Package 'metalcor'

May 12, 2026

Title Meta-Analysis of Correlated Genetic Association Studies

Version 1.0.0

Description The main function performs meta-analysis of genetic association study summary statistics that may be correlated due to cryptic relatedness or other confounders, generalizing inverse variance weighted methods. The function that estimates the correlation structure is also provided standalone. Another key innovation, the estimation of the correlation parameter from the median product of correlated standard normal variables, is provided, as well as a complete set of functions for their underlying distribution: density, cumulative, quantile, and random deviates. Described in Tu and Ochoa (2025) <doi:10.1101/2025.05.10.653279>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 8.0.0

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Imports stats, dplyr, tibble, tidyselect

Config/testthat/edition 3

VignetteBuilder knitr

URL `https://github.com/OchoaLab/metalcor`

BugReports `https://github.com/OchoaLab/metalcor/issues`

NeedsCompilation no

Author Alejandro Ochoa [aut, cre] (ORCID:
<`https://orcid.org/0000-0003-4928-3403`>),
Tiffany Tu [aut] (ORCID: <`https://orcid.org/0000-0002-9357-6660`>)

Maintainer Alejandro Ochoa <`alejandro.ochoa@duke.edu`>

Repository CRAN

Date/Publication 2026-05-12 19:00:02 UTC

Contents

`estimate_R` *Estimate Covariance Matrix of Study Z-scores*

Description

The default "median" estimator is expected to be more robust to outliers (present due to causal variants and linkage disequilibrium). For completeness, the more standard "mean" estimator is also implemented, which is better when there are no outliers. The determinant is also tested, and if its absolute value is less than `tol` an error is triggered, warning the user that perhaps the same study was included multiple times (as this is otherwise expected to be extremely rare).

Usage

```
estimate_R(Z, median = TRUE, tol = 1e-08)
```

Arguments

`Z`	Matrix of z-scores, loci along rows, `S` studies along columns. Missing values can be present and are appropriately ignored.
`median`	If `TRUE`, uses `rho_from_median()` to calculate covariance values from sample medians; otherwise, sample means are used.
`tol`	Determinant tolerance for singularity test

Value

The `S`-by-`S` covariance matrix.

See Also

`rho_from_median()`

Examples

```
# simulate some correlated Z-scores for 3 studies and `m` loci
m <- 10000
# true correlation matrix
R <- matrix(
  c( 1.0, 0.2, 0.0,
     0.2, 1.0, 0.1,
     0.0, 0.1, 1.0 ),
  nrow = 3, ncol = 3
)
# use Cholesky decomposition to simulate via affine transform
L <- chol( R )
Z <- matrix( rnorm( m * 3 ), nrow = m, ncol = 3 ) %*% L

# calulate estimates
```

```
estimate_R( Z )
estimate_R( Z, median = FALSE )
# compare to naive covariance estimate, which does not assume zero mean for z-scores
cov( Z, use = 'pairwise.complete.obs' )
```

`metalcor` *Meta-Analysis of Correlated Genetic Association Studies*

Description

This function accepts the summary statistics from several studies, estimates their z-score correlation matrix, and performs the meta-analysis taking those correlations into account.

Usage

```
metalcor(studies, median = TRUE, df = 1, tol = 1e-08)
```

Arguments

`studies`	A list of `S` studies, each of which is a data frame with the following required columns: `id`, `chr`, `pos`, `beta`, `se`, `n`. Column names have to match exactly. `id` is used to match variants between studies (they must be unique for different variants, and be the same for the same variant). `chr` and `pos` are used solely to sort variants after the studies have been merged. `beta` and `se` are required to perform meta-analysis. `n` is used to report locus-specific total sample sizes under missingness.
`median`	If `TRUE` (default), estimates correlations from sample medians instead of means. Passed to `estimate_R()`.
`df`	Degree of freedom for calculating the p-values from the z-scores (default of 1 is best practically always).
`tol`	Determinant tolerance for singularity test of R estimate

Value

A list with two named elements:

- `assoc`: the meta-analyzed study, a tibble with columns `id`, `chr`, `pos`, `n`, `n_studies`, `beta`, `se`, `z`, and `p`; sorted by `chr` and `pos`.
- `R`: the `S`-by-`S` estimated z-score covariance matrix

See Also

`estimate_R()`

Examples

```
# construct two toy studies just to run example, with minimal columns
study1 <- data.frame(
  id = paste0( 'rs', 1:5 ),
  chr = 1,
  pos = 1:5,
  n = 2000,
  beta = rnorm( 5 ),
  se = rnorm( 5 )
)
# note the second study is missing the 5th SNP, this is fine
study2 <- data.frame(
  id = paste0( 'rs', 1:4 ),
  chr = 1,
  pos = 1:4,
  n = 5000,
  beta = rnorm( 4 ),
  se = rnorm( 4 )
)

# gather the studies in a list
studies <- list( study1, study2 )
# this performs the meta-analysis modeling covariance!
out <- metalcor( studies )
# this is the meta-analyzed association table
out$assoc
# and this is the estimated study covariance matrix
out$R
```

`prodcor` *The Correlated Standard Normal Product Distribution*

Description

Density, cumulative, quantile, and random generation for the product of two correlated standard normal variables with correlation parameter `rho`.

Usage

```
dprodcor(x, rho)

pprodcor(x, rho, eps = 1e-08)

qprodcor(p, rho)

rprodcor(n, rho)
```

Arguments

`x`	Vector of quantiles.
`rho`	Correlation parameter, must be a scalar in [-1, 1].
`eps`	Near zero value to use for integrating around the divergence at `x=0`.
`p`	Vector of probabilities.
`n`	Number of observations. If `length(n)` > 1, the length is taken to be the number required.

Details

`pprodcor` does not have a closed form, so it is calculated by numerical integration of `dprodcor`. Furthermore, `dprodcor(0)` is infinite, so in order for numerical integration to succeed, we exclude the window `c(-eps, eps)` for arguments near zero. `eps` is the initial value, but if failure is still encountered, `eps` is incremented by factors of 10 until successful. (For positive arguments integration from above is used instead, subtracted from 1.) `qprodcor` also does not have a closed form, so it is calculated using a root finder on `pprodcor`, which makes it very slow.

Value

`dprodcor` gives the density, `pprodcor` the cumulative, and `qprodcor` the quantile function. `rprodcor` generates random deviates. The length of the result is determined by `n` for `rnorm`, and it is the length of `x` or `p` for the other functions.

References

Nadarajah, Saralees, and Tibor K. Pogány. "On the Distribution of the Product of Correlated Normal Random Variables." Comptes Rendus Mathematique 354, no. 2 (2016): 201–4. doi:10.1016/j.crma.2015.10.019.

Gaunt, Robert E. "The Basic Distributional Theory for the Product of Zero Mean Correlated Normal Random Variables." Statistica Neerlandica 76, no. 4 (2022): 450–70. doi:10.1111/stan.12267.

Examples

```
n <- 10
x <- rnorm( n )
p <- runif( n )
rho <- 0.8
dprodcor( x, rho )
pprodcor( x, rho )
qprodcor( p, rho )
rprodcor( n, rho )
```

`rho_from_median` *Estimated Correlations from Sample Medians of the Correlated Standard Normal Product Distribution*

Description

This code implements estimation from the sample median of the correlation of a product of two correlated standard normal variables. While this estimation is straightforward from the mean, using the median requires the transformation implemented here because the distribution has heavy tails, so for positive correlations the median is much smaller than the mean. The median version is desired due to its robustness to outliers.

Usage

```
rho_from_median(x)
```

Arguments

`x`	vector of sample medians

Details

The highest correlation of `rho = 1` results in a median of only `q~0.455`. To permit application to arbitrary data, including data that does not satisfy assumptions and is therefore inflated, the function returns `x/q` for elements whose absolute values of `x/q` exceed 1, and can therefore return values that exceed [-1, 1].

Value

Vector of estimated correlations, extended for inflated cases.

See Also

`qprodcor()`

Examples

```
# a large number of samples is required for the median to yield an accurate estimate
n <- 10000
rho <- 0.3
x <- median( rprodcor( n, rho ) )
rho_from_median( x )
```

Index