Package 'getspres'

October 13, 2022

Title SPRE Statistics for Exploring Heterogeneity in Meta-Analysis

Version 0.2.0

Description An implementation of SPRE (standardised predicted random-effects) statistics in R to explore heterogeneity in genetic association meta-analyses, as described by Magosi et al. (2019) <doi:10.1093/bioinformatics/btz590>. SPRE statistics are precision weighted residuals that indicate the direction and extent with which individual study-effects in a meta-analysis deviate from the average genetic effect. Overly influential positive outliers have the potential to inflate average genetic effects in a meta-analysis whilst negative outliers might lower or change the direction of effect. See the 'getspres' website for documentation and examples <https://magosil86.github.io/getspres/>.

Depends R (>= 3.1.0)

License MIT + file LICENSE

URL https://magosil86.github.io/getspres/

BugReports https://github.com/magosil86/getspres/issues

LazyData true

RoxygenNote 7.1.1

Suggests knitr (>= 1.10.5), testthat, covr, rmarkdown

VignetteBuilder knitr

Imports metafor (>= 1.9-6), dplyr (>= 0.4.1), plotrix (>= 3.5-12), colorspace (>= 1.2-6), RColorBrewer (>= 1.1-2), colorRamps (>= 2.3)

NeedsCompilation no

Author Lerato E Magosi [aut],
Jemma C Hopewell [aut],
Martin Farrall [aut],
Lerato E Magosi [cre]

Maintainer Lerato E Magosi <magosil86@gmail.com>

Repository CRAN

Date/Publication 2021-05-09 05:30:03 UTC

R topics documented:

`getspres` *Exploring Heterogeneity in Meta-Analysis with* SPRE *Statistics.*

Description

`getspres` computes *SPRE* (standardised predicted random-effects) statistics to identify outlier studies in genetic association meta-analyses which might have undue influence on the average genetic effect leading to inflated genetic signals.

Usage

```
getspres(beta_in, se_in, study_names_in, variant_names_in, ...)

## Default S3 method:
getspres(
  beta_in,
  se_in,
  study_names_in,
  variant_names_in,
  tau2_method = "DL",
  verbose_output = FALSE,
  ...
)
```

Arguments

`beta_in`	A numeric vector of study effect-sizes e.g. log odds-ratios.
`se_in`	A numeric vector of standard errors, genomically corrected at study-level.
`study_names_in`	A character vector of study names.
`variant_names_in`	A character vector of variant names e.g. rsIDs.
`...`	other arguments.
`tau2_method`	A character scalar, specifying the method that should be used to estimate heterogeneity either through DerSimonian and Laird's moment-based estimate "DL" or restricted maximum likelihood "REML". Note: The REML method uses the iterative Fisher scoring algorithm (step length = 0.5, maximum iterations = 10000) to estimate tau2. Default is "DL".
`verbose_output`	An optional boolean to display intermediate output. (Default is `FALSE`).

Details

SPRE statistics are precision-weighted residuals that summarise the direction and extent with which observed study effects in a meta-analysis differ from the summary (or average genetic) effect. See the getspres website for more information, documentation and examples.

`getspres` takes as input study effect-size estimates and their corresponding standard errors (i.e. summary data). Study effect estimates could be in the form of linear regression coefficients or log-transformed regression coefficients (per-allele log odds ratios) from logistic regression.

`getspres` uses inverse-variance weighted meta-analysis models in the `metafor` R package to calculate *SPRE* statistics.

Value

Returns a list containing:

- number_variants A numeric scalar for the number of variants
- number_studies A numeric scalar for the number of studies
- spre_dataset A dataframe that is a dataset of computed SPRE statistics and contains the following fields:
  - beta , study effect-size estimates
  - se , corresponding standard errors of the study effect-size estimates
  - variant_names , variant names
  - study_names , study names
  - study , study numbers
  - snp , snp numbers
  - tau2 , tau_squared, estimate of amount of between-study variance
  - I2 , I_squared, heterogeneity index (Higgins inconsistency metric) representing proportion of total observed variation due to between-study variance
  - Q , Q-statistic (Cochran's Q)
  - xb , prediction excluding random effects
  - xbse , standard error of prediction excluding random effects
  - xbu , predictions including random effects
  - stdxbu , standard error of prediction (fitted values) including random effects
  - hat , leverage a.k.a diagonal elements of the projection hat matrix
  - rawresid , raw residuals
  - uncondse , unconditional standard errors
  - spre , *SPRE* statistics (standardised predicted random effects) i.e. raw residuals divided by the unconditional standard errors

Methods (by class)

- `default`: Computes *SPRE* statistics in genetic association meta-analyses

See Also

https://magosil86.github.io/getspres/ to the visit `getspres` website.

Examples

```
library(getspres)


# Calculate SPRE statistics for a subset of variants in the heartgenes214 dataset.
# heartgenes214 is a case-control GWAS meta-analysis of coronary artery disease.
# To learn more about the heartgenes214 dataset ?heartgenes214

# Calculating SPRE statistics for 3 variants in heartgenes214

heartgenes3 <- subset(heartgenes214,
    variants %in% c("rs10139550", "rs10168194", "rs11191416"))

getspres_results <- getspres(beta_in = heartgenes3$beta_flipped,
                             se_in = heartgenes3$gcse,
                       study_names_in = heartgenes3$studies,
                     variant_names_in = heartgenes3$variants)


# Explore results generated by the getspres function
str(getspres_results)

# Retrieve number of studies and variants
getspres_results$number_variants
getspres_results$number_studies

# Retrieve SPRE dataset
df_spres <- getspres_results$spre_dataset
head(df_spres)

# Extract SPREs from SPRE dataset
head(spres <- df_spres[, "spre"])



# Exploring available options in the getspres function:
#     1. Estimate heterogeneity using "REML", default is "DL"
#     2. Calculate SPRE statistics verbosely
getspres_results <- getspres(beta_in = heartgenes3$beta_flipped,
                             se_in = heartgenes3$gcse,
                       study_names_in = heartgenes3$studies,
                     variant_names_in = heartgenes3$variants,
                          tau2_method = "REML",
                       verbose_output = TRUE)

str(getspres_results)
```

`heartgenes214` *heartgenes214.*

Description

heartgenes214 is a multi-ethnic GWAS meta-analysis dataset for coronary artery disease.

Usage

```
heartgenes214
```

Format

A data frame with seven variables:

`beta_flipped` Effect-sizes expressed as log odds ratios. Numeric

`gcse` Standard errors

`studies` Names of participating studies

`variants` Names of genetic variants/SNPs

`cases` Number of cases in each participating study

`controls` Number of controls in each participating study

`fdr214_gwas46` Flag indicating GWAS significant variants, 1: Not GWAS-significant, 2: GWAS-significant

Details

It comprises summary data (effect-sizes and their corresponding standard errors) for 48 studies (68,801 cases and 123,504 controls), at 214 lead variants independently associated with coronary artery disease ($P < 0.00005$, FDR < 5%). Of the 214 lead variants, 44 are genome-wide significant ($p < 5e-08$). The meta-analysis dataset is based on individuals of: African American, Hispanic American, East Asian, South Asian, Middle Eastern and European ancestry.

The study effect-sizes have been flipped to ensure alignment of the effect alleles.

Standard errors were genomically corrected at the study-level.

Source

Magosi LE, Goel A, Hopewell JC, Farrall M, on behalf of the CARDIoGRAMplusC4D Consortium (2017) Identifying systematic heterogeneity patterns in genetic association meta-analysis studies. PLoS Genet 13(5): e1006755. https://doi.org/10.1371/journal.pgen.1006755.

https://magosil86.github.io/getmstatistic/

plotspres plotspres *generates forest plots showing* SPRE *statistics.*

Description

Forest plots showing *SPRE* (standardised predicted random-effects) statistics can be useful in highlighting overly influential outlier studies with the potential to inflate summary effect estimates in genetic association meta-analyses.

Usage

```
plotspres(beta_in, se_in, study_names_in, variant_names_in, spres_in, ...)

## Default S3 method:
plotspres(
  beta_in,
  se_in,
  study_names_in,
  variant_names_in,
  spres_in,
  spre_colour_palette = c("mono_colour", "black"),
  set_studyNOs_as_studyIDs = FALSE,
  set_study_field_width = "%02.0f",
  set_cex = 0.66,
  set_xlim,
  set_ylim,
  set_at,
  tau2_method = "DL",
  adjust_labels = 1,
  save_plot = TRUE,
  verbose_output = FALSE,
  ...
)
```

Arguments

beta_in	A numeric vector of observed study effects e.g. log odds-ratios.
se_in	A numeric vector of standard errors, genomically corrected at study-level.
study_names_in	A character vector of study names.
variant_names_in	A character vector of variant names e.g. rsIDs.
spres_in	A numeric vector of *SPRE* statistics.
...	other arguments.

`spre_colour_palette`
: An optional character vector specifying the colour palette that should be used for observed study effects. There are 3 types of colour palettes available, namely: "mono_colour", "dual_colour" and "multi_colour"; with the "dual_colour" palette, observed study effects with negative *SPRE* statistics are coloured differently from those with positive *SPRE* statistics, and with the "multi_colour" palette observed study effects are colored in a gradient according to the *SPRE* statistic values. Default palette option is `spre_colour_palette = c("mono_colour", "black")`.

`set_studyNOs_as_studyIDs`
: An optional boolean specifying whether study numbers should be used as study IDs in the forest plot. Default is `FALSE`.

`set_study_field_width`
: An optional character vector of format strings, akin to the fmt character vector in the sprintf function. (Default is `set_study_field_width = "%02.0f"`).

`set_cex`
: An optional character scalar and symbol expansion factor indicating the percentage by which text and symbols should be scaled relative to the reference; e.g. 1=reference, 1.3 is 30% larger, 0.3 is 30% smaller. (Default is `cex = 0.66`).

`set_xlim`
: An optional numeric vector of length 2 indicating the horizontal limits of the plot region.

`set_ylim`
: An optional numeric vector of length 2 indicating the y-axis limits of the plot.

`set_at`
: An optional numeric vector indicating position of the x-axis tick marks and corresponding labels.

`tau2_method`
: An optional character scalar, specifying the method that should be used to estimate heterogeneity either through DerSimonian and Laird's moment-based estimate "DL" or restricted maximum likelihood "REML". Note: The REML method uses the iterative Fisher scoring algorithm (step length = 0.5, maximum iterations = 10000) to estimate tau2. Default is "DL".

`adjust_labels`
: An optional numeric scalar value that tweaks label (column header) positions. (Default is `adjust_labels = 1`).

`save_plot`
: An optional boolean to save forestplot as a tiff file. Default is `TRUE`.

`verbose_output`
: An optional boolean to display intermediate output. (Default is `FALSE`).

Details

`plotspres` takes as input *SPRE* statistics, observed study effects and corresponding standard errors (i.e. summary data). The observed study effects (i.e. study effect-size estimates) could be association statistics from either quantitative or binary trait meta-analyses, for instance, linear regression coefficients might be employed for quantitative traits and log-transformed logistic regression coefficients (per-allele log odds ratios) used for case-control meta-analyses. *SPRE* statistics can be calculated using the `getspres` function.

`plotspres` uses inverse-variance weighted fixed and random-effects meta-analysis models in the `metafor` R package to generate forestplots.

Value

Returns a list containing:

- number_variants A numeric scalar indicating the number of variants
- number_studies A numeric scalar indicating the number of studies
- fixed_effect_results A list of fixed-effect meta-analysis results for each variant examined
- random_effects_results A list of random-effects meta-analysis results for each variant examined
- spre_forestplot_dataset A dataframe of the data provided by the user for analysis which contains the following fields:
  - beta , study effect-size estimates
  - se , corresponding standard errors of study effect-size estimates
  - variant_names , variant names
  - study_names , study names
  - spre , *SPRE* (standardised predicted random-effects) statistics
  - study_numbers , study numbers
  - variant_numbers , variant numbers

Methods (by class)

- `default`: Generates forest plots showing *SPRE* statistics

See Also

`getspres` to calculate *SPRE* statistics and the `metafor` package to explore implementations of fixed and random-effects meta-analysis models in R. To access more information and examples visit the getspres website at: https://magosil86.github.io/getspres/.

Examples

```
library(getspres)


# Generate a forest plot showing SPRE statistics for variants in heartgenes214.
# heartgenes214 is a case-control GWAS meta-analysis of coronary artery disease.
# To learn more about the heartgenes214 dataset ?heartgenes214

# Calculating SPRE statistics for 3 variants in heartgenes214

heartgenes3 <- subset(heartgenes214,
    variants %in% c("rs10139550", "rs10168194", "rs11191416"))

getspres_results <- getspres(beta_in = heartgenes3$beta_flipped,
                             se_in = heartgenes3$gcse,
                       study_names_in = heartgenes3$studies,
                     variant_names_in = heartgenes3$variants)

# Explore results generated by the getspres function
```

```
str(getspres_results)

# Retrieve number of studies and variants
getspres_results$number_variants
getspres_results$number_studies

# Retrieve SPRE dataset
df_spres <- getspres_results$spre_dataset
head(df_spres)

# Extract SPREs from SPRE dataset
head(spres <- df_spres[, "spre"])


# Generating forest plots showing SPREs for variants in heartgenes3

# Forest plot with default settings
# Tip: To store plots set save_plot = TRUE (useful when generating multiple plots)
plotspres_res <- plotspres(beta_in = df_spres$beta,
                           se_in = df_spres$se,
                     study_names_in = as.character(df_spres$study_names),
                   variant_names_in = as.character(df_spres$variant_names),
                          spres_in = df_spres$spre,
                         save_plot = FALSE)

# Explore results generated by the plotspres function

# Retrieve number of studies and variants
plotspres_res$number_variants
plotspres_res$number_studies

# Retrieve fixed and random-effects meta-analysis results
fixed_effect_res <- plotspres_res$fixed_effect_results
random_effects_res <- plotspres_res$random_effects_results

# Retrieve dataset that was used to generate forest plots
df_plotspres <- plotspres_res$spre_forestplot_dataset



# Retrieve more detailed meta-analysis output
str(plotspres_res)



# Explore available options for plotspres forest plots:
#   1. Colorize study-effect estimates according to SPRE statistic values
#   2. Label studies by study number instead of study names
#   3. Format study labels (useful when using study numbers as study labels)
#   4. Change text size
#   5. Adjust x and y axes limits
#   6. Change method used to estimate amount of heterogeneity from "DL" to "REML"
#   7. Run verbosely to show intermediate results
```

```
#   8. Adjust label (i.e. column header) positions
#   9. Save plot as a tiff file (useful when generating multiple plots)

# Colorize study-effect estimates according to SPRE statistic values

# Use a dual colour palette for observed study effects so that study effect estimates
#   with negative SPRE statistics are coloured differently from those with positive
#   SPRE statistics.
plotspres_res <- plotspres(beta_in = df_spres$beta,
                           se_in = df_spres$se,
                  study_names_in = as.character(df_spres$study_names),
                 variant_names_in = as.character(df_spres$variant_names),
                         spres_in = df_spres$spre,
               spre_colour_palette = c("dual_colour", c("blue","black")),
                         save_plot = FALSE)


# Use a multi-colour palette for observed study effects so that study effects estimates
#   are colored in a gradient according to SPRE statistic values.
#   Available multi-colour palettes:
#
#       gr_devices_palettes: "rainbow", "cm.colors", "topo.colors", "terrain.colors"
#                            and "heat.colors"
#
#       colorspace_hcl_hsv_palettes: "rainbow_hcl", "diverge_hcl", "terrain_hcl",
#                                    "sequential_hcl" and "diverge_hsl"
#
#       color_ramps_palettes: "matlab.like", "matlab.like2", "magenta2green",
#                             "cyan2yellow", "blue2yellow", "green2red",
#                             "blue2green" and "blue2red"

plotspres_res <- plotspres(beta_in = df_spres$beta,
                           se_in = df_spres$se,
                  study_names_in = as.character(df_spres$study_names),
                 variant_names_in = as.character(df_spres$variant_names),
                         spres_in = df_spres$spre,
               spre_colour_palette = c("multi_colour", "rainbow"),
                         save_plot = FALSE)

# Exploring other options in the plotspres function.
#   Label studies by study number instead of study names (option: set_studyNOs_as_studyIDs)
#     Format study labels (option: set_study_field_width)
#     Adjust text size (option: set_cex)
#     Adjust x and y axes limits (options: set_xlim, set_ylim)
#    Change method used to estimate heterogeneity from "DL" to "REML" (option: tau2_method)
#     Adjust position of x-axis tick marks (option: set_at)
#     Run verbosely (option: verbose_output)

df_rs10139550 <- subset(df_spres, variant_names == "rs10139550")
plotspres_res <- plotspres(beta_in = df_rs10139550$beta,
                           se_in = df_rs10139550$se,
                  study_names_in = as.character(df_rs10139550$study_names),
                 variant_names_in = as.character(df_rs10139550$variant_names),
```

```
                          spres_in = df_rs10139550$spre,
               spre_colour_palette = c("multi_colour", "matlab.like"),
          set_studyNOs_as_studyIDs = TRUE,
             set_study_field_width = "%03.0f",
                           set_cex = 0.75, set_xlim = c(-2,2), set_ylim = c(-1.5,51),
                            set_at = c(-0.6, -0.4, -0.2,  0.0,  0.2,  0.4,  0.6),
                      tau2_method = "REML", verbose_output = TRUE,
                        save_plot = FALSE)

# Adjust label (i.e. column header) position, also keep plot in graphics window rather
#     than save as tiff file
df_rs10139550_3studies <- subset(df_rs10139550, as.numeric(df_rs10139550$study_names) <= 3)

# Before adjusting label positions
plotspres_res <- plotspres(beta_in = df_rs10139550_3studies$beta,
                             se_in = df_rs10139550_3studies$se,
                   study_names_in = as.character(df_rs10139550_3studies$study_names),
                  variant_names_in = as.character(df_rs10139550_3studies$variant_names),
                           spres_in = df_rs10139550_3studies$spre,
              spre_colour_palette = c("dual_colour", c("blue","black")),
                        save_plot = FALSE)

# After adjusting label positions
plotspres_res <- plotspres(beta_in = df_rs10139550_3studies$beta,
                             se_in = df_rs10139550_3studies$se,
                   study_names_in = as.character(df_rs10139550_3studies$study_names),
                  variant_names_in = as.character(df_rs10139550_3studies$variant_names),
                           spres_in = df_rs10139550_3studies$spre,
              spre_colour_palette = c("dual_colour", c("blue","black")),
                    adjust_labels = 1.7, save_plot = FALSE)
```

Index