Package: mrpipeline (via r-universe)

June 10, 2026

Type Package

Title Implements a Ready-for-Use Mendelian Randomisation Pipeline

Version 0.1.0

Author Jamie Robinson

Maintainer Jamie Robinson `<jamie.robinson@biogen.com>`

Description This package implements a pipeline which in turns allows for simple and generally ``hands-free'' Mendelian randomisation analyses to be run. Data may be used from the OpenGWAS DB or locally, using .vcf files. Analyses include MR, colocalisation and standard MR sensitivity analyses. Please see the documentation for more details.

License What license is it under?

Encoding UTF-8

LazyData true

Depends R (>= 3.6.0)

Imports TwoSampleMR, ieugwasr, gwasglue, gwasvcf, epigraphdb, coloc, tools, utils, jsonlite, dplyr, plyr, tidyr, tibble, ggplot2, gassocplot, parallel, VariantAnnotation, magrittr

Remotes github::MRCIEU/TwoSampleMR, github::MRCIEU/ieugwasr, github::MRCIEU/gwasglue, github::MRCIEU/gwasvcf, github::jrs95/gassocplot, bioc::3.12/VariantAnnotation

RoxygenNote 7.1.2

Suggests plotly, biomaRt, knitr, rmarkdown

VignetteBuilder knitr

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

Date/Publication 2022-09-25 12:07:00 UTC

RemoteUrl https://github.com/jwr-git/mrpipeline

RemoteRef HEAD

RemoteSha 2c84a8deaa86703718743cddb57fc8815fc8202d

Contents

`.calc_pve` *Calculate PVE*

Description

Calculates proportion of variance explained. From S1 Text

Usage

```
.calc_pve(b, maf, se, n)
```

Arguments

`b`	Vector or number, beta
`maf`	Vector or number, minor allele frequency
`se`	Vector or number, standard error of beta
`n`	Vector or number, sample size

Value

Vector or number, proportion of variance explained

`.cdat_from_mixed` *Helper function to extract colocalisation regions for when one dataset comes from a local file and another from OpenGWAS.*

Description

Helper function to extract colocalisation regions for when one dataset comes from a local file and another from OpenGWAS.

Usage

```
.cdat_from_mixed(f1, f2, chrpos, verbose = TRUE)
```

Arguments

`f1`	File path or OpenGWAS ID for trait 1
`f2`	File path or OpenGWAS ID for trait 2
`chrpos`	Character of the format chr:pos1-pos2
`verbose`	Display verbose information (Optional, boolean)

Value

list of coloc-ready data

See Also

[gwasglue::gwasvcf_to_coloc()], [gwasglue::ieugwasr_to_coloc()]

`.chunk` *Splits vector into chunks*

Description

Splits vector into chunks

Usage

```
.chunk(x, n)
```

Arguments

`x`	Vector
`n`	Number of chunks to create

Value

list of chunks

`.coloc_sub` *Sub-function for the colocalisation analyses*

Description

Sub-function for the colocalisation analyses

Usage

```
.coloc_sub(
  dat1,
  dat2,
  min_snps = 100,
  p1 = 1e-04,
  p2 = 1e-04,
  p12 = 1e-05,
  susie = FALSE,
  bfile = NULL,
  plink = NULL,
  verbose = TRUE
)
```

Arguments

`dat1`	SNPs, etc. from first dataset
`dat2`	SNPs, etc. from second dataset
`min_snps`	Number of minimum SNPs to check for analysis to continue (Optional)
`p1`	p1 for coloc (Optional)
`p2`	p2 for coloc (Optional)
`p12`	p12 for coloc (Optional)
`susie`	Run SuSiE? (Optional, boolean)
`bfile`	Path to Plink bed/bim/fam files (Optional; required for SuSiE)
`plink`	Path to Plink binary (Optional; required for SuSiE)
`verbose`	Display verbose information (Optional, boolean)

Value

Results data.frame

`.coloc_susie_sub` *Sub function to run SuSiE and coloc*

Description

Sub function to run SuSiE and coloc

Usage

```
.coloc_susie_sub(d1, d2, bfile = NULL, plink = NULL, verbose = TRUE, ...)
```

Arguments

`d1`	Dataset 1
`d2`	Dataset 2
`bfile`	Path to Plink bed/bim/fam files (Optional; required for SuSiE)
`plink`	Path to Plink binary (Optional; required for SuSiE)
`verbose`	Display verbose information (Optional, boolean)
`...`	Other arguments passed to coloc.susie and coloc.bf

Value

Results data.frame

`.dgidb_linkage` *Link ENSG IDs with DGIdb*

Description

Lookup ENSGs using the Drug Genome Interaction DB API.

Usage

```
.dgidb_linkage(ensgs)
```

Arguments

`ensgs`	Vector of ENSG IDs

Value

data.frame of results

`.ensg_to_name` *Convert ENSG IDs -> gene names*

Description

Attempts to convert ENSG IDs to gene names (hgnc_symbol). This is attempting using biomaRt's service and thus requires the optional biomaRt package to be installed.

Usage

```
.ensg_to_name(
  dat,
  ensg_col = "trait",
  new_col = "hgnc_symbol",
  build = "grch37"
)
```

Arguments

`dat`	Data.frame of data
`ensg_col`	Column name containing ENSG IDs (Optional)
`new_col`	Column to append to ‘dat‘ with converted names (Optional)
`build`	Genomic build (Optional)

Value

Data.frame with appended column for names

`.gwasvcf_to_coloc_rsid` *Prepare gwasvcf files for coloc. This method will extract SNPs from one file using one chrompos and then look up those SNPs in the other file – this is to ensure coloc can be conducted upon two datasets of different genomic builds without the need of liftover.*

Description

Prepare gwasvcf files for coloc. This method will extract SNPs from one file using one chrompos and then look up those SNPs in the other file – this is to ensure coloc can be conducted upon two datasets of different genomic builds without the need of liftover.

Usage

```
.gwasvcf_to_coloc_rsid(
  vcf1,
  vcf2,
  chrompos,
  type1 = NULL,
  type2 = NULL,
  build1 = "GRCh37",
  build2 = "GRCh37",
  verbose = TRUE
)
```

Arguments

`vcf1`	VCF object or path to vcf file
`vcf2`	VCF object or path to vcf file
`chrompos`	Character of the format chr:pos1-pos2

Value

list of coloc-ready data, or NA if failed

`.gwasvcf_to_pwcoco` *Prepare gwasvcf files for PWCoCo*

Description

Write files for PWCoCo where data are read from two VCF objects or files.

Usage

```
.gwasvcf_to_pwcoco(vcf1, vcf2, chrompos, type1 = NULL, type2 = NULL, outfile)
```

Arguments

`vcf1`	VCF object or path to VCF file
`vcf2`	VCF object or path to VCF file
`chrompos`	Character of the format chr:pos1-pos2
`type1`	How to treat vcffile1 for coloc, either "quant" or "cc" (Optional)
`type2`	How to treat vcffile2 for coloc, either "quant" or "cc" (Optional)
`outfile`	Path to output files, without file ending

Value

0 if success, 1 if there was a problem

`.ieugwasr_to_pwcoco` *Prepare ieugwasr data for PWCoCo*

Description

Write files for PWCoCo where data are read from the OpenGWAS DB.

Usage

```
.ieugwasr_to_pwcoco(id1, id2, chrompos, type1 = NULL, type2 = NULL, outfile)
```

Arguments

`id1`	ID for trait 1
`id2`	ID for trait 2
`chrompos`	Character of the format chr:pos1-pos2
`type1`	How to treat vcffile1 for coloc, either "quant" or "cc" (Optional)
`type2`	How to treat vcffile2 for coloc, either "quant" or "cc" (Optional)
`outfile`	Path to output files, without file ending

Value

0 if success, 1 if there was a problem

`.ivw_delta` *IVW weighted delta*

Description

Calculates the inverse variance weighted delta method from the MendelianRandomization package

Usage

```
.ivw_delta(dat)
```

Arguments

`object`	Harmonised data.frame

Value

Results data.frame

`.print_msg` *Helper function for message printing.*

Description

Helper function for message printing.

Usage

```
.print_msg(msg, verbose)
```

Arguments

`msg`	Message
`verbose`	Display message or suppress

`.pwcoco_sub` *Sub-function to run PWCoCo*

Description

Sub-function to run PWCoCo

Usage

```
.pwcoco_sub(
  bfile,
  chrpos,
  pwcoco,
  maf = 0.01,
  p1 = 1e-04,
  p2 = 1e-04,
  p12 = 1e-05,
  workdir = tempdir(),
  verbose = TRUE
)
```

Arguments

`bfile`	Path to Plink bed/bim/fam files
`chrpos`	Character of the format chr:pos1-pos2
`pwcoco`	Path to PWCoCo executible
`maf`	MAF cut-off (Optional)
`p1`	p1 for coloc (Optional)
`p2`	p2 for coloc (Optional)
`p12`	p12 for coloc (Optional)
`workdir`	Path to save temporary files (Optional)
`verbose`	Display verbose information (Optional, boolean)

Value

Results data.frame

`.read_dataset` *Read datasets*

Description

Helper function that is called from read_exposure and read_outcome. Extracts exposure and outcome data according to arguments. Should not be called directly.

Usage

```
.read_dataset(
  ids,
  rsids = NULL,
  pval = 5e-08,
  proxies = TRUE,
  proxy_rsq = 0.8,
  proxy_kb = 5000,
  proxy_nsnp = 5000,
  plink = NULL,
  bfile = NULL,
  clump_r2 = 0.01,
  clump_kb = 10000,
  pop = "EUR",
  type = "exposure",
  cores = 1,
  cores_proxy = 1,
  verbose = TRUE
)
```

Arguments

`ids`	List of OpenGWAS IDs or file paths (to gwasvcf files)
`rsids`	List of SNP rsIDs to extract
`pval`	Threshold to extract SNPs (Optional)
`proxies`	Whether to search for proxies (Optional, boolean)
`proxy_rsq`	R2 threshold to use when searching for proxies (Optional)
`proxy_kb`	kb threshold to use when searching for proxies (Optional)
`proxy_nsnp`	Number of SNPs when searching for proxies (Optional)
`plink`	Path to Plink binary (Optional)
`bfile`	Path to Plink .bed/.bim/.fam files (Optional)
`clump_r2`	r2 threshold for clumping SNPs (Optional)
`clump_kb`	Distance outside of which SNPs are considered in linkage equilibrium (Optional)
`pop`	Population (Optional, used only for clumping on OpenGWAS)

`type`	Type of data (Optional, "exposure" or "outcome")
`cores`	Number of cores for multi-threaded tasks (Optional) NB: Unavailable on Windows machines
`cores_proxy`	Number of cores for multi-threaded proxy searching (Optional) NB: Unavailable on Windows machines NB: Should not be more than ‘cores‘ argument!
`verbose`	Display verbose information (Optional, boolean)

Value

Data.frame of datasets

See Also

[read_exposure()], [read_outcome()]

`.wr_taylor_approx` *WR Taylor Approx of SE*

Description

Calculates the second term Taylor approximation for standard error of the Wald ratio method. From supplementary

Usage

```
.wr_taylor_approx(dat)
```

Arguments

`object`	Harmonised data.frame

Value

Results data.frame

`annotate_data` *Annotates the data using given IDs*

Description

Annotates the data using given IDs

Usage

```
annotate_data(dat, id1, id2)
```

Arguments

`dat`	Data.frame of data from vcf files or OpenGWAS DB
`id1`	DatasetsID class of exposure IDs
`id2`	DatasetsID class of outcome IDs

Value

Data.frame of annotated dat

`annotate_efo` *Annotate diseases with EFO IDs*

Description

Attempts to annotate disease names with EFO IDs using the ‘epigraphdb‘ package. Note that the matching is fuzzy and some disease names will have multiple associated EFOs which may differ in definition slightly.

Usage

```
annotate_efo(dat, column = "outcome")
```

Arguments

`dat`	Data.frame of data
`column`	Column name containing disease names

Value

Data.frame with appended column for EFO IDs

`annotate_ensg` *Annotate ENSG IDs with gene names*

Description

Attempts to convert ENSG IDs to gene names (hgnc_symbol). This is attempting using biomaRt's service and thus requires the optional biomaRt package to be installed.

Usage

```
annotate_ensg(
  dat,
  column = "exposure",
  gene_name_col = "hgnc_symbol",
  build = "grch37"
)
```

Arguments

`dat`	Data.frame of data
`gene_name_col`	Column to append to ‘dat‘ with converted names (Optional)
`build`	Genomic build (Optional)
`col`	Column name containing ENSG IDs (Optional)

Value

Data.frame with appended column for names

`calc_f_stat` *Calculate F-statistic*

Description

Calculates portion of variance explained and F-statistic. If the data is lacking key information, i.e. allele frequencies, sample size or consists of only one SNP, then the approximate F-statistic will be used instead: $F = b**2/SE**2$.

Usage

```
calc_f_stat(dat, f_cutoff = 10, force_approx = FALSE, verbose = TRUE)
```

Arguments

`dat`	Data.frame from do_mr()
`f_cutoff`	F-statistic cutoff (Optional)
`force_approx`	Force to use the approximate F-statistic instead (Optional, boolean)
`verbose`	Display verbose information (Optional, boolean)

Value

Modified ‘dat‘ data.frame (if f_cutoff > 0 supplied)

See Also

[do_mr()]

`check_snps` *Check if SNPs are good for use in analyses and mark them as such.*

Description

Check if SNPs are good for use in analyses and mark them as such.

Usage

```
check_snps(dat, analyses = c("mr", "coloc"), drop = T)
```

Arguments

`dat`	A data.frame of formatted data (exposure or outcome)
`analyses`	Which analyses should be checked?
`drop`	Whether to drop SNPs if they failed the check

Details

List of analyses and what data are checked for:

- "MR"beta, SE, P value
- "coloc"chromosome, position, P value

Value

Data.frame

`cis_trans` *Annotate cis/trans SNPs*

Description

Attempts to annotate SNPs as cis or trans depending on their location to the gene coding region. This is achieved using the ‘biomaRt‘ R package.

Usage

```
cis_trans(
  dat,
  cis_region = 5e+05,
  chr_col = "chr.exposure",
  pos_col = "position.exposure",
  snp_col = NULL,
  values_col = "exposure",
  filter = "ensembl_gene_id",
  missing = "include",
  build = "grch37"
)
```

Arguments

`dat`	Data.frame of data
`cis_region`	Cis region definition (Optional, in kb)
`chr_col`	Column name for chromosome (Optional)
`pos_col`	Column name for position (Optional)
`snp_col`	Column name for SNP rsIDs (Optional)
`values_col`	Column name for gene names or ENSG IDs (Optional) NB: Choice must match the ‘filter‘ value
`filter`	How to search for genes in biomaRt, either: 1. "ensembl_gene_id" for ENSG IDs, or 2. "hgnc_symbol" for gene names (Optional)
`missing`	"Include" or "drop" SNPs which could not be matched (Optional)
`build`	Genomic build (Optional)

Value

Data.frame with appended column for cis/trans status

combine_results *Combine MR and coloc results into one data.frame*

Description

Combine MR and coloc results into one data.frame

Usage

```
combine_results(
  mr_res,
  coloc_res,
  mr_res.by = c("id.exposure", "id.outcome"),
  coloc_res.by = c("file.exposure", "file.outcome")
)
```

Arguments

`mr_res`	A data.frame of MR results from ‘do_mr()‘
`coloc_res`	A data.frame of coloc results from ‘do_coloc()‘
`mr_res.by`	MR columns to use for merging
`coloc_res.by`	Coloc columns to use for merging

Value

Data.frame of merged results

dat_to_gwasvcf *Convert data.frame to gwasvcf format.*

Description

Function to convert a data.frame to gwasvcf format.

Usage

```
dat_to_gwasvcf(
  dat,
  out,
  chr_col,
  pos_col,
  nea_col,
  ea_col,
  snp_col = NULL,
  eaf_col = NULL,
```

```
  beta_col = NULL,
  se_col = NULL,
  pval_col = NULL,
  n = NULL,
  n_case = NULL,
  name = NULL,
  bcf_tools = NULL,
  verbose = TRUE
)
```

Arguments

`dat`	Data.frame
`out`	Path to save output
`chr_col`	Column name for chromosome
`pos_col`	Column name for position
`nea_col`	Column name for non-effect allele
`ea_col`	Column name for effect allele
`snp_col`	Column name for SNP (Optional)
`eaf_col`	Column name for effect allele frequency (Optional)
`beta_col`	Column name for beta (Optional)
`se_col`	Column name for standard error (Optional)
`pval_col`	Column name for P value (Optional) NB: P values will be saved as 10^-P
`n`	Sample size (Optional), can be int or column name
`n_case`	Number of cases (Optional), can be int or column name
`name`	Trait name (Optional), can be string or column name
`bcf_tools`	Path to bcf_tools (Optional)
`verbose`	Display verbose information (Optional, boolean)

Value

gwasvcf object

See Also

[file_to_gwasvcf()] for converting files.

`do_coloc` *Run colocalisation analyses*

Description

Runs colocalisation using any of the following methods:

1. Coloc.abf, see coloc::coloc.abf()
2. Coloc.susie, see coloc::coloc.susie()
3. PWCoCo, see PWCoCo

NB: PWCoCo is not available on Windows.

Usage

```
do_coloc(
  dat,
  cdat = NA,
  method = "coloc.abf",
  coloc_window = 5e+05,
  plot_region = F,
  bfile = NULL,
  plink = NULL,
  pwcoco = NULL,
  workdir = tempdir(),
  cores = 1,
  verbose = TRUE
)
```

Arguments

`dat`	A data.frame of harmonised data
`cdat`	A named list of regional data but can be omitted (Optional)
`method`	Which method of colocalisation to use: coloc.abf, coloc.susie, pwcoco (Optional)
`coloc_window`	Size (+/-) of region to extract for colocalisation analyses (Optional)
`plot_region`	Whether to plot the regions or not
`bfile`	Path to Plink bed/bim/fam files (Optional)
`plink`	Path to Plink binary (Optional)
`pwcoco`	If PWCoCo is the selected coloc method, path to PWCoCo binary (Optional)
`workdir`	Path to save temporary files (Optional)
`cores`	Number of cores for multi-threaded tasks (Optional) NB: Unavailable on Windows machines
`verbose`	Display verbose information (Optional, boolean)

Value

A data.frame of colocalistion results

See Also

[coloc::coloc.abf()], [coloc::coloc.susie()]

`do_mr` *Run Mendelian randomisation analyses*

Description

Runs Mendelian randomisation and related analyses:

1. Wald ratio, see .wr_taylor_approx()
2. Inverse variance weighted, see .ivw.delta()
3. Steiger filtering, see TwoSampleMR::directionality_test()

Usage

```
do_mr(dat, f_cutoff = 10, all_wr = TRUE, verbose = TRUE)
```

Arguments

`dat`	A data.frame of harmonised data
`f_cutoff`	Define an F-statistic cutoff (Optional)
`all_wr`	Should the Wald ratio be calculated for all SNPs, even if IVW can be used? (Optional)
`verbose`	Display verbose information (Optional, boolean)

Value

A data.frame of MR results

See Also

[.wr_taylor_approx()], [.ivw_delta()], [TwoSampleMR::directionality_test()]

`drug_target_evidence` *Generate drug target evidence*

Description

Uses the Drug Genome Interaction DB's API to search for drug target-related evidence, including on: Druggable Genome, Clinically Actionable and Drug Resistant ontologies.

Usage

```
drug_target_evidence(dat, ensg_col = "exposure")
```

Arguments

`dat`	A data.frame or named list
`ensg_col`	Column, or name, to be accessed in ‘dat‘

Details

The lookup MUST be ENSG IDs.

Value

data.frame of results

`extract_matched_regions` *Extract SNPs based on region for colocalisation analyses. Can be used before calling the ‘do_coloc‘ function or will be called as part of that function automatically.*

Description

Extract SNPs based on region for colocalisation analyses. Can be used before calling the ‘do_coloc‘ function or will be called as part of that function automatically.

Usage

```
extract_matched_regions(dat, window = 5e+05, cores = 1, verbose = TRUE)
```

Arguments

`dat`	A data.frame of harmonised data
`window`	Window around SNPs to extract (Optional)
`cores`	Number of cores for multi-threaded tasks (Optional) NB: Unavailable on Windows machines
`verbose`	Display verbose information (Optional, boolean)

Value

Named list of matched regional data

See Also

[mrpipeline::do_coloc()]

`file_to_gwasvcf` *Convert file(s) to gwasvcf format.*

Description

Function to convert a file (or files) to gwasvcf format.

Usage

```
file_to_gwasvcf(
  file,
  chr_col,
  pos_col,
  nea_col,
  ea_col,
  snp_col = NULL,
  eaf_col = NULL,
  beta_col = NULL,
  se_col = NULL,
  pval_col = NULL,
  n = NULL,
  n_case = NULL,
  name = NULL,
  header = TRUE,
  sep = "\t",
  cores = 1,
  bcf_tools = NULL,
  verbose = TRUE
)
```

Arguments

`file`	Path to file
`chr_col`	Column name for chromosome
`pos_col`	Column name for position
`nea_col`	Column name for non-effect allele
`ea_col`	Column name for effect allele
`snp_col`	Column name for SNP (Optional)

`eaf_col`	Column name for effect allele frequency (Optional)
`beta_col`	Column name for beta (Optional)
`se_col`	Column name for standard error (Optional)
`pval_col`	Column name for P value (Optional) NB: P values will be saved as 10^-P
`n`	Sample size (Optional), can be int or column name
`n_case`	Number of cases (Optional), can be int or column name
`name`	Trait name (Optional), can be string or column name
`header`	Whether file has a header or not (Optional, boolean)
`sep`	File separater (Optional)
`cores`	Number of cores for multi-threaded tasks (Optional) NB: Unavailable on Windows machines
`bcf_tools`	Path to bcf_tools (Optional)
`verbose`	Display verbose information (Optional, boolean)

Value

gwasvcf object(s)

See Also

[dat_to_gwasvcf()] for converting data.frames

`forest_plot` *Forest plot*

Description

Creates a forest plot of MR estimates from do_mr(). Will plot both the Wald ratios for all SNPs which form the instrument and inverse variance weighted method. However, if you wish for only the "discovery" results to be plotted (i.e. WR for single-SNP instruments and only IVW for multi-SNP instruments), then setting ‘plot_all_res‘ to `FALSE` will achieve this. If the plot is too crowded, subsetting the results before passing them to this plotter will help.

Usage

```
forest_plot(
  res,
  snp_col = "snp",
  beta_col = "b",
  se_col = "se",
  pval_col = NULL,
  or_col = "or",
  or_lci_col = "or_lci95",
  or_uci_col = "or_uci95",
```

```
  method_col = "method",
  exposure_col = "exposure",
  outcome_col = "outcome",
  plot_all_res = TRUE
)
```

Arguments

`res`	A data.frame of MR results
`snp_col`	Column name for SNPs (Optional)
`beta_col`	Column name for beta (Optional)
`se_col`	Column name for standard error (Optional)
`pval_col`	Column name for P value (Optional, unused for now)
`or_col`	Column name for odds ratio (Optional)
`or_lci_col`	Column name for lower CI of OR (Optional)
`or_uci_col`	Column name for upper CI of OR (Optional)
`method_col`	Column name which contains the MR method (Optional)
`exposure_col`	Column name for exposure names (Optional)
`outcome_col`	Column name for outcome names (Optional)
`plot_all_res`	For multi-SNP instruments, also plot the Wald ratios for all SNPs (`TRUE`) or just the inverse variance weighted result (`FALSE`).

Value

Plot

See Also

[do_mr()]

`get_col_name` *Get column names from agnostic but formatted data.frame*

Description

Get column names from agnostic but formatted data.frame

Usage

```
get_col_name(df, data)
```

Arguments

`df`	Data.frame of formatted data (exposure or outcome)
`data`	Name of column to find

Value

Name of column formatted for given data.frame

`harmonise` *Harmonise exposure and outcomes.*

Description

Wrapper function for harmonise_data function in the ‘TwoSampleMR‘ package.

Usage

```
harmonise(exposure, outcome, action = 1, cores = 1, verbose = TRUE)
```

Arguments

`exposure`	Data.frame of exposure dataset(s)
`outcome`	Data.frame of outcome dataset(s)
`action`	How to harmonise alleles; see harmonise_data.
`cores`	Number of cores for multi-threaded tasks (Optional) NB: Unavailable on Windows machines
`verbose`	Display verbose information (Optional, boolean)

Value

Harmonised data.frame

See Also

[TwoSampleMR::harmonise_data()]

`hello` *Hello, World!*

Description

Prints 'Hello, world!'.

Usage

```
hello()
```

Examples

```
hello()
```

`is_ambiguous` *Check if SNP frequencies are ambiguous*

Description

Check if SNP frequencies are ambiguous

Usage

```
is_ambiguous(freq, tol = 0.08)
```

Arguments

`freq`	Frequency
`tol`	Tolerance around 0.5 (Optional)

Value

True/False if ambiguous

`is_palindromic` *Check if SNP is palindromic*

Description

Check if SNP is palindromic

Usage

```
is_palindromic(A1, A2)
```

Arguments

`A1`	Allele 1
`A2`	Allele 2

Value

True/False if palindromic

`mr_pipeline` *Entry point for the pipeline.*

Description

Entry point for the pipeline.

Usage

```
mr_pipeline(ids1, ids2, out_name = "", config_file = "", ...)
```

Arguments

`ids1`	IDs or filenames for summary statistics
`ids2`	IDs or filenames for summary statistics
`out_name`	Name of the analysis given to the markdown report, defaults to time and date
`config_file`	Path to config.yml file. Defaults to file that comes with the package. Please see that file for more details.
`...`	Other arguments for plotting, NOT YET IMPLEMENTED

Value

list of results for debugging

`pairwise_analysis` *Performs pairwise harmonisation and analyses – helpful when analysing many exposure-outcome pairs as performing the standard "linear" approach will be very slow.*

Description

In this function, an exposure-outcome pair are harmonised, analyses are ran on those data and those results are saved to a file. Analyses ran can be MR or colocalisation, as desired.

Usage

```
pairwise_analysis(
  exposure,
  outcome,
  res_path,
  ...,
  do_coloc = FALSE,
  cores = 1,
  verbose = TRUE
)
```

Arguments

`exposure`	Data.frame of exposure dataset(s)
`outcome`	Data.frame of outcome dataset(s)
`res_path`	Path to save result files
`...`	Other arguments for the following functions: harmonise do_mr do_coloc
`do_coloc`	True/False run colocalisation analyses
`cores`	Number of cores for multi-threaded tasks (Optional) NB: Unavailable on Windows machines
`verbose`	Display verbose information (Optional, boolean)

`read_exposure` *Read exposures*

Description

Read a dataset (or datasets) as exposure. Can accept both OpenGWAS IDs or file paths. Accepts only gwasvcf file formats. This function can clump data locally, if supplied with the ‘plink‘ and ‘bfile‘ arguments. If these are not supplied, clumping will take place on the OpenGWAS servers *only* for OpenGWAS IDs. If you would like to clump local files, please provide paths to Plink and bfiles.

Usage

```
read_exposure(
  ids,
  pval = 5e-08,
  plink = NULL,
  bfile = NULL,
  clump_r2 = 0.01,
  clump_kb = 10000,
  pop = "EUR",
  cores = 1,
  verbose = TRUE
)
```

Arguments

`ids`	List of OpenGWAS IDs or file paths (to gwasvcf files)
`pval`	Threshold to extract SNPs (Optional)
`plink`	Path to Plink binary (Optional)
`bfile`	Path to Plink .bed/.bim/.fam files (Optional)
`clump_r2`	r2 threshold for clumping SNPs (Optional)
`clump_kb`	Distance outside of which SNPs are considered in linkage equilibrium (Optional)

`pop`	Population (Optional, used only for clumping on OpenGWAS)
`cores`	Number of cores for multi-threaded tasks (Optional) NB: Unavailable on Windows machines
`verbose`	Display verbose information (Optional, boolean)

Value

Data.frame of exposure datasets

`read_outcome` *Read outcomes*

Description

Read a dataset (or datasets) as outcome. Can accept both OpenGWAS IDs or file paths. Accepts only gwasvcf file formats. This function can search for proxy SNPs locally, if supplied with the ‘plink‘ and ‘bfile‘ arguments. If these are not supplied, proxy searching will take place on the OpenGWAS servers *only* for OpenGWAS IDs. If you would like to search for proxies locally, please provide paths to Plink and bfiles.

Usage

```
read_outcome(
  ids,
  rsids,
  proxies = TRUE,
  proxy_rsq = 0.8,
  proxy_kb = 5000,
  proxy_nsnp = 5000,
  plink = NULL,
  bfile = NULL,
  cores = 1,
  cores_proxy = 1,
  verbose = TRUE
)
```

Arguments

`ids`	List of OpenGWAS IDs or file paths (to gwasvcf files)
`rsids`	List of SNP rsIDs to extract
`proxies`	Whether to search for proxies (Optional, boolean)
`proxy_rsq`	R2 threshold to use when searching for proxies (Optional)
`proxy_kb`	kb threshold to use when searching for proxies (Optional)
`proxy_nsnp`	Number of SNPs when searching for proxies (Optional)
`plink`	Path to Plink binary (Optional)

`bfile`	Path to Plink .bed/.bim/.fam files (Optional)
`cores`	Number of cores for multi-threaded tasks (Optional) NB: Unavailable on Windows machines
`cores_proxy`	Number of cores for multi-threaded proxy searching (Optional) NB: Unavailable on Windows machines NB: Should not be more than ‘cores‘ argument!
`verbose`	Display verbose information (Optional, boolean)

Value

Data.frame of outcome datasets

`regional_plot` *Plots a regional plot of the area being tested for colocalisation*

Description

Plots a regional plot of the area being tested for colocalisation

Usage

```
regional_plot(
  dat,
  exposure,
  outcome,
  bfile = NULL,
  plink = NULL,
  verbose = TRUE
)
```

Arguments

`dat`	A data.frame of harmonised data
`exposure`	Character, name of exposure
`outcome`	Character, name of outcome
`bfile`	Path to Plink bed/bim/fam files
`plink`	Path to Plink binary
`verbose`	Print messages or not

`validate_config` *Validates parameters in config file*

Description

Validates parameters in config file

Usage

```
validate_config(conf)
```

Arguments

`conf`	config::config file of parameters

Value

Validated config class

`volcano_plot` *Volcano plot*

Description

Creates a volcano plot of Wald ratios from [do_mr()]. This function will take all of the Wald ratios in the given data.frame and plot these. If the plot is too crowded, subsetting the results before passing them to this plotter will help. If ‘plotly‘ is installed, the plot will be returned interactive.

Usage

```
volcano_plot(
  res,
  label = "outcome",
  snp_col = "snp",
  beta_col = "b",
  pval_col = "pval",
  or_col = "or",
  or_lci_col = "or_lci95",
  or_uci_col = "or_uci95",
  method_col = "method",
  force_static = FALSE
)
```

Arguments

`res`	A data.frame of MR results
`label`	Column whose values will be used to group results by (Optional)
`snp_col`	Column name for SNPs (Optional)
`beta_col`	Column name for beta (Optional)
`pval_col`	Column name for P value (Optional)
`or_col`	Column name for odds ratio (Optional)
`or_lci_col`	Column name for lower CI of OR (Optional)
`or_uci_col`	Column name for upper CI of OR (Optional)
`method_col`	Column name which contains the MR method (Optional)
`force_static`	True for forcing the plot to be returned as a static plot (Optional)

Value

Plot

`z_comparison_plot` *Z score comparison plot*

Description

If ‘plotly‘ is installed, the plot will be returned interactive.

Usage

```
z_comparison_plot(dat1, dat2, z_col = "z", p_col = "pvalues", verbose = TRUE)
```

Arguments

`dat1`	A list of data
`dat2`	A list of data
`z_col`	Column name for Z scores (Optional)
`verbose`	Display verbose information (Optional, boolean)
`force_static`	True for forcing the plot to be returned as a static plot (Optional)

Value

Plot

`z_plot` *Z plot*

Description

Creates a Z-score plot for SNPs, where $Z = b/SE$. This should follow a parabollic shape and so can be used to find certain SNPs which may not follow this shape. If ‘plotly‘ is installed, the plot will be returned interactive.

Usage

```
z_plot(
  dat,
  snp_col = "SNP",
  beta_col = "beta.exposure",
  se_col = "se.exposure",
  pval_col = "pval.exposure",
  force_static = FALSE
)
```

Arguments

`dat`	A data.frame of data
`snp_col`	Column of SNP names (Optional)
`beta_col`	Column of MR beta estimates (Optional)
`se_col`	Column of standard errors for the beta estimates (Optional)
`pval_col`	Column of P values (Optional)
`force_static`	True for forcing the plot to be returned as a static plot (Optional)

Value

Plot

Index