

# Package ‘powerbrmsINLA’

August 26, 2025

**Title** Bayesian Power Analysis Using 'brms' and 'INLA'

**Version** 1.0.0

**Maintainer** Tony Myers <admyers@aol.com>

**Description** Provides tools for Bayesian power analysis and assurance calculations using the statistical frameworks of 'brms' and 'INLA'. Includes simulation-based approaches, support for multiple decision rules (direction, threshold, ROPE), sequential designs, and visualisation helpers. Methods are based on Kruschke (2014, ISBN:9780124058880) ``Doing Bayesian Data Analysis: A Tutorial with R, JAGS, and Stan'', O'Hagan & Stevens (2001) <doi:10.1177/0272989X0102100307> ``Bayesian Assessment of Sample Size for Clinical Trials of Cost-Effectiveness'', Kruschke (2018) <doi:10.1177/2515245918771304> ``Rejecting or Accepting Parameter Values in Bayesian Estimation'', Rue et al. (2009) <doi:10.1111/j.1467-9868.2008.00700.x> ``Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations'', and Bürkner (2017) <doi:10.18637/jss.v080.i01> ``brms: An R Package for Bayesian Multilevel Models using Stan''.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**Depends** R (>= 4.0.0)

**Imports** brms (>= 2.19.0), dplyr (>= 1.1.0), ggplot2 (>= 3.4.0), rlang (>= 1.1.0), tibble (>= 3.2.0), scales (>= 1.2.0), viridisLite (>= 0.4.0), stats, utils, magrittr (>= 2.0.0)

**Suggests** INLA (>= 22.05.07), testthat (>= 3.0.0), rmarkdown, MASS, circular, sn

**URL** <https://github.com/Tony-Myers/powerbrmsINLA>

**BugReports** <https://github.com/Tony-Myers/powerbrmsINLA/issues>

**Additional\_repositories** <https://inla.r-inla-download.org/R/stable>

**NeedsCompilation** no

**Author** Tony Myers [aut, cre] (<<https://orcid.org/0000-0003-4516-4829>>)

## Contents

<code>.add_contour_lines</code>	2
<code>.auto_data_generator</code>	3
<code>.brms_to_inla_formula2</code>	4
<code>.compute_assurance</code>	5
<code>.geom_line_lw</code>	5
<code>.geom_point_lw</code>	6
<code>.gg_line_arg</code>	7
<code>.map_brms_priors_to_inla</code>	7
<code>.parse_re_terms</code>	8
<code>.scale_fill_viridis_continuous</code>	8
<code>.scale_fill_viridis_discrete</code>	9
<code>.should_stop_binom</code>	9
<code>.to_inla_family</code>	10
<code>beta_binom_power</code>	10
<code>beta_weights_on_grid</code>	11
<code>brms_inla_power</code>	12
<code>brms_inla_power_sequential</code>	14
<code>brms_inla_power_two_stage</code>	17
<code>hdi_of_icdf</code>	18
<code>min_n_beta_binom</code>	19
<code>or_or</code>	20
<code>plot_assurance_with_robustness</code>	20
<code>plot_bf_assurance_curve</code>	21
<code>plot_bf_expected_evidence</code>	22
<code>plot_bf_heatmap</code>	23
<code>plot_decision_assurance_curve</code>	23
<code>plot_decision_threshold_contour</code>	24
<code>plot_interaction_surface</code>	25
<code>plot_power_contour</code>	26
<code>plot_power_heatmap</code>	27
<code>plot_precision_assurance_curve</code>	27
<code>plot_precision_fan_chart</code>	28

---

<code>.add_contour_lines</code>	<i>Add Contour Lines to a ggplot2 Plot</i>
---------------------------------	--

---

## Description

Wrapper around `geom_contour` with preset defaults for color, alpha, width. Uses the correct `linewidth/size` argument depending on ggplot2 version.

## Usage

```
.add_contour_lines(
  mapping = NULL,
  data = NULL,
  breaks = NULL,
  colour = "white",
  alpha = 0.3,
  width = 0.2,
```

```
    bins = NULL
  )

  .add_contour_lines(
    mapping = NULL,
    data = NULL,
    breaks = NULL,
    colour = "white",
    alpha = 0.3,
    width = 0.2,
    bins = NULL
  )
```

### Arguments

<code>mapping</code>	Mapping aesthetic.
<code>data</code>	Data frame.
<code>breaks</code>	Break points for contours.
<code>colour</code>	Colour of contour lines.
<code>alpha</code>	Transparency level.
<code>width</code>	Line width.
<code>bins</code>	Number of bins for contour fill.

### Value

A ggplot2 layer adding contour lines.

A ggplot2 layer adding contour lines.

---

<code>.auto_data_generator</code>	<i>Automatic Data Generator for brms + INLA Simulation (Multi-Effect Ready)</i>
-----------------------------------	---

---

### Description

Creates a simulation function taking `n` (sample size) and a named effect vector/list. Supports multiple predictors, brms-style random effects, and most GLM families.

### Usage

```
.auto_data_generator(
  formula,
  effect_name,
  family = gaussian(),
  family_args = list(),
  error_sd = 1,
  group_sd = 0.5,
  obs_per_group = 10,
  predictor_means = NULL,
  predictor_sds = NULL
)
```

**Arguments**

<code>formula</code>	Model formula (may include brms-style random effects).
<code>effect_name</code>	Character vector of fixed effect names to manipulate.
<code>family</code>	brms family object, e.g. <code>gaussian()</code> , <code>binomial()</code> .
<code>family_args</code>	Named list of family-specific arguments.
<code>error_sd</code>	Residual SD for Gaussian-like families.
<code>group_sd</code>	SD of random effects.
<code>obs_per_group</code>	Number of observations per grouping level.
<code>predictor_means</code>	Named list of means for continuous predictors.
<code>predictor_sds</code>	Named list of SDs for continuous predictors.

**Value**

Function: `function(n, effect)` returns a `data.frame` with `n` rows.

---

`.brms_to_inla_formula2`

*Convert brms Formula to INLA Formula (Multi-Fixed Support)*

---

**Description**

Converts brms-style formula to INLA-ready formula, robust to multi-effect, and random effects.

**Usage**

```
.brms_to_inla_formula2(formula, drop_fixed = NULL)
```

**Arguments**

<code>formula</code>	brms-style formula.
<code>drop_fixed</code>	Character vector of fixed effects to drop (optional).

**Value**

List: `$inla_formula`, `$re_specs`

---

.compute_assurance	<i>Compute Mean Assurance for a Given Metric (Multi-Effect Compatible) Summarises simulation results and computes proportion passing for decision rule metric.</i>
--------------------	--

---

## Description

Compute Mean Assurance for a Given Metric (Multi-Effect Compatible) Summarises simulation results and computes proportion passing for decision rule metric.

## Usage

```
.compute_assurance(
  df,
  metric,
  prob_threshold,
  rope_rule = c(">="),
  direction_p = 0.5,
  effect_cols = NULL
)
```

## Arguments

df	Data frame containing simulation results with columns n, effect cols, ok, and metric columns.
metric	One of "direction", "threshold", "rope".
prob_threshold	Numeric, threshold for "threshold" and "rope" metrics.
rope_rule	Reserved for future use.
direction_p	Numeric cutoff for "direction" metric (default 0.5).
effect_cols	Optional character vector of effect columns to group by. If NULL, auto-detects "true_effect" or others.

## Value

Tibble grouped by n and effects with assurance column.

---

.geom_line_lw	<i>Create a ggplot2 Line Layer with Version-Compatible Width</i>
---------------	--

---

## Description

Creates a geom\_line with a width argument adapted to ggplot2 version.

## Usage

```
.geom_line_lw(mapping = NULL, data = NULL, ..., width = 1)
.geom_line_lw(mapping = NULL, data = NULL, ..., width = 1)
```

**Arguments**

mapping	Mapping aesthetic.
data	Data frame.
...	Additional parameters passed to <code>geom_line</code> .
width	Numeric line width for lines, default 1.

**Value**

ggplot2 layer for lines.

ggplot2 layer for lines.

---

`.geom_point_lw`

*Create a ggplot2 Point Layer with Version-Compatible Width*

---

**Description**

Creates a `geom_point` with a `width` argument adapted to ggplot2 version.

**Usage**

```
.geom_point_lw(mapping = NULL, data = NULL, ..., width = 1.5)
```

```
.geom_point_lw(mapping = NULL, data = NULL, ..., width = 1.5)
```

**Arguments**

mapping	Mapping aesthetic.
data	Data frame.
...	Additional parameters passed to <code>geom_point</code> .
width	Numeric line width for points, default 1.5.

**Value**

ggplot2 layer for points.

ggplot2 layer for points.

---

<code>.gg_line_arg</code>	<i>Compute Mean Assurance for a Given Metric (Modern, Multi-Effect Compatible)</i>
---------------------------	--

---

### Description

Computes the mean assurance (proportion passing) for a given decision metric across grouped cells.

### Usage

```
.gg_line_arg()
```

```
.gg_line_arg()
```

### Details

Determine ggplot2 Line Width Argument Name by Version

Returns the correct argument name for line width in ggplot2, depending on package version ("linewidth" for  $\geq 3.4.0$ , else "size").

### Value

Character string of argument name.

Character string of argument name.

---

`.map_brms_priors_to_inla`

*Map brms Priors to INLA Priors (Multi-Fixed)*

---

### Description

Parses a brms prior specification, mapping fixed effect priors to INLA `control.fixed`. Supports normal and `student_t` (variance-matched if  $df > 2$ ).

### Usage

```
.map_brms_priors_to_inla(priors, approx_t_as_normal = TRUE)
```

### Arguments

`priors` brms prior object or data.frame.

`approx_t_as_normal`

Logical; `student_t` priors with  $df > 2$  are treated as normal.

### Value

List with `$control_fixed` and `$hyper_by_re`.

---

<code>.parse_re_terms</code>	<i>Parse brms-like Random Effects Terms (Modern Robust)</i>
------------------------------	---

---

**Description**

Extracts random-effects specifications from a brms-style formula string. Supports (1 | g), (1 + x | g) - only one slope per term (warns otherwise!).

**Usage**

```
.parse_re_terms(formula)
```

**Arguments**

formula	A model formula.
---------	------------------

**Value**

List of RE spec lists (group, intercept/slope info)

---

<code>.scale_fill_viridis_continuous</code>	<i>Scale Fill for Viridis Continuous Data</i>
---	---

---

**Description**

Scale Fill for Viridis Continuous Data

**Usage**

```
.scale_fill_viridis_continuous(
  name = "Assurance",
  limits = c(0, 1),
  breaks = seq(0, 1, 0.2),
  labels = scales::percent_format(accuracy = 1)
)
```

**Arguments**

name	Legend title
limits	Numeric vector length 2 for limits
breaks	Numeric vector for breaks
labels	Function or vector for labels

**Value**

ggplot2 fill scale object



---

.scale\_fill\_viridis\_discrete

*Scale Fill for Viridis Discrete Data*

---

### Description

Scale Fill for Viridis Discrete Data

Scale Fill for Viridis Discrete Data

### Usage

```
.scale_fill_viridis_discrete(name = "Assurance")
```

```
.scale_fill_viridis_discrete(name = "Assurance")
```

### Arguments

name	Character legend title (default "Assurance")
------	--

### Value

ggplot2 fill scale object

ggplot2 fill scale object

---

.should\_stop\_binom

*Wilson Confidence Interval Early Stopping Rule Determines whether to stop early based on Wilson binomial confidence interval.*

---

### Description

Wilson Confidence Interval Early Stopping Rule Determines whether to stop early based on Wilson binomial confidence interval.

### Usage

```
.should_stop_binom(hits, trials, target, margin = 0.02, conf = 0.95)
```

### Arguments

hits	Number of successes observed
trials	Total number of trials
target	Target proportion
margin	Margin around target for stopping
conf	Confidence level for Wilson CI

### Value

List with stop (logical) and ci (numeric vector)

---

<code>.to_inla_family</code>	<i>Map a brms Family to an INLA Family (Modern, Robust)</i>
------------------------------	---

---

### Description

Map a brms Family to an INLA Family (Modern, Robust)

Convert brms Family to INLA Family Maps brms family specifications to corresponding INLA family names.

### Usage

```
.to_inla_family(family)
```

```
.to_inla_family(family)
```

### Arguments

<code>family</code>	A brms family object or character string.
---------------------	---

### Value

List with `$inla` and `$brms` family names.

List with brms and inla elements containing the family names.

---

<code>beta_binom_power</code>	<i>Analytic Assurance for Beta-Binomial Designs</i>
-------------------------------	---

---

### Description

Computes assurance (power) using generating and audience Beta priors for a binomial count via a Beta-Binomial predictive distribution.

### Usage

```
beta_binom_power(
  n,
  gen_prior_a,
  gen_prior_b,
  aud_prior_a,
  aud_prior_b,
  hdi_mass = 0.95,
  rope = NULL,
  hdi_max_width = NULL
)
```

**Arguments**

n	Sample size (number of trials).
gen_prior_a, gen_prior_b	Generating Beta prior parameters.
aud_prior_a, aud_prior_b	Audience Beta prior parameters.
hdi_mass	HDI mass (e.g., 0.95).
rope	Length-2 numeric vector for ROPE bounds, or NULL for max-width rule.
hdi_max_width	Positive width threshold for the HDI (used if rope=NULL).

**Value**

Assurance value between 0 and 1.

---

beta_weights_on_grid	<i>Beta-Prior Weights Over an Effect Grid</i>
----------------------	---

---

**Description**

Computes prior weights over a grid of true effect values by evaluating a Beta(mode, n) prior. If the grid is not in (0,1), it is rescaled linearly.

**Usage**

```
beta_weights_on_grid(effects, mode, n)
```

**Arguments**

effects	Numeric vector of effect values (grid).
mode	Prior mode in (0,1).
n	Prior concentration (> 2).

**Value**

Normalised numeric weights over the grid (sum to 1).

---

brms_inla_power	<i>Core Bayesian Assurance / Power Simulation (Modern, Multi-Effect Ready)</i>
-----------------	--

---

## Description

Provides Bayesian power analysis and assurance calculation using INLA (Integrated Nested Laplace Approximation) for efficient computation. Implements simulation-based power analysis for generalized linear mixed models with automatic threading optimization.

## Usage

```
brms_inla_power(
  formula,
  family = gaussian(),
  family_control = NULL,
  Ntrials = NULL,
  E = NULL,
  scale = NULL,
  priors = NULL,
  data_generator = NULL,
  effect_name,
  effect_grid = 0.5,
  sample_sizes = c(50, 100, 200, 400),
  nsims = 200,
  power_threshold = 0.8,
  precision_target = NULL,
  prob_threshold = 0.95,
  effect_threshold = 0,
  credible_level = 0.95,
  rope_bounds = NULL,
  error_sd = 1,
  group_sd = 0.5,
  obs_per_group = 10,
  predictor_means = NULL,
  predictor_sds = NULL,
  seed = 123,
  inla_hyper = NULL,
  compute_bayes_factor = FALSE,
  inla_num_threads = NULL,
  progress = c("auto", "text", "none"),
  family_args = list()
)
```

## Arguments

formula	Model formula.
family	brms GLM family (e.g., gaussian(), binomial()).
family_control	Optional list for INLA's control.family.
Ntrials	Optional vector for binomial trials.

E	Optional vector for Poisson exposure.
scale	Optional vector scale parameter for INLA families.
priors	Optional brms::prior specification.
data_generator	Optional function(n, effect) returning a dataset.
effect_name	Character vector of fixed effect names.
effect_grid	Vector/data.frame of effect values (supports multi-effect). For single effects, use a numeric vector. For multiple effects, use a data.frame with column names matching effect_name.
sample_sizes	Vector of sample sizes.
nsims	Number of simulations per cell.
power_threshold	Decision probability threshold for summary.
precision_target	Optional credible interval width target.
prob_threshold	Posterior probability threshold for decision rules.
effect_threshold	Effect-size threshold.
credible_level	Credible interval level (default 0.95).
rope_bounds	Optional Region of Practical Equivalence bounds (length 2 vector).
error_sd	Gaussian residual standard deviation.
group_sd	Random effects standard deviation.
obs_per_group	Observations per group.
predictor_means	Optional named list of predictor means.
predictor_sds	Optional named list of predictor standard deviations.
seed	Random seed.
inla_hyper	Optional INLA-specific hyperparameters.
compute_bayes_factor	Logical, compute Bayes Factor if TRUE.
inla_num_threads	Character string specifying INLA threading (e.g., "4:1" for 4 threads). If NULL (default), automatically detects optimal setting: "4:1" for 4+ cores, "2:1" for 2-3 cores, "1:1" otherwise.
progress	One of "auto", "text", or "none" for progress display.
family_args	List of arguments for family-specific data generators.

**Value**

List with results, summary, and settings.

**Examples**

```
# Basic usage with automatic INLA threading
results <- brms_inla_power(
  formula = outcome ~ treatment,
  effect_name = "treatment",
  effect_grid = c(0.2, 0.5, 0.8),
  sample_sizes = c(50, 100, 200),
  nsims = 3
)
print(results$summary)

# Manual INLA threading control
results <- brms_inla_power(
  formula = outcome ~ treatment,
  effect_name = "treatment",
  effect_grid = c(0.2, 0.5, 0.8),
  sample_sizes = c(50, 100, 200),
  inla_num_threads = "8:1", # Use 8 threads for faster computation
  nsims = 3
)

# Multi-effect design with threading
effect_grid <- expand.grid(
  treatment = c(0, 0.3, 0.6),
  age_effect = c(0, 0.2)
)
results <- brms_inla_power(
  formula = outcome ~ treatment + age_effect,
  effect_name = c("treatment", "age_effect"),
  effect_grid = effect_grid,
  sample_sizes = c(100, 200, 400),
  nsims = 3
)
print(results$summary)

# Quick parameter check (runs instantly)
formals(brms_inla_power)
```

---

brms\_inla\_power\_sequential

*Sequential Bayesian Assurance Simulation Engine (Modern, Multi-Effect Ready)*

---

**Description**

Simulates assurance sequentially in batches, stopping early per cell based on Wilson confidence intervals.

**Usage**

```
brms_inla_power_sequential(
  formula,
  family = gaussian(),
```

```

family_control = NULL,
Ntrials = NULL,
E = NULL,
scale = NULL,
priors = NULL,
data_generator = NULL,
effect_name,
effect_grid,
sample_sizes,
metric = c("direction", "threshold", "rope", "bf"),
target = 0.8,
prob_threshold = 0.95,
effect_threshold = 0,
rope_bounds = NULL,
credible_level = 0.95,
compute_bayes_factor = FALSE,
error_sd = 1,
group_sd = 0.5,
obs_per_group = 10,
predictor_means = NULL,
predictor_sds = NULL,
seed = 1,
batch_size = 20,
min_sims = 40,
max_sims = 600,
ci_conf = 0.95,
margin = 0.02,
inla_num_threads = NULL,
family_args = list(),
progress = TRUE
)

```

### Arguments

formula	brms-style model formula.
family	GLM family (e.g., gaussian(), binomial()).
family_control	Optional list for INLA's control.family.
Ntrials	Optional vector of binomial trial counts (for binomial families).
E	Optional vector of exposures (for Poisson families).
scale	Optional numeric vector for scale parameter in INLA.
priors	brms prior specification object.
data_generator	Optional function(n, effect) to simulate data.
effect_name	Character vector of fixed effects to assess.
effect_grid	Data frame or vector of effect values.
sample_sizes	Vector of sample sizes.
metric	Character; one of "direction", "threshold", "rope", or "bf" for Bayesian decision metric.
target	Target assurance value for stopping.
prob_threshold	Posterior probability threshold for decision metrics.

effect_threshold	Effect-size threshold.
rope_bounds	Numeric length-2 vector defining ROPE.
credible_level	Credible interval level for Bayesian inference.
compute_bayes_factor	Logical; TRUE if metric is "bf".
error_sd	Residual standard deviation.
group_sd	Standard deviation of random effects.
obs_per_group	Number of observations per group.
predictor_means	Optional named list of predictor means.
predictor_sds	Optional named list of predictor standard deviations.
seed	Random seed.
batch_size	Number of simulations per sequential look.
min_sims	Minimum simulations before early stopping.
max_sims	Maximum simulations per cell.
ci_conf	Confidence level for Wilson confidence intervals.
margin	Margin around target for early stopping decision.
inla_num_threads	Character string specifying INLA threading (e.g., "4:1"). If NULL (default), automatically detects optimal setting based on CPU cores.
family_args	List of family-specific args passed to data generator.
progress	Logical; if TRUE, show progress messages.

## Details

Sequential Bayesian Assurance Simulation Engine (Modern, Multi-Effect Ready)

Simulates assurance sequentially in batches, stopping early per cell based on Wilson confidence intervals.

## Value

List containing summary per cell and simulation settings.

## Examples

```
# Sequential design with automatic threading
results <- brms_inla_power_sequential(
  formula = outcome ~ treatment,
  effect_name = "treatment",
  effect_grid = c(0.2, 0.5, 0.8),
  sample_sizes = c(50, 100, 200),
  metric = "direction",
  target = 0.80
)
print(results$summary)
```



---

brms\_inla\_power\_two\_stage

*Two-Stage Bayesian Assurance Simulation (Multi-Effect, User-Friendly API)*


---

## Description

Runs a two-stage Bayesian assurance simulation with formula-based multi-effect grids and adaptive refinement.

## Usage

```
brms_inla_power_two_stage(
  formula,
  effect_name,
  effect_grid,
  n_range,
  stage1_k_n = 8,
  stage1_nsims = 100,
  stage2_nsims = 400,
  refine_metric = c("direction", "threshold", "rope"),
  refine_target = 0.8,
  prob_threshold = 0.95,
  effect_threshold = 0,
  obs_per_group = NULL,
  error_sd = NULL,
  group_sd = 0.5,
  band = 0.06,
  expand = 1L,
  inla_num_threads = NULL,
  ...
)
```

## Arguments

formula	Model formula.
effect_name	Character vector of fixed effect names; must match formula terms.
effect_grid	Data frame with columns named by effect_name specifying effect values.
n_range	Numeric length-2 vector specifying sample size range.
stage1_k_n	Number of grid points in stage 1.
stage1_nsims	Number of simulations per cell in stage 1.
stage2_nsims	Number of simulations per cell in stage 2.
refine_metric	Metric used for refinement; one of "direction", "threshold", or "rope".
refine_target	Target assurance for refined cells.
prob_threshold	Posterior probability threshold for decision.
effect_threshold	Effect-size threshold for decision metric.
obs_per_group	Number of observations per group for grouping factors.

error_sd	Residual standard deviation.
group_sd	Standard deviation of random effects.
band	Numeric width of the target refinement band.
expand	Integer; how much to expand the refinement grid around candidates.
inla_num_threads	Character string specifying INLA threading (e.g., "4:1"). If NULL (default), automatically detects optimal setting based on CPU cores.
...	Additional arguments passed to internal functions.

**Value**

A list with combined simulation results, summary, and stage parameters.

**Examples**

```
# Two-stage design with threading
effect_grid <- expand.grid(
  treatment = c(0.2, 0.5, 0.8),
  covariate = c(0.1, 0.3)
)
results <- brms_inla_power_two_stage(
  formula = outcome ~ treatment + covariate,
  effect_name = c("treatment", "covariate"),
  effect_grid = effect_grid,
  n_range = c(50, 200),
  stage1_nsims = 3,
  stage2_nsims = 3,
  error_sd = 1
)
print(results$summary)
```

---

hdi\_of\_icdf

*Highest Density Interval from an Inverse CDF*


---

**Description**

Computes an HDI of given mass from any distribution for which you have a quantile function (inverse CDF).

**Usage**

```
hdi_of_icdf(qfun, width = 0.95, tol = 1e-08, ...)
```

**Arguments**

qfun	Quantile function, e.g., qbeta, qnorm, ...
width	Desired HDI mass (e.g., 0.95).
tol	Optimizer tolerance.
...	Additional arguments passed to qfun.

**Value**

Named numeric vector with elements l1 and u1.

---

min_n_beta_binom	<i>Minimum n for Target Assurance (Beta-Binomial)</i>
------------------	---

---

**Description**

Minimum n for Target Assurance (Beta-Binomial)

**Usage**

```
min_n_beta_binom(
  gen_prior_mode,
  gen_prior_n,
  desired_power,
  aud_prior_mode = 0.5,
  aud_prior_n = 2,
  hdi_mass = 0.95,
  rope = NULL,
  hdi_max_width = NULL,
  n_start = 20,
  n_max = 1e+05,
  verbose = TRUE
)
```

**Arguments**

gen_prior_mode	Generating prior mode in (0,1).
gen_prior_n	Generating prior concentration ( $\geq 2$ ).
desired_power	Target assurance value in (0,1).
aud_prior_mode	Audience prior mode in (0,1).
aud_prior_n	Audience prior concentration ( $\geq 2$ ).
hdi_mass	HDI mass (e.g., 0.95).
rope	Length-2 numeric vector for ROPE bounds, or NULL for max-width rule.
hdi_max_width	Positive width threshold for the HDI (used if rope=NULL).
n_start	Starting sample size for search.
n_max	Maximum sample size to try.
verbose	If TRUE, prints progress.

**Value**

Smallest n meeting the target assurance.

---

or_or	<i>Internal Coalesce Operator Returns the left-hand side if it is not NULL, otherwise the right-hand side.</i>
-------	--

---

### Description

Internal Coalesce Operator Returns the left-hand side if it is not NULL, otherwise the right-hand side.

### Usage

```
x %||% y
```

### Arguments

x	Left-hand value.
y	Right-hand value.

### Value

If x is not NULL, returns x; else y.

---

plot_assurance_with_robustness	<i>Plot Assurance with Robustness Ribbon (Multi-Effect Grid Friendly)</i>
--------------------------------	---

---

### Description

Compares assurance results from multiple scenarios by showing the range ("ribbon") of values across scenarios for each sample size and effect grid variable.

### Usage

```
plot_assurance_with_robustness(
  power_results_list,
  metric = c("precision", "direction", "threshold", "bf"),
  x_effect = NULL,
  facet_by = NULL,
  precision_target = NULL,
  p_star = 0.95,
  bf_threshold = 10,
  effect_filters = NULL,
  effect_weights = NULL,
  show_individual_scenarios = FALSE,
  title = NULL,
  subtitle = NULL
)
```

**Arguments**

power_results_list	Named list of results objects from brms_inla_power or sequential/two-stage variants.
metric	Which assurance metric to compute: "precision", "direction", "threshold", or "bf".
x_effect	Name of effect grid column for x-axis (default: first detected grid column).
facet_by	Optional effect grid column(s) to facet by.
precision_target	CI width target if metric="precision".
p_star	Posterior probability threshold for "direction"/"threshold".
bf_threshold	BF10 threshold for "bf".
effect_filters	Optional named list for filtering rows (e.g. list(treatment=0)).
effect_weights	Optional named numeric vector for averaging over grid values.
show_individual_scenarios	Logical; if TRUE, overlay each scenario's curve.
title, subtitle	Optional plot labels.

**Value**

A ggplot object.

---

plot\_bf\_assurance\_curve

*Plot Bayes Factor Assurance Curve (Multi-Effect Grid Friendly)*

---

**Description**

Plots the proportion of simulations in which BF10 meets or exceeds a threshold, grouped by any effect grid variable(s) and sample size.

**Usage**

```
plot_bf_assurance_curve(
  power_results,
  bf_threshold = 3,
  x_effect = NULL,
  facet_by = NULL,
  effect_filters = NULL,
  effect_weights = NULL,
  title = NULL,
  subtitle = NULL
)
```

**Arguments**

power_results	List returned by brms_inla_power* or two-stage variant.
bf_threshold	Numeric; BF10 threshold to count as a "success" (default: 3).
x_effect	Name of effect grid column for x-axis (default: first detected grid column).
facet_by	Optional grid column(s) for faceting.
effect_filters	Optional named list to restrict/show only selected grid rows, e.g. list(treatment=0).
effect_weights	Optional named numeric vector of weights for selected x_effect values.
title, subtitle	Optional plot labels.

**Value**

ggplot object.

---

plot\_bf\_expected\_evidence

*Plot Expected Evidence (mean log10 BF10, Multi-Effect Grid Friendly)*

---

**Description**

Plots the average log10 BF10 against any effect grid variable, grouped/faceted.

**Usage**

```
plot_bf_expected_evidence(
  power_results,
  x_effect = NULL,
  facet_by = NULL,
  n = NULL,
  agg_fun = mean,
  title = NULL,
  subtitle = NULL
)
```

**Arguments**

power_results	Simulation results from a brms_inla_power* function with compute_bayes_factor = TRUE.
x_effect	Name of effect grid column for x-axis (default: first grid column).
facet_by	Optional grid column(s) to facet by (default: NULL).
n	Optional sample size to filter to (NULL means plot all; else one curve per grid/facet).
agg_fun	Aggregation function if >1 entries per cell (default: mean).
title, subtitle	Optional plot labels.

**Value**

A ggplot object.

---

plot_bf_heatmap	<i>Plot Bayes Factor Heatmap (mean log10 BF10, Multi-Effect Grid Friendly)</i>
-----------------	--

---

### Description

Heatmap of mean log10 BF10 as a function of two effect grid columns (x/y), with optional faceting.

### Usage

```
plot_bf_heatmap(
  power_results,
  x_effect = NULL,
  y_effect = "n",
  facet_by = NULL,
  n = NULL,
  agg_fun = mean,
  title = NULL,
  subtitle = NULL
)
```

### Arguments

power_results	Simulation results from a brms_inla_power* function with compute_bayes_factor = TRUE.
x_effect	Name of effect grid column for x-axis (default: first grid column).
y_effect	Name of effect grid column for y-axis (default: "n").
facet_by	Optional column(s) to facet by.
n	Optional sample size to filter to (NULL means plot all; else show only that n).
agg_fun	Aggregation function (default: mean).
title, subtitle	Optional plot labels.

### Value

ggplot object.

---

plot_decision_assurance_curve	<i>Plot Decision Assurance Curve (Multi-Effect Grid Friendly)</i>
-------------------------------	---

---

### Description

Plots the assurance (proportion of simulation runs meeting a posterior probability decision rule) versus an effect grid variable, for a given metric ("direction", "threshold", or "rope") at a fixed decision probability threshold p\_star.

**Usage**

```
plot_decision_assurance_curve(
  power_results,
  metric = c("direction", "threshold", "rope"),
  p_star = 0.95,
  x_effect = NULL,
  facet_by = NULL,
  effect_filters = NULL,
  effect_weights = NULL,
  title = NULL,
  subtitle = NULL
)
```

**Arguments**

**power\_results** A list returned by `brms_inla_power*`.

**metric** Decision metric: "direction", "threshold", or "rope".

**p\_star** Numeric decision threshold in (0,1).

**x\_effect** Name of effect grid column for x-axis (default: first grid column).

**facet\_by** Optional effect grid column(s) to facet by.

**effect\_filters** Optional named list for filtering rows, e.g. `list(treatment=0)`.

**effect\_weights** Optional named numeric vector of weights for selected `x_effect` values.

**title, subtitle** Optional plot labels.

**Value**

A ggplot object.

---

`plot_decision_threshold_contour`

*Plot Decision Threshold Contour (Multi-Effect Grid Friendly)*

---

**Description**

Shows assurance as a function of decision threshold  $p^*$  and one effect grid column, optionally faceted.

**Usage**

```
plot_decision_threshold_contour(
  power_results,
  metric = c("direction", "threshold", "rope"),
  p_star_grid = seq(0.5, 0.99, by = 0.01),
  effect_var = NULL,
  facet_by = NULL,
  effect_value = NULL,
  effect_weights = NULL,
  title = NULL,
  subtitle = NULL
)
```



**Arguments**

power_results	brms_inla_power list (or two-stage, etc.)
metric	Which metric: "direction", "threshold", "rope"
p_star_grid	Numeric vector of decision thresholds (default: 0.5 to 0.99 by 0.01)
effect_var	Name of effect grid column for y-axis (default: first detected grid column)
facet_by	Optional effect grid column(s) to facet by
effect_value	Optional value(s) to filter for effect_var, or named list for multi-filter
effect_weights	Optional weights for aggregation (named by effect_var values)
title, subtitle	Optional plot labels.

**Value**

ggplot2 object.

---

plot\_interaction\_surface

*Plot Interaction Assurance Surface/Heatmap/Lines (Multi-Effect Grid Friendly)*

---

**Description**

Visualizes a metric (e.g., assurance) as a function of two effect grid variables for a fixed sample size or averaged over n. Allows line, heatmap, or contour modes.

**Usage**

```
plot_interaction_surface(
  data,
  metric,
  effect1,
  effect2,
  n = NULL,
  line = FALSE,
  facet_by = NULL,
  agg_fun = mean,
  title = NULL,
  subtitle = NULL
)
```

**Arguments**

data	Data frame (typically power_results\$summary).
metric	Name of the summary column to plot, e.g. "power_direction", "power_threshold".
effect1	Name of effect grid column for x-axis.
effect2	Name of effect grid column for y-axis or color/facets.
n	Optional sample size to filter to (else averages/plots all n's).

line	Logical; if TRUE, make a lineplot (effect1 on x, one line for each effect2). If FALSE, make a heatmap or contour.
facet_by	Optional grid column(s) to facet by.
agg_fun	Aggregation function if multiple entries per cell (default = mean).
title, subtitle	Optional plot labels.

**Value**

A ggplot object.

---

plot_power_contour	<i>Plot Bayesian Power / Assurance Contour (Multi-Effect Grid Friendly)</i>
--------------------	---

---

**Description**

Draw a filled contour plot of assurance for a chosen metric, as a function of two effect grid columns and sample size.

**Usage**

```
plot_power_contour(
  power_results,
  power_metric = c("direction", "threshold", "rope"),
  x_effect = NULL,
  y_effect = "n",
  facet_by = NULL,
  power_threshold = 0.8,
  show_threshold_line = TRUE,
  title = NULL,
  subtitle = NULL
)
```

**Arguments**

power_results	Output from a brms_inla_power function.
power_metric	Which metric to plot: "direction", "threshold", or "rope".
x_effect	Name of effect grid column for x-axis (default = first effect).
y_effect	Name of effect grid column for y-axis (default = "n").
facet_by	Optional effect grid column(s) to facet by.
power_threshold	Optional contour line for assurance (default 0.8).
show_threshold_line	Logical; add a red contour at power_threshold.
title, subtitle	Optional plot labels.

**Value**

A ggplot object.

---

plot_power_heatmap	<i>Plot Bayesian Power / Assurance Heatmap (Multi-Effect Grid Friendly)</i>
--------------------	---

---

### Description

Heatmap of assurance for a chosen metric across two selected effect grid variables and sample sizes.

### Usage

```
plot_power_heatmap(
  power_results,
  power_metric = c("direction", "threshold", "rope"),
  x_effect = NULL,
  y_effect = "n",
  facet_by = NULL,
  title = NULL,
  subtitle = NULL
)
```

### Arguments

power_results	Output from a brms_inla_power function.
power_metric	Which metric to plot: "direction", "threshold", or "rope".
x_effect	Name of effect grid column for x-axis (default = first effect).
y_effect	Name of effect grid column for y-axis (default = "n").
facet_by	Optional effect grid column(s) to facet by.
title, subtitle	Optional plot labels.

### Value

A ggplot object.

---

plot_precision_assurance_curve	<i>Plot Precision Assurance Curve (Multi-Effect Grid Friendly)</i>
--------------------------------	--

---

### Description

Plots the assurance (proportion of runs meeting CI width  $\leq$  target) vs. a chosen effect grid variable across sample size(s). Supports faceting, effect filtering, and weights.

**Usage**

```
plot_precision_assurance_curve(
  power_results,
  precision_target,
  x_effect = NULL,
  facet_by = NULL,
  effect_filters = NULL,
  effect_weights = NULL,
  title = NULL,
  subtitle = NULL
)
```

**Arguments**

`power_results` List returned by `brms_inla_power*`.

`precision_target` Numeric; credible interval width threshold for success.

`x_effect` Name of effect grid column for x-axis (default: first grid column).

`facet_by` Optional effect grid column(s) for faceting.

`effect_filters` Optional named list for filtering rows, e.g. `list(treatment=0)`.

`effect_weights` Optional named numeric vector for weights over selected `x_effect` values.

`title, subtitle` Optional plot labels.

**Value**

A ggplot object.

---

`plot_precision_fan_chart`

*Plot Precision Assurance Fan Chart (Multi-Effect Grid Friendly)*

---

**Description**

Shows assurance (proportion of runs meeting CI width  $\leq$  target) across sample size(s) and effect grid. Optionally overlays the range (fan/ribbon) across multiple scenarios.

**Usage**

```
plot_precision_fan_chart(
  power_results_list,
  ci_width_target,
  x_effect = NULL,
  facet_by = NULL,
  effect_filters = NULL,
  effect_weights = NULL,
  show_individual_scenarios = FALSE,
  title = NULL,
  subtitle = NULL
)
```

**Arguments**

<code>power_results_list</code>	Named list of <code>brms_inla_power</code> results (for fan chart across scenarios) or a single object.
<code>ci_width_target</code>	Numeric; target credible interval width.
<code>x_effect</code>	Name of effect grid column for x-axis (default: first detected grid column).
<code>facet_by</code>	Optional grid column(s) for faceting.
<code>effect_filters</code>	Optional named list for filtering rows, e.g. <code>list(treatment=0)</code> .
<code>effect_weights</code>	Optional named numeric vector for averaging over grid values.
<code>show_individual_scenarios</code>	Logical: overlay all scenario curves if TRUE.
<code>title, subtitle</code>	Optional plot labels.

**Value**

A ggplot object.