Package 'DiPPER'

October 30, 2026

Title Differential Prevalence via Probabilistic Estimation in R

Version 0.99.1

Description Implements a Bayesian hierarchical model for differential prevalence analysis, tailored for microbiome studies. The method accommodates both cross-sectional and longitudinal designs, providing finite effect estimates and uncertainty intervals that are naturally adjusted for multiple testing via hierarchical shrinkage. Posterior inference is performed via Markov chain Monte Carlo (MCMC) using 'CmdStan' as the backend.

License Artistic-2.0

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.6.0)

Imports instantiate, SummarizedExperiment, TreeSummarizedExperiment, stats, utils, ggplot2

Suggests testthat (>= 3.1.5), knitr, rmarkdown, BiocStyle

SystemRequirements CmdStan (https://mc-stan.org/users/interfaces/cmdstan)

biocViews Software, Microbiome, Bayesian, Regression, DifferentialExpression, Sequencing

URL https://github.com/jepelt/DiPPER

BugReports https://github.com/jepelt/DiPPER/issues

Config/roxygen2/version 8.1.0

Config/testthat/edition 3

LazyData false

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/DiPPER

git_branch devel

git_last_commit 657a071

git_last_commit_date 2026-09-30

Repository Bioconductor 3.24

Date/Publication 2026-10-29

Author Juho Pelto [aut, cre] (ORCID: <https://orcid.org/0000-0001-8820-8014>),
Leo Lahti [aut] (ORCID: <https://orcid.org/0000-0001-5537-637X>),
Research Council of Finland [fnd] (Strategic Research Council,
decisions 372345 and 374944)

Maintainer Juho Pelto <jepelt@utu.fi>

Contents

dipper 2
DiPPER-extdata 7
fit_example 8
plot.dipper_fit 9
prep_dipper_data 10
print.dipper_fit 11
run_dipper 12
summary.dipper_fit 14
tse_hintikka 15
VatanenT_2016_subset 16

Index **17**

`dipper` *DiPPER (Differential Prevalence via Probabilistic Estimation in R)*

Description

This is the main wrapper function for running DiPPER. It first prepares the input data and then fits the model using CmdStanR.

Usage

```
dipper(
  tse = NULL,
  assay.type = NULL,
  assay = NULL,
  meta = NULL,
  data.type = NULL,
  formula,
  var.of.interest = NULL,
  read.depth = NULL,
  symmetric = FALSE,
  threshold = 0,
  min.present = 5,
  min.absent = min.present,
  niter = 2000,
  niter.warmup = floor(niter/2),
  chains = 4,
  cores = 4,
  adapt.delta = 0.8,
  max.treedepth = 10,
  run.diagnostics = TRUE,
```

```
  diagnostics.level = c("basic", "full"),
  keep.pars = c("beta", "tau", "nu"),
  keep.stanfit = FALSE,
  seed = 1,
  print.progress = 200,
  verbose = TRUE,
  prior.alpha.sd = 4,
  prior.tau.sd = 1,
  prior.nu.sd = 0.05,
  prior.cov.sd = 1,
  prior.reads.mean = 2,
  prior.reads.sd = 2,
  prior.sigma.subj = 1,
  ...
)
```

Arguments

`tse`	A (Tree)SummarizedExperiment object.
`assay.type`	Character. Name of the assay in the TSE object to use. Required if tse is provided.
`assay`	A matrix containing counts, relative abundances, other non-negative abundances, or presence/absence data (required if tse is NULL). Rows should be taxa/features, columns should be samples.
`meta`	A data.frame containing metadata (required if tse is NULL).
`data.type`	Character. Type of the data. Required, one of: • `"counts"`: sequencing read counts (non-negative integers). • `"relabundance"`: relative abundances (values between 0 and 1, with sample sums at most 1). • `"abundance"`: any other non-negative abundances, such as pathway abundances, CPM or TPM values. • `"pa"`: presence/absence data (0s and 1s). Transformed data with negative values (e.g. CLR or log) are not supported, because presence and absence cannot be determined from them.
`formula`	Model formula.
`var.of.interest`	The variable of interest from the formula. If NULL, the first term of the formula is automatically used.
`read.depth`	How to control for sequencing read depth. Required for `data.type = "counts"`; for other data types the default (NULL) means that read depth is not controlled for. One of: • `TRUE`: read depth is computed as the sample totals. Requires `data.type = "counts"`. Use with non-rarefied sequencing counts. • A metadata column name: read depths are taken from the specified column, e.g. read depths computed before filtering. Give the read depths on their original scale (numbers of reads), and do not include the column in `formula`. • `FALSE`: read depth is not controlled for, e.g. for rarefied data.

With TRUE or a column name, read depth is automatically log10-transformed and added to the model as the covariate log10_read_depth.

symmetric — Logical. If TRUE, a symmetric Laplace prior (nu = 0.5) is used for the differential prevalence parameters of interest. If FALSE (default), the asymmetry parameter nu is estimated from the data. See the Details section of dipper for guidance on choosing between the two.

threshold — Numeric or function. A feature is considered present in a sample if its value is greater than the threshold. The threshold is on the scale of the data: counts for "counts", proportions for "relabundance", and the original units for "abundance". The default is 0, so a feature is absent only if its value is exactly zero. If a function (e.g. median), it is applied to each feature to obtain feature-specific thresholds. Ignored if data.type is "pa".

min.present — For taxon/feature filtering. Minimum number or proportion of samples where a taxon must be present. Default is 5.

min.absent — For taxon/feature filtering. Minimum number or proportion of samples where a taxon must be absent. Default is min.present.

niter — Total number of MCMC iterations per chain (including warmup). Default is 2000.

niter.warmup — Number of warmup MCMC iterations per chain. Defaults to floor(niter / 2).

chains — Number of MCMC chains. Default is 4.

cores — Number of CPU cores to use. Default is 4.

adapt.delta — Target average acceptance probability in MCMC sampling. Default is 0.8.

max.treedepth — Maximum depth of the trees in MCMC sampling. Default is 10.

run.diagnostics — Logical. Whether to run convergence diagnostics after MCMC sampling. Default is TRUE.

diagnostics.level — Character. Level of MCMC diagnostics to run. "basic" (default) checks main parameters (alpha, beta, covariates, hyperparameters). "full" checks all parameters, including latent and auxiliary variables.

keep.pars — Character vector of Stan parameter names whose posterior draws are retained in the returned object, or NULL to retain all of them. The default, c("beta", "tau", "nu"), keeps the differential prevalence parameters of the variable of interest together with the two hyperparameters of the hierarchical Laplace prior. nu is dropped automatically when symmetric = TRUE.

keep.stanfit — Logical. Whether to include the underlying CmdStanR fit R6 object in the returned object (e.g. for manual model fit checking). Default is FALSE.

seed — Random seed for reproducibility. Default is 1.

print.progress — How often to print MCMC progress. Set to 0 or FALSE to disable. Default is 200. Ignored if verbose = FALSE.

verbose — Logical. Whether to print progress messages. Default is TRUE.

prior.alpha.sd — Prior standard deviation for alpha (fixed intercept for centered variables). Default is 4.0.

prior.tau.sd — Prior standard deviation for tau (global scale of the Symmetric/Asymmetric Laplace prior). Default is 1.0.

prior.nu.sd — Prior standard deviation for nu (asymmetry parameter). Ignored if symmetric = TRUE. Default is 0.05.

`prior.cov.sd` Default prior standard deviation for covariates. Default is 1.0.

`prior.reads.mean`
Prior mean for the read depth covariate. Default is 2.0.

`prior.reads.sd` Prior standard deviation for the read depth covariate. Default is 2.0.

`prior.sigma.subj`
Prior standard deviation for the half-normal prior distribution of the taxon-specific random intercept standard deviations. Default is 1.0.

`...` Additional arguments passed directly to `run_dipper` (e.g., cmdstanr specific arguments).

Details

DiPPER is a Bayesian hierarchical model-based approach designed for differential prevalence analysis, especially for microbiome studies. It is designed for study designs where there is one **variable of interest** (e.g. treatment group, disease status, etc.) and potentially other covariates (e.g. age, sex, etc.) that may confound the associations between microbe prevalences and the variable of interest. DiPPER can also be applied to longitudinal or repeated measures data.

Technically, DiPPER models the presence/absence of taxonomic features (e.g. genera or species) using multiple logistic regression models. The models are connected by a common (Asymmetric) Laplace prior for the variable of interest. The hierarchical structure guarantees robust (and always finite) differential prevalence estimates and uncertainty intervals that are also effectively multiplicity-adjusted.

Input data and preprocessing: The `dipper` function takes either a `(Tree)SummarizedExperiment` object or an abundance matrix and a metadata `data.frame` as input. The abundance matrix can be sequencing counts, relative abundances, other non-negative abundances (e.g. pathway abundances), or presence/absence data. The `data.type` argument is required, as it determines how presence is defined and whether read depth can be derived from the data.

Before model fitting, the input data undergoes automated preprocessing. Continuous abundance data are converted to presence/absence format based on the specified `threshold`. By default, a feature is absent only if its value is exactly zero. Furthermore, prevalence filtering is applied: taxa that are not present in at least `min.present` samples, or absent in at least `min.absent` samples, are excluded from the analysis.

Model formula and the variable of interest: The `dipper` function uses the standard `formula` argument to specify the model formula. A possible random intercept is included in a standard lme4 style as the last term in the formula (e.g. `+ (1 | subject_id)`).

By default, the first term in the `formula` is treated as the variable of interest, and the results are provided only for this variable. Any other term of the formula can be selected as the variable of interest with `var.of.interest`.

The variable of interest must be either binary (a factor or character variable with exactly two levels) or continuous. The other variables in the model can, however, be factors with more than two levels.

Controlling for read depth (`read.depth`): As the observed presence/absence status of microbes may depend on the sequencing (read) depth, DiPPER can control for it by adding log10 read depth as a covariate. Because whether this is appropriate depends on preprocessing that DiPPER cannot detect, `read.depth` must be chosen explicitly for count data:

- Use `TRUE` for non-rarefied sequencing counts.
- Give a metadata column name when the read depths were computed manually, e.g. before heavy filtering of the count data.

- Use FALSE when read depth is already accounted for, e.g. by rarefying the data.

For other data types, read depth cannot be derived from the abundances themselves, so it is controlled for only if a metadata column of pre-computed read depths is supplied.

Symmetric or asymmetric prior (`symmetric`): By default, `dipper` uses the Asymmetric Laplace prior as the hierarchical prior for the differential prevalence parameters of interest. This means that the (a)symmetry of the Laplace prior is estimated from the data. However, this choice may emphasize some technical biases in the data. This can occur, for instance, when read depth correlates with the variable of interest but one is not able to control for the read depth in the analysis. Therefore, if such technical biases are possible or likely, the user can choose to use a symmetric Laplace prior instead by setting `symmetric = TRUE`.

The two versions often give rather similar results, but the symmetric version generally leads to slightly more robust and/or conservative results while the asymmetric version is more sensitive to detect small effects.

The parameters of the hierarchical prior are reported by `print.dipper_fit`:

- `nu` indicates the asymmetry. It is constrained between 0 and 1, with 0.5 meaning symmetry, values below 0.5 indicating positive skewness and values above 0.5 negative skewness.
- `tau` is the prior scale, that is, small values mean that the differential prevalence parameters are shrunk strongly towards zero/each other.

If `nu` is clearly away from 0.5 (e.g. below 0.30 or above 0.70) and `tau` is at the same time small (e.g. < 0.10), the asymmetry of the prior may have a somewhat strong effect on the results. This may lead to a large number of 'significant' findings in the same direction. If one suspects that these findings may be due to some technical biases in the data, it may be advisable to use the symmetric version of DiPPER.

MCMC sampling and prior settings: The posterior distribution computation for DiPPER is performed using the Hamiltonian Monte Carlo algorithm via CmdStanR. The total number of posterior samples can be controlled via arguments `niter` and `chains`. Higher numbers lead to higher accuracy of the posterior statistics but increase the computation time. Nevertheless, the default values `niter = 2000` and `chains = 4` should be sufficient for most cases. Higher values are generally recommended only if indicated by the automatic MCMC diagnostics (e.g., too high R-hat values, or too low effective sample sizes).

Lastly, adjusting the default prior distribution settings (`prior.alpha.sd`, `prior.tau.sd`, etc.) is generally not recommended unless the user is highly experienced with Bayesian modeling and the details of DiPPER.

Value

A list object of class `dipper_fit` containing:

draws A matrix of posterior draws, with one row per MCMC draw and one column per retained parameter. By default the draws of `beta` (the parameters of interest) and of the hierarchical prior parameters `tau` and `nu` are retained; see `keep.pars`.

dipper_data A list containing the prepared data passed to Stan.

symmetric Logical indicating if a symmetric Laplace prior for differential prevalence parameters was used.

diagnostics A list of MCMC diagnostics. Always contains `num_divergent`, `num_max_treedepth`, `time_total`, `n_chains` and `n_iter_sampling`. If `run.diagnostics = TRUE`, it additionally contains `max_rhat`, `min_ess_bulk`, `min_ess_tail` and `diagnostics_level`.

stanfit The `CmdStanMCMC` object returned by CmdStanR. Present only if `keep.stanfit = TRUE`. Note that this is an R6 object that depends on the cmdstanr namespace, so a fit containing it cannot be reloaded on a machine where cmdstanr is unavailable.

References

Pelto, J., et al. (2026). DiPPER: A Bayesian approach to differential prevalence analysis with applications in microbiome studies. arXiv preprint. https://arxiv.org/abs/2602.05938

Examples

```
data("tse_hintikka")

# Run DiPPER
# Note: niter = 400, chains = 2 and cores = 2 are used here for speed, so
# convergence warnings are expected. In real applications, use higher
# values (e.g. the default niter = 2000, chains = 4, and cores = 4).
if (instantiate::stan_cmdstan_exists()) {
    fit <- dipper(
        tse = tse_hintikka,
        assay.type = "counts",
        data.type = "counts",
        formula = ~ Fat + XOS,
        read.depth = TRUE,
        niter = 400,
        chains = 2,
        cores = 2
    )

    print(fit)

    res <- summary(fit)
    head(res)
}
```

DiPPER-extdata *Pre-computed DiPPER fits*

Description

Two `dipper_fit` objects stored as `.rds` files under `inst/extdata`. They are loaded by the package vignette when CmdStan is unavailable, so that the vignette can be built and display real results also on machines without a Stan toolchain.

Format

fit_hintikka.rds Fitted to the `tse_hintikka` dataset with the formula `~ Fat + XOS`, `assay.type = "counts"`, `data.type = "counts"` and `read.depth = TRUE`.

fit_vatanen.rds Fitted to the `VatanenT_2016_subset` dataset with the formula `~ age_point + antibiotics + gender + (1 | subject_id)`, `assay.type = "relative_abundance"`, `data.type = "relabundance"` and `read.depth = FALSE`.

Value

Each file contains a list of class `dipper_fit`. See the documentation of `dipper()` for a description of the elements. The files are read with `readRDS()`.

Source

Generated from the bundled datasets. See `inst/scripts/fits_vignette.R` for the code that produced them.

See Also

`tse_hintikka`, `VatanenT_2016_subset` and `fit_example` for the datasets and the example fit stored in `data/`.

Examples

```
fit <- readRDS(
    system.file("extdata", "fit_hintikka.rds", package = "DiPPER")
)

print(fit)
head(summary(fit))
```

`fit_example` *Example DiPPER model fit object*

Description

A pre-calculated `dipper_fit` object based on tse_hintikka data. To be used in examples to avoid long MCMC sampling times during package checks. This object was created to enable running the examples of `summary`, `plot` and `print` without using CmdStan. The number of iterations `niter = 400` and chains `chains = 2` used to create this object is too low for reliable inference in practice.

Usage

```
data(fit_example)
```

Format

A list of class `dipper_fit` containing:

draws A matrix of posterior draws, with one row per MCMC draw and one column per taxon (the beta parameters).

dipper_data A list containing the prepared data passed to Stan.

symmetric Logical indicating if a symmetric Laplace prior for the parameters of interest was used.

diagnostics A list of MCMC diagnostics (divergent transitions, maximum treedepth hits, sampling time, convergence statistics, and diagnostics level).

Source

Generated using the `tse_hintikka` dataset with `niter = 400` and `chains = 2`.

`plot.dipper_fit` *Plot method for DiPPER fit objects*

Description

Plot method for DiPPER fit objects

Usage

```
## S3 method for class 'dipper_fit'
plot(
  x,
  prob = 0.95,
  show.taxa = "significant",
  original.scale = TRUE,
  verbose = TRUE,
  ...
)
```

Arguments

`x`	A `dipper_fit` object returned by `dipper`.
`prob`	Numeric. Probability mass for the credible interval (analogous to confidence level in frequentist statistics). Default is 0.95 (i.e., 95 percent credible intervals are produced).
`show.taxa`	Selection criteria for taxa: "significant" (credible intervals not overlapping OR = 1), "all", or an integer k (top k taxa ranked by pseudo_q, see `summary.dipper_fit`). Default is "significant".
`original.scale`	Logical. If `TRUE` (default), continuous variable of interest is back-transformed to their original scale. If `FALSE`, results are presented per one standard deviation increase. Ignored for categorical variable of interest.
`verbose`	Logical. Whether to print a message when no taxa remain after filtering. Default is TRUE.
`...`	Additional arguments (currently ignored).

Details

This function generates a forest plot of the differential prevalence estimates for the variable of interest.

The credible intervals are based on the `(1 - prob) / 2` and `1 - (1 - prob) / 2` posterior quantiles.

Note that the significance is determined by whether the `prob * 100` percent credible interval excludes OR = 1. Significance is thus affected by the `prob` level.

Value

A `ggplot` object or NULL (invisibly) if no taxa remain after filtering.

Examples

```
# Load pre-run model fit for the example dataset (tse_hintikka)
data("fit_example")

# Plot the results for significant taxa (with 95 percent credible intervals)
plot(fit_example)

# Plot the results for all taxa and use 90 percent credible intervals
plot(fit_example, prob = 0.90, show.taxa = "all")
```

`prep_dipper_data` *Prepare data for DiPPER*

Description

Prepare data for DiPPER

Usage

```
prep_dipper_data(
  tse = NULL,
  assay.type = NULL,
  assay = NULL,
  meta = NULL,
  data.type = NULL,
  formula,
  var.of.interest = NULL,
  read.depth = NULL,
  threshold = 0,
  min.present = 5,
  min.absent = min.present,
  verbose = TRUE
)
```

Arguments

`tse`	A (Tree)SummarizedExperiment object.
`assay.type`	Character. Name of the assay in the TSE object to use. Required if tse is provided.
`assay`	A matrix containing counts, relative abundances, other non-negative abundances, or presence/absence data (required if tse is NULL). Rows should be taxa/features, columns should be samples.
`meta`	A data.frame containing metadata (required if tse is NULL).
`data.type`	Character. Type of the data. Required, one of: • `"counts"`: sequencing read counts (non-negative integers). • `"relabundance"`: relative abundances (values between 0 and 1, with sample sums at most 1). • `"abundance"`: any other non-negative abundances, such as pathway abundances, CPM or TPM values.

- "pa": presence/absence data (0s and 1s).

Transformed data with negative values (e.g. CLR or log) are not supported, because presence and absence cannot be determined from them.

`formula` Model formula.

`var.of.interest`

The variable of interest from the formula. If NULL, the first term of the formula is automatically used.

`read.depth` How to control for sequencing read depth. Required for `data.type = "counts"`; for other data types the default (NULL) means that read depth is not controlled for. One of:

- `TRUE`: read depth is computed as the sample totals. Requires `data.type = "counts"`. Use with non-rarefied sequencing counts.
- A metadata column name: read depths are taken from the specified column, e.g. read depths computed before filtering. Give the read depths on their original scale (numbers of reads), and do not include the column in `formula`.
- `FALSE`: read depth is not controlled for, e.g. for rarefied data.

With `TRUE` or a column name, read depth is automatically log10-transformed and added to the model as the covariate `log10_read_depth`.

`threshold` Numeric or function. A feature is considered present in a sample if its value is greater than the threshold. The threshold is on the scale of the data: counts for "counts", proportions for "relabundance", and the original units for "abundance". The default is 0, so a feature is absent only if its value is exactly zero. If a function (e.g. median), it is applied to each feature to obtain feature-specific thresholds. Ignored if data.type is "pa".

`min.present` For taxon/feature filtering. Minimum number or proportion of samples where a taxon must be present. Default is 5.

`min.absent` For taxon/feature filtering. Minimum number or proportion of samples where a taxon must be absent. Default is min.present.

`verbose` Logical. Whether to print progress messages. Default is TRUE.

`print.dipper_fit` *Print a summary of a DiPPER model fit*

Description

Print a summary of a DiPPER model fit

Usage

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

Arguments

`x` A `dipper_fit` object.

`...` Additional arguments (currently ignored).

Details

The hierarchical prior line indicates the type of the chosen hierarchical prior (symmetric or asymmetric) and its hyperparameters with their 90 percent credible intervals. (For a symmetric fit, nu is fixed at 0.5.) See the 'Symmetric or asymmetric prior' section in dipper for how to interpret these values and when a symmetric prior may be preferable.

Value

The object x invisibly.

Examples

```
# Load pre-run model fit for the example dataset (tse_hintikka)
data("fit_example")

print(fit_example)
```

`run_dipper` *Fit the DiPPER model using CmdStan*

Description

Fit the DiPPER model using CmdStan

Usage

```
run_dipper(
  prep.data,
  symmetric = FALSE,
  niter = 2000,
  niter.warmup = floor(niter/2),
  chains = 4,
  cores = 4,
  seed = 1,
  adapt.delta = 0.8,
  max.treedepth = 10,
  run.diagnostics = TRUE,
  diagnostics.level = c("basic", "full"),
  keep.pars = c("beta", "tau", "nu"),
  keep.stanfit = FALSE,
  print.progress = 200,
  verbose = TRUE,
  prior.alpha.sd = 4,
  prior.tau.sd = 1,
  prior.nu.sd = 0.05,
  prior.cov.sd = 1,
  prior.reads.mean = 2,
  prior.reads.sd = 2,
  prior.sigma.subj = 1,
  ...
)
```

Arguments

prep.data	The list returned by prep_dipper_data.
symmetric	Logical. If TRUE, a symmetric Laplace prior (nu = 0.5) is used for the differential prevalence parameters of interest. If FALSE (default), the asymmetry parameter nu is estimated from the data. See the Details section of dipper for guidance on choosing between the two.
niter	Total number of MCMC iterations per chain (including warmup). Default is 2000.
niter.warmup	Number of warmup MCMC iterations per chain. Defaults to floor(niter / 2).
chains	Number of MCMC chains. Default is 4.
cores	Number of CPU cores to use. Default is 4.
seed	Random seed for reproducibility. Default is 1.
adapt.delta	Target average acceptance probability in MCMC sampling. Default is 0.8.
max.treedepth	Maximum depth of the trees in MCMC sampling. Default is 10.
run.diagnostics	Logical. Whether to run convergence diagnostics after MCMC sampling. Default is TRUE.
diagnostics.level	Character. Level of MCMC diagnostics to run. "basic" (default) checks main parameters (alpha, beta, covariates, hyperparameters). "full" checks all parameters, including latent and auxiliary variables.
keep.pars	Character vector of Stan parameter names whose posterior draws are retained in the returned object, or NULL to retain all of them. The default, c("beta", "tau", "nu"), keeps the differential prevalence parameters of the variable of interest together with the two hyperparameters of the hierarchical Laplace prior. nu is dropped automatically when symmetric = TRUE.
keep.stanfit	Logical. Whether to include the underlying CmdStanR fit R6 object in the returned object (e.g. for manual model fit checking). Default is FALSE.
print.progress	How often to print MCMC progress. Set to 0 or FALSE to disable. Default is 200. Ignored if verbose = FALSE.
verbose	Logical. Whether to print any progress messages issued by DiPPER functions. Default is TRUE. Setting verbose = FALSE overrides print.progress.
prior.alpha.sd	Prior standard deviation for alpha (fixed intercept for centered variables). Default is 4.0.
prior.tau.sd	Prior standard deviation for tau (global scale of the Symmetric/Asymmetric Laplace prior). Default is 1.0.
prior.nu.sd	Prior standard deviation for nu (asymmetry parameter). Ignored if symmetric = TRUE. Default is 0.05.
prior.cov.sd	Default prior standard deviation for covariates. Default is 1.0.
prior.reads.mean	Prior mean for the read depth covariate. Default is 2.0.
prior.reads.sd	Prior standard deviation for the read depth covariate. Default is 2.0.
prior.sigma.subj	Prior standard deviation for the half-normal prior distribution of the taxon-specific random intercept standard deviations. Default is 1.0.
...	Additional arguments passed to the $sample() method of the CmdStanR model object.

Value

A list of class `dipper_fit` with elements `draws` (a matrix of posterior draws, with one column per retained parameter. See `keep.pars`), `dipper_data`, `symmetric` and `diagnostics`. If `keep.stanfit = TRUE`, the CmdStanR fit R6 object is included as `stanfit`.

`summary.dipper_fit` *Summarize DiPPER results for the variable of interest*

Description

This function computes and summarizes the posterior quantities (differential prevalence estimate, credible interval, pseudo q-value) for the variable of interest for each taxonomic feature.

Usage

```
## S3 method for class 'dipper_fit'
summary(
  object,
  prob = 0.95,
  or.scale = c("log_odds", "odds_ratio"),
  original.scale = TRUE,
  ...
)
```

Arguments

`object`	A `dipper_fit` object returned by `dipper`.
`prob`	Numeric. Probability mass for the credible interval (analogous to confidence level in frequentist statistics). Default is 0.95 (i.e., 95 percent credible intervals are produced).
`or.scale`	Character string indicating the scale of the estimates. Either `"log_odds"` (default) or `"odds_ratio"`.
`original.scale`	Logical. If `TRUE` (default), continuous covariates are back-transformed to their original scale. If `FALSE`, results are presented per one standard deviation increase. Ignored for categorical variables.
`...`	Additional arguments (currently ignored).

Details

The function computes the median posterior estimates and credible interval boundaries (based on the `(1 - prob) / 2` and `1 - (1 - prob) / 2` posterior quantiles) from the posterior draws of the differential prevalence parameters (beta) of the variable of interest.

Note that as the boundaries of the credible intervals depend on the `prob` level, so does the `significant` logical flag.

`pseudo_q` is two times the minimum of the posterior probabilities that the parameter is strictly positive or strictly negative. Consequently, it indicates the highest "significance level" at which the result would be considered "statistically significant" (i.e., the credible interval excluding zero). It thus bears some resemblance to a multiplicity-adjusted two-sided p-value (q-value), but it does not have any strict statistical interpretation. Due to the finite number of posterior draws, zero values are replaced with `0.5 * (1 / n_draws)`.

Value

A `data.frame` containing the summarized results for each taxon, sorted by `pseudo_q` (smallest to largest). The columns are:

taxon Name of the taxonomic feature.

log_or / odds_ratio Median posterior estimate on the chosen scale.

lwr Lower bound of the credible interval.

upr Upper bound of the credible interval.

significant Logical. `TRUE` if the credible interval on the log-odds scale does not include zero (or on the odds ratio scale does not include one).

pseudo_q Bayesian pseudo q-value.

Examples

```
# Load pre-run model fit for the example dataset (tse_hintikka)
data("fit_example")

# Summarize on log-odds scale with default 95 percent interval
res_log <- summary(fit_example)
head(res_log)

# Summarize on odds ratio scale with 90 percent interval
res_or <- summary(fit_example, prob = 0.90, or.scale = "odds_ratio")
head(res_or)
```

---

tse_hintikka *Gut microbiome dataset from Hintikka et al. (2021)*

Description

A `TreeSummarizedExperiment` object containing a subset of gut microbiome data from an experiment studying the effects of a high-fat diet and xylo-oligosaccharide (XOS) supplementation on rats (N = 20 + 20). The abundance data are sequencing counts and they have been agglomerated to the genus level.

Usage

```
data(tse_hintikka)
```

Format

A `TreeSummarizedExperiment` object.

assay Contains the `counts` matrix.

colData Contains the sample metadata, including `Fat` (diet group, factor with levels Low and High) and `XOS` (xylo-oligosaccharide supplementation).

Source

Derived from the `HintikkaXOData` dataset in the `mia` package. See `inst/scripts/tse_hintikka.R` for how the subset was derived. Original study: https://www.mdpi.com/1660-4601/18/8/4049

VatanenT_2016_subset *Longitudinal gut microbiome dataset from Vatanen et al. (2016)*

Description

A TreeSummarizedExperiment object containing a longitudinal subset of infant gut microbiome data from Vatanen et al. (2016). The data consists of 79 infants with measurements at 1.5 and/or 7 months of age. The abundance data are relative abundances (proportions) and they have been agglomerated to the genus level.

Usage

```
data(VatanenT_2016_subset)
```

Format

A TreeSummarizedExperiment object.

assay Contains the relative_abundance matrix (proportions).

colData Contains the sample metadata: subject_id (factor), antibiotics (factor: no, yes), age_point (factor: 1.5 months, 7 months), and gender (factor: male, female).

rowTree Contains the phylogenetic tree.

Source

Derived from the VatanenT_2016 study, obtained via the curatedMetagenomicData package. See inst/scripts/VatanenT_2016_subset.R for how the subset was derived. Original study: https://pmc.ncbi.nlm.nih.gov/articles/PMC4950857/

Index

∗ **datasets**
DiPPER-extdata, 7
fit_example, 8
tse_hintikka, 15
VatanenT_2016_subset, 16
∗ **internal**
prep_dipper_data, 10
run_dipper, 12

dipper, 2, *4*, *9*, *12–14*
DiPPER-extdata, 7

fit_example, *8*, 8
fit_hintikka *(*DiPPER-extdata*)*, 7
fit_vatanen *(*DiPPER-extdata*)*, 7

plot.dipper_fit, 9
prep_dipper_data, 10
print.dipper_fit, *6*, 11

run_dipper, 12

summary.dipper_fit, *9*, 14

tse_hintikka, *7, 8*, 15

VatanenT_2016_subset, *7, 8*, 16