Package 'FitVerse'

November 13, 2026

Type Package

Title Parametric Distribution Fitting and Analysis

Version 1.0-3

Date 2026-11-13

Description Provides a unified, user-friendly interface for fitting parametric probability distributions to continuous univariate data. 'FitVerse' supports 52 distribution families spanning symmetric, right-skewed, heavy-tailed, bounded, and extreme-value shapes, and three estimation methods: Maximum Likelihood Estimation (MLE), Method of Moments (MOM), and L-Moments (L-MOM). Automatic best-fit selection is performed using AIC, BIC, and goodness-of-fit tests (Kolmogorov-Smirnov, Anderson-Darling, Cramer-von Mises (CvM)). Every fitted model produces a publication-quality diagnostic plot: a histogram overlaid with the fitted density curve and the estimated PDF formula annotated directly on the figure. An optional interactive version is produced via 'plotly'. Additional tools include bootstrap confidence intervals for parameter estimates and return levels, batch fitting across multiple columns for automated workflows and web-upload use cases, JSON serialisation for integration with 'Shiny' web applications, and automated HTML/PDF report generation. 'FitVerse' is designed to support data characterisation in survey sampling, hydrology, and actuarial workflows, where identifying the underlying distribution of a variable is a prerequisite for downstream modelling and inference.
L-moment estimation follows Hosking (1990) <doi:10.1111/j.2517-6161.1990.tb01775.x> and Hosking and Wallis (1997, ISBN:9780521430456).
Model selection via AIC follows Akaike (1974) <doi:10.1109/TAC.1974.1100705> and via BIC follows Schwarz (1978) <doi:10.1214/aos/1176344136>.
Bootstrap confidence intervals follow Efron and Hastie (2016, ISBN:9781107149892).

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.1.0)

Imports stats, graphics, grDevices, utils, tools, actuar (>= 3.1.0), mc2d (>= 0.1.18), evd (>= 2.3.3), sn (>= 2.1.0), fitdistrplus (>= 1.1.0), lmomco (>= 2.3.7), ggplot2 (>= 3.4.0), goftest (>=

1.2.3), gridExtra (>= 2.3)

Suggests shiny (>= 1.7.0), bslib (>= 0.5.0), DT, plotly (>= 4.10.0), testthat (>= 3.0.0), knitr (>= 1.40), rmarkdown (>= 2.20), kableExtra, jsonlite (>= 1.8.0), base64enc, MASS, GeneralizedHyperbolic, truncnorm

VignetteBuilder knitr

NeedsCompilation no

Config/roxygen2/version 8.0.0

Author Karuna G. Reddy [aut, cre] (ORCID: <https://orcid.org/0000-0002-9467-4668>), M. G. M. Khan [aut]

Maintainer Karuna G. Reddy <karuna.reddy@auckland.ac.nz>

Repository CRAN

Date/Publication 2026-11-13 07:30:08 UTC

Contents

FitVerse-package . . . 3
all_fits . . . 6
bootstrap_ci . . . 7
compare_dists . . . 9
compute_data_summary . . . 11
dfitverse . . . 13
dpqr_custom . . . 14
dweibull3p . . . 20
fitverse . . . 22
FitVerseApp . . . 26
fitverse_batch . . . 27
fitverse_censored . . . 29
FitVerse_datasets . . . 32
fitverse_distributions . . . 34
fitverse_grouped . . . 39
fitverse_s3methods . . . 41
fitverse_summary . . . 43
generate_report . . . 45
lmrd . . . 46
plot.fitverse . . . 48
ranking . . . 50
return_level . . . 51
save_plot . . . 53
to_json . . . 54

Index **56**

`FitVerse-package` *FitVerse: Parametric Distribution Fitting and Analysis*

Description

FitVerse provides a unified, user-friendly interface for fitting parametric probability distributions to continuous univariate data. The package supports **52 distribution families** spanning symmetric, right-skewed, heavy-tailed, bounded, and extreme-value shapes, and three estimation methods: **Maximum Likelihood Estimation (MLE)**, **Method of Moments (MOM)**, and **L-Moments (LMOM)**.

Automatic best-fit selection is performed using AIC and BIC, supplemented by four goodness-of-fit tests: Kolmogorov-Smirnov (KS), Anderson-Darling (AD), Cramer-von Mises (CvM), and Chi-Squared.

Every fitted model produces a publication-quality diagnostic plot: a histogram overlaid with the fitted density curve and the estimated PDF formula annotated directly on the figure. Additional diagnostic panels (Q-Q, P-P, CDF, Survival, Hazard, and Cumulative Hazard) are available via `plot.fitverse()`.

FitVerse is designed as the data-characterisation layer of the *OptiStrata* ecosystem, with seamless handoff to **stratifyR** for optimal stratification of survey populations.

Main fitting functions

`fitverse`	Fit distributions to individual observations
`fitverse_grouped`	Fit distributions to grouped/interval data
`fitverse_censored`	Fit distributions to censored data
`fitverse_batch`	Fit all columns of a data frame
`compare_dists`	Return the ranked comparison table only
`fitverse_distributions`	List all 52 supported distributions

Probability tools

`dfitverse`	Density of the best-fit distribution
`pfitverse`	CDF of the best-fit distribution
`qfitverse`	Quantile function of the best-fit distribution
`rfitverse`	Random samples from the best-fit distribution
`return_level`	T-year return levels from the fitted model
`lmrd`	L-moment ratio diagram
`ev_threshold`	Extreme-value threshold stability plots

Inference and reporting

`bootstrap_ci`	Bootstrap confidence intervals for parameters
`generate_report`	Produce a self-contained HTML or PDF report
`save_plot`	Save the diagnostic plot to a file
`to_json`	Serialise a fitverse object to JSON

Sample datasets

`FitVerse_datasets`	Bundled CSV files for individual and grouped data
`read_grouped_csv`	Read a grouped-data CSV into list format

S3 methods for `"fitverse"` objects

`print.fitverse`	Concise best-fit summary
`summary.fitverse`	Full diagnostic report to the console
`plot.fitverse`	Diagnostic plots (density, Q-Q, P-P, CDF, etc.)
`coef.fitverse`	Extract parameter estimates
`logLik.fitverse`	Extract log-likelihood
`AIC.fitverse`	Extract AIC
`BIC.fitverse`	Extract BIC
`as.list.fitverse`	Convert to a serialisation-ready list

Supported distributions (52 families)

Shape	Distribution	Key
Symmetric	Normal	`normal`
Symmetric	Logistic	`logistic`
Symmetric	Cauchy	`cauchy`
Symmetric	Laplace	`laplace`
Symmetric	Student-t (location-scale)	`student_t`
Symmetric	Truncated Normal	`truncnorm`
Asymmetric	Skew-Normal	`skewnormal`
Asymmetric	Generalised Normal	`gnorm`
Asymmetric	Asymmetric Laplace	`alaplace`
Asymmetric	Johnson SU	`johnsonsu`
Right-skewed	Gamma	`gamma`

Right-skewed	Gamma (3-parameter)	`gamma3`
Right-skewed	Log-Normal	`lognormal`
Right-skewed	Log-Normal (3-parameter)	`lognormal3`
Right-skewed	Exponential	`exponential`
Right-skewed	Exponentiated Exponential	`expexp`
Right-skewed	Weibull (2-parameter)	`weibull`
Right-skewed	Weibull (3-parameter)	`weibull3p`
Right-skewed	Exponentiated Weibull	`expweibull`
Right-skewed	Log-Logistic	`loglogistic`
Right-skewed	Inverse Gaussian	`invgaussian`
Right-skewed	Inverse Gamma	`invgamma`
Right-skewed	Inverse Weibull	`invweibull`
Right-skewed	Gompertz	`gompertz`
Right-skewed	Birnbaum-Saunders	`birnbaum_saunders`
Right-skewed	Rayleigh	`rayleigh`
Right-skewed	Half-Normal	`halfnormal`
Right-skewed	Lindley	`lindley`
Right-skewed	Nakagami	`nakagami`
Heavy-tailed	Pareto	`pareto`
Heavy-tailed	Burr XII	`burr`
Heavy-tailed	Lomax (Pareto Type II)	`lomax`
Heavy-tailed	Paralogistic	`paralogistic`
Heavy-tailed	Dagum	`dagum`
Heavy-tailed	GB2 (Generalised Beta II)	`gb2`
Heavy-tailed	Normal Inverse Gaussian	`nig`
Heavy-tailed	Generalised Hyperbolic	`ghyp`
Heavy-tailed	Generalised Gamma	`gengamma`
Extreme value	Generalised Extreme Value	`gev`
Extreme value	Gumbel	`gumbel`
Extreme value	Generalised Pareto	`gpd`
Extreme value	Generalised Logistic	`glo`
Extreme value	Pearson Type III	`pearson3`
Extreme value	Log-Pearson Type III	`lpearson3`
Extreme value	Kappa-4	`kap4`
Extreme value	Wakeby	`wakeby`
Bounded	Beta	`beta`
Bounded	Uniform	`uniform`
Bounded	Triangular	`triangular`
Bounded	Right Triangular	`rtriangle`
Bounded	PERT	`pert`
Bounded	Kumaraswamy	`kumaraswamy`

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz> (University of Auckland, New Zealand)
M. G. M. Khan <mgm.khan@usp.ac.fj> (University of the South Pacific, Fiji)

Maintainer: Karuna G. Reddy <karuna.reddy@auckland.ac.nz>

References

Hosking, J. R. M. (1990). L-moments: Analysis and estimation of distributions using linear combinations of order statistics. *Journal of the Royal Statistical Society, Series B*, **52**(1), 105–124. doi:10.1111/j.25176161.1990.tb01775.x

Delignette-Muller, M. L. and Dutang, C. (2015). fitdistrplus: An R package for fitting distributions. *Journal of Statistical Software*, **64**(4), 1–34. doi:10.18637/jss.v064.i04

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, **19**(6), 716–723. doi:10.1109/TAC.1974.1100705

See Also

fitverse, fitverse_distributions, fitverse_grouped, FitVerse_datasets, **fitdistrplus**, **actuar**, **lmomco**

Examples

```
## Quick start: fit three distributions to simulated Gamma data
set.seed(42)
x <- rgamma(300, shape = 2, rate = 0.5)

fit <- fitverse(x,
                dists   = c("gamma", "lognormal", "weibull"),
                method  = "MLE",
                xlab    = "Waiting time (s)",
                plot    = FALSE,
                verbose = FALSE)

print(fit)

## See all 52 supported distributions
fitverse_distributions()
```

---

all_fits *Extract All Individual Fit Results from a Fitted Object*

Description

Retrieves the list of per-distribution fit results stored inside a `fitverse` object. Each element corresponds to one model that was attempted during the fitting process.

Usage

```
all_fits(object, ...)
## S3 method for class 'fitverse'
all_fits(object, ...)
```

Arguments

`object`	A `fitverse` object.
`...`	Passed to methods.

Value

A named list of fit results, one per distribution attempted.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

`fitverse`, `ranking`, `summary.fitverse`

Examples

```
set.seed(1)
x    <- rgamma(200, shape = 2, rate = 0.5)
fit  <- fitverse(x)
fits <- all_fits(fit)
length(fits)
```

---

bootstrap_ci *Bootstrap Confidence Intervals for Distribution Parameters*

Description

Computes non-parametric percentile bootstrap confidence intervals for the parameters of the best-fit distribution in a `fitverse` object. Optionally also bootstraps confidence intervals for T-year return levels.

Usage

```
bootstrap_ci(fit,
             B              = 999L,
             conf           = 0.95,
             return_periods = NULL,
             progress       = NULL,
             seed           = NULL,
             verbose        = TRUE)

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

Arguments

`fit`	An object of class `"fitverse"` as returned by `fitverse()`. The bootstrap uses the best-fit distribution and estimation method stored in `fit$best_fit`.
`B`	Integer: number of bootstrap replicates. Default `999`. At least `199` is recommended; `1999` for publication-quality intervals.
`conf`	Numeric in $(0, 1)$: confidence level. Default `0.95` (95% CI).
`return_periods`	Numeric vector of return periods for which bootstrap CIs on return levels should be computed. `NULL` (default) skips this step. Example: `return_periods = c(10, 50, 100)`.
`progress`	Optional function of the form `function(p)` where `p` is a numeric value in $[0, 1]$. Called after every 10% of replicates. Useful for **shiny** progress bars. Default `NULL`.
`seed`	Integer or `NULL`: random seed for reproducibility. Default `NULL` (no seed set).
`verbose`	Logical: print a progress message? Default `TRUE`.
`x`	An object of class `"fitverse_bootstrap"`.
`...`	Currently unused.

Details

Algorithm: The non-parametric percentile bootstrap proceeds as follows:

1. Draw B bootstrap samples of size n with replacement from the observed data.
2. Re-fit the same distribution and estimation method to each sample, using the same internal engine as `fitverse()`.
3. Collect the $B \times k$ matrix of parameter estimates (where k is the number of parameters).
4. For each parameter, report the $\alpha/2$ and $1 - \alpha/2$ empirical percentiles of the bootstrap distribution as the lower and upper confidence bounds.

Replicates that fail to converge are excluded; a warning is issued if more than 10% fail.

Return-level CIs: When `return_periods` is supplied, the T-year return level is computed for each bootstrap replicate using `return_level()`, and the percentile CI is reported for each T.

Value

An S3 object of class `"fitverse_bootstrap"` with the following elements:

`params_ci` Data frame with one row per parameter and columns: `parameter`, `estimate`, `lower`, `upper`, `se_boot` (bootstrap standard error).

`return_level_ci` Data frame (or `NULL` if `return_periods = NULL`) with columns: `T`, `return_level`, `lower`, `upper`.

`boot_params` Numeric matrix of dimension $B_{eff} \times k$: all successful bootstrap parameter estimates. Useful for plotting bootstrap distributions.

`B` Integer: number of replicates requested.

`B_eff` Integer: number of successful replicates.

`conf` Numeric: confidence level.

`dist_key` Character: distribution key.

`method` Character: estimation method.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

References

Efron, B. and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. Chapman and Hall, New York.

Davison, A. C. and Hinkley, D. V. (1997). *Bootstrap Methods and Their Application*. Cambridge University Press.

See Also

`return_level` for point estimates without CIs.
`fitverse` for the initial fitting step.

Examples

```
set.seed(42)
x <- rgamma(200, shape = 2, rate = 0.5)
fit <- fitverse(x, dists = "gamma", plot = FALSE, verbose = FALSE)

boot <- bootstrap_ci(fit, B = 499, conf = 0.95, seed = 1, verbose = FALSE)
print(boot)

# Parameter CI data frame
boot$params_ci

# With return-level CIs
boot2 <- bootstrap_ci(fit, B = 499, return_periods = c(10, 50, 100),
                      seed = 1, verbose = FALSE)
boot2$return_level_ci
```

---

compare_dists *Compare Distribution Fits and Return a Ranking Table*

Description

Fits one or more parametric distributions to a numeric vector and returns an ordered ranking table without producing a plot or a full `fitverse` object. Use this function when you want a quick tabular comparison and do not need the fitted-object infrastructure.

Usage

```
compare_dists(x,
              dists     = "all",
              method    = "MLE",
              criterion = "AIC",
              xlab      = "x",
              verbose   = TRUE)
```

Arguments

x	A numeric vector of observations. Non-finite values are removed automatically before fitting.
dists	Character vector of distribution keys, or "all" to try every compatible distribution. See fitverse_distributions().
method	Character vector: one or more of "MLE", "MOM", "LMOM". All results are combined into a single table when multiple methods are specified.
criterion	"AIC" (default) or "BIC": the criterion by which rows are sorted.
xlab	Character: a label for the data variable, passed through to any downstream functions. Default "x".
verbose	Logical: print progress messages? Default TRUE.

Details

compare_dists() calls the same internal fitting engine as fitverse() but skips GoF test computation, data-summary computation, and plotting. It is therefore faster and better suited to exploratory screening of many distributions or methods.

The returned data frame can be used directly with kable or datatable for reporting.

Value

A data frame with one row per distribution-method combination, ordered by the chosen information criterion (ascending), with columns:

Rank Integer rank (1 = best).

Distribution Full distribution name.

Method Estimation method ("MLE", "MOM", or "LMOM").

N_params Number of free parameters.

LogLik Maximised log-likelihood.

AIC Akaike Information Criterion.

Delta_AIC AIC difference relative to the best model.

BIC Bayesian Information Criterion.

Delta_BIC BIC difference relative to the best model.

Converged Logical: did the optimiser converge?

dist_key Internal distribution key (character).

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

fitverse for a full fitted object plus diagnostic plot.

Examples

```
set.seed(1)
x <- rgamma(200, shape = 3, rate = 0.5)

tbl <- compare_dists(x,
                     dists   = c("gamma", "lognormal", "weibull", "exponential"),
                     method  = "MLE",
                     verbose = FALSE)
head(tbl)
```

---

compute_data_summary *Compute a Descriptive Statistics Summary*

Description

compute_data_summary() computes a comprehensive set of descriptive statistics for a numeric vector and returns them as a named list. It is called automatically by fitverse() and summary.fitverse(), but can also be used independently for exploratory data analysis.

Usage

```
compute_data_summary(x, xlab = "x", n_raw = NULL)
```

Arguments

x	A numeric vector, already cleaned of non-finite values (or not – the function removes them internally).
xlab	Character: variable name, stored in the returned list for labelling. Default "x".
n_raw	Integer or NULL: the original sample size before cleaning. If supplied, n_missing is computed as n_raw - n. If NULL, n_missing is set to NA.

Details

Skewness and kurtosis are computed using the bias-corrected (unbiased) estimators:

$$g_1 = \frac{n}{(n-1)(n-2)} \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s} \right)^3$$

$$g_2 = \frac{n(n+1)}{(n-1)(n-2)(n-3)} \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s} \right)^4 - \frac{3(n-1)^2}{(n-2)(n-3)}$$

These correspond to the `skewness` and `kurtosis` functions in the **moments** package (type 3 in `e1071`) and are consistent with the convention in *Applied Statistics* textbooks. The Normal distribution has skewness 0 and excess kurtosis 0.

Value

`compute_data_summary()` returns a named list with the following elements:

`n` Integer: number of finite (non-missing) observations used.

`n_missing` Integer or `NA`: number of non-finite values removed.

`mean` Arithmetic mean.

`sd` Standard deviation (denominator $n - 1$).

`cv` Coefficient of variation (`sd / mean`). `NA` when `mean == 0`.

`median` Sample median.

`iqr` Interquartile range ($Q_{0.75} - Q_{0.25}$).

`skewness` Sample skewness (Fisher's g_1).

`kurtosis` Excess kurtosis (Fisher's g_2).

`min` Sample minimum.

`max` Sample maximum.

`range` Range (`max - min`).

`q25` 25th percentile.

`q75` 75th percentile.

`xlab` Character: the variable label.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

`fitverse` (calls this automatically), `summary.fitverse`.

Examples

```
set.seed(3)
x <- rlnorm(500, meanlog = 1, sdlog = 0.5)

ds <- compute_data_summary(x, xlab = "Claim amount", n_raw = 510)
ds$skewness
ds$cv
```

`dfitverse` *Density, CDF, Quantile, and Random Sampling for a Fitted Distribution*

Description

Convenience functions that evaluate the probability density function (PDF), cumulative distribution function (CDF), quantile function (inverse CDF), and random-number generator for the best-fit distribution stored in a `fitverse` object.

These follow the standard R d/p/q/r naming convention and delegate to the underlying `d*`, `p*`, `q*`, and `r*` functions for the best-fit distribution.

Usage

```
dfitverse(x,   fit, log = FALSE)
pfitverse(q,   fit, lower.tail = TRUE, log.p = FALSE)
qfitverse(p,   fit, lower.tail = TRUE, log.p = FALSE)
rfitverse(n,   fit)
```

Arguments

`x`	Numeric vector of quantiles at which to evaluate the density.
`q`	Numeric vector of quantiles at which to evaluate the CDF.
`p`	Numeric vector of probabilities $p \in (0, 1)$.
`n`	Integer: number of random samples to generate.
`fit`	An object of class `"fitverse"` as returned by `fitverse()`.
`log`, `log.p`	Logical: if `TRUE`, return (or accept) log-scale values. Default `FALSE`.
`lower.tail`	Logical: if `TRUE` (default), return $P(X \leq q)$; if `FALSE`, return $P(X > q)$.

Details

The correct d/p/q/r functions are resolved via `.lookup_dpqr()`, which follows a two-step priority:

1. **Internal closed-form implementations** (highest priority): GEV, Gumbel, GPD, Pearson III, Log-Pearson III, and GLO all use FitVerse's own vectorised implementations registered in `.FITVERSE_DPQR_OVERRIDES`. This avoids any package-version signature issues with **evd** and provides full support for the new hydrological distributions that have no dedicated CRAN d/p/q/r package.
2. **Package namespace lookup**: For all other distributions, **stats** functions are found via `match.fun()`; Imports/Suggests packages (**actuar**, **sn**, **mc2d**) are resolved via `getExportedValue()`.

The estimated parameters are extracted from `fit$best_fit$params` and passed as named arguments to the underlying function. This is equivalent to calling, for example, `dgamma(x, shape = fit$best_fit$params["shape"], rate = fit$best_fit$params["rate"])`.

Value

`dfitverse()` Numeric vector of density values.

`pfitverse()` Numeric vector of probabilities.

`qfitverse()` Numeric vector of quantiles.

`rfitverse()` Numeric vector of random samples.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

`return_level` for T-year return levels.
`fitverse` for the fitting step.

Examples

```
set.seed(42)
x <- rgamma(300, shape = 2, rate = 0.5)
fit <- fitverse(x, dists = "gamma", plot = FALSE, verbose = FALSE)

# Density at x = 2
dfitverse(2, fit)

# Probability that X <= 5
pfitverse(5, fit)

# 90th percentile
qfitverse(0.90, fit)

# Generate 10 random samples
rfitverse(10, fit)
```

---

dpqr_custom *Custom d/p/q/r Functions for Additional Distributions*

Description

Density, CDF, quantile, and random-generation functions for distribution families that are either unavailable on CRAN, parameterised differently from existing packages, or implemented from scratch for performance and consistency within **FitVerse**.

The following families are covered:

Student-t (location-scale) Three-parameter Student-t with location μ, scale $\sigma > 0$, and degrees of freedom $\nu > 0$: `dstudent_t`, `pstudent_t`, `qstudent_t`, `rstudent_t`.

Laplace Symmetric double-exponential with location μ and scale $b > 0$: `dlaplace`, `plaplace`, `qlaplace`, `rlaplace`.

Inverse Gaussian Two-parameter (mean $\mu > 0$, shape $\lambda > 0$): `dinvgaussian`, `pinvgaussian`, `qinvgaussian`, `rinvgaussian`.

Lomax (Pareto Type II) Shape $\alpha > 0$ and scale $\lambda > 0$: `dlomax`, `plomax`, `qlomax`, `rlomax`.

Generalised Gamma Three-parameter generalisation of the Gamma (shape $p > 0$, scale $d > 0$, power $b > 0$): `dgengamma`, `pgengamma`, `qgengamma`, `rgengamma`.

Gompertz Parameters $\alpha > 0$ and $\beta > 0$: `dgompertz`, `pgompertz`, `qgompertz`, `rgompertz`.

Birnbaum-Saunders (Fatigue-Life) Shape $\alpha > 0$ and scale $\beta > 0$: `dbirnbaum_saunders`, `pbirnbaum_saunders`, `qbirnbaum_saunders`, `rbirnbaum_saunders`.

Rayleigh Scale $\sigma > 0$: `drayleigh`, `prayleigh`, `qrayleigh`, `rrayleigh`.

Half-Normal Scale $\sigma > 0$: `dhalfnormal`, `phalfnormal`, `qhalfnormal`, `rhalfnormal`.

Exponentiated Weibull Shape parameters `shape1 > 0` and `shape2 > 0`, scale `scale > 0`: `dexpweibull`, `pexpweibull`, `qexpweibull`, `rexpweibull`.

Three-Parameter Gamma Gamma with location threshold γ (shape $\alpha > 0$, scale $\beta > 0$, threshold γ): `dgamma3`, `pgamma3`, `qgamma3`, `rgamma3`.

Three-Parameter Log-Normal Log-Normal with threshold γ (meanlog μ, sdlog $\sigma > 0$, threshold γ): `dlnorm3`, `plnorm3`, `qlnorm3`, `rlnorm3`.

Right Triangular Special triangular with `min` and `max`, where the mode equals `max`: `drtriangle`, `prtriangle`, `qrtriangle`, `rrtriangle`.

Kumaraswamy Bounded distribution with shape parameters `shape1 > 0` and `shape2 > 0` on $(0, 1)$: `dkumaraswamy`, `pkumaraswamy`, `qkumaraswamy`, `rkumaraswamy`.

Generalised Beta of the Second Kind (GB2) Four-parameter family with `shape1 > 0`, `scale > 0`, `shape2 > 0`, `shape3 > 0`: `dgb2`, `pgb2`, `qgb2`, `rgb2`.

Kappa Four-Parameter Generalised extreme-value family with location `xi`, scale `alpha > 0`, and shape parameters `kappa` and `h`: `dkap4`, `pkap4`, `qkap4`, `rkap4`.

Lindley One-parameter distribution with `theta > 0`: `dlindley`, `plindley`, `qlindley`, `rlindley`.

Generalised Normal Three-parameter distribution with location `mu`, scale `sigma > 0`, and shape `beta > 0`: `dgnorm`, `pgnorm`, `qgnorm`, `rgnorm`.

Dagum Three-parameter Burr Type III with `shape1 > 0`, `scale > 0`, `shape2 > 0`: `ddagum`, `pdagum`, `qdagum`, `rdagum`.

Johnson SU Four-parameter unbounded system with `gamma`, `delta > 0`, `xi`, `lambda > 0`: `djohnsonsu`, `pjohnsonsu`, `qjohnsonsu`, `rjohnsonsu`.

Asymmetric Laplace Three-parameter asymmetric Laplace with location `mu`, scale `sigma > 0`, and asymmetry `kappa > 0`: `dalaplace`, `palaplace`, `qalaplace`, `ralaplace`.

Exponentiated Exponential Two-parameter family with shape `alpha > 0` and rate `lambda > 0`: `dexpexp`, `pexpexp`, `qexpexp`, `rexpexp`.

Normal-Inverse Gaussian (NIG) Four-parameter family with location `mu`, spread `delta > 0`, steepness `alpha > 0`, and asymmetry `beta`: `dnig`, `pnig`, `qnig`, `rnig`.

Wakeby Five-parameter quantile-defined distribution with location `xi`, scale parameters `alpha` and `beta`, and shape parameters `gamma` and `delta`: `dwakeby`, `pwakeby`, `qwakeby`, `rwakeby`.

Generalised Hyperbolic (GHyp) Five-parameter family with `lambda`, `alpha > 0`, `beta`, `delta > 0`, `mu`: `dghyp`, `pghyp`, `qghyp`, `rghyp`.

Truncated Normal Normal distribution truncated to $[a, b]$ with `mean` and `sd > 0`: `dtruncnorm`, `ptruncnorm`, `qtruncnorm`, `rtruncnorm`.

Nakagami Two-parameter fading distribution with shape `m >= 0.5` and spread `Omega > 0`: `dnakagami`, `pnakagami`, `qnakagami`, `rnakagami`.

Usage

```
dstudent_t(x,   nu, mu, sigma, log = FALSE)
pstudent_t(q,   nu, mu, sigma)
qstudent_t(p,   nu, mu, sigma)
rstudent_t(n,   nu, mu, sigma)

dlaplace(x,   mu, b, log = FALSE)
plaplace(q,   mu, b)
qlaplace(p,   mu, b)
rlaplace(n,   mu, b)

dinvgaussian(x,   mu, lambda, log = FALSE)
pinvgaussian(q,   mu, lambda)
qinvgaussian(p,   mu, lambda)
rinvgaussian(n,   mu, lambda)

dlomax(x,   alpha, lambda, log = FALSE)
plomax(q,   alpha, lambda)
qlomax(p,   alpha, lambda)
rlomax(n,   alpha, lambda)

dgengamma(x,   p, d, b, log = FALSE)
pgengamma(q,   p, d, b)
qgengamma(prob, p, d, b)
rgengamma(n,   p, d, b)

dgompertz(x,   alpha, beta, log = FALSE)
pgompertz(q,   alpha, beta)
qgompertz(p,   alpha, beta)
rgompertz(n,   alpha, beta)

dbirnbaum_saunders(x,   alpha, beta, log = FALSE)
pbirnbaum_saunders(q,   alpha, beta)
qbirnbaum_saunders(p,   alpha, beta)
rbirnbaum_saunders(n,   alpha, beta)

drayleigh(x,   sigma, log = FALSE)
prayleigh(q,   sigma, lower.tail = TRUE, log.p = FALSE)
qrayleigh(p,   sigma, lower.tail = TRUE, log.p = FALSE)
rrayleigh(n,   sigma)
```

```
dhalfnormal(x,   sigma, log = FALSE)
phalfnormal(q,   sigma, lower.tail = TRUE, log.p = FALSE)
qhalfnormal(p,   sigma, lower.tail = TRUE, log.p = FALSE)
rhalfnormal(n,   sigma)

dexpweibull(x,   shape1, scale, shape2, log = FALSE)
pexpweibull(q,   shape1, scale, shape2, lower.tail = TRUE, log.p = FALSE)
qexpweibull(p,   shape1, scale, shape2, lower.tail = TRUE, log.p = FALSE)
rexpweibull(n,   shape1, scale, shape2)

dgamma3(x,   shape, scale, thres, log = FALSE)
pgamma3(q,   shape, scale, thres, lower.tail = TRUE, log.p = FALSE)
qgamma3(p,   shape, scale, thres, lower.tail = TRUE, log.p = FALSE)
rgamma3(n,   shape, scale, thres)

dlnorm3(x,   meanlog, sdlog, thres, log = FALSE)
plnorm3(q,   meanlog, sdlog, thres, lower.tail = TRUE, log.p = FALSE)
qlnorm3(p,   meanlog, sdlog, thres, lower.tail = TRUE, log.p = FALSE)
rlnorm3(n,   meanlog, sdlog, thres)

drtriangle(x,   min, max, log = FALSE)
prtriangle(q,   min, max)
qrtriangle(p,   min, max)
rrtriangle(n,   min, max)

dkumaraswamy(x,   shape1, shape2, log = FALSE)
pkumaraswamy(q,   shape1, shape2)
qkumaraswamy(p,   shape1, shape2)
rkumaraswamy(n,   shape1, shape2)

dgb2(x, shape1, scale, shape2, shape3, log = FALSE)
pgb2(q, shape1, scale, shape2, shape3, lower.tail = TRUE, log.p = FALSE)
qgb2(p, shape1, scale, shape2, shape3, lower.tail = TRUE, log.p = FALSE)
rgb2(n, shape1, scale, shape2, shape3)

dkap4(x, xi, alpha, kappa, h, log = FALSE)
pkap4(q, xi, alpha, kappa, h, lower.tail = TRUE, log.p = FALSE)
qkap4(p, xi, alpha, kappa, h, lower.tail = TRUE, log.p = FALSE)
rkap4(n, xi, alpha, kappa, h)

dlindley(x, theta, log = FALSE)
plindley(q, theta, lower.tail = TRUE, log.p = FALSE)
qlindley(p, theta, lower.tail = TRUE, log.p = FALSE)
rlindley(n, theta)

dgnorm(x, mu, sigma, beta, log = FALSE)
pgnorm(q, mu, sigma, beta, lower.tail = TRUE, log.p = FALSE)
```

```
qgnorm(p, mu, sigma, beta, lower.tail = TRUE, log.p = FALSE)
rgnorm(n, mu, sigma, beta)

ddagum(x, shape1, scale, shape2, log = FALSE)
pdagum(q, shape1, scale, shape2, lower.tail = TRUE, log.p = FALSE)
qdagum(p, shape1, scale, shape2, lower.tail = TRUE, log.p = FALSE)
rdagum(n, shape1, scale, shape2)

djohnsonsu(x, gamma, delta, xi, lambda, log = FALSE)
pjohnsonsu(q, gamma, delta, xi, lambda, lower.tail = TRUE, log.p = FALSE)
qjohnsonsu(p, gamma, delta, xi, lambda, lower.tail = TRUE, log.p = FALSE)
rjohnsonsu(n, gamma, delta, xi, lambda)

dalaplace(x, mu, sigma, kappa, log = FALSE)
palaplace(q, mu, sigma, kappa, lower.tail = TRUE, log.p = FALSE)
qalaplace(p, mu, sigma, kappa, lower.tail = TRUE, log.p = FALSE)
ralaplace(n, mu, sigma, kappa)

dexpexp(x, alpha, lambda, log = FALSE)
pexpexp(q, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
qexpexp(p, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
rexpexp(n, alpha, lambda)

dnig(x, mu, delta, alpha, beta, log = FALSE)
pnig(q, mu, delta, alpha, beta, lower.tail = TRUE, log.p = FALSE)
qnig(p, mu, delta, alpha, beta, lower.tail = TRUE, log.p = FALSE)
rnig(n, mu, delta, alpha, beta)

dwakeby(x, xi, alpha, beta, gamma, delta, log = FALSE)
pwakeby(q, xi, alpha, beta, gamma, delta, lower.tail = TRUE, log.p = FALSE)
qwakeby(p, xi, alpha, beta, gamma, delta, lower.tail = TRUE, log.p = FALSE)
rwakeby(n, xi, alpha, beta, gamma, delta)

dghyp(x, lambda, alpha, beta, delta, mu, log = FALSE)
pghyp(q, lambda, alpha, beta, delta, mu, lower.tail = TRUE, log.p = FALSE)
qghyp(p, lambda, alpha, beta, delta, mu, lower.tail = TRUE, log.p = FALSE)
rghyp(n, lambda, alpha, beta, delta, mu)

dtruncnorm(x, mean, sd, a, b, log = FALSE)
ptruncnorm(q, mean, sd, a, b, lower.tail = TRUE, log.p = FALSE)
qtruncnorm(p, mean, sd, a, b, lower.tail = TRUE, log.p = FALSE)
rtruncnorm(n, mean, sd, a, b)

dnakagami(x, m, Omega, log = FALSE)
pnakagami(q, m, Omega, lower.tail = TRUE, log.p = FALSE)
qnakagami(p, m, Omega, lower.tail = TRUE, log.p = FALSE)
rnakagami(n, m, Omega)
```

Arguments

x, q	Numeric vector of quantiles.
p	Numeric vector of probabilities in $(0, 1)$ for quantile functions; also the power shape parameter in dgengamma, pgengamma, qgengamma, and rgengamma.
prob	Numeric vector of probabilities in $(0, 1)$ for qgengamma (first argument, to avoid ambiguity with the p shape parameter).
n	Integer: number of observations to generate.
nu	Degrees of freedom for the Student-t ($\nu > 0$).
mu, sigma	Location and scale parameters.
b	Scale parameter for Laplace; also the shape parameter of the Generalised Gamma (used as the third shape in dgengamma etc.); also the upper truncation bound for dtruncnorm and related functions.
d	Depth shape parameter of the Generalised Gamma.
lambda	Inverse Gaussian shape parameter; Lomax scale parameter; Johnson SU scale parameter ($\lambda > 0$).
scale, shape	Scale and shape parameters.
shape1, shape2	Shape parameters.
alpha, beta	Gompertz and Birnbaum-Saunders parameters; Lomax uses alpha.
meanlog	Mean of the log-transformed variable (log-scale location) for dlnorm3/plnorm3/qlnorm3/rlnorm3.
sdlog	Standard deviation of the log-transformed variable (log-scale spread, > 0) for dlnorm3/plnorm3/qlnorm3/rlnorm3.
thres	Lower threshold (location shift) for 3-parameter distributions.
min, max	Support bounds for the Right Triangular.
log, log.p	Logical: if TRUE, probabilities are on the log scale.
lower.tail	Logical: if TRUE (default), $P(X \leq q)$; otherwise $P(X > q)$.
shape3	Third shape parameter for dgb2 and related functions.
xi	Location parameter for Kappa, Wakeby, and Johnson SU distributions.
h	Lower shape parameter for the Kappa four-parameter distribution.
kappa	Upper shape parameter for the Kappa four-parameter distribution; also the asymmetry parameter for the Asymmetric Laplace.
theta	Rate parameter for the Lindley distribution ($\theta > 0$).
gamma	Shape parameter for the Johnson SU distribution; also a scale parameter in the Wakeby distribution.
delta	Shape parameter for the Johnson SU and Wakeby distributions; also the spread parameter for the NIG and GHyp distributions ($\delta > 0$).
m	Shape parameter for the Nakagami distribution ($m \geq 0.5$).
Omega	Spread parameter for the Nakagami distribution ($\Omega > 0$).
mean, sd	Mean and standard deviation for the Truncated Normal.
a	Lower truncation bound for dtruncnorm and related functions.

Value

A numeric vector of densities, probabilities, quantiles, or random samples, consistent with the standard R d/p/q/r convention.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

fitverse to fit these distributions to data; fitverse_distributions for the full distribution catalogue; dfitverse for evaluating the fitted best-fit distribution.

Examples

```
## Rayleigh
curve(drayleigh(x, sigma = 2), from = 0, to = 10,
      main = "Rayleigh(sigma=2)", ylab = "Density")

## Kumaraswamy (bounded to (0,1))
curve(dkumaraswamy(x, shape1 = 2, shape2 = 5), from = 0, to = 1,
      main = "Kumaraswamy(2, 5)", ylab = "Density")

## Birnbaum-Saunders round-trip
p <- c(0.1, 0.5, 0.9)
q <- qbirnbaum_saunders(p, alpha = 0.5, beta = 100)
pbirnbaum_saunders(q, alpha = 0.5, beta = 100)   # recovers p

## Inverse Gaussian random samples
set.seed(1)
x <- rinvgaussian(200, mu = 5, lambda = 10)
hist(x, breaks = 20, main = "Inverse Gaussian(5, 10)")
```

---

dweibull3p *Three-Parameter Weibull Distribution*

Description

Density, cumulative distribution function, quantile function, and random generation for the three-parameter Weibull distribution with shape $\beta > 0$, scale $\alpha > 0$, and threshold (location) γ. The support is $x > \gamma$.

Usage

```
dweibull3p(x,     shape, scale, thres, log = FALSE)
pweibull3p(q,     shape, scale, thres, lower.tail = TRUE, log.p = FALSE)
qweibull3p(p,     shape, scale, thres, lower.tail = TRUE, log.p = FALSE)
rweibull3p(n,     shape, scale, thres)
```

Arguments

x, q	Numeric vector of quantiles.
p	Numeric vector of probabilities in $(0, 1)$.
n	Integer: number of observations to generate.
shape	Positive shape parameter β.
scale	Positive scale parameter α.
thres	Real-valued threshold (location) parameter γ. Values of x at or below thres have zero density.
log, log.p	Logical: if TRUE, return log-probabilities.
lower.tail	Logical: if TRUE (default), return $P(X \leq q)$; otherwise $P(X > q)$.

Details

The three-parameter Weibull is a location-shifted version of the standard two-parameter Weibull. Let $Y = X - \gamma$; then $Y \sim \text{Weibull}(\beta, \alpha)$. The PDF for $x > \gamma$ is

$$f(x) = \frac{\beta}{\alpha}\left(\frac{x-\gamma}{\alpha}\right)^{\beta-1} \exp\left[-\left(\frac{x-\gamma}{\alpha}\right)^{\beta}\right].$$

Internally these functions shift x by thres and delegate to dweibull, pweibull, qweibull, and rweibull.

Value

dweibull3p Numeric vector of density values.

pweibull3p Numeric vector of probabilities.

qweibull3p Numeric vector of quantiles.

rweibull3p Numeric vector of random samples.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

dweibull for the two-parameter Weibull; fitverse with dists = "weibull3" to fit this distribution by MLE.

Examples

```
## PDF and CDF
curve(dweibull3p(x, shape = 2, scale = 10, thres = 5),
      from = 5, to = 35, ylab = "Density",
      main = "Weibull(2, 10) shifted by 5")
```

```
## Quantile / CDF round-trip
p  <- c(0.1, 0.5, 0.9)
q  <- qweibull3p(p, shape = 2, scale = 10, thres = 5)
pweibull3p(q, shape = 2, scale = 10, thres = 5)   # should recover p

## Random samples
set.seed(1)
x <- rweibull3p(200, shape = 1.8, scale = 50, thres = 10)
hist(x, breaks = 20, main = "Weibull(1.8, 50) with threshold 10")
```

`fitverse` *Fit Probability Distributions to Univariate Continuous Data*

Description

The primary function of the **FitVerse** package. Given a numeric data vector, `fitverse()` fits one or more parametric probability distributions, ranks them by an information criterion, selects the best-fitting model, computes goodness-of-fit statistics, and optionally produces a publication-quality diagnostic plot.

Three estimation methods are supported: Maximum Likelihood Estimation (`"MLE"`), Method of Moments (`"MOM"`), and L-Moments (`"LMOM"`). When multiple methods are requested, all results are combined into a single ranking table and the global best model is returned.

Tip: For a point-and-click experience, launch the built-in interactive Shiny application with `FitVerseApp()`.

Usage

```
fitverse(x,
         dists       = "all",
         method      = "MLE",
         criterion   = "AIC",
         xlab        = "x",
         plot        = TRUE,
         interactive = FALSE,
         top_n       = 3L,
         progress    = NULL,
         verbose     = TRUE,
         ...)
```

Arguments

`x`	A numeric vector of observations. At least 4 finite values are required. Non-finite values (`NA`, `NaN`, `Inf`, `-Inf`) are removed automatically before fitting; a message is printed if any are found.
`dists`	Character vector of distribution keys to try, or `"all"` (default) to attempt every distribution whose support is compatible with the data. Use `fitverse_distributions()` to see all valid keys. Example: `dists = c("gamma", "lognormal", "weibull")`.
`method`	Character vector specifying which estimation method(s) to use. One or more of:

"MLE" Maximum Likelihood Estimation via numerical optimisation (**fitdistrplus** or custom stats::optim). Available for all 52 distributions.

"MOM" Method of Moments. Uses closed-form expressions where available; numerical solution otherwise. Available for most of the 52 distributions.

"LMOM" L-Moment estimation via **lmomco**. Available for distributions with L-moment parameter estimators.

When multiple methods are supplied (e.g. method = c("MLE", "MOM")), all fits are combined and ranked together. Default: "MLE".

criterion — Character: the information criterion used for ranking and best-model selection. Either "AIC" (default) or "BIC".

AIC is recommended for exploratory analysis and prediction; BIC imposes a stronger penalty for model complexity and is preferred when the goal is identifying the true generating distribution.

xlab — Character: a label for the data variable used in plot axis titles and the auto-generated report. Default "x". Example: xlab = "Annual maximum flood (m^3/s)".

plot — Logical: should a diagnostic plot be produced and printed? Default TRUE. Set to FALSE for batch workflows or when embedding in scripts. The plot can always be regenerated later with plot.fitverse().

interactive — Logical: if TRUE and the **plotly** package is installed, produces an interactive HTML figure instead of a static ggplot2 image. Default FALSE.

top_n — Integer: number of top-ranked distributions to overlay on the density panel of the diagnostic plot. The best-fit model is always drawn as a solid line; the remaining top_n - 1 are drawn as dashed lines. Default 3. Range: 1 to nrow(ranking).

progress — Optional function of the form function(p) where p is a numeric value in $[0, 1]$ indicating the proportion of work completed. Called at several points during fitting. Useful for updating a **shiny** progress bar. Default NULL (no callback). Example: progress = function(p) shiny::setProgress(p).

verbose — Logical: print progress messages to the console? Default TRUE. Set to FALSE in batch workflows or when running inside **shiny**.

... — Additional arguments passed to plot.fitverse().

Details

Workflow: fitverse() executes the following steps:

1. **Validate and clean** – removes non-finite values; checks that at least 4 observations remain.
2. **Filter distributions** – distributions whose support is incompatible with the data (e.g. positive-only Gamma when data contains negative values) are silently excluded.
3. **Fit** – calls fit_mle_all(), fit_mom_all(), and/or fit_lmom_all() according to method.
4. **Goodness-of-fit** – attaches Kolmogorov-Smirnov (KS), Anderson-Darling (AD), and Cramer-von Mises (CvM) test results to each fit result.
5. **Rank** – builds a table ordered by criterion (ascending AIC or BIC).
6. **Select** – chooses the model with the lowest AIC (or BIC).
7. **Summarise** – computes compute_data_summary() and stores it in the returned object.
8. **Plot** (optional) – calls plot.fitverse().

Estimation methods:

MLE Maximises the log-likelihood $\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log f(x_i; \boldsymbol{\theta})$. Implemented via `fitdistrplus::fitdist()` with bounded L-BFGS-B optimisation where parameter constraints exist.

MOM Equates the first k sample moments to their theoretical counterparts and solves for the k parameters. Closed-form solutions are used where available (Normal, Gamma, Log-Normal, Exponential, Beta, Logistic, Uniform, Gumbel, Pareto, Inverse Gamma); numerical solution via `stats::optimise()` for Weibull and Log-Logistic. **fitdistrplus** MME is used for standard **stats** distributions.

LMOM Equates sample L-moments (linear combinations of order statistics) to their theoretical counterparts. L-moments are more robust to outliers than conventional moments and are widely used in hydrology. Implemented via **lmomco**; available for Normal, Gamma, Exponential, Log-Normal, Weibull, GEV, GLO, Gumbel, Logistic, and Log-Logistic. Reference: Hosking (1990).

Model selection: The best-fit model is the one with the lowest AIC (or BIC) across all fitted distributions and methods. Delta AIC values in the ranking table quantify evidence against each model relative to the best: models with Delta AIC < 2 have "substantial support", 4–7 "considerably less", and > 10 "essentially none" (Burnham and Anderson, 2002).

Goodness-of-fit tests: Three non-parametric tests are applied to the best-fit model:

Kolmogorov-Smirnov (KS) Measures the maximum absolute difference between the empirical and fitted CDFs. Conservative; not recommended as the primary criterion.

Anderson-Darling (AD) Gives extra weight to the distribution tails. More powerful than KS for detecting tail departures. Implemented via **goftest**.

Cramer-von Mises (CvM) Integrates the squared difference between empirical and fitted CDFs. Implemented via **goftest**.

A large p-value (> 0.05) indicates insufficient evidence to reject the fitted distribution.

Value

An S3 object of class `"fitverse"`. It is a named list with the following elements:

`best_fit` Named list describing the best-fitting model, with elements:

- `dist_key` Character: distribution key (e.g. `"gamma"`).
- `method` Character: estimation method (`"MLE"`, `"MOM"`, or `"LMOM"`).
- `params` Named numeric vector of parameter estimates.
- `loglik` Numeric: maximised log-likelihood.
- `aic` Numeric: Akaike Information Criterion.
- `bic` Numeric: Bayesian Information Criterion.
- `n_params` Integer: number of estimated parameters.
- `n` Integer: sample size used.
- `converged` Logical: did the optimiser converge?
- `gof` Data frame of goodness-of-fit test results (Test, Statistic, P_value, Stars), or `NULL` if unavailable.

`all_fits` Named list of all fit-result lists, one per distribution-method combination. Keys are of the form `"distkey_METHOD"` (e.g. `"gamma_MLE"`).

`ranking` Data frame with columns: Rank, Distribution, Method, N_params, LogLik, AIC, Delta_AIC, BIC, Delta_BIC, KS_p, AD_p, Converged, dist_key. Ordered by the chosen `criterion` (ascending).

`data` Numeric vector: the cleaned input data.

`n` Integer: number of finite observations used.

`xlab` Character: the variable label.

`data_summary` Named list from `compute_data_summary()`: n, n_missing, mean, sd, cv, median, iqr, skewness, kurtosis, min, max, range, q25, q75.

`criterion` Character: `"AIC"` or `"BIC"`.

`method` Character vector: method(s) used.

`call` The matched function call.

The object is returned invisibly. Use `print()`, `summary()`, `plot()`, `coef()`, `logLik()`, `AIC()`, and `BIC()` for standard S3 method access.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

References

Burnham, K. P. and Anderson, D. R. (2002). *Model Selection and Multimodel Inference* (2nd ed.). Springer.

Delignette-Muller, M. L. and Dutang, C. (2015). fitdistrplus: An R package for fitting distributions. *Journal of Statistical Software*, **64**(4), 1–34. doi:10.18637/jss.v064.i04

Hosking, J. R. M. (1990). L-moments: Analysis and estimation of distributions using linear combinations of order statistics. *Journal of the Royal Statistical Society, Series B*, **52**(1), 105–124.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, **19**(6), 716–723.

See Also

`compare_dists` for a ranking table only (no plot).
`fitverse_distributions` to list all supported distributions.
`plot.fitverse`, `summary.fitverse`, `bootstrap_ci`, `return_level`, `generate_report`.

Examples

```
## ------------------------------------------------------------
## Example 1: Basic MLE fit -- three distributions
## ------------------------------------------------------------
set.seed(42)
x <- rgamma(300, shape = 2, rate = 0.5)

fit <- fitverse(x,
                dists   = c("gamma", "lognormal", "weibull"),
```

```
                method  = "MLE",
                xlab    = "Waiting time (s)",
                plot    = FALSE,
                verbose = FALSE)

print(fit)
coef(fit)        # parameter estimates
AIC(fit)         # AIC of the best-fit model
logLik(fit)      # log-likelihood

## ------------------------------------------------------------
## Example 2: All three estimation methods, ranked by BIC
## ------------------------------------------------------------

fit2 <- fitverse(x,
                 method    = c("MLE", "MOM", "LMOM"),
                 criterion = "BIC",
                 plot      = FALSE,
                 verbose   = FALSE)
summary(fit2)


## ------------------------------------------------------------
## Example 3: Automatic best-fit selection across all distributions
## ------------------------------------------------------------

set.seed(7)
x3 <- rlnorm(500, meanlog = 1.5, sdlog = 0.6)

fit3 <- fitverse(x3,
                 xlab    = "Insurance claim size",
                 method  = "MLE",
                 plot    = FALSE,
                 verbose = FALSE)
fit3$ranking[1:5, c("Distribution","Method","AIC","Delta_AIC","AD_p")]


## ------------------------------------------------------------
## Example 4: Accessing the data summary
## ------------------------------------------------------------

fit4 <- fitverse(x, plot = FALSE, verbose = FALSE)
fit4$data_summary$mean
fit4$data_summary$skewness
```

FitVerseApp *Launch the FitVerse Interactive Shiny Application*

Description

Opens the FitVerse web interface in the default browser. The app provides a point-and-click interface for uploading data, fitting distributions, inspecting diagnostic plots, and downloading results — without writing any R code.

Usage

```
FitVerseApp(...)
```

Arguments

`...`	Additional arguments passed to `runApp`, for example `port` or `launch.browser`.

Details

Requires the **shiny** package (listed in `Suggests`). An informative error is raised if it is not installed.

The app files are stored in `inst/app/` and located at runtime via `system.file("app", package = "FitVerse")`.

Value

Called for its side effect (launches a Shiny app); returns `NULL` invisibly.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

`fitverse` for the programmatic fitting interface.

Examples

```
if (interactive()) {
  FitVerseApp()
}
```

`fitverse_batch` *Fit Distributions to All Columns of a Data Frame*

Description

Applies `fitverse()` to every numeric column of a data frame (or matrix, or named list), collects the results, and returns a tidy summary table alongside the full list of fitted objects.

Use `fitverse_batch()` for automated data characterisation workflows, survey-population profiling, or as a pre-processing step before optimal stratification with **stratifyR**.

Usage

```
fitverse_batch(data,
               cols      = NULL,
               dists     = "all",
               method    = "MLE",
               criterion = "AIC",
               top_n     = 3L,
               plot      = FALSE,
               progress  = NULL,
               verbose   = TRUE)

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

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

Arguments

data	A data.frame, matrix, or named list of numeric vectors. Non-numeric columns are silently ignored.
cols	Character vector of column names to process. If NULL (default), all numeric columns are processed.
dists	Distribution keys to try, or "all". Passed to fitverse().
method	Estimation method(s): "MLE", "MOM", "LMOM", or a character vector of multiple methods. Default "MLE".
criterion	"AIC" or "BIC". Default "AIC".
top_n	Integer: number of top-ranked distributions stored in each fit. Default 3.
plot	Logical: produce and print a diagnostic plot for each variable? Default FALSE (recommended for batch use).
progress	Optional callback function(p) for reporting overall batch progress, where p is a value in $[0, 1]$.
verbose	Logical: print per-column progress messages? Default TRUE.
x, object	An object of class "fitverse_batch".
...	Currently unused.

Details

fitverse_batch() iterates over columns with a tryCatch() wrapper so that a fitting failure for one column does not abort the entire batch. Failed columns are collected in the failed element of the returned object.

Column names containing spaces or special characters are handled correctly via make.names() internally.

Value

An S3 object of class "fitverse_batch" with elements:

fits Named list of "fitverse" objects, one per successfully fitted column.

summary Data frame with one row per column, columns: variable, n, best_dist, method, AIC, BIC, loglik, KS_p, AD_p.

failed Character vector of column names for which fitting failed, or character(0) if all succeeded.

criterion Character: the criterion used.

call The matched call.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

fitverse for single-column fitting.
generate_report for producing a report from batch results.

Examples

```
# Fit all numeric columns of the airquality dataset
data(airquality)
batch <- fitverse_batch(airquality,
                        method  = "MLE",
                        verbose = FALSE)
print(batch)
summary(batch)

# Access the fit for the Ozone column
batch$fits[["Ozone"]]
```

---

fitverse_censored *Fit Parametric Distributions to Censored or Truncated Data*

Description

Estimates distribution parameters by maximising the censored log-likelihood for data that may be right-censored, left-censored, interval-censored, and/or left-/right-truncated, or any combination of these. Seven distribution families are supported. Fits are ranked by AIC or BIC, and a nonparametric Kaplan–Meier estimate is overlaid automatically on the survival or CDF plot.

Usage

```
fitverse_censored(x,
                  status      = NULL,
                  x_upper     = NULL,
                  trunc_lower = -Inf,
                  trunc_upper =  Inf,
                  dists       = c("normal", "lognormal", "weibull",
                                  "exponential", "gamma",
                                  "loglogistic", "gumbel"),
                  criterion   = c("AIC", "BIC"),
                  n_starts    = 3L,
                  xlab        = "x",
                  draw        = TRUE)

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

## S3 method for class 'fitverse_censored'
plot(x, top_n = 3L, ...)
```

Arguments

`x`	Numeric vector of observed or censored values. For right-censored observations this is the last observed time; for left-censored observations it is the detection-limit value; for interval-censored observations it is the lower bound of the interval. In `print` and `plot` methods, `x` is the `"fitverse_censored"` object.
`status`	Integer vector of the same length as `x` encoding the observation type using the **survival**-package convention: `1` Completely observed (event occurred). `0` Right-censored (event had not yet occurred at last follow-up). `2` Left-censored (true value is below a detection limit). `3` Interval-censored (true value lies in `[x, x_upper]`). If `NULL` (default), all observations are treated as completely observed (`status = 1`).
`x_upper`	Numeric vector of upper bounds for interval-censored observations (`status == 3`). Must be the same length as `x`. Ignored for all rows where `status != 3`.
`trunc_lower`	Numeric scalar: left (lower) truncation bound. Observations below this value are never recorded; the likelihood is normalised by $P(X >$ `trunc_lower`$)$. Default `-Inf` (no left truncation).
`trunc_upper`	Numeric scalar: right (upper) truncation bound. Default `Inf` (no right truncation).
`dists`	Character vector of distribution keys to fit. Any subset of `c("normal", "lognormal", "weibull", "exponential", "gamma", "loglogistic", "gumbel")`. Default: all seven.
`criterion`	Character: `"AIC"` (default) or `"BIC"`. Used to rank the fitted distributions.

n_starts	Positive integer: number of random starting points per distribution. Increasing this reduces the risk of the optimiser converging to a local optimum. Default 3.
xlab	Character: variable label used in plot axis titles. Default "x".
draw	Logical: produce a survival-function (right-censored) or cumulative-distribution (otherwise) plot? Default TRUE.
top_n	Positive integer: number of top-ranked distributions to overlay on the plot. Default 3.
...	Further arguments passed to or from other methods.

Value

An object of class "fitverse_censored", which is a named list with the following elements:

fits List of fit objects ordered by criterion (best first). Each element is a list with: dist_key (character), display (character display name), par (named numeric vector of parameter estimates), log_lik (numeric), aic (numeric), bic (numeric), n_par (integer), converged (logical).

rankings Data frame with columns Rank, Distribution, LogLik, AIC, BIC, Converged, sorted by criterion.

x, status, x_upper Cleaned copies of the input data vectors (non-finite values removed).

trunc_lower, trunc_upper Truncation bounds as supplied.

xlab Variable label.

criterion Ranking criterion used ("AIC" or "BIC").

References

Meeker, W. Q. and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. Wiley, New York.

Lawless, J. F. (2003). *Statistical Models and Methods for Lifetime Data*, 2nd ed. Wiley, Hoboken.

Kaplan, E. L. and Meier, P. (1958). Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, **53**(282), 457–481. doi:10.1080/01621459.1958.10501452

See Also

fitverse for uncensored data fitting; fitverse_distributions for the full distribution library.

Examples

```
## Right-censored survival data (Weibull true model)
set.seed(1)
times  <- rweibull(100, shape = 1.8, scale = 50)
censor <- rexp(100, rate = 1/60)
x      <- pmin(times, censor)
status <- as.integer(times <= censor)   # 1 = event, 0 = right-censored

fit_s <- fitverse_censored(x, status,
                           dists = c("weibull", "lognormal", "gamma"),
```

```
                            xlab  = "Survival time (hours)")
print(fit_s)
plot(fit_s, top_n = 3)


## Left-censored environmental data (below detection limit)
set.seed(2)
conc       <- rlnorm(80, meanlog = 1.5, sdlog = 0.7)
dl         <- 3.0
x_env      <- ifelse(conc >= dl, conc, dl)
status_env <- ifelse(conc >= dl, 1L, 2L)

fit_e <- fitverse_censored(x_env, status_env,
                            dists = c("lognormal", "gamma"),
                            xlab  = "Concentration (mg/L)")
fit_e$rankings


## Left-truncated occupational exposure data
set.seed(3)
exposure <- rlnorm(300, meanlog = 2.5, sdlog = 0.5)
exposure <- exposure[exposure > 5]   # only workers above LOQ recorded

fit_t <- fitverse_censored(exposure, trunc_lower = 5,
                            dists = c("lognormal", "gamma", "weibull"),
                            xlab  = "Exposure (mg/m3)")
fit_t$fits[[1L]]$par
```

`FitVerse_datasets` *Sample Datasets Bundled with FitVerse*

Description

FitVerse includes several comma-separated value (CSV) files in `inst/extdata/` that can be used to explore the package's fitting functions. All files are accessed with `system.file()`.

Individual-observation datasets (for `fitverse()`):

`annual_max_flows.csv` Annual maximum daily peak flows (m^3/s) for the Rewa River at Navolau, Fiji, covering a 75-year record (1950–2024). The data are right-skewed; Gumbel, GEV, Log-Pearson III, and Log-Normal are natural candidates.

`wind_speeds.csv` Daily maximum wind gust speeds (m/s) recorded at Nadi Airport, Fiji, over 200 days (2020–2024). Weibull and Rayleigh distributions are commonly used for wind data.

`failure_times.csv` Failure times (hours) from a life test of 150 aluminium electrolytic capacitors under 85 °C stress. Weibull, Gamma, Log-Normal, and Birnbaum-Saunders are typical reliability choices.

`insurance_claims.csv` Residential property insurance claim amounts (in thousands of USD) from 180 claims settled by Pacific Insurance Ltd between 2018 and 2024. Log-Normal, Gamma, and Pareto distributions are common in loss modelling.

`concrete.csv` Compressive strength (MPa) and mix-design variables for 1030 concrete specimens (UCI Machine Learning Repository). The `Strength` column is suitable for fitting; Normal, Gamma, and Weibull are plausible choices.

Grouped / interval-censored datasets (for `fitverse_grouped()`):

`rainfall_grouped.csv` Annual maximum daily rainfall (mm) at Suva, Fiji, summarised into a frequency table with columns `midpoint` and `count` (80 years of record).

`failure_times_grouped.csv` Failure times for 120 electronic components given as an interval frequency table with columns `lower`, `upper`, and `count`.

`insurance_losses_grouped.csv` Property insurance claim amounts for 250 claims given as an interval frequency table with columns `lower`, `upper`, and `count`.

Format

Each file is a plain-text CSV with a column-name header row. Lines beginning with # are comments describing the dataset. All numeric values use the standard decimal point.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

`fitverse` for fitting to individual observations.
`fitverse_grouped` for fitting to grouped / interval data.

Examples

```
## Annual maximum flows
path <- system.file("extdata", "annual_max_flows.csv", package = "FitVerse")
x <- read.csv(path, comment.char = "#")$peak_flow_m3s
summary(x)


fit <- fitverse(x, xlab = "Peak flow (m3/s)", plot = FALSE)
print(fit)


## Grouped rainfall data
gpath <- system.file("extdata", "rainfall_grouped.csv", package = "FitVerse")
dat   <- read_grouped_csv(gpath)

gfit  <- fitverse_grouped(midpoints = dat$midpoints, counts = dat$counts,
                          xlab = "Rainfall (mm)")
print(gfit)
```

`fitverse_distributions` *List All Distributions Supported by FitVerse*

Description

Returns a data frame describing all 52 probability distributions that **FitVerse** can fit, including their distribution keys, full names, number of parameters, support, available estimation methods, and the underlying R function source.

This page also documents the mathematical details of the four hydrological / extreme-value distributions: Generalised Pareto (GPD), Pearson Type III (P3), Log-Pearson Type III (LP3), and Generalised Logistic (GLO).

Usage

```
fitverse_distributions(method = NULL)
```

Arguments

`method`	Optional character scalar: filter to distributions available for a specific estimation method. One of `"MLE"`, `"MOM"`, or `"LMOM"`. If `NULL` (default), all 52 distributions are returned.

Details

Complete distribution table (52 families)

Key	Distribution	Params	Support	Family
`normal`	Normal	2	$(-\infty, \infty)$	Symmetric
`logistic`	Logistic	2	$(-\infty, \infty)$	Symmetric
`cauchy`	Cauchy	2	$(-\infty, \infty)$	Symmetric
`laplace`	Laplace	2	$(-\infty, \infty)$	Symmetric
`student_t`	Student-t (location-scale)	3	$(-\infty, \infty)$	Symmetric
`truncnorm`	Truncated Normal	4	$[a, b]$	Symmetric
`skewnormal`	Skew-Normal	3	$(-\infty, \infty)$	Asymmetric
`gnorm`	Generalised Normal	3	$(-\infty, \infty)$	Asymmetric
`alaplace`	Asymmetric Laplace	3	$(-\infty, \infty)$	Asymmetric
`johnsonsu`	Johnson SU	4	$(-\infty, \infty)$	Asymmetric
`gamma`	Gamma	2	$(0, \infty)$	Right-skewed
`gamma3`	Gamma (3-parameter)	3	(μ, ∞)	Right-skewed
`lognormal`	Log-Normal	2	$(0, \infty)$	Right-skewed
`lnorm3`	Log-Normal (3-parameter)	3	(μ, ∞)	Right-skewed
`exponential`	Exponential	1	$(0, \infty)$	Right-skewed
`expexp`	Exponentiated Exponential	2	$(0, \infty)$	Right-skewed
`weibull`	Weibull (2-parameter)	2	$(0, \infty)$	Right-skewed
`weibull3p`	Weibull (3-parameter)	3	(μ, ∞)	Right-skewed
`expweibull`	Exponentiated Weibull	3	$(0, \infty)$	Right-skewed

loglogistic	Log-Logistic	2	$(0, \infty)$	Right-skewed
invgaussian	Inverse Gaussian	2	$(0, \infty)$	Right-skewed
invgamma	Inverse Gamma	2	$(0, \infty)$	Right-skewed
invweibull	Inverse Weibull	2	$(0, \infty)$	Right-skewed
gompertz	Gompertz	2	$(0, \infty)$	Right-skewed
birnbaum_saunders	Birnbaum-Saunders	2	$(0, \infty)$	Right-skewed
rayleigh	Rayleigh	1	$(0, \infty)$	Right-skewed
halfnormal	Half-Normal	1	$(0, \infty)$	Right-skewed
lindley	Lindley	1	$(0, \infty)$	Right-skewed
nakagami	Nakagami	2	$(0, \infty)$	Right-skewed
gengamma	Generalised Gamma	3	$(0, \infty)$	Right-skewed
pareto	Pareto	2	(x_m, ∞)	Heavy-tailed
burr	Burr XII	3	$(0, \infty)$	Heavy-tailed
lomax	Lomax (Pareto Type II)	2	$(0, \infty)$	Heavy-tailed
paralogistic	Paralogistic	2	$(0, \infty)$	Heavy-tailed
dagum	Dagum	3	$(0, \infty)$	Heavy-tailed
gb2	GB2 (Generalised Beta II)	4	$(0, \infty)$	Heavy-tailed
nig	Normal Inverse Gaussian	4	$(-\infty, \infty)$	Heavy-tailed
ghyp	Generalised Hyperbolic	5	$(-\infty, \infty)$	Heavy-tailed
gev	Generalised Extreme Value	3	depends on shape	Extreme-value
gumbel	Gumbel (Type I EV)	2	$(-\infty, \infty)$	Extreme-value
gpd	Generalised Pareto	3	$[\mu, \infty)$ or bdd	Extreme-value
glo	Generalised Logistic	3	depends on shape	Extreme-value
pearson3	Pearson Type III	3	$(-\infty, \infty)$	Extreme-value
lpearson3	Log-Pearson Type III	3	$(0, \infty)$	Extreme-value
kap4	Kappa-4	4	depends on shape	Extreme-value
wakeby	Wakeby	5	$(0, \infty)$	Extreme-value
beta	Beta	2	$(0, 1)$	Bounded
uniform	Uniform	2	$[a, b]$	Bounded
triangular	Triangular	3	$[a, b]$	Bounded
rtriangle	Right Triangular	2	$[a, b]$	Bounded
pert	PERT (Beta-PERT)	3	$[a, b]$	Bounded
kumaraswamy	Kumaraswamy	2	$(0, 1)$	Bounded

Hydrological and extreme-value distributions (v1.0 additions)

The four distributions below – GPD, Pearson III, Log-Pearson III, and GLO – are central to flood-frequency analysis and Peaks-Over-Threshold modelling. They share a common implementation strategy: closed-form `d/p/q/r` functions are registered in the internal `.FITVERSE_DPQR_OVERRIDES` table, bypassing any external package dependency and guaranteeing consistent behaviour across platforms.

Generalised Pareto Distribution (GPD) – key: `"gpd"`:

Parameters: μ (location / threshold), $\sigma > 0$ (scale), ξ (shape).

PDF ($\xi \neq 0$):

$$f(x\,;\,\mu,\sigma,\xi) = \frac{1}{\sigma}\Big(1 + \xi\,\frac{x-\mu}{\sigma}\Big)^{-1/\xi - 1}$$

Support: $x \geq \mu$ when $\xi \geq 0$; $\mu \leq x \leq \mu - \sigma/\xi$ when $\xi < 0$. Reduces to $\mathrm{Exp}(1/\sigma)$ when $\xi = 0$.

Quantile function:

$$Q(p) = \mu + \frac{\sigma}{\xi}\big[(1-p)^{-\xi} - 1\big], \quad \xi \neq 0 \qquad Q(p) = \mu - \sigma \log(1-p), \quad \xi = 0$$

Method of Moments: Fix $\hat{\mu} = \min(x) - \varepsilon$; let $y_i = x_i - \hat{\mu}$. Then $\hat{\xi} = (1 - \bar{y}^2/s_y^2)/2$, $\hat{\sigma} = \bar{y}(1-\hat{\xi})$. Valid only when $\hat{\xi} < 1/2$.

L-Moments: via `lmomco::pargpa()`. Sign convention: FitVerse $\xi = -\kappa_{\text{lmomco}}$.

Use case: Peaks-Over-Threshold analysis; insurance excess-of-loss modelling (Pickands, 1975).

Pearson Type III (P3) – key: `"pearson3"`**:**

Parameters: $\alpha > 0$ (shape), β (scale; may be negative), ζ (location).

Relationship to Gamma:

$$X \sim \text{P3}(\alpha, \beta, \zeta) \;\Leftrightarrow\; (X - \zeta)/\beta \sim \text{Gamma}(\alpha,\, \text{rate} = 1)$$

PDF:

$$f(x\,;\,\alpha,\beta,\zeta) = \frac{|\beta|^{-1}}{\Gamma(\alpha)}\Big(\frac{x-\zeta}{\beta}\Big)^{\alpha-1}\exp\Big(-\frac{x-\zeta}{\beta}\Big), \quad \frac{x-\zeta}{\beta} > 0$$

Positive β gives right-skewed; negative β gives left-skewed data.

Method of Moments: Let $\bar{x}, s, g$ be the sample mean, standard deviation, and skewness coefficient. Then:

$$\hat{\alpha} = 4/g^2, \qquad \hat{\beta} = sg/2, \qquad \hat{\zeta} = \bar{x} - 2s/g$$

L-Moments: via `lmomco::parpe3()`, which returns (μ, σ, γ). FitVerse converts using the MOM formulae above.

Use case: Annual peak-flow frequency analysis (US Bulletin 17C; England *et al.*, 2019); meteorological extremes.

Log-Pearson Type III (LP3) – key: `"lpearson3"`**:**

Parameters: $\alpha > 0$ (shape), β (scale), ζ (location) – all defined in log-space.

$$X \sim \text{LP3}(\alpha, \beta, \zeta) \;\Leftrightarrow\; \log X \sim \text{P3}(\alpha, \beta, \zeta)$$

PDF:

$$f(x\,;\,\alpha,\beta,\zeta) = \frac{f_{\text{P3}}(\log x\,;\,\alpha,\beta,\zeta)}{x}, \quad x > 0$$

CDF and quantile:

$$F(x) = F_{\text{P3}}(\log x), \qquad Q(p) = \exp\big[Q_{\text{P3}}(p)\big]$$

Method of Moments and L-Moments: Pearson III estimators applied to $\log x_i$. L-Moments use `lmomco::parpe3()` on the log-transformed sample.

Use case: The US federal standard for flood-frequency estimation (Bulletin 17C; England *et al.*, 2019).

Generalised Logistic Distribution (GLO) – key: `"glo"`**:**

Parameters: ξ (location), $\alpha > 0$ (scale), κ (shape).

CDF ($\kappa \neq 0$):

$$F(x\,;\,\xi,\alpha,\kappa) = \frac{1}{1 + \left[1 - \kappa(x-\xi)/\alpha\right]^{1/\kappa}}$$

Reduces to $\text{Logistic}(\xi, \alpha)$ when $\kappa = 0$.

Quantile function:

$$Q(p) = \xi + \frac{\alpha}{\kappa}\Big[1 - \Big(\frac{1-p}{p}\Big)^{\kappa}\Big], \quad \kappa \neq 0 \qquad Q(p) = \xi + \alpha \log\frac{p}{1-p}, \quad \kappa = 0$$

Support: $(-\infty,\ \xi + \alpha/\kappa)$ for $\kappa > 0$; $(\xi + \alpha/\kappa,\ \infty)$ for $\kappa < 0$; all real for $\kappa = 0$.

L-Moments: via `lmomco::parglo()`. No sign flip: FitVerse $\kappa = \kappa_{\text{lmomco}}$.

MOM is not implemented for GLO (complex moment system; use LMOM or MLE).

Use case: The recommended flood-frequency distribution in the United Kingdom (WINFAP; Robson & Reed, 1999).

L-moment shape-parameter sign conventions:

lmomco uses the opposite sign for the shape parameter of GEV and GPD:

Distribution	lmomco κ	FitVerse shape
GEV	κ	$-\kappa$ (sign flip)
GPD	κ	$-\kappa$ (sign flip)
GLO	κ	κ (no flip)

Value

A data frame with one row per distribution and columns:

Key Character: the string to pass in the `dists` argument of `fitverse()` and related functions.

Name Full distribution name.

Family Broad family tag (e.g. `"Extreme-value"`, `"Right-skewed"`).

Package The R package supplying `d/p/q/r` functions, or `NA` for distributions with FitVerse-internal implementations.

MLE Character `"Yes"` or `"-"`: MLE available?

MOM Character `"Yes"` or `"-"`: MOM available?

LMOM Character `"Yes"` or `"-"`: L-Moments available?

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

References

England, J. F., Jr., et al. (2019). *Guidelines for Determining Flood Flow Frequency – Bulletin 17C.* USGS Techniques and Methods, Book 4, Chapter B5.

Hosking, J. R. M. and Wallis, J. R. (1997). *Regional Frequency Analysis: An Approach Based on L-moments.* Cambridge University Press.

Pickands, J. (1975). Statistical inference using extreme order statistics. *The Annals of Statistics*, **3**(1), 119–131.

Robson, A. and Reed, D. (1999). *Flood Estimation Handbook, Volume 3.* Institute of Hydrology, Wallingford, UK.

See Also

`fitverse` for fitting any of the 52 distributions.
`compare_dists` for quick multi-distribution screening.
`return_level` for T-year return levels.
`bootstrap_ci` for parametric bootstrap confidence intervals.

Examples

```
# All 52 distributions
fitverse_distributions()

# Only those with L-moment support
fitverse_distributions(method = "LMOM")

# Extreme-value / hydrological families
ev <- fitverse_distributions()
ev[ev$Key %in% c("gev","gumbel","gpd","pearson3","lpearson3","glo"), ]

## ---- GPD (Peaks-Over-Threshold) ----------------------------------------
set.seed(42)
x_excess <- evd::rgpd(300, loc = 0, scale = 2, shape = 0.1)
fit_gpd   <- fitverse(x_excess, dists = "gpd", method = "LMOM",
                      plot = FALSE, verbose = FALSE)
coef(fit_gpd)
return_level(fit_gpd, return_periods = c(10, 50, 100))

## ---- Pearson III (Bulletin 17C style) ----------------------------------
## Pearson III with shape=5, scale=3, loc=10 is equivalent to
## a shifted Gamma: X = loc + Gamma(shape, scale)
set.seed(42)
x_p3  <- rgamma(200, shape = 5, scale = 3) + 10
fit_p3 <- fitverse(x_p3, dists = "pearson3", method = "LMOM",
                   plot = FALSE, verbose = FALSE)
coef(fit_p3)

## ---- Log-Pearson III ---------------------------------------------------
## Log-Pearson III: log(X) ~ Pearson III(shape, scale, loc)
## so X = exp(loc + Gamma(shape, scale))
set.seed(42)
```

```
x_lp3  <- exp(rgamma(200, shape = 5, scale = 0.4) + 2)
fit_lp3 <- fitverse(x_lp3, dists = "lpearson3", method = "LMOM",
                    plot = FALSE, verbose = FALSE)
coef(fit_lp3)

## ---- GLO (UK WINFAP style) ------------------------------------------------
## Generate GLO samples via the lmomco quantile function
set.seed(42)
x_glo  <- lmomco::qlmomco(runif(200),
            lmomco::vec2par(c(50, 8, 0.12), type = "glo"))
fit_glo <- fitverse(x_glo, dists = "glo", method = "LMOM",
                    plot = FALSE, verbose = FALSE)
coef(fit_glo)
return_level(fit_glo, return_periods = c(2, 10, 50, 100, 200))
```

`fitverse_grouped` *Fit Distributions to Grouped or Interval-Censored Data*

Description

`fitverse_grouped()` fits one or more parametric distributions to data that are available only in grouped (frequency-table) or interval-censored form. It accepts either class midpoints with counts, or explicit lower and upper interval boundaries with counts.

`read_grouped_csv()` reads a two- or three-column CSV file into the format expected by `fitverse_grouped()`.

Usage

```
fitverse_grouped(midpoints = NULL,
                 counts,
                 lower   = NULL,
                 upper   = NULL,
                 dists   = "auto",
                 method  = "All",
                 xlab    = "x",
                 top_n   = 5L,
                 verbose = FALSE)

read_grouped_csv(file, sep = ",")
```

Arguments

`midpoints`	Numeric vector of class midpoints (one per class). Supply either `midpoints` or both `lower` and `upper`, but not both.
`counts`	Integer vector of observed frequencies for each class. Must be the same length as `midpoints` (or `lower`/`upper`), all non-negative, and sum to at least 10.
`lower`	Numeric vector of lower interval boundaries. Required when `midpoints` is not supplied.

upper	Numeric vector of upper interval boundaries. Required when midpoints is not supplied. Each element must exceed the corresponding element of lower.
dists	Character vector of distribution names to fit, or "auto" (default) to use all distributions in the FitVerse registry that support MLE.
method	Fitting method: "MLE", "MOM", "LMOM", or "All" (default).
xlab	Character label for the data axis used in output summaries. Default "x".
top_n	Integer: number of top-ranked distributions to include in the printed summary. Default 5L.
verbose	Logical: print per-distribution fitting progress? Default FALSE.
file	Path to a CSV file. The file may contain comment lines beginning with #.
sep	Column separator for read_grouped_csv(). Default ",".

Details

Input formats: fitverse_grouped() accepts two input formats.

Midpoint format: Supply midpoints and counts. Interval boundaries are reconstructed by assuming equal-width classes.

Boundary format: Supply lower, upper, and counts. Classes need not be equal-width. Open-ended classes (e.g., an upper bound of Inf) are handled by replacing the infinite bound with a large finite value for midpoint computation.

Estimation: For each candidate distribution, parameters are estimated by maximising the interval-censored log-likelihood

$$\ell(\theta) = \sum_j n_j \log\bigl[F(u_j;\theta) - F(l_j;\theta)\bigr]$$

where n_j is the count in class j, and l_j, u_j are its lower and upper boundaries.

CSV format for read_grouped_csv()**:** The CSV file should have either two columns (midpoint, count) or three columns (lower, upper, count). Column names are detected automatically from a set of common variants (mid, midpoint, lower, upper, etc.). Lines starting with # are treated as comments and ignored.

Value

fitverse_grouped() returns an object of class "fitverse_grouped" containing:

results A data frame of fitted distributions ranked by AIC, with columns for distribution name, AIC, BIC, log-likelihood, estimated parameters, and goodness-of-fit statistics.

best The name of the best-fitting distribution.

data A list with elements midpoints, counts, lower, and upper.

call The matched call.

read_grouped_csv() returns a named list with elements midpoints and counts (midpoint format) or lower, upper, and counts (boundary format), ready to be passed directly to fitverse_grouped().

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

`fitverse` for fitting to individual (ungrouped) data.
`fitverse_distributions` for the full distribution catalogue.

Examples

```
## Midpoint format
mid   <- c(5, 15, 25, 35, 45, 55, 65, 75)
count <- c(3, 12, 25, 30, 18, 8, 3, 1)
fit <- fitverse_grouped(midpoints = mid, counts = count,
                        method = "MLE", verbose = FALSE)
print(fit)

## Boundary format
low  <- c(0, 10, 20, 30, 40, 50, 60, 70)
high <- c(10, 20, 30, 40, 50, 60, 70, 80)
fit2 <- fitverse_grouped(lower = low, upper = high, counts = count,
                         method = "MLE", verbose = FALSE)
print(fit2)

## Reading from CSV
## A file with columns: lower, upper, count
tmp <- tempfile(fileext = ".csv")
writeLines(c(
  "# Example grouped data",
  "lower,upper,count",
  "0,10,5",
  "10,20,18",
  "20,30,30",
  "30,40,25",
  "40,50,12",
  "50,60,4"
), tmp)
dat <- read_grouped_csv(tmp)
fit3 <- fitverse_grouped(lower = dat$lower, upper = dat$upper,
                         counts = dat$counts)
print(fit3)
```

`fitverse_s3methods` *S3 Methods for* `fitverse` *Objects*

Description

Standard S3 methods for objects of class `"fitverse"` as returned by `fitverse()`. These methods provide a consistent interface for extracting, displaying, and serialising fitted distribution results.

Usage

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

## S3 method for class 'fitverse'
coef(object, ...)

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

## S3 method for class 'fitverse'
AIC(object, ..., k = 2)

## S3 method for class 'fitverse'
BIC(object, ...)

## S3 method for class 'fitverse'
as.list(x, ...)
```

Arguments

`x, object`	An object of class `"fitverse"`.
`k`	Numeric: penalty per parameter used by `AIC()`. Default `2` (standard AIC). Pass `k = log(n)` to obtain BIC-equivalent penalisation.
`...`	Currently unused.

Details

`print.fitverse()` Prints a concise single-screen summary: best-fit distribution, estimation method, number of parameters, parameter estimates, log-likelihood, AIC, BIC, and up to three goodness-of-fit test results.

`coef.fitverse()` Returns the named numeric vector of parameter estimates for the best-fit model. For example, `c(shape = 2.14, rate = 0.43)` for a Gamma fit.

`logLik.fitverse()` Returns an object of class `"logLik"` with the maximised log-likelihood and the degrees of freedom (number of parameters). Compatible with `stats::AIC()` and `stats::BIC()`.

`AIC.fitverse()` Returns the Akaike Information Criterion $\text{AIC} = -2\ell + k \cdot p$, where ℓ is the log-likelihood, p is the number of parameters, and k is the penalty (default 2).

`BIC.fitverse()` Returns the Bayesian Information Criterion $\text{BIC} = -2\ell + p \log n$.

`as.list.fitverse()` Converts the `fitverse` object to a plain list suitable for JSON serialisation (used by `to_json()`). Non-serialisable internals (raw fit objects, closures) are stripped. Returns: `best_fit` (dist_key, method, params, loglik, aic, bic), `ranking`, `data_summary`, `n`, `xlab`, `criterion`.

Value

`print.fitverse()` Invisibly returns `x`.

`coef.fitverse()` Named numeric vector of parameter estimates.

`logLik.fitverse()` Object of class `"logLik"`.

`AIC.fitverse()` Numeric scalar.

`BIC.fitverse()` Numeric scalar.

`as.list.fitverse()` A named list safe for JSON serialisation.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

`fitverse`, `to_json`, `plot.fitverse`.

Examples

```
set.seed(42)
x <- rgamma(300, shape = 2, rate = 0.5)
fit <- fitverse(x, plot = FALSE, verbose = FALSE)

print(fit)
coef(fit)
logLik(fit)
AIC(fit)
BIC(fit)

# Serialisation-ready list
lst <- as.list(fit)
names(lst)
```

---

fitverse_summary *Summary of a fitverse Object*

Description

`summary.fitverse` returns an object of class `"fitverse_summary"` containing the best-fit results and a ranked table of all models fitted. Printing the object (via `print.fitverse_summary`) displays a formatted diagnostic report; assigning the result to a variable suppresses output.

Usage

```
## S3 method for class 'fitverse'
summary(object, top_n = 10L, ...)
## S3 method for class 'fitverse_summary'
print(x, ...)
```

Arguments

object	A fitverse object.
top_n	Integer: number of rows to show in the ranking table. Default 10.
x	A fitverse_summary object.
...	Ignored.

Value

An object of class "fitverse_summary", a list with components:

object The original fitverse object.

top_n The number of rows requested.

data_summary Data summary list (or NULL).

ranking Data frame with the top top_n rows of the ranking table.

print.fitverse_summary returns x invisibly.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

fitverse, print.fitverse, ranking, all_fits

Examples

```
set.seed(1)
x   <- rgamma(200, shape = 2, rate = 0.5)
fit <- fitverse(x)

# Assign without printing:
s <- summary(fit)

# Inspect individual components:
head(s$ranking)
s$data_summary$mean

# Print explicitly:
print(s)
```

generate_report *Generate an HTML or PDF Report for a Fitted Distribution*

Description

Produces a standalone, self-contained HTML (or PDF) report that documents the fitted distribution, estimated parameters, goodness-of-fit tests, descriptive statistics, diagnostic plots, and (optionally) bootstrap confidence intervals.

If **rmarkdown** is installed, the report is rendered from an R Markdown template. Otherwise, a pure HTML fallback is used with the diagnostic plot embedded as a base64-encoded PNG image.

Usage

```
generate_report(fit,
                output_file = NULL,
                title       = "FitVerse: Distribution Fitting Report",
                author      = "",
                boot        = NULL,
                top_n       = 5L,
                open        = TRUE)
```

Arguments

fit	An object of class "fitverse" as returned by fitverse().
output_file	Character: path to the output file. If NULL (default), a temporary file is used: fitverse_report_<dist>_<timestamp>.html.
title	Character: report title. Defaults to "FitVerse: Distribution Fitting Report".
author	Character: author name to include in the report header. Default "" (no author line).
boot	An object of class "fitverse_bootstrap" (from bootstrap_ci()), or NULL. If supplied, a confidence-interval table is included in the report.
top_n	Integer: number of distributions to show in the ranking table. Default 3.
open	Logical: open the report in the system's default browser after writing? Default TRUE. Set to FALSE in automated pipelines or when running inside **shiny**.

Details

R Markdown path (recommended): When **rmarkdown** is available, generate_report() writes a temporary .Rmd file and calls rmarkdown::render() to produce a self-contained HTML document using the html_document format with self_contained = TRUE. The fitted object is passed via a temporary .rds file to avoid workspace pollution.

HTML fallback path: When **rmarkdown** is unavailable, a self-contained HTML file is built in pure R:

- The diagnostic plot is saved to a temporary PNG and encoded as a base64 data URI using **base64enc** (if available) or a manual hex-to-base64 conversion.
- Tables are rendered as HTML `<table>` elements.
- The output is a single-file HTML document with embedded CSS.

Value

Invisibly returns the normalised path to the output file (character).

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

`bootstrap_ci` for confidence intervals to include in the report.
`save_plot` to save the diagnostic plot separately.

Examples

```
set.seed(42)
x <- rgamma(300, shape = 2, rate = 0.5)
fit <- fitverse(x, xlab = "Waiting time (s)", plot = FALSE, verbose = FALSE)

# Write report to a temporary directory so no files are left in the working dir
path <- generate_report(fit,
                        title       = "Gamma Fit Example",
                        author      = "Karuna Reddy",
                        output_file = file.path(tempdir(), "fitverse_report.html"),
                        open        = FALSE)
message("Report saved to: ", path)
```

---

lmrd *L-Moment Ratio Diagram and Extreme-Value Threshold Selection*

Description

Two diagnostic tools for exploratory data analysis in **FitVerse**.

`lmrd()` produces an L-moment ratio diagram plotting the sample (τ_3, τ_4) pair against the theoretical curves of major distribution families (GLO, GEV, GNO, PE3, GPA) and their known special cases (Normal, Logistic, Gumbel, Exponential, Uniform). The diagram guides distributional identification before formal model fitting.

`ev_threshold()` plots three threshold-stability diagnostics for Peaks-Over-Threshold (POT) extreme-value analysis: the Mean Residual Life (MRL) plot, the shape-stability plot, and the modified-scale-stability plot. These are used to identify the lowest threshold above which the Generalised Pareto Distribution (GPD) is a valid model for exceedances.

Usage

```
lmrd(x = NULL,
     xlab = NULL,
     distributions = c("glo", "gev", "gno", "pe3", "gpa"),
     show_ref_points = TRUE,
     draw = TRUE)

ev_threshold(x,
             thresholds      = NULL,
             n_thresholds    = 40L,
             min_exceedances = 10L,
             conf            = 0.95,
             xlab            = "x",
             draw            = TRUE)
```

Arguments

x	Numeric vector of observations. For lmrd, may be NULL to produce a blank reference diagram.
xlab	Character: x-axis label used in plots. Defaults to NULL (auto-labelled) for lmrd and "x" for ev_threshold.
distributions	Character vector: distribution families whose theoretical L-moment ratio curves are drawn. Default covers GLO, GEV, GNO, PE3, and GPA.
show_ref_points	Logical: whether to plot the known special-case reference points (Normal, Logistic, Gumbel, Exponential, Uniform). Default TRUE.
draw	Logical: if TRUE (default) the plot is printed; otherwise the ggplot object is returned invisibly.
thresholds	Optional numeric vector of candidate threshold values. If NULL (default), thresholds are chosen automatically.
n_thresholds	Integer: number of candidate threshold values to evaluate when thresholds = NULL. Default 40.
min_exceedances	Integer: minimum number of exceedances required at each threshold. Thresholds with fewer exceedances are excluded. Default 10.
conf	Numeric in $(0, 1)$: confidence level for the pointwise confidence bands. Default 0.95.

Details

lmrd() computes the sample L-skewness $\hat{\tau}_3$ and L-kurtosis $\hat{\tau}_4$ via lmoms and plots the sample point against the theoretical loci for the GLO, GEV, GPA, GNO, and PE3 families. Exact closed-form curves are used for GLO, GEV, and GPA (Hosking, 1990); polynomial approximations are used for GNO and PE3 (Hosking and Wallis, 1997).

ev_threshold() fits a GPD to exceedances at each candidate threshold u and extracts the estimated shape $\hat{\xi}$ and modified scale $\hat{\sigma}^* = \hat{\sigma} - \hat{\xi}u$. A stable region in both plots, combined with approximate linearity of the MRL plot, indicates a suitable threshold.

Value

Both functions return a ggplot object invisibly and (by default) display the plot.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

References

Hosking, J. R. M. (1990). L-moments: Analysis and estimation of distributions using linear combinations of order statistics. *Journal of the Royal Statistical Society B*, **52**(1), 105–124. doi:10.1111/j.25176161.1990.tb01775.x

Hosking, J. R. M. and Wallis, J. R. (1997). *Regional Frequency Analysis: An Approach Based on L-Moments*. Cambridge University Press.

Davison, A. C. and Smith, R. L. (1990). Models for exceedances over high thresholds. *Journal of the Royal Statistical Society B*, **52**(3), 393–442.

See Also

fitverse with dists = "gpd" for formal GPD fitting after threshold selection.

Examples

```
set.seed(42)
x <- rgamma(300, shape = 2, rate = 0.5)
lmrd(x, xlab = "Simulated Gamma data")



set.seed(1)
x_ev <- rexp(500, rate = 0.1)
ev_threshold(x_ev, n_thresholds = 30, xlab = "Simulated exponential")
```

---

plot.fitverse *Diagnostic Plot for a* fitverse *Object*

Description

Produces a two-panel diagnostic figure for a fitted fitverse object:

1. A histogram of the data overlaid with the fitted density curves for the top-ranked distributions. The PDF formula is annotated directly on the figure.
2. A Q-Q plot of the empirical quantiles against theoretical quantiles from the best-fit distribution.

Either or both panels can be requested. The plot can optionally be returned as an interactive **plotly** HTML widget.

Usage

```
## S3 method for class 'fitverse'
plot(x,
     which       = "both",
     top_n       = 3L,
     xlab        = NULL,
     interactive = FALSE,
     draw        = TRUE,
     ...)
```

Arguments

x	An object of class "fitverse" as returned by fitverse().
which	Character scalar controlling which panels to draw. One of: "both" (default) Both density and Q-Q panels side by side. "density" Density histogram with overlaid PDF curves only. "qq" Q-Q plot only.
top_n	Integer: number of top-ranked models to overlay on the density panel. The best-fit model is drawn as a solid line; others as dashed lines with distinct colours. Default 3.
xlab	Character: x-axis label. If NULL (default), falls back to the xlab stored in x (set when fitverse() was called), then to "x" if that is also absent.
interactive	Logical: if TRUE and **plotly** is installed, converts the ggplot2 figure to a **plotly** interactive widget. Default FALSE.
draw	Logical: if TRUE (default), print the plot to the current graphics device. Set to FALSE to return the plot object silently without displaying it.
...	Currently unused; reserved for future extensions.

Details

The density panel uses ggplot2::geom_histogram(aes(y = after_stat(density))) with Freedman-Diaconis binning, overlaid with geom_line() curves evaluated on a fine grid of 512 points spanning the observed data range.

The Q-Q panel compares theoretical quantiles from the best-fit distribution (evaluated via qfitverse()) against the empirical quantiles of the data, with a 45-degree reference line.

When which = "both", the two ggplot2 objects are arranged side-by-side using gridExtra::grid.arrange(). When interactive = TRUE and which != "both", a single panel is converted with plotly::ggplotly().

Value

Invisibly returns the plot object(s):

- A single ggplot2 object when which is "density" or "qq".
- A grob (arranged grid object) when which = "both".
- A plotly widget when interactive = TRUE.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

save_plot to write the figure to a PNG or PDF file.
fitverse which calls this function automatically when plot = TRUE.

Examples

```
set.seed(42)
x <- rgamma(300, shape = 2, rate = 0.5)
fit <- fitverse(x, plot = FALSE, verbose = FALSE)

# Both panels (default)
plot(fit)

# Density panel only with custom x-axis label
plot(fit, which = "density", xlab = "Value", top_n = 5)

# Q-Q plot only
plot(fit, which = "qq")
```

---

ranking *Extract the Ranking Table from a Fitted Object*

Description

Retrieves the full ranking table from a fitverse object without using the $ operator. The table contains one row per distribution fitted, sorted by the selection criterion (AIC or BIC).

Usage

```
ranking(object, ...)
## S3 method for class 'fitverse'
ranking(object, ...)
```

Arguments

object	A fitverse object.
...	Passed to methods.

Value

A data frame with columns including Rank, Distribution, Method, AIC, Delta_AIC, BIC, KS_p, AD_p, and others depending on the methods used.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

fitverse, all_fits, summary.fitverse

Examples

```
set.seed(1)
x   <- rgamma(200, shape = 2, rate = 0.5)
fit <- fitverse(x)
r   <- ranking(fit)
head(r)
```

---

return_level *Compute T-Year Return Levels from a Fitted Distribution*

Description

Computes return levels (quantiles) corresponding to specified return periods from the best-fit distribution in a `fitverse` object. Return levels are widely used in hydrology, climatology, and actuarial science to characterise the magnitude of rare events.

Usage

```
return_level(fit,
             return_periods = c(2, 5, 10, 20, 50, 100, 200, 500))
```

Arguments

`fit`	An object of class `"fitverse"` as returned by `fitverse()`.
`return_periods`	Numeric vector of return periods (in years, or whichever time unit the data represent). Must be ≥ 1. Default: `c(2, 5, 10, 20, 50, 100, 200, 500)`.

Details

A T-year return level x_T is the value that is exceeded on average once every T years (or time units), i.e.:

$$P(X > x_T) = \frac{1}{T}$$

$$x_T = F^{-1}\left(1 - \frac{1}{T}\right)$$

where F^{-1} is the quantile function of the fitted distribution.

Quantiles are computed using `qfitverse()`, which resolves the quantile function from the internal distribution registry.

Value

A data frame with one row per return period and columns:

`T` Numeric: the return period.

`exceedance_prob` Numeric: $1/T$, the annual exceedance probability.

`return_level` Numeric: the estimated T-year return level (upper $1/T$ quantile of the fitted distribution).

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

References

Coles, S. (2001). *An Introduction to Statistical Modeling of Extreme Values*. Springer, London.

Katz, R. W., Parlange, M. B. and Naveau, P. (2002). Statistics of extremes in hydrology. *Advances in Water Resources*, **25**(8-12), 1287–1304.

See Also

`qfitverse` for quantiles at arbitrary probabilities.
`bootstrap_ci` for confidence intervals on return levels.

Examples

```
# Flood frequency analysis with GEV distribution
set.seed(10)
floods <- evd::rgev(80, loc = 200, scale = 40, shape = 0.15)

fit <- fitverse(floods,
                dists   = c("gev", "gumbel", "lognormal"),
                method  = "MLE",
                xlab    = "Annual maximum flood (m3/s)",
                plot    = FALSE,
                verbose = FALSE)

rl <- return_level(fit)
print(rl)

# 100-year return level
rl[rl$T == 100, "return_level"]
```

save_plot *Save a FitVerse Diagnostic Plot to File*

Description

Renders the two-panel diagnostic plot for a `fitverse` object and saves it as a high-resolution PNG (or PDF) file, without displaying the figure on the current graphics device. Suitable for embedding in reports or automated pipelines.

Usage

```
save_plot(fit,
          file        = file.path(tempdir(), "fitverse_plot.png"),
          which       = "both",
          top_n       = 3L,
          xlab        = NULL,
          width       = 14,
          height      = 5.5,
          dpi         = 600,
          ...)
```

Arguments

`fit`	An object of class `"fitverse"`.
`file`	Character: output file path. The file extension determines the format: `.png` (default), `.pdf`, or `.svg`. Default `file.path(tempdir(), "fitverse_plot.png")`.
`which`	Which panels to save: `"both"` (default), `"density"`, or `"qq"`. See `plot.fitverse` for details.
`top_n`	Number of top-ranked distributions to show on the density panel. Default `3`.
`xlab`	Character: x-axis label. Defaults to `fit$xlab` if present, otherwise `"x"`.
`width`	Numeric: plot width in inches. Default `14`.
`height`	Numeric: plot height in inches. Default `5.5`.
`dpi`	Integer: resolution in dots per inch. Default `600`. Ignored for PDF output.
`...`	Additional arguments passed to `plot.fitverse()`.

Details

Uses `grDevices::png()` (or `grDevices::pdf()` / `grDevices::svg()` depending on file extension) and `gridExtra::grid.arrange()` to render both panels in a single device call, ensuring consistent alignment.

The function opens the graphics device, renders the figure, and closes the device via `on.exit()`, so the current interactive device is unaffected.

Value

Invisibly returns the normalised path to the saved file (character).

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

plot.fitverse for interactive display.

Examples

```
set.seed(42)
x <- rgamma(300, shape = 2, rate = 0.5)
fit <- fitverse(x, plot = FALSE, verbose = FALSE)

save_plot(fit, file = tempfile(fileext = ".png"))
```

---

to_json *Serialise a* fitverse *Object to JSON*

Description

Converts a fitverse object to a JSON string. Useful for passing fit results to web front-ends, storing results in document databases, or integrating with REST APIs.

The serialisation uses toJSON() on the output of as.list.fitverse(), which strips non-serialisable internals (raw optimiser fit objects, closures, environments) and returns a flat, portable representation.

Usage

```
to_json(x, pretty = TRUE, ...)

## S3 method for class 'fitverse'
to_json(x, pretty = TRUE, ...)
```

Arguments

x	An object with a to_json method; currently "fitverse" objects are supported.
pretty	Logical: if TRUE (default), produce indented, human-readable JSON. If FALSE, produce compact JSON (smaller output, faster parsing).
...	Additional arguments passed to jsonlite::toJSON(), for example digits, na or null.

Details

Requires the **jsonlite** package (listed in `Suggests`). An informative error is raised if it is not installed.

The JSON output includes:

- `best_fit`: distribution key, method, parameter estimates, loglik, AIC, BIC.
- `ranking`: full ranking data frame.
- `data_summary`: descriptive statistics.
- `n`, `xlab`, `criterion`.

The raw data vector is deliberately *not* included to keep the output compact and privacy-friendly for web contexts.

Value

A single character string containing the JSON representation of `x`.

Author(s)

Karuna G. Reddy <karuna.reddy@auckland.ac.nz>
M. G. M. Khan <mgm.khan@usp.ac.fj>

See Also

`as.list.fitverse` for the underlying list conversion.
`fromJSON` to parse the JSON back to R.

Examples

```
if (requireNamespace("jsonlite", quietly = TRUE)) {
  set.seed(42)
  x <- rgamma(300, shape = 2, rate = 0.5)
  fit <- fitverse(x, plot = FALSE, verbose = FALSE)

  json_str <- to_json(fit)
  cat(substr(json_str, 1, 400))

  # Round-trip back to R list
  lst <- jsonlite::fromJSON(json_str)
  lst$best_fit$params
}
```

Index

∗ **IO**
generate_report, 45
to_json, 54
∗ **datasets**
FitVerse_datasets, 32
∗ **descriptive**
compute_data_summary, 11
∗ **distribution**
bootstrap_ci, 7
compare_dists, 9
dfitverse, 13
dpqr_custom, 14
dweibull3p, 20
fitverse, 22
FitVerse-package, 3
fitverse_batch, 27
fitverse_distributions, 34
fitverse_grouped, 39
generate_report, 45
lmrd, 46
plot.fitverse, 48
return_level, 51
to_json, 54
∗ **extreme**
return_level, 51
∗ **hplot**
lmrd, 46
plot.fitverse, 48
save_plot, 53
∗ **htest**
bootstrap_ci, 7
∗ **methods**
all_fits, 6
fitverse_s3methods, 41
fitverse_summary, 43
ranking, 50
∗ **models**
compare_dists, 9
fitverse, 22
fitverse_batch, 27
fitverse_distributions, 34
fitverse_grouped, 39
∗ **package**
FitVerse-package, 3
∗ **print**
fitverse_s3methods, 41
∗ **univar**
compute_data_summary, 11
fitverse, 22
FitVerse-package, 3
∗ **utilities**
FitVerseApp, 26

AIC.fitverse, *4*
AIC.fitverse *(*fitverse_s3methods*)*, 41
all_fits, 6, *44*, *51*
annual_max_flows *(*FitVerse_datasets*)*, 32
as.list.fitverse, *4*, *54, 55*
as.list.fitverse *(*fitverse_s3methods*)*, 41

BIC.fitverse, *4*
BIC.fitverse *(*fitverse_s3methods*)*, 41
bootstrap_ci, *4*, 7, *25*, *38*, *45, 46*, *52*

coef.fitverse, *4*
coef.fitverse *(*fitverse_s3methods*)*, 41
compare_dists, *3*, 9, *25*, *38*
compute_data_summary, 11, *23*, *25*
concrete.csv *(*FitVerse_datasets*)*, 32

dalaplace *(*dpqr_custom*)*, 14
datatable, *10*
dbirnbaum_saunders *(*dpqr_custom*)*, 14
ddagum *(*dpqr_custom*)*, 14
dexpexp *(*dpqr_custom*)*, 14
dexpweibull *(*dpqr_custom*)*, 14
dfitverse, *3*, 13, *20*
dgamma3 *(*dpqr_custom*)*, 14

dgb2 (dpqr_custom), 14
dgengamma (dpqr_custom), 14
dghyp (dpqr_custom), 14
dgnorm (dpqr_custom), 14
dgompertz (dpqr_custom), 14
dhalfnormal (dpqr_custom), 14
dinvgaussian (dpqr_custom), 14
djohnsonsu (dpqr_custom), 14
dkap4 (dpqr_custom), 14
dkumaraswamy (dpqr_custom), 14
dlaplace (dpqr_custom), 14
dlindley (dpqr_custom), 14
dlnorm3 (dpqr_custom), 14
dlomax (dpqr_custom), 14
dnakagami (dpqr_custom), 14
dnig (dpqr_custom), 14
dpqr_custom, 14
drayleigh (dpqr_custom), 14
drtriangle (dpqr_custom), 14
dstudent_t (dpqr_custom), 14
dtruncnorm (dpqr_custom), 14
dwakeby (dpqr_custom), 14
dweibull, *21*
dweibull3p, 20

ev_threshold, *3*
ev_threshold (lmrd), 46

failure_times (FitVerse_datasets), 32
failure_times_grouped.csv (FitVerse_datasets), 32
FitVerse (FitVerse-package), 3
fitverse, *3*, *6–14*, *20*, *21*, 22, *27–29*, *31–33*, *37*, *38*, *41–45*, *48–51*
FitVerse-package, 3
fitverse_batch, *3*, 27
fitverse_censored, *3*, 29
FitVerse_datasets, *4*, *6*, 32
fitverse_distributions, *3*, *6*, *10*, *20*, *22*, *25*, *31*, 34, *41*
fitverse_grouped, *3*, *6*, *33*, 39
fitverse_s3methods, 41
fitverse_summary, 43
FitVerseApp, *22*, 26
fromJSON, *55*

generate_report, *4*, *25*, *29*, 45

insurance_claims (FitVerse_datasets), 32
insurance_losses_grouped.csv (FitVerse_datasets), 32

kable, *10*

lmoms, *47*
lmrd, *3*, 46
logLik.fitverse, *4*
logLik.fitverse (fitverse_s3methods), 41

palaplace (dpqr_custom), 14
pbirnbaum_saunders (dpqr_custom), 14
pdagum (dpqr_custom), 14
pexpexp (dpqr_custom), 14
pexpweibull (dpqr_custom), 14
pfitverse, *3*
pfitverse (dfitverse), 13
pgamma3 (dpqr_custom), 14
pgb2 (dpqr_custom), 14
pgengamma (dpqr_custom), 14
pghyp (dpqr_custom), 14
pgnorm (dpqr_custom), 14
pgompertz (dpqr_custom), 14
phalfnormal (dpqr_custom), 14
pinvgaussian (dpqr_custom), 14
pjohnsonsu (dpqr_custom), 14
pkap4 (dpqr_custom), 14
pkumaraswamy (dpqr_custom), 14
plaplace (dpqr_custom), 14
plindley (dpqr_custom), 14
plnorm3 (dpqr_custom), 14
plomax (dpqr_custom), 14
plot.fitverse, *4*, *23*, *25*, *43*, 48, *53*, *54*
plot.fitverse_censored (fitverse_censored), 29
pnakagami (dpqr_custom), 14
pnig (dpqr_custom), 14
prayleigh (dpqr_custom), 14
print.fitverse, *4*, *44*
print.fitverse (fitverse_s3methods), 41
print.fitverse_batch (fitverse_batch), 27
print.fitverse_bootstrap (bootstrap_ci), 7
print.fitverse_censored (fitverse_censored), 29
print.fitverse_summary (fitverse_summary), 43
prtriangle (dpqr_custom), 14

pstudent_t (dpqr_custom), 14
ptruncnorm (dpqr_custom), 14
pwakeby (dpqr_custom), 14
pweibull, *21*
pweibull3p (dweibull3p), 20

qalaplace (dpqr_custom), 14
qbirnbaum_saunders (dpqr_custom), 14
qdagum (dpqr_custom), 14
qexpexp (dpqr_custom), 14
qexpweibull (dpqr_custom), 14
qfitverse, *3*, *49*, *51*, *52*
qfitverse (dfitverse), 13
qgamma3 (dpqr_custom), 14
qgb2 (dpqr_custom), 14
qgengamma (dpqr_custom), 14
qghyp (dpqr_custom), 14
qgnorm (dpqr_custom), 14
qgompertz (dpqr_custom), 14
qhalfnormal (dpqr_custom), 14
qinvgaussian (dpqr_custom), 14
qjohnsonsu (dpqr_custom), 14
qkap4 (dpqr_custom), 14
qkumaraswamy (dpqr_custom), 14
qlaplace (dpqr_custom), 14
qlindley (dpqr_custom), 14
qlnorm3 (dpqr_custom), 14
qlomax (dpqr_custom), 14
qnakagami (dpqr_custom), 14
qnig (dpqr_custom), 14
qrayleigh (dpqr_custom), 14
qrtriangle (dpqr_custom), 14
qstudent_t (dpqr_custom), 14
qtruncnorm (dpqr_custom), 14
qwakeby (dpqr_custom), 14
qweibull, *21*
qweibull3p (dweibull3p), 20

rainfall_grouped.csv
(FitVerse_datasets), 32
ralaplace (dpqr_custom), 14
ranking, *7*, *44*, 50
rbirnbaum_saunders (dpqr_custom), 14
rdagum (dpqr_custom), 14
read_grouped_csv, *4*
read_grouped_csv (fitverse_grouped), 39
return_level, *3*, *8*, *9*, *14*, *25*, *38*, 51
rexpexp (dpqr_custom), 14
rexpweibull (dpqr_custom), 14
rfitverse, *3*
rfitverse (dfitverse), 13
rgamma3 (dpqr_custom), 14
rgb2 (dpqr_custom), 14
rgengamma (dpqr_custom), 14
rghyp (dpqr_custom), 14
rgnorm (dpqr_custom), 14
rgompertz (dpqr_custom), 14
rhalfnormal (dpqr_custom), 14
rinvgaussian (dpqr_custom), 14
rjohnsonsu (dpqr_custom), 14
rkap4 (dpqr_custom), 14
rkumaraswamy (dpqr_custom), 14
rlaplace (dpqr_custom), 14
rlindley (dpqr_custom), 14
rlnorm3 (dpqr_custom), 14
rlomax (dpqr_custom), 14
rnakagami (dpqr_custom), 14
rnig (dpqr_custom), 14
rrayleigh (dpqr_custom), 14
rrtriangle (dpqr_custom), 14
rstudent_t (dpqr_custom), 14
rtruncnorm (dpqr_custom), 14
runApp, *27*
rwakeby (dpqr_custom), 14
rweibull, *21*
rweibull3p (dweibull3p), 20

save_plot, *4*, *46*, *50*, 53
summary.fitverse, *4*, *7*, *11*, *12*, *25*, *51*
summary.fitverse (fitverse_summary), 43
summary.fitverse_batch
(fitverse_batch), 27

to_json, *4*, *42*, *43*, 54
toJSON, *54*

wind_speeds (FitVerse_datasets), 32