Package 'bayprior'

October 20, 2026

Title Bayesian Prior Elicitation, Diagnostics, and Regulatory Reporting

Version 0.5.0

Description A toolkit for constructing, validating, and justifying Bayesian priors in clinical trial settings. Implements expert elicitation via quantile matching, the roulette method, and moment matching across six distribution families, linear and logarithmic expert pooling, meta-analytic-predictive (MAP) priors derived from historical trials, prior-data conflict diagnostics including the Box p-value (computed from the exact prior predictive distribution where one has a closed form), an exploratory surprisal (S-value) companion to it, surprise index, information divergence, and Mahalanobis distance, sensitivity analyses with tornado and influence heatmap plots, sceptical, robust, and power priors, and automated prior justification reports. Includes a fully modular 'Shiny' application for interactive use.
Methods based on O'Hagan et al. (2006, ISBN:9780470029886),
Box (1980) <doi:10.2307/2982063>,
Greenland (2023) <doi:10.1111/sjos.12625>,
Oakley and O'Hagan (2010) <https://tonyohagan.co.uk/shelf/>,
Schmidli et al. (2014) <doi:10.1111/biom.12242>,
Neuenschwander et al. (2010) <doi:10.1177/1740774509356002>,
Roever et al. (2021) <doi:10.1002/jrsm.1475>,
Ibrahim and Chen (2000) <doi:10.1214/ss/1009212673>,
Spiegelhalter et al. (1994) <doi:10.2307/2983527>.

License GPL-3

URL https://github.com/ndohpenngit/bayprior

BugReports https://github.com/ndohpenngit/bayprior/issues

Depends R (>= 4.1.0)

Imports cli (>= 3.6.0), config (>= 0.3.1), dplyr (>= 1.1.0), DT (>= 0.27), ggplot2 (>= 3.4.0), glue (>= 1.6.2), golem (>= 0.4.1), plotly (>= 4.10.1), purrr (>= 1.0.0), rlang (>= 1.2.0), shiny (>= 1.7.4), shinycssloaders (>= 1.0.0), shinydashboard (>= 0.7.2), shinyjs (>= 2.1.0), shinyWidgets (>= 0.7.6), stats

Suggests brms, htmlwidgets (>= 1.5.4), knitr (>= 1.40), metafor (>= 4.0.0), patchwork (>= 1.1.0), quarto (>= 1.4), rmarkdown (>= 2.20), rstanarm, shinytest2 (>= 0.3.1), spelling (>= 2.2.0), testthat (>= 3.0.0), vdiffr (>= 1.0.0), withr (>= 2.5.0)

VignetteBuilder knitr

Config/Needs/vignettes knitr, rmarkdown, rstanarm, brms

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

RoxygenNote 7.3.3

SystemRequirements Quarto CLI (>= 1.5, https://quarto.org)

NeedsCompilation no

Author Ndoh Penn [aut, cre] (ORCID: <https://orcid.org/0009-0003-9054-465X>)

Maintainer Ndoh Penn <ndohpenn9@gmail.com>

Repository CRAN

Date/Publication 2026-10-20 21:30:02 UTC

Contents

aggregate_experts . . . 3
as_prior . . . 5
bayprior . . . 6
calibrate_power_prior . . . 7
conflict_mahalanobis . . . 9
elicit_beta . . . 11
elicit_exponential . . . 12
elicit_gamma . . . 14
elicit_lognormal . . . 15
elicit_mixture . . . 16
elicit_normal . . . 17
elicit_roulette . . . 18
elicit_weibull . . . 19
historical_effect_sizes . . . 20
map_prior . . . 22
plot.bayprior_power_prior . . . 24
plot_prior_likelihood . . . 25
plot_sensitivity . . . 26
plot_tau_posterior . . . 26
plot_tornado . . . 27
print.bayprior . . . 28
print.bayprior_conflict . . . 28
print.bayprior_conflict_mv . . . 29
print.bayprior_power_prior . . . 29
prior_conflict . . . 30

prior_report 32
resolve_tau_prior 34
robust_prior 35
run_app 36
sceptical_prior 37
sensitivity_cri 38
sensitivity_grid 39
summary.bayprior 40

Index **42**

`aggregate_experts` *Aggregate multiple expert priors into a consensus prior*

Description

Combines elicited priors from multiple experts into a single consensus distribution using either linear pooling (mixture) or logarithmic pooling (normalised product of densities). Includes diagnostics for inter-expert disagreement.

Usage

```
aggregate_experts(
  priors,
  weights = NULL,
  method = c("linear", "logarithmic"),
  disagreement_threshold = 0.5
)
```

Arguments

`priors`	A named list of `bayprior` objects, one per expert.
`weights`	Numeric vector of expert weights (summing to 1). If `NULL`, equal weights are applied.
`method`	Character. `"linear"` (default) or `"logarithmic"` pooling.
`disagreement_threshold`	Numeric in (0, 1). Triggers a warning when the pairwise Bhattacharyya coefficient drops below this value, flagging substantial expert disagreement. Default `0.5`.

Details

Linear pooling satisfies the marginalization property (McConway, 1981): pooling a joint distribution and then marginalising gives the same result as marginalising each expert's distribution first and then pooling. The consensus density is a weighted mixture $\pi(\theta) = \sum_k w_k \pi_k(\theta)$. This is the most commonly used approach in clinical trial settings (O'Hagan et al., 2006). The resulting prior always lies within the convex hull of individual expert priors. The mixture's summary mean and

SD (in `$fit_summary`, used e.g. by `sensitivity_grid` to derive a working prior) are computed exactly from the component means/SDs and weights, without numerical integration:

$$\text{mean} = \sum_k w_k \bar{x}_k, \quad \text{var} = \sum_k w_k \left(s_k^2 + \bar{x}_k^2\right) - \text{mean}^2$$

Logarithmic pooling satisfies external Bayesianity (Genest, Weerahandi & Zidek, 1984): pooling experts' priors and then updating on data gives the same result as updating each expert's prior individually and then pooling the posteriors – pooling and Bayesian updating commute. This is in fact the *only* pooling operator with this property (Genest, McConway & Schervish, 1986); no pooling method can satisfy both external Bayesianity and marginalization simultaneously. The consensus density is proportional to $\prod_k \pi_k(\theta)^{w_k}$, which produces a sharper consensus when experts agree, but can be severely influenced by outlying expert opinions. Unlike linear pooling, no closed-form summary SD exists for the pooled density in general; `$fit_summary$sd` is `NULL` for logarithmically-pooled priors.

Value

A `bayprior` object (`dist = "mixture"` for linear pooling, `dist = "log_pool"` for logarithmic), with an additional `$aggregation` component containing:

`method` Pooling method used

`weights` Applied weights

`disagreement` Pairwise Bhattacharyya coefficients

`n_experts` Number of experts

References

O'Hagan, A., et al. (2006). *Uncertain Judgements: Eliciting Experts' Probabilities*. Wiley.

McConway, K. J. (1981). Marginalization and linear opinion pools. *Journal of the American Statistical Association*, 76(374), 410-414.

Genest, C., Weerahandi, S., & Zidek, J. V. (1984). Aggregating opinions through logarithmic pooling. *Theory and Decision*, 17(1), 61-70.

Genest, C., McConway, K. J., & Schervish, M. J. (1986). Characterization of externally Bayesian pooling operators. *The Annals of Statistics*, 14(2), 487-501.

Examples

```
p1 <- elicit_beta(mean = 0.25, sd = 0.08, method = "moments", expert_id = "E1",
                  label = "Response rate")
p2 <- elicit_beta(mean = 0.35, sd = 0.10, method = "moments", expert_id = "E2",
                  label = "Response rate")
p3 <- elicit_beta(mean = 0.30, sd = 0.09, method = "moments", expert_id = "E3",
                  label = "Response rate")

consensus <- aggregate_experts(
  priors  = list(E1 = p1, E2 = p2, E3 = p3),
  weights = c(0.4, 0.3, 0.3),
  method  = "linear"
```

```
)
print(consensus)
plot(consensus)
```

`as_prior` *Constructor for bayprior from raw parameters*

Description

Construct a `bayprior` object directly from known hyperparameters (e.g., literature-based priors), bypassing elicitation.

Usage

```
as_prior(dist, params, label = "Prior", expert_id = "Literature")
```

Arguments

`dist`	Character. One of `"beta"`, `"normal"`, `"gamma"`.
`params`	Named list of hyperparameters.
`label`	Character. Description of the quantity.
`expert_id`	Character. Source identifier.

Value

A `bayprior` object.

Examples

```
# Historical Beta(2, 8) prior on response rate
prior <- as_prior("beta", list(alpha = 2, beta = 8),
                  label = "Historical response rate prior")
```

bayprior *bayprior: Bayesian Prior Elicitation, Diagnostics, and Regulatory Reporting*

Description

A toolkit for constructing, validating, and justifying Bayesian priors in clinical trial settings. Implements SHELF-style expert elicitation (quantile matching, roulette method, moment matching) across six distribution families, linear and logarithmic expert pooling with compatibility validation, meta-analytic-predictive (MAP) priors derived from historical trials, prior-data conflict diagnostics (Box p-value, exact prior predictive where available, S-value, surprise index, KL divergence, Bhattacharyya overlap, Mahalanobis check) for binary, continuous, Poisson/count, and survival data types, sensitivity analyses with tornado and influence plots, sceptical/robust/power priors, and automated HTML/PDF/Word regulatory reports informed by FDA and EMA guidance on Bayesian methods. Includes a modular Shiny application for interactive use.

Main workflow

1. **Elicitation** – `elicit_beta`, `elicit_normal`, `elicit_gamma`, `elicit_lognormal`, `elicit_exponential`, `elicit_weibull`, `elicit_roulette`, `elicit_mixture`, `as_prior`
2. **Expert pooling** – `aggregate_experts`
3. **MAP priors** – `map_prior`, `resolve_tau_prior`, `historical_effect_sizes`
4. **Conflict diagnostics** – `prior_conflict`, `conflict_mahalanobis`
5. **Sensitivity analysis** – `sensitivity_grid`, `sensitivity_cri`
6. **Robust priors** – `sceptical_prior`, `robust_prior`, `calibrate_power_prior`
7. **Reporting** – `prior_report`
8. **Visualisation** – `plot_prior_likelihood`, `plot_sensitivity`, `plot_tornado`, `plot_tau_posterior`
9. **Shiny app** – `run_app`

Distribution families

`beta` Response rates and proportions – support (0, 1)

`normal` Mean differences and log odds ratios – support (-Inf, Inf)

`gamma` Event rates and median survival – support (0, Inf)

`lognormal` Hazard ratios and PK parameters – support (0, Inf)

`exponential` Constant hazard rates and Poisson rate priors – support (0, Inf). Conjugate with Poisson and survival data via Gamma-Poisson/Exponential updating.

`weibull` Non-constant hazard survival times (OS, PFS) – support (0, Inf). Posterior approximated via Normal matching.

Data types for conflict diagnostics and sensitivity

`"binary"` Events / sample size (x, n). Conjugate: Beta-Binomial.

`"continuous"` Observed mean, SD, sample size (x, sd, n). Conjugate: Normal-Normal.

`"poisson"` Event count / exposure person-time (x, n). Conjugate: Gamma-Poisson.

`"survival"` Events / total follow-up time (x, n). Conjugate: Gamma-Exponential.

References

- O'Hagan et al. (2006). *Uncertain Judgements*. Wiley.
- Box (1980). JRSS-A, 143, 383–430.
- Greenland (2023). *Scandinavian Journal of Statistics*, 50, 54–88.
- Oakley & O'Hagan (2010). *SHELF: the Sheffield Elicitation Framework*. https://tonyohagan.co.uk/shelf/.
- Schmidli et al. (2014). *Biometrics*, 70, 1023–1032.
- Roever et al. (2021). *Research Synthesis Methods*, 12, 448–474.
- Neuenschwander et al. (2010). *Clinical Trials*, 7, 5–18.
- Ibrahim & Chen (2000). *Statistical Science*, 15, 46–60.
- Spiegelhalter et al. (1994). JRSS-A, 157, 357–416.
- FDA (2026). Use of Bayesian Methodology in Clinical Trials of Drug and Biological Products.

Author(s)

Maintainer: Ndoh Penn <ndohpenn9@gmail.com>

See Also

Useful links:

- https://github.com/ndohpenngit/bayprior
- Report bugs at https://github.com/ndohpenngit/bayprior/issues

---

`calibrate_power_prior` *Calibrate power prior weight via Bayes Factor*

Description

Selects the power prior weight $\delta \in (0, 1)$ that down-weights historical data before incorporating it into the current analysis.

Usage

```
calibrate_power_prior(
  historical_data,
  current_data,
  base_prior,
  target_bf = 3,
  delta_grid = seq(0.05, 1, by = 0.05),
  method = c("bayes_factor", "compatibility")
)
```

Arguments

historical_data	Named list: type, x, n, optionally sd.
current_data	Named list (same structure as historical_data).
base_prior	A bayprior object (usually a vague prior).
target_bf	Numeric. Target Bayes Factor. Default 3.
delta_grid	Numeric vector of δ values. Default seq(0.05, 1.0, by = 0.05).
method	Character. "bayes_factor" (default) or "compatibility".

Value

A list of class bayprior_power_prior with components:

delta_opt Optimal power prior weight selected by the chosen method.

delta_grid The grid of delta values evaluated.

bf_grid Bayes Factor at each delta value.

compatibility_grid Box p-value at each delta value.

results Data frame with all diagnostic metrics across the grid.

power_prior A bayprior object updated with the optimal delta.

target_bf The target Bayes Factor supplied by the user.

method The calibration method used.

References

Ibrahim, J. G. & Chen, M.-H. (2000). Power prior distributions for regression models. *Statistical Science*, 15, 46-60.

Gravestock, I. & Held, L. (2017). Adaptive power priors with empirical Bayes for clinical trials. *Pharmaceutical Statistics*, 16, 349-360.

Examples

```
base  <- elicit_beta(mean = 0.5, sd = 0.2, method = "moments",
                     label = "Response rate")
calib <- calibrate_power_prior(
  historical_data = list(type = "binary", x = 12, n = 40),
  current_data    = list(type = "binary", x = 18, n = 50),
  base_prior      = base,
  target_bf       = 3
)
print(calib)
plot(calib)
```

---

conflict_mahalanobis *Multivariate prior-data conflict via Mahalanobis distance*

Description

Tests joint prior-data conflict across two correlated endpoints using the Mahalanobis distance. Under the null (no conflict) the squared distance follows a chi-squared distribution with degrees of freedom equal to the number of endpoints.

Usage

```
conflict_mahalanobis(
  prior_means,
  prior_cov,
  obs_means,
  obs_cov,
  alpha = 0.05,
  labels = NULL
)
```

Arguments

prior_means	Numeric vector of length k. Prior means for each endpoint.
prior_cov	k x k numeric matrix. Prior covariance matrix.
obs_means	Numeric vector of length k. Observed data means.
obs_cov	k x k numeric matrix. Observed data covariance (Var/n for each diagonal; Cov/n for off-diagonal).
alpha	Numeric. Significance level for the chi-squared test. Default 0.05.
labels	Character vector of length k. Endpoint labels for output.

Details

Assumptions: This test assumes that the prior and observed summary statistics are approximately multivariate Normal. For proportion endpoints (e.g. response rates), transform to the **log-odds scale** before entering means and variances. For hazard ratios, use the **log scale**. Results may be unreliable if the Normal approximation is poor (e.g. for small samples with extreme proportions).

Current limitation: The function is designed for bivariate (k = 2) endpoints. While it will accept k > 2, the Shiny interface currently only exposes two endpoints. Support for k >= 3 is a planned extension.

Distribution family: The Mahalanobis approach is distribution-agnostic at the summary statistic level – it does not require a specific prior family (Beta, Normal, etc.). Any continuous prior whose mean and covariance can be extracted is supported.

Value

A named list of class `bayprior_conflict_mv` with components:

mahal_distance Mahalanobis distance D.

mahal_D2 Squared distance D^2.

pvalue Chi-squared p-value (df = p).

df Degrees of freedom (number of endpoints, p).

conflict_flag Logical. TRUE if pvalue < alpha.

marginal_z Named numeric vector of per-parameter marginal z-scores (difference standardised by the marginal predictive SD).

interpretation Character. Plain-language summary.

labels Endpoint labels.

alpha The significance level used.

References

Mahalanobis, P. C. (1936). On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India*, 2, 49–55.

Examples

```
pm   <- c(0.35, 0.60)
pcov <- matrix(c(0.010, 0.003, 0.003, 0.015), 2, 2)
om   <- c(0.55, 0.58)
ocov <- matrix(c(2e-4, 4e-5, 4e-5, 2e-4), 2, 2)
conflict_mahalanobis(pm, pcov, om, ocov, labels = c("Response rate", "OS rate"))
```

`elicit_beta` *Elicit a Beta prior via quantile matching or moment matching*

Description

Fits a Beta(alpha, beta) distribution to expert-specified quantiles or moments. Implements the structured elicitation framework of the SHELF methodology, supporting the documented, reproducible prior derivation called for in the FDA's 2026 draft guidance on Bayesian methods.

Usage

```
elicit_beta(
  quantiles = NULL,
  mean = NULL,
  sd = NULL,
  method = c("quantile", "moments"),
  expert_id = "Expert_1",
  label = "Unknown quantity",
  tol = 1e-06
)
```

Arguments

`quantiles`	Named numeric vector of quantile specifications, e.g. `c("0.05" = 0.1, "0.50" = 0.3, "0.95" = 0.6)`. At least two quantiles required.
`mean`	Optional numeric. Expert-specified mean for moment matching.
`sd`	Optional numeric. Expert-specified SD for moment matching.
`method`	Character. One of `"quantile"` (default) or `"moments"`.
`expert_id`	Character. Identifier for this expert's elicitation.
`label`	Character. Description of the quantity being elicited.
`tol`	Numeric. Optimisation tolerance. Default `1e-6`.

Details

For `method = "moments"`, the mean and SD are matched analytically to the Beta family using the standard method-of-moments identities:

$$\alpha = \bar{x}\left(\frac{\bar{x}(1-\bar{x})}{s^2} - 1\right), \quad \beta = (1-\bar{x})\left(\frac{\bar{x}(1-\bar{x})}{s^2} - 1\right)$$

where $\bar{x}$ is `mean` and s is `sd`. This requires $s^2 < \bar{x}(1-\bar{x})$; an SD too large for the given mean implies a negative α or β and raises an error. For `method = "quantile"`, α and β are instead found numerically by minimising squared error between the fitted quantiles and the expert-specified quantiles.

Value

An object of class `bayprior` with components:

`dist` "beta"

`params` Named list with `alpha` and `beta`

`method` Elicitation method used

`expert_id` Expert identifier

`label` Quantity label

`fit_summary` Summary statistics of fitted prior

Examples

```
# Expert believes response rate is ~30%, with 90% CI of [10%, 60%]
prior <- elicit_beta(
  quantiles = c("0.05" = 0.10, "0.50" = 0.30, "0.95" = 0.60),
  expert_id = "Expert_1",
  label     = "Response rate (treatment arm)"
)
print(prior)
plot(prior)

# Moment-based elicitation
prior_mom <- elicit_beta(
  mean   = 0.30, sd = 0.12,
  method = "moments",
  label  = "Response rate"
)
```

---

elicit_exponential *Elicit an Exponential prior via moments, rate, or quantile matching*

Description

Fits an Exponential(rate) prior from expert-specified moments or quantiles. Suitable for constant-hazard survival models and Poisson rate priors. The Exponential distribution is the conjugate prior likelihood for a Gamma prior on the rate parameter.

Usage

```
elicit_exponential(
  rate = NULL,
  mean = NULL,
  quantiles = NULL,
  method = c("moments", "rate", "quantile"),
  expert_id = "Expert",
  label = "Quantity"
)
```

Arguments

rate	Numeric > 0. Rate parameter (= 1 / mean). Used when method = "rate".
mean	Numeric > 0. Prior mean (= 1 / rate). Used when method = "moments".
quantiles	Named numeric vector with at least one interior quantile. Used when method = "quantile".
method	Character. One of "moments", "rate", or "quantile". Default "moments".
expert_id	Character. Identifier for the eliciting expert.
label	Character. Human-readable label for the quantity.

Details

The Exponential distribution has a single parameter $\lambda > 0$ (the rate). Its mean is $1/\lambda$ and its SD equals its mean.

Typical use cases:

Hazard rates OS/PFS hazard in oncology trials

Event rates Adverse event rates (events per person-time)

Poisson rate priors Conjugate prior for Poisson count data

Value

A bayprior object with dist = "exponential".

Examples

```
# Mean survival 20 months => hazard rate 1/20 = 0.05
p <- elicit_exponential(mean = 0.05, method = "moments",
                        label     = "Hazard rate",
                        expert_id = "Expert_1")
print(p)
plot(p)

# Direct rate specification
p2 <- elicit_exponential(rate = 0.10, method = "rate",
                         label = "AE rate per person-year")

# Quantile matching
p3 <- elicit_exponential(
  quantiles = c("0.25" = 0.02, "0.50" = 0.05, "0.75" = 0.10),
  method    = "quantile",
  label     = "Hazard rate"
)
```

`elicit_gamma` *Elicit a Gamma prior via quantile matching or moment matching*

Description

Elicit a Gamma prior via quantile matching or moment matching

Usage

```
elicit_gamma(
  quantiles = NULL,
  mean = NULL,
  sd = NULL,
  method = c("quantile", "moments"),
  expert_id = "Expert_1",
  label = "Unknown quantity",
  tol = 1e-06
)
```

Arguments

`quantiles`	Named numeric vector of quantiles. Values must be positive.
`mean`	Optional numeric. Expert mean.
`sd`	Optional numeric. Expert SD.
`method`	Character. `"quantile"` or `"moments"`.
`expert_id`	Character. Expert identifier.
`label`	Character. Quantity description.
`tol`	Numeric. Optimisation tolerance.

Details

For `method = "moments"`, mean and SD are matched to the Gamma's shape-rate parameterisation via

$$\text{shape} = \left(\frac{\bar{x}}{s}\right)^2, \quad \text{rate} = \frac{\bar{x}}{s^2}$$

Value

An object of class `bayprior`.

Examples

```
prior <- elicit_gamma(
  mean   = 5, sd = 2,
  method = "moments",
  label  = "Median OS (months)"
)
```

`elicit_lognormal` *Elicit a Log-Normal prior via quantile matching or moment matching*

Description

Fits a Log-Normal distribution to expert-specified quantiles or moments. Appropriate for positive-valued quantities such as hazard ratios, fold changes, median survival times, or PK parameters.

Usage

```
elicit_lognormal(
  quantiles = NULL,
  mean = NULL,
  sd = NULL,
  method = c("quantile", "moments"),
  expert_id = "Expert_1",
  label = "Unknown quantity",
  tol = 1e-06
)
```

Arguments

`quantiles`	Named numeric vector. Values must be strictly positive. E.g. `c("0.05" = 0.5, "0.50" = 2.0, "0.95" = 8.0)`.
`mean`	Optional numeric. Expert mean on the original scale.
`sd`	Optional numeric. Expert SD on the original scale.
`method`	Character. `"quantile"` (default) or `"moments"`.
`expert_id`	Character. Expert identifier.
`label`	Character. Quantity description.
`tol`	Numeric. Optimisation tolerance.

Details

For `method = "moments"`, the mean and SD (on the original, untransformed scale) are matched to the Log-Normal's own `meanlog`/`sdlog` parameterisation via

$$\text{sdlog} = \sqrt{\log\left(1 + \frac{s^2}{\bar{x}^2}\right)}, \quad \text{meanlog} = \log(\bar{x}) - \frac{\text{sdlog}^2}{2}$$

Value

An object of class `bayprior` with `dist = "lognormal"`.

Examples

```
prior <- elicit_lognormal(
  quantiles = c("0.05" = 0.40, "0.50" = 0.70, "0.95" = 1.20),
  label     = "Hazard ratio (treatment vs control)"
)
print(prior)
```

`elicit_mixture` *Elicit a mixture prior*

Description

Constructs a finite mixture prior from a list of component `bayprior` objects (e.g., from `elicit_beta`, `elicit_normal`). Mixing weights can be specified or estimated via linear pooling.

Usage

```
elicit_mixture(components, weights = NULL, label = "Mixture prior")
```

Arguments

`components`	List of `bayprior` objects.
`weights`	Numeric vector of mixing weights (must sum to 1). If `NULL`, equal weights are assigned.
`label`	Character. Label for the mixture prior.

Details

Same-family mixtures (e.g. Beta + Beta, Normal + Normal) have mixture means and SDs computed in closed form. **Cross-family mixtures** (e.g. Beta + Normal) also have closed-form mixture means and SDs, but the mixture *density* must be computed numerically by evaluating and summing the component densities on a grid. A warning is issued in this case:

```
"Components have different distribution families.
 Mixture densities computed numerically."
```

This warning is informative, not an error. The numerical approximation is accurate in the body of the distribution but may be less reliable at the tails, particularly for components with very different supports (e.g. a Beta defined on (0, 1) mixed with a Normal defined on (-Inf, Inf)). Use `suppressWarnings()` only when you have verified the mixture is appropriate for your use case.

Value

A `bayprior` object with `dist = "mixture"`.

Examples

```
p1  <- elicit_beta(mean = 0.2, sd = 0.08, method = "moments", expert_id = "E1")
p2  <- elicit_beta(mean = 0.4, sd = 0.10, method = "moments", expert_id = "E2")
mix <- elicit_mixture(list(p1, p2), weights = c(0.5, 0.5),
                      label = "Pooled prior")
```

---

elicit_normal *Elicit a Normal prior via quantile matching or moment matching*

Description

Elicit a Normal prior via quantile matching or moment matching

Usage

```
elicit_normal(
  quantiles = NULL,
  mean = NULL,
  sd = NULL,
  method = c("quantile", "moments"),
  expert_id = "Expert_1",
  label = "Unknown quantity",
  tol = 1e-06
)
```

Arguments

quantiles	Named numeric vector. E.g. `c("0.025" = -0.5, "0.50" = 0.2, "0.975" = 0.9)`.
mean	Optional numeric. Expert mean for moment matching.
sd	Optional numeric. Expert SD for moment matching.
method	Character. `"quantile"` or `"moments"`.
expert_id	Character. Expert identifier.
label	Character. Quantity description.
tol	Numeric. Optimisation tolerance.

Details

For `method = "moments"`, the Normal is parameterised directly by its mean and SD, so matching is exact with no transformation required: $\mu = \bar{x}$, $\sigma = s$.

Value

An object of class `bayprior`.

Examples

```
prior <- elicit_normal(
  quantiles = c("0.025" = -0.5, "0.50" = 0.2, "0.975" = 0.9),
  label     = "Log odds ratio"
)
```

`elicit_roulette` *Roulette-method elicitation (chip-allocation)*

Description

Implements the SHELF roulette method: the expert allocates a fixed number of "chips" across a set of pre-defined bins representing the range of the quantity. The resulting histogram is fitted to a parametric distribution.

Usage

```
elicit_roulette(
  chips,
  breaks,
  family = c("beta", "normal", "gamma", "lognormal"),
  expert_id = "Expert_1",
  label = "Unknown quantity"
)
```

Arguments

`chips`	Integer vector. Number of chips in each bin (left-to-right).
`breaks`	Numeric vector of length `length(chips) + 1` defining the bin edges.
`family`	Character. Distribution to fit. One of `"beta"`, `"normal"`, `"gamma"`, `"lognormal"`.
`expert_id`	Character. Expert identifier.
`label`	Character. Quantity description.

Details

In the Shiny app (`Prior elicitation tab`) the roulette grid is rendered interactively. This function provides the **fitting back-end** that can also be called programmatically when chips are known.

Chips are converted to relative frequencies, and bin midpoints are used as representative values. The chosen `family` is then fitted by minimising the weighted sum of squared CDF differences (a histogram-matching approach).

Value

A `bayprior` object fitted to the chip histogram.

References

Oakley, J. E. & O'Hagan, A. (2010). SHELF: the Sheffield Elicitation Framework. University of Sheffield.

Examples

```
# Expert places 0, 2, 5, 8, 5, 2, 1 chips across bins [0,.1,.2,...,.7]
prior <- elicit_roulette(
  chips   = c(0L, 2L, 5L, 8L, 5L, 2L, 1L),
  breaks  = seq(0, 0.7, by = 0.1),
  family  = "beta",
  label   = "Response rate"
)
print(prior)
```

`elicit_weibull` *Elicit a Weibull prior via moments, direct parameters, or quantile matching*

Description

Fits a Weibull(shape, scale) prior from expert-specified moments or quantiles. The Weibull distribution generalises the Exponential and is widely used for survival analysis with non-constant hazard.

Usage

```
elicit_weibull(
  shape = NULL,
  scale = NULL,
  mean = NULL,
  sd = NULL,
  quantiles = NULL,
  method = c("moments", "params", "quantile"),
  expert_id = "Expert",
  label = "Quantity"
)
```

Arguments

`shape`	Numeric > 0. Shape parameter k. Used when `method = "params"`.
`scale`	Numeric > 0. Scale parameter λ. Used when `method = "params"`.
`mean`	Numeric > 0. Prior mean. Used with `sd` when `method = "moments"`.
`sd`	Numeric > 0. Prior SD. Used with `mean` when `method = "moments"`.
`quantiles`	Named numeric vector with at least two interior quantiles. Used when `method = "quantile"`.

method	Character. One of "moments", "params", or "quantile". Default "moments".
expert_id	Character. Identifier for the eliciting expert.
label	Character. Human-readable label for the quantity.

Details

Parameterised as in R's stats::dweibull: shape k and scale λ, with mean $\lambda\Gamma(1 + 1/k)$ and variance $\lambda^2[\Gamma(1 + 2/k) - \Gamma(1 + 1/k)^2]$.

Shape parameter interpretation:

- $k < 1$: decreasing hazard (e.g. early mortality selecting out)
- $k = 1$: constant hazard (reduces to Exponential)
- $k > 1$: increasing hazard (e.g. ageing, post-surgical)

Value

A bayprior object with dist = "weibull".

Examples

```
# Moment matching: mean 20 months, SD 10 months
p <- elicit_weibull(mean = 20, sd = 10, method = "moments",
                    label     = "Survival time (months)",
                    expert_id = "Expert_1")
print(p)
plot(p)

# Direct parameters (shape = 2 = increasing hazard)
p2 <- elicit_weibull(shape = 2, scale = 20, method = "params",
                     label = "PFS (months)")

# Quantile matching (at least 2 required)
p3 <- elicit_weibull(
  quantiles = c("0.10" = 5, "0.50" = 18, "0.90" = 40),
  method    = "quantile",
  label     = "OS (months)"
)
```

---

historical_effect_sizes

Compute (y, se) inputs for map_prior() from raw historical-trial data

---

Description

`map_prior()` takes an already-computed point estimate and SE per historical trial; it has no opinion on where those came from. In practice they're usually derived from each trial's own reported summary statistics (event counts, arm means/SDs, ...), and **metafor**'s `escalc` is the standard tool for that conversion. This is a thin convenience wrapper: it maps `map_prior()`'s `outcome_type` vocabulary onto the matching **metafor** `measure`, calls `escalc()`, and returns a data frame with `y`/`se` columns ready to feed straight into `map_prior(y = ..., se = ...)`. It is not the only way to get there – `y`/`se` can equally be read by hand off a published point estimate and 95% CI (`se = (log(upper) - log(lower)) / (2 * 1.96)` on whatever scale the CI was reported) – and it is not loaded unless you have **metafor** installed.

Usage

```
historical_effect_sizes(outcome_type, data, ...)
```

Arguments

`outcome_type`	One of `map_prior()`'s outcome types (see `map_prior`); selects the **metafor** `measure` and which columns are required from `data` (see Details).
`data`	A data frame of raw per-trial summary statistics, with columns named exactly as **metafor** expects for the selected measure (see Details).
`...`	Additional arguments passed to `escalc` (e.g. `add`/`to` for zero-cell handling). Do **not** pass column-mapping arguments here (`xi`, `ai`, ...) – those are read from `data` by the fixed names in Details.

Details

`outcome_type` maps to a **metafor** `measure` as follows; see `escalc` for the columns each one requires:

`"log_or"` `"OR"` (log odds ratio) – 2x2 event counts `ai`, `bi`, `ci`, `di`.

`"smd"` `"SMD"` (Hedges' g) – arm `m1i`/`m2i`, `sd1i`/`sd2i`, `n1i`/`n2i`.

`"log_irr"` `"IRR"` (log incidence-rate ratio) – event counts and person-time `x1i`/`x2i`, `t1i`/`t2i`.

`"mean_difference"` `"MD"` (raw mean difference) – same columns as `"smd"`.

`"single_arm_log_odds"` `"PLO"` (logit of a single proportion) – event count `xi` out of `ni`. This is the typical case: a historical control-arm response rate.

`"correlation"` `"ZCOR"` (Fisher z) – correlation `ri` and sample size `ni`.

`escalc()`'s own zero-cell handling (`add`/`to`) applies as usual and can be overridden via `....`

Required columns must be named exactly as **metafor** expects for that measure (see Details) – `data` is read by column name, not position, and extra columns are carried through untouched.

Value

`data` with two columns appended: `y` (the point estimate) and `se` (its standard error) – pass these directly as `map_prior(y = es$y, se = es$se, outcome_type = ...)`.

See Also

map_prior, escalc

Examples

```
if (requireNamespace("metafor", quietly = TRUE)) {
  # Historical control-arm event counts -> single-arm log-odds.
  trials <- data.frame(xi = c(12, 8, 15), ni = c(40, 35, 50))
  es <- historical_effect_sizes("single_arm_log_odds", trials)
  print(es[, c("y", "se")])
  map <- map_prior(y = es$y, se = es$se, outcome_type = "single_arm_log_odds")
  print(map)
}
```

---

map_prior *Derive a meta-analytic-predictive (MAP) prior from historical trials*

Description

Combines summary-level data from multiple historical (or supplementary) trials into an informative prior for the response parameter of a new trial, using a random-effects meta-analysis with an explicit prior on the between-trial heterogeneity parameter τ. This is the meta-analytic-predictive (MAP) approach of Schmidli et al. (2014) and Neuenschwander et al. (2010).

Usage

```
map_prior(
  y,
  se,
 outcome_type = c("log_or", "smd", "log_irr", "mean_difference", "single_arm_log_odds",
    "correlation"),
  tau_prior = NULL,
  label = "MAP prior",
  tau_grid_n = 400,
  mu_prior_sd = 10000
)
```

Arguments

y	Numeric vector of historical trial point estimates (one per trial), on the scale implied by outcome_type.
se	Numeric vector of standard errors for y, same length.
outcome_type	One of "log_or" (two-arm log-odds ratio), "smd" (standardised mean difference), "log_irr" (log incidence-rate ratio), "mean_difference" (raw, unstandardised mean difference), "single_arm_log_odds" (log-odds of a single-arm proportion – the typical case for historical *control-arm* response rates), or "correlation".

Unless tau_prior is supplied explicitly, this selects the default τ prior for this outcome type (see resolve_tau_prior for the source and rationale). **This choice materially changes the result**: "log_or" and "single_arm_log_odds" are both "log-odds" in casual speech but get different default scales (0.5 vs 1.0) because they are different quantities with different heterogeneity properties – conflating them is not a labelling nicety. "mean_difference" has no default: its units are entirely endpoint-specific (mmHg, cm, days, ...), so tau_prior must be supplied explicitly for it.

tau_prior — Optional list with elements family ("half_normal" or "half_cauchy") and scale (positive numeric), specifying the prior on the heterogeneity parameter τ. If NULL (default), resolved from outcome_type via resolve_tau_prior. An explicit value here always overrides the outcome_type preset.

label — Character label for the resulting prior (used in plots, prints, and reports). Default "MAP prior".

tau_grid_n — Number of points used for the **plotting** grid stored in $tau_posterior (its upper bound is chosen automatically from the fitted posterior's 99.9th percentile, located by root-finding, not a fixed constant, so it adapts to the prior's actual tail). This does **not** affect inference – that is handled separately by adaptive integration (see Details). Default 400.

mu_prior_sd — SD of the (effectively flat, vague) Normal prior placed on the meta-analytic mean μ. Default 1e4, vague relative to any realistic effect-size scale; increase only if 1e4 units is not actually vague for your outcome.

Details

Unlike robust_prior, which mixes an *already-specified* informative prior with a vague component for robustness, map_prior() *derives* an informative prior from historical data in the first place. Use map_prior() to turn historical trial summaries into a prior; use robust_prior() afterwards if you want to add a vague/robust mixture component to that (or any other) informative prior.

Model. Trial i contributes $y_i \sim \text{Normal}(\theta_i, se_i^2)$, with random effects $\theta_i \sim \text{Normal}(\mu, \tau^2)$ and a proper (vague) prior $\mu \sim \text{Normal}(0, \sigma_\mu^2)$, where σ_μ is mu_prior_sd. Marginalising θ_i gives $y_i \sim \text{Normal}(\mu, se_i^2 + \tau^2)$ independently across trials – the standard random-effects meta-analysis model. An explicit prior on τ (half-normal or half-Cauchy, via tau_prior or an outcome_type preset) follows the weakly-informative recommendations of Roever et al. (2021).

Computation. map_prior() has no external statistical dependency; the posterior of μ given τ is available in closed form (a precision-weighted Normal), so the only numerical step is a one-dimensional integral over $\tau \in [0, \infty)$, evaluated with integrate (adaptive quadrature).

The full τ posterior (not just its effect on μ) is retained in the returned object's $tau_posterior element, and can be visualised with plot_tau_posterior or exported as part of prior_report.

Value

A bayprior object (dist = "normal", matched by moments to the marginal posterior of μ) with an additional prior_type = "map" tag and a $tau_posterior element (a data frame with columns tau and density) giving the marginal posterior of the heterogeneity parameter.

References

Schmidli, H., Gsteiger, S., Roychoudhury, S., O'Hagan, A., Spiegelhalter, D., & Neuenschwander, B. (2014). Robust meta-analytic-predictive priors in clinical trials with historical control information. *Biometrics*, 70(4), 1023-1032.

Roever, C., Bender, R., Dias, S., Schmid, C. H., Schmidli, H., Sturtz, S., Weber, S., & Friede, T. (2021). On weakly informative prior distributions for the heterogeneity parameter in Bayesian random-effects meta-analysis. *Research Synthesis Methods*, 12(4), 448-474.

Neuenschwander, B., Capkun-Niggli, G., Branson, M., & Spiegelhalter, D. J. (2010). Summarizing historical information on controls in clinical trials. *Clinical Trials*, 7(1), 5-18.

See Also

`resolve_tau_prior()` for the default heterogeneity priors, `plot_tau_posterior()` to inspect tau, `historical_effect_sizes()` to build y/se from raw trial summaries, and `robust_prior()` to add a vague-mixture component.

Examples

```
# Historical control-arm response rates (single-arm log-odds) --
# the typical Schmidli et al. (2014) MAP use case.
map <- map_prior(
  y  = c(-0.85, -0.62, -1.10),
  se = c(0.25, 0.30, 0.28),
  outcome_type = "single_arm_log_odds",
  label = "Historical control (3 trials)"
)
print(map)
plot_tau_posterior(map)
```

---

plot.bayprior_power_prior

Plot calibration curve for power prior weight selection

Description

Plot calibration curve for power prior weight selection

Usage

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

Arguments

x	A bayprior_power_prior object.
...	Ignored.

Value

A ggplot object, or a list of two ggplots if 'patchwork' is not installed.

---

plot_prior_likelihood *Plot prior, likelihood, and posterior density overlays*

Description

Plot prior, likelihood, and posterior density overlays

Usage

```
plot_prior_likelihood(
  prior,
  data_summary,
  show_posterior = TRUE,
  show_conflict = TRUE,
  n_grid = 500,
  title = NULL
)
```

Arguments

prior	A bayprior object from any elicit_*() function.
data_summary	Named list with n, x, optionally sd and type.
show_posterior	Logical. Default TRUE.
show_conflict	Logical. Default TRUE.
n_grid	Integer. Default 500.
title	Character. Plot title.

Value

A ggplot object.

Examples

```
prior <- elicit_beta(mean = 0.30, sd = 0.10, method = "moments",
                     label = "Response rate")
plot_prior_likelihood(prior, list(n = 40, x = 20, type = "binary"))
```

plot_sensitivity *Plot sensitivity analysis results*

Description

Plot sensitivity analysis results

Usage

```
plot_sensitivity(
  sensitivity,
  target = NULL,
  highlight_reference = TRUE,
  interactive = FALSE
)
```

Arguments

sensitivity	A bayprior_sensitivity object.
target	Character. Which target quantity to plot.
highlight_reference	Logical. Default TRUE.
interactive	Logical. If TRUE, return an interactive **plotly** object with readable hover text instead of a static ggplot. Default FALSE.

Value

A ggplot object, or a plotly object when interactive = TRUE.

plot_tau_posterior *Plot the posterior distribution of the heterogeneity parameter (tau)*

Description

Visualises the marginal posterior density of the between-trial heterogeneity parameter τ from a prior fitted with map_prior, together with its prior density for comparison (so the amount of learning about heterogeneity is visible directly).

Usage

```
plot_tau_posterior(prior, show_prior = TRUE)
```

Arguments

prior	A bayprior object with a non-NULL $tau_posterior element (i.e. returned by map_prior).
show_prior	Logical; overlay the tau prior density for comparison. Default TRUE. Has no effect if prior$tau_prior is NULL.

Value

A ggplot object.

See Also

map_prior(), which returns the object plotted here, and resolve_tau_prior().

Examples

```
map <- map_prior(
  y = c(-0.85, -0.62, -1.10), se = c(0.25, 0.30, 0.28),
  outcome_type = "single_arm_log_odds"
)
plot_tau_posterior(map)
```

---

plot_tornado *Tornado plot of prior influence on posterior quantities*

Description

Tornado plot of prior influence on posterior quantities

Usage

```
plot_tornado(
  sensitivity,
  title = "Prior influence on posterior estimates",
  interactive = FALSE
)
```

Arguments

sensitivity	A bayprior_sensitivity object.
title	Character. Plot title.
interactive	Logical. If TRUE, return an interactive **plotly** object with readable hover text instead of a static ggplot. Default FALSE.

Value

A ggplot object, or a plotly object when interactive = TRUE.

print.bayprior *Print method for bayprior objects*

Description

Print method for bayprior objects

Usage

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

Arguments

x	A bayprior object.
...	Ignored.

Value

Invisibly returns the input bayprior object. Called for its side effect of printing a formatted summary of the prior distribution including family, parameters, mean, SD, and 95\ credible interval.

print.bayprior_conflict *Print method for bayprior_conflict objects*

Description

Print method for bayprior_conflict objects

Usage

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

Arguments

x	A bayprior_conflict object.
...	Ignored.

Value

Invisibly returns the input bayprior_conflict object. Called for its side effect of printing conflict diagnostic statistics including Box p-value, S-value (surprisal), surprise index, information divergence, Bhattacharyya overlap, and colour-coded conflict severity.

print.bayprior_conflict_mv

Print method for multivariate conflict objects

Description

Print method for multivariate conflict objects

Usage

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

Arguments

x	A bayprior_conflict_mv object.
...	Ignored.

Value

Invisibly returns the input bayprior_conflict_mv object. Called for its side effect of printing a formatted summary of the multivariate Mahalanobis conflict check, including the Mahalanobis distance, chi-squared p-value, conflict flag, per-parameter marginal z-scores, and an interpretation string.

print.bayprior_power_prior

Print method for bayprior_power_prior objects

Description

Print method for bayprior_power_prior objects

Usage

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

Arguments

x	A bayprior_power_prior object.
...	Ignored.

Value

Invisibly returns the input bayprior_power_prior object. Called for its side effect of printing a formatted summary of the power prior calibration, including the calibration method, target Bayes Factor, optimal delta weight, and the mean and SD of the resulting power prior.

`prior_conflict` *Compute prior-data conflict diagnostics*

Description

Evaluates conflict between a specified prior and observed data using multiple complementary diagnostics: Box's (1980) predictive p-value, the surprise index (standardised distance), Kullback-Leibler divergence, and the Bhattacharyya overlap coefficient between the prior and the (normalised) likelihood.

Usage

```
prior_conflict(prior, data_summary, alpha = 0.05, exact = NULL)
```

Arguments

`prior`	A bayprior object.
`data_summary`	Named list describing the observed data: `type` `"binary"`, `"continuous"`, `"poisson"`, or `"survival"`. `x` Number of events (binary / poisson / survival) or observed mean (continuous). `n` Sample size (binary / continuous), total exposure (poisson: person-time), or total follow-up time (survival). `sd` Observed standard deviation (continuous only).
`alpha`	Numeric. Significance level for the Box p-value flag. Default `0.05`.
`exact`	`NULL` (the default), `TRUE`, or `FALSE`. Controls how `box_pvalue` (and the `s_value` derived from it) is computed. `NULL` **(default)** Use the **exact** prior predictive distribution whenever a closed form exists, and the Normal approximation otherwise, silently. Check `$box_pvalue_method` on the returned object to see which was used. `TRUE` As `NULL`, but signal a message when the prior/data combination has no closed-form check and the Normal approximation is used instead. `FALSE` Always use the Normal approximation to both the prior and the data likelihood (the behaviour of bayprior 0.4.0 and earlier). Closed forms exist for a Beta prior with `type = "binary"` data (Beta-Binomial) and for a Gamma prior with `type = "poisson"` or `type = "survival"` data (Negative Binomial; Gamma-Poisson and Gamma-Exponential conjugacy are the same model under this package's (events, exposure) parameterisation). The exact check also requires whole-number counts; otherwise the Normal approximation is used. `surprise_index`, `kl_prior_likelihood`, and `overlap` are unaffected by `exact` – they are defined with respect to the Normal-approximated prior and likelihood (see Details).

Details

s_value and surprise_index are easily confused by name but measure different things. surprise_index is a *distance*: how many prior-SD units the observed data sits from the prior mean (abs(z)). s_value is an *evidence-strength* reparametrisation of box_pvalue itself, expressed in bits – intended as an exploratory companion to box_pvalue rather than a thresholded decision rule, since prior-data conflict checks are typically not powered for a binary accept/reject decision. The two are correlated but nonlinearly related: s_value grows faster than surprise_index as evidence strengthens, since it derives from box_pvalue's tail probability, not from z directly.

Mixture priors. A linear mixture of Beta (binary data) or Gamma (Poisson/survival data) components, such as a robust_prior or a linearly pooled consensus from aggregate_experts, has as prior predictive the weighted mixture of its components' predictives, so its p-value is exact as well. severity is classified on the reported box_pvalue: "mild" for alpha > p >= 0.0027 and "severe" below 0.0027 (the two-sided Normal tail beyond 3 standard deviations).

The numerical integration uses 4,000 (count data) to 40,000 (binary data, small n) quadrature points, so the p-value is accurate to roughly 1e-5 in the predictive probabilities. Run time grows with the number of possible counts: well under a second typically, a few seconds when the count support reaches several thousand (for example a heavy-tailed prior with exposure in the hundreds), where the Normal approximation is also adequate.

Why exact is the default. The Normal approximation estimates the sampling standard error from the observed data (for example $\sqrt{\hat{p}(1-\hat{p})/n}$), which is zero when no events are observed. For an informative prior and a small sample, an observation of zero events can be common under the prior predictive distribution yet receive a very small approximate p-value, so the approximate check can flag conflict far more often than its nominal level when there is none. The exact prior predictive p-value does not have this problem: it is computed from the prior predictive distribution itself, and its rejection rate under no conflict is at most the nominal level (it can be below it because the data are discrete). The two agree closely for moderate samples and weakly informative priors. Use exact = FALSE to reproduce the Normal-approximation results of earlier versions.

The exact p-value is the prior predictive check of Box (1980) in the form used by Evans and Moshonov (2006): the total prior predictive probability of the outcomes that are no more probable than the observed one. The Normal approximation with a plug-in standard error is a convenience approximation to it, not part of either definition.

What the check can and cannot tell you. In the package's simulation study (see the validation folder of the source repository), the check had little power against true values within about one prior standard deviation of the prior mean, whichever version was used, so a non-significant result is weak evidence that a prior is adequate. For skewed priors the probability ordering can also be less sensitive to true values below the prior mean than to values above it, because low counts (including zero) are probable under a right-skewed prior predictive distribution. Use the result together with sensitivity_grid and judgement about the source of the prior, not as a stand-alone test.

Value

An object of class bayprior_conflict containing:

box_pvalue Box's prior predictive p-value.

box_pvalue_method "exact" (closed-form prior predictive), "numerical" (the same prior predictive p-value, with the predictive computed by quadrature) or "normal_approx", indicat-

ing how box_pvalue (and hence s_value) was computed. Always "normal_approx" when exact = FALSE.

box_pvalue_caution NULL, or a short message when the Normal approximation produced box_pvalue for discrete data with a boundary (zero or all events) or small (fewer than 10) counts, where it can be anti-conservative.

s_value Surprisal, $-\log_2(\text{box_pvalue})$, in bits. See Details.

surprise_index Standardised distance between prior mean and observed data.

kl_prior_likelihood KL divergence from prior to likelihood.

overlap Bhattacharyya overlap coefficient in [0, 1].

conflict_severity One of "none", "mild", "severe".

conflict_flag Logical; TRUE if box_pvalue < alpha.

recommendation Plain-language guidance string.

data_summary The data summary passed in.

prior The input prior.

References

Evans, M. and Moshonov, H. (2006). Checking for prior-data conflict. *Bayesian Analysis*, 1(4).

Box, G. E. P. (1980). Sampling and Bayes' inference in scientific modelling and robustness. *Journal of the Royal Statistical Society A*, 143, 383-430.

Greenland, S. (2023). Divergence versus decision P-values: A distinction worth making in theory and keeping in practice. *Scandinavian Journal of Statistics*, 50(1), 54-88.

Examples

```
prior <- elicit_beta(mean = 0.30, sd = 0.10, method = "moments",
                     label = "Response rate")
cd <- prior_conflict(prior, list(type = "binary", x = 18, n = 40))
print(cd)
```

---

prior_report *Generate a Prior Justification Report*

Description

Renders an HTML, PDF, or Word (.docx) report using quarto::quarto_render().

Usage

```
prior_report(
  prior,
  conflict = NULL,
  sensitivity = NULL,
  robust_prior = NULL,
  sceptical_prior = NULL,
  power_prior = NULL,
  output_format = c("html", "pdf", "docx"),
  output_file = NULL,
  trial_name = "Clinical Trial",
  sponsor = "Sponsor",
  date = as.character(Sys.Date()),
  author = "Biostatistics",
  notes = "",
  prior_plot = NULL,
  overlay_plot = NULL,
  tornado_plot = NULL,
  heatmap_plot = NULL,
  robust_plot = NULL,
  sceptical_plot = NULL,
  power_plot = NULL,
  tau_posterior_plot = NULL,
  open_after = interactive()
)
```

Arguments

prior	A bayprior object.
conflict	Optional bayprior_conflict from prior_conflict().
sensitivity	Optional bayprior_sensitivity from sensitivity_grid().
robust_prior	Optional output of robust_prior().
sceptical_prior	Optional output of sceptical_prior().
power_prior	Optional output of calibrate_power_prior().
output_format	"html" (default), "pdf", or "docx".
output_file	Output path without extension.
trial_name	Trial / protocol identifier.
sponsor	Sponsor name.
date	Report date string. Default Sys.Date().
author	Responsible statistician.
notes	Optional narrative text.
prior_plot	Optional pre-captured ggplot from `plot(prior)`.
overlay_plot	Optional pre-captured ggplot from `plot_prior_likelihood()`.

tornado_plot	Optional pre-captured ggplot from plot_tornado().
heatmap_plot	Optional pre-captured ggplot from plot_sensitivity().
robust_plot	Optional pre-captured ggplot of the robust prior.
sceptical_plot	Optional pre-captured ggplot of the sceptical prior.
power_plot	Optional pre-captured ggplot of the power prior.
tau_posterior_plot	Optional pre-captured ggplot from plot_tau_posterior(), shown when prior was derived with map_prior() (i.e. prior$prior_type == "map"). If omitted and prior is a MAP prior, the report attempts to render it directly from prior$tau_posterior.
open_after	Open after rendering. Default TRUE interactively.

Value

Path to the rendered report, invisibly.

---

resolve_tau_prior *Resolve the default heterogeneity (tau) prior for an outcome type*

Description

Looks up the weakly-informative τ prior Roever et al. (2021) recommend for a given outcome/effect scale. Used internally by map_prior whenever tau_prior is left NULL, and exported so the mapping can be inspected or reused directly.

Usage

```
resolve_tau_prior(outcome_type)
```

Arguments

outcome_type	One of "log_or", "smd", "log_irr", "single_arm_log_odds", "correlation". "mean_difference" is a valid outcome_type for map_prior but has no preset here (see Details) and will error if passed to this function.

Details

Verified directly against Roever et al. (2021) (not assumed): a half-normal(scale = 0.5) prior is their suggested weakly-informative default for **several** unit-free effect scales alike – two-arm log-odds ratios, standardised mean differences, and log incidence-rate ratios – it is *not* uniquely calibrated to "log-odds" as a from-scratch reading might suggest. The scale that actually differs is the log-odds of a **single-arm** proportion (e.g. a historical control response rate, the case Schmidli et al. 2014 built the MAP framework around), for which the recommended scale is half-normal(scale = 1.0) – twice as wide. Conflating the two log-odds cases under one generic "log_odds" label, as earlier versions of this function did, silently used the wrong default for whichever of the two was not the log-OR case.

"mean_difference" (raw, unstandardised) has no listed preset: its units are entirely endpoint-specific, so map_prior requires tau_prior to be supplied explicitly for it rather than guessing.

Value

A list with elements `family` and `scale`.

See Also

`map_prior()`, which uses these defaults, and `plot_tau_posterior()`.

Examples

```
resolve_tau_prior("log_or")
resolve_tau_prior("single_arm_log_odds")
```

---

robust_prior *Construct a robust (heavy-tailed mixture) prior*

Description

Builds a robust prior by mixing an informative component with a vague (diffuse) component, following the RBesT/MAP robust mixture approach (Schmidli et al., 2014). This protects against prior misspecification by ensuring the posterior is not dominated by a conflicting informative prior.

Usage

```
robust_prior(
  informative,
  vague_weight = 0.2,
  vague_sd = NULL,
  label = "Robust mixture prior"
)
```

Arguments

`informative`	A `bayprior` object representing the informative component (e.g. an elicited or historical prior).
`vague_weight`	Numeric in (0, 1). Weight assigned to the vague (diffuse) component. Default `0.20` (80% informative, 20% vague).
`vague_sd`	Numeric. SD of the vague Normal component (on the natural scale). If `NULL`, defaults to 10x the informative prior's SD.
`label`	Character. Label for the robust prior.

Details

The vague component is always a **Normal distribution** centred at the informative prior's mean with SD = `vague_sd` (default: 10x the informative SD). When the informative prior is itself Normal, both components share the same family and the mixture density is computed analytically. For any other informative prior family (Beta, Gamma, Log-Normal, Exponential, Weibull), the components have *different distribution families*, and the mixture density is **computed numerically**. A warning is issued in this case.

Value

A bayprior object with dist = "mixture" and prior_type = "robust".

References

Schmidli, H. et al. (2014). Robust meta-analytic-predictive priors in clinical trials with historical control information. *Biometrics*, 70, 1023-1032.

Examples

```
informative <- elicit_normal(mean = 0.30, sd = 0.10,
                             method = "moments", label = "Response rate")
robust      <- robust_prior(informative, vague_weight = 0.20)
plot(robust)
```

---

run_app *Run the bayprior Shiny Application*

Description

Launches the golem-structured shinydashboard application.

Usage

```
run_app(
  onStart = NULL,
  options = list(),
  enableBookmarking = NULL,
  uiPattern = "/",
  ...
)
```

Arguments

onStart	A function called before the app runs.
options	List of options passed to shiny::shinyApp().
enableBookmarking	Passed to shiny::shinyApp().
uiPattern	Passed to shiny::shinyApp().
...	Additional arguments passed to golem::with_golem_options().

Value

A shiny.appobj, invisibly.

`sceptical_prior` *Construct a sceptical (penalised-enthusiasm) prior*

Description

Generates a sceptical prior that places most mass at or near the null value of the treatment effect, representing a conservative stance for regulatory submissions. Implements the Spiegelhalter-Freedman sceptical prior approach, commonly paired with an enthusiastic prior as a sensitivity-analysis pair in regulatory submissions.

Usage

```
sceptical_prior(
  null_value = 0,
  family = c("normal", "beta", "lognormal"),
  strength = c("moderate", "weak", "strong"),
  label = "Treatment effect",
  expert_id = "Sceptic"
)
```

Arguments

`null_value`	Numeric. The null treatment effect. For `family = "normal"` or `family = "beta"`, provide this on the natural scale (e.g. `0` for a mean difference, `0.5` for a response-rate difference). For `family = "lognormal"`, provide this on the **log scale** (e.g. `0` for a hazard ratio null of 1, since `log(1) = 0`) – `null_value = 1` for `family = "lognormal"` centres the prior at `exp(1) ~ 2.72`, **not** at a hazard ratio of 1. This mirrors how `0` already represents "no difference" for the normal family; it does not carry over to the natural (ratio) scale for lognormal. For `family = "beta"`, must be strictly in (0, 1).
`family`	Character. Distribution family. One of `"normal"`, `"beta"`, `"lognormal"`.
`strength`	Character. How concentrated the prior is around the null: `"weak"`, `"moderate"` (default), or `"strong"`.
`label`	Character. Description of the quantity.
`expert_id`	Character. Identifier for provenance.

Details

For a Normal family, the sceptical prior is centred at `null_value` with SD calibrated to the `strength` argument:

- `weak`: SD = 1.0 (vague half-normal)
- `moderate`: SD = 0.5 (2-SD departure from null has ~5% prior probability)
- `strong`: SD = 0.25 (very concentrated at null)

For `family = "beta"`, `null_value` must be in (0, 1).

Value

A bayprior object tagged with prior_type = "sceptical".

References

Spiegelhalter, D. J., Freedman, L. S. & Parmar, M. K. B. (1994). Bayesian approaches to randomized trials. *JRSS-A*, 157, 357-416.

Examples

```
sc <- sceptical_prior(null_value = 0, family = "normal",
                      strength = "moderate", label = "Mean difference")
print(sc)
plot(sc)

# Beta sceptical prior centred at a null response rate
sc_b <- sceptical_prior(null_value = 0.20, family = "beta",
                        strength = "moderate", label = "Response rate")
plot(sc_b)

# Lognormal sceptical prior for a hazard ratio: null_value is on the LOG
# scale, so 0 (not 1) represents a null hazard ratio of exp(0) = 1.
sc_hr <- sceptical_prior(null_value = 0, family = "lognormal",
                         strength = "moderate", label = "Hazard ratio")
print(sc_hr)  # mean should be at or near 1, the null hazard ratio
```

---

sensitivity_cri *Credible interval sensitivity over prior hyperparameters*

Description

Evaluates how the posterior credible interval width and bounds change as prior hyperparameters vary over a specified grid. This is the preferred function for demonstrating that key regulatory conclusions (e.g., whether the CrI excludes a null value) are robust to prior choice.

Usage

```
sensitivity_cri(
  prior,
  data_summary,
  param_grid,
  cri_level = 0.95,
  threshold = NULL
)
```

Arguments

prior	A bayprior object (the reference prior).
data_summary	Named list as for prior_conflict.
param_grid	Named list of numeric vectors, one per hyperparameter.
cri_level	Numeric in (0, 1). Credible interval level. Default 0.95.
threshold	Optional numeric. Computes Pr(theta > threshold) at each grid point if supplied.

Details

As with sensitivity_grid, when prior is a mixture, a single-family working prior is derived via .mixture_working_prior() – moment-matching the mixture's pooled mean/SD to the dominant component's family – rather than analysing the dominant component alone.

Value

An object of class bayprior_sensitivity whose grid contains columns cri_lower, cri_upper, and cri_width, plus optionally posterior_mean, posterior_sd, and prob_efficacy.

Examples

```
prior <- elicit_beta(mean = 0.30, sd = 0.10, method = "moments",
                     label = "Response rate")
cri_sa <- sensitivity_cri(
  prior,
  data_summary = list(type = "binary", x = 14, n = 40),
  param_grid   = list(alpha = seq(1, 8, 0.5), beta = seq(2, 20, 1)),
  cri_level    = 0.95
)
plot_sensitivity(cri_sa, target = "cri_width")
```

---

sensitivity_grid *Sensitivity grid over prior hyperparameters*

Description

Evaluates how posterior inferences change as prior hyperparameters vary over a specified grid. This is the core function for demonstrating robustness of trial conclusions to prior choice.

Usage

```
sensitivity_grid(
  prior,
  data_summary,
  param_grid,
  target = c("posterior_mean", "posterior_sd", "prob_efficacy"),
  threshold = 0.3
)
```

Arguments

prior	A bayprior object (the reference prior).
data_summary	Named list as for prior_conflict.
param_grid	Named list of numeric vectors, one per hyperparameter to vary. Names must match hyperparameter names in prior$params. Example: list(alpha = seq(1, 8, 0.5), beta = seq(2, 20, 1)).
target	Character vector. Which posterior quantities to compute. Any of "posterior_mean", "posterior_sd", "prob_efficacy".
threshold	Numeric. Efficacy threshold used in Pr(theta > threshold). Default 0.30.

Details

The hyperparameter grid is defined over a single distribution family, so when prior is a mixture (e.g. from aggregate_experts or robust_prior), a single-family *working prior* is derived first, via the internal .mixture_working_prior() helper. As of this version, that working prior is obtained in two steps: first, the mixture's pooled mean and SD are computed exactly (see the mean/var formula in aggregate_experts); second, those pooled moments are matched to the dominant component's distribution family using that family's own moment-matching identities (see elicit_beta, elicit_normal, elicit_gamma, or elicit_lognormal for the specific formula used). The grid therefore reflects the full pooled information, not just one component. A message reports the working prior used. If the dominant component's family cannot be moment-matched from mean/SD alone ("exponential" or "weibull"), the function falls back to the dominant component by weight and issues an explicit warning identifying which component was used and why.

Value

An object of class bayprior_sensitivity.

Examples

```
prior <- elicit_beta(mean = 0.30, sd = 0.10, method = "moments",
                     label = "Response rate")
sa <- sensitivity_grid(
  prior,
  data_summary = list(type = "binary", x = 14, n = 40),
  param_grid   = list(alpha = seq(1, 8, 0.5), beta = seq(2, 20, 1))
)
plot_tornado(sa)
plot_sensitivity(sa, target = "posterior_mean")
```

---

summary.bayprior *Summary method for bayprior objects*

Description

Summary method for bayprior objects

Usage

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

Arguments

`object`	A `bayprior` object.
`...`	Ignored.

Value

A list with summary statistics (invisibly).

Index

aggregate_experts, 3, *6*, *31*, *40*
as_prior, 5, *6*

bayprior, 6
bayprior-package *(*bayprior*)*, 6

calibrate_power_prior, *6*, 7
conflict_mahalanobis, *6*, 9

elicit_beta, *6*, 11, *40*
elicit_exponential, *6*, 12
elicit_gamma, *6*, 14, *40*
elicit_lognormal, *6*, 15, *40*
elicit_mixture, *6*, 16
elicit_normal, *6*, 17, *40*
elicit_roulette, *6*, 18
elicit_weibull, *6*, 19
escalc, *21, 22*

historical_effect_sizes, *6*, 20
historical_effect_sizes(), *24*

integrate, *23*

map_prior, *6*, *21, 22*, 22, *26, 27*, *34*
map_prior(), *27*, *35*

plot.bayprior_power_prior, 24
plot_prior_likelihood, *6*, 25
plot_sensitivity, *6*, 26
plot_tau_posterior, *6*, *23*, 26
plot_tau_posterior(), *24*, *35*
plot_tornado, *6*, 27
print.bayprior, 28
print.bayprior_conflict, 28
print.bayprior_conflict_mv, 29
print.bayprior_power_prior, 29
prior_conflict, *6*, 30, *39, 40*
prior_report, *6*, *23*, 32

resolve_tau_prior, *6*, *23*, 34
resolve_tau_prior(), *24*, *27*
robust_prior, *6*, *23*, *31*, 35, *40*
robust_prior(), *24*
run_app, *6*, 36

sceptical_prior, *6*, 37
sensitivity_cri, *6*, 38
sensitivity_grid, *4*, *6*, *31*, *39*, 39
summary.bayprior, 40