

Package ‘fitdistrBayes’

October 1, 2026

Type Package

Title Objective Bayesian Distribution Fitting

Version 0.5.1

Description Fits common univariate distributions using registered objective Bayesian priors, including Jeffreys, reference, and maximal data information priors, and supports user-defined distributions and priors through an extensible model specification. Model-specific posterior propriety and moment conditions are checked before computation when registered or supplied. Exact simulation, marginalization, slice sampling, adaptive Metropolis, and user-supplied posterior samplers share a common interface for summaries, diagnostics, prediction, and pointwise log-likelihood evaluation. A separate interface fits independently right-censored observations using the registered complete-data priors, observed-data likelihood sampling or data augmentation, with sufficient posterior-propriety checks. Optional post-processing provides WAIC, PSIS-LOO, and DIC for observed-data likelihoods. The reference-prior framework follows Bernardo (1979)
<[doi:10.1111/j.2517-6161.1979.tb01066.x](https://doi.org/10.1111/j.2517-6161.1979.tb01066.x)>.

License GPL-3

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Suggests posterior, loo (>= 2.7.0)

NeedsCompilation no

Author Pedro Luiz Ramos [aut, cre, cph]

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

Repository CRAN

Date/Publication 2026-09-30 22:20:02 UTC

Contents

criteria 2

fitcensBayes	5
fitcensBayes-methods	9
fitcensBayes_models	12
fitdistrBayes	13
fitdistrBayes-methods	20
fitdistrBayes_model	22
fitdistrBayes_routes	25

Index	27
--------------	-----------

criteria	<i>Optional Bayesian Information Criteria and Model Comparison</i>
----------	--

Description

Computes WAIC, Pareto-smoothed importance-sampling leave-one-out cross-validation (PSIS-LOO), and DIC from existing posterior draws. Both complete and independently right-censored fits use the pointwise observed-data likelihood. No sampling or automatic refitting is performed.

Usage

```
criteria(object, methods = c("waic", "looic", "dic"), cores = 1L)
WAIC(object)
LOOIC(object, cores = 1L)
DIC(object)
compare_models(..., criterion = "looic", cores = 1L)
## S3 method for class 'fitdistrBayes_criteria'
print(x, digits = 4L, ...)
## S3 method for class 'fitdistrBayes_criteria'
as.data.frame(x, row.names = NULL,
  optional = FALSE, ...)
## S3 method for class 'fitdistrBayes_comparison'
print(x, digits = 4L, ...)
```

Arguments

object	A fitted fitdistrBayes or fitcensBayes object retaining its log-likelihood callback (store_callables = TRUE, the default).
methods	Character vector selecting "waic", "looic", and/or "dic". Names ignore case; "loo" aliases "looic".
cores	Number of PSIS-LOO computation cores: one (default) or two. This does not change the number of sampling chains.
criterion	One criterion used to compare all supplied models.
...	For compare_models, two or more preferably named fits or previously computed fitdistrBayes_criteria objects. In methods, additional arguments passed to the corresponding data-frame method.
x	A criteria or comparison result, as appropriate.

digits Printing precision.
row.names, optional
 Passed to as.data.frame.

Details

The fitting functions default to `criteria = FALSE`. They do not evaluate the extra pointwise likelihood matrix, load **loo**, or calculate these criteria unless requested. `criteria = TRUE` adds post-processing after the chains are complete; a vector such as `criteria = c("waic", "dic")` selects a subset. The chains and random-number state are unaffected by the built-in criterion computations. Existing convergence diagnostics are separate and remain available regardless of this option.

Post-processing uses all saved draws and temporarily requires a matrix with one row per draw and one column per observation, approximately $8Sn$ bytes, plus working copies. A combined call evaluates this matrix only once; the result does not retain it. Each on-demand call recomputes its criteria, so use the combined call for efficiency. PSIS-LOO alone requires the suggested package **loo**; WAIC and DIC use base R. An automatic request for LOO without **loo**, or for any criterion without stored callbacks, fails before sampling starts. If a numerical error occurs in optional post-processing after sampling, the fitting functions retain the fitted object and chains, issue a warning, and mark the requested criteria unavailable. The error is recorded in `fit$criteria$diagnostics$computation_error`. Direct calls to `criteria()` still report invalid likelihood matrices as errors. If finite log-likelihood inputs lead to numerically unrepresentable pointwise WAIC components or aggregate scores, the affected criterion is returned as NA, with a warning and explanation, and is never labelled reliable.

For exact observations, $l_{si} = \log f(x_i | \theta_s)$. For censored observations, $l_{si} = \log S(x_i | \theta_s)$ instead. The censoring mechanism is assumed independent and identical across compared models; its parameter-free factors are omitted. Imputed lifetimes never replace the observed likelihood in these criteria, including for augmentation fits. For discrete data, censoring means the strict event $T > x_i$.

WAIC uses

$$p_{WAIC} = \sum_i \text{Var}_s(l_{si}), \quad WAIC = -2 \sum_i \left[\log \left(S^{-1} \sum_s e^{l_{si}} \right) - \text{Var}_s(l_{si}) \right].$$

The variance uses denominator $S - 1$. Centering and scaling avoid intermediate overflow without requiring extended floating-point precision. Pointwise variances above 0.4 produce a warning. PSIS-LOO uses `loo : : loo`, with chain-aware relative effective sample sizes (one for independent exact draws). Its result includes Pareto-k and effective-sample-size diagnostics. The diagnostic threshold is $\min(0.7, 1 - 1/\log_{10}(S))$. Large Pareto-k values indicate unreliable importance sampling; exact refits and moment matching are not performed. $LOOIC = -2 \text{elpd}_{loo}$.

For a positive continuous distribution, a record `x = 0`, `status = 0` is the sure event $T > 0$. Its pointwise log predictive value, penalty, and Monte Carlo error are exactly zero. Such rows are handled analytically and restored to the full observation order after PSIS on the informative rows. Their Pareto-k entry is `-Inf` as a sentinel, not an estimated tail shape. Their indices are stored in `loo_zero_information_observations` and in an attribute of the extended `psis_loo` result. This exception does not apply to discrete censoring at zero.

With improper objective priors, a proper full-data posterior does not imply proper leave-one-out training posteriors. The registered sufficient conditions are checked for every training set before

PSIS-LOO. Under censoring these are applied to the exact-event subset; the exponential case is checked analytically. If any training posterior is not certified, LOOIC is returned as NA with an explanation, not an unchecked numerical score. These sufficient checks can be conservative. LOOIC is not certified automatically for models or priors defined by the user; WAIC and DIC rely on the user's correctly normalized densities, proper posterior, and moment declarations.

DIC uses $D(\theta) = -2 \sum_i \log p(y_i | \theta)$, $\bar{D} = E[D(\theta) | y]$, $p_D = \bar{D} - D(E[\theta | y])$, and $DIC = \bar{D} + p_D$. It is parameterization-dependent and additionally assumes a finite expected deviance; the moment audit checks parameter means, not this latter expectation. All parameter means must be certified finite. Otherwise DIC is NA, including the registered Weibull scale and unrestricted Student-t degrees of freedom. Medians are not substituted. Even when parameter means exist, the likelihood at their joint mean can be undefined, for example in a disconnected parameter space. In this case DIC is unavailable, but other computable criteria and the chains are retained. Negative p_D is flagged. DIC has no reported predictive standard error.

`compare_models` requires identical retained data, order, missing-row indices, censoring indicators, and likelihood measures; continuous and discrete likelihoods cannot be mixed. Keep measurement scales and censoring definitions identical. For custom models, automatic identity checks cannot verify the common likelihood measure; this remains the user's responsibility. Lower criteria are better; differences do not represent posterior model probabilities. For WAIC/LOOIC, `delta` is relative to the lowest score and `se_delta` is $\sqrt{n \text{Var}_i(d_i)}$ using paired pointwise IC differences. This is uncertainty across observations, not MCMC error. With one observation it is undefined. Diagnostic failures are retained and warned about; a ranking is not evidence that those failures are harmless.

Value

`criteria`, `WAIC`, `LOOIC`, and `DIC` return a list of class `"fitdistrBayes_criteria"`, with:

`estimates` Data frame with criterion name, IC estimate, its observation-level se, effective parameter count `p_eff`, `elpd`, `available`, `reliable`, and `explanatory reason`. Unavailable criteria and undefined quantities are NA. The reliability flag only means the implemented diagnostic checks passed, not a guarantee. Scores from custom fits without a posterior-propriety declaration are not labelled reliable.

`details` Pointwise WAIC components (unrepresentable components are NA, with observation indices in `waic$unrepresentable_observations`); the native `psis_loo` result (extended with analytic zero-information rows when necessary), relative efficiencies and Pareto diagnostics; and DIC components with the posterior-mean parameter vector, as applicable.

`diagnostics` Sampling diagnostics, per-observation LOO propriety certification, and captured warning messages.

`model`, `prior`, `nobs`, `ndraws` Model/prior identifiers and numbers of observations and saved posterior draws.

`observation` Retained data, status, omitted-row indices, and likelihood measure used to prevent incompatible comparisons.

`compare_models` returns class `"fitdistrBayes_comparison"`, a list with sorted data frame `table`, named criterion results, `criterion`, and the lowest-score reference label. The table contains model labels, estimates, differences, paired difference SEs, individual SEs, effective parameter counts, and diagnostic reliability flags. The print methods return their argument invisibly after printing. The data-frame method returns the estimates data frame.

References

- Gelman A, Hwang J, Vehtari A (2014). Understanding predictive information criteria for Bayesian models. *Statistics and Computing*, 24, 997–1016. doi:10.1007/s1122201394162.
- Vehtari A, Gelman A, Gabry J (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, 27, 1413–1432. doi:10.1007/s11222016-96964.
- Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society B*, 64, 583–639. doi:10.1111/1467-9868.00353.

See Also

[fitdistrBayes](#), [fitcensBayes](#), [log_lik](#).

Examples

```
set.seed(62)
x <- rexp(40, rate = 0.8)
fit <- fitdistrBayes(x, "exponential", "reference", seed = 63,
  iter = 1000, warmup = 500, chains = 2, criteria = c("waic", "dic"))
fit$criteria
DIC(fit)

status <- as.integer(x <= 2)
observed <- pmin(x, 2)
cens <- fitcensBayes(observed, status, "exponential", "reference",
  seed = 64, iter = 1000, warmup = 500, chains = 2)
WAIC(cens)

if (requireNamespace("loo", quietly = TRUE)) {
  LOOIC(cens)
  fit_mdi <- fitdistrBayes(x, "exponential", "mdi", seed = 65,
    iter = 1000, warmup = 500, chains = 2)
  compare_models(Reference = fit, MDI = fit_mdi, criterion = "looic")
}
```

fitcensBayes

Objective Bayesian Fitting with Independent Right Censoring

Description

Fits registered univariate distributions to exact and independently right-censored observations using the objective priors of the complete-data model. This is separate from the complete-data interface [fitdistrBayes](#).

Usage

```
fitcensBayes(x, status, distr, prior = NULL, start = NULL, fixed = NULL,
  iter = 4000L, warmup = floor(iter / 2), thin = 1L,
  chains = 4L, seed = NULL,
  method = c("auto", "direct", "augmentation"),
  na.action = c("fail", "omit"), control = list(), ...,
  criteria = FALSE)
```

Arguments

<code>x</code>	Nonempty numeric vector of observed values: the event time when <code>status = 1</code> , or the lower censoring limit when <code>status = 0</code> . Values must satisfy the support of the selected model.
<code>status</code>	Numeric or logical vector of the same length as <code>x</code> . One (TRUE) denotes an exact event; zero (FALSE) denotes the strict event $T > x$. For a discrete model, a record meaning $T \geq 3$ must be entered as <code>x = 2, status = 0</code> .
<code>distr</code>	One of the 20 registered distribution names, ignoring case. See fitcensBayes_models and the Details section. A density function or model specification defined by the user is not supported by this censored interface.
<code>prior</code>	A registered prior name. It must be supplied explicitly. See fitcensBayes_models for the available names by model.
<code>start</code>	Optional named list or numeric vector of initial parameter values on the original scale. Missing starting values use the classical initialization of the complete-data model applied to the exact events. Starts initialize computation; they do not determine propriety.
<code>fixed</code>	Optional named list of fixed parameters. The negative binomial requires positive size. The usual Student-t routes require positive df; its "independence-jeffreys" route estimates df.
<code>iter</code>	Total iterations per chain, including warmup; at least 20.
<code>warmup</code>	Number of initial iterations discarded; nonnegative and smaller than <code>iter</code> .
<code>thin</code>	Retain every thin-th iteration after warmup. At least ten draws per chain must remain.
<code>chains</code>	Number of independently initialized chains; at least two.
<code>seed</code>	Optional nonnegative integer seed passed to <code>set.seed()</code> before posterior simulation. The RNG state advances normally.
<code>method</code>	Sampling method. "auto" uses the original sampler without censoring and the observed likelihood with censoring. "direct" uses the observed likelihood (not necessarily independent simulation); "augmentation" samples latent censored lifetimes and updates the parameters conditionally.
<code>na.action</code>	Whether to reject missing values or omit the entire <code>x, status</code> pair. Infinite values are always rejected.
<code>control</code>	Named list of controls, described below.
<code>...</code>	Reserved for future extensions. Unused arguments are errors.

criteria FALSE by default: no model-comparison computation. TRUE requests WAIC, PSIS-LOO, and DIC after sampling; a character vector selects "waic", "looic", or "dic". Must be named in full. PSIS-LOO requires the optional **loo** package. See [criteria](#).

Details

Censoring is assumed independent and non-informative about the lifetime parameters. The observed likelihood is proportional to

$$L(\theta) = \prod_{i:\delta_i=1} f(x_i | \theta) \prod_{i:\delta_i=0} S(x_i | \theta),$$

where $S(t | \theta) = P(T > t | \theta)$. The censoring distribution is not estimated. Only right censoring is implemented.

The supported distributions are beta, Cauchy, chi-squared, exponential, gamma, geometric, log-normal, logistic, negative binomial, normal, Poisson, Student-t, Weibull, Frechet, Gumbel, Lomax, Nakagami-m, exponential-logarithmic, Rician, and weighted Lindley. Their parameterizations and aliases are those of [fitdistrBayes](#). There are 56 registered model-prior routes. Not every prior is available for every model.

Interpretation of the priors. The function reuses Jeffreys, reference, and maximal data information priors derived for the complete-data sampling model, including the registered first-rule and parameter-ordered variants. It does not claim that these are the corresponding objective priors derived from the information matrix of a particular censoring design.

Posterior propriety and moments. Since every survival factor is at most one, a proper posterior based on the exact-event subset implies a proper censored posterior. The complete-data sufficient conditions are therefore applied to this subset, not to the total or imputed sample size. A finite nonnegative posterior moment also transfers by the same bound. Failure of this sufficient criterion is not claimed to prove impropriety; the function stops without sampling. Uncertified moments are labelled NA and are not estimated from a finite chain. Known infinite moments are distinguished by FALSE. In particular, positive moments of the Weibull scale diverge under the registered priors, even though medians and credible intervals exist.

For the exponential model, let $m = \sum_i \delta_i$ and $A = \sum_i x_i$. The rate posterior is $\text{Gamma}(m, A)$ under Jeffreys/reference and $\text{Gamma}(m + 2, A)$ under MDI, using the shape-rate convention. It requires $A > 0$ finite and positive shape. Thus all-censored exponential data are accepted under MDI; other all-censored routes are not certified. For geometric Jeffreys/reference, the probability posterior is $\text{Beta}(m, \sum_i x_i + n_c + 1/2)$, where n_c is the number censored and counts start at zero.

Computation. Observed-likelihood sampling uses exact simulation for the analytic cases, marginal slice sampling with conditional scale simulation for Weibull, and preconditioned slice sampling otherwise. Augmentation generates $T_i | T_i > x_i, \theta$ and uses conditional parameter updates; normal and lognormal models have exact blocked updates. Initial values use classical estimates from the exact events, optional optimization of the observed log posterior, and dispersed chain starts. The geometry is fixed during retained simulation. Imputed data are never used to certify propriety. Difficult tails and heavy censoring can require longer runs; inspect diagnostics rather than relying on default iteration counts.

Controls.

`rhat_threshold`, `ess_threshold` Diagnostic targets, default 1.01 and 400 for rank-normalized split/folded R-hat and bulk/tail ESS.

`slice_width`, `slice_steps`, `max_shrink` Slice width (1), maximum stepping-out budget (40), and shrinkage budget (1000).

`init_jitter`, `max_init_tries` Initial dispersion (2) and maximum attempts to find a finite initial target (100).

`optimize_start` Use observed-posterior optimization for automatic starts (default TRUE). A supplied start keeps its center.

`warn_convergence` Warn if MCMC diagnostic targets fail (default TRUE). Turning off the warning does not alter diagnostics.

`store_callables` Store callbacks for prediction and log-likelihood extraction (default TRUE).

`entropy_tol`, `entropy_exact_limit` Tolerance ($1e-10$) and exact-sum limit (2000) for applicable MDI calculations.

Unknown controls are errors. No new mandatory packages are required.

Value

A list of class "fitcensBayes", separate from "fitdistrBayes":

`call`, `model`, `prior` The matched call; model name, parameter names, fixed values and sample size; and prior label, kernel, complete-data origin and posterior propriety justification.

`censoring` Censoring type, exact/censored counts, censored fraction and the non-informative censoring assumption.

`initialization`, `engine` Starting procedure and values when applicable; selected algorithm, iteration settings and seed.

`estimates` Named numeric vector of posterior medians.

`summary` Data frame with one row per parameter: justified posterior mean and standard deviation, median, equal-tail 95 percent limits, R-hat, ESS, mean Monte Carlo standard error, and moment status.

`moment_status` Data frame recording whether each mean and variance is finite (TRUE), infinite (FALSE), or not certified (NA), with explanations. Uncertified or infinite means, standard deviations and associated MCSEs are suppressed in the summary.

`diagnostics` Diagnostic targets, extrema, explanatory messages and a converged flag. This flag denotes meeting the diagnostic targets, not a mathematical proof of convergence. `exact_or_independent` identifies independent posterior simulation, for which MCMC convergence checks do not apply.

`chains` List of numeric matrices, one per chain, with retained iterations in rows and model parameters in columns.

`draws` Combined data frame with columns `.chain`, `.iteration`, `.draw`, and the parameter values.

`data`, `status`, `omitted` Observed values and indicators after missing-pair removal, and omitted original indices.

`control` Validated computational settings.

`criteria` Present only when requested: a "fitdistrBayes_criteria" object based on the observed likelihood, not the density of imputed lifetimes. Does not change the chains.

When `store_callables = TRUE`, internal callbacks are stored for the observed log likelihood, predictive sampling, survival and imputation. See [fitcensBayes-methods](#) for extraction and prediction methods.

See Also

[fitdistrBayes](#), [fitcensBayes_models](#), [fitcensBayes-methods](#).

Examples

```
set.seed(42)
lifetime <- rexp(80, rate = 0.7)
censoring <- rexp(80, rate = 0.3)
x <- pmin(lifetime, censoring)
status <- as.integer(lifetime <= censoring)
fit <- fitcensBayes(x, status, "exponential", "reference", seed = 43)
fit
confint(fit)

# Discrete censoring means strictly greater than the reported count.
fit_geom <- fitcensBayes(c(0, 1, 2, 3, 4, 5), c(1, 1, 0, 1, 0, 0),
                        "geometric", "jeffreys", seed = 44)

coef(fit_geom)

# Inspect the installed tutorial without executing long MCMC runs.
system.file("examples", "tutorial_fitcensBayes.R", package = "fitdistrBayes")
```

fitcensBayes-methods *Methods for Objective Bayesian Fits with Right Censoring*

Description

Summarizes posterior draws, assesses simulation diagnostics, generates uncensored future observations, computes survival probabilities, imputes censored lifetimes, and extracts the observed-data log likelihood.

Usage

```
## S3 method for class 'fitcensBayes'
print(x, digits = 4L, ...)
## S3 method for class 'fitcensBayes'
summary(object, ...)
## S3 method for class 'summary.fitcensBayes'
print(x, digits = 4L, ...)
## S3 method for class 'fitcensBayes'
coef(object, ...)
## S3 method for class 'fitcensBayes'
confint(object,
  parm = object$model$parameters, level = 0.95, ...)
```

```
## S3 method for class 'fitcensBayes'
as.data.frame(x,
  row.names = NULL, optional = FALSE, ...)
## S3 method for class 'fitcensBayes'
plot(x,
  type = c("trace", "density", "acf", "pairs"),
  pars = x$model$parameters, ...)
## S3 method for class 'fitcensBayes'
predict(object,
  type = c("response", "survival", "impute"),
  times = NULL, draws = 1000L, size = 1L, seed = NULL, ...)
## S3 method for class 'fitcensBayes'
log_lik(object, draws = NULL, seed = NULL, ...)
log_lik_cens(object, draws = NULL, seed = NULL)
```

Arguments

<code>x, object</code>	A "fitcensBayes" object, or for its summary print method an object of class "summary.fitcensBayes".
<code>digits</code>	Number of significant digits printed.
<code>parm</code>	Parameter names for credible intervals.
<code>level</code>	Credible probability, strictly between zero and one.
<code>row.names, optional</code>	Compatibility arguments, currently ignored. Stored draw identifiers are retained.
<code>type</code>	Plot or prediction type. See Details for prediction.
<code>pars</code>	Parameter names to plot. Pairs plots need at least two.
<code>times</code>	Finite numeric evaluation times for survival prediction. Must be NULL for other prediction types.
<code>draws</code>	Number of posterior samples to select. For <code>log_lik</code> , NULL uses all stored draws. Prediction samples posterior rows without replacement unless more rows are requested than are stored.
<code>size</code>	Number of future observations per draw for response prediction. A positive integer; irrelevant to the other prediction types.
<code>seed</code>	Optional seed passed to <code>set.seed()</code> before simulation or posterior row selection. RNG state advances normally.
<code>...</code>	Additional graphical arguments for plot. Unused arguments to prediction or log-likelihood extraction are errors; other methods accept them for generic compatibility.

Details

The methods dispatch separately from those for uncensored fits. `predict(type = "response")` generates future uncensored values from the lifetime distribution, not an observed value/indicator pair. `predict(type = "survival")` returns $P(T > t \mid \theta)$ at each requested time for sampled

parameter values. `predict(type = "impute")` samples latent lifetimes conditionally on each censored record. Every imputed value is strictly above its observed limit, including in discrete models. Imputation does not change the fit. No imputation is possible without censored records.

`log_lik` dispatches to the censored method. `log_lik_cens` is an explicit convenience wrapper with the same result. Each observation contributes either its log density or its log survival, according to the observed indicator, not a completed-data log density. The censoring mechanism's own likelihood is excluded under the non-informative censoring assumption.

Prediction and likelihood extraction require callbacks stored by `control = list(store_callables = TRUE)`, the fitting default.

Value

`print`, `print.summary` The supplied object, invisibly, after printing model, censoring, posterior and diagnostic information. The summary print method also displays the moment audit.

`summary` A list of class "summary.fitcensBayes" containing model, prior, summary, moment_status, diagnostics, initialization, engine, and censoring from the fit.

`coef` A named numeric vector of posterior medians.

`confint` A numeric matrix with one row per requested parameter and two columns of equal-tail posterior credible limits. These are credible intervals, not frequentist confidence intervals.

`as.data.frame` A data frame of retained posterior draws with chain, iteration and draw identifiers and parameter columns.

`plot` The original fitted object, invisibly. Produces trace, density, autocorrelation or pairs plots as a side effect.

`predict` Always a numeric matrix with draws rows. For response prediction it has size columns of future values; for survival it has `length(times)` columns of conditional survival probabilities; for imputation it has one column per censored individual, in their order in the retained data. Rows incorporate posterior parameter uncertainty.

`log_lik`, `log_lik_cens` A numeric matrix with selected posterior draws in rows and retained observations in columns. Entries are observed-data log-likelihood contributions.

See Also

[fitcensBayes](#), [fitcensBayes_models](#)

Examples

```
fit <- fitcensBayes(c(0.3, 0.7, 1.2, 1.5, 2),
                  c(1, 1, 0, 1, 0), "exponential", "jeffreys",
                  seed = 15)

summary(fit)
coef(fit)
confint(fit, level = 0.9)
head(as.data.frame(fit))
predict(fit, type = "survival", times = c(1, 2), draws = 3, seed = 16)
predict(fit, type = "impute", draws = 3, seed = 17)
predict(fit, type = "response", draws = 3, size = 2, seed = 18)
log_lik(fit, draws = 3, seed = 19)
plot(fit, type = "density")
```

fitcensBayes_models *Catalogue of Registered Censored Model-Prior Routes*

Description

Lists distributions and complete-data objective priors supported by the separate right-censoring interface, including sufficient propriety conditions and sampling options.

Usage

```
fitcensBayes_models()
```

Details

Prior formulas, parameterizations, and aliases are inherited from [fitdistrBayes](#) and [fitdistrBayes_routes](#). The catalogue is not an assertion that objective priors are invariant to the censoring design. Conditions are sufficient and conservative. Exponential MDI additionally permits zero exact events when the total observed time is positive and finite; the resulting Gamma posterior has shape two. The function documentation describes this analytic case.

Value

A data frame with 56 rows (one per registered model-prior route) for 20 distributions. Its character columns are:

`model` Canonical distribution name.

`prior` Registered prior name accepted by the fitting function.

`parameters` Names of parameters that are estimated.

`required_fixed` Required fixed parameters, if any.

`complete_data_engine` Sampler registered for complete data; not a claim that the same algorithm is used under censoring.

`exact_subset_sufficient_condition` Sufficient condition applied to the exact events alone. Failure means not certified, not necessarily improper. See the analytic Exponential exception below.

`censored_methods` Available observed-likelihood and data-augmentation methods.

`censoring` Right-censoring status convention.

`prior_origin` The complete-data sampling model.

See Also

[fitcensBayes](#), [fitdistrBayes_routes](#)

Examples

```
routes <- fitdensBayes_models()
routes[routes$model == "gamma", c("model", "prior", "parameters")]
length(unique(routes$model))
nrow(routes)
```

fitdistrBayes

Objective Bayesian Distribution Fitting

Description

Fits a univariate parametric distribution under a registered objective Bayesian prior or a user-supplied model and 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 = NULL, start = NULL, fixed = NULL,
  iter = 4000L, warmup = floor(iter / 2), thin = 1L,
  chains = 4L, seed = NULL, na.action = c("fail", "omit"),
  control = list(), ..., criteria = FALSE)
```

Arguments

x	Numeric vector of observations.
distr	A recognized distribution name, matched without regard to case, a density function evaluated at its first argument, or an object made by <code>fitdistrBayes_model()</code> .
prior	A registered objective-prior name or, for a custom density function, a prior function on the documented parameterization. Omit this argument when <code>distr</code> is a <code>fitdistrBayes_model</code> object.
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, and the weighted Lindley start uses its closed-form likelihood estimator with a numerical maximum-likelihood fallback. 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 <code>set.seed()</code> 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. Unused arguments are errors for built-in distributions.
criteria	Optional post-processing: FALSE (default) does no information-criterion computation; TRUE requests WAIC, PSIS-LOO, and DIC. A character vector selects "waic", "looic", or "dic". Must be named in full. PSIS-LOO requires the optional loo package. See criteria for assumptions and diagnostics.

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, Weibull, and weighted Lindley. 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 weighted Lindley, "reference" is the one-group reference prior and is equal to the Fisher-information Jeffreys prior; "reference-lambda" and "reference-phi" select the exact ordered reference prior for the stated scalar target. Its MDI posterior is improper and therefore disabled.

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 density function, 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. Use `fitdistrBayes_model()` for an explicit reusable specification that can additionally select the generic slice sampler or a user-supplied posterior sampler and carry executable support, propriety, and moment information.

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 $\hat{R}$, 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 $\hat{R}$ and effective sample sizes, acceptance information when applicable, and explanatory messages. A non-moving MCMC chain sets $\hat{R}$ to Inf and ESS to zero for the affected parameter; it cannot certify convergence. Independent exact draws do not require MCMC convergence diagnostics.

`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 original indices of omitted missing observations, and the validated computational controls.

`criteria` Present only when requested: an object of class "fitdistrBayes_criteria". Computed after sampling, without changing the chains. Criteria can also be computed later from the fit.

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
weighted lindley	lambda, phi	jeffreys, reference, first-rule independence-jeffreys reference-lambda, reference-phi

For Student t, independence-jeffreys estimates df and requires at least two pairwise distinct observations. Tied observations produce an improper posterior when df is unrestricted and are rejected before sampling. The other three priors require `fixed = list(df = ...)`; their propriety checks account for the largest number of repeated observations and the fixed value of 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 $x \geq 0$; for Nakagami-m,

spread is $E(X^2)$. Negative-binomial size is known and supplied in fixed, whereas mu is estimated. Weighted Lindley observations are strictly positive, and its density is

$$f(x \mid \lambda, \phi) = \frac{\lambda^{\phi+1}}{(\lambda + \phi)\Gamma(\phi)} x^{\phi-1} (1+x)e^{-\lambda x}.$$

Its MCMC is carried out in the exactly Fisher-orthogonal mean/shape parameterization while the returned parameters remain lambda and phi.

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 $\hat{R}$, 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 20 built-in families, all 56 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.


```

# 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)

lambda_wl <- 2.5
phi_wl <- 0.8
component_wl <- runif(40) < lambda_wl / (lambda_wl + phi_wl)
x_wl <- rgamma(40, shape = phi_wl + as.numeric(!component_wl),
  rate = lambda_wl)
fit_wl <- fitdistrBayes(x_wl, "weighted lindley", "reference-lambda",
  iter = 120, warmup = 40, chains = 2, seed = 121, 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. The log_lik generic also accepts "fitcensBayes" objects; see fitcensBayes-methods for their separate methods.
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.

<code>fitdistrBayes_model</code>	<i>Define a User-Supplied Bayesian Distribution Model</i>
----------------------------------	---

Description

Creates a non-stateful model specification for distributions or priors not in the built-in objective-prior catalogue. The same `fitdistrBayes()` output and methods are used for built-in and user-supplied models.

Usage

```
fitdistrBayes_model(density, prior, start, name = "user-defined",
  lower = NULL, upper = NULL, fixed = NULL,
  engine = c("adaptive_metropolis", "slice", "custom"),
  sampler = NULL, independent = FALSE, engine_label = NULL,
  propriety = NULL, moments = NULL, rng = NULL,
  rng_validator = NULL, validate = NULL,
```

```

density_is_log = NULL, prior_is_log = NULL,
prior_style = c("auto", "scalar", "vector"),
prior_label = "user-defined",
prior_kernel = "user-supplied function", reference = NULL)

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

```

Arguments

density	A density function whose first argument is the observation vector. Remaining named arguments are model parameters. An explicit logical log argument is recommended.
prior	A prior-density function accepting either named scalar parameters or a named parameter vector.
start	A finite, uniquely named numeric vector of starting values.
name	A nonempty model label used in fitted output.
lower, upper	Optional scalar, complete, or partially named parameter bounds. Missing bounds are infinite.
fixed	Optional named list of fixed model quantities.
engine	One of "adaptive_metropolis", "slice", or "custom". The slice route requires one unknown parameter.
sampler	For engine = "custom", a posterior-sampling function. See Details for its input and output contract.
independent	Whether draws returned by a custom sampler are independent. If true, MCMC convergence diagnostics are not applicable.
engine_label	Optional descriptive label for a custom sampler.
propriety	Optional user declaration or executable check of posterior propriety. It can be NULL, one logical value, one explanatory string, or a function of (x, fixed). A function returns a logical value or a list with proper and message.
moments	Optional data frame, or function of (x, fixed) returning a data frame, with columns parameter, mean_exists, variance_exists, and optionally note.
rng	Optional posterior-predictive generator. Its first argument is the requested sample size and its remaining arguments are model parameters.
rng_validator	Optional function verifying values generated by rng.
validate	Optional data-support function of (x, fixed). It returns TRUE for valid data; FALSE or an explanatory string rejects the data before posterior computation.
density_is_log, prior_is_log	Optional logical declarations for functions that do not expose a log argument.
prior_style	Whether the prior accepts named scalars, one named vector, or should be detected automatically.
prior_label, prior_kernel	Prior metadata stored in the fitted object.
reference	Optional bibliographic or methodological note stored with the specification.
x	A fitdistrBayes_model object.
...	Additional arguments, currently ignored by the print method.

Details

The adaptive Metropolis and slice engines construct the posterior kernel from density, prior, and the Jacobian implied by lower and upper. The custom engine delegates posterior simulation while retaining the package's validation, summaries, diagnostics, prediction, and pointwise log-likelihood interface.

A custom sampler receives the subset of its formal arguments matching the following names:

- Data and parameters: `x`, `start`, `fixed`, `lower`, and `upper`.
- Posterior evaluation: `log_posterior` and `log_posterior_unconstrained`.
- Parameter transformations: `to_unconstrained` and `from_unconstrained`.
- Sampling controls: `iter`, `warmup`, `thin`, `chains`, `n_save`, `control`, and `dots`.

It returns either one matrix when a single chain was requested, a list of chain matrices, or a list containing a chains component and optional independent, engine, acceptance, and initialization components. Chain matrices are on the natural parameter scale, have `n_save` rows, and have one named column per parameter.

Propriety and moment declarations supplied through this constructor are reported explicitly as user-supplied. They are executable metadata, not a mathematical certification by the package authors. If propriety is omitted, the fit warns and records it as the user's responsibility. If moment conditions are omitted, posterior medians and quantiles remain available, but means, standard deviations, and their Monte Carlo errors are not reported.

Value

`fitdistrBayes_model()` returns an object of class `"fitdistrBayes_model"`. It is a list containing the density and prior functions; named starting values and parameter bounds; fixed quantities; the selected posterior engine and optional custom sampler; user-supplied propriety, moment, support, and prediction components; and descriptive metadata. It contains no fitted values until passed to `fitdistrBayes(x, distr = model)`.

The print method returns its input invisibly and is called for the side effect of displaying the model name, parameters, prior, engine, and availability of a propriety declaration.

Examples

```
# A new Laplace model with a proper, non-objective prior.
d_laplace <- function(x, location, scale, log = FALSE) {
  value <- -log(2 * scale) - abs(x - location) / scale
  if (log) value else exp(value)
}
p_laplace <- function(location, scale, log = FALSE) {
  value <- dnorm(location, 0, 5, log = TRUE) +
    dlnorm(scale, 0, 0.75, log = TRUE)
  if (log) value else exp(value)
}
r_laplace <- function(n, location, scale) {
  location + scale * ifelse(runif(n) < 0.5, -1, 1) * rexp(n)
}
laplace_model <- fitdistrBayes_model(
```

```

d_laplace, p_laplace, start = c(location = 0, scale = 1),
lower = c(scale = 0), name = "Laplace",
propriety = "proper Normal--Lognormal prior for a nonconstant sample",
moments = data.frame(
  parameter = c("location", "scale"),
  mean_exists = c(TRUE, TRUE), variance_exists = c(TRUE, TRUE),
  note = rep("finite under the stated proper prior", 2)
),
rng = r_laplace,
validate = function(x) length(x) >= 2 && diff(range(x)) > 0,
prior_label = "Normal--Lognormal"
)
set.seed(1)
x_laplace <- r_laplace(25, location = 1, scale = 1.5)
fit_laplace <- fitdistrBayes(
  x_laplace, laplace_model, iter = 200, warmup = 80,
  chains = 2, seed = 2,
  control = list(warn_convergence = FALSE)
)
coef(fit_laplace)

```

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.

required_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.

`posterior_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")  
fitdistrBayes_routes("weighted lindley")
```

Index

- * **models**
 - criteria, [2](#)
 - fitcensBayes, [5](#)
 - fitcensBayes-methods, [9](#)
 - fitcensBayes_models, [12](#)
- * **survival**
 - fitcensBayes, [5](#)
- as.data.frame.fitcensBayes
 - (fitcensBayes-methods), [9](#)
- as.data.frame.fitdistrBayes
 - (fitdistrBayes-methods), [20](#)
- as.data.frame.fitdistrBayes_criteria
 - (criteria), [2](#)
- coef.fitcensBayes
 - (fitcensBayes-methods), [9](#)
- coef.fitdistrBayes
 - (fitdistrBayes-methods), [20](#)
- compare_models(criteria), [2](#)
- confint.fitcensBayes
 - (fitcensBayes-methods), [9](#)
- confint.fitdistrBayes
 - (fitdistrBayes-methods), [20](#)
- criteria, [2](#), [7](#), [14](#)
- DIC(criteria), [2](#)
- fitcensBayes, [5](#), [5](#), [11](#), [12](#), [18](#)
- fitcensBayes-methods, [9](#)
- fitcensBayes_models, [6](#), [9](#), [11](#), [12](#)
- fitdistrBayes, [5](#), [7](#), [9](#), [12](#), [13](#)
- fitdistrBayes-methods, [20](#)
- fitdistrBayes_model, [22](#)
- fitdistrBayes_routes, [12](#), [25](#)
- log_lik, [5](#), [11](#)
- log_lik(fitdistrBayes-methods), [20](#)
- log_lik.fitcensBayes
 - (fitcensBayes-methods), [9](#)
- log_lik_cens(fitcensBayes-methods), [9](#)
- LOOIC(criteria), [2](#)
- plot.fitcensBayes
 - (fitcensBayes-methods), [9](#)
- plot.fitdistrBayes
 - (fitdistrBayes-methods), [20](#)
- predict.fitcensBayes
 - (fitcensBayes-methods), [9](#)
- predict.fitdistrBayes
 - (fitdistrBayes-methods), [20](#)
- print.fitcensBayes
 - (fitcensBayes-methods), [9](#)
- print.fitdistrBayes
 - (fitdistrBayes-methods), [20](#)
- print.fitdistrBayes_comparison
 - (criteria), [2](#)
- print.fitdistrBayes_criteria
 - (criteria), [2](#)
- print.fitdistrBayes_model
 - (fitdistrBayes_model), [22](#)
- print.summary.fitcensBayes
 - (fitcensBayes-methods), [9](#)
- print.summary.fitdistrBayes
 - (fitdistrBayes-methods), [20](#)
- summary.fitcensBayes
 - (fitcensBayes-methods), [9](#)
- summary.fitdistrBayes
 - (fitdistrBayes-methods), [20](#)
- WAIC(criteria), [2](#)