

# Package ‘MungeSumstats’

September 30, 2022

**Type** Package

**Title** Standardise summary statistics from GWAS

**Version** 1.5.14

**Description** The \*MungeSumstats\* package is designed to facilitate the standardisation of GWAS summary statistics. It reformats inputted summary statistics to include SNP, CHR, BP and can look up these values if any are missing. It also performs dozens of QC and filtering steps to ensure high data quality and minimise inter-study differences.

**URL** <https://github.com/neurogenomics/MungeSumstats>

**BugReports** <https://github.com/neurogenomics/MungeSumstats/issues>

**License** Artistic-2.0

**Depends** R(>= 4.1)

**Imports** magrittr, data.table, utils, R.utils, dplyr, stats,  
GenomicRanges, IRanges, GenomeInfoDb, BSgenome, Biostrings,  
stringr, VariantAnnotation, googleAuthR, httr, jsonlite,  
methods, parallel, rtracklayer, RCurl

**biocViews** SNP, WholeGenome, Genetics, ComparativeGenomics,  
GenomeWideAssociation, GenomicVariation, Preprocessing

**RoxygenNote** 7.2.1

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**Suggests** SNPlocs.Hsapiens.dbSNP144.GRCh37,  
SNPlocs.Hsapiens.dbSNP144.GRCh38,  
SNPlocs.Hsapiens.dbSNP155.GRCh37,  
SNPlocs.Hsapiens.dbSNP155.GRCh38,  
BSgenome.Hsapiens.1000genomes.hs37d5,  
BSgenome.Hsapiens.NCBI.GRCh38, BiocGenerics, S4Vectors,  
rmarkdown, markdown, knitr, testthat (>= 3.0.0), UpSetR,  
BiocStyle, covr, Rsamtools, MatrixGenerics, badger,  
BiocParallel, GenomicFiles

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**git\_url** <https://git.bioconductor.org/packages/MungeSumstats>

**git\_branch** master

**git\_last\_commit** 3a9aa57

**git\_last\_commit\_date** 2022-09-20

**Date/Publication** 2022-09-30

**Author** Alan Murphy [aut, cre] (<<https://orcid.org/0000-0002-2487-8753>>),

Brian Schilder [aut, ctb] (<<https://orcid.org/0000-0001-5949-2191>>),

Nathan Skene [aut] (<<https://orcid.org/0000-0002-6807-3180>>)

**Maintainer** Alan Murphy <[alanmurph94@hotmail.com](mailto:alanmurph94@hotmail.com)>

## R topics documented:

|                                |           |
|--------------------------------|-----------|
| check_ldsc_format . . . . .    | 3         |
| compute_nsize . . . . .        | 4         |
| download_vcf . . . . .         | 5         |
| find_sumstats . . . . .        | 6         |
| formatted_example . . . . .    | 8         |
| format_sumstats . . . . .      | 9         |
| get_genome_builds . . . . .    | 14        |
| hg19ToHg38 . . . . .           | 15        |
| hg38ToHg19 . . . . .           | 16        |
| ieu-a-298 . . . . .            | 17        |
| import_sumstats . . . . .      | 17        |
| index_tabular . . . . .        | 22        |
| liftover . . . . .             | 23        |
| list_sumstats . . . . .        | 24        |
| load_ref_genome_data . . . . . | 25        |
| load_snp_loc_data . . . . .    | 26        |
| parse_logs . . . . .           | 26        |
| raw_ALSvcf . . . . .           | 27        |
| raw_eduAttainOkbay . . . . .   | 28        |
| read_header . . . . .          | 28        |
| read_sumstats . . . . .        | 29        |
| read_vcf . . . . .             | 30        |
| register_cores . . . . .       | 32        |
| standardise_header . . . . .   | 33        |
| sumstatsColHeaders . . . . .   | 34        |
| vcf2df . . . . .               | 34        |
| write_sumstats . . . . .       | 36        |
| <b>Index</b>                   | <b>38</b> |

---

|                   |  |
|-------------------|--|
| check_ldsc_format | <i>Ensures that parameters are compatible with LDSC format</i> |
|-------------------|--|

---

## Description

Format summary statistics for direct input to Linkage Disequilibrium Score (LDSC) regression without the need to use their `munge_sumstats.py` script first.

## Usage

```
check_ldsc_format(
    sumstats_dt,
    save_format,
    convert_n_int,
    allele_flip_check,
    compute_z,
    compute_n
)
```

## Arguments

|                                |  |
|--------------------------------|--|
| <code>sumstats_dt</code>       | data table obj of the summary statistics file for the GWAS.  |
| <code>save_format</code>       | Output format of sumstats. Options are NULL - standardised output format from MungeSumstats, LDSC - output format compatible with LDSC and openGWAS - output compatible with openGWAS VCFs. Default is NULL.   |
| <code>convert_n_int</code>     | Binary, if N (the number of samples) is not an integer, should this be rounded? Default is TRUE.   |
| <code>allele_flip_check</code> | Binary Should the allele columns be checked against reference genome to infer if flipping is necessary. Default is TRUE.   |
| <code>compute_z</code>         | Whether to compute Z-score column from P. Default is FALSE. <b>Note</b> that imputing the Z-score for every SNP will not correct be perfectly correct and may result in a loss of power. This should only be done as a last resort.  |
| <code>compute_n</code>         | Whether to impute N. Default of 0 won't impute, any other integer will be imputed as the N (sample size) for every SNP in the dataset. <b>Note</b> that imputing the sample size for every SNP is not correct and should only be done as a last resort. N can also be inputted with "ldsc", "sum", "giant" or "metal" by passing one of these for this field or a vector of multiple. Sum and an integer value creates an N column in the output whereas giant, metal or ldsc create an Neff or effective sample size. If multiples are passed, the formula used to derive it will be indicated. |

## Details

[LDSC documentation.](#)

**Value**

Formatted summary statistics

**Source**

[LDSC GitHub](#)

---

|               |   |
|---------------|---|
| compute_nsize | <i>Check for N column if not present and user wants, impute N based on user's sample size. <b>NOTE</b> this will be the same value for each SNP which is not necessarily correct and may cause issues down the line. N can also be inputted with "ldsc", "sum", "giant" or "metal" by passing one or multiple of these.</i> |
|---------------|---|

---

**Description**

Check for N column if not present and user wants, impute N based on user's sample size. **NOTE** this will be the same value for each SNP which is not necessarily correct and may cause issues down the line. N can also be inputted with "ldsc", "sum", "giant" or "metal" by passing one or multiple of these.

**Usage**

```
compute_nsize(
  sumstats_dt,
  imputation_ind = FALSE,
  compute_n = c("ldsc", "giant", "metal", "sum"),
  standardise_headers = FALSE,
  force_new = FALSE,
  return_list = TRUE
)
```

**Arguments**

|                |  |
|----------------|--|
| sumstats_dt    | data table obj of the summary statistics file for the GWAS.  |
| imputation_ind | Binary Should a column be added for each imputation step to show what SNPs have imputed values for differing fields. This includes a field denoting SNP allele flipping (flipped). <b>Note</b> these columns will be in the formatted summary statistics returned. Default is FALSE.   |
| compute_n      | How to compute per-SNP sample size (new column "N"). <ul style="list-style-type: none"> <li>• 0: N will not be computed.</li> <li>• &gt;0: If any number &gt;0 is provided, that value will be set as N for every row. <b>Note:</b> Computing N this way is incorrect and should be avoided if at all possible.</li> <li>• "sum": N will be computed as: cases (N_CAS) + controls (N_CON), so long as both columns are present.</li> </ul> |

- "ldsc": N will be computed as effective sample size:  $N_{\text{eff}} = (N_{\text{CAS}} + N_{\text{CON}}) * (N_{\text{CAS}} / (N_{\text{CAS}} + 1 / \text{mean}((N_{\text{CAS}} / (N_{\text{CAS}} + N_{\text{CON}})) (N_{\text{CAS}} + N_{\text{CON}}) == \max(N_{\text{CAS}} + N_{\text{CON}})))$ .
- "giant": N will be computed as effective sample size:  $N_{\text{eff}} = 2 / (1/N_{\text{CAS}} + 1/N_{\text{CON}})$ .
- "metal": N will be computed as effective sample size:  $N_{\text{eff}} = 4 / (1/N_{\text{CAS}} + 1/N_{\text{CON}})$ .

standardise\_headers

Standardise headers first.

force\_new

If "Neff" (or "N") already exists in sumstats\_dt, replace it with the recomputed version.

return\_list

Return the sumstats\_dt within a named list (default: TRUE).

## Value

list("sumstats\_dt"=sumstats\_dt)

## Examples

```
sumstats_dt <- MungeSumstats::formatted_example()
sumstats_dt2 <- MungeSumstats::compute_nsize(sumstats_dt=sumstats_dt,
                                             compute_n=10000)
```

---

download\_vcf

*Download VCF file and its index file from Open GWAS*

---

## Description

Ideally, we would use [gwasvcf](#) instead but it hasn't been made available on CRAN or Bioconductor yet, so we can't include it as a dep.

## Usage

```
download_vcf(
  vcf_url,
  vcf_dir = tempdir(),
  vcf_download = TRUE,
  download_method = "download.file",
  force_new = FALSE,
  quiet = FALSE,
  timeout = 10 * 60,
  nThread = 1
)
```

**Arguments**

|                 |   |
|-----------------|---|
| vcf_url         | Remote URL to VCF file.   |
| vcf_dir         | Where to download the original VCF from Open GWAS. <i>WARNING</i> : This is set to <code>tempdir()</code> by default. This means the raw (pre-formatted) VCFs be deleted upon ending the R session. Change this to keep the raw VCF file on disk (e.g. <code>vcf_dir="/raw_vcf"</code> ). |
| vcf_download    | Download the original VCF from Open GWAS.   |
| download_method | "axel" (multi-threaded) or "download.file" (single-threaded) .  |
| force_new       | Overwrite a previously downloaded VCF with the same path name.  |
| quiet           | Run quietly.  |
| timeout         | How many seconds before giving up on download. Passed to <code>download.file</code> . Default: 10*60 (10min).   |
| nThread         | Number of threads to parallelize over.  |

**Value**

List containing the paths to the downloaded VCF and its index file.

**Examples**

```
#only run the examples if user has internet access:
if(try(is.character(getURL("www.google.com")))==TRUE){
vcf_url <- "https://gwas.mrcieu.ac.uk/files/ieu-a-298/ieu-a-298.vcf.gz"
out_paths <- download_vcf(vcf_url = vcf_url)
}
```

---

find\_sumstats

*Search Open GWAS for datasets matching criteria*


---

**Description**

For each argument, searches for any datasets matching a case-insensitive substring search in the respective metadata column. Users can supply a single character string or a list/vector of character strings.

**Usage**

```
find_sumstats(
  ids = NULL,
  traits = NULL,
  years = NULL,
  consortia = NULL,
  authors = NULL,
  populations = NULL,
```

```

categories = NULL,
subcategories = NULL,
builds = NULL,
pmids = NULL,
min_sample_size = NULL,
min_ncase = NULL,
min_ncontrol = NULL,
min_nsnp = NULL,
include_NAs = FALSE,
access_token = check_access_token()
)

```

### Arguments

|                 |  |
|-----------------|--|
| ids             | List of Open GWAS study IDs (e.g. c("prot-a-664", "ieu-b-4760")).  |
| traits          | List of traits (e.g. c("parkinson", "Alzheimer")).   |
| years           | List of years (e.g. seq(2015,2021) or c(2010, 2012, 2021)).  |
| consortia       | List of consortia (e.g. c("MRC-IEU", "Neale Lab")).  |
| authors         | List of authors (e.g. c("Elsworth", "Kunkle", "Neale")).   |
| populations     | List of populations (e.g. c("European", "Asian")).   |
| categories      | List of categories (e.g. c("Binary", "Continuous", "Disease", "Risk factor"))).                              |
| subcategories   | List of categories (e.g. c("neurological", "Immune", "cardio"))).  |
| builds          | List of genome builds (e.g. c("hg19", "grch37")).  |
| pmids           | List of PubMed ID (exact matches only) (e.g. c(29875488, 30305740, 28240269)).                               |
| min_sample_size | Minimum total number of study participants (e.g. 5000).  |
| min_ncase       | Minimum number of case participants (e.g. 1000).   |
| min_ncontrol    | Minimum number of control participants (e.g. 1000).  |
| min_nsnp        | Minimum number of SNPs (e.g. 200000).  |
| include_NAs     | Include datasets with missing metadata for size criteria (i.e. min_sample_size, min_ncase, or min_ncontrol). |
| access_token    | Google OAuth2 access token. Used to authenticate level of access to data                                     |

### Details

By default, returns metadata for all studies currently in Open GWAS database.

### Value

(Filtered) GWAS metadata table.

## Examples

```
# Only run the examples if user has internet access:
if(try(is.character(getURL("www.google.com")))==TRUE){
### By ID
metagwas <- find_sumstats(ids = c(
  "ieu-b-4760",
  "prot-a-1725",
  "prot-a-664"
))
### By ID and sample size
metagwas <- find_sumstats(
  ids = c("ieu-b-4760", "prot-a-1725", "prot-a-664"),
  min_sample_size = 5000
)
### By criteria
metagwas <- find_sumstats(
  traits = c("alzheimer", "parkinson"),
  years = seq(2015, 2021)
)
}
```

---

formatted\_example

*Formatted example*

---

## Description

Returns an example of summary stats that have had their column names already standardised with [standardise\\_header](#).

## Usage

```
formatted_example(
  path = system.file("extdata", "eduAttainOkbay.txt", package = "MungeSumstats"),
  formatted = TRUE,
  sorted = TRUE
)
```

## Arguments

|           |  |
|-----------|--|
| path      | Path to raw example file. Default to built-in dataset.                   |
| formatted | Whether the column names should be formatted (default:TRUE).             |
| sorted    | Whether the rows should be sorted by genomic coordinates (default:TRUE). |

## Value

sumstats\_dt



## Examples

```
sumstats_dt <- MungeSumstats::formatted_example()
```

---

|                 |  |
|-----------------|--|
| format_sumstats | <i>Check that summary statistics from GWAS are in a homogeneous format</i> |
|-----------------|--|

---

## Description

Check that summary statistics from GWAS are in a homogeneous format

## Usage

```
format_sumstats(  
  path,  
  ref_genome = NULL,  
  convert_ref_genome = NULL,  
  convert_small_p = TRUE,  
  convert_large_p = TRUE,  
  convert_neg_p = TRUE,  
  compute_z = FALSE,  
  force_new_z = FALSE,  
  compute_n = 0L,  
  convert_n_int = TRUE,  
  impute_beta = FALSE,  
  impute_se = FALSE,  
  analysis_trait = NULL,  
  INFO_filter = 0.9,  
  FRQ_filter = 0,  
  pos_se = TRUE,  
  effect_columns_nonzero = FALSE,  
  N_std = 5,  
  N_dropNA = TRUE,  
  rmv_chr = c("X", "Y", "MT"),  
  rmv_chrPrefix = TRUE,  
  on_ref_genome = TRUE,  
  strand_ambig_filter = FALSE,  
  allele_flip_check = TRUE,  
  allele_flip_drop = TRUE,  
  allele_flip_z = TRUE,  
  allele_flip_frq = TRUE,  
  bi_allelic_filter = TRUE,  
  snp_ids_are_rs_ids = TRUE,  
  remove_multi_rs_snp = FALSE,  
  frq_is_maf = TRUE,  
  indels = TRUE,  
  dbSNP = 155,
```

```

check_dups = TRUE,
sort_coordinates = TRUE,
nThread = 1,
save_path = tempfile(fileext = ".tsv.gz"),
write_vcf = FALSE,
tabix_index = FALSE,
return_data = FALSE,
return_format = "data.table",
ldsc_format = FALSE,
save_format = NULL,
log_folder_ind = FALSE,
log_mungesumstats_msgs = FALSE,
log_folder = tempdir(),
imputation_ind = FALSE,
force_new = FALSE,
mapping_file = sumstatsColHeaders
)

```

## Arguments

|                    |   |
|--------------------|---|
| path               | Filepath for the summary statistics file to be formatted. A dataframe or datatable of the summary statistics file can also be passed directly to MungeSumstats using the path parameter.  |
| ref_genome         | name of the reference genome used for the GWAS ("GRCh37" or "GRCh38"). Argument is case-insensitive. Default is NULL which infers the reference genome from the data.   |
| convert_ref_genome | name of the reference genome to convert to ("GRCh37" or "GRCh38"). This will only occur if the current genome build does not match. Default is not to convert the genome build (NULL).  |
| convert_small_p    | Binary, should non-negative p-values $\leq 5e-324$ be converted to 0? Small p-values pass the R limit and can cause errors with LDSC/MAGMA and should be converted. Default is TRUE.  |
| convert_large_p    | Binary, should p-values $>1$ be converted to 1? P-values $>1$ should not be possible and can cause errors with LDSC/MAGMA and should be converted. Default is TRUE.   |
| convert_neg_p      | Binary, should p-values $<0$ be converted to 0? Negative p-values should not be possible and can cause errors with LDSC/MAGMA and should be converted. Default is TRUE.   |
| compute_z          | Whether to compute Z-score column from P. Default is FALSE. <b>Note</b> that imputing the Z-score for every SNP will not correct be perfectly correct and may result in a loss of power. This should only be done as a last resort. |
| force_new_z        | When a "Z" column already exists, it will be used by default. To override and compute a new Z-score column from P set force_new_z=TRUE.   |

|                        |   |
|------------------------|---|
| compute_n              | Whether to impute N. Default of 0 won't impute, any other integer will be imputed as the N (sample size) for every SNP in the dataset. <b>Note</b> that imputing the sample size for every SNP is not correct and should only be done as a last resort. N can also be inputted with "ldsc", "sum", "giant" or "metal" by passing one of these for this field or a vector of multiple. Sum and an integer value creates an N column in the output whereas giant, metal or ldsc create a Neff or effective sample size. If multiples are passed, the formula used to derive it will be indicated. |
| convert_n_int          | Binary, if N (the number of samples) is not an integer, should this be rounded? Default is TRUE.  |
| impute_beta            | Binary, whether BETA should be imputed using other effect data if it isn't present in the sumstats. Note that this imputation is an approximation so could have an effect on downstream analysis. Use with caution. The different methods MungeSumstats will try and impute beta (in this order or priority) are: <ol style="list-style-type: none"> <li>1. log(OR)</li> <li>2. Z x SE</li> </ol> Default value is FALSE.   |
| impute_se              | Binary, whether the standard error should be imputed using other effect data if it isn't present in the sumstats. Note that this imputation is an approximation so could have an effect on downstream analysis. Use with caution. The different methods MungeSumstats will try and impute se (in this order or priority) are: <ol style="list-style-type: none"> <li>1. BETA / Z</li> <li>2. abs(BETA/ qnorm(P/2))</li> </ol> Default is FALSE.   |
| analysis_trait         | If multiple traits were studied, name of the trait for analysis from the GWAS. Default is NULL.   |
| INFO_filter            | numeric The minimum value permissible of the imputation information score (if present in sumstats file). Default 0.9.   |
| FRQ_filter             | numeric The minimum value permissible of the frequency(FRQ) of the SNP (i.e. Allele Frequency (AF)) (if present in sumstats file). By default no filtering is done, i.e. value of 0.  |
| pos_se                 | Binary Should the standard Error (SE) column be checked to ensure it is greater than 0? Those that are, are removed (if present in sumstats file). Default TRUE.  |
| effect_columns_nonzero | Binary should the effect columns in the data BETA,OR (odds ratio),LOG_ODDS,SIGNED_SUMSTAT be checked to ensure no SNP=0. Those that do are removed(if present in sumstats file). Default FALSE.   |
| N_std                  | numeric The number of standard deviations above the mean a SNP's N is needed to be removed. Default is 5.   |
| N_dropNA               | Drop rows where N is missing.Default is TRUE.   |
| rmv_chr                | vector or character The chromosomes on which the SNPs should be removed. Use NULL if no filtering necessary. Default is X, Y and mitochondrial.   |
| rmv_chrPrefix          | Remove "chr" or "CHR" from chromosome names. Default is TRUE.   |
| on_ref_genome          | Binary Should a check take place that all SNPs are on the reference genome by SNP ID. Default is TRUE.  |
| strand_ambig_filter    | Binary Should SNPs with strand-ambiguous alleles be removed. Default is FALSE.  |

|                     |   |
|---------------------|---|
| allele_flip_check   | Binary Should the allele columns be checked against reference genome to infer if flipping is necessary. Default is TRUE.  |
| allele_flip_drop    | Binary Should the SNPs for which neither their A1 or A2 base pair values match a reference genome be dropped. Default is TRUE.  |
| allele_flip_z       | Binary should the Z-score be flipped along with effect and FRQ columns like Beta? It is assumed to be calculated off the effect size not the P-value and so will be flipped i.e. default TRUE.  |
| allele_flip_frq     | Binary should the frequency (FRQ) column be flipped along with effect and z-score columns like Beta? Default TRUE.  |
| bi_allelic_filter   | Binary Should non-biallelic SNPs be removed. Default is TRUE.   |
| snp_ids_are_rs_ids  | Binary Should the supplied SNP ID's be assumed to be RSIDs. If not, imputation using the SNP ID for other columns like base-pair position or chromosome will not be possible. If set to FALSE, the SNP RS ID will be imputed from the reference genome if possible. Default is TRUE.  |
| remove_multi_rs_snp | Binary Sometimes summary statistics can have multiple RSIDs on one row (i.e. related to one SNP), for example "rs5772025_rs397784053". This can cause an error so by default, the first RS ID will be kept and the rest removed e.g. "rs5772025". If you want to just remove these SNPs entirely, set it to TRUE. Default is FALSE.   |
| frq_is_maf          | Conventionally the FRQ column is intended to show the minor/effect allele frequency (MAF) but sometimes the major allele frequency can be inferred as the FRQ column. This logical variable indicates that the FRQ column should be renamed to MAJOR_ALLELE_FRQ if the frequency values appear to relate to the major allele i.e. >0.5. By default this mapping won't occur i.e. is TRUE. |
| indels              | Binary does your Sumstats file contain Indels? These don't exist in our reference file so they will be excluded from checks if this value is TRUE. Default is TRUE.   |
| dbSNP               | version of dbSNP to be used for imputation (144 or 155).  |
| check_dups          | whether to check for duplicates - if formatting QTL datasets this should be set to FALSE otherwise keep as TRUE. Default is TRUE.   |
| sort_coordinates    | Whether to sort by coordinates of resulting sumstats  |
| nThread             | Number of threads to use for parallel processes.  |
| save_path           | File path to save formatted data. Defaults to tempfile(fileext=".tsv.gz").  |
| write_vcf           | Whether to write as VCF (TRUE) or tabular file (FALSE).   |
| tabix_index         | Index the formatted summary statistics with <code>tabix</code> for fast querying.   |
| return_data         | Return data .table, GRanges or VRanges directly to user. Otherwise, return the path to the save data. Default is FALSE.   |
| return_format       | If return_data is TRUE. Object type to be returned ("data.table", "vranges", "granges").  |

|                        |  |
|------------------------|--|
| ldsc_format            | DEPRECATED, do not use. Use save_format="LDSC" instead.  |
| save_format            | Output format of sumstats. Options are NULL - standardised output format from MungeSumstats, LDSC - output format compatible with LDSC and openGWAS - output compatible with openGWAS VCFs. Default is NULL.   |
| log_folder_ind         | Binary Should log files be stored containing all filtered out SNPs (separate file per filter). The data is outputted in the same format specified for the resulting sumstats file. The only exception to this rule is if output is vcf, then log file saved as .tsv.gz. Default is FALSE.  |
| log_mungesumstats_msgs | Binary Should a log be stored containing all messages and errors printed by MungeSumstats in a run. Default is FALSE   |
| log_folder             | Filepath to the directory for the log files and the log of MungeSumstats messages to be stored. Default is a temporary directory.  |
| imputation_ind         | Binary Should a column be added for each imputation step to show what SNPs have imputed values for differing fields. This includes a field denoting SNP allele flipping (flipped). On the flipped value, this denoted whether the alleles were switched based on MungeSumstats initial choice of A1, A2 from the input column headers and thus may not align with what the creator intended. <b>Note</b> these columns will be in the formatted summary statistics returned. Default is FALSE. |
| force_new              | If a formatted file of the same names as save_path exists, formatting will be skipped and this file will be imported instead (default). Set force_new=TRUE to override this.   |
| mapping_file           | MungeSumstats has a pre-defined column-name mapping file which should cover the most common column headers and their interpretations. However, if a column header that is in your file is missing of the mapping we give is incorrect you can supply your own mapping file. Must be a 2 column dataframe with column names "Uncorrected" and "Corrected". See data(sumstatsColHeaders) for default mapping and necessary format.   |

### Value

The address for the modified sumstats file or the actual data dependent on user choice. Also, if log files wanted by the user, the return in both above instances are a list.

### Examples

```
# Pass path to Educational Attainment Okbay sumstat file to a temp directory

eduAttainOkbayPth <- system.file("extdata", "eduAttainOkbay.txt",
  package = "MungeSumstats"
)

## Call uses reference genome as default with more than 2GB of memory,
## which is more than what 32-bit Windows can handle so remove certain checks
## Using dbSNP = 144 for speed as it's smaller but you should use 155 unless
## you know what you are doing and need 144
```

```

is_32bit_windows <-
  .Platform$OS.type == "windows" && .Platform$r_arch == "i386"
if (!is_32bit_windows) {
  reformatted <- format_sumstats(
    path = eduAttainOkbayPth,
    ref_genome = "GRCh37",
    dbSNP = 144
  )
} else {
  reformatted <- format_sumstats(
    path = eduAttainOkbayPth,
    ref_genome = "GRCh37",
    on_ref_genome = FALSE,
    strand_ambig_filter = FALSE,
    bi_allelic_filter = FALSE,
    allele_flip_check = FALSE,
    dbSNP=144
  )
}
# returned location has the updated summary statistics file

```

---

get\_genome\_builds      *Infer genome builds*

---

### Description

Infers the genome build of summary statistics files (GRCh37 or GRCh38) from the data. Uses SNP (RSID) & CHR & BP to get genome build.

### Usage

```

get_genome_builds(
  sumstats_list,
  header_only = TRUE,
  sampled_snps = 10000,
  names_from_paths = FALSE,
  dbSNP = 155,
  nThread = 1
)

```

### Arguments

|               |   |
|---------------|---|
| sumstats_list | A named list of paths to summary statistics, or a named list of data.table objects.   |
| header_only   | Instead of reading in the entire sumstats file, only read in the first N rows where N=sampled_snps. This should help speed up cases where you have to read in sumstats from disk each time. |
| sampled_snps  | Downsample the number of SNPs used when inferring genome build to save time.  |

names\_from\_paths Infer the name of each item in sumstats\_list from its respective file path. Only works if sumstats\_list is a list of paths.

dbSNP version of dbSNP to be used (144 or 155). Default is 155.

nThread Number of threads to use for parallel processes.

### Details

Iterative version of get\_genome\_build.

### Value

ref\_genome the genome build of the data

### Examples

```
# Pass path to Educational Attainment Okbay sumstat file to a temp directory
eduAttainOkbayPth <- system.file("extdata", "eduAttainOkbay.txt",
  package = "MungeSumstats"
)
sumstats_list <- list(ss1 = eduAttainOkbayPth, ss2 = eduAttainOkbayPth)

## Call uses reference genome as default with more than 2GB of memory,
## which is more than what 32-bit Windows can handle so remove certain checks
is_32bit_windows <-
  .Platform$OS.type == "windows" && .Platform$r_arch == "i386"
if (!is_32bit_windows) {

  #multiple sumstats can be passed at once to get all their genome builds:
  #ref_genomes <- get_genome_builds(sumstats_list = sumstats_list)
  #just passing first here for speed
  sumstats_list_quick <- list(ss1 = eduAttainOkbayPth)
  ref_genomes <- get_genome_builds(sumstats_list = sumstats_list_quick,
    dbSNP=144)
}
```

---

hg19ToHg38

*UCSC Chain file hg19 to hg38*

---

### Description

UCSC Chain file hg19 to hg38, .chain.gz file, downloaded from <https://hgdownload.cse.ucsc.edu/goldenpath/hg19/liftOver/> on 09/10/21

### Format

gunzipped chain file

**Details**

UCSC Chain file hg19 to hg38, .chain.gz file, downloaded on 09/10/21 To be used as a back up if the download from UCSC fails.

**hg19ToHg38.over.chain.gz**

NA

**Source**

The chain file was downloaded from <https://hgdownload.cse.ucsc.edu/goldenpath/hg19/liftOver/chain.gz>  
 utils::download.file('ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/liftOver/hg19ToHg38.over.chain.gz')

---

hg38ToHg19

*UCSC Chain file hg38 to hg19*

---

**Description**

UCSC Chain file hg38 to hg19, .chain.gz file, downloaded from <https://hgdownload.cse.ucsc.edu/goldenpath/hg19/liftOver/chain.gz> on 09/10/21

**Format**

gunzipped chain file

**Details**

UCSC Chain file hg38 to hg19, .chain.gz file, downloaded on 09/10/21 To be used as a back up if the download from UCSC fails.

**hg38ToHg19.over.chain.gz**

NA

**Source**

The chain file was downloaded from <https://hgdownload.cse.ucsc.edu/goldenpath/hg38/liftOver/chain.gz>  
 utils::download.file('ftp://hgdownload.cse.ucsc.edu/goldenPath/hg38/liftOver/hg38ToHg19.over.chain.gz')



---

|           |  |
|-----------|--|
| ieu-a-298 | <i>Local ieu-a-298 file from IEU Open GWAS</i> |
|-----------|--|

---

**Description**

Local ieu-a-298 file from IEU Open GWAS, downloaded on 09/10/21.

**Format**

gunzipped tsv file

**Details**

Local ieu-a-298 file from IEU Open GWAS, downloaded on 09/10/21. This is done in case the download in the package vignette fails.

**ieu-a-298.tsv.gz**

NA

**Source**

The file was downloaded with: `MungeSumstats::import_sumstats(ids = "ieu-a-298", ref_genome = "GRCH37")`

---

|                 |   |
|-----------------|---|
| import_sumstats | <i>Import full genome-wide GWAS summary statistics from Open GWAS</i> |
|-----------------|---|

---

**Description**

Requires internet access to run.

**Usage**

```
import_sumstats(  
  ids,  
  vcf_dir = tempdir(),  
  vcf_download = TRUE,  
  save_dir = tempdir(),  
  write_vcf = FALSE,  
  download_method = "download.file",  
  quiet = TRUE,  
  force_new = FALSE,  
  force_new_vcf = FALSE,  
  nThread = 1,  
  parallel_across_ids = FALSE,  
  ...  
)
```

**Arguments**

|                     |   |
|---------------------|---|
| ids                 | List of Open GWAS study IDs (e.g. c("prot-a-664", "ieu-b-4760")).   |
| vcf_dir             | Where to download the original VCF from Open GWAS. <i>WARNING</i> : This is set to tempdir() by default. This means the raw (pre-formatted) VCFs be deleted upon ending the R session. Change this to keep the raw VCF file on disk (e.g. vcf_dir="./raw_vcf"). |
| vcf_download        | Download the original VCF from Open GWAS.   |
| save_dir            | Directory to save formatted summary statistics in.  |
| write_vcf           | Whether to write as VCF (TRUE) or tabular file (FALSE).   |
| download_method     | "axel" (multi-threaded) or "download.file" (single-threaded).   |
| quiet               | Run quietly.  |
| force_new           | If a formatted file of the same names as save_path exists, formatting will be skipped and this file will be imported instead (default). Set force_new=TRUE to override this.  |
| force_new_vcf       | Overwrite a previously downloaded VCF with the same path name.  |
| nThread             | Number of threads to use for parallel processes.  |
| parallel_across_ids | If parallel_across_ids=TRUE and nThread>1, then each ID in ids will be processed in parallel.   |
| ...                 | Arguments passed on to <a href="#">format_sumstats</a>  |
| path                | Filepath for the summary statistics file to be formatted. A dataframe or datatable of the summary statistics file can also be passed directly to MungeSumstats using the path parameter.  |
| ref_genome          | name of the reference genome used for the GWAS ("GRCh37" or "GRCh38"). Argument is case-insensitive. Default is NULL which infers the reference genome from the data.   |
| convert_ref_genome  | name of the reference genome to convert to ("GRCh37" or "GRCh38"). This will only occur if the current genome build does not match. Default is not to convert the genome build (NULL).  |
| convert_small_p     | Binary, should non-negative p-values $\leq 5e-324$ be converted to 0? Small p-values pass the R limit and can cause errors with LDSC/MAGMA and should be converted. Default is TRUE.  |
| convert_large_p     | Binary, should p-values $>1$ be converted to 1? P-values $>1$ should not be possible and can cause errors with LDSC/MAGMA and should be converted. Default is TRUE.   |
| convert_neg_p       | Binary, should p-values $<0$ be converted to 0? Negative p-values should not be possible and can cause errors with LDSC/MAGMA and should be converted. Default is TRUE.   |
| compute_z           | Whether to compute Z-score column from P. Default is FALSE. <b>Note</b> that imputing the Z-score for every SNP will not correct be perfectly correct and may result in a loss of power. This should only be done as a last resort.                             |

- force\_new\_z** When a "Z" column already exists, it will be used by default. To override and compute a new Z-score column from P set `force_new_z=TRUE`.
- compute\_n** Whether to impute N. Default of 0 won't impute, any other integer will be imputed as the N (sample size) for every SNP in the dataset. **Note** that imputing the sample size for every SNP is not correct and should only be done as a last resort. N can also be inputted with "ldsc", "sum", "giant" or "metal" by passing one of these for this field or a vector of multiple. Sum and an integer value creates an N column in the output whereas giant, metal or ldsc create an Neff or effective sample size. If multiples are passed, the formula used to derive it will be indicated.
- convert\_n\_int** Binary, if N (the number of samples) is not an integer, should this be rounded? Default is TRUE.
- impute\_beta** Binary, whether BETA should be imputed using other effect data if it isn't present in the sumstats. Note that this imputation is an approximation so could have an effect on downstream analysis. Use with caution. The different methods MungeSumstats will try and impute beta (in this order or priority) are:
1.  $\log(\text{OR})$
  2.  $Z \times \text{SE}$
- Default value is FALSE.
- impute\_se** Binary, whether the standard error should be imputed using other effect data if it isn't present in the sumstats. Note that this imputation is an approximation so could have an effect on downstream analysis. Use with caution. The different methods MungeSumstats will try and impute se (in this order or priority) are:
1.  $\text{BETA} / Z$
  2.  $\text{abs}(\text{BETA} / \text{qnorm}(P/2))$
- Default is FALSE.
- analysis\_trait** If multiple traits were studied, name of the trait for analysis from the GWAS. Default is NULL.
- INFO\_filter** numeric The minimum value permissible of the imputation information score (if present in sumstats file). Default 0.9.
- FRQ\_filter** numeric The minimum value permissible of the frequency (FRQ) of the SNP (i.e. Allele Frequency (AF)) (if present in sumstats file). By default no filtering is done, i.e. value of 0.
- pos\_se** Binary Should the standard Error (SE) column be checked to ensure it is greater than 0? Those that are, are removed (if present in sumstats file). Default TRUE.
- effect\_columns\_nonzero** Binary should the effect columns in the data BETA,OR (odds ratio),LOG\_ODDS,SIGNED\_SUMSTAT be checked to ensure no SNP=0. Those that do are removed(if present in sumstats file). Default FALSE.
- N\_std** numeric The number of standard deviations above the mean a SNP's N is needed to be removed. Default is 5.
- N\_dropNA** Drop rows where N is missing.Default is TRUE.
- rmv\_chr** vector or character The chromosomes on which the SNPs should be removed. Use NULL if no filtering necessary. Default is X, Y and mitochondrial.
- rmv\_chrPrefix** Remove "chr" or "CHR" from chromosome names. Default is TRUE.

**on\_ref\_genome** Binary Should a check take place that all SNPs are on the reference genome by SNP ID. Default is TRUE.

**strand\_ambig\_filter** Binary Should SNPs with strand-ambiguous alleles be removed. Default is FALSE.

**allele\_flip\_check** Binary Should the allele columns be checked against reference genome to infer if flipping is necessary. Default is TRUE.

**allele\_flip\_drop** Binary Should the SNPs for which neither their A1 or A2 base pair values match a reference genome be dropped. Default is TRUE.

**allele\_flip\_z** Binary should the Z-score be flipped along with effect and FRQ columns like Beta? It is assumed to be calculated off the effect size not the P-value and so will be flipped i.e. default TRUE.

**allele\_flip\_frq** Binary should the frequency (FRQ) column be flipped along with effect and z-score columns like Beta? Default TRUE.

**bi\_allelic\_filter** Binary Should non-biallelic SNPs be removed. Default is TRUE.

**snp\_ids\_are\_rs\_ids** Binary Should the supplied SNP ID's be assumed to be RSIDs. If not, imputation using the SNP ID for other columns like base-pair position or chromosome will not be possible. If set to FALSE, the SNP RS ID will be imputed from the reference genome if possible. Default is TRUE.

**remove\_multi\_rs\_snp** Binary Sometimes summary statistics can have multiple RSIDs on one row (i.e. related to one SNP), for example "rs5772025\_rs397784053". This can cause an error so by default, the first RS ID will be kept and the rest removed e.g. "rs5772025". If you want to just remove these SNPs entirely, set it to TRUE. Default is FALSE.

**frq\_is\_maf** Conventionally the FRQ column is intended to show the minor/effect allele frequency (MAF) but sometimes the major allele frequency can be inferred as the FRQ column. This logical variable indicates that the FRQ column should be renamed to MAJOR\_ALLELE\_FRQ if the frequency values appear to relate to the major allele i.e. >0.5. By default this mapping won't occur i.e. is TRUE.

**indels** Binary does your Sumstats file contain Indels? These don't exist in our reference file so they will be excluded from checks if this value is TRUE. Default is TRUE.

**dbSNP** version of dbSNP to be used for imputation (144 or 155).

**check\_dups** whether to check for duplicates - if formatting QTL datasets this should be set to FALSE otherwise keep as TRUE. Default is TRUE.

**sort\_coordinates** Whether to sort by coordinates of resulting sumstats

**save\_path** File path to save formatted data. Defaults to `tempfile(fileext=".tsv.gz")`.

**tabix\_index** Index the formatted summary statistics with `tabix` for fast querying.

**return\_data** Return `data.table`, `GRanges` or `VRanges` directly to user. Otherwise, return the path to the save data. Default is FALSE.

**return\_format** If `return_data` is TRUE. Object type to be returned ("`data.table`", "`vranges`", "`granges`").

**ldsc\_format** DEPRECATED, do not use. Use `save_format="LDSC"` instead.

`save_format` Output format of sumstats. Options are NULL - standardised output format from MungeSumstats, LDSC - output format compatible with LDSC and openGWAS - output compatible with openGWAS VCFs. Default is NULL.

`log_folder_ind` Binary Should log files be stored containing all filtered out SNPs (separate file per filter). The data is outputted in the same format specified for the resulting sumstats file. The only exception to this rule is if output is vcf, then log file saved as .tsv.gz. Default is FALSE.

`log_mungesumstats_msgs` Binary Should a log be stored containing all messages and errors printed by MungeSumstats in a run. Default is FALSE

`log_folder` Filepath to the directory for the log files and the log of MungeSumstats messages to be stored. Default is a temporary directory.

`imputation_ind` Binary Should a column be added for each imputation step to show what SNPs have imputed values for differing fields. This includes a field denoting SNP allele flipping (flipped). On the flipped value, this denoted whether the alleles were switched based on MungeSumstats initial choice of A1, A2 from the input column headers and thus may not align with what the creator intended. **Note** these columns will be in the formatted summary statistics returned. Default is FALSE.

`mapping_file` MungeSumstats has a pre-defined column-name mapping file which should cover the most common column headers and their interpretations. However, if a column header that is in your file is missing of the mapping we give is incorrect you can supply your own mapping file. Must be a 2 column dataframe with column names "Uncorrected" and "Corrected". See `data(sumstatsColHeaders)` for default mapping and necessary format.

## Value

Either a named list of data objects or paths, depending on the arguments passed to `format_sumstats`.

## Examples

```
#only run the examples if user has internet access:
if(try(is.character(getURL("www.google.com")))==TRUE){
### Search by criteria
metagwas <- find_sumstats(
  traits = c("parkinson", "alzheimer"),
  min_sample_size = 5000
)
### Only use a subset for testing purposes
ids <- (dplyr::arrange(metagwas, nsnp))$id

### Default usage
## You can supply \code{import_sumstats()}
## with a list of as many OpenGWAS IDs as you want,
## but we'll just give one to save time.

## Call uses reference genome as default with more than 2GB of memory,
## which is more than what 32-bit Windows can handle so remove certain checks
## commented out down to runtime
```

```
# datasets <- import_sumstats(ids = ids[1])
}
```

---

|               |                                |
|---------------|--------------------------------|
| index_tabular | <i>Tabix-index file: table</i> |
|---------------|--------------------------------|

---

## Description

Convert summary stats file to tabix format.

## Usage

```
index_tabular(
  path,
  chrom_col = "CHR",
  start_col = "BP",
  end_col = start_col,
  overwrite = TRUE,
  remove_tmp = TRUE,
  verbose = TRUE
)
```

## Arguments

|            |   |
|------------|---|
| path       | Path to GWAS summary statistics file.   |
| chrom_col  | Name of the chromosome column in sumstats_dt (e.g. "CHR").  |
| start_col  | Name of the starting genomic position column in sumstats_dt (e.g. "POS", "start").  |
| end_col    | Name of the ending genomic position column in sumstats_dt (e.g. "POS", "end").<br>Can be the same as start_col when sumstats_dt only contains SNPs that span 1 base pair (bp) each. |
| overwrite  | A logical(1) indicating whether dest should be over-written, if it already exists.  |
| remove_tmp | Remove the temporary uncompressed version of the file (.tsv).   |
| verbose    | Print messages.   |

## Value

Path to tabix-indexed tabular file

## Source

Borrowed function from [echotabix](#).

## See Also

Other tabix: [index\\_vcf\(\)](#)

## Examples

```
sumstats_dt <- MungeSumstats::formatted_example()
path <- tempfile(fileext = ".tsv")
MungeSumstats::write_sumstats(sumstats_dt = sumstats_dt, save_path = path)
indexed_file <- MungeSumstats::index_tabular(path = path)
```

---

|          |                              |
|----------|------------------------------|
| liftover | <i>Genome build liftover</i> |
|----------|------------------------------|

---

## Description

Transfer genomic coordinates from one genome build to another.

## Usage

```
liftover(
  sumstats_dt,
  convert_ref_genome,
  ref_genome,
  imputation_ind = TRUE,
  chrom_col = "CHR",
  start_col = "BP",
  end_col = start_col,
  as_granges = FALSE,
  style = "NCBI",
  verbose = TRUE
)
```

## Arguments

|                    |  |
|--------------------|--|
| sumstats_dt        | data table obj of the summary statistics file for the GWAS.  |
| convert_ref_genome | name of the reference genome to convert to ("GRCh37" or "GRCh38"). This will only occur if the current genome build does not match. Default is not to convert the genome build (NULL).   |
| ref_genome         | name of the reference genome used for the GWAS ("GRCh37" or "GRCh38"). Argument is case-insensitive. Default is NULL which infers the reference genome from the data.  |
| imputation_ind     | Binary Should a column be added for each imputation step to show what SNPs have imputed values for differing fields. This includes a field denoting SNP allele flipping (flipped). On the flipped value, this denoted whether the alleles were switched based on MungeSumstats initial choice of A1, A2 from the input column headers and thus may not align with what the creator intended. <b>Note</b> these columns will be in the formatted summary statistics returned. Default is FALSE. |
| chrom_col          | Name of the chromosome column in sumstats_dt (e.g. "CHR").   |

|            |   |
|------------|---|
| start_col  | Name of the starting genomic position column in sumstats_dt (e.g. "POS", "start").  |
| end_col    | Name of the ending genomic position column in sumstats_dt (e.g. "POS", "end").<br>Can be the same as start_col when sumstats_dt only contains SNPs that span 1 base pair (bp) each. |
| as_granges | Return results as <a href="#">GRanges</a> instead of a <a href="#">data.table</a> (default: FALSE).   |
| style      | Style to return <a href="#">GRanges</a> object in (e.g. "NCBI" = 4; "UCSC" = "chr4"); (default: "NCBI").  |
| verbose    | Print messages.   |

**Value**

Lifted summary stats in [data.table](#) or [GRanges](#) format.

**Source**

[liftOver](#)  
[UCSC chain files](#)

**Examples**

```
sumstats_dt <- MungeSumstats::formatted_example()

sumstats_dt_hg38 <- liftOver(sumstats_dt=sumstats_dt,
                             ref_genome = "hg19",
                             convert_ref_genome="hg38")
```

---

|               |                                       |
|---------------|---------------------------------------|
| list_sumstats | <i>List munged summary statistics</i> |
|---------------|---------------------------------------|

---

**Description**

Searches for and lists local GWAS summary statistics files munged by [format\\_sumstats](#) or [import\\_sumstats](#).

**Usage**

```
list_sumstats(
  save_dir = getwd(),
  pattern = "*.tsv.gz$",
  ids_from_file = TRUE,
  verbose = TRUE
)
```



**Arguments**

|               |  |
|---------------|--|
| save_dir      | Top-level directory to recursively search for summary statistics files within.                         |
| pattern       | Regex pattern to search for files with.  |
| ids_from_file | Try to extract dataset IDs from file names. If FALSE, will infer IDs from the directory names instead. |
| verbose       | Print messages.  |

**Value**

Named vector of summary stats paths.

**Examples**

```
save_dir <- system.file("extdata", package = "MungeSumstats")
munged_files <- MungeSumstats::list_sumstats(save_dir = save_dir)
```

---

load\_ref\_genome\_data *Load the reference genome data for SNPs of interest*

---

**Description**

Load the reference genome data for SNPs of interest

**Usage**

```
load_ref_genome_data(snps, ref_genome, dbSNP = c(144, 155), msg = NULL)
```

**Arguments**

|            |   |
|------------|---|
| snps       | Character vector SNPs by rs_id from sumstats file of interest.                    |
| ref_genome | Name of the reference genome used for the GWAS (GRCh37 or GRCh38)                 |
| dbSNP      | version of dbSNP to be used (144 or 155)  |
| msg        | Optional name of the column missing from the dataset in question. Default is NULL |

**Value**

data table of snpsById, filtered to SNPs of interest.

**Source**

```
sumstats_dt <- formatted_example()
rsids <- MungeSumstats:::load_ref_genome_data(snps
= sumstats_dt$SNP, ref_genome = "GRCh37", dbSNP=144)
```

---

|                   |   |
|-------------------|---|
| load_snp_loc_data | <i>Loads the SNP locations and alleles for Homo sapiens extracted from NCBI dbSNP Build 144. Reference genome version is dependent on user input.</i> |
|-------------------|---|

---

### Description

Loads the SNP locations and alleles for Homo sapiens extracted from NCBI dbSNP Build 144. Reference genome version is dependent on user input.

### Usage

```
load_snp_loc_data(ref_genome, dbSNP = c(144, 155), msg = NULL)
```

### Arguments

|            |   |
|------------|---|
| ref_genome | name of the reference genome used for the GWAS (GRCh37 or GRCh38) |
| dbSNP      | version of dbSNP to be used (144 or 155)                          |
| msg        | Optional name of the column missing from the dataset in question  |

### Value

SNP\_LOC\_DATA SNP positions and alleles for Homo sapiens extracted from NCBI dbSNP Build 144

### Examples

```
SNP_LOC_DATA <- load_snp_loc_data("GRCH37", dbSNP=144)
```

---

|            |                                  |
|------------|----------------------------------|
| parse_logs | <i>Parse data from log files</i> |
|------------|----------------------------------|

---

### Description

Parses data from the log files generated by [format\\_sumstats](#) or [import\\_sumstats](#) when the argument log\_mungesumstats\_msgs is set to TRUE.

### Usage

```
parse_logs(
  save_dir = getwd(),
  pattern = "MungeSumstats_log_msg.txt$",
  verbose = TRUE
)
```

**Arguments**

save\_dir            Top-level directory to recursively search for log files within.  
 pattern            Regex pattern to search for files with.  
 verbose            Print messages.

**Value**

[data.table](#) of parsed log data.

**Examples**

```
save_dir <- system.file("extdata", package = "MungeSumstats")
log_data <- MungeSumstats::parse_logs(save_dir = save_dir)
```

---

|            |  |
|------------|--|
| raw_ALSvcf | <i>GWAS Amyotrophic lateral sclerosis ieu open GWAS project - Subset</i> |
|------------|--|

---

**Description**

VCF (VCFv4.2) of the GWAS Amyotrophic lateral sclerosis ieu open GWAS project Dataset: ebi-a-GCST005647. A subset of 99 SNPs

**Format**

vcf document with 528 items relating to 99 SNPs

**Details**

A VCF file (VCFv4.2) of the GWAS Amyotrophic lateral sclerosis ieu open GWAS project has been subsetted here to act as an example summary statistic file in VCF format which has some issues in the formatting. MungeSumstats can correct these issues and produced a standardised summary statistics format.

**ALSvcf.vcf**

NA

**Source**

The summary statistics VCF (VCFv4.2) file was downloaded from <https://gwas.mrcieu.ac.uk/datasets/ebi-a-GCST005647/> and formatted to a .rda with the following: #Get example VCF dataset, use GWAS Amyotrophic lateral sclerosis ALS\_GWAS\_VCF <- readLines("ebi-a-GCST005647.vcf.gz") #Subset to just the first 99 SNPs ALSvcf <- ALS\_GWAS\_VCF[1:528] writeLines(ALSvcf, "inst/extdata/ALSvcf.v")

---

raw\_eduAttainOkbay      *GWAS Educational Attainment Okbay 2016 - Subset*

---

### Description

GWAS Summary Statistics on Educational Attainment by Okbay et al 2016: PMID: 27898078  
 PMCID: PMC5509058 DOI: 10.1038/ng1216-1587b. A subset of 93 SNPs

### Format

txt document with 94 items

### Details

GWAS Summary Statistics on Educational Attainment by Okbay et al 2016 has been subsetted here to act as an example summary statistic file which has some issues in the formatting. MungeSumstats can correct these issues.

### eduAttainOkbay.txt

NA

### Source

The summary statistics file was downloaded from <https://www.nature.com/articles/ng.3552> and formatted to a .rda with the following: #Get example dataset, use Educational-Attainment\_Okbay\_2016  
 link<-"Educational-Attainment\_Okbay\_2016/EduYears\_Discovery\_5000.txt" eduAttainOkbay<-readLines(link)  
 #There is an issue where values end with .0, this 0 is removed in func #There are also SNPs  
 not on ref genome or arebi/tri allelic #So need to remove these in this dataset as its used  
 for testing tmp <- tempfile() writelines(eduAttainOkbay,con=tmp) eduAttainOkbay <- data.table::fread(tm  
 #DT read removes the .0's #remove those not on ref genome and withbi/tri allelic rmv <-  
 c("rs192818565","rs79925071","rs1606974","rs1871109","rs73074378","rs7955289") eduAttainOkbay  
 <- eduAttainOkbay[!MarkerName data.table::fwrite(eduAttainOkbay,file=tmp,sep="\t")  
 eduAttainOkbay <- readLines(tmp) writelines(eduAttainOkbay,"inst/extdata/eduAttainOkbay.txt")

---

read\_header

*Read in file header*

---

### Description

Read in file header

### Usage

```
read_header(path, n = 2L, skip_vcf_metadata = FALSE, nThread = 1)
```

**Arguments**

|                   |  |
|-------------------|--|
| path              | Filepath for the summary statistics file to be formatted. A dataframe or datatable of the summary statistics file can also be passed directly to MungeSumstats using the path parameter. |
| n                 | integer. The (maximal) number of lines to read. Negative values indicate that one should read up to the end of input on the connection.  |
| skip_vcf_metadata | logical, should VCF metadata be ignored  |
| nThread           | Number of threads to use for parallel processes.   |

**Value**

First n lines of the VCF header

**Examples**

```
path <- system.file("extdata", "eduAttain0kbay.txt",
                    package = "MungeSumstats")
header <- read_header(path = path)
```

---

read\_sumstats

*Determine summary statistics file type and read them into memory*


---

**Description**

Determine summary statistics file type and read them into memory

**Usage**

```
read_sumstats(
  path,
  nrows = Inf,
  standardise_headers = FALSE,
  samples = 1,
  sampled_rows = 10000L,
  nThread = 1,
  mapping_file = sumstatsColHeaders
)
```

**Arguments**

|       |  |
|-------|--|
| path  | Filepath for the summary statistics file to be formatted. A dataframe or datatable of the summary statistics file can also be passed directly to MungeSumstats using the path parameter. |
| nrows | integer. The (maximal) number of lines to read. If Inf, will read in all rows.   |

|                     |  |
|---------------------|--|
| standardise_headers | Standardise headers first.   |
| samples             | Which samples to use: <ul style="list-style-type: none"> <li>• 1 : Only the first sample will be used (<i>DEFAULT</i>).</li> <li>• NULL : All samples will be used.</li> <li>• c("&lt;sample_id1&gt;",&lt;sample_id2&gt;",&lt;...&gt;") : Only user-selected samples will be used (case-insensitive).</li> </ul>   |
| sampled_rows        | First N rows to sample. Set NULL to use full sumstats_file. when determining whether cols are empty.   |
| nThread             | Number of threads to use for parallel processes.   |
| mapping_file        | MungeSumstats has a pre-defined column-name mapping file which should cover the most common column headers and their interpretations. However, if a column header that is in your file is missing of the mapping we give is incorrect you can supply your own mapping file. Must be a 2 column dataframe with column names "Uncorrected" and "Corrected". See data(sumstatsColHeaders) for default mapping and necessary format. |

**Value**

data.table of formatted summary statistics

**Examples**

```
path <- system.file("extdata", "eduAttainOkbay.txt",
  package = "MungeSumstats"
)
eduAttainOkbay <- read_sumstats(path = path)
```

---

read\_vcf

*Read in VCF file*

---

**Description**

Read in a VCF file as a [VCF](#) or a [data.table](#). Can optionally save the VCF/data.table as well.

**Usage**

```
read_vcf(
  path,
  as_datatable = TRUE,
  save_path = NULL,
  tabix_index = FALSE,
  samples = 1,
  which = NULL,
  use_params = TRUE,
  sampled_rows = 10000L,
```

```

download = TRUE,
vcf_dir = tempdir(),
download_method = "download.file",
force_new = FALSE,
mt_thresh = 100000L,
nThread = 1,
verbose = TRUE
)

```

## Arguments

|                 |   |
|-----------------|---|
| path            | Path to local or remote VCF file.   |
| as_data_table   | Return the data as a <a href="#">data.table</a> (default: TRUE) or a <a href="#">VCF</a> (FALSE).   |
| save_path       | File path to save formatted data. Defaults to <code>tempfile(fileext=".tsv.gz")</code> .  |
| tabix_index     | Index the formatted summary statistics with <a href="#">tabix</a> for fast querying.  |
| samples         | Which samples to use: <ul style="list-style-type: none"> <li>• 1 : Only the first sample will be used (<i>DEFAULT</i>).</li> <li>• NULL : All samples will be used.</li> <li>• <code>c("&lt;sample_id1&gt;", "&lt;sample_id2&gt;", ...)</code> : Only user-selected samples will be used (case-insensitive).</li> </ul> |
| which           | A <a href="#">GRanges</a> describing the sequences and ranges to be queried. Variants whose POS lies in the interval(s) [ <code>start</code> , <code>end</code> ] are returned. If <code>which</code> is not specified all ranges are returned.   |
| use_params      | When TRUE (default), increases the speed of reading in the VCF by omitting columns that are empty based on the head of the VCF (NAs only). NOTE that that this requires the VCF to be sorted, bgzip-compressed, tabix-indexed, which <a href="#">read_vcf</a> will attempt to do.                                       |
| sampled_rows    | First N rows to sample. Set NULL to use full <code>sumstats_file</code> . when determining whether cols are empty.  |
| download        | Download the VCF (and its index file) to a temp folder before reading it into R. This is important to keep TRUE when <code>nThread</code> >1 to avoid making too many queries to remote file.   |
| vcf_dir         | Where to download the original VCF from Open GWAS. <i>WARNING</i> : This is set to <code>tempdir()</code> by default. This means the raw (pre-formatted) VCFs be deleted upon ending the R session. Change this to keep the raw VCF file on disk (e.g. <code>vcf_dir="/raw_vcf"</code> ).                               |
| download_method | "axel" (multi-threaded) or "download.file" (single-threaded) .  |
| force_new       | If a formatted file of the same names as <code>save_path</code> exists, formatting will be skipped and this file will be imported instead (default). Set <code>force_new=TRUE</code> to override this.  |
| mt_thresh       | When the number of rows (variants) in the VCF is < <code>mt_thresh</code> , only use single-threading for reading in the VCF. This is because the overhead of parallelisation outweighs the speed benefits when VCFs are small.   |
| nThread         | Number of threads to use for parallel processes.  |
| verbose         | Print messages.   |

**Value**

The VCF file in data.table format.

**Source**

```
##### Benchmarking ##### library(VCFwrenchR) library(VariantAnnotation) path <- "https://gwas.mrcieu.ac.uk/files/ieua-298/ieua-298.vcf.gz"
vcf <- VariantAnnotation::readVcf(file = path) N <- 1e5 vcf_sub <- vcf[1:N, ] res <- microbenchmark::microbenchmark("vcf2df" = {dat1 <- MungeSumstats::vcf2df(vcf = vcf_sub)}, "VCFwrenchR" = {dat2 <- as.data.frame(x = vcf_sub)}, "VRanges" = {dat3 <- data.table::as.data.table(methods::as(vcf_sub, "VRanges"))}, times = 1)
```

[Discussion on VariantAnnotation GitHub](#)

[Discussion on VariantAnnotation GitHub](#)

**Examples**

```
##### Local file #####
path <- system.file("extdata", "ALSvcf.vcf", package = "MungeSumstats")
sumstats_dt <- read_vcf(path = path)

##### Remote file #####
## Small GWAS (0.2Mb)
# path <- "https://gwas.mrcieu.ac.uk/files/ieua-298/ieua-298.vcf.gz"
# sumstats_dt2 <- read_vcf(path = path)

## Large GWAS (250Mb)
# path <- "https://gwas.mrcieu.ac.uk/files/ubm-a-2929/ubm-a-2929.vcf.gz"
# sumstats_dt3 <- read_vcf(path = path, nThread = 11)

### Very large GWAS (500Mb)
# path <- "https://gwas.mrcieu.ac.uk/files/ieua-1124/ieua-1124.vcf.gz"
# sumstats_dt4 <- read_vcf(path = path, nThread = 11)
```

---

register\_cores

*Register cores*

---

**Description**

Register a multi-threaded instances using **BiocParallel**.

**Usage**

```
register_cores(workers = 1, progressbar = TRUE)
```

**Arguments**

|             |   |
|-------------|---|
| workers     | integer(1) Number of workers. Defaults to all cores available as determined by detectCores. For a SOCK cluster workers can be a character() vector of host names. |
| progressbar | logical(1) Enable progress bar (based on plyr:::progress_text).   |



**Value**

Null output.

---

|                    |   |
|--------------------|---|
| standardise_header | <i>Standardise the column headers in the Summary Statistics files</i> |
|--------------------|---|

---

**Description**

Use a reference data table of common column header names (stored in `sumstatsColHeaders` or user inputted mapping file) to convert them to a standard set, i.e. `chromosome` -> `CHR`. This function does not check that all the required column headers are present. The amended header is written directly back into the file

**Usage**

```
standardise_header(
  sumstats_dt,
  mapping_file = sumstatsColHeaders,
  uppercase_unmapped = TRUE,
  return_list = TRUE
)
```

**Arguments**

|                                 |  |
|---------------------------------|--|
| <code>sumstats_dt</code>        | data table obj of the summary statistics file for the GWAS.  |
| <code>mapping_file</code>       | <code>MungeSumstats</code> has a pre-defined column-name mapping file which should cover the most common column headers and their interpretations. However, if a column header that is in your file is missing of the mapping we give is incorrect you can supply your own mapping file. Must be a 2 column dataframe with column names "Uncorrected" and "Corrected". See <code>data(sumstatsColHeaders)</code> for default mapping and necessary format. |
| <code>uppercase_unmapped</code> | For columns that could not be identified in the <code>mapping_file</code> , return them in the same format they were input as (without forcing them to uppercase).   |
| <code>return_list</code>        | Return the <code>sumstats_dt</code> within a named list (default: <code>TRUE</code> ).   |

**Value**

list containing `sumstats_dt`, the modified summary statistics data table object

**Examples**

```
sumstats_dt <- data.table::fread(system.file("extdata", "eduAttainOkbay.txt",
                                             package = "MungeSumstats"))
sumstats_dt2 <- standardise_header(sumstats_dt=sumstats_dt)
```

---

sumstatsColHeaders      *Summary Statistics Column Headers*

---

### Description

List of uncorrected column headers often found in GWAS Summary Statistics column headers. Note the effect allele will always be the A2 allele, this is the approach done for VCF(<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC>) This is enforced with the column header corrections here and also the check allele flipping test.

### Usage

```
data("sumstatsColHeaders")
```

### Format

dataframe with 2 columns

### Source

The code to prepare the .Rda file from the marker file is: # Most the data in the below table comes from the LDSC github wiki data("sumstatsColHeaders") # Make additions to sumstatsColHeaders using github version of MungeSumstats-# shown is an example of adding columns for Standard Error (SE) #se\_cols <- data.frame("Uncorrected"=c("SE", "se", "STANDARD.ERROR", # "STANDARD\_ERROR", "STANDARD\_ERROR") #Corrected"=rep("SE",5)) #sumstatsColHeaders <- rbind(sumstatsColHeaders,se\_cols) #Once additions are made, order & save the new mapping dataset #now sort ordering -important for logic that # uncorrected=corrected comes first sumstatsColHeaders\$ordering <- sumstatsColHeaders\$Uncorrected sumstatsColHeaders <- sumstatsColHeaders[order(sumstatsColHeaders\$Corrected, sumstatsColHeaders\$ordering = TRUE),] rownames(sumstatsColHeaders)<-1:nrow(sumstatsColHeaders) sumstatsColHeaders\$ordering <- NULL #manually move FRWQUENCY to above MAR - github issue 95 frequency <- sumstatsColHeaders[sumstatsColHeaders\$Corrected=="MAF",] if(as.integer(rownames(frequency))> sumstatsColHeaders[as.integer(rownames(frequency)),] <- maf sumstatsColHeaders[as.integer(rownames(maf))> sumstatsColHeaders[as.integer(rownames(frequency)),] <- maf sumstatsColHeaders[as.integer(rownames(maf))> sumstatsColHeaders[as.integer(rownames(frequency)),] <- maf sumstatsColHeaders[as.integer(rownames(maf))> sumstatsColHeaders[as.integer(rownames(frequency)),] <- maf } usethis::use\_data(sumstatsColHeaders,overwrite = TRUE, internal=TRUE) save(sumstatsColHeaders, file="data/sumstatsColHeaders.rda") # You will need to restart your r session for effects to take account

---

vcf2df

*VCF to DF*

---

### Description

Function to convert a **VariantAnnotation** CollapsedVCF/ExpandedVCF object to a data.frame.

**Usage**

```
vcf2df(
  vcf,
  add_sample_names = TRUE,
  add_rowranges = TRUE,
  drop_empty_cols = TRUE,
  unique_cols = TRUE,
  unique_rows = TRUE,
  unlist_cols = TRUE,
  sampled_rows = NULL,
  verbose = TRUE
)
```

**Arguments**

|                  |  |
|------------------|--|
| vcf              | Variant Call Format (VCF) file imported into R as a <b>VariantAnnotation CollapsedVCF/ ExpandedVCF</b> object. |
| add_sample_names | Append sample names to column names (e.g. "EZ" -> "EZ_ubm-a-2929").  |
| add_rowranges    | Include rowRanges from VCF as well.  |
| drop_empty_cols  | Drop columns that are filled entirely with: NA, ".", or "".  |
| unique_cols      | Only keep uniquely named columns.  |
| unique_rows      | Only keep unique rows.   |
| unlist_cols      | If any columns are lists instead of vectors, unlist them. Required to be TRUE when unique_rows=TRUE.           |
| sampled_rows     | First N rows to sample. Set NULL to use full sumstats_file. when determining whether cols are empty.           |
| verbose          | Print messages.  |

**Value**

data.frame version of VCF

**Source**

[Original code source](#)

**vcfR:**

```
if(!require("pinfsc50")) install.packages("pinfsc50") vcf_file <- system.file("extdata", "pinf_sc50.vcf.gz",
package = "pinfsc50") vcf <- read.vcfR( vcf_file, verbose = FALSE ) vcf_df_list <- vcfR::vcfR2tidy(vcf,
single_frame=TRUE) vcf_df <- data.table::data.table(vcf_df_list$dat)
```

**Examples**

```
#### VariantAnnotation ####
# path <- "https://github.com/brentp/vcfanno/raw/master/example/exac.vcf.gz"
path <- system.file("extdata", "ALSvcf.vcf",
                   package = "MungeSumstats")

vcf <- VariantAnnotation::readVcf(file = path)
vcf_df <- MungeSumstats::vcf2df(vcf = vcf)
```

---

|                |                                     |
|----------------|-------------------------------------|
| write_sumstats | <i>Write sum stats file to disk</i> |
|----------------|-------------------------------------|

---

**Description**

Write sum stats file to disk

**Usage**

```
write_sumstats(
  sumstats_dt,
  save_path,
  ref_genome,
  sep = "\t",
  write_vcf = FALSE,
  save_format = NULL,
  tabix_index = FALSE,
  nThread = 1,
  return_path = FALSE,
  save_path_check = FALSE
)
```

**Arguments**

|             |   |
|-------------|---|
| sumstats_dt | data table obj of the summary statistics file for the GWAS.   |
| save_path   | File path to save formatted data. Defaults to <code>tempfile(fileext=".tsv.gz")</code> .  |
| ref_genome  | name of the reference genome used for the GWAS ("GRCh37" or "GRCh38"). Argument is case-insensitive. Default is NULL which infers the reference genome from the data.   |
| sep         | The separator between columns. Defaults to the character in the set <code>[, \t   ; : ]</code> that separates the sample of rows into the most number of lines with the same number of fields. Use NULL or "" to specify no separator; i.e. each line a single character column like <code>base::readLines</code> does. |
| write_vcf   | Whether to write as VCF (TRUE) or tabular file (FALSE).   |
| save_format | Output format of sumstats. Options are NULL - standardised output format from MungeSumstats, LDSC - output format compatible with LDSC and openGWAS - output compatible with openGWAS VCFs. Default is NULL.  |

|                 |   |
|-----------------|---|
| tabix_index     | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread         | The number of threads to use. Experiment to see what works best for your data on your hardware.   |
| return_path     | Return save_path. This will have been modified in some cases (e.g. after compressing and tabix-indexing a previously un-compressed file). |
| save_path_check | Ensure path name is valid (given the other arguments) before writing (default: FALSE).  |

**Value**

If return\_path=TRUE, returns save\_path. Else returns NULL.

**Source**

**VariantAnnotation::writeVcf has some unexpected/silent file renaming behavior**

**Examples**

```
path <- system.file("extdata", "eduAttain0kbay.txt",
  package = "MungeSumstats"
)
eduAttain0kbay <- read_sumstats(path = path)
write_sumstats(
  sumstats_dt = eduAttain0kbay,
  save_path = tempfile(fileext = ".tsv.gz")
)
```

# Index

- \* **datasets**
  - sumstatsColHeaders, [34](#)
- \* **tabix**
  - index\_tabular, [22](#)
  
- check\_ldsc\_format, [3](#)
- CollapsedVCF, [35](#)
- compute\_nsize, [4](#)
  
- data.table, [24](#), [27](#), [30](#), [31](#)
- download\_vcf, [5](#)
  
- ExpandedVCF, [35](#)
  
- find\_sumstats, [6](#)
- format\_sumstats, [9](#), [18](#), [24](#), [26](#)
- formatted\_example, [8](#)
  
- get\_genome\_builds, [14](#)
- GRanges, [24](#), [31](#)
  
- hg19ToHg38, [15](#)
- hg38ToHg19, [16](#)
  
- ieu-a-298, [17](#)
- import\_sumstats, [17](#), [24](#), [26](#)
- index\_tabular, [22](#)
- index\_vcf, [22](#)
  
- liftover, [23](#)
- list\_sumstats, [24](#)
- load\_ref\_genome\_data, [25](#)
- load\_snp\_loc\_data, [26](#)
  
- parse\_logs, [26](#)
  
- raw\_ALSvcf, [27](#)
- raw\_eduAttainOkbay, [28](#)
- read\_header, [28](#)
- read\_sumstats, [29](#)
- read\_vcf, [30](#), [31](#)
  
- register\_cores, [32](#)
  
- standardise\_header, [8](#), [33](#)
- sumstatsColHeaders, [34](#)
  
- VCF, [30](#), [31](#)
- vcf2df, [34](#)
  
- write\_sumstats, [36](#)