Package: bayesDiagnostics (via r-universe)

May 15, 2026

Type Package

Title Comprehensive Bayesian Model Diagnostics and Comparison Tools

Version 0.1.0

Description Provides comprehensive tools for Bayesian model diagnostics and comparison. Includes prior sensitivity analysis, posterior predictive checks (Gelman et al. (2013) <doi:10.1201/b16018>), advanced model comparison using Pareto-smoothed importance sampling leave-one-out cross-validation (Vehtari et al. (2017) <doi:10.1007/s11222-016-9696-4>), convergence diagnostics, and prior elicitation tools. Integrates with 'brms' (Burkner (2017) <doi:10.18637/jss.v080.i01>), 'rstan', and 'rstanarm' packages for comprehensive Bayesian workflow diagnostics.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports brms (>= 2.18.0), checkmate (>= 2.1.0), ggplot2 (>= 3.4.0), gridExtra, loo (>= 2.5.0), posterior (>= 1.0.0), matrixStats, bridgesampling, rstan (>= 2.26.0), tidyr (>= 1.3.0)

Suggests bayesplot (>= 1.10.0), covr, knitr, magrittr (>= 2.0.0), purrr (>= 1.0.0), rmarkdown, rstanarm (>= 2.21.0), scales (>= 1.2.0), spelling, testthat (>= 3.0.0), tibble (>= 3.2.0), lme4

VignetteBuilder knitr

URL https://github.com/ikrakib/bayesDiagnostics

BugReports https://github.com/ikrakib/bayesDiagnostics/issues

Language en-US

Config/testthat/edition 3

Config/pak/sysreqs make libicu-dev

Repository https://ikrakib.r-universe.dev

Date/Publication 2026-01-14 10:25:11 UTC

RemoteUrl https://github.com/ikrakib/bayesdiagnostics

RemoteRef HEAD

RemoteSha 06dedad214f96f457a5aac56b942007940fd1cf0

Contents

`automated_ppc` *Automated Posterior Predictive Checks*

Description

Automatically computes a suite of posterior predictive checks to diagnose model fit. It compares observed data against posterior predictive samples across multiple statistics (mean, sd, min, max, skewness, kurtosis).

Usage

```
automated_ppc(model, observed_data, n_samples = 1000, p_value_threshold = 0.05)
```

Arguments

model	A fitted brmsfit object.
observed_data	Numeric vector of observed data.
n_samples	Integer. Number of posterior draws to use (default: 1000).
p_value_threshold	Numeric. Threshold for flagging extreme p-values (default: 0.05).

Value

A list of class automated_ppc containing diagnostics and flags.

---

bayes_factor_comparison

Bayesian Factor Comparison Between Models

Description

Computes and compares Bayes Factors between two or more Bayesian models using both marginal likelihood approximation and bridge sampling methods.

Usage

```
bayes_factor_comparison(
  ...,
  method = "bridge_sampling",
  repetitions = 5,
  silent = TRUE
)
```

Arguments

...	Named or unnamed brmsfit objects to compare
method	Character. Method for computing marginal likelihood. Options: "bridge_sampling" (default), "waic"
repetitions	Integer. Number of bridge sampling repetitions (default: 5)
silent	Logical. Suppress messages? (default: TRUE)

Value

A list of class bayes_factor_comparison containing:

bayes_factor	Bayes Factor for 2-model comparison
log_bf	Log Bayes Factor
interpretation	Interpretation of BF strength

`marginal_likelihoods`
: Data frame with model-level MLs

`model_names`
: Character vector of model names

`pairwise_comparisons`
: Data frame of pairwise BFs if 3+ models

`bayesDiagnostics` *bayesDiagnostics Package*

Description

Provides comprehensive tools for Bayesian model diagnostics and comparison.

Author(s)

Maintainer: Ibrahim Kholil Rakib <ikrakib1010@gmail.com>

See Also

Useful links:

- https://github.com/ikrakib/bayesDiagnostics
- Report bugs at https://github.com/ikrakib/bayesDiagnostics/issues

---

`bayesian_p_values` *Calculate Bayesian P-Values*

Description

A flexible utility to calculate Bayesian p-values for any custom test statistic.

Usage

```
bayesian_p_values(yrep, y, statistic)
```

Arguments

`yrep`	Matrix. Posterior predictive samples (rows = samples, cols = observations).
`y`	Vector. Observed data.
`statistic`	Function. The test statistic to compute (e.g., mean, max).

Value

A list with the observed stat, replicated stats, and the p-value.

`diagnostic_report` *Diagnostic Report for Bayesian Models*

Description

Generates comprehensive diagnostics and creates a formatted report for fitted Bayesian models (brmsfit, stanfit, etc.)

Usage

```
diagnostic_report(
  model,
  output_file = NULL,
  output_format = "pdf",
  include_sections = c("model_summary", "convergence", "posterior_summary",
    "recommendations"),
  rhat_threshold = 1.01,
  ess_threshold = 0.1,
  open_report = TRUE
)
```

Arguments

`model`	A fitted model object (brmsfit, stanfit, etc.)
`output_file`	Character. Path for output file
`output_format`	Character. Format: "pdf", "html", "docx"
`include_sections`	Character vector. Sections to include
`rhat_threshold`	Numeric. R-hat threshold for flagging (default: 1.01)
`ess_threshold`	Numeric. Effective sample size ratio threshold
`open_report`	Logical. Open report after generation?

Value

Invisibly returns output file path

`effective_sample_size_diagnostics` *Effective Sample Size Diagnostics*

Description

Comprehensive diagnostics for effective sample size (ESS) in MCMC chains, including bulk ESS, tail ESS, and per-chain analysis.

Usage

```
effective_sample_size_diagnostics(
  model,
  parameters = NULL,
  min_ess = 400,
  tail_quantiles = c(0.025, 0.975),
  by_chain = TRUE,
  plot = TRUE,
  ...
)

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

## S3 method for class 'ess_diagnostics'
plot(x, ...)
```

Arguments

`model`	A fitted Bayesian model (brmsfit, stanfit, or compatible)
`parameters`	Character vector of parameter names to analyze (default: all)
`min_ess`	Numeric. Minimum acceptable ESS (default: 400)
`tail_quantiles`	Numeric vector. Quantiles for tail ESS (default: c(0.025, 0.975))
`by_chain`	Logical. Whether to compute ESS per chain (default: TRUE)
`plot`	Logical. Whether to generate diagnostic plots (default: TRUE)
`...`	Additional arguments passed to plotting functions
`x`	Object of class `ess_diagnostics` (for print/plot methods).

Details

Effective Sample Size (ESS) measures the number of independent samples in MCMC chains after accounting for autocorrelation. This function provides:

- **Bulk ESS**: ESS for central posterior mass (mean, median)
- **Tail ESS**: ESS for extreme quantiles (credible intervals)

- **Per-chain ESS**: Identifies which chains have low ESS

Low ESS indicates high autocorrelation and may require:

- Longer chains (more iterations)
- Better parameterization
- Stronger priors
- Different sampler settings

Value

An object of class `ess_diagnostics` containing:

`ess_summary`	Summary statistics for ESS across parameters
`bulk_ess`	Bulk ESS for each parameter
`tail_ess`	Tail ESS for each parameter
`by_chain_ess`	Per-chain ESS if `by_chain = TRUE`
`problematic_params`	Parameters with ESS below threshold
`recommendations`	Specific recommendations for improving ESS

Examples

```
library(brms)
fit <- brm(mpg ~ hp + wt, data = mtcars)

# Comprehensive ESS diagnostics
ess_diag <- effective_sample_size_diagnostics(
  model = fit,
  min_ess = 400,
  by_chain = TRUE
)

print(ess_diag)
plot(ess_diag)
```

`extract_posterior_unified` *Extract Posterior Draws (Unified Interface)*

Description

Provides a unified interface for extracting posterior draws from multiple Bayesian modeling packages (brms, rstanarm, cmdstanr, etc.).

Usage

```
extract_posterior_unified(
  model,
  parameters = NULL,
  format = c("draws_df", "draws_matrix", "draws_array", "list"),
  n_draws = NULL,
  include_warmup = FALSE,
  chains = NULL,
  ...
)
```

Arguments

`model`	A fitted Bayesian model object
`parameters`	Character vector of parameter names to extract (default: all)
`format`	Output format: "draws_df", "draws_matrix", "draws_array", or "list"
`n_draws`	Number of draws to extract (default: all available)
`include_warmup`	Logical. Include warmup/burn-in draws (default: FALSE)
`chains`	Numeric vector of chain IDs to extract (default: all chains)
`...`	Additional arguments for specific model types

Details

This function provides a consistent interface across different Bayesian modeling packages, handling their different internal formats automatically.

Supported model types:

- `brmsfit` (brms)
- `stanfit` (rstan)
- `stanreg` (rstanarm)
- `CmdStanMCMC` (cmdstanr)
- `mcmc.list` (coda)

Value

Posterior draws in the requested format:

- **draws_df**: Data frame with one row per draw
- **draws_matrix**: Matrix with draws in rows, parameters in columns
- **draws_array**: 3D array (iterations x chains x parameters)
- **list**: Named list of parameter vectors

Examples

```
library(brms)
fit <- brm(mpg ~ hp + wt, data = mtcars)

# Extract as data frame
draws_df <- extract_posterior_unified(fit, format = "draws_df")

# Extract specific parameters
slopes <- extract_posterior_unified(
  fit,
  parameters = c("b_hp", "b_wt"),
  format = "draws_matrix"
)

# Extract first 1000 draws from chain 1
subset_draws <- extract_posterior_unified(
  fit,
  n_draws = 1000,
  chains = 1
)
```

---

graphical_ppc *Graphical Posterior Predictive Checks*

Description

Generates professional visualization of PPCs, including ribbon plots for uncertainty intervals and comparison densities.

Usage

```
graphical_ppc(model, observed_data, type = "density", n_draws = 50)
```

Arguments

model	A fitted brmsfit object.
observed_data	Numeric vector.
type	Character. "density" (default), "intervals", or "ribbon".
n_draws	Integer.

Value

A ggplot2 object.

`hierarchical_convergence` *Hierarchical Model Convergence Diagnostics*

Description

Performs specialized convergence diagnostics for hierarchical/multilevel Bayesian models, checking convergence at both group-level and population-level parameters.

Usage

```
hierarchical_convergence(
  model,
  group_vars = NULL,
  rhat_threshold = 1.01,
  ess_threshold = 400,
  check_shrinkage = TRUE,
  plot = TRUE,
  ...
)

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

## S3 method for class 'hierarchical_convergence'
plot(x, ...)
```

Arguments

`model`	A fitted hierarchical Bayesian model (brmsfit, stanfit, or compatible)
`group_vars`	Character vector of grouping variable names (e.g., "subject", "school")
`rhat_threshold`	Numeric. Threshold for R-hat diagnostic (default: 1.01)
`ess_threshold`	Numeric. Minimum effective sample size threshold (default: 400)
`check_shrinkage`	Logical. Whether to assess shrinkage patterns (default: TRUE)
`plot`	Logical. Whether to generate diagnostic plots (default: TRUE)
`...`	Additional arguments passed to plotting functions
`x`	Object of class `hierarchical_convergence` (for print/plot methods).

Details

Hierarchical models require special attention to convergence because:

- Group-level parameters often have slower mixing
- Variance components can be difficult to estimate

- Extreme shrinkage may indicate identification problems

The function checks:

- R-hat values for all parameters
- Effective sample sizes (bulk and tail ESS)
- Between-chain variance
- Shrinkage factor (ratio of group SD to pooled SD)

Value

An object of class `hierarchical_convergence` containing:

`population_diagnostics`
: Diagnostics for population-level (fixed) effects

`group_diagnostics`
: Diagnostics for group-level (random) effects

`shrinkage_metrics`
: Shrinkage statistics if `check_shrinkage = TRUE`

`convergence_summary`
: Overall convergence assessment

`warnings`
: List of convergence warnings

`model`
: Original fitted model

---

mcmc_diagnostics_summary

MCMC Diagnostics Summary

Description

Provides a comprehensive summary of MCMC convergence diagnostics, including R-hat, Effective Sample Size (ESS), and NUTS-specific issues like divergent transitions and tree depth saturation.

Usage

```
mcmc_diagnostics_summary(model, rhat_threshold = 1.01, ess_threshold = 400)
```

Arguments

`model`
: A fitted brmsfit object

`rhat_threshold`
: Numeric. Threshold for R-hat warning (default: 1.01)

`ess_threshold`
: Numeric. Threshold for ESS warning (default: 400)

Value

A list of class `mcmc_diagnostics` containing:

- `rhat_issues`: Parameters with high R-hat
- `ess_issues`: Parameters with low ESS
- `divergences`: Number of divergent transitions
- `tree_depth`: Number of iterations hitting max tree depth
- `summary_table`: Tibble of all diagnostics per parameter
- `converged`: Logical summary of overall convergence

`model_comparison_suite` *Comprehensive Model Comparison Suite*

Description

Compares multiple Bayesian models using information criteria (LOO, WAIC, Bayes R2) and generates comparison tables with rankings and visualizations.

Usage

```
model_comparison_suite(..., criterion = "loo", plot = TRUE, detailed = TRUE)
```

Arguments

`...`	Multiple brmsfit objects to compare
`criterion`	Character vector of criteria to use. Options: "loo" (default), "waic", "bayes_r2", "all"
`plot`	Logical. Generate comparison plots? (default: TRUE)
`detailed`	Logical. Return detailed statistics? (default: TRUE)

Value

A list of class `model_comparison` containing:

`comparison_table`	Data frame with model rankings and IC values
`ic_differences`	Data frame with IC differences and weights
`model_names`	Character vector of model names
`plots`	List of ggplot objects (if plot = TRUE)
`criterion_used`	Character vector of criteria used

plot.bayes_factor_comparison *Plot Bayes Factor Comparison Results*

Description

Plot Bayes Factor Comparison Results

Usage

```
## S3 method for class 'bayes_factor_comparison'
plot(x, ...)
```

Arguments

x	A bayes_factor_comparison object
...	Additional arguments passed to ggplot2 functions

Value

A ggplot2 plot object

posterior_predictive_check *Posterior Predictive Checks*

Description

Conducts posterior predictive checks to assess whether a fitted model generates data similar to the observed data. This is a key diagnostic for model adequacy and serves to identify systematic misspecifications.

Usage

```
posterior_predictive_check(
  model,
  observed_data,
  n_samples = 1000,
  test_statistics = c("mean", "sd", "median"),
  plot = TRUE,
  alpha = 0.7,
  ...
)

## S3 method for class 'posterior_predictive_check'
```

```
print(x, ...)

## S3 method for class 'posterior_predictive_check'
plot(x, ...)
```

Arguments

model	A fitted brmsfit object
observed_data	Vector or matrix of observed data
n_samples	Number of posterior predictive samples (default: 1000)
test_statistics	Character vector of test statistics to compute. Options: "mean", "sd", "median", "min", "max", "range", "skewness", "kurtosis"
plot	Logical. Whether to generate visualization (default: TRUE)
alpha	Numeric. Transparency level for plots (default: 0.7)
...	Additional arguments passed to plotting functions
x	Object of class posterior_predictive_check (for print/plot methods).

Details

Posterior predictive checks work by:

1. Extracting posterior draws from the fitted model
2. For each posterior draw, simulating new data from that parameter set
3. Computing test statistics on both observed and simulated data
4. Comparing the distributions to assess model adequacy

A well-fitting model should produce test statistics from simulated data similar to the observed test statistics. P-values near 0.5 indicate good model fit.

Value

Object of class posterior_predictive_check containing:

- observed_stats - Test statistics from observed data
- replicated_stats - Test statistics from posterior predictive samples
- p_values - Bayesian p-values for each test statistic
- model - Original fitted model
- n_samples - Number of samples used

ppc_crossvalidation *PPC Cross-Validation (LOO-PIT)*

Description

Performs Leave-One-Out (LOO) Probability Integral Transform (PIT) checks. A uniform distribution of PIT values indicates a well-calibrated model.

Usage

```
ppc_crossvalidation(model, observed_y, n_draws = NULL)
```

Arguments

model	A fitted brmsfit object.
observed_y	Numeric vector of response variable.
n_draws	Integer. Number of posterior draws to use for calculation. If NULL, uses all draws (recommended for accuracy).

Value

A list containing PIT values and a diagnostic plot object.

predictive_performance *Predictive Performance Evaluation*

Description

Comprehensive evaluation of Bayesian model predictive performance using multiple metrics: RMSE, MAE, Coverage, and proper scoring rules.

Usage

```
predictive_performance(
  model,
  newdata = NULL,
  observed_y,
  metrics = "all",
  credible_level = 0.95,
  n_draws = NULL
)
```

Arguments

model	A brmsfit object
newdata	Optional data frame for out-of-sample predictions. If NULL, uses model's original data.
observed_y	Numeric vector of observed response values. Must match nrow(newdata) or length(newdata).
metrics	Character vector of metrics to compute. Options: "rmse" (root mean square error), "mae" (mean absolute error), "coverage" (credible interval coverage), "crps" (continuous ranked prob score), "all" (default - all metrics)
credible_level	Numeric. Credible interval level for coverage (default: 0.95)
n_draws	Integer. Number of posterior draws to use (NULL = all)

Details

Predictive performance metrics evaluate how well posterior predictions align with data:

Point metrics:

- RMSE: Square root of mean squared prediction error
- MAE: Mean absolute error
- Correlation: Pearson correlation of predictions vs observed

Interval metrics:

- Coverage: Proportion of observations within credible interval
- Width: Average width of credible intervals

Proper scoring rules:

- CRPS: Continuous Ranked Probability Score (lower is better) Computed using empirical cumulative distribution function

Value

A list of class `predictive_performance` containing:

point_metrics	Data frame with RMSE, MAE, and correlation
interval_metrics	Data frame with coverage and interval width
proper_scores	Data frame with CRPS and log-score
prediction_summary	Data frame with mean, lower CI, upper CI for each observation
metrics_requested	Character vector of requested metrics
model_formula	Formula from the fitted model
sample_size	Number of observations

Examples

```
library(brms)

data <- data.frame(y = rnorm(100, mean = 5), x = rnorm(100))
model <- brm(y ~ x, data = data, chains = 1, iter = 1000, refresh = 0)

perf <- predictive_performance(model, observed_y = data$y, metrics = "all")
print(perf)
```

---

print.bayes_factor_comparison

Print Bayes Factor Comparison Results

Description

Print Bayes Factor Comparison Results

Usage

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

Arguments

x	A bayes_factor_comparison object
...	Additional arguments (currently unused)

Value

Invisibly returns the input object

---

prior_elicitation_helper

Prior Elicitation Helper

Description

Interactive tool to translate expert knowledge into statistical priors. Guides users through specification of prior distributions based on domain expertise and data characteristics.

Usage

```
prior_elicitation_helper(
  expert_beliefs,
  parameter_type = "continuous",
  method = "quantile",
  data_sample = NULL,
  visualize = TRUE,
  ...
)

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

Arguments

`expert_beliefs`	List containing expert beliefs about parameters. Elements: parameter_name, plausible_range, most_likely_value, confidence
`parameter_type`	Character: "continuous", "discrete", or "proportion"
`method`	Character: "quantile", "histogram", or "interactive"
`data_sample`	Numeric vector of observed data (optional, for context)
`visualize`	Logical. Show comparison plots (default: TRUE)
`...`	Additional arguments
`x`	Object of class `prior_elicitation` (for print method).

Details

This function helps bridge the gap between domain expertise and statistical prior specification. It uses several methods:

1. Quantile method: Expert specifies percentiles
2. Histogram method: Expert draws rough distribution shape
3. Interactive: Step-by-step guided elicitation

The function then matches the inputs to standard distributions (normal, t, gamma, beta, etc.) and suggests sensitivity analysis.

Value

Object of class `prior_elicitation` containing:

- `recommended_prior` - Prior specification as prior() object
- `parameter_summary` - Summary of expert inputs
- `diagnostic_plots` - Visualizations of prior
- `alternatives` - Alternative prior specifications
- `sensitivity_note` - Guidance on sensitivity analysis

`prior_robustness` *Prior Robustness Analysis*

Description

Comprehensive assessment of posterior robustness to alternative prior specifications using multiple sensitivity dimensions.

Usage

```
prior_robustness(
  model,
  prior_specifications,
  parameters,
  perturbation_direction = "expand",
  dimensions = c(0.5, 1, 2, 4),
  comparison_metric = "KL",
  credible_level = 0.95,
  plot = TRUE,
  ...
)

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

Arguments

`model`	A fitted brmsfit object
`prior_specifications`	List of alternative prior specifications. Each element should be a prior() object or named list of priors.
`parameters`	Character vector of parameters to analyze
`perturbation_direction`	Character: "expand", "contract", or "shift"
`dimensions`	Numeric vector of perturbation magnitudes (default: c(0.5, 1, 2, 4))
`comparison_metric`	One of "KL", "Wasserstein", "correlation", "coverage"
`credible_level`	Numeric. Credible interval level (default: 0.95)
`plot`	Logical. Generate visualizations (default: TRUE)
`...`	Additional arguments
`x`	Object of class `prior_robustness` (for print method).

Value

Object of class `prior_robustness` containing:

- `sensitivity_surfaces` - Multi-dimensional sensitivity results
- `robustness_index` - Composite robustness score
- `concerning_parameters` - Parameters with low robustness
- `recommendations` - Suggested prior refinements

`prior_sensitivity` *Prior Sensitivity Analysis*

Description

Conducts comprehensive prior sensitivity analysis to assess how robust posterior inferences are to alternative prior specifications.

Usage

```
prior_sensitivity(
  model,
  parameters,
  prior_grid,
  comparison_metric = "KL",
  plot = TRUE,
  n_draws = 2000,
  ...
)

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

## S3 method for class 'prior_sensitivity'
plot(x, ...)
```

Arguments

`model`	A fitted Bayesian model (brmsfit, stanfit, or compatible)
`parameters`	Character vector of parameter names to analyze
`prior_grid`	List of prior specifications to compare (named list)
`comparison_metric`	One of "KL", "Wasserstein", or "overlap"
`plot`	Logical. Whether to generate plots (default: TRUE)
`n_draws`	Number of posterior draws to use (default: 2000)
`...`	Additional arguments passed to plotting functions
`x`	Object of class `prior_sensitivity` (for print/plot methods).

Details

Prior sensitivity analysis assesses how much posterior inferences depend on the choice of prior distribution. Small sensitivity metrics indicate that conclusions are robust to prior specification.

Value

An object of class `prior_sensitivity` containing:

`sensitivity_metrics`
: Data frame with sensitivity metrics

`posteriors`
: List of posterior distributions for each prior

`comparison_metric`
: Metric used for comparison

`parameters`
: Parameters analyzed

`model`
: The original fitted model

Examples

```
library(brms)
fit <- brm(mpg ~ hp + wt, data = mtcars)

result <- prior_sensitivity(
  model = fit,
  parameters = c("b_hp", "b_wt"),
  prior_grid = list(
    weak = set_prior("normal(0, 10)", class = "b"),
    strong = set_prior("normal(0, 1)", class = "b")
  ),
  comparison_metric = "KL"
)

print(result)
plot(result)
```

Index