Package: pleiotest (via r-universe)

November 1, 2024

Title Fast Sequential Pleiotropy Test

Version 1.0.0

Description It performs a fast multi-trait genome-wide association analysis based on seemingly unrelated regressions. It tests for pleiotropic effects based on a series of Intersection-Union Wald tests. The package can handle large and unbalanced data and plot results.

License MIT + file LICENSE

Depends R (>= 2.10)

Imports Rcpp, RColorBrewer

LinkingTo Rcpp, RcppArmadillo

URL https://github.com/FerAguate/pleiotest

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Encoding UTF-8

Repository https://feraguate.r-universe.dev

RemoteUrl https://github.com/feraguate/pleiotest

RemoteRef HEAD

RemoteSha 2cde7fb28ac357661f17200152a9ee254874dff2

Contents

`identify_subsets` *Internal function to identify sub-sets of data and return a list with IDs.*

Description

This function is used internally in pleioR.

Usage

```
identify_subsets(trait, id)
```

Arguments

`trait`	character indicating traits.
`id`	character indicating IDs.

Value

list with an ID matrix and ID subsets.

Author(s)

Original code by Fernando Aguate.

`manhattan_plot` *Single Trait Manhattan plot*

Description

Manhattan plot of results from mt_gwas function.

Usage

```
manhattan_plot(mt_gwas_results, trait, bp_positions, ...)
```

Arguments

`mt_gwas_results`	Object returned by mt_gwas
`trait`	integer indicating the position of the trait (see: names(mt_gwas_results)) to be plotted.
`bp_positions`	dataframe with SNPs base pair positions. colnames msut be 'chr' and 'position', rownames must be SNP identifiers matching names in mt_gwas.
`...`	further graphical parameters. Options include: title=, bty=, pch=, cex.lab=, and cex.main=.

Value

Manhattan plot

`mt_gwas` *Multi-trait Genome wide association model.*

Description

Performs a multi-trait model with correlated errors (seemingly unrelated regressions), and generates results by trait in a list.

Usage

```
mt_gwas(pleio_results, save_at = NULL)
```

Arguments

`pleio_results`	object of class pleio_class (returned by pleioR() function).
`save_at`	character with directory and/or file name (.rdata) to save the results. This is useful when handling multiple results such as in parallel jobs.

Value

list with by trait dataframes that contain results of the multi-trait model.

`pleioR` *Fit a multi-trait model to test for genetic pleiotropy*

Description

Fits a seemingly unrelated regression with, possibly unbalanced data, and/or covariates. It returns a pleio_class object to perform the sequential test with pleio_test() or to obtain by-trait estimates with mt_gwas().

Usage

```
pleioR(pheno, geno, i = NULL, j = NULL, covariates = NULL, drop_subsets = 10)
```

Arguments

pheno	dataframe with phenotypic data. Must have columns 'id', 'trait', and 'y'. Column 'y' must contain the observations for the corresponding 'trait' and 'id'. See function melt() in the 'reshape2' package for a simple formatting of your data.
geno	matrix with SNPs in columns and IDs in rownames. This can also be a memory-mapped matrix returned by BEDMatrix() in the 'BEDMatrix' package.
i	integers indexing rows from geno to use in the model.
j	integers indexing columns from geno to use in the model. Useful when working with multiple jobs in parallel.
covariates	(optional) dataframe or matrix containing covariates in columns and IDs as rownames. These IDs must match those in geno.
drop_subsets	minimum sample size of sub-data sets to consider for analysis, 10 by default. When working with unbalanced data (a.k.a. fragmented data), save computation time by dropping small fragments of data.

Value

pleio_class list of left and right hand side solutions of the model.

Examples

```
# Random generated example with 3 traits, 1e4 individuals, 1000 SNPs and 10% missing values.
sim1 <- pleio_simulate(n_traits = 3, n_individuals = 1e4, n_snp = 1e3, percentage_mv = 0.1)
pleio_model <- pleioR(pheno = sim1$pheno, geno = sim1$geno)
pleio_model_test <- pleio_test(pleio_model)
```

---

pleio_ideogram *Plot ideogram from pleio_test results*

Description

Plots genomic segments that contain significant pleiotropic SNPs using results of pleio_test(). It also returns a dataframe with segment information.

Usage

```
pleio_ideogram(
  pleio_res,
  alpha = "bonferroni05",
  n_traits = 2,
  bp_positions,
  window_size = 1e+06,
  centromeres = NULL,
  color_bias = 1,
  set_plot = T,
  set_legend = T,
```

```
  set_ylim_prop = 1.1,
  ...
)
```

Arguments

pleio_res	list returned by pleio_test().
alpha	numeric threshold for significance level (Bonferroni correction by default).
n_traits	integer indicating the level of pleiotropy to test (a.k.a. number of traits).
bp_positions	dataframe with colnames 'chr' and 'pos' indicating the chromosome and position for each SNP. Rownames must contain SNP names matching results of pleio_test.
window_size	numeric value indicating the minimum size (in base pairs) of the genomic region that contains significant SNPs.
centromeres	string 'human' or dataframe (or matrix) with chromosome and position (in mbp) of the centromeres in the first and second columns. If NULL (default) does not plot the centromeres.
color_bias	number for bias of the color scale. See help(colorRampPalette). By default color_bias = 1
set_plot	logical indicating whether to plot the ideogram (TRUE by default).
set_legend	logical indicating whether to plot a legend (TRUE by default).
set_ylim_prop	numeric proportion of upper margin to fit the legend (no margin by default). 1 = no margin, 1.1 = 10% left for margin, etc.
...	more plot arguments.

Value

Ideogram plot and a dataframe with genomic segments information.

See Also

pleio_plot

---

pleio_plot *Pleiotropic manhattan plot*

Description

Plots the p-values that test the hypothesis of pleiotropic effects on n_traits. This function also returns a dataframe with information of the significant SNPs.

Usage

```
pleio_plot(
  pleio_res,
  alpha = "bonferroni05",
  n_traits = 2,
  bp_positions = NULL,
  set_colors = NULL,
  set_text = NULL,
  set_plot = TRUE,
  chr_spacing = 1e+05,
  ...
)
```

Arguments

pleio_res	object returned by pleio_test().
alpha	numeric threshold for significance level (Bonferroni correction by default).
n_traits	integer indicating the level of pleiotropy to test (a.k.a. number of traits).
bp_positions	dataframe with colnames 'chr' and 'pos' indicating the chromosome and position for each SNP. Rownames must contain SNP names matching results of pleio_test.
set_colors	string with 3 colors to use in the plot (by default: c('goldenrod4', 'brown4', 'royalblue2')).
set_text	dataframe or matrix with strings to add as text to identify SNPs or genes. Rownames must be SNP names matching results of pleio_test. The first column of the dataframe must have strings to plot as text.
set_plot	logical indicating whether to return the manhattan plot (TRUE by default).
chr_spacing	integer indicating the spacing (in base pair positions) between chromosomes. 1e5 by default.
...	additional graphic parameters for the plot.

Value

Manhattan plot and dataframe with information related to significant SNPs.

---

pleio_simulate *Create simulations*

Description

Example function to create simulations with no effects.

Usage

```
pleio_simulate(n_traits, n_individuals, n_snp, percentage_mv = 0)
```

Arguments

n_traits	number of traits to simulate.
n_individuals	number of individuals to simulate.
n_snp	number of SNPs to simulate.
percentage_mv	proportion of missing values. By default = 0.

Value

a list with pheno and geno to test the pleioR function.

Author(s)

Original code by Fernando Aguate.

Examples

```
sim1 <- pleio_simulate(n_traits = 3, n_individuals = 1e4, n_snp = 1e3, percentage_mv = 0.1)
```

---

pleio_test *Sequential Wald test for pleiotropy*

Description

Performs the sequential test of pleiotropic effects using results of pleioR().

Usage

```
pleio_test(
  pleio_results,
  loop_breaker = 1,
  save_at = NULL,
  contrast_matrices_list = NULL
)
```

Arguments

pleio_results	pleio_class object returned by pleioR().
loop_breaker	numeric value for a maximum p-value used to stop the sequence if a higher p-value is obtained. This saves computation time if there are many tests to perform.
save_at	character with directory and/or file name (.rdata) to save the results. This is useful when handling multiple results such as in parallel jobs.
contrast_matrices_list	user-specified contrast matrices within a list of lists, or a single contrast matrix (see example). Each matrix must have the same number of columns, and must be equal to the number of traits.

Value

list of p-values, indices, and trait numeric identifier.

Examples

```
# Example of user-specified contrast matrices with 3 traits
cm1 <- matrix(c(-1, 0, 1), ncol = 3)
cm2 <- matrix(c(0, -1, 1), ncol = 3)
contrast_matrices <- list('1vs3' = list(cm1), '2vs3' = list(cm2))
# or a single contrast matrix as:
contrast_matrices <- cm1
```

`xrsx_xrsy` *Calculate XRsX and XRsY*

Description

internal function to calculate crossproducts within pleioR.

Usage

```
xrsx_xrsy(id_matrix, sets_rs, xx, xy)
```

Arguments

`id_matrix`	matrix of IDs
`sets_rs`	list of inverses of matrix R
`xx`	numeric vector with crossproducts of the X matrix
`xy`	matrix with X transpose Y products.

Index