Package 'spima'

May 28, 2026

Title Simulated Pseudo-Individual Data Meta-Analysis with ABC-SMC

Version 0.2.0

Description Meta-analysis via Approximate Bayesian Computation Sequential Monte Carlo (ABC-SMC) by simulating pseudo-individual data from published group-level summary statistics. Handles binary, continuous, and generic effect-size outcomes within a one-stage mixed-model framework. Supports subgroup analysis.

License MIT + file LICENSE

Depends R (>= 3.5.0)

Imports lme4, parallel, stats, methods, Rcpp

Suggests ggplot2, testthat, knitr, rmarkdown

LinkingTo Rcpp, RcppArmadillo

SystemRequirements GNU make

URL https://github.com/HaichuanYu0703/SPIMA

BugReports https://github.com/HaichuanYu0703/SPIMA/issues

Encoding UTF-8

LazyData true

NeedsCompilation yes

Config/roxygen2/version 8.0.0

Author Yu Haichuan [aut, cre, cph]

Maintainer Yu Haichuan <yuhaichuan@whu.edu.cn>

Repository CRAN

Date/Publication 2026-05-28 11:30:01 UTC

Contents

`bp_cont` *Blood Pressure Continuous Outcome Data*

Description

A dataset of study-level summary statistics for continuous outcomes (blood pressure) from multiple clinical trials. Contains mean, standard deviation, and sample size per arm, suitable for the continuous module.

Usage

```
bp_cont
```

Format

A data frame with columns:

study Study identifier.

group Treatment group indicator (0 = control, 1 = treatment).

n Sample size per arm.

mean Mean blood pressure.

sd Standard deviation of blood pressure.

---

distance_functions *Distance Functions for ABC-SMC*

Description

Each module exports a distance function that compares simulated summary statistics to the observed summary statistics.

Usage

```
spima_bin_distance(sim_stats, obs_stats)

spima_cont_distance(sim_stats, obs_stats)
```

Arguments

`sim_stats`	Simulated summary statistics (vector or list).
`obs_stats`	Observed summary statistics (same structure).

Value

A non-negative scalar distance.

Functions

- `spima_bin_distance()`: Binary outcome: Euclidean distance on (possibly weighted) log-odds scale.
- `spima_cont_distance()`: Continuous outcome: inverse-variance weighted Euclidean distance on study-level mean differences.

forest *Generic forest plot*

Description

Draws a forest plot showing study-level effect estimates with 95% CIs and the SPI-MA pooled posterior estimate.

Usage

```
forest(x, ...)

## S3 method for class 'spima'
forest(
  x,
  log_scale = FALSE,
  study_labels = NULL,
  col = "grey40",
  pooled_col = "#2166AC",
  xlab = NULL,
  ...
)

spima_forest(x, ...)
```

Arguments

x	A spima result object.
...	Additional arguments passed to plot.
log_scale	If TRUE, use logarithmic x-axis (for OR/HR).
study_labels	Optional character vector of study labels.
col	Color for study-level points and CIs.
pooled_col	Color for the pooled diamond.
xlab	X-axis label (auto-detected if NULL).

Value

A plot object; see the method documentation for details.

A ggplot object (invisibly) if ggplot2 is available, or NULL with a plot drawn to the active device.

gen_effect *Generic Effect Size Data*

Description

A dataset of study-level summary statistics for generic (continuous) effect sizes. Contains effect size estimates and their standard errors, suitable for the generic module.

Usage

```
gen_effect
```

Format

A data frame with columns:

study Study identifier.

yi Effect size estimate.

sei Standard error of the effect size.

kidney_bin *Kidney Disease Binary Outcome Data*

Description

A dataset of study-level summary statistics for binary outcomes (kidney disease) from multiple clinical trials. Contains event counts and sample sizes per arm, suitable for the binary module.

Usage

```
kidney_bin
```

Format

A data frame with columns:

study Study identifier.

group Treatment group indicator (0 = control, 1 = treatment).

n Sample size per arm.

event Number of events per arm.

log_prior_density *Evaluate log-prior density for a parameter vector*

Description

Evaluate log-prior density for a parameter vector

Usage

```
log_prior_density(theta, prior_obj)
```

Arguments

theta	Named numeric vector of parameters.
prior_obj	A spima_prior object.

Value

Log-density value (summed across independent priors).

plot.spima_int *Plot Treatment Effect Modification*

Description

Generates a plot showing how the predicted treatment effect (absolute risk difference or risk ratio) varies across the range of a continuous covariate, based on interaction estimates from spima_int.

Usage

```
## S3 method for class 'spima_int'
plot(
  x,
  covariate = NULL,
  ci_level = 0.95,
  at = NULL,
  scale = c("absolute", "relative"),
  ...
)
```

Arguments

x	A spima_int object from spima_int.
covariate	Character; name of the covariate to plot. If NULL, the first covariate is used.
ci_level	Confidence level for the uncertainty band (default 0.95).
at	Numeric vector of covariate values at which to evaluate the treatment effect. If NULL, 50 points are generated from the observed data range.
scale	"absolute" (default) for risk difference, "relative" for risk ratio.
...	Additional arguments (ignored).

Details

The underlying model is either the pseudo-IPD individual-level GLMM (preferred) or the aggregate ecological GLMM (fallback). Uncertainty is propagated by sampling from the multivariate normal approximation of the fixed effects.

Value

A ggplot object.

Examples

```
res <- spima_int(data, input_spec)
plot(res, covariate = "X1", scale = "absolute")
```

---

prior *Define Prior Distributions for ABC-SMC Parameters*

Description

Define Prior Distributions for ABC-SMC Parameters

Usage

```
prior(mu = "normal(0, 10)", tau = "halfnormal(0, 1)", ...)
```

Arguments

mu	Prior specification for overall effect μ, e.g. "normal(0, 10)".
tau	Prior specification for heterogeneity τ, e.g. "halfnormal(0, 1)".
...	Additional named priors (e.g. gamma = "uniform(0, 5)").

Value

A list of class spima_prior with elements name, pars, rfun (random generation), dfun (density), and default.

Examples

```
prior(mu = "normal(0, 10)", tau = "halfnormal(0, 1)")
```

`run_abc_smc` *Run ABC-SMC Inference*

Description

Run ABC-SMC Inference

Usage

```
run_abc_smc(prior_obj, sim_fn, distance_fn, obs_stats, ctrl, ...)
```

Arguments

`prior_obj`	A `spima_prior` object.
`sim_fn`	Simulation function: `function(theta, ...)` returning simulated summary statistics.
`distance_fn`	Distance function: `function(sim, obs)` returning a scalar.
`obs_stats`	Observed (target) summary statistics.
`ctrl`	An `smc_control` list.
`...`	Additional arguments passed to `sim_fn`.

Value

A list of class `spima_abc` containing posterior samples, weights, diagnostics, and generation records.

`sample_prior` *Sample from the joint prior*

Description

Sample from the joint prior

Usage

```
sample_prior(n, prior_obj)
```

Arguments

`n`	Number of samples.
`prior_obj`	A `spima_prior` object.

Value

A matrix with `n` rows and one column per prior.

`smc_control` *Control Parameters for ABC-SMC*

Description

Control Parameters for ABC-SMC

Usage

```
smc_control(
  n_particles = 2000,
  n_particles_max = 10000,
  n_generations = 10,
  epsilon_init = NULL,
  epsilon_decay = 0.85,
  ess_min = 0.3,
  kernel = "gaussian",
  accept_rate_target = 0.2,
  verbose = TRUE,
  parallel = FALSE,
  n_cores = NULL
)
```

Arguments

`n_particles`	Number of particles (simulations) per generation.
`n_particles_max`	Maximum number of particles for adaptive doubling.
`n_generations`	Maximum number of SMC generations.
`epsilon_init`	Initial acceptance threshold. If `NULL`, it is set to the median of distances from an initial pilot run.
`epsilon_decay`	Multiplicative factor applied to epsilon each generation (0 < decay < 1).
`ess_min`	Minimum effective-sample-size ratio (relative to `n_particles`); algorithm halts when ESS drops below this.
`kernel`	Perturbation kernel type: `"gaussian"` (default).
`accept_rate_target`	Target acceptance rate used for adaptive epsilon tuning.
`verbose`	Print progress information?
`parallel`	Logical; if `TRUE`, run particle simulations in parallel using `parallel::mclapply` (Unix) or a PSOCK cluster (Windows).
`n_cores`	Number of CPU cores for parallel execution. If `NULL` (default), uses `getOption("mc.cores")` or `parallel::detectCores() - 1`.

Value

A list of class `smc_control`.

Examples

```
smc_control(n_particles = 500, n_generations = 8)
```

`spima` *spima: Simulated Pseudo-Individual Data Meta-Analysis*

Description

The main entry point. Dispatches to the appropriate module based on `outcome_type` and runs ABC-SMC for meta-analytic inference.

Usage

```
spima(
  data,
  outcome_type = c("binary", "continuous", "generic"),
  input_spec,
  prior,
  smc_control,
  parallel = FALSE,
  subgroup = NULL,
  family = c("gaussian", "Gamma"),
  ...
)
```

Arguments

`data`	A data frame of study-level summary statistics; one row per study (or per study-arm when `group` is specified in `input_spec`).
`outcome_type`	Outcome type: `"binary"`, `"continuous"`, or `"generic"`.
`input_spec`	A named list mapping column names to roles. The required entries depend on `outcome_type`: **binary** `event`, `n`, optionally `group` and `study`. **continuous** `mean`, `sd`, `n`, optionally `group` and `study`. Alternative formats: `median`, `q1`, `q3`, `n` (median+IQR); `median`, `min`, `max`, `n` (range); or `median`, `q1`, `q3`, `min`, `max`, `n` (five-number summary). **generic** `yi` (effect size), `sei` (standard error), optionally `study`.
`prior`	A `spima_prior` object created by `prior()`.
`smc_control`	An `smc_control` list created by `smc_control()`.
`parallel`	Logical; if `TRUE`, use `parallel::mclapply` for particle simulations (Unix only).

subgroup	Optional column name for subgroup analysis. When specified, the analysis is run separately for each level of this variable.
family	Distributional family for the pseudo-IPD likelihood. Only used when outcome_type = "continuous". "gaussian" (default) assumes normally-distributed outcomes and estimates Mean Differences. "Gamma" assumes Gamma-distributed outcomes and estimates Rate Ratios on the log scale (log-RR).
...	Additional arguments passed to module functions.

Value

A spima object with components:

call	The matched call.
outcome_type	The outcome type.
abc_result	Full ABC-SMC output (generations, posterior, etc.).
data	The input data.
input_spec	The column mapping.

Examples

```
# Quick demo (runs in < 5 seconds)
data_bin <- data.frame(
  study = 1:4,
  event = c(30, 45, 28, 32),
  n     = c(100, 100, 80, 80)
)
res <- spima(data_bin, "binary",
             input_spec = list(study = "study", event = "event", n = "n"),
             prior = prior(mu = "normal(0, 10)", tau = "halfnormal(0, 1)"),
             smc_control = smc_control(n_particles = 100, n_generations = 2))
print(res)
summary(res)


# Full analysis (two-arm per study)
data_bin2 <- data.frame(
  study = 1:4,
  group = c(0, 1, 0, 1, 0, 1, 0, 1),
  event = c(30, 45, 28, 32, 40, 58, 18, 22),
  n     = c(100, 100, 80, 80, 120, 120, 60, 60)
)
res2 <- spima(data_bin2, "binary",
              input_spec = list(study = "study", event = "event",
                                n = "n", group = "group"),
              prior = prior(mu = "normal(0, 10)", tau = "halfnormal(0, 1)"),
              smc_control = smc_control(n_particles = 500, n_generations = 5))
```

spima_bin_analyze *Analyze Pseudo-IPD for Binary Outcome*

Description

Computes per-study log odds ratios from the simulated pseudo-IPD by constructing 2x2 tables. Also fits a one-stage logistic mixed model (`glmer`) for the overall treatment effect estimate.

Usage

```
spima_bin_analyze(pseudo_ipd, input_spec)
```

Arguments

`pseudo_ipd`	A data frame from `spima_bin_simulate`.
`input_spec`	Column mapping.

Value

A list with `estimates` (mixed-model fixed effects), `summary_stats` (named vector of per-study log ORs, matching the format from `spima_bin_observed_stats`), and `converged` (logical).

spima_bin_observed_stats *Compute Observed Summary Statistics for Binary Data*

Description

Compute Observed Summary Statistics for Binary Data

Usage

```
spima_bin_observed_stats(data, input_spec)
```

Arguments

`data`	Original data frame per blueprint.
`input_spec`	Column mapping.

Value

Named vector of observed log-ORs (or log-odds) with optional inverse-variance weights as attribute.

spima_bin_simulate *Simulate Pseudo-IPD for Binary Outcome*

Description

For each study, the control-group proportion is taken from observed data, and the treatment-group log-odds are shifted by a study-specific effect drawn from N(mu, tau^2). Individual Bernoulli outcomes are then generated.

Usage

```
spima_bin_simulate(study_spec, params, input_spec)
```

Arguments

study_spec	A data frame (subset for one study) containing the observed counts.
params	Named vector c(mu = ..., tau = ...).
input_spec	Column mapping (passed through from spima).

Value

A data frame with columns study, group, y.

spima_bin_validate *Validate Binary Outcome Input*

Description

Validate Binary Outcome Input

Usage

```
spima_bin_validate(data, input_spec)
```

Arguments

data	A data frame with columns for events and sample sizes.
input_spec	A named list specifying column mappings, e.g. list(event = "event_t", n = "n_t", group = "trt") for two-group data, or list(event = "event", n = "n") for single-group data.

Value

TRUE invisibly; stops with a message on failure.

spima_cont_analyze *Analyze Pseudo-IPD for Continuous Outcome*

Description

Computes per-study mean differences from the simulated pseudo-IPD. Also attempts a linear mixed model (`lmer`) for overall estimate.

Usage

```
spima_cont_analyze(pseudo_ipd, input_spec)
```

Arguments

`pseudo_ipd`	A data frame from `spima_bin_simulate`.
`input_spec`	Column mapping.

Value

A list with `estimates`, `summary_stats` (per-study mean differences and pooled SDs, matching observed_stats format), and `converged`.

spima_cont_observed_stats *Compute Observed Summary Statistics for Continuous Data*

Description

Compute Observed Summary Statistics for Continuous Data

Usage

```
spima_cont_observed_stats(data, input_spec)
```

Arguments

`data`	Original data frame.
`input_spec`	Column mapping.

Value

A list with `means` and `sds` (named vectors).

spima_cont_simulate *Simulate Pseudo-IPD for Continuous Outcome*

Description

For each study, individual data are drawn from a normal (or skew-normal) distribution matching the observed mean and SD. The treatment group mean is shifted by a study-specific effect drawn from N(mu, tau^2).

Usage

```
spima_cont_simulate(study_spec, params, input_spec)
```

Arguments

study_spec	A data frame (subset for one study) containing the observed counts.
params	Named vector c(mu = ..., tau = ...).
input_spec	Column mapping (passed through from spima).

Value

A data frame with columns study, group, y.

spima_cont_validate *Validate Continuous Outcome Input*

Description

Validate Continuous Outcome Input

Usage

```
spima_cont_validate(data, input_spec)
```

Arguments

data	A data frame with means, SDs, and sample sizes.
input_spec	A named list, e.g. list(mean = "mean_t", sd = "sd_t", n = "n_t", group = "trt"). For median+IQR input, use median and q1, q3.

Value

TRUE invisibly.

spima_gamma_analyze *Analyze Pseudo-IPD for Gamma Outcome*

Description

Fits a one-stage Gamma GLMM via `glmer(y ~ group + (1 | study), family = Gamma(link = "log"))` on the simulated pseudo-IPD. Also returns per-study log-Rate Ratio values for use as summary statistics in the ABC distance computation.

Usage

```
spima_gamma_analyze(pseudo_ipd, input_spec, quick = TRUE)
```

Arguments

`pseudo_ipd`	A data frame from `spima_bin_simulate`.
`input_spec`	Column mapping.
`quick`	If `TRUE` (default), skip the GLMM fit and only compute per-study log-RR summary statistics.

Details

Use `quick = TRUE` (default) during ABC-SMC sampling where only the per-study summary statistics are needed for distance computation. Set `quick = FALSE` to additionally fit the full GLMM (useful for external diagnostics).

Value

A list with components:

`estimates` Named vector of fixed effects from the Gamma GLMM (or `NULL` if `quick = TRUE` or the model does not converge).

`summary_stats` Named vector of per-study log-RR values.

`converged` Logical indicating GLMM convergence (`TRUE` when `quick = TRUE`).

`fit` The `glmer` fit object (or `NULL`).

spima_gamma_distance *Distance Function for Gamma Outcome*

Description

Weighted Euclidean distance on the per-study log-Rate Ratio vector. Delegates to `spima_generic_distance`.

Usage

```
spima_gamma_distance(sim_stats, obs_stats)
```

Arguments

`sim_stats`	Simulated summary statistics (vector or list).
`obs_stats`	Observed summary statistics (same structure).

Value

A non-negative scalar distance (lower = better match).

spima_gamma_observed_stats *Compute Observed Summary Statistics for Gamma Data*

Description

For each study, computes the observed log-Rate Ratio and its delta-method variance for inverse-variance weighting.

Usage

```
spima_gamma_observed_stats(data, input_spec)
```

Arguments

`data`	Original data frame with arm-level means, SDs, and sample sizes.
`input_spec`	Column mapping.

Value

A named vector of per-study log-RR values with an attribute `"weights"` containing inverse-variance weights.

spima_gamma_simulate *Simulate Pseudo-IPD for Gamma Outcome*

Description

For each study, individual data are drawn from a Gamma distribution matching the observed mean and SD via method-of-moments. The treatment group mean is shifted by a multiplicative factor `exp(theta_i)` where `theta_i ~ N(mu, tau^2)` — this encodes the log-Rate Ratio treatment effect on the original scale. The shape parameter is held constant within each study, preserving the variance structure implied by the Gamma GLM with log link.

Usage

```
spima_gamma_simulate(study_spec, params, input_spec)
```

Arguments

`study_spec`	A data frame (subset for one study) containing the observed counts.
`params`	Named vector `c(mu = ..., tau = ...)`.
`input_spec`	Column mapping (passed through from `spima`).

Value

A data frame with columns `study`, `group`, `y`.

spima_gamma_validate *Validate Gamma Outcome Input*

Description

Delegates to `spima_cont_validate` (same data format: mean, sd, n per arm) and additionally checks that all means are positive (Gamma distribution is supported on the positive real line).

Usage

```
spima_gamma_validate(data, input_spec)
```

Arguments

`data`	A data frame with means, SDs, and sample sizes.
`input_spec`	A named list, e.g. `list(mean = "mean_t", sd = "sd_t", n = "n_t", group = "trt")`. For median+IQR input, use `median` and `q1`, `q3`.

Value

`TRUE` invisibly.

spima_generic_analyze *Analyze Pseudo-Data for Generic Effect-Size*

Description

For the generic module, the "pseudo-IPD" is already the summary statistics (a vector of effect sizes). This function simply passes them through with converged = TRUE.

Usage

```
spima_generic_analyze(pseudo_ipd, input_spec)
```

Arguments

pseudo_ipd	A data frame from spima_bin_simulate.
input_spec	Column mapping.

Value

A list with estimates = NULL, summary_stats (the effect-size vector), and converged = TRUE.

spima_generic_distance *Generic (effect-size) distance: weighted Euclidean distance on effects*

Description

Weighted Euclidean distance using inverse-variance weights.

Usage

```
spima_generic_distance(sim_stats, obs_stats)

spima_generic_distance(sim_stats, obs_stats)
```

Arguments

sim_stats	Simulated summary statistics (vector or list).
obs_stats	Observed summary statistics (same structure).

Value

A non-negative scalar distance (lower = better match).

`spima_generic_observed_stats` *Compute Observed Summary Statistics for Generic Effect-Size*

Description

Compute Observed Summary Statistics for Generic Effect-Size

Usage

```
spima_generic_observed_stats(data, input_spec)
```

Arguments

`data`	Original data frame.
`input_spec`	Column mapping.

Value

Named vector of effect sizes with `"weights"` attribute (inverse-variance: 1/sei^2).

`spima_generic_simulate` *Simulate Pseudo-Data for Generic Effect-Size*

Description

No individual-level data is generated. Instead, study-level effect sizes are drawn from the random-effects model: theta_i ~ N(mu, tau^2) y_i* ~ N(theta_i, sei_i^2)

Usage

```
spima_generic_simulate(study_spec, params, input_spec)
```

Arguments

`study_spec`	A data frame (subset for one study) containing the observed counts.
`params`	Named vector `c(mu = ..., tau = ...)`.
`input_spec`	Column mapping (passed through from `spima`).

Details

The returned vector can be treated as "pseudo-IPD" since it directly represents the summary statistics needed for distance computation.

Value

A named numeric vector of simulated effect sizes (one per study).

`spima_generic_validate` *Validate Generic Effect-Size Input*

Description

Validate Generic Effect-Size Input

Usage

```
spima_generic_validate(data, input_spec)
```

Arguments

`data`	A data frame with effect-size and SE columns.
`input_spec`	A named list, e.g. `list(yi = "yi", sei = "sei")`.

Value

`TRUE` invisibly.

`spima_int` *SPI-MA Interaction Analysis*

Description

Tests whether continuous covariate(s) modify the treatment effect using aggregate data only. The primary method fits a mixed-effects logistic regression on the aggregate data. A sensitivity analysis generates pseudo-IPD and fits an individual-level model.

Usage

```
spima_int(data, input_spec, rho = 0, ...)

spima_int_validate(data, input_spec)
```

Arguments

`data`	Data frame, one row per study-arm.
`input_spec`	Named list: **study** Study identifier column name. **event** Event count column name. **n** Sample size column name. **group** Treatment group column (0 = control, 1 = treatment). **covariate** Character vector of covariate name(s), e.g. `c("X1","X2")`. The function looks for columns `mean_<name>` and `sd_<name>` in `data`.
`rho`	Assumed between-covariate correlation for pseudo-IPD generation. Default 0.
`...`	Additional arguments to `glmer`.

Value

`spima_int` object.

Index