# Package: genepi.utils (via r-universe)

September 17, 2024

**Title** GenEpi Utility Functions

**Version** 0.0.31

**Description** The genepi.utils package is a collection of utility functions for working with genetic epidemiology data.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**Imports** data.table, fst, furrr, future, MendelianRandomization, TwoSampleMR, indexevent, SlopeHunter, ggplot2, ggpubr, ggrepel, gridExtra, progressr, stats, utils, yaml, viridis, tibble, GenomicRanges, IRanges, rtracklayer, S7

**Remotes** mrcieu/TwoSampleMR, Osmahmoud/SlopeHunter, DudbridgeLab/indexevent

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0),

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Collate** 'chrpos_to_rsid.R' 'class_colmap.R' 'plink.R' 'class_gwas.R' 'class_mr.R' 'clump.R' 'collider_bias.R' 'collider_bias_old.R' 'colocalisation.R' 'drug_target_proxy.R' 'eaf_plot.R' 'genepi.utils-package.R' 'harmonise.R' 'ld_matrix.R' 'lift.R' 'manhattan.R' 'pz_plot.R' 'qq_plot.R' 'standardise_gwas.R' 'utils.R' 'zzz.R'

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

**RemoteUrl** https://github.com/nicksunderland/genepi.utils

**RemoteRef** HEAD

**RemoteSha** 2d97072d80f962c2f3f4493e96d89c3b1503e8ef

# Contents

## as.data.table — *as.data.table*

### Description

as.data.table

### Usage

```
as.data.table(object, ...)
```

### Arguments

| | |
|---|---|
| object | GWAS object to covert to data.table |
| ... | argument for data.table generic, ignored in this implementation |

## chrpos_to_rsid — *Chromosome & position data to variant RSID*

### Description

Chromosome & position data to variant RSID

### Usage

```
chrpos_to_rsid(
  dt,
  chr_col,
  pos_col,
  ea_col = NULL,
  nea_col = NULL,
  flip = "allow",
  alt_rsids = FALSE,
  build = "b37_dbsnp156",
  dbsnp_dir = genepi.utils::which_dbsnp_directory(),
  parallel_cores = parallel::detectCores(),
  verbose = TRUE
)
```

### Arguments

| | |
|---|---|
| dt | a data.frame like object, or file path, with at least columns (chrom, pos, ea, nea) |
| chr_col | a string column name; chromosome position |
| pos_col | a string column name; base position |

| | |
|---|---|
| `ea_col` | a string column name; effect allele |
| `nea_col` | a string column name; non effect allele |
| `flip` | a string, options: "report", "allow", "no_flip" |
| `alt_rsids` | a logical, whether to return additional alternate RSIDs |
| `build` | a string, options: "b37_dbsnp156", "b38_dbsnp156" (corresponds to the appropriate data directory) |
| `dbsnp_dir` | a string file path to the dbSNP .fst file directory - see setup documentation |
| `parallel_cores` | an integer, the number of cores/workers to set up the `future::multisession` with |
| `verbose` | a logical, runtime reporting |

## Value

a data.table with an RSID column (or a list: 1-data.table; 2-list of alternate rsids IDs)

---

`clump` *Clump a GWAS*

---

## Description

Clump variants in a GWAS using PLINK2 and an appropriate reference panel. For example, the 1000 genomes phase 3 data can be downloaded from the PLINK website (https://www.cog-genomics.org/plink/2.0/resources#phase3_1kg). To remove duplicates you can run:

plink2
–pfile all_phase3
–rm-dup force-first
–make-pgen
–out all_phase3_nodup

The path to the reference (without the plink extensions) should be passed as the `plink_ref` argument. The path to the plink2 executable should be passed as the `plink2` argument.

## Usage

```
clump(
  gwas,
  p1 = 1,
  p2 = 1,
  r2 = 0.1,
  kb = 250,
  plink2 = genepi.utils::which_plink2(),
  plink_ref = genepi.utils::which_1000G_reference(build = "GRCh37"),
  logging = TRUE,
  parallel_cores = parallel::detectCores()
)
```

### Arguments

| | |
|---|---|
| gwas | a data.frame like object with at least columns rsid, ea, oa, and p |
| p1 | a numeric, the p-value threshold for inclusion as a clump |
| p2 | a numeric, the p-value threshold for incorporation into a clump |
| r2 | a numeric, the r2 value |
| kb | a integer, the window for clumping |
| plink2 | a string, path to the plink executable |
| plink_ref | a string, path to the pfile genome reference |
| logging | a logical, whether to set the plink logging information as attributes (log, missing_id, missing_allele) on the returned data.table |
| parallel_cores | an integer, how many cores / threads to use |

### Value

a data.table with additional columns index (logical, whether the variant is an index SNP) and clump (integer, the clump the variant belongs to)

---

## clump_mr *Clump MR object exposure*

---

### Description

Clump MR object exposure

### Usage

```
clump_mr(
  x,
  p1 = 1,
  p2 = 1,
  r2 = 0.001,
  kb = 250,
  plink2 = genepi.utils::which_plink2(),
  plink_ref = genepi.utils::which_1000G_reference(build = "GRCh37"),
  parallel_cores = parallel::detectCores()
)
```

### Arguments

| | |
|---|---|
| x | an object of class MR description |
| p1 | a numeric, the p-value threshold for inclusion as a clump |
| p2 | a numeric, the p-value threshold for incorporation into a clump |
| r2 | a numeric, the r2 value |

| | |
|---|---|
| kb | a integer, the window for clumping |
| plink2 | a string, path to the plink executable |
| plink_ref | a string, path to the pfile genome reference |
| parallel_cores | an integer, how many cores / threads to use |

---

`collider_bias` *Run collider bias assessment*

---

## Description

Run collider bias assessment

## Usage

```
collider_bias(
  x,
  bias_method = "dudbridge",
  r2 = 0.001,
  p1 = 5e-08,
  kb = 250,
  plink2 = genepi.utils::which_plink2(),
  plink_ref = genepi.utils::which_1000G_reference(build = "GRCh37"),
  ip = 0.001,
  pi0 = 0.6,
  sxy1 = 1e-05,
  bootstraps = 100,
  weighted = TRUE,
  method = "Simex",
  B = 1000,
  seed = 2023
)
```

## Arguments

| | |
|---|---|
| x | an object of class MR |
| bias_method | a character or character vector, one or more of c("dudbridge", "slope-hunter", "mr_ivw", "mr_egger", "mr_weighted_median", "mr_weighted_mode") |
| r2 | a numeric 0-1, r2 used for clumping - set all clumping params to NA to turn off |
| p1 | a numeric 0-1, p1 used for clumping - set all clumping params to NA to turn off |
| kb | an integer, kb used for clumping - set all clumping params to NA to turn off |
| plink2 | a path, the plink2 binary |

| | |
|---|---|
| `plink_ref` | a path, the reference genome pfile |
| `ip` | a numeric 0-1, threshold for removing incidence variants; see `xp_thresh` SlopeHunter::hunt() |
| `pi0` | a numeric 0-1, proportion of SNPs in the incidence only cluster; see `init_pi` SlopeHunter::hunt() |
| `sxy1` | a numeric, the covariance between incidence and progression Gip SNPs; see `init_sigmaIP` SlopeHunter::hunt() |
| `bootstraps` | an integer, number of bootstraps to estimate SE; see `M` SlopeHunter::hunt() |
| `weighted` | see `weighted` indexevent::indexevent() |
| `method` | see `method` indexevent::indexevent() |
| `B` | see `B` indexevent::indexevent() |
| `seed` | seed, for reproducibility |

---

`Column` *Column object*

---

## Description

Column object

## Usage

```
Column(name = class_missing, alias = class_missing, type = class_missing)
```

## Arguments

| | |
|---|---|
| `name` | the standard column name |
| `alias` | a character vector of aliases (other column names) for this column |
| `type` | a character, an atomic R type |

## Value

an S7 class genepi.utils::Column object

## Slots

`name` the standard column name

`alias` a character vector of aliases (other column names) for this column

`type` a character, an atomic R type

## ColumnMap *ColumnMap object*

### Description

A mapping to the standardised column names used in this package. Available names: 'rsid', 'chr', 'bp', 'ea', 'oa', 'eaf', 'p', 'beta', 'se', 'or', 'or_se', 'or_lb', 'or_ub', 'beta_lb', 'beta_ub', 'z', 'q_stat', 'i2', 'nstudies', 'n'

### Usage

```
ColumnMap(x)
```

### Arguments

| | |
|---|---|
| x | either a list of `Column` class objects, a valid string for a pre-defined map: default, metal, ieu_ukb, ieugwasr, ns_map, gwama, giant, or a named character vector or list (standard name = old name) |

### Value

an S7 class genepi.utils::ColumnMap object

### Slots

`map` a list of `Column` class objects

## cwls *Corrected Weighted Least Squares collider bias method*

### Description

Corrected Weighted Least Squares collider bias method

### Usage

```
cwls(x, ...)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |
| ... | parameter sink, additional ignored parameters |

### Value

an object of class MRResult

---

`dudbridge` *Dudbridge collider bias method*

---

## Description

Dudbridge collider bias method

## Usage

```
dudbridge(
  x,
  weighted = TRUE,
  prune = NULL,
  method = "Simex",
  B = 1000,
  lambda = seq(0.25, 5, 0.25),
  seed = 2018,
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | an object of class MR |
| `weighted` | see indexevent::indexevent() |
| `prune` | see indexevent::indexevent() |
| `method` | see indexevent::indexevent() |
| `B` | see indexevent::indexevent() |
| `lambda` | see indexevent::indexevent() |
| `seed` | see indexevent::indexevent() |
| `...` | parameter sink, additional ignored parameters |

## Value

an object of class MRResult

---

`eaf_plot` *Effect allele frequency plot*

---

## Description

Plotting reported effect allele frequencies (EAF) against a reference set to identify study variants which significantly deviate from the expected population frequencies.

### Usage

```
eaf_plot(
  gwas,
  eaf_col = "EAF",
  ref_eaf_col = "EUR_EAF",
  tolerance = 0.2,
  colours = list(missing = "#5B1A18", outlier = "#FD6467", within = "#7294D4"),
  title = NULL,
  facet_grid_row_col = NULL,
  facet_grid_col_col = NULL
)
```

### Arguments

| | |
|---|---|
| gwas | a data.table |
| eaf_col | a string, the column containing the study EAF data |
| ref_eaf_col | a string, the column containing the reference EAF data |
| tolerance | a numeric, frequency difference that determines outliers |
| colours | a 3 element list of colour codes, e.g. list(missing="#5B1A18", outlier="#FD6467", within="#7294D4") |
| title | a string, the plot title |
| facet_grid_row_col | (optional), a column by which to facet the plot by rows |
| facet_grid_col_col | (optional), a column by which to facet the plot by columns |

### Value

a ggplot

---

## generate_random_gwas_data

*Generate random GWAS data*

---

### Description

Generates rows of synthetic GWAS summary stats data. Useful for developing plotting and other methods. No attempt is made to make this data at all realistic.

### Usage

```
generate_random_gwas_data(n, seed = 2023)
```

### Arguments

| | |
|---|---|
| n | number of fake variants to generate |
| seed | seed, for reproducibility |

### Value

a data.table with columns SNP, CHR, BP, OA, EA, EAF, BETA, P, EUR_EAF

---

`get_pfile_variants` *Extract variants from plink binary*

---

### Description

Extract variants from plink binary

### Usage

```
get_pfile_variants(
  snp,
  win_kb,
  chr,
  from_bp,
  to_bp,
  plink2 = genepi.utils::which_plink2(),
  pfile = genepi.utils::which_1000G_reference(build = "GRCh37")
)
```

### Arguments

| | |
|---|---|
| `snp` | character, an rsid |
| `win_kb` | numeric, window size around snp in kb |
| `chr` | character, the chromosome (use instead of snp and win_kb, not in addition) |
| `from_bp` | numeric, the start base position (use instead of snp and win_kb, not in addition) |
| `to_bp` | numeric, the end base position (use instead of snp and win_kb, not in addition) |
| `plink2` | character / path, the plink2 executable |
| `pfile` | character / path, the plink pfile set |

### Value

a data.table

`get_proxies` *Get proxies for variants from plink binary*

## Description

Get proxies for variants from plink binary

## Usage

```
get_proxies(
  x,
  stat = "r2-unphased",
  win_kb = 125,
  win_r2 = 0.8,
  win_ninter = Inf,
  proxy_eaf = NULL,
  plink2 = genepi.utils::which_plink2(),
  pfile = genepi.utils::which_1000G_reference(build = "GRCh37"),
  ...
)
```

## Arguments

| | |
|---|---|
| `x` | a character vector of rsids or a GWAS object |
| `stat` | character, the R stat to calculate, one of "r2-unphased", "r2-phased", "r-unphased", "r-phased" |
| `win_kb` | numeric, the window to look in around the variants |
| `win_r2` | numeric, the lower r2 limit to include in output, (for –r-phased and –r-unphased, this means \|r\| sqrt(0.2)) |
| `win_ninter` | numeric, controls the maximum number of other variants allowed between variant-pairs in the report. Inf = off. |
| `proxy_eaf` | numeric, the minimal effect allele frequency for proxy variants. NULL = eaf filtering off. |
| `plink2` | character / path, the plink2 executable |
| `pfile` | character / path, the plink pfile set |
| `...` | other arguments (see below) |
| `snps` | a character vector (available if `x` is a `GWAS` object), a vector of rsids to ensure exist, or else try and find proxies for |
| `then` | a string (available if `x` is a `GWAS` object), either `add` (adds proxies to current GWAS) or `subset` (subsets GWAS to variants and potential proxies for variants in `x`) |

## Value

a data.table of variants and their proxies (if `x` is a `character` vector) or a `GWAS` object if `x` is a `GWAS` object.

GWAS *GWAS object*

## Description

A GWAS object is a container for vectors of GWAS data, a correlation matrix, and metadata regarding quality control procedures applied at the point of object creation / data import.

## Usage

```
GWAS(
  dat,
  map = "default",
  drop = FALSE,
  fill = FALSE,
  fill_rsid = FALSE,
  missing_rsid = "fill_CHR:BP",
  parallel_cores = parallel::detectCores(),
  dbsnp_dir = genepi.utils::which_dbsnp_directory(),
  filters = list(beta_invalid = "!is.infinite(beta) & abs(beta) < 20", eaf_invalid =
    "eaf > 0 & eaf < 1", p_invalid = "!is.infinite(p)", se_invalid = "!is.infinite(se)",
    alleles_invalid = "!is.na(ea) & !is.na(oa)", chr_missing = "!is.na(chr)", bp_missing
    = "!is.na(bp)", beta_missing = "!is.na(beta)", se_missing = "!is.na(se)", p_missing =
    "!is.na(p)", eaf_missing = "!is.na(eaf)"),
  reference = NULL,
  ref_map = NULL,
  verbose = TRUE,
  ...
)
```

## Arguments

| | |
|---|---|
| `dat` | a valid string file path to be read by `data.table::fread` or a `data.table::data.table` object; the GWAS data source |
| `map` | a valid input to the `ColumnMap` class constructor (a predefined map string id, a named list or character vector, or a ColumMap object) |
| `drop` | a logical, whether to drop data source columns not in the column `map` |
| `fill` | a logical, whether to add (NAs) missing columns present in the column `map` but not present in the data source |
| `fill_rsid` | either FALSE or a valid argument for the chrpos_to_rsid `build` argument, e.g. "b37_dbsnp156" |
| `missing_rsid` | a string, how to handle missing rsids: one of "fill_CHR:BP", "fill_CHR:BP_OA_EA", "overwrite_CHR:BP", "overwrite_CHR:BP:OA:EA", "none", or "leave" |

`parallel_cores`
: an integer, number of cores to used for RSID mapping, default is maximum machine cores

`dbsnp_dir`
: path to the dbsnp directory of fst files see chrpos_to_rsid `dbsnp_dir` argument

`filters`
: a list of named strings, each to be evaluated as an expression to filter the data during the quality control steps (above)

`reference`
: a valid string file path to be read by `data.table::fread` or a `data.table::data.table` object; the reference data

`ref_map`
: a valid input to the `ColumnMap` class constructor (a predefined map id (a string), a named list or character vector, or a ColumMap object) defining at least columns `rsid` (or `chr`, `bp`), `ea`, `oa` and `eaf`.

`verbose`
: a logical, whether to print details

`...`
: variable capture to be passed to the constructor, e.g. individual vectors for the slots, rather that `dat`

## Value

an S7 class genepi.utils::GWAS object

## Slots

`rsid` character, variant ID - usually in rs12345 format, however this can be changed with the `missing_rsid` argument

`chr` character, chromosome identifier

`bp` integer, base position

`ea` character, effect allele

`oa` character, other allele

`eaf` numeric, effect allele frequency

`beta` numeric, effect size

`se` numeric, effect size standard error

`p` numeric, p-value

`n` integer, total number of samples

`ncase` integer, number of cases

`strand` character, the strand + or -

`imputed` logical, whether imputed

`info` numeric, the info score

`q` numeric, the Q statistic for meta analysis results

`q_p` numeric, the Q statistic P-value

`i2` numeric, the I2 statistic

`proxy_rsid` character, proxy variant ID

`proxy_chr` character, proxy chromosome identifier

`proxy_bp` integer, proxy base position

`proxy_ea` character, proxy effect allele

`proxy_oa` character, proxy other allele

`proxy_eaf` numeric, proxy effect allele frequency

`proxy_r2` numeric, proxy r2 with rsid

`trait` character, the GWAS trait

`id` character, the GWAS identifier

`source` character, data source; either the file path, or "data.table" if loaded directly

`correlation` matrix, a correlation matrix of signed R values between variants

`map ColumnMap`, a mapping of class ColumnMap

`qc` list, a named list of filters; name is the filter expression and value is an integer vector of rows that fail the filter

---

`harmonise` *Harmonise GWAS effects*

---

## Description

Harmonise effects across two GWAS datasets. The `gwas` objects are expected to be in standard format - see `standardise_gwas()`. Corrects strand for non-palindromic SNPs (flip BETAs). Drop palindromic variants. Drop non-matching variant alleles.
This function is based on the same function in the Slopehunter and TwoSampleMR packages. I have re-written it here to use the standardised column names and use pure data.table syntax for increased processing speed.

## Usage

```
harmonise(
  gwas1,
  gwas2,
  gwas1_trait = "incidence",
  gwas2_trait = "progression",
  merge = NULL
)
```

## Arguments

| | |
|---|---|
| `gwas1` | a data.frame like object or file path, GWAS 1 |
| `gwas2` | a data.frame like object or file path, GWAS 2 |
| `gwas1_trait` | a string, suffix to add to gwas1 column names e.g. progression –> columns chr_progression, bp_progression, ..., etc. |
| `gwas2_trait` | a string, suffix to add to gwas2 column names e.g. progression –> columns chr_progression, bp_progression, ..., etc. |
| `merge` | a named character vector c(gwas1_col=gwas2_col) of the columns to join on; e.g. c("CHR1"="CHR2","BP1"="BP2"); can be NULL, in which case the gwases must be data.tables and already have keys set |

### Value

a data.table, the harmonised dataset

### References

Slopehunter https://github.com/Osmahmoud/SlopeHunter

TwoSampleMR https://github.com/MRCIEU/TwoSampleMR

---

`ld_matrix` *Calculate LD matrix*

---

### Description

Based on the ieugwasr function (see reference)

### Usage

```
ld_matrix(
  dat,
  colmap = NULL,
  method = "r",
  plink2 = genepi.utils::which_plink2(),
  plink_ref = genepi.utils::which_1000G_reference(build = "GRCh37"),
  ukbb_ref = NULL
)
```

### Arguments

| | |
|---|---|
| `dat` | data.frame like object, or file path, with at least column `rsid`; if columns `ea`,`oa`,`beta`,`eaf` are provided then the variants will be return harmonised to the reference panel (effect allele, data = major allele, reference) |
| `colmap` | a list, mapping to columns list(rsid=?,ea=?,oa=?,beta=?,eaf=?) where ? can be a character vector in the case of harmonised datasets. Warning - it is assumed that harmonised datasets are indeed harmonised, if not, any unharmonised variants will be inappropriately removed. |
| `method` | a string, either `r` or `r2` |
| `plink2` | a string, path to the plink executable |
| `plink_ref` | a string, path to the pfile genome reference |
| `ukbb_ref` | path to a UKBB reference file |

### Value

an LD matrix if only variants provided, else if alleles provided a list(dat=harmonised data, ld_mat=ld_matrix)

### References

ieugwasr::ld_matrix_local()

`lift` *Liftover GWAS positions*

## Description

Determine GWAS build and liftover to required build. This is the same function from the GwasDataImport package, the only difference being that you can specify the build rather than it trying to guess the build (which fails if you are trying to liftover small segments of the genome).

## Usage

```
lift(
  gwas,
  from = "Hg19",
  to = "Hg38",
  snp_col = "snp",
  chr_col = "chr",
  pos_col = "pos",
  ea_col = "ea",
  oa_col = "oa",
  remove_duplicates = TRUE
)
```

## Arguments

| | |
|---|---|
| `gwas` | a data.table, or file path, chr, pos, snp name, effect allele, non-effect allele columns |
| `from` | which build to lift from, one of c("Hg18", "Hg19", "Hg38") |
| `to` | which build to lift over to, one of c("Hg18", "Hg19", "Hg38") |
| `snp_col` | Name of SNP column name. Optional. Uses less certain method of matching if not available |
| `chr_col` | Name of chromosome column name. Required |
| `pos_col` | Name of position column name. Required |
| `ea_col` | Name of effect allele column name. Optional. Might lead to duplicated rows if not presented |
| `oa_col` | Name of other allele column name. Optional. Might lead to duplicated rows if not presented |
| `remove_duplicates` | a logical, whether to remove duplicate IDs |

## Value

data.table with updated position columns

### References

https://github.com/MRCIEU/GwasDataImport

---

`manhattan` *Manhattan plot*

---

### Description

Create a Manhattan plot with ggplot2 geom_point.

### Usage

```
manhattan(
  gwas,
  highlight_snps = NULL,
  highlight_win = 100,
  annotate_snps = NULL,
  colours = c("#d9d9d9", "#bfbfbf"),
  highlight_colour = "#e15758",
  highlight_shape = 16,
  highlight_alpha = 1,
  sig_line_1 = 5e-08,
  sig_line_2 = NULL,
  y_limits = c(NULL, NULL),
  title = NULL,
  subtitle = NULL,
  base_text_size = 14,
  hit_table = FALSE,
  max_table_hits = 10,
  downsample = 0.9,
  downsample_pval = 0.7
)
```

### Arguments

| | |
|---|---|
| `gwas` | a data.table with a minimum of columns SNP, CHR, BP, and P |
| `highlight_snps` | (optional) a character vector of SNPs to highlight |
| `highlight_win` | (optional) a numeric, the number of kb either side of the highlight_snps to also highlight (i.e create peaks) |
| `annotate_snps` | (optional) a character vector of SNPs to annotate |
| `colours` | (optional) a character vector colour codes to be replicated along the chromosomes |
| `highlight_colour` | (optional) a character colour code; the colour to highlight points in |

`highlight_shape`
: (optional) a numeric shape code; the shape of the highlight points (see ggplot2 shape codes)

`highlight_alpha`
: (optional) a numeric value between 0 and 1; the alpha of the highlighted points colour

`sig_line_1`
: (optional) a numeric value (-log10(P)) for where to draw a horizontal line

`sig_line_2`
: (optional) a numeric value (-log10(P)) for where to draw a second horizontal line

`y_limits`
: (optional) a numeric length 2 vector c(min-Y, max-Y)

`title`
: (optional) a string title

`subtitle`
: (optional) a string subtitle

`base_text_size`
: an integer, `base_size` for the ggplot2 theme

`hit_table`
: (optional) a logical, whether to display a table of top hits (lowest P values)

`max_table_hits`
: (optional) an integer, how many top hits to show in the table

`downsample`
: (optional) a numeric between 0 and 1, the proportion by which to downsample by, e.g. 0.6 will remove 60% of points above the downsample_pval threshold (can help increase plotting speed with minimal impact on plot appearance)

`downsample_pval`
: (optional) a numeric between 0 and 1, the p-values affected by downsampling, default >0.1

## Value

a ggplot

---

`miami` *Miami plot*

---

## Description

Create a Miami plot. Please look carefully at the parameters as these largely map to the `manhattan()` parameters, the main difference being that you need to supply a 2 element list of the parameter, one for the upper and one for the lower plot aspect of the Miami plot. Some parameters are not duplicated however - see the example defaults below.

## Usage

```
miami(
  gwases,
  highlight_snps = list(top = NULL, bottom = NULL),
  highlight_win = list(top = 100, bottom = 100),
  annotate_snps = list(top = NULL, bottom = NULL),
  colours = list(top = c("#d9d9d9", "#bfbfbf"), bottom = c("#bfbfbf", "#d9d9d9")),
  highlight_colour = list(top = "#e15758", bottom = "#4f79a7"),
  highlight_shape = list(top = 16, bottom = 16),
  sig_line_1 = list(top = 5e-08, bottom = 5e-08),
  sig_line_2 = list(top = NULL, bottom = NULL),
  y_limits = list(top = c(NULL, NULL), bottom = c(NULL, NULL)),
  title = NULL,
  subtitle = list(top = NULL, bottom = NULL),
  base_text_size = 14,
  hit_table = FALSE,
  max_table_hits = 10,
  downsample = 0.1,
  downsample_pval = 0.1
)
```

## Arguments

| | |
|---|---|
| `gwases` | a list of 2 data.tables |
| `highlight_snps` | (optional) a character vector of SNPs to highlight |
| `highlight_win` | (optional) a numeric, the number of kb either side of the highlight_snps to also highlight (i.e create peaks) |
| `annotate_snps` | (optional) a character vector of SNPs to annotate |
| `colours` | (optional) a character vector colour codes to be replicated along the chromosomes |
| `highlight_colour` | (optional) a character colour code; the colour to highlight points in |
| `highlight_shape` | (optional) a numeric shape code; the shape of the highlight points (see ggplot2 shape codes) |
| `sig_line_1` | (optional) a numeric value (-log10(P)) for where to draw a horizontal line |
| `sig_line_2` | (optional) a numeric value (-log10(P)) for where to draw a second horizontal line |
| `y_limits` | (optional) a numeric length 2 vector c(min-Y, max-Y) |
| `title` | (optional) a string title |
| `subtitle` | (optional) a string subtitle |
| `base_text_size` | an integer, `base_size` for the ggplot2 theme |
| `hit_table` | (optional) a logical, whether to display a table of top hits (lowest P values) |

`max_table_hits`
: (optional) an integer, how many top hits to show in the table

`downsample`
: (optional) a numeric between 0 and 1, the proportion by which to downsample by, e.g. 0.6 will remove 60% of points above the downsample_pval threshold (can help increase plotting speed with minimal impact on plot appearance)

`downsample_pval`
: (optional) a numeric between 0 and 1, the p-values affected by downsampling, default >0.1

## Value

a ggplot

---

`MR` *MR object*

---

## Description

An MR object is a container for vectors and matrices of 2 or more GWAS data.

## Usage

```
MR(
  exposure,
  outcome,
  harmonise_strictness = 2,
  correlation = NULL,
  verbose = TRUE
)
```

## Arguments

`exposure`
: a `GWAS` object or list of `GWAS` objects

`outcome`
: a `GWAS` object

`harmonise_strictness`
: an integer (1,2,3) corresponding to the TwoSampleMR harmonisation options of the same name.

`correlation`
: a matrix, correlation matrix of signed R values between variants

`verbose`
: a logical, print more information

## Value

an S7 class genepi.utils::MR object

**Slots**

`snps` character, variant ID

`chr` character, chromosome identifier

`bp` integer, base position

`ea` character, effect allele

`oa` character, other allele

`eafx` numeric, exposure effect allele frequency

`nx` integer, exposure total number of samples

`ncasex` integer, exposure number of cases

`bx` numeric, exposure effect size

`bxse` numeric, exposure effect size standard error

`px` numeric, exposure p-value

`eafy` numeric, exposure effect allele frequency

`ny` integer, exposure total number of samples

`ncasey` integer, exposure number of cases

`by` numeric, exposure effect size

`byse` numeric, exposure effect size standard error

`py` numeric, exposure p-value

`exposure_id` character, the GWAS identifier

`exposure` character, the GWAS exposure

`outcome_id` character, the GWAS identifier

`outcome` character, the GWAS outcome

`group` integer, grouping variable used for plotting

`index_snp` logical, whether the variant is an index variant (via clumping)

`proxy_snp` character, the id of the proxy snp

`ld_info` logical, whether there is LD information

`info` data.frame, information about the loaded GWAS objects

`correlation` matrix, a correlation matrix of signed R values between variants

## mr_egger *Run Egger MR*

### Description

Run Egger MR

### Usage

```
mr_egger(x, corr = FALSE, ...)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |
| corr | a logical, whether to use the correlation matrix when running MR |
| ... | parameter sink, not used |

## mr_ivw *Run IVW MR*

### Description

Run IVW MR

### Usage

```
mr_ivw(x, corr = FALSE, ...)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |
| corr | a logical, whether to use the correlation matrix when running MR |
| ... | parameter sink, not used |

---

`mr_pcgmm` *Run PC-GMM MR*

---

## Description

Run PC-GMM MR

## Usage

```
mr_pcgmm(x, corr = TRUE, ...)
```

## Arguments

| | |
|---|---|
| `x` | an object of class MR |
| `corr` | a logical, whether to use the correlation matrix when running MR |
| `...` | parameter sink, not used |

---

`mr_results_to_data_table` *MR results to data.table*

---

## Description

MR results to data.table

## Usage

```
mr_results_to_data_table(x)
```

## Arguments

| | |
|---|---|
| `x` | MRResult object to covert to data.table |

## mr_weighted_median *Run weighted median MR*

### Description

Run weighted median MR

### Usage

```
mr_weighted_median(x, corr = FALSE, ...)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |
| corr | a logical, whether to use the correlation matrix when running MR |
| ... | parameter sink, not used |

## mr_weighted_mode *Run weighted mode MR*

### Description

Run weighted mode MR

### Usage

```
mr_weighted_mode(x, corr = FALSE, ...)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |
| corr | a logical, whether to use the correlation matrix when running MR |
| ... | parameter sink, not used |

## plot_coloc_probabilities *Coloc probability plot*

### Description

A plotting wrapper for the `coloc` package. Produces a ggplot for either the prior or posterior probability sensitivity analyses. See the coloc package vignettes for details.

### Usage

```
plot_coloc_probabilities(coloc, rule = "H4 > 0.5", type = "prior", row = 1)
```

### Arguments

| | |
|---|---|
| `coloc` | coloc object, output from `coloc::coloc.abf()` |
| `rule` | a string, a valid rule indicating success e.g. "H4 > 0.5" |
| `type` | a string, either `prior` or `posterior` |
| `row` | an integer, row in a `coloc.susie` or `coloc.signals` object |

### Value

a ggplot

### References

coloc

## plot_mr *Plot MR results*

### Description

Plot MR results

### Usage

```
plot_mr(mr, res)
```

### Arguments

| | |
|---|---|
| `mr` | an object of class MR |
| `res` | a data.table output from run_mr or other MR methods |

`qq_plot` *QQ plot*

## Description

QQ plot

## Usage

```
qq_plot(
  gwas,
  pval_col = "p",
  colours = list(raw = "#2166AC"),
  title = NULL,
  subtitle = NULL,
  plot_corrected = FALSE,
  facet_grid_row_col = NULL,
  facet_grid_col_col = NULL,
  facet_nrow = NULL,
  facet_ncol = NULL
)
```

## Arguments

| | |
|---|---|
| `gwas` | a data.frame like object or valid file path |
| `pval_col` | the P value column |
| `colours` | a 2 element list of colour codes (1-the uncorrected points, 2-the GC corrected points) |
| `title` | a string, the title for the plot |
| `subtitle` | a string, the subtitle for the plot |
| `plot_corrected` | a logical, whether to apply and plot the lambda correction |
| `facet_grid_row_col` | a string, the column name in `gwas` by which to facet the plot (rows) |
| `facet_grid_col_col` | a string, the column name in `gwas` by which to facet the plot (cols) |
| `facet_nrow` | an integer, passed to facet_wrap, the number of rows to facet by (if only facet_grid_row_col is provided) |
| `facet_ncol` | an integer, passed to facet_wrap, the number of cols to facet by (if only facet_grid_col_col is provided) |

## Value

a ggplot

## reset_index_snp *Reset index SNP*

### Description

Reset index SNP

### Usage

```
reset_index_snp(x)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |

## run_mr *Run MR*

### Description

Run MR

### Usage

```
run_mr(
  x,
  corr = FALSE,
  methods = c("mr_ivw", "mr_egger", "mr_weighted_median", "mr_weighted_mode"),
  ...
)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |
| corr | a logical, whether to use the correlation matrix when running MR |
| methods | a string, one of c('mr_ivw','mr_egger','mr_weighted_median','mr_weighted_mode', 'mr_pcgmm') |
| ... | parameter sink, not used |

---

`set_1000G_reference` *Set the 1000G reference path*

---

## Description

Set the 1000G reference path

## Usage

```
set_1000G_reference(path, build = "GRCh37")
```

## Arguments

| | |
|---|---|
| `path` | path to the 1000G reference pfile |
| `build` | one of c("GRCh37", "GRCh38") |

## Value

NULL, updated config file

---

`set_dbsnp_directory` *Set dbSNP directory*

---

## Description

Set dbSNP directory

## Usage

```
set_dbsnp_directory(path)
```

## Arguments

| | |
|---|---|
| `path` | path to the `dbsnp` directory |

## Value

NULL, updated config file

## set_ld_mat *Set the LD matrix*

### Description

Set the LD matrix

### Usage

```
set_ld_mat(x, correlation)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |
| correlation | a matrix, the correlation ('r') matrix |

## set_plink2 *Set the PLINK2 path*

### Description

Set the PLINK2 path

### Usage

```
set_plink2(path)
```

### Arguments

| | |
|---|---|
| path | path to the PLINK2 executable |

### Value

NULL, updated config file

## slopehunter *Slope-Hunter collider bias method*

### Description

Slope-Hunter collider bias method

### Usage

```
slopehunter(
  x,
  ip = 0.001,
  pi0 = 0.6,
  sxy1 = 1e-05,
  bootstraps = 100,
  seed = 777,
  ...
)
```

### Arguments

| | |
|---|---|
| `x` | an object of class MR |
| `ip` | see `xp_thresh` SlopeHunter::hunt() |
| `pi0` | see `init_pi` SlopeHunter::hunt() |
| `sxy1` | see `init_sigmaIP` SlopeHunter::hunt() |
| `bootstraps` | see `M` SlopeHunter::hunt() |
| `seed` | see `seed` SlopeHunter::hunt() |
| `...` | parameter sink, additional ignored parameters |

### Value

an object of class MRResult

## subset_gwas *subset_gwas*

### Description

subset_gwas

### Usage

```
subset_gwas(x, snps)
```

### Arguments

| | |
|---|---|
| x | GWAS object |
| snps | a vector, either row indicies (integers) into the GWAS object (e.g. obtained with filters such as which(GWAS'at'p < 5e-8), or rsids (characters) to be found in the GWAS rsid slot. |

### Value

GWAS object subsetted by snps

---

## to_MRInput *Convert to MendelianRandomization::MRInput object*

---

### Description

Convert to MendelianRandomization::MRInput object

### Usage

```
to_MRInput(x, corr = FALSE)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |
| corr | a logical, whether to use the correlation matrix when running MR |

---

## to_MRMVInput *Convert to MendelianRandomization::MRMVInput object*

---

### Description

Convert to MendelianRandomization::MRMVInput object

### Usage

```
to_MRMVInput(x, corr = FALSE)
```

### Arguments

| | |
|---|---|
| x | an object of class MR |
| corr | a logical, whether to use the correlation matrix when running MR |

---

`which_1000G_reference` *Get 1000G reference path(s)*

---

## Description

Get 1000G reference path(s)

## Usage

```
which_1000G_reference(build = NULL)
```

## Arguments

| | |
|---|---|
| `build` | one of "GRCh37" or "GRCh38", or null to return both |

## Value

a string file path, the currently set 1000G reference path

---

`which_dbsnp_builds` *Get available dbSNP builds*

---

## Description

Get available dbSNP builds

## Usage

```
which_dbsnp_builds(build = NULL)
```

## Arguments

| | |
|---|---|
| `build` | a dbSNP build |

## Value

a list of available dbSNP builds - name(dbSNP build): value(directory_path)

---

`which_dbsnp_directory` *Get dbSNP directory*

---

## Description

Get dbSNP directory

## Usage

```
which_dbsnp_directory()
```

## Value

a string file path, the currently set dbSNP directory path

---

`which_plink2` *Get plink2 path*

---

## Description

Get plink2 path

## Usage

```
which_plink2()
```

## Value

a string file path, the currently set plink2 path

# Index