Package ‘fitdistrBayes’

August 6, 2026

Type Package

Title Objective Bayesian Distribution Fitting

Version 0.2.0

Description Fits common univariate distributions using registered objective Bayesian priors, including Jeffreys, reference, and maximal data information priors. Model-specific posterior propriety and moment conditions are checked before computation. Exact simulation, marginalization, slice sampling, and adaptive Metropolis algorithms are selected from posterior structure, with a common interface for summaries, diagnostics, prediction, and pointwise log-likelihood evaluation. The reference-prior framework follows Bernardo (1979) <doi:10.1111/j.2517-6161.1979.tb01066.x>.

License GPL-3

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Suggests posterior

NeedsCompilation no

Author Pedro Luiz Ramos [aut, cre, cph]

Maintainer Pedro Luiz Ramos <pedro.ramos@uc.cl>

Repository CRAN

Date/Publication 2026-08-06 13:40:14 UTC

Contents

fitdistrBayes *Objective Bayesian Distribution Fitting*

Description

Fits a univariate parametric distribution under a registered objective Bayesian prior. For built-in models, data-dependent posterior-propriety checks are executed before any numerical integration or simulation. Posterior means, variances, and their Monte Carlo errors are reported only when the corresponding moments have been certified to exist.

Usage

```
fitdistrBayes(x, distr, prior, start = NULL, fixed = NULL,
  iter = 4000L, warmup = floor(iter / 2), thin = 1L,
  chains = 4L, seed = NULL, na.action = c("fail", "omit"),
  control = list(), ...)
```

Arguments

`x`	Numeric vector of observations.
`distr`	A recognized distribution name, matched without regard to case, or a density function evaluated at its first argument.
`prior`	A registered objective-prior name or, for a custom density, a prior function on the documented parameterization.
`start`	Optional named starting values. If omitted for a built-in non-exact route, closed-form classical estimates are used: ordinary moments where available, quantile matching for models without the required moments, and L-moments for Weibull, Frechet, and Lomax. The Exponential-Logarithmic start solves one scalar moment equation. Required for custom densities and ignored with a warning for exact independent posterior simulation.
`fixed`	Optional named list of fixed parameters.
`iter`	Total iterations per chain, including warmup.
`warmup`	Warmup iterations per chain.
`thin`	Positive thinning interval.
`chains`	Number of chains.
`seed`	Optional reproducibility seed passed to `set.seed()` before posterior simulation.
`na.action`	Either `"fail"` or `"omit"`.
`control`	Named list of computational, diagnostic, storage, and custom model controls described in Details.
`...`	Additional fixed arguments forwarded to a custom density and predictive generator.

Details

Recognized distributions are beta, Cauchy, chi-squared, exponential, Exponential-Logarithmic, Frechet, Gamma, geometric, Gumbel, lognormal, logistic, Lomax, Nakagami-m, negative binomial, Normal, Poisson, Rician, Student t, and Weibull. Available priors depend on the model and parameter of interest; unsupported or known-improper model-prior combinations stop with an informative error.

Use `fitdistrBayes_routes()` to obtain the machine-readable catalogue of all enabled combinations, their estimated and required fixed parameters, computational engines, and concise propriety conditions. The fitting function performs the authoritative sample-dependent check.

For the Exponential-Logarithmic model, `"reference"` and `"reference-theta"` select the ordering in which theta is the parameter of interest; `"reference-rate"` selects the rate parameter. The Lomax reference posterior and Nakagami-m MDI posterior are disabled because they are improper. Rician currently has only the proper joint-Jeffreys route.

For numerical routes, the automatic center and any user overrides are recorded in `fit$initialization`. Each MCMC chain is dispersed from this center on the unconstrained scale using `control$init_jitter`. For conditionally exact Gamma, Frechet, Nakagami-m, and Weibull algorithms, only the marginal shape parameter requires initialization; rate or scale is drawn from its exact conditional distribution.

The negative-binomial routes estimate `mu` and require a known positive `fixed$size`. For the Frechet model, the implemented distribution function is $F(x) = \exp\{-scale\, x^{-shape}\}$. Thus, the argument named `scale` is the coefficient in the exponent; the conventional quantile scale is $scale^{1/shape}$.

Sampler controls are `target_accept`, `adapt_interval`, `proposal_scale`, `init_jitter`, `slice_width`, `slice_steps`, and `max_init_tries`. Diagnostic controls are `rhat_threshold`, `ess_threshold`, and `warn_convergence`; both bulk and tail ESS must meet the ESS threshold. The `overall` acceptance column uses post-warmup iterations, with warmup and all-iteration rates stored separately. Entropy evaluation uses `entropy_tol` and the positive integer `entropy_exact_limit`. Set `store_callables = FALSE` to omit the closures needed by `predict()` and `log_lik()`.

For a custom model, `lower` and `upper` define componentwise bounds; the package supplies the corresponding transforms and Jacobian. Log mode is inferred only from an explicit `log` formal. It can be forced with `density_is_log` and `prior_is_log`; an initial ordinary/log consistency check is made when possible. `prior_style` is one of `"auto"`, `"scalar"`, or `"vector"`. A predictive function can be supplied as `rng`, and `rng_validator` may check its returned support. Posterior propriety and moment existence remain the user's responsibility for custom targets.

Value

An object of class `"fitdistrBayes"`. It is a list with the following principal components:

`call` The matched fitting call.

`model` The canonical model name, estimated parameter names, fixed parameters, sample size, and support information.

`prior` The normalized prior identifier, its displayed label and kernel, and the posterior-propriety condition verified before fitting.

`initialization` The automatic or user-supplied initialization rule, its classical estimation method, and the center used to initialize non-exact algorithms.

`engine` The computational algorithm and the requested chain, iteration, warmup, thinning, saved-draw, and seed settings.

`estimates` A named numeric vector of posterior medians, used as the default point estimates.

`summary` A data frame containing posterior means and standard deviations when certified to exist, medians, equal-tail 95 percent credible limits, Monte Carlo standard errors, rank-normalized split/folded $\widehat{R}$, bulk and tail effective sample sizes, and moment-existence indicators.

`moment_status` A data frame recording whether the posterior mean and variance of each parameter are known to exist and explaining the corresponding analytical condition.

`diagnostics` A list containing the overall convergence decision, thresholds, extrema of $\widehat{R}$ and effective sample sizes, acceptance information when applicable, and explanatory messages.

`capabilities` Logical indicators describing whether prediction and pointwise log-likelihood evaluation are available.

`chains` A list of posterior-draw matrices, one matrix per chain.

`draws` A data frame combining all posterior draws. Columns `.chain`, `.iteration`, and `.draw` identify each draw and the remaining columns contain parameter values.

`data`, `omitted`, `control` The analyzed data, the number of omitted missing observations, and the validated computational controls.

Density and random-generation closures are retained internally by default to support `log_lik()` and `predict()`; they are omitted when `control = list(store_callables = FALSE)`.

Built-in models, parameters, and priors

The table below gives the canonical model name, the parameters returned by the fit, and every registered objective-prior label. Distribution names and prior labels are matched without regard to case. Hyphens, spaces, and common aliases such as `"log-normal"`, `"negative binomial"`, `"student-t"`, `"el"`, and `"nakagami"` are normalized internally.

Model	Estimated parameters	Available priors
beta	shape1, shape2	jeffreys, reference
cauchy	location, scale	jeffreys, reference, mdi
chi-squared	df	jeffreys, reference
exponential	rate	jeffreys, reference, mdi
exponential-logarithmic	theta, rate	jeffreys, mdi reference-theta, reference-rate
frechet	shape, scale	jeffreys, reference
gamma	shape, rate	jeffreys, first-rule reference-shape, reference-rate
geometric	prob	jeffreys, reference, mdi
gumbel	location, scale	jeffreys, reference, mdi
lognormal	meanlog, sdlog	jeffreys, reference
logistic	location, scale	jeffreys, reference, mdi
lomax	shape, scale	jeffreys
nakagami-m	shape, spread	jeffreys, reference
negative binomial	mu	jeffreys, reference, mdi
normal	mean, sd	jeffreys, reference, mdi
Poisson	lambda	jeffreys, reference, mdi
rician	noncentrality, scale	jeffreys
t	location, scale, df	jeffreys, reference, mdi

		independence-jeffreys
weibull	shape, scale	jeffreys, reference

For Student t, `independence-jeffreys` estimates `df`; the other three priors require `fixed = list(df = ...)`. The generic label `"reference"` is used only where its parameter ordering is unambiguous. For Gamma and Exponential-Logarithmic models, the parameter of interest is therefore stated in the label. Unsupported combinations, including objective priors known to yield an improper posterior, are rejected before sampling. Run `fitdistrBayes_routes()` for the precise data condition and computational engine attached to every route.

Beta observations must lie strictly between 0 and 1; chi-squared and Rician observations are positive; Poisson and geometric observations are nonnegative integers. Exponential, Gamma, geometric, lognormal, logistic, Normal, Poisson, and Weibull follow the corresponding base-R parameterizations. Geometric observations count failures before the first success. The Gumbel model is the location-scale distribution for maxima. For Lomax, the support is $x >= 0$; for Nakagami-m, `spread` is $E(X^2)$. Negative-binomial `size` is known and supplied in `fixed`, whereas `mu` is estimated.

Posterior propriety and reported moments

Most objective priors are improper as prior measures. A fit is attempted only after the built-in route verifies its sample-dependent posterior-propriety condition. Typical requirements include positivity, a nonconstant sample, a minimum sample size, or multiplicity restrictions for location-scale models. The complete concise conditions are returned by `fitdistrBayes_routes()`, while the authoritative check is performed on the supplied `x` by `fitdistrBayes()`.

Posterior propriety does not imply that every posterior moment exists. Thus, `mean` or `sd` can legitimately be `NA` even when medians, credible intervals, draws, and diagnostics are available. Inspect `fit$moment_status` and its explanatory `note` column before treating an `NA` as a computational failure.

Output and diagnostics

Printing a fit reports the model, prior, computational engine, initialization, propriety decision, posterior summaries, and diagnostic status. The full draws are in `fit$draws`; chain-specific matrices are in `fit$chains`. The summary contains posterior means when they exist, medians, central credible limits, MCSE, rank-normalized split/folded $\widehat{R}$, and bulk and tail effective sample sizes.

Use `summary()`, `coef()`, `confint()`, `as.data.frame()`, and `plot()` for inspection. Use `predict()` for posterior predictive simulation and `log_lik()` for a pointwise log-likelihood matrix. The last two methods are computed on demand and require retained callables.

Complete console tutorial

The installed file `system.file("examples", "tutorial_fitdistrBayes_all_models.R",package = "fitdistrBayes")` contains simulated data for all 19 built-in families, all 50 enabled model-prior routes, comparisons with `MASS::fitdistr()` where directly available, and a final diagnostic table. It prints to the console and does not save analysis results. Set `quick_mode <- TRUE` for a short demonstration or `FALSE` for the longer teaching run.

References

Hosking JRM (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.

Ramos PL, Louzada F, Ramos E, Dey S (2020). "The Frechet distribution: Estimation and application—an overview." *Journal of Statistics and Management Systems*, 23(3), 549–578. doi:10.1080/09720510.2019.1645400.

Ferreira PH, Ramos E, Ramos PL, et al. (2020). "Objective Bayesian analysis for the Lomax distribution." *Statistics & Probability Letters*, 159, 108677. doi:10.1016/j.spl.2019.108677.

Ramos PL, Louzada F, Ramos E (2018). "Posterior properties of the Nakagami-m distribution using noninformative priors and applications in reliability." *IEEE Transactions on Reliability*, 67(1), 105–117. doi:10.1109/TR.2017.2778139.

Moala FA, Achire Quispe E, Ramos PL (2026). "Objective Bayesian inference for the Exponential-Logarithmic distribution." *Journal of Statistical Computation and Simulation*, 96(8), 1773–1801. doi:10.1080/00949655.2025.2608789.

Achire E, Ramos E, Ramos PL (2025). "On the posterior property of the Rician distribution." *Statistics*, 59(1), 167–186. doi:10.1080/02331888.2024.2425688.

Examples

```
# A first exact-posterior example.
set.seed(1)
x <- rexp(30, rate = 2)
fit <- fitdistrBayes(x, "exponential", "jeffreys",
                     iter = 400, warmup = 100, chains = 2, seed = 2)
coef(fit)
confint(fit)
fitdistrBayes_routes("exponential")

# Group different objective priors for the same Gamma data.
set.seed(3)
x_gamma <- rgamma(40, shape = 2.5, rate = 1.3)
gamma_fits <- list(
  Jeffreys = fitdistrBayes(x_gamma, "gamma", "jeffreys",
    iter = 400, warmup = 100, chains = 2, seed = 4,
    control = list(warn_convergence = FALSE)),
  Reference_shape = fitdistrBayes(x_gamma, "gamma", "reference-shape",
    iter = 400, warmup = 100, chains = 2, seed = 5,
    control = list(warn_convergence = FALSE)),
  Reference_rate = fitdistrBayes(x_gamma, "gamma", "reference-rate",
    iter = 400, warmup = 100, chains = 2, seed = 6,
    control = list(warn_convergence = FALSE))
)
lapply(gamma_fits, coef)

# A fixed parameter: negative-binomial size is known and mu is estimated.
set.seed(7)
x_nb <- rnbinom(40, size = 5, mu = 8)
fit_nb <- fitdistrBayes(x_nb, "negative binomial", "reference",
  fixed = list(size = 5), iter = 400, warmup = 100,
```

```
  chains = 2, seed = 8)
coef(fit_nb)

# Student t: fixed df versus estimated df.
set.seed(9)
x_t <- 1 + 2 * rt(50, df = 7)
fit_t_fixed <- fitdistrBayes(x_t, "t", "jeffreys",
  fixed = list(df = 7), iter = 400, warmup = 100,
  chains = 2, seed = 10, control = list(warn_convergence = FALSE))
fit_t_unknown <- fitdistrBayes(x_t, "t", "independence-jeffreys",
  iter = 400, warmup = 100, chains = 2, seed = 11,
  control = list(warn_convergence = FALSE))
coef(fit_t_fixed)
coef(fit_t_unknown)

# One short fit for every built-in family. The installed tutorial additionally
# runs every available prior for each family.
quick <- list(warn_convergence = FALSE)

set.seed(101)
fit_beta <- fitdistrBayes(rbeta(30, 2, 5), "beta", "jeffreys",
  iter = 120, warmup = 40, chains = 2, seed = 102, control = quick)
fit_cauchy <- fitdistrBayes(rcauchy(40, 1, 2), "cauchy", "reference",
  iter = 120, warmup = 40, chains = 2, seed = 103, control = quick)
fit_chisq <- fitdistrBayes(rchisq(30, 6), "chi-squared", "jeffreys",
  iter = 120, warmup = 40, chains = 2, seed = 104, control = quick)
fit_exponential <- fitdistrBayes(rexp(30, 1.5), "exponential", "mdi",
  iter = 120, warmup = 40, chains = 2, seed = 105, control = quick)
fit_gamma <- fitdistrBayes(rgamma(30, 2.5, rate = 1.3),
  "gamma", "reference-shape", iter = 120, warmup = 40,
  chains = 2, seed = 106, control = quick)
fit_geometric <- fitdistrBayes(rgeom(30, 0.35), "geometric", "reference",
  iter = 120, warmup = 40, chains = 2, seed = 107, control = quick)
fit_lognormal <- fitdistrBayes(rlnorm(30, 1, 0.6), "lognormal", "jeffreys",
  iter = 120, warmup = 40, chains = 2, seed = 108, control = quick)
fit_logistic <- fitdistrBayes(rlogis(40, 1, 2), "logistic", "mdi",
  iter = 120, warmup = 40, chains = 2, seed = 109, control = quick)
fit_nb <- fitdistrBayes(rnbinom(30, size = 5, mu = 8),
  "negative binomial", "jeffreys", fixed = list(size = 5),
  iter = 120, warmup = 40, chains = 2, seed = 110, control = quick)
fit_normal <- fitdistrBayes(rnorm(30, 3, 2), "normal", "reference",
  iter = 120, warmup = 40, chains = 2, seed = 111, control = quick)
fit_poisson <- fitdistrBayes(rpois(30, 4), "Poisson", "mdi",
  iter = 120, warmup = 40, chains = 2, seed = 112, control = quick)
fit_t <- fitdistrBayes(rt(40, 7), "t", "jeffreys", fixed = list(df = 7),
  iter = 120, warmup = 40, chains = 2, seed = 113, control = quick)
fit_weibull <- fitdistrBayes(rweibull(30, 1.6, 2), "weibull", "reference",
  iter = 120, warmup = 40, chains = 2, seed = 114, control = quick)

x_gumbel <- 1 - 2 * log(-log(runif(40)))
fit_gumbel <- fitdistrBayes(x_gumbel, "gumbel", "reference",
  iter = 120, warmup = 40, chains = 2, seed = 115, control = quick)
```

```
x_frechet <- (4 / rexp(40))^(1 / 2.5)
fit_frechet <- fitdistrBayes(x_frechet, "frechet", "jeffreys",
  iter = 120, warmup = 40, chains = 2, seed = 116, control = quick)

x_lomax <- 2 * expm1(-log(runif(40)) / 3)
fit_lomax <- fitdistrBayes(x_lomax, "lomax", "jeffreys",
  iter = 120, warmup = 40, chains = 2, seed = 117, control = quick)

x_nakagami <- sqrt(rgamma(40, 2.5, rate = 2.5 / 4))
fit_nakagami <- fitdistrBayes(x_nakagami, "nakagami-m", "reference",
  iter = 120, warmup = 40, chains = 2, seed = 118, control = quick)

theta_el <- 0.4
rate_el <- 1.3
u_el <- runif(40)
x_el <- -(log(-expm1((1 - u_el) * log(theta_el))) -
  log1p(-theta_el)) / rate_el
fit_el <- fitdistrBayes(x_el, "exponential-logarithmic", "reference-rate",
  iter = 120, warmup = 40, chains = 2, seed = 119, control = quick)

x_rician <- sqrt(rnorm(40, 5, 2)^2 + rnorm(40, 0, 2)^2)
fit_rician <- fitdistrBayes(x_rician, "rician", "jeffreys",
  iter = 120, warmup = 40, chains = 2, seed = 120, control = quick)

tutorial <- system.file(
  "examples", "tutorial_fitdistrBayes_all_models.R",
  package = "fitdistrBayes"
)
tutorial
if (interactive()) file.show(tutorial)
```

`fitdistrBayes-methods` *Methods for Objective Bayesian Distribution Fits*

Description

Methods for inspecting posterior summaries and draws, extracting posterior medians and credible intervals, producing standard diagnostic plots, simulating posterior predictive observations, and computing pointwise log-likelihood matrices. Prediction and pointwise log-likelihood are computed on demand. They are unavailable when the fitted object was created with `control = list(store_callables = FALSE)`; inspect `object$capabilities` before calling them in reusable workflows.

Usage

```
## S3 method for class 'fitdistrBayes'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
## S3 method for class 'fitdistrBayes'
summary(object, ...)
```

```
## S3 method for class 'summary.fitdistrBayes'
print(x,
  digits = max(3L, getOption("digits") - 3L), ...)
## S3 method for class 'fitdistrBayes'
coef(object, ...)
## S3 method for class 'fitdistrBayes'
confint(object, parm = object$model$parameters,
  level = 0.95, ...)
## S3 method for class 'fitdistrBayes'
as.data.frame(x, row.names = NULL,
  optional = FALSE, ...)
## S3 method for class 'fitdistrBayes'
plot(x,
  type = c("trace", "density", "acf", "pairs"),
  pars = x$model$parameters, ...)
## S3 method for class 'fitdistrBayes'
predict(object, draws = 1000L, size = 1L,
  seed = NULL, ...)
log_lik(object, ...)
## S3 method for class 'fitdistrBayes'
log_lik(object, draws = NULL, seed = NULL, ...)
```

Arguments

`x`, `object`	A fitted "`fitdistrBayes`" object.
`digits`	Number of printed significant digits.
`parm`	Parameter names for interval extraction.
`level`	Credible level.
`row.names`, `optional`	Arguments for data-frame conversion.
`type`	Diagnostic plot type.
`pars`	Optional subset of parameters.
`draws`	Number of posterior or predictive draws.
`size`	Number of observations in each predictive data set.
`seed`	Optional reproducibility seed passed to `set.seed()` before sampling posterior draws or posterior predictive observations.
`...`	Additional arguments.

Value

The returned value depends on the method:

- `print.fitdistrBayes()` returns `x`, invisibly, and prints the model, prior, computational engine, posterior summaries, and diagnostic status.

- `summary.fitdistrBayes()` returns an object of class `"summary.fitdistrBayes"`. It is a list containing the original call; model, prior, initialization, and computational-engine records; the posterior summary table; the posterior-moment audit; convergence diagnostics; and the available prediction and log-likelihood capabilities.
- `print.summary.fitdistrBayes()` returns `x`, invisibly, and prints the contents of the summary object.
- `coef.fitdistrBayes()` returns a named numeric vector containing the posterior median of each estimated parameter. These medians are the default point estimates used by the package.
- `confint.fitdistrBayes()` returns a numeric matrix with one row per requested parameter and two columns containing the lower and upper equal-tail posterior credible limits.
- `as.data.frame.fitdistrBayes()` returns a data frame of the combined posterior draws. The columns `.chain`, `.iteration`, and `.draw` identify the simulation draw; the remaining columns contain parameter values.
- `plot.fitdistrBayes()` returns `x`, invisibly, and produces the requested diagnostic plot as a side effect.
- `predict.fitdistrBayes()` returns posterior predictive observations. It is a numeric vector of length `draws` when `size = 1`, and otherwise a numeric matrix with `draws` rows and `size` columns. Each row is one replicated data set generated after selecting a posterior draw.
- `log_lik()` and `log_lik.fitdistrBayes()` return a numeric matrix whose rows correspond to posterior draws and whose columns correspond to observations. Each entry is that observation's log-likelihood contribution at the selected posterior draw.

`fitdistrBayes_routes` *List the Built-In Objective Bayesian Fitting Routes*

Description

Returns the machine-readable catalogue used to document and test the available model–prior combinations. The condition column is a concise summary; the fitting function remains the authoritative executable check for the observed sample.

Usage

```
fitdistrBayes_routes(model = NULL)
```

Arguments

`model`	Optional recognized distribution name. If supplied, only the routes for that distribution are returned.

Value

A data frame with one row for every enabled model–prior combination and the following character columns:

`model` Canonical distribution name.

`prior` Accepted objective-prior label.

`parameters` Comma-separated parameters estimated by the route.

`fixed` Parameters that the user must supply in `fixed`, or an empty string when none are required.

`engine` Posterior simulation or numerical-integration strategy used by the route.

`condition` Concise sample condition under which the posterior is proper. The fitting function performs the authoritative check on the supplied observations.

Examples

```
fitdistrBayes_routes()
fitdistrBayes_routes("gamma")
fitdistrBayes_routes("t")
```

Index