchy_cp(
```

```
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dcauchy_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pcauchy_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tcauchy_cp(n, x, d1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

`x`	a vector of training data values
`p`	a vector of probabilities at which to generate predictive quantiles
`d1`	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter
`fd2`	if `aderivs=FALSE`, the fractional delta used for numerical derivatives with respect to the second parameter
`means`	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
`waicscores`	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
`logscores`	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)

dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Cauchy distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\pi\sigma}\left(1+\left(\frac{x-\mu}{\sigma}\right)^2\right)^{-1}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),

- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d42cauchy_example_data_v1
p=c(1:9)/10
q=qlogis_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qcauchy_cp)",
main="Cauchy: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`cauchy_f1f` *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
cauchy_f1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`cauchy_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
cauchy_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

cauchy_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
cauchy_f2f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
cauchy_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_fd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

cauchy_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_fdd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

cauchy_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
cauchy_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

cauchy_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
cauchy_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
cauchy_lddd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

cauchy_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
cauchy_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

cauchy_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
cauchy_lmn(x, v1, d1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

cauchy_lmnp *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
cauchy_lmnp(x, v1, d1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

cauchy_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
cauchy_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

`cauchy_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`cauchy_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

cauchy_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
cauchy_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

cauchy_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
cauchy_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

cauchy_mu1f — *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
cauchy_mu1f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_mu2f — *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
cauchy_mu2f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`cauchy_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
cauchy_p1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`cauchy_p1_cp` *Cauchy Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.

- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qcauchy_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rcauchy_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
```

```
)

dcauchy_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pcauchy_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tcauchy_p1_cp(n, x, t, d1 = 0.01, d2 = 0.01, fd3 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that `length(t)=length(x)`
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
n0	an index for the predictor (specify either `t0` or `n0` but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter
d2	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the second parameter

fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.

- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Cauchy distribution with a predictor has probability density function

$$f(x; a, b, \sigma) = \frac{1}{\pi\sigma}\left(1 + \left(\frac{x - \mu(a, b)}{\sigma}\right)^2\right)^{-1}$$

where x is the random variable, $\mu = a + bt$ is the location parameter as a function of parameters a, b, and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson `<stephen.jewson@gmail.com>`

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d64cauchy_p1_example_data_v1_x
tt=fitdistcp::d64cauchy_p1_example_data_v1_t
p=c(1:9)/10
n0=10
```

```
q=qcauchy_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qcauchy_p1_cp)",
main="Cauchy w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`cauchy_p1_f1f` *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
cauchy_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_p1_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
cauchy_p1_f1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

cauchy_p1_f2f *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
cauchy_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`cauchy_p1_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
cauchy_p1_f2fa(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`cauchy_p1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_fd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`cauchy_p1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_fdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

Matrix

`cauchy_p1_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
cauchy_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`cauchy_p1_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
cauchy_p1_ldda(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

cauchy_p1_lddd — *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
cauchy_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

cauchy_p1_lddda — *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
cauchy_p1_lddda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`cauchy_p1_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
cauchy_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate

Value

Scalar value

cauchy_p1_lmnp   *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
cauchy_p1_lmnp(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

cauchy_p1_logf   *Logf for RUST*

Description

Logf for RUST

Usage

```
cauchy_p1_logf(params, x, t)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Scalar value.

`cauchy_p1_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

cauchy_p1_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

cauchy_p1_loglik — *Cauchy-with-p1 observed log-likelihood function*

Description

Cauchy-with-p1 observed log-likelihood function

Usage

```
cauchy_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

`cauchy_p1_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
cauchy_p1_logscores(logscores, x, t, d1, d2, fd3, aderivs = TRUE)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`cauchy_p1_means` *Cauchy distribution: RHP mean*

Description

Cauchy distribution: RHP mean

Usage

```
cauchy_p1_means(t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

---

cauchy_p1_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
cauchy_p1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_p1_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
cauchy_p1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_p1_p1f *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
cauchy_p1_p1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

cauchy_p1_p2f *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
cauchy_p1_p2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_p1_predictordata   *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
cauchy_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

cauchy_p1_waic   *Waic*

Description

Waic

Usage

```
cauchy_p1_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  lddi,
```

```
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

cauchy_p2f *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
cauchy_p2f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

cauchy_waic *Waic*

Description

Waic

Usage

```
cauchy_waic(waicscores, x, v1hat, d1, v2hat, fd2, lddi, lddd, lambdad, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`crhpflat_dmgs_cpmethod` *Generates a comment about the method*

Description

Generates a comment about the method

Usage

```
crhpflat_dmgs_cpmethod()
```

Value

String

`d100gamma_example_data_v1` *This is data to be included in my package*

Description

This is data to be included in my package

`d101invgamma_example_data_v1` *This is data to be included in my package*

Description

This is data to be included in my package

`d102invgauss_example_data_v1` *This is data to be included in my package*

Description

This is data to be included in my package

`d105burr_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

`d10exp_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

`d110gev_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

`d11pareto_k2_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

`d120gpd_k1_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

d150gev_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d150gev_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d151gev_p12_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d151gev_p12_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d152gev_p123_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

`d152gev_p123_example_data_v1_x`

This is data to be included in my package

Description

This is data to be included in my package

`d20halfnorm_example_data_v1`

This is data to be included in my package

Description

This is data to be included in my package

`d25unif_example_data_v1`

This is data to be included in my package

Description

This is data to be included in my package

`d30norm_example_data_v1`

This is data to be included in my package

Description

This is data to be included in my package

`d31norm_dmgs_example_data_v1`

This is data to be included in my package

Description

This is data to be included in my package

`d32gnorm_k3_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

`d35lnorm_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

`d36lnorm_dmgs_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

`d40logis_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

`d41lst_k3_example_data_v1`
This is data to be included in my package

Description

This is data to be included in my package

d42cauchy_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d50gumbel_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d51frechet_k1_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d52weibull_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d53gev_k3_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d55exp_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d55exp_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d56pareto_p1k2_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d56pareto_p1k2_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d60norm_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d60norm_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d61lnorm_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d61lnorm_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d62logis_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d62logis_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d63lst_p1k3_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d63lst_p1k3_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d64cauchy_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d64cauchy_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d70gumbel_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

`d70gumbel_p1_example_data_v1_x`
This is data to be included in my package

Description

This is data to be included in my package

`d71frechet_p2k1_example_data_v1_t`
This is data to be included in my package

Description

This is data to be included in my package

`d71frechet_p2k1_example_data_v1_x`
This is data to be included in my package

Description

This is data to be included in my package

`d72weibull_p1_example_data_v1_t`
This is data to be included in my package

Description

This is data to be included in my package

`d72weibull_p1_example_data_v1_x`
This is data to be included in my package

Description

This is data to be included in my package

d73weibull_p2_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d73weibull_p2_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d74gev_p1k3_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d74gev_p1k3_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d80norm_p12_example_data_v1_t1
This is data to be included in my package

Description

This is data to be included in my package

d80norm_p12_example_data_v1_t2
This is data to be included in my package

Description

This is data to be included in my package

d80norm_p12_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d81lst_p12k3_example_data_v1_t1
This is data to be included in my package

Description

This is data to be included in my package

d81lst_p12k3_example_data_v1_t2
This is data to be included in my package

Description

This is data to be included in my package

d81lst_p12k3_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d82weibull_p12_example_data_v1_t1 *This is data to be included in my package*

Description

This is data to be included in my package

d82weibull_p12_example_data_v1_t2 *This is data to be included in my package*

Description

This is data to be included in my package

d82weibull_p12_example_data_v1_x *This is data to be included in my package*

Description

This is data to be included in my package

dcauchysub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dcauchysub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dcauchy_p1 — *Cauchy-with-p1 density function*

Description

Cauchy-with-p1 density function

Usage

```
dcauchy_p1(x, t0, ymn, slope, scale, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution
log	logical for the density evaluation

Value

Vector

---

dcauchy_p1sub — *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dcauchy_p1sub(x, t, y, t0, d1, d2, fd3, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`deriv_copyfdd` *Extract the results from derivatives and put them into f2*

Description

Extract the results from derivatives and put them into f2

Usage

```
deriv_copyfdd(temp1, nx, dim)
```

Arguments

`temp1`	output from derivative calculations
`nx`	number of x values
`dim`	number of parameters

Value

3d array

deriv_copyld2 — *Extract the results from derivatives and put them into ldd*

Description

Extract the results from derivatives and put them into ldd

Usage

```
deriv_copyld2(temp1, nx, dim)
```

Arguments

`temp1`	output from derivative calculations
`nx`	number of x values
`dim`	number of parameters

Value

3d array

deriv_copyldd — *Extract the results from derivatives and put them into ldd*

Description

Extract the results from derivatives and put them into ldd

Usage

```
deriv_copyldd(temp1, nx, dim)
```

Arguments

`temp1`	output from derivative calculations
`nx`	number of x values
`dim`	number of parameters

Value

Matrix

deriv_copylddd *Extract the results from derivatives and put them into lddd*

Description

Extract the results from derivatives and put them into lddd

Usage

```
deriv_copylddd(temp1, nx, dim)
```

Arguments

temp1	output from derivative calculations
nx	number of x values
dim	number of parameters

Value

3d array

dexpsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dexpsub(x, y, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dexp_p1 *Exponential-with-p1 density function*

Description

Exponential-with-p1 density function

Usage

```
dexp_p1(x, t0, ymn, slope, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
log	logical for the density evaluation

Value

Vector

dexp_p1sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dexp_p1sub(x, t, y, t0, d1, d2, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dfrechetsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dfrechetsub(x, y, kloc, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
kloc	the known location parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dfrechet_p2k1 *Frechet_k1-with-p2 density function*

Description

Frechet_k1-with-p2 density function

Usage

```
dfrechet_p2k1(x, t0, ymn, slope, lambda, log = FALSE, kloc)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
lambda	the lambda parameter of the distribution
log	logical for the density evaluation
kloc	the known location parameter

Value

Vector

---

dfrechet_p2k1sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dfrechet_p2k1sub(x, t, y, t0, d1, d2, fd3, kloc, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgammasub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgammasub(x, y, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgevsub *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgevsub(
  x,
  y,
  ics,
  d1 = 0.01,
  fd2 = 0.01,
  d3 = 0.01,
  customprior,
  minxi,
  maxxi,
  extramodels = FALSE,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
d3	the delta used in the numerical derivatives with respect to the parameter
customprior	a custom value for the slope of the log prior at the maxlik estimate
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
extramodels	logical that indicates whether to add three additional prediction models
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dgev_k3sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgev_k3sub(x, y, d1 = 0.01, fd2 = 0.01, kshape, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgev_p1 *GEVD-with-p1: Density function*

Description

GEVD-with-p1: Density function

Usage

```
dgev_p1(x, t0, ymn, slope, sigma, xi, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
log	logical for the density evaluation

Value

Vector

dgev_p12 *GEVD-with-p1: Density function*

Description

GEVD-with-p1: Density function

Usage

```
dgev_p12(x, t1, t2, ymn, slope, sigma1, sigma2, xi, log = FALSE)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi	the shape parameter of the distribution
log	logical for the density evaluation

Value

Vector

---

dgev_p123 — *GEVD-with-p1: Density function*

Description

GEVD-with-p1: Density function

Usage

```
dgev_p123(x, t1, t2, t3, ymn, slope, sigma1, sigma2, xi1, xi2, log = FALSE)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution
log	logical for the density evaluation

Value

Vector

dgev_p123sub *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgev_p123sub(
  x,
  t1,
  t2,
  t3,
  y,
  t01,
  t02,
  t03,
  ics,
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  extramodels,
  debug,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
ics	initial conditions for the maximum likelihood search
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
d3	the delta used in the numerical derivatives with respect to the parameter

d4	the delta used in the numerical derivatives with respect to the parameter
d5	the delta used in the numerical derivatives with respect to the parameter
d6	the delta used in the numerical derivatives with respect to the parameter
extramodels	logical that indicates whether to add three additional prediction models
debug	debug flag
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dgev_p12sub *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgev_p12sub(
  x,
  t1,
  t2,
  y,
  t01,
  t02,
  ics,
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  minxi,
  maxxi,
  debug,
  extramodels = FALSE,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd

`y`	a vector of values at which to calculate the density and distribution functions
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`ics`	initial conditions for the maximum likelihood search
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter
`d4`	the delta used in the numerical derivatives with respect to the parameter
`d5`	the delta used in the numerical derivatives with respect to the parameter
`minxi`	minimum value of shape parameter xi
`maxxi`	maximum value of shape parameter xi
`debug`	debug flag
`extramodels`	logical that indicates whether to add three additional prediction models
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dgev_p1k3` *GEV-with-known-shape-with-p1 density function*

Description

GEV-with-known-shape-with-p1 density function

Usage

```
dgev_p1k3(x, t0, ymn, slope, sigma, log = FALSE, kshape)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma`	the sigma parameter of the distribution
`log`	logical for the density evaluation
`kshape`	the known shape parameter

Value

Vector

dgev_p1k3sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgev_p1k3sub(x, t, y, t0, d1, d2, fd3, kshape, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dgev_p1sub *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgev_p1sub(
  x,
  t,
  y,
  t0,
  ics,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  d4 = 0.01,
  minxi,
  maxxi,
  extramodels = FALSE,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
ics	initial conditions for the maximum likelihood search
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
d4	the delta used in the numerical derivatives with respect to the parameter
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
extramodels	logical that indicates whether to add three additional prediction models
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgnorm_k3sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgnorm_k3sub(x, y, d1 = 0.01, fd2 = 0.01, kbeta, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgpdsub *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgpdsub(
  x,
  y,
  ics,
  fd1 = 0.01,
  d2 = 0.01,
  kloc = 0,
  dlogpi = 0,
  minxi,
  maxxi,
  extramodels = FALSE,
  aderivs = TRUE
)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
dlogpi	gradient of the log prior
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
extramodels	logical that indicates whether to add three additional prediction models
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dgumbelsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgumbelsub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgumbel_p1 *Gumbel-with-p1 density function*

Description

Gumbel-with-p1 density function

Usage

```
dgumbel_p1(x, t0, ymn, slope, sigma, log = FALSE)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma`	the sigma parameter of the distribution
`log`	logical for the density evaluation

Value

Vector

dgumbel_p1sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgumbel_p1sub(x, t, y, t0, d1, d2, fd3, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dhalfnormsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dhalfnormsub(x, y, fd1 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dinvgammasub *Densities from MLE and cp*

Description

Densities from MLE and cp

Usage

```
dinvgammasub(x, y, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dinvgausssub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dinvgausssub(x, y, prior, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions
`prior`	logical indicating which prior to use
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dlnormsub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlnormsub(x, y, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dlnorm_dmgssub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlnorm_dmgssub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dlnorm_p1 *Normal-with-p1 density function*

Description

Normal-with-p1 density function

Usage

```
dlnorm_p1(x, t0, ymn, slope, sigma, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation

Value

Vector

`dlnorm_p1sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlnorm_p1sub(x, t, y, t0, debug = FALSE, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`debug`	debug flag
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dlogis2sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlogis2sub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions
`d1`	the delta used in the numerical derivatives with respect to the parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dlogis_p1 *Logistic-with-p1 density function*

Description

Logistic-with-p1 density function

Usage

```
dlogis_p1(x, t0, ymn, slope, scale, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution
log	logical for the density evaluation

Value

Vector

---

dlogis_p1sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlogis_p1sub(x, t, y, t0, d1, d2, fd3, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dlst_k3sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlst_k3sub(x, y, d1 = 0.01, fd2 = 0.01, kdf, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlst_p1k3 *LST-with-p1 density function*

Description

LST-with-p1 density function

Usage

```
dlst_p1k3(x, t0, ymn, slope, sigma, log = FALSE, kdf)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation
kdf	the known degrees of freedom parameter

Value

Vector

dlst_p1k3sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlst_p1k3sub(x, t, y, t0, d1, d2, fd3, kdf, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dmgs` *Evaluate DMGS equation 3.3*

Description

Evaluate DMGS equation 3.3

Usage

```
dmgs(lddi, lddd, mu1, pidopi, mu2, dim)
```

Arguments

`lddi`	inverse of second derivative of observed log-likelihood
`lddd`	third derivative of observed log-likelihood
`mu1`	DMGS mu1 vector
`pidopi`	derivative of log prior
`mu2`	DMGS mu2 matrix
`dim`	number of parameters

Value

Vector

dnormsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dnormsub(x, y, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dnorm_dmgssub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dnorm_dmgssub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dnorm_p1 — *Normal-with-p1 density function*

Description

Normal-with-p1 density function

Usage

```
dnorm_p1(x, t0, ymn, slope, sigma, log = FALSE)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma`	the sigma parameter of the distribution
`log`	logical for the density evaluation

Value

Vector

dnorm_p1sub — *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dnorm_p1sub(x, t, y, t0, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dnorm_p1_formula — *Linear regression formula, densities*

Description

Linear regression formula, densities

Usage

```
dnorm_p1_formula(y, ta, ta0, nx, muhat0, v3hat)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
ta	predictor residuals
ta0	predictor residual at the point being predicted
nx	length of training data
muhat0	muhat at the point being predicted
v3hat	third parameter

Value

Vector

dpareto_k2_sub — *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dpareto_k2_sub(x, y, kscale, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
kscale	the known scale parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

`dpareto_p1k2` *pareto_k1-with-p2 density function*

Description

pareto_k1-with-p2 density function

Usage

```
dpareto_p1k2(x, t0, ymn, slope, kscale, log = FALSE)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`kscale`	the known scale parameter
`log`	logical for the density evaluation

Value

Vector

`dpareto_p1k2sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dpareto_p1k2sub(x, t, y, t0, d1, d2, kscale, aderivs = TRUE, debug = FALSE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)
`debug`	debug flag

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dunif_formula` *Predictive PDFs*

Description

Predictive PDFs

Usage

```
dunif_formula(x, y)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions

Value

Two vectors

`dweibullsub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dweibullsub(x, y, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dweibull_p2` *Weibull-with-p1 density function*

Description

Weibull-with-p1 density function

Usage

```
dweibull_p2(x, t0, shape, ymn, slope, log = FALSE)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`shape`	the shape parameter of the distribution
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`log`	logical for the density evaluation

Value

Vector

`dweibull_p2sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dweibull_p2sub(x, t, y, t0, fd1, d2, d3, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

exp_cp *Exponential Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qexp_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rexp_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE, aderivs = TRUE)

dexp_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

pexp_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

texp_cp(n, x, debug = FALSE)
```

Arguments

`x`	a vector of training data values
`p`	a vector of probabilities at which to generate predictive quantiles
`fd1`	if `aderivs=FALSE`, the fractional delta used for numerical derivatives with respect to the first parameter
`means`	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
`waicscores`	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
`logscores`	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
`rust`	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
`nrust`	the number of posterior samples used in the RUST calculations
`debug`	logical for turning on debug messages
`aderivs`	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
`n`	the number of random samples required
`mlcp`	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
`y`	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The exponential distribution has exceedance distribution function

$$S(x;\lambda) = \exp(-\lambda x)$$

where $x \geq 0$ is the random variable and $\lambda > 0$ is the rate parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\lambda) \propto \frac{1}{\lambda}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson `<stephen.jewson@gmail.com>`

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),

- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),
- Weibull with linear predictor on the scale (`weibull_p2`),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d10exp_example_data_v1
p=c(1:9)/10
q=qexp_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qexp_cp)",
main="Exponential: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

exp_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
exp_f1f(y, v1, fd1)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

exp_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
exp_f1fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

`exp_f2f` *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
exp_f2f(y, v1, fd1)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`exp_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
exp_f2fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

`exp_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_fd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

`exp_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_fdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

exp_l111 *Third derivative of the normalized log-likelihood*

Description

Third derivative of the normalized log-likelihood

Usage

```
exp_l111(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

exp_ldd *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
exp_ldd(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

exp_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
exp_ldda(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

exp_lddd *Third derivative tensor of the log-likelihood*

Description

Third derivative tensor of the log-likelihood

Usage

```
exp_lddd(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

exp_lddda — *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
exp_lddda(x, v1)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter

Value

3d array

exp_logf — *Logf for RUST*

Description

Logf for RUST

Usage

```
exp_logf(params, x)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values

Value

Scalar value.

exp_logfdd

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_logfdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

exp_logfddd

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_logfddd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

3d array

---

exp_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
exp_logscores(logscores, x)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values

Value

Two scalars

---

exp_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
exp_p1fa(x, v1)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter

Value

Vector

exp_p1_cp — *Exponential Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qexp_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
```

```
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rexp_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dexp_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pexp_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)
```

```
texp_p1_cp(n, x, t, d1 = 0.01, d2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	the fractional delta used in the numerical derivatives with respect to the location parameter
d2	the fractional delta used in the numerical derivatives with respect to the slope parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.

- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The exponential distribution with a predictor has exceedance distribution function

$$S(x; a, b) = \exp(-x\lambda(a, b))$$

where $x \geq 0$ is the random variable and $\lambda(a, b) = e^{-a-bt}$ is the rate parameter, modelled as a function of the parameters a, b and a predictor t.

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b) \propto 1$$

. as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d55exp_p1_example_data_v1_x
tt=fitdistcp::d55exp_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qexp_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qexp_p1_cp)",
main="Exponential w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

exp_p1_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
exp_p1_f1f(y, t0, v1, d1, v2, d2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

exp_p1_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
exp_p1_f1fa(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Vector

exp_p1_f2f *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
exp_p1_f2f(y, t0, v1, d1, v2, d2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

exp_p1_f2fa — *The second derivative of the density*

Description

The second derivative of the density

Usage

```
exp_p1_f2fa(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_fd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Vector

exp_p1_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_fdd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
exp_p1_ldd(x, t, v1, d1, v2, d2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`exp_p1_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
exp_p1_ldda(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

`exp_p1_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
exp_p1_lddd(x, t, v1, d1, v2, d2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

---

exp_p1_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
exp_p1_lddda(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

3d array

---

exp_p1_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
exp_p1_lmn(x, t, v1, d1, v2, d2, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`exp_p1_lmnp` *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
exp_p1_lmnp(x, t, v1, d1, v2, d2, mm, nn, rr)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate
`rr`	an index for which derivative to calculate

Value

Scalar value

`exp_p1_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
exp_p1_logf(params, x, t)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Scalar value.

`exp_p1_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_logfdd(x, t, v1, v2)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter

Value

Matrix

`exp_p1_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_logfddd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

3d array

`exp_p1_loglik` *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
exp_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

exp_p1_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
exp_p1_logscores(logscores, x, t, d1, d2, aderivs)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

exp_p1_means *exp distribution: RHP means*

Description

exp distribution: RHP means

Usage

```
exp_p1_means(means, t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ml_params`	parameters
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad_rhp`	derivative of the log RHP prior
`nx`	length of training data
`dim`	number of parameters

Value

Two scalars

`exp_p1_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
exp_p1_mu1f(alpha, t0, v1, d1, v2, d2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

exp_p1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
exp_p1_mu1fa(alpha, t, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Vector

exp_p1_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
exp_p1_mu2f(alpha, t0, v1, d1, v2, d2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

exp_p1_mu2fa — *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
exp_p1_mu2fa(alpha, t, v1, v2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter

Value

Matrix

exp_p1_p1f — *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
exp_p1_p1f(y, t0, v1, d1, v2, d2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

exp_p1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
exp_p1_p1fa(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Vector

exp_p1_p2f *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
exp_p1_p2f(y, t0, v1, d1, v2, d2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

exp_p1_p2fa — *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
exp_p1_p2fa(x, t, v1, v2)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter

Value

Matrix

exp_p1_pd — *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_pd(x, t, v1, v2)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter

Value

Vector

`exp_p1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_pdd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

`exp_p1_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
exp_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
params	model parameters for calculating logf

Value

Two vectors

`exp_p1_waic` *Waic*

Description

Waic

Usage

```
exp_p1_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1hat`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad`	derivative of the log prior
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

exp_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
exp_p2fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

exp_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_pd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

exp_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_pdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

---

exp_waic *Waicscores*

Description

Waicscores

Usage

```
exp_waic(waicscores, x, v1hat, fd1, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

fixgevrange *Deal with situations in which the user wants d or p outside the GEV range*

Description

Deal with situations in which the user wants d or p outside the GEV range

Usage

```
fixgevrange(y, v1, v2, v3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

fixgpdrange *Deal with situations in which the user wants d or p outside the GPD range*

Description

Deal with situations in which the user wants d or p outside the GPD range

Usage

```
fixgpdrange(y, v1, v2, v3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`frechet_k1_cp` *Frechet Distribution Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q`, `r`, `d`, `p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qfrechet_k1_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kloc = 0,
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rfrechet_k1_cp(
  n,
  x,
  kloc = 0,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dfrechet_k1_cp(
  x,
  y = x,
  kloc = 0,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pfrechet_k1_cp(
  x,
  y = x,
  kloc = 0,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tfrechet_k1_cp(n, x, kloc = 0, fd1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kloc	the known location parameter
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter

means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Frechet distribution has distribution function

$$F(x;\sigma,\lambda) = \exp\left(-\left(\frac{x-\mu}{\sigma}\right)^{-\lambda}\right)$$

where $x > \mu$ is the random variable, $\sigma > 0, \lambda > 0$ are the parameters and we consider μ to be known (hence the k1 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma,\lambda) \propto \frac{1}{\sigma\lambda}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d51frechet_k1_example_data_v1
p=c(1:9)/10
q=qfrechet_k1_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
```

```
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qfrechet_k1_cp)",
main="Frechet: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`frechet_k1_f1f` *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
frechet_k1_f1f(y, v1, fd1, v2, fd2, kloc)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_k1_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
frechet_k1_f1fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

`frechet_k1_f2f` *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
frechet_k1_f2f(y, v1, fd1, v2, fd2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

`frechet_k1_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
frechet_k1_f2fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

`frechet_k1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

frechet_k1_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

frechet_k1_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
frechet_k1_ldd(x, v1, fd1, v2, fd2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Square scalar matrix

`frechet_k1_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
frechet_k1_ldda(x, v1, v2, kloc)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_k1_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
frechet_k1_lddd(x, v1, fd1, v2, fd2, kloc)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

Cubic scalar array

`frechet_k1_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
frechet_k1_lddda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

3d array

`frechet_k1_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
frechet_k1_lmn(x, v1, fd1, v2, fd2, kloc, mm, nn)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate

Value

Scalar value

`frechet_k1_lmnp` *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
frechet_k1_lmnp(x, v1, fd1, v2, fd2, kloc, mm, nn, rr)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate
`rr`	an index for which derivative to calculate

Value

Scalar value

frechet_k1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
frechet_k1_logf(params, x, kloc)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kloc	the known location parameter

Value

Scalar value.

frechet_k1_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`frechet_k1_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_logfddd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

`frechet_k1_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
frechet_k1_mu1f(alpha, v1, fd1, v2, fd2, kloc)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_k1_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
frechet_k1_mu1fa(alpha, v1, v2, kloc)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

Vector

`frechet_k1_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
frechet_k1_mu2f(alpha, v1, fd1, v2, fd2, kloc)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

3d array

`frechet_k1_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
frechet_k1_mu2fa(alpha, v1, v2, kloc)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_k1_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
frechet_k1_p1f(y, v1, fd1, v2, fd2, kloc)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_k1_p1fa` *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
frechet_k1_p1fa(x, v1, v2, kloc)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

Vector

`frechet_k1_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
frechet_k1_p2f(y, v1, fd1, v2, fd2, kloc)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

3d array

`frechet_k1_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
frechet_k1_p2fa(x, v1, v2, kloc)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_k1_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_pd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`frechet_k1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_pdd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`frechet_k1_waic` *Waic*

Description

Waic

Usage

```
frechet_k1_waic(
  waicscores,
  x,
  v1hat,
  fd1,
  v2hat,
  fd2,
  kloc,
```

```
    lddi,
    lddd,
    lambdad,
    aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

frechet_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
frechet_loglik(vv, x, kloc)
```

Arguments

vv	parameters
x	a vector of training data values
kloc	the known location parameter

Value

Scalar value.

frechet_logscores	*Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
frechet_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, kloc, aderivs)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

frechet_means	*MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
frechet_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2, kloc)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kloc	the known location parameter

Value

Two scalars

---

frechet_p2k1_cp *Frechet Distribution with Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qfrechet_p2k1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  kloc = 0,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rfrechet_p2k1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kloc = 0,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dfrechet_p2k1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
```

```
    fd3 = 0.01,
    kloc = 0,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  pfrechet_p2k1_cp(
    x,
    t,
    t0 = NA,
    n0 = NA,
    y = x,
    d1 = 0.01,
    d2 = 0.01,
    fd3 = 0.01,
    kloc = 0,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  tfrechet_p2k1_cp(
    n,
    x,
    t,
    d1 = 0.01,
    d2 = 0.01,
    fd3 = 0.01,
    kloc = 0,
    debug = FALSE
  )
```

Arguments

`x`	a vector of training data values
`t`	a vector of predictors, such that `length(t)=length(x)`
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`n0`	an index for the predictor (specify either `t0` or `n0` but not both)
`p`	a vector of probabilities at which to generate predictive quantiles
`d1`	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter
`d2`	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the second parameter

fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
kloc	the known location parameter
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Frechet distribution with predictor has distribution function

$$F(x; a, b, \lambda) = \exp\left(-\left(\frac{x-\mu}{\sigma(a,b)}\right)^{-\lambda}\right)$$

where $x > \mu$ is the random variable, $\sigma = e^{a+bt}$ is the scale parameter, modelled as a function of parameters a, b and predictor t, and $\lambda > 0$ is the shape parameter. We consider μ to be known (hence the `k1` in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b) \propto 1$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d71frechet_p2k1_example_data_v1_x
tt=fitdistcp::d71frechet_p2k1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qfrechet_p2k1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qfrechet_p2k1_cp)",
main="Frechet w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

frechet_p2k1_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
frechet_p2k1_f1f(y, t0, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

`frechet_p2k1_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
frechet_p2k1_f1fa(x, t, v1, v2, v3, kloc)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`kloc`	the known location parameter

Value

Vector

`frechet_p2k1_f2f` *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
frechet_p2k1_f2f(y, t0, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

---

frechet_p2k1_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
frechet_p2k1_f2fa(x, t, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

`frechet_p2k1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_fd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Vector

`frechet_p2k1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_fdd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`frechet_p2k1_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
frechet_p2k1_ldd(x, t, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

Square scalar matrix

frechet_p2k1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
frechet_p2k1_ldda(x, t, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
frechet_p2k1_lddd(x, t, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Cubic scalar array

---

frechet_p2k1_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
frechet_p2k1_lddda(x, t, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

3d array

frechet_p2k1_lmn — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
frechet_p2k1_lmn(x, t, v1, d1, v2, d2, v3, fd3, kloc, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

frechet_p2k1_lmnp — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
frechet_p2k1_lmnp(x, t, v1, d1, v2, d2, v3, fd3, kloc, mm, nn, rr)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate
`rr`	an index for which derivative to calculate

Value

Scalar value

`frechet_p2k1_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
frechet_p2k1_logf(params, x, t, kloc)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`kloc`	the known location parameter

Value

Scalar value.

frechet_p2k1_logfdd

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

frechet_p2k1_logfddd

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

---

frechet_p2k1_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
frechet_p2k1_loglik(vv, x, t, kloc)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kloc	the known location parameter

Value

Scalar value.

`frechet_p2k1_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
frechet_p2k1_logscores(logscores, x, t, d1, d2, fd3, kloc, aderivs = TRUE)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`frechet_p2k1_means` *frechet_k1 distribution: RHP mean*

Description

frechet_k1 distribution: RHP mean

Usage

```
frechet_p2k1_means(
  means,
  t0,
  ml_params,
  lddi,
  lddd,
  lambdad_rhp,
  nx,
  dim,
  kloc
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kloc	the known location parameter

Value

Two scalars

---

frechet_p2k1_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
frechet_p2k1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_p2k1_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
frechet_p2k1_mu1fa(alpha, t, v1, v2, v3, kloc)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`kloc`	the known location parameter

Value

Vector

frechet_p2k1_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
frechet_p2k1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

frechet_p2k1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
frechet_p2k1_mu2fa(alpha, t, v1, v2, v3, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

---

frechet_p2k1_p1f *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
frechet_p2k1_p1f(y, t0, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
frechet_p2k1_p1fa(x, t, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Vector

frechet_p2k1_p2f *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
frechet_p2k1_p2f(y, t0, v1, d1, v2, d2, v3, fd3, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

---

frechet_p2k1_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
frechet_p2k1_p2fa(x, t, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

`frechet_p2k1_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_pd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

`frechet_p2k1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_pdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

---

frechet_p2k1_predictordata
Predicted Parameter and Generalized Residuals

Description

Predicted Parameter and Generalized Residuals

Usage

```
frechet_p2k1_predictordata(predictordata, x, t, t0, params, kloc)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kloc	the known location parameter

Value

Two vectors

frechet_p2k1_waic *Waic*

Description

Waic

Usage

```
frechet_p2k1_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  kloc,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

gamma_cp *Gamma Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgamma_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  prior = "type 1",
  debug = FALSE,
```

```
  aderivs = TRUE
)

rgamma_cp(
  n,
  x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgamma_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pgamma_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgamma_cp(n, x, fd1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
prior	logical indicating which prior to use
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Gamma distribution has probability density function

$$f(x;\alpha,\sigma)=\frac{1}{\sigma^{\alpha}\Gamma(\alpha)}x^{\alpha-1}e^{-x/\sigma}$$

where $x \geq 0$ is the random variable and $\alpha > 0, \sigma > 0$ are the parameters.

The calibrating prior we use is

$$\pi(\alpha,\sigma) \propto \frac{1}{\alpha\sigma}$$

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d100gamma_example_data_v1
p=c(1:9)/10
q=qgamma_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgamma_cp)",
main="Gamma: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

gamma_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
gamma_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gamma_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gamma_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`gamma_f2f` *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
gamma_f2f(y, v1, fd1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gamma_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gamma_f2fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

gamma_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gamma_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gamma_gg — *Second derivative matrix of the expected log-likelihood*

Description

Second derivative matrix of the expected log-likelihood

Usage

```
gamma_gg(v1, fd1, v2, fd2)
```

Arguments

v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gamma_gmn — *One component of the second derivative of the expected log-likelihood*

Description

One component of the second derivative of the expected log-likelihood

Usage

```
gamma_gmn(alpha, v1, fd1, v2, fd2, mm, nn)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`gamma_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gamma_ldd(x, v1, fd1, v2, fd2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`gamma_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gamma_ldda(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

`gamma_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gamma_lddd(x, v1, fd1, v2, fd2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

`gamma_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gamma_lddda(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

3d array

`gamma_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gamma_lmn(x, v1, fd1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`gamma_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gamma_lmnp(x, v1, fd1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

gamma_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gamma_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

gamma_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gamma_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

gamma_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
gamma_loglik(vv, x)
```

Arguments

`vv`	parameters
`x`	a vector of training data values

Value

Scalar value.

gamma_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gamma_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`gamma_means` *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
gamma_means(means, ml_params, lddi, lddd, lambdad_cp, nx, dim = 2)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`ml_params`	parameters
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad_cp`	derivative of the log prior
`nx`	length of training data
`dim`	number of parameters

Value

Two scalars

`gamma_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gamma_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`gamma_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gamma_mu2f(alpha, v1, fd1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gamma_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
gamma_p1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

gamma_p2f *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
gamma_p2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

gamma_waic *Waic*

Description

Waic

Usage

```
gamma_waic(waicscores, x, v1hat, fd1, v2hat, fd2, lddi, lddd, lambdad, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gev_checkmle *Check MLE*

Description

Check MLE

Usage

```
gev_checkmle(ml_params, minxi, maxxi)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

---

gev_cp *Generalized Extreme Value Distribution, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0),
  d1 = 0.01,
```

```
  fd2 = 0.01,
  d3 = 0.01,
  fdalpha = 0.01,
  minxi = -1,
  maxxi = 999,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  customprior = 0,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  pwm = FALSE,
  debug = FALSE,
  aderivs = TRUE
)

rgev_cp(
  n,
  x,
  ics = c(0, 0, 0),
  d1 = 0.01,
  fd2 = 0.01,
  d3 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgev_cp(
  x,
  y = x,
  ics = c(0, 0, 0),
  d1 = 0.01,
  fd2 = 0.01,
  d3 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
```

```
)

pgev_cp(
  x,
  y = x,
  ics = c(0, 0, 0),
  d1 = 0.01,
  fd2 = 0.01,
  d3 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgev_cp(
  n,
  x,
  ics = c(0, 0, 0),
  d1 = 0.01,
  fd2 = 0.01,
  d3 = 0.01,
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
d3	if aderivs=FALSE, the delta used for numerical derivatives with respect to the third parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
customprior	a custom value for the slope of the log prior at the maxlik estimate
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
pwm	logical for whether to include PWM results (longer runtime)
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.

- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution has distribution function

$$F(x;\mu,\sigma,\xi) = \exp\left(-t(x;\mu,\sigma,\xi)\right)$$

where

$$t(x;\mu,\sigma,\xi) = \begin{cases} \left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable and $\mu, \sigma > 0, \xi$ are the parameters.

The calibrating prior we use is given by

$$\pi(\mu,\sigma,\xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
shape=-0.4
x=fitdistcp::d110gev_example_data_v1
p=c(1:9)/10
q=qgev_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_cp)",
main="GEVD: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gev_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
gev_f1f(y, v1, d1, v2, fd2, v3, d3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gev_f1fa(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
gev_f2f(y, v1, d1, v2, fd2, v3, d3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gev_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gev_f2fa(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gev_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

---

gev_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

gev_ggd_mev *Derivative of expected information matrix, based on MEV routine gev.infomat*

Description

Derivative of expected information matrix, based on MEV routine gev.infomat

Usage

```
gev_ggd_mev(v1, d1, v2, fd2, v3, d3)
```

Arguments

`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`gev_ggid_mev` *Derivative of inverse expected information matrix, based on MEV routine gev.infomat*

Description

Derivative of inverse expected information matrix, based on MEV routine gev.infomat

Usage

```
gev_ggid_mev(v1, d1, v2, fd2, v3, d3)
```

Arguments

`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

gev_k12_ppm_minusloglik *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
gev_k12_ppm_minusloglik(x)
```

Arguments

x	a vector of training data values

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

gev_k3_cp *Generalized Extreme Value Distribution with Known Shape, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_k3_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  fdalpha = 0.01,
  kshape = 0,
  means = FALSE,
  waicscores = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rgev_k3_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  kshape = 0,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgev_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kshape = 0,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pgev_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kshape = 0,
```

```
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgev_k3_cp(n, x, d1 = 0.01, fd2 = 0.01, kshape = 0, debug = FALSE)
```

Arguments

`x`	a vector of training data values
`p`	a vector of probabilities at which to generate predictive quantiles
`d1`	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter
`fd2`	if `aderivs=FALSE`, the fractional delta used for numerical derivatives with respect to the second parameter
`fdalpha`	if `pdf=TRUE`, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
`kshape`	the known shape parameter
`means`	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
`waicscores`	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
`pdf`	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
`dmgs`	logical that indicates whether DMGS calculations should be run or not (longer run time)
`rust`	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
`nrust`	the number of posterior samples used in the RUST calculations
`debug`	logical for turning on debug messages
`aderivs`	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
`n`	the number of random samples required
`mlcp`	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
`y`	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.

- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with known shape has distribution function

$$F(x;\mu,\sigma) = \exp\left(-t(x;\mu,\sigma)\right)$$

where

$$t(x;\mu,\sigma) = \begin{cases} \left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu, \sigma > 0$ are the parameters and ξ is known (hence the `k3` in the name).

The calibrating prior we use is given by

$$\pi(\mu,\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),

- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
kshape=-0.4
x=fitdistcp::d53gev_k3_example_data_v1
p=c(1:9)/10
q=qgev_k3_cp(x,p,kshape=kshape,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_k3_cp)",
main="GEV: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
muhat=q$ml_params[1]
sghat=q$ml_params[2]
xi=kshape
qmax=ifelse(xi<0,muhat-sghat/xi,Inf)
cat(" ml_params=",q$ml_params,",")
cat(" qmax=",qmax,"\n")
```

gev_k3_f1f — *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
gev_k3_f1f(y, v1, d1, v2, fd2, kshape)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Matrix

gev_k3_f1fa — *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gev_k3_f1fa(x, v1, v2, kshape)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Vector

gev_k3_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
gev_k3_f2f(y, v1, d1, v2, fd2, kshape)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

3d array

gev_k3_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gev_k3_f2fa(x, v1, v2, kshape)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Matrix

`gev_k3_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gev_k3_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_k3_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_k3_ldd(x, v1, d1, v2, fd2, kshape)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Square scalar matrix

gev_k3_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_k3_ldda(x, v1, v2, kshape)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Matrix

`gev_k3_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gev_k3_lddd(x, v1, d1, v2, fd2, kshape)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kshape`	the known shape parameter

Value

Cubic scalar array

`gev_k3_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_k3_lddda(x, v1, v2, kshape)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kshape`	the known shape parameter

Value

3d array

`gev_k3_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_k3_lmn(x, v1, d1, v2, fd2, kshape, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`gev_k3_lmnp` *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
gev_k3_lmnp(x, v1, d1, v2, fd2, kshape, mm, nn, rr)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kshape`	the known shape parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate
`rr`	an index for which derivative to calculate

Value

Scalar value

`gev_k3_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_k3_logf(params, x, kshape)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`kshape`	the known shape parameter

Value

Scalar value.

gev_k3_logfdd	*Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_k3_logfddd	*Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_k3_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
gev_k3_loglik(vv, x, kshape)
```

Arguments

vv	parameters
x	a vector of training data values
kshape	the known shape parameter

Value

Scalar value.

gev_k3_means *MLE and RHP means*

Description

MLE and RHP means

Usage

```
gev_k3_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2, kshape)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kshape	the known shape parameter

Value

Two scalars

`gev_k3_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gev_k3_mu1f(alpha, v1, d1, v2, fd2, kshape)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kshape`	the known shape parameter

Value

Matrix

`gev_k3_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_k3_mu1fa(alpha, v1, v2, kshape)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`v2`	second parameter
`kshape`	the known shape parameter

Value

Vector

---

gev_k3_mu2f — *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gev_k3_mu2f(alpha, v1, d1, v2, fd2, kshape)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kshape`	the known shape parameter

Value

3d array

---

gev_k3_mu2fa — *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_k3_mu2fa(alpha, v1, v2, kshape)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`v2`	second parameter
`kshape`	the known shape parameter

Value

Matrix

`gev_k3_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gev_k3_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gev_k3_waic` *Waic*

Description

Waic

Usage

```
gev_k3_waic(
  waicscores,
  x,
  v1hat,
  d1,
  v2hat,
  fd2,
  kshape,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`v1hat`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kshape`	the known shape parameter
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad`	derivative of the log prior
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gev_ld12a *The combined derivative of the normalized log-likelihood*

Description

The combined derivative of the normalized log-likelihood

Usage

```
gev_ld12a(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_lda *The first derivative of the normalized log-likelihood*

Description

The first derivative of the normalized log-likelihood

Usage

```
gev_lda(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gev_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_ldd(x, v1, d1, v2, fd2, v3, d3)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`gev_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_ldda(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

gev_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gev_lddd(x, v1, d1, v2, fd2, v3, d3)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

---

gev_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_lddda(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

---

gev_lmn — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_lmn(x, v1, d1, v2, fd2, v3, d3, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gev_lmnp *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_lmnp(x, v1, d1, v2, fd2, v3, d3, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gev_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

---

gev_logfd *First derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_logfd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

---

gev_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

gev_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_loglik	*log-likelihood function*

Description

log-likelihood function

Usage

```
gev_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

gev_means	*Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_means(
  means,
  ml_params,
  lddi,
  lddi_k3,
  lddd,
  lddd_k3,
  lambdad_flat,
  lambdad_rh_mle,
  lambdad_rh_flat,
  lambdad_jp,
  lambdad_custom,
  nx,
  dim = 3
)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`ml_params`	parameters
`lddi`	inverse observed information matrix
`lddi_k3`	inverse observed information matrix, fixed shape parameter
`lddd`	third derivative of log-likelihood
`lddd_k3`	third derivative of log-likelihood, fixed shape parameter
`lambdad_flat`	derivative of the log flat prior
`lambdad_rh_mle`	derivative of the log CRHP-MLE prior
`lambdad_rh_flat`	derivative of the log CRHP-FLAT prior
`lambdad_jp`	derivative of the log JP prior
`lambdad_custom`	custom value of the derivative of the log prior
`nx`	length of training data
`dim`	number of parameters

Value

Two scalars

`gev_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gev_mu1f(alpha, v1, d1, v2, fd2, v3, d3)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`gev_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_mu1fa(alpha, v1, v2, v3)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`gev_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gev_mu2f(alpha, v1, d1, v2, fd2, v3, d3)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gev_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_mu2fa(alpha, v1, v2, v3)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`gev_p123_checkmle` *Check MLE*

Description

Check MLE

Usage

```
gev_p123_checkmle(ml_params, minxi, maxxi, t1, t2, t3)
```

Arguments

`ml_params`	parameters
`minxi`	minimum value of shape parameter xi
`maxxi`	maximum value of shape parameter xi
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`t3`	a vector of predictors for the shape

Value

No return value (just a message to the screen).

---

gev_p123_cp *Generalized Extreme Value Distribution with Three Predictors, Predictions based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  p = seq(0.1, 0.9, 0.1),
```

```
  ics = c(0, 0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  fdalpha = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rgev_p123_cp(
  n,
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  ics = c(0, 0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE,
```

```
  aderivs = TRUE
)

dgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 10,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
```

```
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tgev_p123_cp(
  n,
  x,
  t1,
  t2,
  t3,
  ics = c(0, 0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  d6 = 0.01,
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t3	a vector of predictors for the shape, such that length(t3)=length(x)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)
n03	an index for the predictor (specify either t03 or n03 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter

d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
d3	if aderivs=FALSE, the delta used for numerical derivatives with respect to the third parameter
d4	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fourth parameter
d5	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fifth parameter
d6	if aderivs=FALSE, the delta used for numerical derivatives with respect to the sixth parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.

- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with three predictors has distribution function

$$F(x; a_1, b_1, a_2, b_2, a_3, b_3) = \exp\left(-t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi(a_3, b_3)\right)$$

where

$$t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi(a_3, b_3)) = \begin{cases} \left[1 + \xi(a_3, b_3)\left(\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right)\right]^{-1/\xi(a_3,b_3)} & \text{if } \xi(a_3, b_3) \neq 0 \\ \exp\left(-\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right) & \text{if } \xi(a_3, b_3) = 0 \end{cases}$$

where x is the random variable, $\mu = a_1 + b_1 t_1$ is the location parameter, modelled as a function of parameters a_1, b_1 and predictor t_1, $\sigma = e^{a_2 + b_2 t_2}$ is the scale parameter, modelled as a function of parameters a_2, b_2 and predictor t_2, and $\xi = a_3 + b_3 t_3$ is the shape parameter, modelled as a function of parameters a_3, b_3 and predictor t_3.

The calibrating prior we use is given by

$$\pi(a_1, b_1, a_2, b_2, a_3, b_3) \propto 1$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d152gev_p123_example_data_v1_x
tt=fitdistcp::d152gev_p123_example_data_v1_t
t1=tt[,1]
t2=tt[,2]
t3=tt[,3]
p=c(1:9)/10
n01=10
n02=10
n03=10
q=qgev_p123_cp(x=x,t1=t1,t2=t2,t3=t3,n01=n01,n02=n02,n03=n03,t01=NA,t02=NA,t03=NA,
p=p,rust=FALSE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
```

```
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_p123_cp)",
main="GEVD w/ p123: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

`gev_p123_f1f` *DMGS equation 2.1, f1 term, fixed shape parameter DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term, fixed shape parameter DMGS equation 2.1, f1 term

Usage

```
gev_p123_f1f(y, t01, t02, t03, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, v6, d6)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`t03`	a single value of the predictor (specify either `t03` or `n03` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter
`v4`	fourth parameter
`d4`	the delta used in the numerical derivatives with respect to the parameter
`v5`	fifth parameter
`d5`	the delta used in the numerical derivatives with respect to the parameter
`v6`	sixth parameter
`d6`	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_p123_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gev_p123_f1fa(x, t01, t02, t03, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p123_f2f *GEVD-with-p1: DMGS equation 1.2 f2 term*

Description

GEVD-with-p1: DMGS equation 1.2 f2 term

Usage

```
gev_p123_f2f(y, t01, t02, t03, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, v6, d6)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

gev_p123_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gev_p123_f2fa(x, t01, t02, t03, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter

v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

`gev_p123_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_fd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p123_fdd   *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_fdd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_ldd   *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_p123_ldd(x, t1, t2, t3, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, v6, d6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

---

gev_p123_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p123_ldda(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter

v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

`gev_p123_lddd` *Third derivative tensor of the normalized log-likelihood, with fixed shape parameter*

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter

Usage

```
gev_p123_lddd(x, t1, t2, t3, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, v6, d6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

gev_p123_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p123_lddda(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

3d array

gev_p123_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p123_lmn(
  x,
  t1,
  t2,
  t3,
  v1,
  d1,
  v2,
  d2,
  v3,
  d3,
  v4,
  d4,
  v5,
  d5,
  v6,
  d6,
  mm,
  nn
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`gev_p123_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p123_lmnp(
  x,
  t1,
  t2,
  t3,
  v1,
  d1,
  v2,
  d2,
  v3,
  d3,
  v4,
  d4,
  v5,
  d5,
  v6,
  d6,
  mm,
  nn,
  rr
)
```

Arguments

`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`t3`	a vector of predictors for the shape
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

gev_p123_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_p123_logf(params, x, t1, t2, t3)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape

Value

Scalar value.

gev_p123_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_logfdd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_logfddd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`t3`	a vector of predictors for the shape
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter
`v5`	fifth parameter
`v6`	sixth parameter

Value

3d array

`gev_p123_loglik` *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
gev_p123_loglik(vv, x, t1, t2, t3)
```

Arguments

`vv`	parameters
`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`t3`	a vector of predictors for the shape

Value

Scalar value.

`gev_p123_means` *Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p123_means(means, t01, t02, t03, ml_params, nx)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t01	a single value of the predictor (specify either `t01` or `n01` but not both)
t02	a single value of the predictor (specify either `t02` or `n02` but not both)
t03	a single value of the predictor (specify either `t03` or `n03` but not both)
ml_params	parameters
nx	length of training data

Value

Two scalars

`gev_p123_mu1f` *GEVD-with-p1: DMGS equation 3.3 mu1 term*

Description

GEVD-with-p1: DMGS equation 3.3 mu1 term

Usage

```
gev_p123_mu1f(
  alpha,
  t01,
  t02,
  t03,
  v1,
  d1,
  v2,
```

```
  d2,
  v3,
  d3,
  v4,
  d4,
  v5,
  d5,
  v6,
  d6
)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

gev_p123_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p123_mu1fa(alpha, t01, t02, t03, v1, v2, v3, v4, v5, v6)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

---

gev_p123_mu2f *GEVD-with-p1: DMGS equation 3.3 mu2 term*

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term

Usage

```
gev_p123_mu2f(
  alpha,
  t01,
  t02,
  t03,
  v1,
  d1,
  v2,
  d2,
  v3,
  d3,
  v4,
  d4,
  v5,
  d5,
  v6,
  d6
)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

gev_p123_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p123_mu2fa(alpha, t01, t02, t03, v1, v2, v3, v4, v5, v6)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter

v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

`gev_p123_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_pd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p123_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_pdd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_predictordata *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p123_predictordata(x, t1, t2, t3, t01, t02, t03, params)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
params	model parameters for calculating logf

Value

Two vectors

---

gev_p123_setics *Set initial conditions*

Description

Set initial conditions

Usage

```
gev_p123_setics(x, t1, t2, t3, ics)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ics	initial conditions for the maximum likelihood search

Value

Vector

gev_p123_waic *Waic*

Description

Waic

Usage

```
gev_p123_waic(
  waicscores,
  x,
  t1,
  t2,
  t3,
  v1h,
  d1,
  v2h,
  d2,
  v3h,
  d3,
  v4h,
  d4,
  v5h,
  d5,
  v6h,
  d6,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1h	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2h	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

v3h	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4h	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5h	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6h	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`gev_p12k3_f1f` *DMGS equation 2.1, f1 term, fixed shape parameter*

Description

DMGS equation 2.1, f1 term, fixed shape parameter

Usage

```
gev_p12k3_f1f(y, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either `t01` or `n01` but not both)
t02	a single value of the predictor (specify either `t02` or `n02` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

Matrix

`gev_p12k3_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gev_p12k3_f1fa(x, t, v1, v2, v3, v4, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Vector

`gev_p12k3_f2f` *GEVD-with-p1: DMGS equation 1.2 f2 term*

Description

GEVD-with-p1: DMGS equation 1.2 f2 term

Usage

```
gev_p12k3_f2f(y, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

3d array

---

gev_p12k3_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gev_p12k3_f2fa(x, t, v1, v2, v3, v4, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Matrix

`gev_p12k3_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_fd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

`gev_p12k3_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_fdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

`gev_p12k3_ldd` *Second derivative matrix of the normalized log-likelihood, with fixed shape parameter*

Description

Second derivative matrix of the normalized log-likelihood, with fixed shape parameter

Usage

```
gev_p12k3_ldd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

Square scalar matrix

`gev_p12k3_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p12k3_ldda(x, t, v1, v2, v3, v4, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Matrix

`gev_p12k3_lddd` *Third derivative tensor of the normalized log-likelihood, with fixed shape parameter*

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter

Usage

```
gev_p12k3_lddd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

Cubic scalar array

---

gev_p12k3_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p12k3_lddda(x, t, v1, v2, v3, v4, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

3d array

`gev_p12k3_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_logfdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

`gev_p12k3_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_logfddd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

---

gev_p12k3_mu1f *GEVD-with-p1: DMGS equation 3.3 mu1 term, fixed shape parameter*

Description

GEVD-with-p1: DMGS equation 3.3 mu1 term, fixed shape parameter

Usage

```
gev_p12k3_mu1f(alpha, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either `t01` or `n01` but not both)
t02	a single value of the predictor (specify either `t02` or `n02` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

Matrix

gev_p12k3_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p12k3_mu1fa(alpha, t, v1, v2, v3, v4, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Vector

gev_p12k3_mu2f *GEVD-with-p1: DMGS equation 3.3 mu2 term, fixed shape parameter*

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term, fixed shape parameter

Usage

```
gev_p12k3_mu2f(alpha, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter

Value

3d array

---

gev_p12k3_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p12k3_mu2fa(alpha, t, v1, v2, v3, v4, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Matrix

gev_p12k3_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_pd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12k3_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_pdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

---

gev_p12_checkmle *Check MLE*

Description

Check MLE

Usage

```
gev_p12_checkmle(ml_params, minxi, maxxi)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

gev_p12_cp — *Generalized Extreme Value Distribution with Two Predictors, Predictions based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
```

```
    fdalpha = 0.01,
    minxi = -0.45,
    maxxi = 0.45,
    means = FALSE,
    waicscores = FALSE,
    extramodels = FALSE,
    pdf = FALSE,
    dmgs = TRUE,
    rust = FALSE,
    nrust = 1e+05,
    predictordata = TRUE,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  rgev_p12_cp(
    n,
    x,
    t1,
    t2,
    t01 = NA,
    t02 = NA,
    n01 = NA,
    n02 = NA,
    ics = c(0, 0, 0, 0, 0),
    d1 = 0.01,
    d2 = 0.01,
    d3 = 0.01,
    d4 = 0.01,
    d5 = 0.01,
    minxi = -0.45,
    maxxi = 0.45,
    extramodels = FALSE,
    rust = FALSE,
    mlcp = TRUE,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  dgev_p12_cp(
    x,
    t1,
    t2,
    t01 = NA,
    t02 = NA,
    n01 = NA,
```

```
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 10,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tgev_p12_cp(
  n,
  x,
  t1,
  t2,
```

```
  ics = c(0, 0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  d4 = 0.01,
  d5 = 0.01,
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
d3	if aderivs=FALSE, the delta used for numerical derivatives with respect to the third parameter
d4	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fourth parameter
d5	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fifth parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)

pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer run-time)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with two predictors has distribution function

$$F(x; a_1, b_1, a_2, b_2, \xi) = \exp\left(-t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi)\right)$$

where

$$t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi) = \begin{cases} \left[1 + \xi\left(\frac{x - \mu(a_1, b_1)}{\sigma(a_2, b_2)}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x - \mu(a_1, b_1)}{\sigma(a_2, b_2)}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a_1 + b_1 t_1$ is the location parameter, modelled as a function of parameters a_1, b_1 and predictor t_1, $\sigma = e^{a_2 + b_2 t_2}$ is the scale parameter, modelled as a function of parameters a_2, b_2 and predictor t_2, and ξ is the shape parameter.

The calibrating prior we use is given by

$$\pi(a_1, b_1, a_2, b_2, \xi) \propto 1$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
# example 1
x=fitdistcp::d151gev_p12_example_data_v1_x
tt=fitdistcp::d151gev_p12_example_data_v1_t
t1=tt[,1]
t2=tt[,2]
p=c(1:9)/10
n01=10
n02=10
q=qgev_p12_cp(x=x,t1=t1,t2=t2,n01=n01,n02=n02,t01=NA,t02=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_p12_cp)",
main="GEVD w/ p12: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

`gev_p12_f1f` *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
gev_p12_f1f(y, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter
`v4`	fourth parameter
`d4`	the delta used in the numerical derivatives with respect to the parameter
`v5`	fifth parameter
`d5`	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`gev_p12_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gev_p12_f1fa(x, t01, t02, v1, v2, v3, v4, v5)
```

Arguments

`x`	a vector of training data values
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter
`v5`	fifth parameter

Value

Vector

`gev_p12_f2f` *GEVD-with-p1: DMGS equation 1.2 f2 term*

Description

GEVD-with-p1: DMGS equation 1.2 f2 term

Usage

```
gev_p12_f2f(y, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

gev_p12_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gev_p12_f2fa(x, t01, t02, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

`gev_p12_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_fd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter
`v5`	fifth parameter

Value

Vector

`gev_p12_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_fdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

`gev_p12_ggd` *Derivative of information matrix, based on ldd*

Description

Derivative of information matrix, based on ldd

Usage

```
gev_p12_ggd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gev_p12_ldd — *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_p12_ldd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gev_p12_ldda — *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p12_ldda(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter
`v5`	fifth parameter

Value

Matrix

`gev_p12_lddd` *Third derivative tensor of the normalized log-likelihood, with fixed shape parameter*

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter

Usage

```
gev_p12_lddd(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)
```

Arguments

`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter
`v4`	fourth parameter
`d4`	the delta used in the numerical derivatives with respect to the parameter
`v5`	fifth parameter
`d5`	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

gev_p12_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p12_lddda(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

gev_p12_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p12_lmn(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, mm, nn)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

---

gev_p12_lmnp   *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p12_lmnp(x, t1, t2, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5, mm, nn, rr)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter

d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

gev_p12_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_p12_logf(params, x, t1, t2)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd

Value

Scalar value.

gev_p12_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_logfdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_logfddd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

---

gev_p12_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
gev_p12_loglik(vv, x, t1, t2)
```

Arguments

vv	parameters
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd

Value

Scalar value.

gev_p12_means	*Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p12_means(means, t01, t02, ml_params, nx)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t01	a single value of the predictor (specify either `t01` or `n01` but not both)
t02	a single value of the predictor (specify either `t02` or `n02` but not both)
ml_params	parameters
nx	length of training data

Value

Two scalars

gev_p12_mu1f	*GEVD-with-p1: DMGS equation 3.3 mu1 term*

Description

GEVD-with-p1: DMGS equation 3.3 mu1 term

Usage

```
gev_p12_mu1f(alpha, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

gev_p12_mu1fa — *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p12_mu1fa(alpha, t01, t02, v1, v2, v3, v4, v5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12_mu2f *GEVD-with-p1: DMGS equation 3.3 mu2 term*

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term

Usage

```
gev_p12_mu2f(alpha, t01, t02, v1, d1, v2, d2, v3, d3, v4, d4, v5, d5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

gev_p12_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p12_mu2fa(alpha, t01, t02, v1, v2, v3, v4, v5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

---

gev_p12_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_pd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

`gev_p12_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_pdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

`gev_p12_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p12_predictordata(predictordata, x, t1, t2, t01, t02, params)
```

Arguments

`predictordata`	logical that indicates whether to calculate and return predictordata
`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`params`	model parameters for calculating logf

Value

Two vectors

`gev_p12_setics` *Set initial conditions*

Description

Set initial conditions

Usage

```
gev_p12_setics(x, t1, t2, ics)
```

Arguments

`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`ics`	initial conditions for the maximum likelihood search

Value

Vector

gev_p12_waic *Waic*

Description

Waic

Usage

```
gev_p12_waic(
  waicscores,
  x,
  t1,
  t2,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  d3,
  v4hat,
  d4,
  v5hat,
  d5,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
v4hat	fourth parameter

d4	the delta used in the numerical derivatives with respect to the parameter
v5hat	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

gev_p1k3_cp *GEV Distribution with Known Shape with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  fdalpha = 0.01,
  kshape = 0,
  means = FALSE,
  waicscores = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rgev_p1k3_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kshape = 0,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgev_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
```

```
    d2 = 0.01,
    fd3 = 0.01,
    kshape = 0,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  pgev_p1k3_cp(
    x,
    t,
    t0 = NA,
    n0 = NA,
    y = x,
    d1 = 0.01,
    d2 = 0.01,
    fd3 = 0.01,
    kshape = 0,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  tgev_p1k3_cp(
    n,
    x,
    t,
    d1 = 0.01,
    d2 = 0.01,
    fd3 = 0.01,
    kshape = 0,
    debug = FALSE
  )
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that `length(t)=length(x)`
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
n0	an index for the predictor (specify either `t0` or `n0` but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter

d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
kshape	the known shape parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with known shape with a predictor has distribution function

$$F(x;a,b,\sigma) = \exp\left(-t(x;\mu(a,b),\sigma)\right)$$

where

$$t(x;a,b,\sigma) = \begin{cases} \left[1+\xi\left(\frac{x-\mu(a,b)}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu(a,b)}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, $\sigma > 0$ is the shape parameter and ξ is known (hence the k3 in the name).

The calibrating prior we use is given by

$$\pi(\mu, \sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson `<stephen.jewson@gmail.com>`

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),

- Weibull with linear predictor on the scale (`weibull_p2`),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d150gev_p1_example_data_v1_x #use data for 150
tt=fitdistcp::d150gev_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qgev_p1k3_cp(x=x,t=tt,n0=n0,t0=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_p1k3_cp)",
main="GEVD w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gev_p1k3_f1f — *DMGS equation 2.1, f1 term, fixed shape parameter*

Description

DMGS equation 2.1, f1 term, fixed shape parameter

DMGS equation 2.1, f1 term

Usage

```
gev_p1k3_f1f(y, t0, v1, d1, v2, d2, v3, fd3, kshape)

gev_p1k3_f1f(y, t0, v1, d1, v2, d2, v3, fd3, kshape)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter

d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Matrix

---

gev_p1k3_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gev_p1k3_f1fa(x, t, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Vector

`gev_p1k3_f2f` *GEVD-with-p1: DMGS equation 1.2 f2 term*

Description

GEVD-with-p1: DMGS equation 1.2 f2 term

DMGS equation 2.1, f2 term

Usage

```
gev_p1k3_f2f(y, t0, v1, d1, v2, d2, v3, fd3, kshape)
```



```
gev_p1k3_f2f(y, t0, v1, d1, v2, d2, v3, fd3, kshape)
```


Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kshape`	the known shape parameter

Value

3d array

`gev_p1k3_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gev_p1k3_f2fa(x, t, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Matrix

---

gev_p1k3_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_fd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

`gev_p1k3_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

`gev_p1k3_ldd` *Second derivative matrix of the normalized log-likelihood, with fixed shape parameter*

Description

Second derivative matrix of the normalized log-likelihood, with fixed shape parameter

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_p1k3_ldd(x, t, v1, d1, v2, d2, v3, fd3, kshape)

gev_p1k3_ldd(x, t, v1, d1, v2, d2, v3, fd3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Square scalar matrix

---

gev_p1k3_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p1k3_ldda(x, t, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Matrix

`gev_p1k3_lddd` *Third derivative tensor of the normalized log-likelihood, with fixed shape parameter*

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter

Third derivative tensor of the normalized log-likelihood

Usage

```
gev_p1k3_lddd(x, t, v1, d1, v2, d2, v3, fd3, kshape)

gev_p1k3_lddd(x, t, v1, d1, v2, d2, v3, fd3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Cubic scalar array

`gev_p1k3_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p1k3_lddda(x, t, v1, v2, v3, kshape)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`kshape`	the known shape parameter

Value

3d array

`gev_p1k3_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p1k3_lmn(x, t, v1, d1, v2, d2, v3, fd3, kshape, mm, nn)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kshape`	the known shape parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate

Value

Scalar value

`gev_p1k3_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p1k3_lmnp(x, t, v1, d1, v2, d2, v3, fd3, kshape, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

`gev_p1k3_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_p1k3_logf(params, x, t, kshape)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`kshape`	the known shape parameter

Value

Scalar value.

`gev_p1k3_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

gev_p1k3_logfddd	*Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

gev_p1k3_loglik	*GEV-with-known-shape-with-p1 observed log-likelihood function*

Description

GEV-with-known-shape-with-p1 observed log-likelihood function

Usage

```
gev_p1k3_loglik(vv, x, t, kshape)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kshape	the known shape parameter

Value

Scalar value.

`gev_p1k3_means` *Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p1k3_means(means, t0, ml_params, kshape, nx)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ml_params`	parameters
`kshape`	the known shape parameter
`nx`	length of training data

Value

Two scalars

`gev_p1k3_mu1f` *GEVD-with-p1: DMGS equation 3.3 mu1 term, fixed shape parameter*

Description

GEVD-with-p1: DMGS equation 3.3 mu1 term, fixed shape parameter

DMGS equation 3.3, mu1 term

Usage

```
gev_p1k3_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, kshape)
```


```
gev_p1k3_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

Matrix

---

gev_p1k3_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p1k3_mu1fa(alpha, t, v1, v2, v3, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Vector

gev_p1k3_mu2f — *GEVD-with-p1: DMGS equation 3.3 mu2 term, fixed shape parameter*

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term, fixed shape parameter

DMGS equation 3.3, mu2 term

Usage

```
gev_p1k3_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, kshape)

gev_p1k3_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter

Value

3d array

gev_p1k3_mu2fa — *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p1k3_mu2fa(alpha, t, v1, v2, v3, kshape)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`kshape`	the known shape parameter

Value

Matrix

---

gev_p1k3_pd — *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_pd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Vector

gev_p1k3_pdd — *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_pdd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

gev_p1k3_predictordata — *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p1k3_predictordata(predictordata, x, t, t0, params, kshape)
```

Arguments

`predictordata`	logical that indicates whether to calculate and return predictordata
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`params`	model parameters for calculating logf
`kshape`	the known shape parameter

Value

Two vectors

---

gev_p1k3_waic *Waic*

Description

Waic

Usage

```
gev_p1k3_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  kshape,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kshape	the known shape parameter
lddi	inverse observed information matrix

lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

gev_p1_checkmle *Check MLE*

Description

Check MLE

Usage

```
gev_p1_checkmle(ml_params, minxi, maxxi)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

---

gev_p1_cp *Generalized Extreme Value Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.

- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  d4 = 0.01,
  fdalpha = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rgev_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  ics = c(0, 0, 0, 0),
```

```
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  d4 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgev_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  ics = c(0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  d4 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pgev_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  ics = c(0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  d4 = 0.01,
  minxi = -0.45,
  maxxi = 0.45,
  extramodels = FALSE,
```

```
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tgev_p1_cp(
  n,
  x,
  t,
  ics = c(0, 0, 0, 0),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  d4 = 0.01,
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that `length(t)=length(x)`
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
n0	an index for the predictor (specify either `t0` or `n0` but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
d1	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter
d2	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the second parameter
fd3	if `aderivs=FALSE`, the fractional delta used for numerical derivatives with respect to the third parameter
d4	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the fourth parameter
fdalpha	if `pdf=TRUE`, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)

extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with a predictor has distribution function

$$F(x;a,b,\sigma,\xi) = \exp\left(-t(x;\mu(a,b),\sigma,\xi)\right)$$

where

$$t(x;\mu(a,b),\sigma,\xi) = \begin{cases} \left[1+\xi\left(\frac{x-\mu(a,b)}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu(a,b)}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, modelled as a function of parameters a, b and predictor t, and $\sigma > 0, \xi$ are the scale and shape parameters.

The calibrating prior we use is given by

$$\pi(a,b,\sigma,\xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),

- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),
- Weibull with linear predictor on the scale (`weibull_p2`),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
# example 1
x=fitdistcp::d150gev_p1_example_data_v1_x
tt=fitdistcp::d150gev_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qgev_p1_cp(x=x,t=tt,n0=n0,t0=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgev_p1_cp)",
main="GEVD w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

`gev_p1_f1f` *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
gev_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3, v4, d4)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gev_p1_f1fa — *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gev_p1_f1fa(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

gev_p1_f2f — *GEVD-with-p1: DMGS equation 1.2 f2 term*

Description

GEVD-with-p1: DMGS equation 1.2 f2 term

Usage

```
gev_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3, v4, d4)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

`v3` third parameter

`fd3` the fractional delta used in the numerical derivatives with respect to the parameter

`v4` fourth parameter

`d4` the delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

gev_p1_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gev_p1_f2fa(x, t, v1, v2, v3, v4)
```

Arguments

`x` a vector of training data values

`t` a vector or matrix of predictors

`v1` first parameter

`v2` second parameter

`v3` third parameter

`v4` fourth parameter

Value

Matrix

gev_p1_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_fd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

gev_p1_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

---

gev_p1_ggd *Derivative of information matrix, based on ldd*

Description

Derivative of information matrix, based on ldd

Usage

```
gev_p1_ggd(x, t, v1, d1, v2, d2, v3, fd3, v4, d4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gev_p1_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gev_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3, v4, d4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gev_p1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p1_ldda(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

`gev_p1_lddd` *Third derivative tensor of the normalized log-likelihood, with fixed shape parameter*

Description

Third derivative tensor of the normalized log-likelihood, with fixed shape parameter

Usage

```
gev_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3, v4, d4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

gev_p1_lddda — *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p1_lddda(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

gev_p1_lmn — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, v4, d4, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

---

gev_p1_lmnp *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gev_p1_lmnp(x, t, v1, d1, v2, d2, v3, fd3, v4, d4, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

gev_p1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

---

gev_p1_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

---

gev_p1_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

gev_p1_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
gev_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

gev_p1_means *Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p1_means(
  means,
  t0,
  ml_params,
  lddi,
  lddi_k4,
  lddd,
  lddd_k4,
  lambdad_flat,
  lambdad_rh_mle,
  lambdad_rh_flat,
  lambdad_jp,
  nx,
  dim = 4
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddi_k4	inverse observed information matrix, fixed shape parameter
lddd	third derivative of log-likelihood
lddd_k4	third derivative of log-likelihood, fixed shape parameter
lambdad_flat	derivative of the log flat prior
lambdad_rh_mle	derivative of the log CRHP-MLE prior
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
lambdad_jp	derivative of the log JP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

---

gev_p1_mu1f *GEVD-with-p1: DMGS equation 3.3 mu1 term*

Description

GEVD-with-p1: DMGS equation 3.3 mu1 term

Usage

```
gev_p1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, v4, d4)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`gev_p1_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p1_mu1fa(alpha, t, v1, v2, v3, v4)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

`gev_p1_mu2f` *GEVD-with-p1: DMGS equation 3.3 mu2 term*

Description

GEVD-with-p1: DMGS equation 3.3 mu2 term

Usage

```
gev_p1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, v4, d4)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`v4`	fourth parameter
`d4`	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gev_p1_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p1_mu2fa(alpha, t, v1, v2, v3, v4)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`gev_p1_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_pd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Vector

`gev_p1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1_pdd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`gev_p1_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

`predictordata`	logical that indicates whether to calculate and return predictordata
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`params`	model parameters for calculating logf

Value

Two vectors

gev_p1_setics — *Set initial conditions*

Description

Set initial conditions

Usage

```
gev_p1_setics(x, t, ics)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
ics	initial conditions for the maximum likelihood search

Value

Vector

gev_p1_waic — *Waic*

Description

Waic

Usage

```
gev_p1_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  v4hat,
  d4,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4hat	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

gev_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gev_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gev_pwm_params` *PWM parameter estimation*

Description

PWM parameter estimation

Usage

```
gev_pwm_params(x)
```

Arguments

x	a vector of training data values

Value

Vector

`gev_setics` *Set initial conditions*

Description

Set initial conditions

Usage

```
gev_setics(x, ics)
```

Arguments

`x`	a vector of training data values
`ics`	initial conditions for the maximum likelihood search

Value

Vector

`gev_waic` *Waic*

Description

Waic

Usage

```
gev_waic(
  waicscores,
  x,
  v1hat,
  d1,
  v2hat,
  fd2,
  v3hat,
  d3,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

gnorm_k3_cp *Generalized Normal Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgnorm_k3_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kbeta = 4,
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rgnorm_k3_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  kbeta = 4,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgnorm_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kbeta = 4,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)
```

```
pgnorm_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kbeta = 4,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgnorm_k3_cp(n, x, d1 = 0.01, fd2 = 0.01, kbeta = 4, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kbeta	the known beta parameter
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The generalized normal distribution has probability density function

$$f(x;\mu,\alpha) = \frac{\beta}{2\alpha\Gamma(1/\beta)} e^{-(|x-\mu|/\alpha)^{\beta}}$$

where x is the random variable, $\mu, \alpha > 0$ are the parameters and we consider β to be known (hence the k3 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\alpha) \propto \frac{1}{\alpha}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),

- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d32gnorm_k3_example_data_v1
p=c(1:9)/10
q=qgnorm_k3_cp(x,p,kbeta=4,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgnorm_k3_cp)",
main="gnorm: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

gnorm_k3_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
gnorm_k3_f1f(y, v1, d1, v2, fd2, kbeta)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kbeta`	the known beta parameter

Value

Matrix

`gnorm_k3_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gnorm_k3_f1fa(x, v1, v2, kbeta)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kbeta`	the known beta parameter

Value

Vector

gnorm_k3_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
gnorm_k3_f2f(y, v1, d1, v2, fd2, kbeta)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

3d array

gnorm_k3_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gnorm_k3_f2fa(x, v1, v2, kbeta)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kbeta	the known beta parameter

Value

Matrix

Matrix

`gnorm_k3_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gnorm_k3_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_fdd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`gnorm_k3_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gnorm_k3_ldd(x, v1, d1, v2, fd2, kbeta)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kbeta`	the known beta parameter

Value

Square scalar matrix

gnorm_k3_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gnorm_k3_ldda(x, v1, v2, kbeta)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kbeta	the known beta parameter

Value

Matrix

gnorm_k3_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gnorm_k3_lddd(x, v1, d1, v2, fd2, kbeta)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

Cubic scalar array

gnorm_k3_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gnorm_k3_lddda(x, v1, v2, kbeta)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kbeta	the known beta parameter

Value

3d array

gnorm_k3_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gnorm_k3_lmn(x, v1, d1, v2, fd2, kbeta, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`gnorm_k3_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
gnorm_k3_logf(params, x, kbeta)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`kbeta`	the known beta parameter

Value

Scalar value.

`gnorm_k3_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_logfdd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`gnorm_k3_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_logfddd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

`gnorm_k3_loglik` *log-likelihood function*

Description

log-likelihood function

Usage

```
gnorm_k3_loglik(vv, x, kbeta)
```

Arguments

`vv`	parameters
`x`	a vector of training data values
`kbeta`	the known beta parameter

Value

Scalar value.

gnorm_k3_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gnorm_k3_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, kbeta, aderivs)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

gnorm_k3_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gnorm_k3_mu1f(alpha, v1, d1, v2, fd2, kbeta)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

Matrix

---

gnorm_k3_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gnorm_k3_mu2f(alpha, v1, d1, v2, fd2, kbeta)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

3d array

gnorm_k3_p1f — *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
gnorm_k3_p1f(y, v1, d1, v2, fd2, kbeta)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

Matrix

gnorm_k3_p2f — *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
gnorm_k3_p2f(y, v1, d1, v2, fd2, kbeta)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

3d array

`gnorm_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gnorm_lmnp(x, v1, d1, v2, fd2, kbeta, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gnorm_waic *Waic for RUST*

Description

Waic for RUST

Usage

```
gnorm_waic(
  waicscores,
  x,
  v1hat,
  d1,
  v2hat,
  fd2,
  kbeta,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gpd_k13_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
gpd_k13_f1f(y, v1, fd1, v2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k13_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gpd_k13_f1fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

gpd_k13_f2f — *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
gpd_k13_f2f(y, v1, fd1, v2, kloc)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

3d array

gpd_k13_f2fa — *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gpd_k13_f2fa(x, v1, v2, kloc)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

Matrix

`gpd_k13_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gpd_k13_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k13_l11 — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gpd_k13_l11(x, v1, fd1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Scalar value

gpd_k13_l111 — *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
gpd_k13_l111(x, v1, fd1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Scalar value

`gpd_k13_ldd` *Second derivative matrix of the normalized log-likelihood, with fixed shape*

Description

Second derivative matrix of the normalized log-likelihood, with fixed shape

Usage

```
gpd_k13_ldd(x, v1, fd1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Square scalar matrix

`gpd_k13_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gpd_k13_ldda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

`gpd_k13_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gpd_k13_lddd(x, v1, fd1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Cubic scalar array

`gpd_k13_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gpd_k13_lddda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

3d array

`gpd_k13_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gpd_k13_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`gpd_k13_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gpd_k13_mu1f(alpha, v1, fd1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

`gpd_k13_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gpd_k13_mu1fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

`gpd_k13_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gpd_k13_mu2f(alpha, v1, fd1, v2, kloc)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

3d array

`gpd_k13_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gpd_k13_mu2fa(alpha, v1, v2, kloc)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

Matrix

---

gpd_k13_p1f *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
gpd_k13_p1f(y, v1, fd1, v2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

---

gpd_k13_p2f *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
gpd_k13_p2f(y, v1, fd1, v2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
kloc	the known location parameter

Value

3d array

---

gpd_k13_pd   *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

---

gpd_k13_pdd   *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gpd_k1_checkmle` *Check MLE*

Description

Check MLE

Usage

```
gpd_k1_checkmle(ml_params, kloc, minxi, maxxi)
```

Arguments

ml_params	parameters
kloc	the known location parameter
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

`gpd_k1_cp` *Generalized Pareto Distribution with Known Location Parameter, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`

- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgpd_k1_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  fdalpha = 0.01,
  customprior = 0,
  minxi = -0.45,
  maxxi = 2,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rgpd_k1_cp(
  n,
  x,
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  minxi = -0.45,
  maxxi = 2,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
```

```
)

dgpd_k1_cp(
  x,
  y = x,
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  customprior = 0,
  minxi = -0.45,
  maxxi = 2,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pgpd_k1_cp(
  x,
  y = x,
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  customprior = 0,
  minxi = -0.45,
  maxxi = 2,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgpd_k1_cp(
  n,
  x,
  kloc = 0,
  ics = c(0, 0),
  fd1 = 0.01,
  d2 = 0.01,
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kloc	the known location parameter
ics	initial conditions for the maximum likelihood search
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
customprior	a custom value for the slope of the log prior at the maxlik estimate
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Details

The GP distribution has exceedcance distribution function

$$S(x;\mu,\sigma,\xi) = \begin{cases} \left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable and $\mu, \sigma > 0, \xi$ are the parameters.

The calibrating prior we use is given by

$$\pi(\mu, \sigma, \xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.

- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d120gpd_k1_example_data_v1
p=c(1:9)/10
q=qgpd_k1_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgpd_k1_cp)",
main="GPD: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gpd_k1_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
gpd_k1_f1f(y, v1, fd1, v2, d2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

gpd_k1_f1fa — *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gpd_k1_f1fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

gpd_k1_f2f — *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
gpd_k1_f2f(y, v1, fd1, v2, d2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

gpd_k1_f2fa — *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gpd_k1_f2fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k1_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gpd_k1_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k1_ggd_mev — *Derivative of expected information matrix, based on MEV routine gpd.infomat*

Description

Derivative of expected information matrix, based on MEV routine gpd.infomat

Usage

```
gpd_k1_ggd_mev(v1, fd1, v2, d2, kloc)
```

Arguments

v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Cubic scalar array

---

gpd_k1_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gpd_k1_ldd(x, v1, fd1, v2, d2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Square scalar matrix

---

gpd_k1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gpd_k1_ldda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

`gpd_k1_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gpd_k1_lddd(x, v1, fd1, v2, d2, kloc)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

Cubic scalar array

`gpd_k1_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gpd_k1_lddda(x, v1, v2, kloc)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kloc`	the known location parameter

Value

3d array

`gpd_k1_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gpd_k1_lmn(x, v1, fd1, v2, d2, kloc, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`gpd_k1_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gpd_k1_lmnp(x, v1, fd1, v2, d2, kloc, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

gpd_k1_logf — *Logf for RUST*

Description

Logf for RUST

Usage

```
gpd_k1_logf(params, x, kloc)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kloc	the known location parameter

Value

Scalar value.

`gpd_k1_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_logfdd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`gpd_k1_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_logfddd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

gpd_k1_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
gpd_k1_loglik(vv, x, kloc)
```

Arguments

vv	parameters
x	a vector of training data values
kloc	the known location parameter

Value

Scalar value.

gpd_k1_means *Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gpd_k1_means(
  means,
  ml_params,
  lddi,
  lddi_k2,
  lddd,
  lddd_k2,
  lambdad_flat,
  lambdad_rh_mle,
  lambdad_rh_flat,
  lambdad_jp,
  nx,
  dim = 2,
  kloc = 0
)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`ml_params`	parameters
`lddi`	inverse observed information matrix
`lddi_k2`	inverse observed information matrix, fixed shape parameter
`lddd`	third derivative of log-likelihood
`lddd_k2`	third derivative of log-likelihood, fixed shape parameter
`lambdad_flat`	derivative of the log flat prior
`lambdad_rh_mle`	derivative of the log CRHP-MLE prior
`lambdad_rh_flat`	derivative of the log CRHP-FLAT prior
`lambdad_jp`	derivative of the log JP prior
`nx`	length of training data
`dim`	number of parameters
`kloc`	the known location parameter

Value

Two scalars

`gpd_k1_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gpd_k1_mu1f(alpha, v1, fd1, v2, d2, kloc)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`kloc`	the known location parameter

Value

Matrix

gpd_k1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gpd_k1_mu1fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

gpd_k1_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gpd_k1_mu2f(alpha, v1, fd1, v2, d2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

`gpd_k1_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gpd_k1_mu2fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

`gpd_k1_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
gpd_k1_p1f(y, v1, fd1, v2, d2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

Matrix

gpd_k1_p2f *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
gpd_k1_p2f(y, v1, fd1, v2, d2, kloc)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter

Value

3d array

gpd_k1_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gpd_k1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gpd_k1_setics` *Set initial conditions*

Description

Set initial conditions

Usage

```
gpd_k1_setics(x, ics)
```

Arguments

x	a vector of training data values
ics	initial conditions for the maximum likelihood search

Value

Vector

gpd_k1_waic *Waic*

Description

Waic

Usage

```
gpd_k1_waic(
  waicscores,
  x,
  v1hat,
  fd1,
  v2hat,
  d2,
  kloc,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kloc	the known location parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gumbel_cp *Gumbel Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgumbel_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rgumbel_cp(
```

```
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgumbel_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pgumbel_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgumbel_cp(n, x, d1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)

dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Gumbel distribution has distribution function

$$F(x;\mu,\sigma) = \exp\left(-\exp\left(-\frac{x-\mu}{\sigma}\right)\right)$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible

- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson `<stephen.jewson@gmail.com>`

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d50gumbel_example_data_v1
p=c(1:9)/10
q=qgumbel_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgumbel_cp)",
main="Gumbel: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

gumbel_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
gumbel_f1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gumbel_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gumbel_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`gumbel_f2f` *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
gumbel_f2f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gumbel_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gumbel_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gumbel_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_ldd — *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gumbel_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gumbel_ldda — *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gumbel_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_lddd — *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gumbel_lddd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

gumbel_lddda — *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gumbel_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

gumbel_lmn	*One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gumbel_lmn(x, v1, d1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gumbel_lmnp	*One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
gumbel_lmnp(x, v1, d1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

gumbel_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gumbel_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

`gumbel_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`gumbel_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

gumbel_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
gumbel_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

gumbel_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gumbel_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

gumbel_means *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
gumbel_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

gumbel_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gumbel_mu1f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`gumbel_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gumbel_mu1fa(alpha, v1, v2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`v2`	second parameter

Value

Vector

`gumbel_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gumbel_mu2f(alpha, v1, d1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

gumbel_mu2fa — *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gumbel_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_p1f — *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
gumbel_p1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gumbel_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
gumbel_p1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gumbel_p1_cp *Gumbel Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgumbel_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rgumbel_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgumbel_p1_cp(
  x,
  t,
```

```
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pgumbel_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tgumbel_p1_cp(n, x, t, d1 = 0.01, d2 = 0.01, fd3 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	the fractional delta used in the numerical derivatives with respect to the location parameter
d2	the fractional delta used in the numerical derivatives with respect to the slope parameter
fd3	the fractional delta used in the numerical derivatives with respect to the scale parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Gumbel distribution with a predictor has distribution function

$$F(x; a, b, \sigma) = \exp\left(-\exp\left(-\frac{x - \mu(a, b)}{\sigma}\right)\right)$$

where x is the random variable, $\mu = a + bt$ is the shape parameter as a function of parameters a, b and predictor t, and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d70gumbel_p1_example_data_v1_x
tt=fitdistcp::d70gumbel_p1_example_data_v1_t
p=c(1:9)/10
n0=10
```

```
q=qgumbel_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qgumbel_p1_cp)",
main="Gumbel w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`gumbel_p1_f1f` *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
gumbel_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gumbel_p1_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gumbel_p1_f1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gumbel_p1_f2f *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
gumbel_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gumbel_p1_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gumbel_p1_f2fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gumbel_p1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

---

gumbel_p1_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

gumbel_p1_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gumbel_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

---

gumbel_p1_ldda — *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gumbel_p1_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gumbel_p1_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gumbel_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

`gumbel_p1_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gumbel_p1_lddda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`gumbel_p1_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gumbel_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gumbel_p1_lmnp *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gumbel_p1_lmnp(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gumbel_p1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gumbel_p1_logf(params, x, t)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Scalar value.

`gumbel_p1_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

gumbel_p1_logfddd	*Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gumbel_p1_loglik	*observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
gumbel_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

gumbel_p1_logscores — *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gumbel_p1_logscores(logscores, x, t, d1, d2, fd3, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

gumbel_p1_means — *Gumbel distribution: RHP mean*

Description

Gumbel distribution: RHP mean

Usage

```
gumbel_p1_means(means, t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

---

gumbel_p1_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gumbel_p1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gumbel_p1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gumbel_p1_mu1fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gumbel_p1_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gumbel_p1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gumbel_p1_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gumbel_p1_mu2fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gumbel_p1_p1f` *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
gumbel_p1_p1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`gumbel_p1_p1fa` *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
gumbel_p1_p1fa(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

gumbel_p1_p2f *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
gumbel_p1_p2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

gumbel_p1_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
gumbel_p1_p2fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`gumbel_p1_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gumbel_p1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_pdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

gumbel_p1_predictordata
Predicted Parameter and Generalized Residuals

Description

Predicted Parameter and Generalized Residuals

Usage

```
gumbel_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

gumbel_p1_waic *Waic*

Description

Waic

Usage

```
gumbel_p1_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`gumbel_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
gumbel_p2f(y, v1, d1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gumbel_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
gumbel_p2fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

`gumbel_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_pd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

`gumbel_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_pdd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

gumbel_waic	*Waic*

Description

Waic

Usage

```
gumbel_waic(waicscores, x, v1hat, d1, v2hat, fd2, lddi, lddd, lambdad, aderivs)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`v1hat`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad`	derivative of the log prior
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

halfnorm_cp	*Half-Normal Distribution Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qhalfnorm_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rhalfnorm_cp(
  n,
  x,
  fd1 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dhalfnorm_cp(
  x,
  y = x,
  fd1 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
```

```
  aderivs = TRUE
)

phalfnorm_cp(
  x,
  y = x,
  fd1 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

thalfnorm_cp(n, x, fd1 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.

- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The half-normal distribution has probability density function

$$f(x;\theta) = \frac{2\theta}{\pi} e^{-\theta^2 x^2/\pi}$$

where $x \geq 0$ is the random variable and $\theta > 0$ is the parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\theta) \propto \frac{1}{\theta}$$

as given in Jewson et al. (2025). Some other authors may parametrize the half-normal differently.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson `<stephen.jewson@gmail.com>`

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),

- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d20halfnorm_example_data_v1
p=c(1:9)/10
q=qhalfnorm_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles)
xmax=max(q$ml_quantiles,q$cp_quantiles)
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qhalfnorm_cp)",
main="Halfnorm: quantile estimates")
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

halfnorm_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
halfnorm_f1f(y, v1, fd1)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`halfnorm_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
halfnorm_f1fa(x, v1)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter

Value

Vector

`halfnorm_f2f` *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
halfnorm_f2f(y, v1, fd1)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

halfnorm_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
halfnorm_f2fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

halfnorm_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_fd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

halfnorm_fdd	*Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_fdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

halfnorm_gg	*Expected information matrix*

Description

Expected information matrix

Usage

```
halfnorm_gg(v1, fd1)
```

Arguments

v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

halfnorm_gg11 *Second derivative of the expected log-likelihood*

Description

Second derivative of the expected log-likelihood

Usage

```
halfnorm_gg11(alpha, v1, fd1)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

halfnorm_l111 *Third derivative of the normalized log-likelihood*

Description

Third derivative of the normalized log-likelihood

Usage

```
halfnorm_l111(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

halfnorm_ldd *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
halfnorm_ldd(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

halfnorm_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
halfnorm_ldda(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

`halfnorm_lddd` *Third derivative tensor of the log-likelihood*

Description

Third derivative tensor of the log-likelihood

Usage

```
halfnorm_lddd(x, v1, fd1)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

`halfnorm_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
halfnorm_lddda(x, v1)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter

Value

3d array

halfnorm_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
halfnorm_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

halfnorm_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_logfdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

`halfnorm_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_logfddd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

3d array

`halfnorm_loglik` *Log-likelihood function*

Description

Log-likelihood function

Usage

```
halfnorm_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

halfnorm_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
halfnorm_logscores(logscores, x, fd1 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

halfnorm_means *MLE and RHP predictive means RHP mean based on the expectation of DMGS equation 2.1*

Description

MLE and RHP predictive means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
halfnorm_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 1)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

`halfnorm_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
halfnorm_mu1f(alpha, v1, fd1)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`halfnorm_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
halfnorm_mu2f(alpha, v1, fd1)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

halfnorm_p1f *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
halfnorm_p1f(y, v1, fd1)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

halfnorm_p2f *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
halfnorm_p2f(y, v1, fd1)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

halfnorm_waic *Waic*

Description

Waic

Usage

```
halfnorm_waic(waicscores, x, v1hat, fd1, lddi, lddd, lambdad, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

invgamma_cp *Inverse Gamma Distribution, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.

- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qinvgamma_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)

rinvgamma_cp(
  n,
  x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dinvgamma_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
```

```
  debug = FALSE,
  aderivs = TRUE
)

pinvgamma_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tinvgamma_cp(n, x, fd1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
prior	logical indicating which prior to use
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Inverse Gamma distribution has probability density function

$$f(x;\alpha,\sigma) = \frac{1}{x\Gamma(\alpha)}\left(\frac{\sigma}{x}\right)^{\alpha} e^{-\sigma/x}$$

where $x \geq 0$ is the random variable and $\alpha > 0, \sigma > 0$ are the parameters.

The calibrating prior we use is

$$\pi(\alpha,\sigma) \propto \frac{1}{\alpha\sigma}$$

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),

- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d101invgamma_example_data_v1
p=c(1:9)/10
q=qinvgamma_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qinvgamma_cp)",
main="Invgamma: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

invgamma_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
invgamma_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

invgamma_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
invgamma_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgamma_f2f	*DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
invgamma_f2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

invgamma_f2fa	*The second derivative of the density*

Description

The second derivative of the density

Usage

```
invgamma_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_fd

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgamma_fdd

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
invgamma_ldd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

invgamma_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
invgamma_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
invgamma_lddd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

invgamma_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
invgamma_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

invgamma_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgamma_lmn(x, v1, fd1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

invgamma_lmnp *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgamma_lmnp(x, v1, fd1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

invgamma_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
invgamma_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

invgamma_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

invgamma_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
invgamma_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

invgamma_logscores *Log scores for MLE and cp predictions calculated using leave-one-out*

Description

Log scores for MLE and cp predictions calculated using leave-one-out

Usage

```
invgamma_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

invgamma_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
invgamma_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

invgamma_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
invgamma_mu2f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`invgamma_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
invgamma_p1f(y, v1, fd1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`invgamma_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
invgamma_p2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`invgamma_waic` *Waic*

Description

Waic

Usage

```
invgamma_waic(
  waicscores,
  x,
  v1hat,
  fd1,
  v2hat,
  fd2,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter

fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

invgauss_cp *Inverse Gauss Distribution, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qinvgauss_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)

rinvgauss_cp(
  n,
  x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  prior = "type 1",
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dinvgauss_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)

pinvgauss_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
```

```
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)

tinvgauss_cp(n, x, fd1 = 0.01, fd2 = 0.01, prior = "type 1", debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
prior	logical indicating which prior to use
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.

- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Inverse Gaussian distribution has probability density function

$$f(x;\mu,\phi) = \left(\frac{1}{2\pi\phi x^3}\right)^{1/2} \exp\left(-\frac{(x-\mu)^2}{2\mu^2\phi x}\right)$$

where $x \geq 0$ is the random variable and $\mu > 0, \phi > 0$ are the parameters.

The calibrating prior we use by default is

$$\pi(\alpha,\sigma) \propto \frac{1}{\phi}$$

The prior

$$\pi(\alpha,\sigma) \propto \frac{1}{\mu\phi}$$

is also available as an option with prior="type 2".

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),

- Weibull with linear predictor on the scale (`weibull_p2`),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
debug=FALSE
# example 1 can go wrong for small sample sizes, so I've increased to 50
#
# example 1
if(debug)cat("example 1\n")
x=fitdistcp::d102invgauss_example_data_v1
if(debug)cat("x=",x,"\n")
p=c(1:9)/10
q=qinvgauss_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qinvgauss_cp)",
main="Invgauss: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

invgauss_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
invgauss_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`invgauss_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
invgauss_f1fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

`invgauss_f2f` *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
invgauss_f2f(y, v1, fd1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`invgauss_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
invgauss_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`invgauss_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgauss_fdd	*Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgauss_ldd	*Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
invgauss_ldd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

invgauss_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
invgauss_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgauss_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
invgauss_lddd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

`invgauss_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
invgauss_lddda(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

3d array

`invgauss_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgauss_lmn(x, v1, fd1, v2, fd2, mm, nn)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate

Value

Scalar value

`invgauss_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgauss_lmnp(x, v1, fd1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

`invgauss_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
invgauss_logf(params, x, prior)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`prior`	logical indicating which prior to use

Value

Scalar value.

`invgauss_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_logfdd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

`invgauss_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_logfddd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

3d array

`invgauss_loglik` *log-likelihood function*

Description

log-likelihood function

Usage

```
invgauss_loglik(vv, x)
```

Arguments

`vv`	parameters
`x`	a vector of training data values

Value

Scalar value.

`invgauss_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
invgauss_logscores(logscores, x, prior, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`prior`	logical indicating which prior to use
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`invgauss_means` *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
invgauss_means(means, ml_params, lddi, lddd, lambdad_cp, nx, dim = 2)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`ml_params`	parameters
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad_cp`	derivative of the log prior
`nx`	length of training data
`dim`	number of parameters

Value

Two scalars

invgauss_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
invgauss_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

invgauss_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
invgauss_mu2f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`invgauss_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
invgauss_p1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`invgauss_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
invgauss_p2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

invgauss_waic *Waic*

Description

Waic

Usage

```
invgauss_waic(
  waicscores,
  x,
  v1hat,
  fd1,
  v2hat,
  fd2,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter

`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad`	derivative of the log prior
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`jpf2p` *Jeffreys' Prior with two parameters*

Description

Jeffreys' Prior with two parameters

Usage

```
jpf2p(ggd, detg, ggi)
```

Arguments

`ggd`	gradient of the expected information matrix
`detg`	determinant of the expected information matrix
`ggi`	inverse of the expected information matrix

Value

Vector of 2 values

`jpf3p` *Jeffreys' Prior with three parameters*

Description

Jeffreys' Prior with three parameters

Usage

```
jpf3p(ggd, detg, ggi)
```

Arguments

ggd	gradient of the expected information matrix
detg	determinant of the expected information matrix
ggi	inverse of the expected information matrix

Value

Vector of 3 values

---

jpf4p — *Jeffreys' Prior with four parameters*

Description

Jeffreys' Prior with four parameters

Usage

```
jpf4p(ggd, detg, ggi)
```

Arguments

ggd	gradient of the expected information matrix
detg	determinant of the expected information matrix
ggi	inverse of the expected information matrix

Value

Vector of 4 values

---

lnorm_cp — *Log-normal Distribution Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlnorm_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rlnorm_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE, aderivs = TRUE)

dlnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

plnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

tlnorm_cp(n, x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter

means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.

- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The log normal distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma x} e^{-(\log(x)-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d35lnorm_example_data_v1
p=c(1:9)/10
q=qlnorm_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlnorm_cp)",
main="Log-normal: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

lnorm_dmgs_cp *Log-normal Distribution Predictions Based on a Calibrating Prior, using DMGS (for testing only)*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlnorm_dmgs_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rlnorm_dmgs_cp(
  n,
```

```
  x,
  d1 = 0.01,
  fd2 = 0.01,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dlnorm_dmgs_cp(x, y = x, d1 = 0.01, fd2 = 0.01, debug = FALSE, aderivs = TRUE)

plnorm_dmgs_cp(x, y, d1 = 0.01, fd2 = 0.01, debug = FALSE, aderivs = TRUE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.

- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The log normal distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma x} e^{-(\log(x)-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d35lnorm_example_data_v1
p=c(1:9)/10
q=qlnorm_dmgs_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlnorm_dmgs_cp)",
main="Log-normal_DMGS: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

---

lnorm_dmgs_gg11 *One component of the second derivative of the expected log-likelihood*

Description

One component of the second derivative of the expected log-likelihood

Usage

```
lnorm_dmgs_gg11(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

`lnorm_dmgs_gg12` *One component of the second derivative of the expected log-likelihood*

Description

One component of the second derivative of the expected log-likelihood

Usage

```
lnorm_dmgs_gg12(alpha, v1, d1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

`lnorm_dmgs_gg22` *One component of the second derivative of the expected log-likelihood*

Description

One component of the second derivative of the expected log-likelihood

Usage

```
lnorm_dmgs_gg22(alpha, v1, d1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

`lnorm_dmgs_loglik` *log-likelihood function*

Description

log-likelihood function

Usage

```
lnorm_dmgs_loglik(vv, x)
```

Arguments

`vv`	parameters
`x`	a vector of training data values

Value

Scalar value.

`lnorm_dmgs_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lnorm_dmgs_logscores(logscores, x, d1 = 0.01, fd2 = 0.01)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`d1`	the delta used in the numerical derivatives with respect to the parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Two scalars

`lnorm_dmgs_means` *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
lnorm_dmgs_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

`lnorm_dmgs_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
lnorm_dmgs_mu1f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`lnorm_dmgs_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
lnorm_dmgs_mu2f(alpha, v1, d1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`lnorm_dmgs_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
lnorm_dmgs_p1f(y, v1, d1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`lnorm_dmgs_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
lnorm_dmgs_p2f(y, v1, d1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`lnorm_dmgs_waic` *Waic*

Description

Waic

Usage

```
lnorm_dmgs_waic(
  waicscores,
  x,
  v1hat,
  d1,
  v2hat,
  fd2,
  lddi,
```

```
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

lnorm_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
lnorm_f1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`lnorm_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
lnorm_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`lnorm_f2f` *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
lnorm_f2f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

lnorm_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
lnorm_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`lnorm_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`lnorm_ldd` *Second derivative matrix of the lnormalized log-likelihood*

Description

Second derivative matrix of the lnormalized log-likelihood

Usage

```
lnorm_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

lnorm_ldda	*The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
lnorm_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_lddd	*Third derivative tensor of the lnormalized log-likelihood*

Description

Third derivative tensor of the lnormalized log-likelihood

Usage

```
lnorm_lddd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

lnorm_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
lnorm_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

lnorm_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lnorm_lmn(x, v1, d1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

lnorm_lmnp — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lnorm_lmnp(x, v1, d1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

lnorm_logf — *Logf for RUST*

Description

Logf for RUST

Usage

```
lnorm_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

lnorm_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

lnorm_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lnorm_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

lnorm_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
lnorm_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

lnorm_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
lnorm_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
lnorm_p1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

lnorm_p1_cp *Log-normal Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  centering = TRUE,
```

```
  debug = FALSE,
  aderivs = TRUE
)

rlnorm_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dlnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

plnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tlnorm_p1_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that `length(t)=length(x)`

t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.

- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The log normal distribution with a predictor has probability density function

$$f(x;a,b,\sigma) = \frac{1}{\sqrt{2\pi}x\sigma} e^{-(\log(x)-\mu(a,b))^2/(2\sigma^2)}$$

where x is the random variable, $\mu = a + bt$ is the location parameter of the log of the random variable, modelled as a function of parameters a, b and predictor t, and $\sigma > 0$ is the scale parameter of the log of the random variable.

The calibrating prior is given by the right Haar prior, which is

$$\pi(a,b,\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d61lnorm_p1_example_data_v1_x
tt=fitdistcp::d61lnorm_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qlnorm_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlnorm_p1_cp)",
main="Log-Normal w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

lnorm_p1_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
lnorm_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`lnorm_p1_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
lnorm_p1_f1fa(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`lnorm_p1_f2f` *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
lnorm_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`lnorm_p1_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
lnorm_p1_f2fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`lnorm_p1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`lnorm_p1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`lnorm_p1_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
lnorm_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

---

lnorm_p1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
lnorm_p1_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_lddd — *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
lnorm_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

lnorm_p1_lddda — *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
lnorm_p1_lddda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`lnorm_p1_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lnorm_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate

Value

Scalar value

lnorm_p1_lmnp — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lnorm_p1_lmnp(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

lnorm_p1_logf — *Logf for RUST*

Description

Logf for RUST

Usage

```
lnorm_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

---

lnorm_p1_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

lnorm_p1_loglik *Log-normal-with-p1 observed log-likelihood function*

Description

Log-normal-with-p1 observed log-likelihood function

Usage

```
lnorm_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

lnorm_p1_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lnorm_p1_logscores(logscores, x, t)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors

Value

Two scalars

lnorm_p1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
lnorm_p1_mu1fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

lnorm_p1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
lnorm_p1_mu2fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
lnorm_p1_p1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

lnorm_p1_p2fa — *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
lnorm_p1_p2fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_pd — *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`lnorm_p1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_pdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`lnorm_p1_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
lnorm_p1_predictordata(x, t, t0, params)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`params`	model parameters for calculating logf

Value

Two vectors

`lnorm_p1_waic` *Waic*

Description

Waic

Usage

```
lnorm_p1_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  aderivs = TRUE
)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1hat`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3hat`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`lnorm_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
lnorm_p2fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

`lnorm_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_pd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

lnorm_pdd — *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_pdd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

lnorm_waic — *Waic for RUST*

Description

Waic for RUST

Usage

```
lnorm_waic(waicscores, x, v1hat, d1, v2hat, fd2, aderivs)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`v1hat`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`logis_cp` *Logistic Distribution Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlogis_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
```

```
)

rlogis_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dlogis_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

plogis_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tlogis_cp(n, x, d1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.

- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The logistic distribution has distribution function

$$f(x;\mu,\sigma) = \frac{1}{1+e^{-(x-\mu)/\sigma}}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson `<stephen.jewson@gmail.com>`

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d40logis_example_data_v1
p=c(1:9)/10
q=qlogis_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlogis_cp)",
main="Logistic: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

logis_f1f — *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
logis_f1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_f1fa — *The first derivative of the density*

Description

The first derivative of the density

Usage

```
logis_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

logis_f2f — *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
logis_f2f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

logis_f2fa — *The second derivative of the density*

Description

The second derivative of the density

Usage

```
logis_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

logis_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
logis_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

logis_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
logis_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
logis_lddd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

logis_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
logis_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

logis_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
logis_lmn(x, v1, d1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

logis_lmnp *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
logis_lmnp(x, v1, d1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

logis_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
logis_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

`logis_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_logfdd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

`logis_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_logfddd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

3d array

logis_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
logis_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

logis_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
logis_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

logis_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
logis_mu1f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
logis_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

logis_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
logis_mu2f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

logis_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
logis_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

logis_p1f — *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
logis_p1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_p1fa — *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
logis_p1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

logis_p1_cp *Logistic Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlogis_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
```

```
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rlogis_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dlogis_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

plogis_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
```

```
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tlogis_p1_cp(n, x, t, d1 = 0.01, d2 = 0.01, fd3 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The logistic distribution with a predictor has distribution function

$$f(x; a, b, \sigma) = \frac{1}{1 + e^{-(x-\mu(a,b))/\sigma}}$$

where x is the random variable, $\mu = a+bt$ is the location paramter, and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),

- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d62logis_p1_example_data_v1_x
tt=fitdistcp::d62logis_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qlogis_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlogis_p1_cp)",
main="Logistic w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

logis_p1_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
logis_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

logis_p1_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
logis_p1_f1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`logis_p1_f2f` *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
logis_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`logis_p1_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
logis_p1_f2fa(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`logis_p1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`logis_p1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

logis_p1_ldd — *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
logis_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

logis_p1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
logis_p1_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

logis_p1_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
logis_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

`logis_p1_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
logis_p1_lddda(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

`logis_p1_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
logis_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`logis_p1_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
logis_p1_lmnp(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

`logis_p1_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
logis_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

`logis_p1_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`logis_p1_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`logis_p1_loglik` *Logistic-with-p1 observed log-likelihood function*

Description

Logistic-with-p1 observed log-likelihood function

Usage

```
logis_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

`logis_p1_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
logis_p1_logscores(logscores, x, t, d1, d2, fd3, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`logis_p1_means` *Logistic distribution: RHP mean*

Description

Logistic distribution: RHP mean

Usage

```
logis_p1_means(t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

---

logis_p1_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
logis_p1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

logis_p1_mu1fa — *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
logis_p1_mu1fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

logis_p1_mu2f — *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
logis_p1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

logis_p1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
logis_p1_mu2fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

logis_p1_p1f *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
logis_p1_p1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`logis_p1_p1fa` *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
logis_p1_p1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

logis_p1_p2f *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
logis_p1_p2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

logis_p1_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
logis_p1_p2fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`logis_p1_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`logis_p1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_pdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`logis_p1_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
logis_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

`predictordata`	logical that indicates whether to calculate and return predictordata
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`params`	model parameters for calculating logf

Value

Two vectors

logis_p1_waic *Waic*

Description

Waic

Usage

```
logis_p1_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  lddi,
  lddd,
  lambdad,
  aderivs = TRUE
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`logis_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
logis_p2f(y, v1, d1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`logis_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
logis_p2fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

logis_pd — *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_pd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

logis_pdd — *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_pdd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

`logis_waic` *Waic*

Description

Waic

Usage

```
logis_waic(waicscores, x, v1hat, d1, v2hat, fd2, lddi, lddd, lambdad, aderivs)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`v1hat`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad`	derivative of the log prior
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`lst_k3_cp` *t Distribution Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlst_k3_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kdf = 5,
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rlst_k3_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  kdf = 5,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dlst_k3_cp(
  x,
  y = x,
```

```
  d1 = 0.01,
  fd2 = 0.01,
  kdf = 5,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

plst_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kdf = 5,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tlst_k3_cp(n, x, d1 = 0.01, fd2 = 0.01, kdf = 5, debug = FALSE)
```

Arguments

`x`	a vector of training data values
`p`	a vector of probabilities at which to generate predictive quantiles
`kdf`	the known degrees of freedom parameter
`d1`	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter
`fd2`	if `aderivs=FALSE`, the fractional delta used for numerical derivatives with respect to the second parameter
`means`	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
`waicscores`	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
`logscores`	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
`dmgs`	logical that indicates whether DMGS calculations should be run or not (longer run time)
`rust`	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
`nrust`	the number of posterior samples used in the RUST calculations
`debug`	logical for turning on debug messages
`aderivs`	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.

n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The t distribution (also known as the location-scale t distribution, hence the name `lst`), has probability density function

$$f(x;\mu,\sigma) = \frac{\Gamma((\nu+1)/2)}{\sqrt{\pi\nu}\sigma\Gamma(\nu/2)}\left(1+\frac{(x-\mu)^2}{\sigma^2\nu}\right)^{(\nu+1)/2}$$

where x is the random variable, $\mu, \sigma > 0$ are the parameters, and we consider the degrees of freedom ν to be known (hence the `k3` in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),

- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d41lst_k3_example_data_v1
p=c(1:9)/10
q=qlst_k3_cp(x,p,kdf=5,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlst_k3_cp)",
main="t: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

lst_k3_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
lst_k3_f1f(y, v1, d1, v2, fd2, kdf)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter

Value

Matrix

`lst_k3_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
lst_k3_f1fa(x, v1, v2, kdf)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kdf`	the known degrees of freedom parameter

Value

Vector

`lst_k3_f2f` *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
lst_k3_f2f(y, v1, d1, v2, fd2, kdf)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter

Value

3d array

`lst_k3_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
lst_k3_f2fa(x, v1, v2, kdf)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kdf`	the known degrees of freedom parameter

Value

Matrix

`lst_k3_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_k3_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`lst_k3_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_k3_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lst_k3_ldd — *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
lst_k3_ldd(x, v1, d1, v2, fd2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Square scalar matrix

lst_k3_ldda — *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
lst_k3_ldda(x, v1, v2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_k3_lddd — *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
lst_k3_lddd(x, v1, d1, v2, fd2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Cubic scalar array

lst_k3_lddda — *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
lst_k3_lddda(x, v1, v2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kdf	the known degrees of freedom parameter

Value

3d array

`lst_k3_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lst_k3_lmn(x, v1, d1, v2, fd2, kdf, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`lst_k3_lmnp` *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
lst_k3_lmnp(x, v1, d1, v2, fd2, kdf, mm, nn, rr)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate
`rr`	an index for which derivative to calculate

Value

Scalar value

`lst_k3_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
lst_k3_logf(params, x, kdf)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`kdf`	the known degrees of freedom parameter

Value

Scalar value.

lst_k3_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_k3_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lst_k3_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_k3_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`lst_k3_loglik` *log-likelihood function*

Description

log-likelihood function

Usage

```
lst_k3_loglik(vv, x, kdf)
```

Arguments

`vv`	parameters
`x`	a vector of training data values
`kdf`	the known degrees of freedom parameter

Value

Scalar value.

`lst_k3_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lst_k3_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, kdf, aderivs = TRUE)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`d1`	the delta used in the numerical derivatives with respect to the parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`lst_k3_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
lst_k3_mu1f(alpha, v1, d1, v2, fd2, kdf)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter

Value

Matrix

`lst_k3_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
lst_k3_mu2f(alpha, v1, d1, v2, fd2, kdf)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter

Value

3d array

`lst_k3_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
lst_k3_p1f(y, v1, d1, v2, fd2, kdf)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter

Value

Matrix

`lst_k3_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
lst_k3_p2f(y, v1, d1, v2, fd2, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

---

lst_k3_waic *Waic*

Description

Waic

Usage

```
lst_k3_waic(
  waicscores,
  x,
  v1hat,
  d1,
  v2hat,
  fd2,
  kdf,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter

fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

lst_p1k3_cp *t Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlst_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kdf = 10,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rlst_p1k3_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kdf = 10,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dlst_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
```

```
    fd3 = 0.01,
    kdf = 10,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  plst_p1k3_cp(
    x,
    t,
    t0 = NA,
    n0 = NA,
    y = x,
    d1 = 0.01,
    d2 = 0.01,
    fd3 = 0.01,
    kdf = 10,
    rust = FALSE,
    nrust = 1000,
    centering = TRUE,
    debug = FALSE,
    aderivs = TRUE
  )

  tlst_p1k3_cp(
    n,
    x,
    t,
    d1 = 0.01,
    d2 = 0.01,
    fd3 = 0.01,
    kdf = 10,
    debug = FALSE
  )
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that `length(t)=length(x)`
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
n0	an index for the predictor (specify either `t0` or `n0` but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter
d2	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the second parameter

fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
kdf	the known degrees of freedom parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The t distribution with a predictor (also known as the location-scale t distribution with a predictor, hence the name `lst`), has probability density function

$$f(x;a,b,\sigma) = \frac{\Gamma((\nu+1)/2)}{\sqrt{\pi\nu}\sigma\Gamma(\nu/2)}\left(1+\frac{(x-\mu(a,b))^2}{\sigma^2\nu}\right)^{(\nu+1)/2}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, and $\sigma > 0$ is the scale parameter. We consider the degrees of freedom ν to be known (hence the `k3` in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson `<stephen.jewson@gmail.com>`

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),

- Weibull with linear predictor on the scale (`weibull_p2`),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d63lst_p1k3_example_data_v1_x
tt=fitdistcp::d63lst_p1k3_example_data_v1_t
p=c(1:9)/10
n0=10
q=qlst_p1k3_cp(x,tt,n0=n0,p=p,kdf=5,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qlst_p1k3_cp)",
main="t w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`lst_p1k3_f1f` *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
lst_p1k3_f1f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter

Value

Matrix

`lst_p1k3_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
lst_p1k3_f1fa(x, t, v1, v2, v3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

Vector

`lst_p1k3_f2f` *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
lst_p1k3_f2f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter

Value

3d array

`lst_p1k3_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
lst_p1k3_f2fa(x, t, v1, v2, v3, kdf)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`kdf`	the known degrees of freedom parameter

Value

Matrix

`lst_p1k3_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_p1k3_fd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

`lst_p1k3_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_p1k3_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

`lst_p1k3_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
lst_p1k3_ldd(x, t, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Square scalar matrix

lst_p1k3_ldda — *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
lst_p1k3_ldda(x, t, v1, v2, v3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_p1k3_lddd — *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
lst_p1k3_lddd(x, t, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Cubic scalar array

---

lst_p1k3_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
lst_p1k3_lddda(x, t, v1, v2, v3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_p1k3_lmn   *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lst_p1k3_lmn(x, t, v1, d1, v2, d2, v3, fd3, kdf, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

lst_p1k3_lmnp   *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lst_p1k3_lmnp(x, t, v1, d1, v2, d2, v3, fd3, kdf, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

lst_p1k3_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
lst_p1k3_logf(params, x, t, kdf)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors
kdf	the known degrees of freedom parameter

Value

Scalar value.

`lst_p1k3_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_p1k3_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

`lst_p1k3_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_p1k3_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

---

lst_p1k3_loglik *LST-with-p1 observed log-likelihood function*

Description

LST-with-p1 observed log-likelihood function

Usage

```
lst_p1k3_loglik(vv, x, t, kdf)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kdf	the known degrees of freedom parameter

Value

Scalar value.

`lst_p1k3_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lst_p1k3_logscores(logscores, x, t, d1, d2, fd3, kdf, aderivs)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`lst_p1k3_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
lst_p1k3_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Matrix

---

lst_p1k3_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
lst_p1k3_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_p1k3_p1f — *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
lst_p1k3_p1f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

y	value of random variable
t0	value of predictor
v1	first parameter
d1	delta for numerical derivative
v2	second parameter
d2	delta for numerical derivative
v3	third parameter
fd3	fractional delta for numerical derivative
kdf	the known number of degrees of freedom

Value

Matrix

lst_p1k3_p2f — *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
lst_p1k3_p2f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

---

lst_p1k3_predictordata

Predicted Parameter and Generalized Residuals

Description

Predicted Parameter and Generalized Residuals

Usage

```
lst_p1k3_predictordata(predictordata, x, t, t0, params, kdf)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kdf	the known degrees of freedom parameter

Value

Two vectors

`lst_p1k3_setics` *Set initial conditions*

Description

Set initial conditions

Usage

```
lst_p1k3_setics(x, t, ics)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`ics`	initial conditions for the maximum likelihood search

Value

Vector

`lst_p1k3_waic` *Waic*

Description

Waic

Usage

```
lst_p1k3_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  kdf,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1hat`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3hat`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad`	derivative of the log prior
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`makemuhat0` *Make muhat0*

Description

Make muhat0

Usage

```
makemuhat0(t0, n0, t, mle_params)
```

Arguments

`t0`	the value of the predictor vector at which to make the prediction (if n0 not specified)
`n0`	the position in the predictor vector at which to make the prediction (positive integer less than or equal to the length of x) (if t0 not specified)
`t`	predictor
`mle_params`	MLE params

Value

Scalar

`makeq` *Calculates quantiles from simulations by inverting the Hazen CDF*

Description

Calculates quantiles from simulations by inverting the Hazen CDF

Usage

```
makeq(yy, pp)
```

Arguments

`yy`	vector of samples
`pp`	vector of probabilities

Value

Vector

`maket0` *Determine t0*

Description

Determine t0

Usage

```
maket0(t0, n0, t)
```

Arguments

`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`n0`	an index for the predictor (specify either `t0` or `n0` but not both)
`t`	a vector or matrix of predictors

Value

Scalar

maketa0 *Make ta0*

Description

Make ta0

Usage

```
maketa0(t0, n0, t)
```

Arguments

t0	the value of the predictor vector at which to make the prediction (if n0 not specified)
n0	the position in the predictor vector at which to make the prediction (positive integer less than or equal to the length of x) (if t0 not specified)
t	predictor

Value

Scalar

make_cwaic *Make WAIC*

Description

Make WAIC

Usage

```
make_cwaic(x, fhatx, lddi, lddd, f1f, lambdad, f2f, dim)
```

Arguments

x	the training data
fhatx	density of x at the maximum likelihood parameters
lddi	inverse of the second derivative log-likelihood matrix
lddd	the third derivative log-likelihood tensor
f1f	the f1 term from DMGS equation 2.1
lambdad	the slope of the log prior
f2f	the f2 term from DMGS equation 2.1
dim	number of free parameters

Value

Two scalars

`make_maic` *Calculate MAIC*

Description

Calculate MAIC

Usage

```
make_maic(ml_value, nparams)
```

Arguments

`ml_value`	maximum of the likelihood
`nparams`	number of parameters

Value

Vector of 3 values Returns the two compoments of MAIC, and their sum

---

`make_se` *Make Standard Errors from lddi*

Description

Make Standard Errors from lddi

Usage

```
make_se(nx, lddi)
```

Arguments

`nx`	length of training data
`lddi`	the inverse log-likelihood matrix

Value

Vector

make_waic *Make WAIC*

Description

Make WAIC

Usage

```
make_waic(x, fhatx, lddi, lddd, f1f, lambdad, f2f, dim)
```

Arguments

x	the training data
fhatx	density of x at the maximum likelihood parameters
lddi	inverse of the second derivative log-likelihood matrix
lddd	the third derivative log-likelihood tensor
f1f	the f1 term from DMGS equation 2.1
lambdad	the slope of the log prior
f2f	the f2 term from DMGS equation 2.1
dim	number of free parameters

Value

Two scalars

man *A blank function I use for setting up the man page information*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y

- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
man(
  x,
  t,
  t1,
  t2,
  t3,
  t0,
  t01,
  t02,
  t03,
  t10,
  t20,
  n0,
  n01,
  n02,
  n03,
  n10,
  n20,
  p,
  n,
  y,
  ics,
  kloc,
  kscale,
  kshape,
  kdf,
  kbeta,
  d1,
  fd1,
  d2,
  fd2,
  d3,
  fd3,
  d4,
  fd4,
  d5,
```

```
  fd5,
  d6,
  fd6,
  fdalpha,
  minxi,
  maxxi,
  dlogpi,
  means,
  waicscores,
  logscores,
  extramodels,
  pdf,
  customprior,
  dmgs,
  mlcp,
  predictordata,
  centering,
  nonnegslopesonly,
  rnonnegslopesonly,
  prior,
  debug,
  rust,
  nrust,
  pwm,
  unbiasedv,
  aderivs
)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t3	a vector of predictors for the shape, such that length(t3)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
t10	a single value of the predictor for the mean (specify either t10 or n10 but not both)
t20	a single value of the predictor for the sd (specify either t20 or n20 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)

n03	an index for the predictor (specify either t03 or n03 but not both)
n10	an index for the predictor for the mean (specify either t10 or n10 but not both)
n20	an index for the predictor for the sd (specify either t20 or n20 but not both)
p	a vector of probabilities at which to generate predictive quantiles
n	the number of random samples required
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
kloc	the known location parameter
kscale	the known scale parameter
kshape	the known shape parameter
kdf	the known degrees of freedom parameter
kbeta	the known beta parameter
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
d3	if aderivs=FALSE, the delta used for numerical derivatives with respect to the third parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
d4	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fourth parameter
fd4	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the fourth parameter
d5	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fifth parameter
fd5	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the fourth parameter
d6	if aderivs=FALSE, the delta used for numerical derivatives with respect to the sixth parameter
fd6	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the fourth parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)

dlogpi	gradient of the log prior
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
customprior	a custom value for the slope of the log prior at the maxlik estimate
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
nonnegslopesonly	logical that indicates whether to disallow non-negative slopes
rnonnegslopesonly	logical that indicates whether to disallow non-negative slopes
prior	logical indicating which prior to use
debug	logical for turning on debug messages
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
pwm	logical for whether to include PWM results (longer runtime)
unbiasedv	logical for whether to include unbiased variance results in norm
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.

- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

`q****` optionally returns the following, for EVT models only:

- `cp_pdf`: the density function at quantiles corresponding to input probabilities `p`. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If `extramodels=TRUE`:

- `flat_quantiles`: predictive quantiles calculated from Bayesian integration with a flat prior.
- `rh_ml_quantiles`: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.

- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The `cp` outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <`stephen.jewson@gmail.com`>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

---

man1f *Return message for f1f, p1f, mu1f*

Description

Return message for f1f, p1f, mu1f

Usage

```
man1f()
```

Value

Matrix

---

man2f *Return message for f2f, p2f, mu2f*

Description

Return message for f2f, p2f, mu2f

Usage

```
man2f()
```

Value

3d array

mandsub *Return message for dsub*

Description

Return message for dsub

Usage

```
mandsub()
```

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

manf *Blank function I use for setting up the man page information for the functions*

Description

Blank function I use for setting up the man page information for the functions

Usage

```
manf(
  dim,
  vv,
  ml_params,
  nx,
  nxx,
  x,
  xx,
  t,
  t1,
  t2,
  t3,
  tt,
  tt1,
  tt2,
  tt3,
  tt2d,
  tt3d,
  t0,
  t01,
```

```
t02,
t03,
t10,
t20,
t30,
n0,
n10,
n20,
p,
n,
y,
ics,
ta,
ta0,
muhat0,
v1,
v1hat,
v1h,
d1,
fd1,
v2,
v2hat,
v2h,
d2,
fd2,
v3,
v3hat,
v3h,
d3,
fd3,
v4,
v4hat,
v4h,
d4,
fd4,
v5,
v5hat,
v5h,
d5,
v6,
v6hat,
v6h,
d6,
minxi,
maxxi,
ximin,
ximax,
fdalpha,
```

```
kscale,
kloc,
kshape,
kdf,
kbeta,
alpha,
ymn,
slope,
mu,
sigma,
sigma1,
sigma2,
scale,
shape,
xi,
xi1,
xi2,
lambda,
log,
mm,
nn,
rr,
lddi,
lddi_k2,
lddi_k3,
lddi_k4,
lddd,
lddd_k2,
lddd_k3,
lddd_k4,
lambdad,
lambdad_cp,
lambdad_rhp,
lambdad_flat,
lambdad_rh_mle,
lambdad_rh_flat,
lambdad_jp,
lambdad_custom,
means,
waicscores,
logscores,
extramodels,
pdf,
predictordata,
nonnegslopesonly,
rnonnegslopesonly,
customprior,
prior,
```

```
    params,
    yy,
    pp,
    dlogpi,
    debug,
    centering,
    aderivs
)
```

Arguments

`dim`	number of parameters
`vv`	parameters
`ml_params`	parameters
`nx`	length of training data
`nxx`	length of training data
`x`	a vector of training data values
`xx`	a vector of training data values
`t`	a vector or matrix of predictors
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`t3`	a vector of predictors for the shape
`tt`	a vector of predictors
`tt1`	a vector of predictors for the mean
`tt2`	a vector of predictors for the sd
`tt3`	a vector of predictors for the shape
`tt2d`	a matrix of predictors (nx by 2)
`tt3d`	a matrix of predictors (nx by 3)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`t03`	a single value of the predictor (specify either `t03` or `n03` but not both)
`t10`	a single value of the predictor for the mean (specify either `t10` or `n10` but not both)
`t20`	a single value of the predictor for the sd (specify either `t20` or `n20` but not both)
`t30`	a single value of the predictor for the shape (specify either `t30` or `n30` but not both)
`n0`	an index for the predictor (specify either `t0` or `n0` but not both)
`n10`	an index for the predictor for the mean (specify either `t10` or `n10` but not both)
`n20`	an index for the predictor for the sd (specify either `t10` or `n10` but not both)

p	a vector of probabilities at which to generate predictive quantiles
n	number of random samples required
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
ta	predictor residuals
ta0	predictor residual at the point being predicted
muhat0	muhat at the point being predicted
v1	first parameter
v1hat	first parameter
v1h	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
v2hat	second parameter
v2h	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
v3hat	third parameter
v3h	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
v4hat	fourth parameter
v4h	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
fd4	the fractional delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
v5hat	fifth parameter
v5h	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
v6hat	sixth parameter
v6h	sixth parameter

d6	the delta used in the numerical derivatives with respect to the parameter
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
ximin	minimum value of shape parameter xi
ximax	maximum value of shape parameter xi
fdalpha	the fractional delta used in the numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
kscale	the known scale parameter
kloc	the known location parameter
kshape	the known shape parameter
kdf	the known degrees of freedom parameter
kbeta	the known beta parameter
alpha	a vector of values of alpha (one minus probability)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
scale	the scale parameter of the distribution
shape	the shape parameter of the distribution
xi	the shape parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution
lambda	the lambda parameter of the distribution
log	logical for the density evaluation
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate
lddi	inverse observed information matrix
lddi_k2	inverse observed information matrix, fixed shape parameter
lddi_k3	inverse observed information matrix, fixed shape parameter
lddi_k4	inverse observed information matrix, fixed shape parameter
lddd	third derivative of log-likelihood
lddd_k2	third derivative of log-likelihood, fixed shape parameter
lddd_k3	third derivative of log-likelihood, fixed shape parameter
lddd_k4	third derivative of log-likelihood, fixed shape parameter

lambdad	derivative of the log prior
lambdad_cp	derivative of the log prior
lambdad_rhp	derivative of the log RHP prior
lambdad_flat	derivative of the log flat prior
lambdad_rh_mle	derivative of the log CRHP-MLE prior
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
lambdad_jp	derivative of the log JP prior
lambdad_custom	custom value of the derivative of the log prior
means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
extramodels	logical that indicates whether to add three additional prediction models
pdf	logical that indicates whether to return density functions evaluated at quantiles specified by input probabilities
predictordata	logical that indicates whether to calculate and return predictordata
nonnegslopesonly	logical that indicates whether to disallow non-negative slopes
rnonnegslopesonly	logical that indicates whether to disallow non-negative slopes
customprior	a custom value for the slope of the log prior at the maxlik estimate
prior	logical indicating which prior to use
params	model parameters for calculating logf
yy	vector of samples
pp	vector of probabilities
dlogpi	gradient of the log prior
debug	debug flag
centering	indicates whether the routine should center the data or not
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

No return value

`manldd` *Return message for ldd*

Description

Return message for ldd

Usage

```
manldd()
```

Value

Square scalar matrix

`manlddd` *Return message for lddd*

Description

Return message for lddd

Usage

```
manlddd()
```

Value

Cubic scalar array

`manlnn` *Return message for lnn*

Description

Return message for lnn

Usage

```
manlnn()
```

Value

Scalar value

`manlnnn` *Return message for lnnn*

Description

Return message for lnnn

Usage

```
manlnnn()
```

Value

Scalar value

`manlogf` *Return message for Logf*

Description

Return message for Logf

Usage

```
manlogf()
```

Value

Scalar value.

`manloglik` *Return message for loglik*

Description

Return message for loglik

Usage

```
manloglik()
```

Value

Scalar value.

`manlogscores` *Return message for logscores*

Description

Return message for logscores

Usage

```
manlogscores()
```

Value

Two scalars

`manmeans` *Return message for means*

Description

Return message for means

Usage

```
manmeans()
```

Value

Two scalars

`manpredictor` *Return message for predictor.*

Description

Return message for predictor.

Usage

```
manpredictor()
```

Value

Two vectors

`manvector` *Return message for vector*

Description

Return message for vector

Usage

```
manvector()
```

Value

Vector

`manwaic` *Return message for WAIC*

Description

Return message for WAIC

Usage

```
manwaic()
```

Value

Two numeric values.

`movexiawayfromzero` *Move xi away from zero a bit*

Description

Move xi away from zero a bit

Usage

```
movexiawayfromzero(xi)
```

Arguments

`xi`	xi

Value

Scalar

ms_flat_1tail *Illustration of Model Selection Among 10 One Tail Distributions from the* `fitdistcp` *Package*

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, for 10 one tailed models in the `fitdistcp` package (although for the GPD, the logscore is NA for mathematical reasons).

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the `fitdistcp` routines.

The input data may be automatically shifted so that the minimum value is positive.

For the Pareto, the data may be further shifted so that the minimum value is slightly greater than 1.

Usage

```
ms_flat_1tail(x)
```

Arguments

x	data vector

Details

The 10 models are: `exp`, `pareto_k2`, `halfnorm`, `lnorm`, `frechet_k1`, `weibull`, `gamma`, `invgamma`, `invgauss` and `gpd_k1`.

Value

Plots QQ plots to the screen, for each of the models, and returns a data frame containing

- MLE parameter values
- AIC scores (times -0.5), AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores, logscore weights
- maximum likelihood and calibrating prior means
- maximum likelihood and calibrating prior standard deviations

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
 # because it's too slow for CRAN
set.seed(1)
nx=50
x=rlnorm(nx)
print(ms_flat_1tail(x))
```

`ms_flat_2tail` *Illustration of Model Selection Among 18 Distributions from the* `fitdistcp` *Package*

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, for 7 two tailed models in the `fitdistcp` packages

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the `fitdistcp` routines.

Usage

```
ms_flat_2tail(x)
```

Arguments

`x`	data vector

Details

The 7 models are: `norm`, `gnorm_k3`, `gumbel`, `logis`, `lst_k3`, `cauchy`, `gev`

Value

Plots QQ plots to the screen, for each of the models, and returns a data frame containing

- AIC scores (times -0.5), AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores, logscore weights
- maximum likelihood and calibrating prior means
- maximum likelihood and calibrating prior standard deviations

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
 # because it's too slow for CRAN
set.seed(1)
nx=50
x=rnorm(nx)
print(ms_flat_2tail(x))
```

`ms_predictors_1tail` *Model Selection Among 5 Distributions with predictors from the* `fitdistcp` *Package*

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, t, for 5 one tailed models with predictors in the `fitdistcp` package.

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the `fitdistcp` routines.

The input data may be automatically shifted so that the minimum value is positive.

For the Pareto, the data is so that the minimum value is slightly greater than 1.

Usage

```
ms_predictors_1tail(x, t)
```

Arguments

`x`	data vector
`t`	predictor vector

Details

The 5 models are: `exp_p1`, `pareto_p1k2`, `lnorm_p1`, `frechet_p2k1`, `weibull_p2`.

Value

Plots QQ plots to the screen, for each of the 5 models, and returns a data frame containing

- AIC scores, AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores and logscore weights

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
 # because it's too slow for CRAN
set.seed(2)
nx=100
predictor=c(1:nx)/nx
x=rlnorm(nx,meanlog=predictor,sdlog=0.1)
print(ms_predictors_1tail(x,predictor))
```

---

ms_predictors_2tail — *Model Selection Among 6 Distributions with predictors from the* `fitdistcp` *Package*

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, t, for 6 two tail models with predictors in the `fitdistcp` packages (although for the GEV, the logscore is NA for mathematical reasons).

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the `fitdistcp` routines.

GEVD is temperamental in that it doesn't work if the shape parameter is extreme.

Usage

```
ms_predictors_2tail(x, t)
```

Arguments

x	data vector
t	predictor vector

Details

The 11 models are: `norm_p1`, `gumbel_p1`, `logis_p1`, `lst_k3_p1`, `cauchy_p1` and `gev_p1`.

Value

Plots QQ plots to the screen, for each of the 6 models, and returns a data frame containing

- AIC scores, AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores and logscore weights

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
 # because it's too slow for CRAN
set.seed(2)
nx=100
predictor=c(1:nx)/nx
x=rnorm(nx,mean=predictor,sd=1)
print(ms_predictors_2tail(x,predictor))
```

---

nopdfcdfmsg — *Message to explain why GEV and GPD* `d***` *and* `p***` *routines don't return DMGS pdfs and cdfs*

Description

Message to explain why GEV and GPD `d***` and `p***` routines don't return DMGS pdfs and cdfs

Usage

```
nopdfcdfmsg(yy, pp)
```

Arguments

yy	vector of samples
pp	vector of probabilities

Value

String

norm_cp *Normal Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  unbiasedv = FALSE,
  debug = FALSE,
  aderivs = TRUE
)

rnorm_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE, aderivs = TRUE)
```

```
dnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

pnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE, aderivs = TRUE)

tnorm_cp(n, x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
unbiasedv	logical for whether to include unbiased variance results in norm
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.

- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The normal distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d30norm_example_data_v1
p=c(1:9)/10
q=qnorm_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qnorm_cp)",
main="Normal: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`norm_dmgs_cp` *Normal Distribution Predictions Based on a Calibrating Prior, using DMGS (for testing only)*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_dmgs_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rnorm_dmgs_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dnorm_dmgs_cp(x, y = x, d1 = 0.01, fd2 = 0.01, debug = FALSE, aderivs = TRUE)

pnorm_dmgs_cp(x, y = x, d1 = 0.01, fd2 = 0.01, debug = FALSE, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`p`	a vector of probabilities at which to generate predictive quantiles
`d1`	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter
`fd2`	if `aderivs=FALSE`, the fractional delta used for numerical derivatives with respect to the second parameter
`means`	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
`waicscores`	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
`logscores`	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
`dmgs`	logical that indicates whether DMGS calculations should be run or not (longer run time)
`debug`	logical for turning on debug messages

aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The normal distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),

- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d30norm_example_data_v1
p=c(1:9)/10
q=qnorm_dmgs_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qnorm_dmgs_cp)",
main="Normal_DMGS: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

---

norm_dmgs_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
norm_dmgs_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

---

norm_dmgs_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
norm_dmgs_logscores(logscores, x, d1 = 0.01, fd2 = 0.01)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Two scalars

---

norm_dmgs_means *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
norm_dmgs_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`ml_params`	parameters
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad_rhp`	derivative of the log RHP prior
`nx`	length of training data
`dim`	number of parameters

Value

Two scalars

`norm_dmgs_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
norm_dmgs_mu1f(alpha, v1, d1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

norm_dmgs_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
norm_dmgs_mu2f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

norm_dmgs_p1f *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
norm_dmgs_p1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`norm_dmgs_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
norm_dmgs_p2f(y, v1, d1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`norm_dmgs_waic` *Waic*

Description

Waic

Usage

```
norm_dmgs_waic(
  waicscores,
  x,
  v1hat,
  d1,
  v2hat,
  fd2,
  lddi,
```

```
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

norm_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
norm_f1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`norm_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
norm_f1fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

`norm_f2f` *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
norm_f2f(y, v1, d1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

norm_f2fa — *The second derivative of the density*

Description

The second derivative of the density

Usage

```
norm_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

norm_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_gg *Second derivative matrix of the expected per-observation log-likelihood*

Description

Second derivative matrix of the expected per-observation log-likelihood

Usage

```
norm_gg(nx, v1, d1, v2, fd2)
```

Arguments

nx	length of training data
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

norm_gmn *One component of the second derivative of the expected log-likelihood*

Description

One component of the second derivative of the expected log-likelihood

Usage

```
norm_gmn(alpha, v1, d1, v2, fd2, mm, nn)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

norm_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
norm_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`norm_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
norm_ldda(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

`norm_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
norm_lddd(x, v1, d1, v2, fd2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

`norm_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
norm_lddda(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

3d array

`norm_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

One component of the second derivative of the normalized log-likelihood

Usage

```
norm_lmn(x, v1, d1, v2, fd2, mm, nn)
```



```
norm_lmn(x, v1, d1, v2, fd2, mm, nn)
```


Arguments

`x`	a vector of training data values
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate

Value

Scalar value

`norm_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

One component of the third derivative of the normalized log-likelihood

Usage

```
norm_lmnp(x, v1, d1, v2, fd2, mm, nn, rr)

norm_lmnp(x, v1, d1, v2, fd2, mm, nn, rr)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate
`rr`	an index for which derivative to calculate

Value

Scalar value

`norm_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
norm_logf(params, x)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values

Value

Scalar value.

`norm_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_logfdd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

norm_logfddd	*Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

norm_logscores	*Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
norm_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

norm_ml_params *Maximum likelihood estimator*

Description

Maximum likelihood estimator

Usage

```
norm_ml_params(x)
```

Arguments

x	a vector of training data values

Value

Scalar value.

norm_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
norm_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

norm_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
norm_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

norm_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
norm_p1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

norm_p1_cp *Normal Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  centering = TRUE,
```

```
  debug = FALSE,
  aderivs = TRUE
)

rnorm_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tnorm_p1_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that `length(t)=length(x)`

t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.

- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The normal distribution with a predictor has probability density function

$$f(x; a, b, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu(a,b))^2/(2\sigma^2)}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, modelled as a function of parameters a, b and predictor t, and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b, \sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson `<stephen.jewson@gmail.com>`

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d60norm_p1_example_data_v1_x
tt=fitdistcp::d60norm_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qnorm_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qnorm_p1_cp)",
main="Normal w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

norm_p1_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
norm_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

norm_p1_f1fa *The first derivative of the density*

Description

The first derivative of the density

The first derivative of the density

Usage

```
norm_p1_f1fa(x, t, v1, v2, v3)
```


```
norm_p1_f1fa(x, t, v1, v2, v3)
```


Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_f2f *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
norm_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

norm_p1_f2fa *The second derivative of the density*

Description

The second derivative of the density

The second derivative of the density

Usage

```
norm_p1_f2fa(x, t, v1, v2, v3)
```


```
norm_p1_f2fa(x, t, v1, v2, v3)
```


Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_fd(x, t, v1, v2, v3)

norm_p1_fd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

norm_p1_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_fdd(x, t, v1, v2, v3)

norm_p1_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

norm_p1_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
norm_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

`norm_p1_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

The second derivative of the normalized log-likelihood

Usage

```
norm_p1_ldda(x, t, v1, v2, v3)

norm_p1_ldda(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`norm_p1_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
norm_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

`norm_p1_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

The third derivative of the normalized log-likelihood

Usage

```
norm_p1_lddda(x, t, v1, v2, v3)
```


```
norm_p1_lddda(x, t, v1, v2, v3)
```


Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

norm_p1_lmn	*One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
norm_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

norm_p1_lmnp	*One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
norm_p1_lmnp(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

norm_p1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
norm_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

---

norm_p1_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_logfdd(x, t, v1, v2, v3)

norm_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

norm_p1_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_logfddd(x, t, v1, v2, v3)

norm_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

---

norm_p1_loglik *Normal-with-p1 observed log-likelihood function*

Description

Normal-with-p1 observed log-likelihood function

Usage

```
norm_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

norm_p1_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
norm_p1_logscores(logscores, x, t, aderivs = TRUE)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

norm_p1_mlparams *Maximum likelihood estimator*

Description

Maximum likelihood estimator

Usage

```
norm_p1_mlparams(x, t)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Vector

norm_p1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Minus the first derivative of the cdf, at alpha

Usage

```
norm_p1_mu1fa(alpha, t, v1, v2, v3)

norm_p1_mu1fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Minus the second derivative of the cdf, at alpha

Usage

```
norm_p1_mu2fa(alpha, t, v1, v2, v3)

norm_p1_mu2fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

norm_p1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

The first derivative of the cdf

Usage

```
norm_p1_p1fa(x, t, v1, v2, v3)

norm_p1_p1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

The second derivative of the cdf

Usage

```
norm_p1_p2fa(x, t, v1, v2, v3)

norm_p1_p2fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_pd(x, t, v1, v2, v3)

norm_p1_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

---

norm_p1_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_pdd(x, t, v1, v2, v3)

norm_p1_pdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_predictordata *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
norm_p1_predictordata(x, t, t0, params)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

norm_p1_waic *Waic*

Description

Waic

Usage

```
norm_p1_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  aderivs = TRUE
)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1hat`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3hat`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`norm_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
norm_p2fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

---

norm_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

---

norm_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_unbiasedv_params *Method of moments estimator*

Description

Method of moments estimator

Usage

```
norm_unbiasedv_params(x)
```

Arguments

x	a vector of training data values

Value

Vector

norm_waic *Waic*

Description

Waic

Usage

```
norm_waic(waicscores, x, v1hat, d1, v2hat, fd2, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

pareto_k2_cp *Pareto Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qpareto_k2_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kscale = 1,
  fd1 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rpareto_k2_cp(
  n,
```

```
  x,
  kscale = 1,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dpareto_k2_cp(
  x,
  y = x,
  kscale = 1,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

ppareto_k2_cp(
  x,
  y = x,
  kscale = 1,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tpareto_k2_cp(n, x, kscale = 1, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kscale	the known scale parameter
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations

debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.

- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Pareto distribution has various forms. The form we are using has exceedance distribution function

$$S(x;\alpha) = \left(\frac{\sigma}{x}\right)^{\alpha}$$

where $x \geq \sigma$ is the random variable and $\alpha > 0, \sigma > 0$ are the shape and scale parameters. We consider the scale parameter σ to be known (hence the k2 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\alpha) \propto \frac{1}{\alpha}$$

as given in Jewson et al. (2025). Some others authors may refer to the shape and scale parameters as the scale and location parameters, respectively.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),

- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d11pareto_k2_example_data_v1
p=c(1:9)/10
q=qpareto_k2_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles)
xmax=max(q$ml_quantiles,q$cp_quantiles)
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qpareto_k2_cp)",
main="Pareto: quantile estimates")
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

pareto_k2_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
pareto_k2_f1f(y, v1, fd1, kscale)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter

Value

Matrix

`pareto_k2_f1fa` *The first derivative of the density*

Description

The first derivative of the density

Usage

```
pareto_k2_f1fa(x, v1, kscale)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`kscale`	the known scale parameter

Value

Vector

`pareto_k2_f2f` *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
pareto_k2_f2f(y, v1, fd1, kscale)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

3d array

---

pareto_k2_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
pareto_k2_f2fa(x, v1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

Matrix

---

pareto_k2_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`pareto_k2_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`pareto_k2_l111` *Third derivative of the normalized log-likelihood*

Description

Third derivative of the normalized log-likelihood

Usage

```
pareto_k2_l111(x, v1, fd1, kscale)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter

Value

Scalar value

`pareto_k2_ldd` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
pareto_k2_ldd(x, v1, fd1, kscale)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter

Value

Square scalar matrix

`pareto_k2_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
pareto_k2_ldda(x, v1, kscale)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`kscale`	the known scale parameter

Value

Matrix

`pareto_k2_lddd` *Third derivative tensor of the log-likelihood*

Description

Third derivative tensor of the log-likelihood

Usage

```
pareto_k2_lddd(x, v1, fd1, kscale)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter

Value

Cubic scalar array

pareto_k2_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
pareto_k2_lddda(x, v1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

3d array

pareto_k2_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
pareto_k2_logf(params, x, kscale)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kscale	the known scale parameter

Value

Scalar value.

pareto_k2_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

pareto_k2_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

`pareto_k2_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
pareto_k2_logscores(logscores, x, kscale)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`kscale`	the known scale parameter

Value

Two scalars

`pareto_k2_ml_params` *Maximum likelihood estimator*

Description

Maximum likelihood estimator

Usage

```
pareto_k2_ml_params(x, kscale)
```

Arguments

`x`	a vector of training data values
`kscale`	the known scale parameter

Value

Scalar value.

`pareto_k2_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
pareto_k2_mu1fa(alpha, v1, kscale)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`kscale`	the known scale parameter

Value

Vector

`pareto_k2_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
pareto_k2_mu2fa(alpha, v1, kscale)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`kscale`	the known scale parameter

Value

Matrix

pareto_k2_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
pareto_k2_p1fa(x, v1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

Vector

pareto_k2_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
pareto_k2_p2fa(x, v1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

Matrix

`pareto_k2_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`pareto_k2_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

pareto_k2_waic *Waic*

Description

Waic

Usage

```
pareto_k2_waic(waicscores, x, v1hat, fd1, kscale, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

pareto_p1k2_cp *Pareto Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.

- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qpareto_p1k2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  kscale = 1,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rpareto_p1k2_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  kscale = 1,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
```

```
)

dpareto_p1k2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  kscale = 1,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

ppareto_p1k2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  kscale = 1,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tpareto_p1k2_cp(n, x, t, d1 = 0.01, d2 = 0.01, kscale = 1, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that `length(t)=length(x)`
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
n0	an index for the predictor (specify either `t0` or `n0` but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the first parameter
d2	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the second parameter

kscale	the known scale parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.

- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Pareto distribution with a predictor has various forms. The form we are using has exceedance distribution function

$$S(x; a, b) = \left(\frac{\sigma}{x}\right)^{\alpha(a,b)}$$

where $x \geq \sigma$ is the random variable, $\alpha = \exp(-a - bt)$ is the shape parameter, modelled as a function of parameters a, b, and σ is the scale parameter. We consider the scale parameter σ to be known (hence the k2 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b) \propto 1$$

as given in Jewson et al. (2025). Note that others authors have referred to the shape and scale parameters as the scale and location parameters, respectively.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d56pareto_p1k2_example_data_v1_x
tt=fitdistcp::d56pareto_p1k2_example_data_v1_t
p=c(1:9)/10
n0=10
```

```
q=qpareto_p1k2_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qpareto_p1k2_cp)",
main="Pareto w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`pareto_p1k2_f1f` *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
pareto_p1k2_f1f(y, t0, v1, d1, v2, d2, kscale)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter

Value

Matrix

pareto_p1k2_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
pareto_p1k2_f1fa(x, t, v1, v2, kscale)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`kscale`	the known scale parameter

Value

Vector

pareto_p1k2_f2f *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
pareto_p1k2_f2f(y, t0, v1, d1, v2, d2, kscale)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter

Value

3d array

`pareto_p1k2_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
pareto_p1k2_f2fa(x, t, v1, v2, kscale)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`kscale`	the known scale parameter

Value

Matrix

`pareto_p1k2_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_fd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`pareto_p1k2_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`pareto_p1k2_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
pareto_p1k2_ldd(x, t, v1, d1, v2, d2, kscale)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

Square scalar matrix

---

pareto_p1k2_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
pareto_p1k2_ldda(x, t, v1, v2, kscale)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Matrix

pareto_p1k2_lddd — *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
pareto_p1k2_lddd(x, t, v1, d1, v2, d2, kscale)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

Cubic scalar array

pareto_p1k2_lddda — *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
pareto_p1k2_lddda(x, t, v1, v2, kscale)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

3d array

---

pareto_p1k2_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
pareto_p1k2_lmn(x, t, v1, d1, v2, d2, kscale, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

---

pareto_p1k2_lmnp *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
pareto_p1k2_lmnp(x, t, v1, d1, v2, d2, kscale, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

pareto_p1k2_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
pareto_p1k2_logf(params, x, t, kscale)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors
kscale	the known scale parameter

Value

Scalar value.

pareto_p1k2_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

pareto_p1k2_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`pareto_p1k2_loglik` *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
pareto_p1k2_loglik(vv, x, t, kscale)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kscale	the known scale parameter

Value

Scalar value.

`pareto_p1k2_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
pareto_p1k2_logscores(logscores, x, t, d1, d2, kscale, aderivs, debug)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)
`debug`	debug flag

Value

Two scalars

`pareto_p1k2_means` *pareto_k1 distribution: RHP mean*

Description

pareto_k1 distribution: RHP mean

Usage

```
pareto_p1k2_means(
  means,
  t0,
  ml_params,
  lddi,
  lddd,
  lambdad_rhp,
  nx,
  dim = 2,
  kscale
)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ml_params`	parameters
`lddi`	inverse observed information matrix

lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kscale	the known scale parameter

Value

Two scalars

---

pareto_p1k2_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
pareto_p1k2_mu1f(alpha, t0, v1, d1, v2, d2, kscale)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

Matrix

pareto_p1k2_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
pareto_p1k2_mu1fa(alpha, t, v1, v2, kscale)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Vector

pareto_p1k2_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
pareto_p1k2_mu2f(alpha, t0, v1, d1, v2, d2, kscale)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter

Value

3d array

`pareto_p1k2_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
pareto_p1k2_mu2fa(alpha, t, v1, v2, kscale)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`kscale`	the known scale parameter

Value

Matrix

`pareto_p1k2_p1f` *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
pareto_p1k2_p1f(y, t0, v1, d1, v2, d2, kscale)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter

Value

Matrix

`pareto_p1k2_p1fa` *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
pareto_p1k2_p1fa(x, t, v1, v2, kscale)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`kscale`	the known scale parameter

Value

Vector

`pareto_p1k2_p2f` *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
pareto_p1k2_p2f(y, t0, v1, d1, v2, d2, kscale)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`kscale`	the known scale parameter

Value

3d array

`pareto_p1k2_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
pareto_p1k2_p2fa(x, t, v1, v2, kscale)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`kscale`	the known scale parameter

Value

Matrix

`pareto_p1k2_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_pd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`pareto_p1k2_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_pdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`pareto_p1k2_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
pareto_p1k2_predictordata(predictordata, x, t, t0, params, kscale)
```

Arguments

`predictordata`	logical that indicates whether to calculate and return predictordata
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`params`	model parameters for calculating logf
`kscale`	the known scale parameter

Value

Two vectors

`pareto_p1k2_waic` *Waic*

Description

Waic

Usage

```
pareto_p1k2_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  kscale,
  lddi,
  lddd,
  lambdad
)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1hat`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter

d2	the delta used in the numerical derivatives with respect to the parameter
kscale	the known scale parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

---

pcauchy_p1 *Cauchy-with-p1 distribution function*

Description

Cauchy-with-p1 distribution function

Usage

```
pcauchy_p1(x, t0, ymn, slope, scale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

`pexp_p1` *Exponential-with-p1 distribution function*

Description

Exponential-with-p1 distribution function

Usage

```
pexp_p1(x, t0, ymn, slope)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor

Value

Vector

`pfrechet_p2k1` *Frechet_k1-with-p2 distribution function*

Description

Frechet_k1-with-p2 distribution function

Usage

```
pfrechet_p2k1(x, t0, ymn, slope, lambda, kloc)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`lambda`	the lambda parameter of the distribution
`kloc`	the known location parameter

Value

Vector

pgev_p1 *GEVD-with-p1: Distribution function*

Description

GEVD-with-p1: Distribution function

Usage

```
pgev_p1(y, t0, ymn, slope, sigma, xi)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution

Value

Vector

pgev_p12 *GEVD-with-p1: Distribution function*

Description

GEVD-with-p1: Distribution function

Usage

```
pgev_p12(y, t1, t2, ymn, slope, sigma1, sigma2, xi)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi	the shape parameter of the distribution

Value

Vector

---

pgev_p123 *GEVD-with-p1: Distribution function*

Description

GEVD-with-p1: Distribution function

Usage

```
pgev_p123(y, t1, t2, t3, ymn, slope, sigma1, sigma2, xi1, xi2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution

Value

Vector

pgev_p1k3 — *GEV-with-known-shape-with-p1 distribution function*

Description

GEV-with-known-shape-with-p1 distribution function

Usage

```
pgev_p1k3(x, t0, ymn, slope, sigma, kshape)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma`	the sigma parameter of the distribution
`kshape`	the known shape parameter

Value

Vector

pgumbel_p1 — *Gumbel-with-p1 distribution function*

Description

Gumbel-with-p1 distribution function

Usage

```
pgumbel_p1(x, t0, ymn, slope, sigma)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma`	the sigma parameter of the distribution

Value

Vector

plnorm_p1 — *Normal-with-p1 distribution function*

Description

Normal-with-p1 distribution function

Usage

```
plnorm_p1(x, t0, ymn, slope, sigma)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

plogis_p1 — *Logistic-with-p1 distribution function*

Description

Logistic-with-p1 distribution function

Usage

```
plogis_p1(x, t0, ymn, slope, scale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

plst_p1k3	*LST-with-p1 distribution function*

Description

LST-with-p1 distribution function

Usage

```
plst_p1k3(x, t0, ymn, slope, sigma, kdf)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
kdf	the known degrees of freedom parameter

Value

Vector

pnorm_p1	*Normal-with-p1 distribution function*

Description

Normal-with-p1 distribution function

Usage

```
pnorm_p1(x, t0, ymn, slope, sigma)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

pnorm_p1_formula *Linear regression formula, densities*

Description

Linear regression formula, densities

Usage

```
pnorm_p1_formula(y, ta, ta0, nx, muhat0, v3hat)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
ta	predictor residuals
ta0	predictor residual at the point being predicted
nx	length of training data
muhat0	muhat at the point being predicted
v3hat	third parameter

Value

Vector

ppareto_p1k2 *pareto_k1-with-p2 distribution function*

Description

pareto_k1-with-p2 distribution function

Usage

```
ppareto_p1k2(x, t0, ymn, slope, kscale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
kscale	the known scale parameter

Value

Vector

punif_formula *Predictive CDFs*

Description

Predictive CDFs

Usage

```
punif_formula(x, y)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions

Value

Two vectors

pweibull_p2 *Weibull-with-p1 distribution function*

Description

Weibull-with-p1 distribution function

Usage

```
pweibull_p2(x, t0, shape, ymn, slope)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
shape	the shape parameter of the distribution
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor

Value

Vector

qcauchy_p1 *Cauchy-with-p1 quantile function*

Description

Cauchy-with-p1 quantile function

Usage

```
qcauchy_p1(p, t0, ymn, slope, scale)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

qexp_p1 *-with-p1 quantile function*

Description

-with-p1 quantile function

Usage

```
qexp_p1(p, t0, ymn, slope)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor

Value

Vector

qfrechet_p2k1 *Frechet_k1-with-p2 quantile function*

Description

Frechet_k1-with-p2 quantile function

Usage

```
qfrechet_p2k1(p, t0, ymn, slope, lambda, kloc)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
lambda	the lambda parameter of the distribution
kloc	the known location parameter

Value

Vector

qgamma_k1_ppm *Temporary dummy for one of the cp models*

Description

Temporary dummy for one of the cp models

Usage

```
qgamma_k1_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

qgamma_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgamma_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

qgev_k12_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_k12_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

qgev_mpd_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_mpd_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

`qgev_p1` *GEVD-with-p1: Quantile function*

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p1(p, t0, ymn, slope, sigma, xi)
```

Arguments

`p`	a vector of probabilities at which to generate predictive quantiles
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma`	the sigma parameter of the distribution
`xi`	the shape parameter of the distribution

Value

Vector

`qgev_p12` *GEVD-with-p1: Quantile function*

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p12(p, t1, t2, ymn, slope, sigma1, sigma2, xi)
```

Arguments

`p`	a vector of probabilities at which to generate predictive quantiles
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma1`	first coefficient for the sigma parameter of the distribution
`sigma2`	second coefficient for the sigma parameter of the distribution
`xi`	the shape parameter of the distribution

Value

Vector

---

qgev_p123 *GEVD-with-p1: Quantile function*

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p123(p, t1, t2, t3, ymn, slope, sigma1, sigma2, xi1, xi2)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution

Value

Vector

qgev_p1k3 *GEV-with-known-shape-with-p1 quantile function*

Description

GEV-with-known-shape-with-p1 quantile function

Usage

```
qgev_p1k3(p, t0, ymn, slope, sigma, kshape)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
kshape	the known shape parameter

Value

Vector

qgev_p1_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_p1_ppm(x, t, n0, p)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

qgev_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

---

qgpd_k1_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgpd_k1_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

qgumbel_p1 *Gumbel-with-p1 quantile function*

Description

Gumbel-with-p1 quantile function

Usage

```
qgumbel_p1(p, t0, ymn, slope, sigma)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

---

qlnorm_p1 — *Normal-with-p1 quantile function*

Description

Normal-with-p1 quantile function

Usage

```
qlnorm_p1(p, t0, ymn, slope, sigma)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

qlogis_p1 *Logistic-with-p1 quantile function*

Description

Logistic-with-p1 quantile function

Usage

```
qlogis_p1(p, t0, ymn, slope, scale)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

qlst_p1k3 *LST-with-p1 quantile function*

Description

LST-with-p1 quantile function

Usage

```
qlst_p1k3(p, t0, ymn, slope, sigma, kdf)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
kdf	the known degrees of freedom parameter

Value

Vector

`qnorm_p1` *Normal-with-p1 quantile function*

Description

Normal-with-p1 quantile function

Usage

```
qnorm_p1(p, t0, ymn, slope, sigma)
```

Arguments

`p`	a vector of probabilities at which to generate predictive quantiles
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma`	the sigma parameter of the distribution

Value

Vector

`qnorm_p1_formula` *Linear regression formula, quantiles*

Description

Linear regression formula, quantiles

Usage

```
qnorm_p1_formula(alpha, ta, ta0, nx, muhat0, v3hat)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`ta`	predictor residuals
`ta0`	predictor residual at the point being predicted
`nx`	length of training data
`muhat0`	muhat at the point being predicted
`v3hat`	third parameter

Value

Vector

qntt_ppm — *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qntt_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

qpareto_p1k2 *pareto_k1-with-p2 quantile function*

Description

pareto_k1-with-p2 quantile function

Usage

```
qpareto_p1k2(p, t0, ymn, slope, kscale)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
kscale	the known scale parameter

Value

Vector

qunif_formula *Predictive Quantiles*

Description

Predictive Quantiles

Usage

```
qunif_formula(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

Two vectors

qweibull_p2 *Weibull-with-p1 quantile function*

Description

Weibull-with-p1 quantile function

Usage

```
qweibull_p2(p, t0, shape, ymn, slope)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
shape	the shape parameter of the distribution
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor

Value

Vector

reltest *Evaluation of Reliability for Models in the* fitdistcp *Package*

Description

Uses simulations to evaluate the reliability of the predictive quantiles produced by the q****_cp routines in the fitdistcp package.

Usage

```
reltest(
  model = "exp",
  ntrials = 1000,
  nrepeats = 3,
  nx = 20,
  params = c(1),
  alpha = seq(0.005, 0.995, 0.005),
  plotflag = TRUE,
  verbose = TRUE,
  dmgs = TRUE,
  debug = FALSE,
  aderivs = TRUE,
  unbiasedv = FALSE,
  pwm = FALSE,
  minxi = -10,
  maxxi = 10
)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k1", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k3", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k4", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k4", "norm_p12", "lst_p12k5", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence. 3 is a good choice.
nx	the length of the training data to use.
params	values for the parameters for the specified distribution
alpha	the exceedance probability values at which to test
plotflag	logical to turn the plotting on and off
verbose	logical to turn loop counting on and off

dmgs	logical to turn DMGS calculations on and off (to optimize speed for maxlik only calculations)
debug	logical for turning debug messages on and off
aderivs	logical for whether to use analytic derivatives (instead of numerical)
unbiasedv	logical for whether to use the unbiased variance instead of maxlik (for the normal)
pwm	logical for whether to use PWM instead of maxlik (for the GEV)
minxi	minimum value for EVT shape parameter
maxxi	maximum value for EVT shape parameter

Details

The maximum likelihood quantiles (plotted in blue) do not give good reliability. They typically underestimate the tails (see panel (f)).

For "exp", "pareto_k1", "unif", "norm", "lnorm", "norm_p1" and "lnorm_p1", the calibrating prior quantiles are calculated using the right Haar prior and an exact solution for the Bayesian prediction integral. They will converge towards exact reliability with a large enough number of trials, for any sample size.

For "halfnorm", "norm_dmgs", "lnorm_dmgs", "gnorm_k3", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k3", "gumbel_p1", "logis_p1" and "lst_p1k4" "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k4", "norm_p12", "lst_p12k5" the calibrating prior quantiles are calculated using the right Haar prior, with the DMGS asymptotic solution for the Bayesian prediction integral. They will converge towards good reliability with a large enough number of trials, with the only deviation from exact reliability being due to the neglect of higher order terms in the asymptotic expansion. They will converge towards exact reliability with a large enough number of trials and a large enough sample size.

For "gamma", "invgamma", "invgauss", "gev", "gpd_k1" and "gev_p1", "gev_p12", "gev_p123", the calibrating prior quantiles are calculated using the "fitdistcp" recommended calibrating priors, with the DMGS asymptotic solution for the Bayesian prediction integral. The chosen priors give reasonably good reliability with a large enough number of trials, and for large sample sizes, but may give poor reliability for small sample sizes (e.g., n<20).

Value

A plot showing 9 different reliability checks, and a list containing various outputs, including the probabilities shown in the plot.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),

- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
set.seed(1)
# example 1
# -runs the default settings, which test reliability for the exponential distribution
reltest()
```

---

reltest2 *Evaluation of Reliability for Certain Additional Models in the* fitdistcp *Package*

Description

This routine is mainly for reproducing certain results in Jewson et al. (2025), and not of general interest.

It uses simulations to evaluate the reliability of the predictive quantiles produced by the qgev_cp, ggpd_cp and qgev_p1_cp routines in the fitdistcp package. For each model, results for 5 models are calculated. This is to illustrate that the calibrating prior predictions dominate the ml, flat, crhp_ml and jp predictions, in terms of reliability.

Usage

```
reltest2(
  model = "gev",
  ntrials = 100,
  nrepeats = 3,
  nx = 50,
  params = c(0, 1, 0),
  alpha = seq(0.005, 0.995, 0.005),
  plotflag = TRUE,
  verbose = TRUE
)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_p1".
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence. 3 is a good choice.
nx	the length of the training data.
params	values for the parameters for the specified distribution
alpha	the alpha values at which to test
plotflag	logical to turn the plotting on and off
verbose	logical to turn loop counting on and off

Details

The maximum likelihood quantiles (plotted in blue) do not give good reliability. They typically underestimate the tails (see panel (f)).

The cp predictive quantiles generally give reasonably good reliability, especially for sample sizes of ~100. The other predictions generally give poor reliability.

Value

A plot showing 9 different reliability checks, and a list containing various outputs, including the probabilities shown in the plot.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),

- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
set.seed(1)
# example 1
# -runs the default settings, which test reliability for the GEV distribution
reltest2(nrepeats=1)
```

`reltest2_cases` *Cases*

Description

Cases

Usage

```
reltest2_cases(model = "gev", nx = 50, params)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_pred1".
nx	length of training data
params	model parameters

Value

Two integers

`reltest2_makeep` *Cases*

Description

Cases

Usage

```
reltest2_makeep(model, pred1, tt0, params)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_pred1".
pred1	quantile predictions
tt0	value of predictor vector
params	model parameters

Value

Vector

`reltest2_plot` *Plotting routine for reltest2*

Description

Plots 9 diagnostics related to predictive probability matching.

Usage

```
reltest2_plot(
  model,
  ntrials,
  nrepeats,
  nx,
  params,
  nmethods,
  alpha,
  freqexceeded,
  case
)
```

Arguments

`model`	which distribution to test. Possibles values are "gev", "gpd", "gev_p1".
`ntrials`	the number of trials o run. 5000 typically gives good results.
`nrepeats`	the number of entire repeats of the test to run, to check for convergence
`nx`	the length of the training data.
`params`	values for the parameters for the specified distribution
`nmethods`	the number of methods being tested
`alpha`	the values of alpha being tested
`freqexceeded`	the exceedance counts
`case`	there are 3 cases (must be set to case=1 except for my testing)

Value

Plots the results of reliability testing

reltest2_predict *Make prediction from one model*

Description

Make prediction from one model

Usage

```
reltest2_predict(model = "gev", xx, tt, n0, pp, params, case, nmethods)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k1", "halfnorm", "norm", "lnorm", "gumbel", "frechet_k1", "weibull", "gev_k3", "logis", "lst_k3", "cauchy", "norm_p1", "lnorm_p1", "logis_p1", "lst_k3p1", "gumbel_p1", "norm_p12", "gev", "gpd", "gev_p1".
xx	training data
tt	predictor vector
n0	index for predictor vector
pp	probabilities to predict
params	model parameters
case	the case number: different models have different lists of methods
nmethods	the number of methods: different models have different numbers of methods

Value

Vector

reltest2_simulate *Random training data from one model*

Description

Random training data from one model

Usage

```
reltest2_simulate(model = "gev", nx = 50, tt, params)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_pred1".
nx	the length of the training data.
tt	the predictor
params	values for the parameters for the specified distribution

Value

Vector

---

reltest_makeep *Calculate EP from one model*

Description

Calculate EP from one model

Usage

```
reltest_makeep(model, pred1, tt0, tt10, tt20, tt30, params)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k3", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
pred1	quantile predictions
tt0	value of the predictor
tt10	value of predictor 1
tt20	value of predictor 2
tt30	value of predictor 3
params	the model parameters

Value

Vector

`reltest_makemaxep` *Calculate MaxEP from one model*

Description

Calculate MaxEP from one model

Usage

```
reltest_makemaxep(model, ml_max, tt0, tt10, tt20, tt30, params)
```

Arguments

`model`	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
`ml_max`	predicted max value
`tt0`	value of the predictor
`tt10`	value of predictor 1
`tt20`	value of predictor 2
`tt30`	value of predictor 3
`params`	the model parameters

Value

Vector

`reltest_predict` *Make prediction from one model*

Description

Make prediction from one model

Usage

```
reltest_predict(
  model,
  xx,
  tt,
  tt1,
  tt2,
  tt3,
  n0,
  n10,
```

```
  n20,
  n30,
  pp,
  params,
  dmgs = TRUE,
  debug = FALSE,
  aderivs = TRUE,
  unbiasedv = FALSE,
  pwm = FALSE,
  minxi = -10,
  maxxi = 10
)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "exp_p1k4", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
xx	training data
tt	predictor vector
tt1	predictor vector 1
tt2	predictor vector 2
tt3	predictor vector 3
n0	index for predictor vector
n10	index for predictor vector 1
n20	index for predictor vector 2
n30	index for predictor vector 2
pp	probabilites at which to make quantile predictions
params	model parameters
dmgs	flag for whether to run dmgs calculations or not
debug	flag for turning debug messages on
aderivs	a logical for whether to use analytic derivatives (instead of numerical)
unbiasedv	a logical for whether to use the unbiased variance instead of maxlik (for the normal)
pwm	a logical for whether to use PWM instead of maxlik (for the GEV)
minxi	minimum value for EVT shape parameter
maxxi	maximum value for EVT shape parameter

Value

Two vectors

`reltest_simulate` *Random training data from one model*

Description

Random training data from one model

Usage

```
reltest_simulate(
  model = "exp",
  nx = 20,
  tt,
  tt1,
  tt2,
  tt3,
  params,
  minxi = -10,
  maxxi = -10
)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k3", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
nx	the length of the training data to use.
tt	predictor vector
tt1	predictor vector 1
tt2	predictor vector 2
tt3	predictor vector 2
params	values for the parameters for the specified distribution
minxi	minimum value for EVT shape parameter
maxxi	maximum value for EVT shape parameter

Value

Vector

rgev_minmax — *rgev but with maxlik xi guaranteed within bounds*

Description

rgev but with maxlik xi guaranteed within bounds

Usage

```
rgev_minmax(nx, mu, sigma, xi, minxi = -0.45, maxxi = 0.45)
```

Arguments

`nx`	length of training data
`mu`	the location parameter of the distribution
`sigma`	the sigma parameter of the distribution
`xi`	the shape parameter of the distribution
`minxi`	minimum value of shape parameter xi
`maxxi`	maximum value of shape parameter xi

Value

Vector

rgev_p123_minmax — *rgev for gev_p123 but with maxlik xi within bounds*

Description

rgev for gev_p123 but with maxlik xi within bounds

Usage

```
rgev_p123_minmax(
  nx,
  mu,
  sigma,
  xi,
  t1,
  t2,
  t3,
  minxi = -0.45,
  maxxi = 0.45,
  centering = TRUE
)
```

Arguments

nx	length of training data
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
centering	indicates whether the routine should center the data or not

Value

Vector

---

rgev_p12_minmax   *rgev for gev_p12 but with maxlik xi within bounds*

Description

rgev for gev_p12 but with maxlik xi within bounds

Usage

```
rgev_p12_minmax(
  nx,
  mu,
  sigma,
  xi,
  t1,
  t2,
  minxi = -0.45,
  maxxi = 0.45,
  centering = TRUE
)
```

Arguments

nx	length of training data
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution

t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
centering	indicates whether the routine should center the data or not

Value

Vector

`rgev_p1_minmax` *rgev for gev_p1 but with maxlik xi within bounds*

Description

rgev for gev_p1 but with maxlik xi within bounds

Usage

```
rgev_p1_minmax(
  nx,
  mu,
  sigma,
  xi,
  tt,
  minxi = -0.45,
  maxxi = 0.45,
  centering = TRUE
)
```

Arguments

nx	length of training data
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
tt	a vector of predictors
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
centering	indicates whether the routine should center the data or not

Value

Vector

rgpd_k1_minmax *rgpd for gpd_k1 but with maxlik xi within bounds*

Description

rgpd for gpd_k1 but with maxlik xi within bounds

Usage

```
rgpd_k1_minmax(nx, kloc, sigma, xi, minxi = -0.45, maxxi = 0.45)
```

Arguments

nx	length of training data
kloc	the known location parameter
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

Vector

rhp_dmgs_cpmethod *Generates a comment about the method*

Description

Generates a comment about the method

Usage

```
rhp_dmgs_cpmethod()
```

Value

String

`rust_pumethod` *Generates a comment about the method*

Description

Generates a comment about the method

Usage

```
rust_pumethod()
```

Value

String

`testppm_plot` *Plotting routine for testppm*

Description

Plots 9 diagnostics related to predictive probability matching.

Usage

```
testppm_plot(
  model,
  ntrials,
  nrepeats,
  nx,
  params,
  nmethods,
  alpha,
  freqexceeded
)
```

Arguments

`model`	which distribution to test. Possibles values are
`ntrials`	the number of trials to run. 5000 typically gives good results.
`nrepeats`	the number of entire repeats of the test to run, to check for convergence
`nx`	the length of the training data.
`params`	values for the parameters for the specified distribution
`nmethods`	the number of methods being tested
`alpha`	the values of alpha being tested
`freqexceeded`	the exceedance counts

Value

Plots the results of reliability testing

---

unif_cp *Uniform Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qunif_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  debug = FALSE,
  aderivs = TRUE
)

runif_cp(n, x, mlcp = TRUE, debug = FALSE, aderivs = TRUE)

dunif_cp(x, y = x, debug = FALSE, aderivs = TRUE)

punif_cp(x, y = x, debug = FALSE, aderivs = TRUE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The uniform distribution has probability density function

$$f(x; min, max) = \frac{1}{max - min}$$

and zero otherwise, where $min \le x \le max$ is the random variable and min, max are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\lambda) \propto \frac{1}{max - min}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),

- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d25unif_example_data_v1
cat("length(x)=",length(x),"\n")
p=c(1:9)/10
q=qunif_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qunif_cp)",
main="unif: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

weibull_cp — *Weibull Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qweibull_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rweibull_cp(
```

```
  n,
  x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dweibull_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pweibull_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tweibull_cp(n, x, fd1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

`x`	a vector of training data values
`p`	a vector of probabilities at which to generate predictive quantiles
`fd1`	if `aderivs=FALSE`, the fractional delta used for numerical derivatives with respect to the first parameter
`fd2`	if `aderivs=FALSE`, the fractional delta used for numerical derivatives with respect to the second parameter
`means`	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
`waicscores`	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
`logscores`	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)

dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Weibull distribution has exceedance distribution function

$$S(x; k, \sigma) = \exp\left(-\left(\frac{x}{\sigma}\right)^k\right)$$

where $x \geq 0$ is the random variable and $k > 0, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(k, \sigma) \propto \frac{1}{k\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible

- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),

- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d52weibull_example_data_v1
p=c(1:9)/10
q=qweibull_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qweibull_cp)",
main="Weibull: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

weibull_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
weibull_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

weibull_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
weibull_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

weibull_f2f	*DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
weibull_f2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

weibull_f2fa	*The second derivative of the density*

Description

The second derivative of the density

Usage

```
weibull_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`weibull_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`weibull_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_ldd — *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
weibull_ldd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

weibull_ldda — *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
weibull_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
weibull_lddd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

weibull_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
weibull_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

`weibull_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
weibull_lmn(x, v1, fd1, v2, fd2, mm, nn)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate

Value

Scalar value

`weibull_lmnp` *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
weibull_lmnp(x, v1, fd1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

`weibull_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
weibull_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

weibull_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

weibull_loglik	*log-likelihood function*

Description

log-likelihood function

Usage

```
weibull_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

weibull_logscores	*Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
weibull_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

weibull_means — *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
weibull_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`ml_params`	parameters
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad_rhp`	derivative of the log RHP prior
`nx`	length of training data
`dim`	number of parameters

Value

Two scalars

weibull_mu1f — *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
weibull_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`weibull_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
weibull_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

`weibull_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
weibull_mu2f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`weibull_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
weibull_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

`weibull_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
weibull_p1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`weibull_p1fa` *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
weibull_p1fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

`weibull_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
weibull_p2f(y, v1, fd1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`weibull_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
weibull_p2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`weibull_p2_cp` *weibull Distribution with a Predictor on the Scale Parameter, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`

- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rweibull_p2_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  fd1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dweibull_p2_cp(
  x,
```

```
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  fd1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  fd1 = 0.01,
  d2 = 0.01,
  d3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tweibull_p2_cp(n, x, t, fd1 = 0.01, d2 = 0.01, d3 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that `length(t)=length(x)`
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
n0	an index for the predictor (specify either `t0` or `n0` but not both)
p	a vector of probabilities at which to generate predictive quantiles
fd1	if `aderivs=FALSE`, the fractional delta used for numerical derivatives with respect to the first parameter
d2	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the second parameter
d3	if `aderivs=FALSE`, the delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Weibull distribution with predictor on the scale parameter has exceedance distribution function

$$S(x; k, a, b) = \exp\left(-\left(\frac{x}{\sigma(a, b)}\right)^k\right)$$

where $x \geq 0$ is the random variable, $k > 0$ is the shape parameter and $\sigma = e^{a+bt}$ is the scale parameter, modelled as a function of parameters a, b and predictor t.

The calibrating prior is given by the right Haar prior, which is

$$\pi(k, \sigma) \propto \frac{1}{k}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d73weibull_p2_example_data_v1_x
tt=fitdistcp::d73weibull_p2_example_data_v1_t
p=c(1:9)/10
n0=10
```

```
q=qweibull_p2_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
sub="(from qweibull_p2_cp)",
main="Weibull w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`weibull_p2_f1f` *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
weibull_p2_f1f(y, t0, v1, fd1, v2, d2, v3, d3)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

weibull_p2_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
weibull_p2_f1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

weibull_p2_f2f *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
weibull_p2_f2f(y, t0, v1, fd1, v2, d2, v3, d3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

`weibull_p2_f2fa` *The second derivative of the density*

Description

The second derivative of the density

Usage

```
weibull_p2_f2fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`weibull_p2_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`weibull_p2_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`weibull_p2_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
weibull_p2_ldd(x, t, v1, fd1, v2, d2, v3, d3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

---

weibull_p2_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
weibull_p2_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

weibull_p2_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
weibull_p2_lddd(x, t, v1, fd1, v2, d2, v3, d3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

weibull_p2_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
weibull_p2_lddda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`weibull_p2_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
weibull_p2_lmn(x, t, v1, fd1, v2, d2, v3, d3, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

weibull_p2_lmnp — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
weibull_p2_lmnp(x, t, v1, fd1, v2, d2, v3, d3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

weibull_p2_logf — *Logf for RUST*

Description

Logf for RUST

Usage

```
weibull_p2_logf(params, x, t)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Scalar value.

`weibull_p2_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_logfdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`weibull_p2_loglik` *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
weibull_p2_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

`weibull_p2_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
weibull_p2_logscores(logscores, x, t, fd1, d2, d3, aderivs)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`weibull_p2_means` *weibull distribution: RHP mean*

Description

weibull distribution: RHP mean

Usage

```
weibull_p2_means(means, t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

---

weibull_p2_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
weibull_p2_mu1f(alpha, t0, v1, fd1, v2, d2, v3, d3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`weibull_p2_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
weibull_p2_mu1fa(alpha, t, v1, v2, v3)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`weibull_p2_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
weibull_p2_mu2f(alpha, t0, v1, fd1, v2, d2, v3, d3)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`d3`	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

`weibull_p2_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
weibull_p2_mu2fa(alpha, t, v1, v2, v3)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_p1f` *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
weibull_p2_p1f(y, t0, v1, fd1, v2, d2, v3, d3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

weibull_p2_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
weibull_p2_p1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`weibull_p2_p2f` *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
weibull_p2_p2f(y, t0, v1, fd1, v2, d2, v3, d3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either `t0` or `n0` but not both)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter

Value

3d array

`weibull_p2_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
weibull_p2_p2fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`weibull_p2_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`weibull_p2_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_pdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
weibull_p2_predictordata(predictordata, x, t, t0, params)
```

Arguments

`predictordata`	logical that indicates whether to calculate and return predictordata
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`params`	model parameters for calculating logf

Value

Two vectors

weibull_p2_waic *Waic*

Description

Waic

Usage

```
weibull_p2_waic(
  waicscores,
  x,
  t,
  v1hat,
  fd1,
  v2hat,
  d2,
  v3hat,
  d3,
  lddi,
  lddd,
  lambdad,
  aderivs = TRUE
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`weibull_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`weibull_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`weibull_waic` *Waic for RUST*

Description

Waic for RUST

Usage

```
weibull_waic(
  waicscores,
  x,
  v1hat,
  fd1,
  v2hat,
  fd2,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

Index