

# Package ‘MungeSumstats’

May 16, 2024

**Type** Package

**Title** Standardise summary statistics from GWAS

**Version** 1.13.1

**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(>= 1.59.1), RCurl

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

**RoxygenNote** 7.3.1

**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** devel

**git\_last\_commit** 08bb31d

**git\_last\_commit\_date** 2024-05-02

**Repository** Bioconductor 3.20

**Date/Publication** 2024-05-15

**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>

## Contents

|  |    |
|--|----|
| api_query . . . . .                    | 5  |
| axel . . . . .                         | 6  |
| check_access_token . . . . .           | 7  |
| check_allele_flip . . . . .            | 7  |
| check_allele_merge . . . . .           | 9  |
| check_bi_allelic . . . . .             | 10 |
| check_bp_range . . . . .               | 11 |
| check_chr . . . . .                    | 12 |
| check_col_order . . . . .              | 13 |
| check_drop_indels . . . . .            | 13 |
| check_dup_bp . . . . .                 | 14 |
| check_dup_col . . . . .                | 15 |
| check_dup_row . . . . .                | 16 |
| check_dup_snp . . . . .                | 17 |
| check_effect_columns_nonzero . . . . . | 18 |
| check_empty_cols . . . . .             | 19 |
| check_four_step_col . . . . .          | 19 |
| check_frq . . . . .                    | 20 |
| check_frq_maf . . . . .                | 21 |
| check_info_score . . . . .             | 21 |
| check_ldsc_format . . . . .            | 22 |
| check_miss_data . . . . .              | 23 |
| check_multi_gwas . . . . .             | 24 |
| check_multi_rs_snp . . . . .           | 25 |
| check_no_allele . . . . .              | 26 |
| check_no_chr_bp . . . . .              | 28 |
| check_no_rs_snp . . . . .              | 29 |
| check_no_snp . . . . .                 | 30 |
| check_numeric . . . . .                | 31 |
| check_n_int . . . . .                  | 32 |
| check_n_num . . . . .                  | 32 |

|                                     |    |
|-------------------------------------|----|
| check_on_ref_genome . . . . .       | 33 |
| check_pos_se . . . . .              | 35 |
| check_range_p_val . . . . .         | 36 |
| check_row_snp . . . . .             | 37 |
| check_save_path . . . . .           | 38 |
| check_signed_col . . . . .          | 39 |
| check_small_p_val . . . . .         | 40 |
| check_strand_ambiguous . . . . .    | 41 |
| check_tabular . . . . .             | 42 |
| check_two_step_col . . . . .        | 42 |
| check_vcf . . . . .                 | 43 |
| check_vital_col . . . . .           | 43 |
| check_zscore . . . . .              | 44 |
| column_dictionary . . . . .         | 45 |
| compute_nsize . . . . .             | 46 |
| compute_sample_size . . . . .       | 47 |
| compute_sample_size_n . . . . .     | 48 |
| compute_sample_size_neff . . . . .  | 49 |
| convert_sumstats . . . . .          | 50 |
| DF_to_dt . . . . .                  | 51 |
| downloader . . . . .                | 51 |
| download_vcf . . . . .              | 52 |
| drop_duplicate_cols . . . . .       | 54 |
| drop_duplicate_rows . . . . .       | 54 |
| find_sumstats . . . . .             | 55 |
| formatted_example . . . . .         | 56 |
| format_sumstats . . . . .           | 57 |
| get_access_token . . . . .          | 64 |
| get_chain_file . . . . .            | 64 |
| get_eff_frq_allele_combns . . . . . | 65 |
| get_genome_build . . . . .          | 65 |
| get_genome_builds . . . . .         | 67 |
| get_query_content . . . . .         | 68 |
| get_unique_name_log_file . . . . .  | 69 |
| get_vcf_sample_ids . . . . .        | 69 |
| granges_to_dt . . . . .             | 70 |
| gwasinfo . . . . .                  | 70 |
| hg19ToHg38 . . . . .                | 71 |
| hg38ToHg19 . . . . .                | 71 |
| ieu-a-298 . . . . .                 | 72 |
| import_sumstats . . . . .           | 72 |
| index_tabular . . . . .             | 78 |
| index_vcf . . . . .                 | 79 |
| infer_effect_column . . . . .       | 80 |
| is_tabix . . . . .                  | 81 |
| legacy_ids . . . . .                | 82 |
| liftover . . . . .                  | 82 |
| list_sumstats . . . . .             | 84 |

|                                      |     |
|--------------------------------------|-----|
| load_ref_genome_data . . . . .       | 85  |
| load_snp_loc_data . . . . .          | 86  |
| logs_example . . . . .               | 86  |
| make_allele_upper . . . . .          | 87  |
| messenger . . . . .                  | 87  |
| message_parallel . . . . .           | 88  |
| parse_dropped_chrom . . . . .        | 88  |
| parse_dropped_duplicates . . . . .   | 89  |
| parse_dropped_INFO . . . . .         | 89  |
| parse_dropped_nonA1A2 . . . . .      | 90  |
| parse_dropped_nonBiallelic . . . . . | 90  |
| parse_dropped_nonRef . . . . .       | 91  |
| parse_flipped . . . . .              | 91  |
| parse_genome_build . . . . .         | 92  |
| parse_idStandard . . . . .           | 92  |
| parse_logs . . . . .                 | 93  |
| parse_pval_large . . . . .           | 93  |
| parse_pval_neg . . . . .             | 94  |
| parse_pval_small . . . . .           | 94  |
| parse_report . . . . .               | 95  |
| parse_snps_freq_05 . . . . .         | 95  |
| parse_snps_not_formatted . . . . .   | 96  |
| parse_time . . . . .                 | 96  |
| preview_sumstats . . . . .           | 97  |
| raw_ALSvcf . . . . .                 | 97  |
| raw_eduAttainOkbay . . . . .         | 98  |
| read_header . . . . .                | 98  |
| read_sumstats . . . . .              | 99  |
| read_vcf . . . . .                   | 100 |
| read_vcf_genome . . . . .            | 102 |
| read_vcf_info . . . . .              | 103 |
| read_vcf_markername . . . . .        | 103 |
| read_vcf_parallel . . . . .          | 104 |
| read_vcf_pval . . . . .              | 105 |
| register_cores . . . . .             | 106 |
| remove_empty_cols . . . . .          | 106 |
| report_summary . . . . .             | 107 |
| select_api . . . . .                 | 107 |
| select_vcf_fields . . . . .          | 108 |
| sort_coords . . . . .                | 109 |
| sort_coords_datatable . . . . .      | 109 |
| sort_coord_genomicranges . . . . .   | 110 |
| standardise_header . . . . .         | 110 |
| sumstatsColHeaders . . . . .         | 111 |
| supported_suffixes . . . . .         | 112 |
| to_granges . . . . .                 | 113 |
| to_vranges . . . . .                 | 113 |
| unlist_dt . . . . .                  | 114 |

|                               |     |
|-------------------------------|-----|
| validate_parameters . . . . . | 114 |
| vcf2df . . . . .              | 119 |
| write_sumstats . . . . .      | 121 |

|              |            |
|--------------|------------|
| <b>Index</b> | <b>123</b> |
|--------------|------------|

---

|           |  |
|-----------|--|
| api_query | <i>Wrapper for sending queries and payloads to API</i> |
|-----------|--|

---

**Description**

There are a number of different GET and POST endpoints in the GWAS database API. This is a generic way to access them

**Usage**

```
api_query(  
  path,  
  query = NULL,  
  access_token = check_access_token(),  
  method = "GET",  
  silent = TRUE,  
  encode = "json",  
  timeout = 300  
)
```

**Arguments**

|              |   |
|--------------|---|
| path         | Either a full query path (e.g. for get) or an endpoint (e.g. for post) queries  |
| query        | If post query, provide a list of arguments as the payload. NULL by default  |
| access_token | Google OAuth2 access token. Used to authenticate level of access to data. By default, checks if already authenticated through get_access_token and if not then does not perform authentication. |
| method       | GET (default) or POST, DELETE etc   |
| silent       | TRUE/FALSE to be passed to httr call. TRUE by default   |
| encode       | Default = json, see httr::POST for options  |
| timeout      | Default = 300, avoid increasing this, preferentially simplify the query first.  |

**Value**

httr response object

---

|      |                        |
|------|------------------------|
| axel | <i>axel downloader</i> |
|------|------------------------|

---

## Description

R wrapper for axel, which enables multi-threaded download of a single large file.

## Usage

```
axel(  
  input_url,  
  output_path,  
  background = FALSE,  
  nThread = 1,  
  force_overwrite = FALSE,  
  quiet = TRUE,  
  alternate = TRUE,  
  check_certificates = FALSE  
)
```

## Arguments

|                    |  |
|--------------------|--|
| input_url          | input_url.                             |
| output_path        | output_path.                           |
| background         | Run in background                      |
| nThread            | Number of threads to parallelize over. |
| force_overwrite    | Overwrite existing file.               |
| quiet              | Run quietly.                           |
| alternate          | alternate,                             |
| check_certificates | check_certificates                     |

## Value

Path where the file has been downloaded

## See Also

<https://github.com/axel-download-accelerator/axel/>

Other downloaders: [downloader\(\)](#)

---

|                    |  |
|--------------------|--|
| check_access_token | <i>Check if authentication has been made</i> |
|--------------------|--|

---

### Description

If a call to get\_access\_token() has been made then it will have generated mrbase.oauth. Pass the token if it is present, if not, return NULL and do not authenticate.

### Usage

```
check_access_token()
```

### Value

NULL or access\_token depending on current authentication state

---

|                   |  |
|-------------------|--|
| check_allele_flip | <i>Ensure A1 &amp; A2 are correctly named, if GWAS SNP constructed as Alternative/Reference or Risk/Nonrisk alleles these SNPs will need to be converted to Reference/Alternative or Nonrisk/Risk. Here non-risk is defined as what's on the reference genome (this may not always be the case).</i> |
|-------------------|--|

---

### Description

Ensure A1 & A2 are correctly named, if GWAS SNP constructed as Alternative/Reference or Risk/Nonrisk alleles these SNPs will need to be converted to Reference/Alternative or Nonrisk/Risk. Here non-risk is defined as what's on the reference genome (this may not always be the case).

### Usage

```
check_allele_flip(
  sumstats_dt,
  path,
  ref_genome,
  rsids,
  allele_flip_check,
  allele_flip_drop,
  allele_flip_z,
  allele_flip_frq,
  bi_allelic_filter,
  flip_frq_as_biallelic,
  imputation_ind,
  log_folder_ind,
  check_save_out,
```

```

    tabix_index,
    nThread,
    log_files,
    standardise_headers = FALSE,
    mapping_file,
    dbSNP
)

```

## 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.  |
| 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.  |
| flip_frq_as_biallelic | Binary Should non-bi-allelic SNPs frequency values be flipped as 1-p despite there being other alternative alleles? Default is FALSE but if set to TRUE, this allows non-bi-allelic SNPs to be kept despite needing flipping.  |
| 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. |
| 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.  |
| tabix_index           | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |



|                     |  |
|---------------------|--|
| nThread             | Number of threads to use for parallel processes.   |
| log_files           | list of log file locations   |
| standardise_headers | Run standardise_sumstats_column_headers_crossplatform first.   |
| 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. |
| dbSNP               | version of dbSNP to be used for imputation (144 or 155).   |

**Value**

A list containing two data tables:

- sumstats\_dt: the modified summary statistics data.table object.
- rsids: snpsById, filtered to SNPs of interest if loaded already. Or else NULL.
- log\_files: log file list

---

|                    |   |
|--------------------|---|
| check_allele_merge | <i>Ensure that A1:A2 or A1/A2 or A1&gt;A2 or A2&gt;A1 aren't merged into 1 column</i> |
|--------------------|---|

---

**Description**

Ensure that A1:A2 or A1/A2 or A1>A2 or A2>A1 aren't merged into 1 column

**Usage**

```
check_allele_merge(sumstats_dt, path)
```

**Arguments**

|             |  |
|-------------|--|
| sumstats_dt | data table obj of the summary statistics file for the GWAS |
| path        | Filepath for the summary statistics file to be formatted   |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object.

---

|                  |                                  |
|------------------|----------------------------------|
| check_bi_allelic | <i>Remove non-biallelic SNPs</i> |
|------------------|----------------------------------|

---

## Description

Remove non-biallelic SNPs

## Usage

```
check_bi_allelic(
  sumstats_dt,
  path,
  ref_genome,
  bi_allelic_filter,
  rsids,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files,
  dbSNP
)
```

## 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.   |
| bi_allelic_filter | Binary Should non-biallelic SNPs be removed. Default is TRUE.   |
| 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. |
| tabix_index       | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread           | Number of threads to use for parallel processes.  |
| log_files         | list of log file locations  |
| dbSNP             | version of dbSNP to be used for imputation (144 or 155).  |

**Value**

A list containing two data tables:

- `sumstats_dt`: the modified summary statistics data table object
- `rsids`: `snpsById`, filtered to SNPs of interest if loaded already. Or else `NULL`.
- `log_files`: log file list

---

|                             |  |
|-----------------------------|--|
| <code>check_bp_range</code> | <i>Ensure that the Base-pair column values are all within the range for the chromosome</i> |
|-----------------------------|--|

---

**Description**

Ensure that the Base-pair column values are all within the range for the chromosome

**Usage**

```
check_bp_range(
  sumstats_dt,
  path,
  ref_genome,
  log_folder_ind,
  imputation_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

**Arguments**

|                             |   |
|-----------------------------|---|
| <code>path</code>           | Filepath for the summary statistics file to be formatted. A dataframe or datatable of the summary statistics file can also be passed directly to <code>MungeSumstats</code> using the <code>path</code> parameter.  |
| <code>ref_genome</code>     | name of the reference genome used for the GWAS ("GRCh37" or "GRCh38"). Argument is case-insensitive. Default is <code>NULL</code> which infers the reference genome from the data.  |
| <code>log_folder_ind</code> | 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 <code>sumstats</code> file. The only exception to this rule is if output is <code>vcf</code> , then log file saved as <code>.tsv.gz</code> . Default is <code>FALSE</code> .  |
| <code>imputation_ind</code> | 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 <code>MungeSumstats</code> 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. |
| tabix_index | Index the formatted summary statistics with <b>tabix</b> for fast querying.           |
| nThread     | Number of threads to use for parallel processes.                                      |
| log_files   | list of log file locations  |

Value

list containing sumstats\_dt, the modified summary statistics data table object and the log file list

---

|           |                                   |
|-----------|-----------------------------------|
| check_chr | <i>Standardize the CHR column</i> |
|-----------|-----------------------------------|

---

Description

Maps chromosome names to the default Ensembl/NCBI naming style and removes SNPs with non-standard CHR entries. Optionally, also removes SNPs on user-specified chromosomes.

Usage

```
check_chr(  
  sumstats_dt,  
  log_files,  
  check_save_out,  
  rmv_chr,  
  nThread,  
  tabix_index,  
  log_folder_ind  
)
```

Arguments

|                |   |
|----------------|---|
| sumstats_dt    | data.table with summary statistics  |
| log_files      | list of locations for all log files   |
| check_save_out | list of parameters for saved files  |
| rmv_chr        | Chromosomes to exclude from the formatted summary statistics file. Use NULL if no filtering is necessary. Default is c("X", "Y", "MT") which removes all non-autosomal SNPs.  |
| nThread        | Number of threads to use for parallel processes.  |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| 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. |

**Value**

list containing the updated summary statistics data.table and the updated log file locations list

---

|                 |  |
|-----------------|--|
| check_col_order | <i>Ensure that the first three columns are SNP, CHR, BP in that order and then A1, A2 if present</i> |
|-----------------|--|

---

**Description**

Ensure that the first three columns are SNP, CHR, BP in that order and then A1, A2 if present

**Usage**

```
check_col_order(sumstats_dt, path)
```

**Arguments**

|             |  |
|-------------|--|
| sumstats_dt | data table obj of the summary statistics file for the GWAS |
| path        | Filepath for the summary statistics file to be formatted   |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object

---

|                   |  |
|-------------------|--|
| check_drop_indels | <i>Drop Indels from summary statistics</i> |
|-------------------|--|

---

**Description**

Drop Indels from summary statistics

**Usage**

```
check_drop_indels(
  sumstats_dt,
  drop_indels,
  path,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

**Arguments**

|                |   |
|----------------|---|
| sumstats_dt    | data table obj of the summary statistics file for the GWAS  |
| drop_indels    | Binary, should any indels found in the sumstats be dropped? These can not be checked against a reference dataset and will have the same RS ID and position as SNPs which can affect downstream analysis. Default is False.  |
| 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.  |
| 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. |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread        | Number of threads to use for parallel processes.  |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object

**Source**

```
sumstats_dt <- MungeSumstats::formatted_example() sumstats <- check_drop_indels(sumstats_dt
= sumstats_dt, drop_indels = TRUE)
```

---

check\_dup\_bp

*Ensure all rows have unique positions, drop those that don't*

---

**Description**

Ensure all rows have unique positions, drop those that don't

**Usage**

```
check_dup_bp(
  sumstats_dt,
  bi_allelic_filter,
  check_dups,
  indels,
  path,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

**Arguments**

|                   |   |
|-------------------|---|
| bi_allelic_filter | Binary Should non-biallelic SNPs be removed. Default is TRUE.   |
| check_dups        | whether to check for duplicates - if formatting QTL datasets this should be set to FALSE otherwise keep as TRUE. Default 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.   |
| 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.  |
| 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. |
| tabix_index       | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread           | Number of threads to use for parallel processes.  |
| log_files         | list of log file locations  |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object and log files list

---

|               |  |
|---------------|--|
| check_dup_col | <i>Ensure that no columns are duplicated</i> |
|---------------|--|

---

**Description**

Ensure that no columns are duplicated

**Usage**

```
check_dup_col(sumstats_dt, path)
```

**Arguments**

|             |  |
|-------------|--|
| sumstats_dt | data table obj of the summary statistics file for the GWAS |
| path        | Filepath for the summary statistics file to be formatted   |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object

---

|               |   |
|---------------|---|
| check_dup_row | <i>Ensure all rows are unique based on SNP,CHR,BP,A1,A2, drop those that aren't</i> |
|---------------|---|

---

## Description

Ensure all rows are unique based on SNP,CHR,BP,A1,A2, drop those that aren't

## Usage

```
check_dup_row(
  sumstats_dt,
  check_dups,
  path,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

## Arguments

|                |   |
|----------------|---|
| check_dups     | whether to check for duplicates - if formatting QTL datasets this should be set to FALSE otherwise keep as TRUE. Default is TRUE.   |
| 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.  |
| 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. |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread        | Number of threads to use for parallel processes.  |
| log_files      | list of log file locations  |

## Value

list containing sumstats\_dt, the modified summary statistics data table object and log files list



---

|               |   |
|---------------|---|
| check_dup_snp | <i>Ensure all rows have unique SNP IDs, drop those that don't</i> |
|---------------|---|

---

## Description

Ensure all rows have unique SNP IDs, drop those that don't

## Usage

```
check_dup_snp(
  sumstats_dt,
  indels,
  path,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files,
  bi_allelic_filter,
  check_dups
)
```

## Arguments

|                   |   |
|-------------------|---|
| 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.   |
| 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.  |
| 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. |
| tabix_index       | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread           | Number of threads to use for parallel processes.  |
| log_files         | list of log file locations  |
| bi_allelic_filter | Binary Should non-biallelic SNPs be removed. Default is TRUE.   |
| check_dups        | whether to check for duplicates - if formatting QTL datasets this should be set to FALSE otherwise keep as TRUE. Default is TRUE.   |

## Value

list containing sumstats\_dt, the modified summary statistics data table object and log files list

---

check\_effect\_columns\_nonzero

*Ensure that the standard error (se) is positive for all SNPs*

---

## Description

Ensure that the standard error (se) is positive for all SNPs

## Usage

```
check_effect_columns_nonzero(
  sumstats_dt,
  path,
  effect_columns_nonzero,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

## 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.  |
| 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.   |
| 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. |
| tabix_index            | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread                | Number of threads to use for parallel processes.  |
| log_files              | list of log file locations  |

## Value

list containing sumstats\_dt, the modified summary statistics data table object and the log file list

---

|                  |                                |
|------------------|--------------------------------|
| check_empty_cols | <i>Check for empty columns</i> |
|------------------|--------------------------------|

---

**Description**

Empty columns contain only ".", NA, or 0

**Usage**

```
check_empty_cols(sumstats_dt, sampled_rows = NULL, verbose = TRUE)
```

**Arguments**

|              |  |
|--------------|--|
| sampled_rows | First N rows to sample. Set NULL to use full sumstats_file. when determining whether cols are empty. |
| verbose      | Print messages.  |

**Value**

empty\_cols

---

|                     |   |
|---------------------|---|
| check_four_step_col | <i>Ensure that CHR:BP:A2:A1 aren't merged into 1 column</i> |
|---------------------|---|

---

**Description**

Ensure that CHR:BP:A2:A1 aren't merged into 1 column

**Usage**

```
check_four_step_col(sumstats_dt, path)
```

**Arguments**

|             |  |
|-------------|--|
| sumstats_dt | data table obj of the summary statistics file for the GWAS |
| path        | Filepath for the summary statistics file to be formatted   |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object

---

|           |   |
|-----------|---|
| check_frq | <i>Ensure all SNPs have frq score above threshold</i> |
|-----------|---|

---

## Description

Ensure all SNPs have frq score above threshold

## Usage

```
check_frq(
  sumstats_dt,
  path,
  FRQ_filter,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

## 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.  |
| 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.  |
| 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. |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread        | Number of threads to use for parallel processes.  |
| log_files      | list of log file locations  |

## Value

list containing sumstats\_dt, the modified summary statistics data table object and the log file list

---

|               |  |
|---------------|--|
| check_frq_maf | <i>Check that FRQ column refers to minor/effect allele frequency not major</i> |
|---------------|--|

---

### Description

Check that FRQ column refers to minor/effect allele frequency not major

### Usage

```
check_frq_maf(sumstats_dt, frq_is_maf)
```

### Arguments

|            |   |
|------------|---|
| 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. |
|------------|---|

### Value

sumstats\_dt, the modified summary statistics data table object

---

|                  |  |
|------------------|--|
| check_info_score | <i>Ensure all SNPs have info score above threshold</i> |
|------------------|--|

---

### Description

Ensure all SNPs have info score above threshold

### Usage

```
check_info_score(
  sumstats_dt,
  INFO_filter,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

**Arguments**

|                |         |  |
|----------------|---------|--|
| INFO_filter    | numeric | The minimum value permissible of the imputation information score (if present in sumstats file). Default 0.9.  |
| 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. |
| tabix_index    |         | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| nThread        |         | Number of threads to use for parallel processes.   |
| log_files      |         | list of log file locations.  |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object and the log file list

---

|                   |  |
|-------------------|--|
| 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**

|             |  |
|-------------|--|
| sumstats_dt | data table obj of the summary statistics file for the GWAS.  |
| 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. <b>NOTE</b> - If LDSC format is used, the naming convention of A1 as the reference (genome build) allele and A2 as the effect allele will be reversed to match LDSC (A1 will now be the effect allele). See more info on this <a href="#">here</a> . Note that any effect columns (e.g. Z) will be in relation to A1 now instead of A2. |

|                   |  |
|-------------------|--|
| convert_n_int     | Binary, if N (the number of samples) is not an integer, should this be rounded? Default is TRUE.   |
| allele_flip_check | Binary Should the allele columns be checked against reference genome to infer if flipping is necessary. Default is TRUE.   |
| compute_z         | Whether to compute Z-score column. Default is FALSE. This can be computed from Beta and SE with (Beta/SE) or P ( $Z:=\text{sign}(\text{BETA})*\sqrt{\text{stats::qchisq}(P,1,\text{lower}=\text{FALSE})}$ ). <b>Note</b> that imputing the Z-score from P for every SNP will not be perfectly correct and may result in a loss of power. This should only be done as a last resort. Use 'BETA' to impute by BETA/SE and 'P' to impute by SNP p-value.  |
| 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 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](#)

---

|                 |                                      |
|-----------------|--------------------------------------|
| check_miss_data | <i>Remove SNPs with missing data</i> |
|-----------------|--------------------------------------|

---

**Description**

Remove SNPs with missing data

**Usage**

```
check_miss_data(
  sumstats_dt,
  path,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
```

```
    log_files,  
    drop_na_cols  
  )
```

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.   |
| 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.  |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| nThread        | Number of threads to use for parallel processes.   |
| log_files      | list of log file locations   |
| drop_na_cols   | A character vector of column names to be checked for missing values. Rows with missing values in any of these columns (if present in the dataset) will be dropped. If NULL, all columns will be checked for missing values. Default columns are SNP, chromosome, position, allele 1, allele2, effect columns (frequency, beta, Z-score, standard error, log odds, signed sumstats, odds ratio), p value and N columns. |

Value

list containing sumstats\_dt, the modified summary statistics data table object and a log file list.

---

|                  |   |
|------------------|---|
| check_multi_gwas | <i>Ensure that only one model in GWAS sumstats or only one trait tested</i> |
|------------------|---|

---

Description

Ensure that only one model in GWAS sumstats or only one trait tested

Usage

```
check_multi_gwas(  
  sumstats_dt,  
  path,  
  analysis_trait,  
  ignore_multi_trait,  
  mapping_file  
)
```



**Arguments**

|                |  |
|----------------|--|
| sumstats_dt    | data table obj of the summary statistics file for the GWAS   |
| path           | Filepath for the summary statistics file to be formatted   |
| analysis_trait | If multiple traits were studied, name of the trait for analysis from the GWAS. Default is NULL   |
| 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**

list containing sumstats\_dt, the modified summary statistics data table object

---

|                    |   |
|--------------------|---|
| check_multi_rs_snp | <i>Ensure that SNP ids don't have multiple rs ids on one line</i> |
|--------------------|---|

---

**Description**

Ensure that SNP ids don't have multiple rs ids on one line

**Usage**

```
check_multi_rs_snp(
  sumstats_dt,
  path,
  remove_multi_rs_snp,
  imputation_ind,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

**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.  |
| 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.   |
| 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. |
| 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.  |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| nThread        | Number of threads to use for parallel processes.   |
| log_files      | list of log file locations   |

Value

list containing sumstats\_dt, the modified summary statistics data table object and the log file list.

---

|                 |  |
|-----------------|--|
| check_no_allele | <i>Ensure that A1 &amp; A2 are present, if not can find it with SNP and other allele</i> |
|-----------------|--|

---

Description

More care needs to be taken if one of A1/A2 is present, before imputing the other allele flipping needs to be checked

Usage

```
check_no_allele(  
  sumstats_dt,  
  path,  
  ref_genome,  
  rsids,  
  imputation_ind,  
  allele_flip_check,  
  log_folder_ind,  
  check_save_out,  
  tabix_index,  
  nThread,  
  log_files,  
  bi_allelic_filter,  
  dbSNP  
)
```

**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.  |
| 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. |
| allele_flip_check | Binary Should the allele columns be checked against reference genome to infer if flipping is necessary. Default is TRUE.   |
| 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.  |
| tabix_index       | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| nThread           | Number of threads to use for parallel processes.   |
| log_files         | list of log file locations   |
| bi_allelic_filter | Binary Should non-biallelic SNPs be removed. Default is TRUE.  |
| dbSNP             | version of dbSNP to be used for imputation (144 or 155).   |

**Value**

A list containing two data tables:

- sumstats\_dt: the modified summary statistics data table object
- rsids: snpsById, filtered to SNPs of interest if loaded already. Or else NULL.
- allele\_flip\_check: does the dataset require allele flip check
- log\_files: log file list
- bi\_allelic\_filter: should multi-allelic SNPs be filtered out

---

|                 |  |
|-----------------|--|
| check_no_chr_bp | <i>Ensure that CHR and BP are missing if SNP is present, can find them</i> |
|-----------------|--|

---

## Description

Ensure that CHR and BP are missing if SNP is present, can find them

## Usage

```
check_no_chr_bp(
  sumstats_dt,
  path,
  ref_genome,
  rsids,
  imputation_ind,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files,
  dbSNP
)
```

## 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.  |
| 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. |
| 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.  |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| nThread        | Number of threads to use for parallel processes.   |
| log_files      | list of log file locations   |
| dbSNP          | version of dbSNP to be used for imputation (144 or 155).   |

**Value**

- A list containing two data tables:
- `sumstats_dt` : the modified summary statistics data table object
  - `rsids` : `snpsById`, filtered to SNPs of interest if loaded already. Or else NULL
  - `log_files` : log file list

---

|                 |   |
|-----------------|---|
| check_no_rs_snp | <i>Ensure that SNP appears to be valid RSIDs (starts with rs)</i> |
|-----------------|---|

---

**Description**

Ensure that SNP appears to be valid RSIDs (starts with rs)

**Usage**

```
check_no_rs_snp(  
  sumstats_dt,  
  path,  
  ref_genome,  
  snp_ids_are_rs_ids,  
  indels,  
  imputation_ind,  
  log_folder_ind,  
  check_save_out,  
  tabix_index,  
  nThread,  
  log_files,  
  dbSNP  
)
```

**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 <code>MungeSumstats</code> using the <code>path</code> 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.  |
| 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. |
| 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.  |

|                |   |
|----------------|---|
| 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 where 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. |
| 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.   |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread        | Number of threads to use for parallel processes.  |
| log_files      | list of log file locations  |
| dbSNP          | version of dbSNP to be used for imputation (144 or 155).  |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object and the log file list.

---

|              |  |
|--------------|--|
| check_no_snp | <i>Ensure that SNP is present if not can find it with CHR and BP</i> |
|--------------|--|

---

**Description**

Ensure that SNP is present if not can find it with CHR and BP

**Usage**

```
check_no_snp(  
  sumstats_dt,  
  path,  
  ref_genome,  
  indels,  
  imputation_ind,  
  log_folder_ind,  
  check_save_out,  
  tabix_index,  
  nThread,  
  log_files,  
  dbSNP,  
  verbose = TRUE  
)
```

**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.  |
| 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.  |
| 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. |
| 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.  |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| nThread        | Number of threads to use for parallel processes.   |
| log_files      | list of log file locations   |
| dbSNP          | version of dbSNP to be used for imputation (144 or 155).   |
| verbose        | should messages be printed. Default it TRUE.   |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object and the log files list

---

|               |                              |
|---------------|------------------------------|
| check_numeric | <i>Check numeric columns</i> |
|---------------|------------------------------|

---

**Description**

Checks for any columns that should be numeric, and ensures that they are indeed numeric.

**Usage**

```
check_numeric(sumstats_dt, cols = c("P", "SE", "FRQ", "MAF", "BETA"))
```

**Arguments**

|             |   |
|-------------|---|
| sumstats_dt | Summary stats with column names already standardised by <a href="#">format_sumstats</a> .                                       |
| cols        | Names of columns that should be numeric. If any of these columns are not actually present in sumstats_dt, they will be skipped. |

**Value**

sumstats\_dt

---

|             |   |
|-------------|---|
| check_n_int | <i>Ensure that the N column is all integers</i> |
|-------------|---|

---

**Description**

Ensure that the N column is all integers

**Usage**

check\_n\_int(sumstats\_dt, path, convert\_n\_int, imputation\_ind)

**Arguments**

- sumstats\_dt      data table obj of the summary statistics file for the GWAS
- path              Filepath for the summary statistics file to be formatted
- convert\_n\_int    Binary, if N (the number of samples) is not an integer, should this be rounded? Default is TRUE.
- 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). **Note** these columns will be in the formatted summary statistics returned. Default is FALSE.

**Value**

list containing sumstats\_dt, the modified summary statistics data table object.

---

|             |  |
|-------------|--|
| check_n_num | <i>Ensure all SNPs have N less than X std dev below mean</i> |
|-------------|--|

---

**Description**

In case some SNPs were genotyped by a specialized genotyping array and have substantially more samples than others. These will be removed.



Usage

```
check_n_num(  
  sumstats_dt,  
  path,  
  N_std,  
  N_dropNA = FALSE,  
  log_folder_ind,  
  check_save_out,  
  tabix_index,  
  nThread,  
  log_files  
)
```

Arguments

|                |  |
|----------------|--|
| path           | Filepath for the summary statistics file to be formatted. A dataframe or datat-<br>able of the summary statistics file can also be passed directly to MungeSumstats<br>using the path parameter.   |
| N_std          | numeric The number of standard deviations above the mean a SNP's N is needed<br>to be removed. Default is 5.   |
| N_dropNA       | Drop rows where N is missing.Default is TRUE.  |
| log_folder_ind | Binary Should log files be stored containing all filtered out SNPs (separate file<br>per filter). The data is outputted in the same format specified for the resulting<br>sumstats file. The only exception to this rule is if output is vcf, then log file<br>saved as .tsv.gz. Default is FALSE. |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| nThread        | Number of threads to use for parallel processes.   |
| log_files      | list of log file locations   |

Value

list containing sumstats\_dt, the modified summary statistics data table object and the log file list

---

|                     |  |
|---------------------|--|
| check_on_ref_genome | <i>Ensure all SNPs are on the reference genome</i> |
|---------------------|--|

---

Description

Ensure all SNPs are on the reference genome

**Usage**

```

check_on_ref_genome(
  sumstats_dt,
  path,
  ref_genome,
  on_ref_genome,
  indels = indels,
  rsids,
  imputation_ind,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files,
  dbSNP
)

```

**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.  |
| on_ref_genome  | Binary Should a check take place that all SNPs are on the reference genome by SNP ID. Default 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.  |
| 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. |
| 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.  |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| nThread        | Number of threads to use for parallel processes.   |
| log_files      | list of log file locations   |
| dbSNP          | version of dbSNP to be used for imputation (144 or 155).   |

Value

A list containing two data tables:

- `sumstats_dt` : the modified summary statistics data table object
- `rsids` : `snpsById`, filtered to SNPs of interest if loaded already. Or else NULL
- `log_files` : log file list

---

|                           |   |
|---------------------------|---|
| <code>check_pos_se</code> | <i>Ensure that the standard error (se) is positive for all SNPs Also impute se if missing</i> |
|---------------------------|---|

---

Description

Ensure that the standard error (se) is positive for all SNPs Also impute se if missing

Usage

```
check_pos_se(  
  sumstats_dt,  
  path,  
  pos_se,  
  log_folder_ind,  
  imputation_ind,  
  check_save_out,  
  tabix_index,  
  nThread,  
  log_files,  
  impute_se  
)
```

Arguments

|                             |   |
|-----------------------------|---|
| <code>path</code>           | Filepath for the summary statistics file to be formatted. A dataframe or datatable of the summary statistics file can also be passed directly to <code>MungeSumstats</code> using the <code>path</code> parameter.  |
| <code>pos_se</code>         | Binary Should the standard Error (SE) column be checked to ensure it is greater than 0? Those that are, are removed (if present in <code>sumstats</code> file). Default TRUE.   |
| <code>log_folder_ind</code> | 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 <code>sumstats</code> file. The only exception to this rule is if output is <code>vcf</code> , then log file saved as <code>.tsv.gz</code> . Default is FALSE.  |
| <code>imputation_ind</code> | 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 <code>MungeSumstats</code> 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.   |
| tabix_index | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread     | Number of threads to use for parallel processes.  |
| log_files   | list of log file locations  |
| 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. |

### Value

list containing sumstats\_dt, the modified summary statistics data table object and the log file list

---

|                   |  |
|-------------------|--|
| check_range_p_val | <i>Ensure that the p values are not &gt;1 and if so set to 1</i> |
|-------------------|--|

---

### Description

Ensure that the p values are not >1 and if so set to 1

### Usage

```
check_range_p_val(sumstats_dt, convert_large_p, convert_neg_p, imputation_ind)
```

### Arguments

|                 |  |
|-----------------|--|
| sumstats_dt     | data table obj of the summary statistics file for the GWAS   |
| 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.  |
| 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. |

Value

list containing sumstats\_dt, the modified summary statistics data table object

Source

```
sumstats_dt <- MungeSumstats::formatted_example() sumstats_dt$P[1:3] <- 5 sumstats_dt$P[6:10]
<- -5 sumstats <- check_range_p_val(sumstats_dt = sumstats_dt, convert_large_p = TRUE,
convert_neg_p = TRUE, imputation_ind = TRUE)
```

---

|               |  |
|---------------|--|
| check_row_snp | <i>Ensure all rows have SNPs beginning with rs or SNP, drop those that don't</i> |
|---------------|--|

---

Description

Ensure all rows have SNPs beginning with rs or SNP, drop those that don't

Usage

```
check_row_snp(
  sumstats_dt,
  path,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

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.  |
| 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. |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread        | Number of threads to use for parallel processes.  |
| log_files      | list of log file locations  |

Value

list containing sumstats\_dt, the modified summary statistics data table object and log file list

---

|                 |   |
|-----------------|---|
| check_save_path | <i>Check if save path and log folder is appropriate</i> |
|-----------------|---|

---

### Description

Check if save path and log folder is appropriate

### Usage

```
check_save_path(
    save_path,
    log_folder,
    log_folder_ind,
    tabix_index,
    write_vcf = FALSE,
    verbose = TRUE
)
```

### Arguments

|                |   |
|----------------|---|
| save_path      | File path to save formatted data. Defaults to <code>tempfile(fileext=".tsv.gz")</code> .  |
| log_folder     | Filepath to the directory for the log files and the log of MungeSumstats messages to be stored. Default is a temporary directory. Note the name of the log files (log messages and log outputs) are now the same as the name of the file specified in the save path parameter with the extension <code>'_log_msg.txt'</code> and <code>'_log_output.txt'</code> respectively. |
| 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 <code>.tsv.gz</code> . Default is FALSE.   |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| write_vcf      | Whether to write as VCF (TRUE) or tabular file (FALSE).   |
| verbose        | Print messages.   |

### Value

Corrected save\_path, the file type, the separator, corrected log\_folder, the log file extension.

---

|                  |  |
|------------------|--|
| check_signed_col | <i>Ensure that there is at least one signed column in summary statistics file Impute beta if user requests</i> |
|------------------|--|

---

## Description

Ensure that there is at least one signed column in summary statistics file Impute beta if user requests

## Usage

```
check_signed_col(
  sumstats_dt,
  impute_beta,
  log_folder_ind,
  rsids,
  imputation_ind,
  check_save_out,
  tabix_index,
  log_files,
  nThread
)
```

## Arguments

|                |  |
|----------------|--|
| sumstats_dt    | data table obj of the summary statistics file for the GWAS   |
| 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 (for Z & SE approach) 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:<br>1. log(OR) 2. Z x SE Default value is FALSE.  |
| 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.  |
| 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. |
| tabix_index    | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| log_files      | list of log file locations   |
| nThread        | Number of threads to use for parallel processes.   |

**Value**

null

---

|                   |  |
|-------------------|--|
| check_small_p_val | <i>Ensure that the non-negative p-values are not 5e-324 or lower, if so set to 0</i> |
|-------------------|--|

---

**Description**

Ensure that the non-negative p-values are not 5e-324 or lower, if so set to 0

**Usage**

```
check_small_p_val(sumstats_dt, convert_small_p, imputation_ind)
```

**Arguments**

|                 |  |
|-----------------|--|
| sumstats_dt     | data table obj of the summary statistics file for the GWAS   |
| 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.   |
| 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. |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object

**Source**

```
sumstats_dt <- MungeSumstats::formatted_example() sumstats_dt$P[1:3] <- 5e-324 sumstats_dt$P[6:10]
<- "5e-324" sumstats <- check_small_p_val(sumstats_dt = sumstats_dt, convert_small_p
= TRUE, imputation_ind = TRUE)
```



---

check\_strand\_ambiguous

*Remove SNPs with strand-ambiguous alleles*


---

## Description

Remove SNPs with strand-ambiguous alleles

## Usage

```
check_strand_ambiguous(
  sumstats_dt,
  path,
  ref_genome,
  strand_ambig_filter,
  log_folder_ind,
  check_save_out,
  tabix_index,
  nThread,
  log_files
)
```

## 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.   |
| strand_ambig_filter | Binary Should SNPs with strand-ambiguous alleles be removed. Default is FALSE.  |
| 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. |
| tabix_index         | Index the formatted summary statistics with <b>tabix</b> for fast querying.   |
| nThread             | Number of threads to use for parallel processes.  |
| log_files           | list of log file locations  |

## Value

list containing sumstats\_dt, the modified summary statistics data table object and the log file list

---

|               |                                    |
|---------------|------------------------------------|
| check_tabular | <i>Ensure valid tabular format</i> |
|---------------|------------------------------------|

---

**Description**

Ensure valid tabular format

**Usage**

```
check_tabular(header)
```

**Arguments**

|        |  |
|--------|--|
| header | The summary statistics file for the GWAS |
|--------|--|

**Value**

Whether the file is tabular

---

|                    |   |
|--------------------|---|
| check_two_step_col | <i>Ensure that CHR:BP aren't merged into 1 column</i> |
|--------------------|---|

---

**Description**

Ensure that CHR:BP aren't merged into 1 column

**Usage**

```
check_two_step_col(sumstats_dt, path)
```

**Arguments**

|             |  |
|-------------|--|
| sumstats_dt | data table obj of the summary statistics file for the GWAS |
| path        | Filepath for the summary statistics file to be formatted   |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object

---

|           |  |
|-----------|--|
| check_vcf | <i>Check if the inputted file is in VCF format</i> |
|-----------|--|

---

**Description**

Check if the inputted file is in VCF format

**Usage**

```
check_vcf(header)
```

**Arguments**

|        |   |
|--------|---|
| header | Header of the GWAS summary statistics file. |
|--------|---|

**Value**

Whether the file is vcf or not

---

|                 |   |
|-----------------|---|
| check_vital_col | <i>Ensure that all necessary columns are in the summary statistics file</i> |
|-----------------|---|

---

**Description**

Ensure that all necessary columns are in the summary statistics file

**Usage**

```
check_vital_col(sumstats_dt)
```

**Arguments**

|             |  |
|-------------|--|
| sumstats_dt | data table obj of the summary statistics file for the GWAS |
|-------------|--|

**Value**

null

---

|              |                                 |
|--------------|---------------------------------|
| check_zscore | <i>Check for Z-score column</i> |
|--------------|---------------------------------|

---

## Description

The following ensures that a Z-score column is present. The Z-score formula we used here is a R implementation of the formula used in **LDSC's** `munge_sumstats.py`:

## Usage

```
check_zscore(
    sumstats_dt,
    imputation_ind,
    compute_z = "BETA",
    force_new_z = FALSE,
    standardise_headers = FALSE,
    mapping_file
)
```

## Arguments

|                                  |  |
|----------------------------------|--|
| <code>sumstats_dt</code>         | data table obj of the summary statistics file for the GWAS.  |
| <code>imputation_ind</code>      | 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.   |
| <code>compute_z</code>           | Whether to compute Z-score column. Default is FALSE. This can be computed from Beta and SE with $(\text{Beta}/\text{SE})$ or P ( $Z := \text{sign}(\text{BETA}) * \sqrt{\text{stats}::\text{qchisq}(P, 1, \text{lower} = \text{FALSE})}$ ). <b>Note</b> that imputing the Z-score from P for every SNP will not be perfectly correct and may result in a loss of power. This should only be done as a last resort. Use 'BETA' to impute by BETA/SE and 'P' to impute by SNP p-value. |
| <code>force_new_z</code>         | When a "Z" column already exists, it will be used by default. To override and compute a new Z-score column from P set <code>force_new_z=TRUE</code> .  |
| <code>standardise_headers</code> | Run <code>standardise_sumstats_column_headers_crossplatform</code> first.  |
| <code>mapping_file</code>        | 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 <code>data(sumstatsColHeaders)</code> for default mapping and necessary format.  |

**Details**

```
np.sqrt(chi2.isf(P, 1))
```

The R implementation is adapted from the `GenomicSEM::munge` function, after optimizing for speed using `data.table`:

```
sumstats_dt[,Z:=sign(BETA)*sqrt(stats::qchisq(P,1,lower=FALSE))]
```

*NOTE:* `compute_z` is set to `TRUE` by default to ensure standardisation of the "Z" column (which can be computed differently in different datasets).

**Value**

```
list("sumstats_dt"=sumstats_dt)
```

---

|                   |                                       |
|-------------------|---------------------------------------|
| column_dictionary | <i>Map column names to positions.</i> |
|-------------------|---------------------------------------|

---

**Description**

Useful in situations where you need to specify columns by index instead of name (e.g. awk queries).

**Usage**

```
column_dictionary(file_path)
```

**Arguments**

|           |  |
|-----------|--|
| file_path | Path to full summary stats file (or any really file you want to make a column dictionary for). |
|-----------|--|

**Value**

Named list of column positions.

**Source**

Borrowed function from [echotabix](#).

```
eduAttainOkbayPth <- system.file("extdata", "eduAttainOkbay.txt", package = "MungeSumstats")
) tmp <- tempfile(fileext = ".tsv") file.copy(eduAttainOkbayPth, tmp) cdict <- MungeSumstats::column_dictionary(tmp)
```

---

|               |   |
|---------------|---|
| 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> <li>• "ldsc": N will be computed as effective sample size: <math>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}})))</math>.</li> <li>• "giant": N will be computed as effective sample size: <math>N_{\text{eff}} = 2 / (1/N_{\text{CAS}} + 1/N_{\text{CON}})</math>.</li> <li>• "metal": N will be computed as effective sample size: <math>N_{\text{eff}} = 4 / (1/N_{\text{CAS}} + 1/N_{\text{CON}})</math>.</li> </ul> |

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)
```

---

|                     |  |
|---------------------|--|
| compute_sample_size | <i>Compute (effective) sample size</i> |
|---------------------|--|

---

### Description

Computes sample sum (as new column "N") or effective sample size (ESS) (as new column "Neff"). Computing ESS is important as it takes into account the proportion of cases to controls (i.e. class imbalance) so as not to overestimate your statistical power.

### Usage

```
compute_sample_size(
  sumstats_dt,
  method = c("ldsc", "giant", "metal", "sum"),
  force_new = FALSE,
  append_method_name = FALSE
)
```

### Arguments

sumstats\_dt      Summary statistics data.table.

method      Method for computing (effective) sample size.

- "ldsc" :  

$$Neff = (N_{CAS} + N_{CON}) * (N_{CAS} / (N_{CAS} + N_{CON})) / mean((N_{CAS} / (N_{CAS} + N_{CON}))[(N_{CAS} + N_{CON}) == max(N_{CAS} + N_{CON})])$$
[bulik/ldsc GitHub Issue](#) [bulik/ldsc GitHub code](#)
- "giant" :  

$$Neff = 2 / (1/N_{CAS} + 1/N_{CON})$$
[Winkler et al. 2014, Nature](#)

|                    |  |
|--------------------|--|
|                    | <ul style="list-style-type: none"> <li>• "metal" :<br/> <math display="block">Neff = 4 / (1/N_{CAS} + 1/N_{CON})</math> <a href="#">Willer et al. 2010, Bioinformatics</a> </li> <li>• "sum" :<br/> <math display="block">N = N_{CAS} + N_{CON}</math> Simple summation of cases and controls that does not account for class imbalance. </li> <li>• "&lt;integer&gt;" :<br/> <math display="block">N = \text{integer}</math> If method is a positive integer, it will be used as N for every row. </li> </ul> |
| force_new          | If "Neff" (or "N") already exists in sumstats_dt, replace it with the recomputed version.  |
| append_method_name | should Neff column have an indicator to explain the method that makes it., Default is FALSE unless multiple methods are passed   |

### Details

There are many different formulas for calculating ESS, but LDSC is probably the best method available here, as it doesn't assume that the proportion of controls:cases is 2:1 (as in GIANT) or 4:1 (as in METAL).

### Value

A data.table with a new column "Neff" or "N"

---

compute\_sample\_size\_n *Add user supplied sample size*

---

### Description

Add user supplied sample size

### Usage

```
compute_sample_size_n(sumstats_dt, method, force_new = FALSE)
```

### Arguments

|             |  |
|-------------|--|
| sumstats_dt | Summary statistics data.table.   |
| method      | Method for computing (effective) sample size. <ul style="list-style-type: none"> <li>• "ldsc" :<br/> <math display="block">Neff = (N_{CAS} + N_{CON}) * (N_{CAS} / (N_{CAS} + N_{CON})) / mean((N_{CAS} / (N_{CAS} + N_{CON}))[(N_{CAS} + N_{CON}) == max(N_{CAS} + N_{CON})])</math> <a href="#">bulik/ldsc GitHub Issue</a> <a href="#">bulik/ldsc GitHub code</a> </li> </ul> |



- "giant" :  
 $N_{eff} = 2/(1/N_{CAS} + 1/N_{CON})$   
[Winkler et al. 2014, Nature](#)
- "metal" :  
 $N_{eff} = 4/(1/N_{CAS} + 1/N_{CON})$   
[Willer et al. 2010, Bioinformatics](#)
- "sum" :  
 $N = N_{CAS} + N_{CON}$   
 Simple summation of cases and controls that does not account for class imbalance.
- "<integer>" :  
 $N = \text{integer}$   
 If method is a positive integer, it will be used as N for every row.

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

### Value

No return

---

compute\_sample\_size\_neff  
*Compute Neff/N*

---

### Description

Compute Neff/N

### Usage

```
compute_sample_size_neff(
  sumstats_dt,
  method,
  force_new = FALSE,
  append_method_name = FALSE
)
```

### Arguments

sumstats\_dt      Summary statistics data.table.

method            Method for computing (effective) sample size.

- "ldsc" :  
 $N_{eff} = (N_{CAS} + N_{CON}) * (N_{CAS} / (N_{CAS} + N_{CON})) / \text{mean}((N_{CAS} / (N_{CAS} + N_{CON}))[(N_{CAS} + N_{CON}) == \max(N_{CAS} + N_{CON})])$   
[bulik/ldsc GitHub Issue](#) [bulik/ldsc GitHub code](#)

- "giant" :  
 $N_{eff} = 2/(1/N_{CAS} + 1/N_{CON})$   
Winkler et al. 2014, Nature
  - "metal" :  
 $N_{eff} = 4/(1/N_{CAS} + 1/N_{CON})$   
Willer et al. 2010, Bioinformatics
  - "sum" :  
 $N = N_{CAS} + N_{CON}$   
Simple summation of cases and controls that does not account for class imbalance.
  - "<integer>" :  
 $N = \text{integer}$   
If method is a positive integer, it will be used as N for every row.
- force\_new      If "Neff" (or "N") already exists in sumstats\_dt, replace it with the recomputed version.
- append\_method\_name      should Neff column have an indicator to explain the method that makes it., Default is FALSE unless multiple methods are passed

**Value**

No return

---

|                  |  |
|------------------|--|
| convert_sumstats | <i>Convert summary statistics to desired object type</i> |
|------------------|--|

---

**Description**

Convert summary statistics to desired object type

**Usage**

```
convert_sumstats(  
  sumstats_dt,  
  return_format = c("data.table", "vranges", "granges")  
)
```

**Arguments**

return\_format      Object type to convert to; "data.table", "GenomicRanges" or "VRanges"(default is "data.table").

**Value**

Summary statistics in the converted format

---

|          |                                |
|----------|--------------------------------|
| DF_to_dt | <i>DataFrame to data.table</i> |
|----------|--------------------------------|

---

**Description**

Efficiently convert [DataFrame](#) to [data.table](#).

**Usage**

```
DF_to_dt(DF)
```

**Arguments**

DF [DataFrame](#) object.

**Value**

VCF data in [data.table](#) format.

**Source**

[Solution from Bioc forum](#)

---

|            |                           |
|------------|---------------------------|
| downloader | <i>downloader wrapper</i> |
|------------|---------------------------|

---

**Description**

R wrapper for [axel](#) (multi-threaded) and [download.file](#) (single-threaded) download functions.

**Usage**

```
downloader(  
  input_url,  
  output_path,  
  download_method = "axel",  
  background = FALSE,  
  force_overwrite = FALSE,  
  quiet = TRUE,  
  show_progress = TRUE,  
  continue = TRUE,  
  nThread = 1,  
  alternate = TRUE,  
  check_certificates = TRUE,  
  timeout = 10 * 60  
)
```

**Arguments**

|                    |  |
|--------------------|--|
| input_url          | input_url.   |
| output_path        | output_path.   |
| download_method    | "axel" (multi-threaded) or "download.file" (single-threaded) .                                     |
| background         | Run in background  |
| force_overwrite    | Overwrite existing file.   |
| quiet              | Run quietly.   |
| show_progress      | show_progress.   |
| continue           | continue.  |
| nThread            | Number of threads to parallelize over.   |
| alternate          | alternate,   |
| check_certificates | check_certificates   |
| timeout            | How many seconds before giving up on download. Passed to download.file.<br>Default: 10*60 (10min). |

**Value**

Local path to downloaded file.

**Source**

Suggestion to avoid 'proc\$get\_built\_file() : Build process failed'

**See Also**

Other downloaders: [axel\(\)](#)

---

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 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.   |
| 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 download.file. 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)
}
```

---

|                     |                               |
|---------------------|-------------------------------|
| drop_duplicate_cols | <i>Drop duplicate columns</i> |
|---------------------|-------------------------------|

---

**Description**

Drop columns with identical names (if any exist) within a data.table.

**Usage**

```
drop_duplicate_cols(dt)
```

**Arguments**

|    |            |
|----|------------|
| dt | data.table |
|----|------------|

**Value**

Null output

---

|                     |                            |
|---------------------|----------------------------|
| drop_duplicate_rows | <i>Drop duplicate rows</i> |
|---------------------|----------------------------|

---

**Description**

Drop rows with duplicate values across all columns.

**Usage**

```
drop_duplicate_rows(dt, verbose = TRUE)
```

**Arguments**

|         |                 |
|---------|-----------------|
| dt      | data.table      |
| verbose | Print messages. |

**Value**

Filtered dt.

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. <code>c("prot-a-664", "ieu-b-4760")</code> ).              |
| traits          | List of traits (e.g. <code>c("parkinson", "Alzheimer")</code> ).                             |
| years           | List of years (e.g. <code>seq(2015, 2021)</code> or <code>c(2010, 2012, 2021)</code> ).      |
| consortia       | List of consortia (e.g. <code>c("MRC-IEU", "Neale Lab")</code> ).                            |
| authors         | List of authors (e.g. <code>c("Elsworth", "Kunkle", "Neale")</code> ).                       |
| populations     | List of populations (e.g. <code>c("European", "Asian")</code> ).                             |
| categories      | List of categories (e.g. <code>c("Binary", "Continuous", "Disease", "Risk factor")</code> ). |
| subcategories   | List of categories (e.g. <code>c("neurological", "Immune", "cardio")</code> ).               |
| builds          | List of genome builds (e.g. <code>c("hg19", "grch37")</code> ).                              |
| pmids           | List of PubMed ID (exact matches only) (e.g. <code>c(29875488, 30305740, 28240269)</code> ). |
| 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 amd 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,
  chain_source = "ensembl",
  local_chain = 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,
  es_is_beta = TRUE,
  impute_se = FALSE,
```

```

analysis_trait = NULL,
ignore_multi_trait = FALSE,
INFO_filter = 0.9,
FRQ_filter = 0,
pos_se = TRUE,
effect_columns_nonzero = FALSE,
N_std = 5,
N_dropNA = TRUE,
chr_style = "Ensembl",
rmv_chr = c("X", "Y", "MT"),
on_ref_genome = TRUE,
infer_eff_direction = 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,
flip_frq_as_biallelic = FALSE,
snp_ids_are_rs_ids = TRUE,
remove_multi_rs_snp = FALSE,
frq_is_maf = TRUE,
indels = TRUE,
drop_indels = FALSE,
drop_na_cols = c("SNP", "CHR", "BP", "A1", "A2", "FRQ", "BETA", "Z", "OR", "LOG_ODDS",
  "SIGNED_SUMSTAT", "SE", "P", "N"),
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,
rmv_chrPrefix = NULL
)

```

**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).   |
| chain_source       | source of the chain file to use in liftover, if converting genome build ("ucsc" or "ensembl"). Note that the UCSC chain files require a license for commercial use. The Ensembl chain is used by default ("ensembl").  |
| local_chain        | Path to local chain file to use instead of downloading. Default of NULL i.e. no local file to use. NOTE if passing a local chain file make sure to specify the path to convert from and to the correct build like GRCh37 to GRCh38. We can not sense check this for local files. The chain file can be submitted as a gz file (as downloaded from source) or unzipped.   |
| 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. Default is FALSE. This can be computed from Beta and SE with $(\text{Beta}/\text{SE})$ or $P(Z = \text{sign}(\text{BETA}) * \sqrt{\text{stats::qchisq}(P, 1, \text{lower} = \text{FALSE})})$ . <b>Note</b> that imputing the Z-score from P for every SNP will not be perfectly correct and may result in a loss of power. This should only be done as a last resort. Use 'BETA' to impute by BETA/SE and 'P' to impute by SNP p-value.   |
| 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 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 (for Z & SE approach) 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:<br>1. log(OR) 2. Z x SE Default value is FALSE. |
| es_is_beta             | Binary, whether to map ES to BETA. We take BETA to be any BETA-like value (including Effect Size). If this is not the case for your sumstats, change this to FALSE. Default is TRUE.  |
| 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:<br>1. BETA / Z 2. abs(BETA/ 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.   |
| ignore_multi_trait     | If you have multiple traits (p-values) in the study but you want to ignore these and instead use a standard named p-value, set to TRUE. By default is FALSE which will check for multi-traits.  |
| 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.   |
| chr_style              | Chromosome naming style to use in the formatted summary statistics file ("NCBI", "UCSC", "dbSNP", or "Ensembl"). The NCBI and Ensembl styles both code chromosomes as 1-22, X, Y, MT; the UCSC style is chr1-chr22, chrX, chrY, chrM; and the dbSNP style is ch1-ch22, chX, chY, chMT. Default is Ensembl.  |
| rmv_chr                | Chromosomes to exclude from the formatted summary statistics file. Use NULL if no filtering is necessary. Default is c("X", "Y", "MT") which removes all non-autosomal SNPs.  |
| on_ref_genome          | Binary Should a check take place that all SNPs are on the reference genome by SNP ID. Default is TRUE.  |

|                       |   |
|-----------------------|---|
| infer_eff_direction   | Binary Should a check take place to ensure the alleles match the effect direction? 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.   |
| flip_frq_as_biallelic | Binary Should non-bi-allelic SNPs frequency values be flipped as 1-p despite there being other alternative alleles? Default is FALSE but if set to TRUE, this allows non-bi-allelic SNPs to be kept despite needing flipping.   |
| 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.   |
| drop_indels           | Binary, should any indels found in the sumstats be dropped? These can not be checked against a reference dataset and will have the same RS ID and position as SNPs which can affect downstream analysis. Default is False.  |

|                        |  |
|------------------------|--|
| drop_na_cols           | A character vector of column names to be checked for missing values. Rows with missing values in any of these columns (if present in the dataset) will be dropped. If NULL, all columns will be checked for missing values. Default columns are SNP, chromosome, position, allele 1, allele2, effect columns (frequency, beta, Z-score, standard error, log odds, signed sumstats, odds ratio), p value and N columns.   |
| 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 <code>tempfile(fileext=".tsv.gz")</code> .   |
| write_vcf              | Whether to write as VCF (TRUE) or tabular file (FALSE).  |
| tabix_index            | Index the formatted summary statistics with <b>tabix</b> 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. <b>NOTE</b> - If LDSC format is used, the naming convention of A1 as the reference (genome build) allele and A2 as the effect allele will be reversed to match LDSC (A1 will now be the effect allele). See more info on this <a href="#">here</a> . Note that any effect columns (e.g. Z) will be in relation to A1 now instead of A2. |
| 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. Note the name of the log files (log messages and log outputs) are now the same as the name of the file specified in the save path parameter with the extension '_log_msg.txt' and '_log_output.txt' respectively.  |
| 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. |
| rmv_chrPrefix | Is now deprecated, do. not use. Use chr_style instead - chr_style = 'Ensembl' will give the same result as rmv_chrPrefix=TRUE used to give.  |

### 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_access_token | <i>Get access token for OAuth2 access to MR Base</i> |
|------------------|--|

---

**Description**

Get access token for OAuth2 access to MR Base

**Usage**

get\_access\_token()

**Value**

access token string

---

|                |   |
|----------------|---|
| get_chain_file | <i>Download chain file for liftover</i> |
|----------------|---|

---

**Description**

Download chain file for liftover

**Usage**

```
get_chain_file(  
  from = c("hg38", "hg19"),  
  to = c("hg19", "hg38"),  
  chain_source = c("ucsc", "ensembl"),  
  save_dir = tempdir(),  
  verbose = TRUE  
)
```

**Arguments**

|              |  |
|--------------|--|
| from         | genome build converted from ("hg38", "hg19")               |
| to           | genome build converted to ("hg19", "hg38")                 |
| chain_source | chain file source used ("ucsc" as default, or "ensembl")   |
| save_dir     | where is the chain file saved? Default is a temp directory |
| verbose      | extra messages printed? Default is TRUE                    |

**Value**

loaded chain file for liftover



**Source**

UCSC chain files

Ensembl chain files

---

```
get_eff_frq_allele_combns
```

*Get combinations of uncorrected allele and effect (and frq) columns*

---

**Description**

Get combinations of uncorrected allele and effect (and frq) columns

**Usage**

```
get_eff_frq_allele_combns(
  mapping_file = sumstatsColHeaders,
  eff_frq_cols = c("BETA", "OR", "LOG_ODDS", "SIGNED_SUMSTAT", "Z", "FRQ")
)
```

**Arguments**

|              |  |
|--------------|--|
| 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. |
| eff_frq_cols | Corrected effect or frequency column names found in a sumstats. Default of BETA, OR, LOG_ODDS, SIGNED_SUMSTAT, Z and FRQ.  |

**Value**

datatable containing uncorrected and corrected combinations

---

|                  |   |
|------------------|---|
| get_genome_build | <i>Infers the genome build of the summary statistics file (GRCh37 or GRCh38) from the data. Uses SNP (RSID) &amp; CHR &amp; BP to get genome build.</i> |
|------------------|---|

---

**Description**

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

**Usage**

```

get_genome_build(
  sumstats,
  nThread = 1,
  sampled_snps = 10000,
  standardise_headers = TRUE,
  mapping_file = sumstatsColHeaders,
  dbSNP = 155,
  header_only = FALSE,
  allele_match_ref = FALSE,
  ref_genome = NULL,
  chr_filt = NULL
)

```

**Arguments**

|                     |   |
|---------------------|---|
| sumstats            | data table/data frame obj of the summary statistics file for the GWAS ,or file path to summary statistics file.   |
| nThread             | Number of threads to use for parallel processes.  |
| sampled_snps        | Downsample the number of SNPs used when inferring genome build to save time.  |
| standardise_headers | Run standardise_sumstats_column_headers_crossplatform.  |
| mapping_file        | <b>MungeSumstats</b> 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. |
| dbSNP               | version of dbSNP to be used (144 or 155). Default is 155.   |
| 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.   |
| allele_match_ref    | Instead of returning the genome_build this will return the propotion of matches to each genome build for each allele (A1,A2).   |
| 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.   |
| chr_filt            | Internal for testing - filter reference genomes and sumstats to specific chromosomes for testing. Pass a list of chroms in format: c("1","2"). Default is NULL i.e. no filtering  |

**Value**

ref\_genome the genome build of the data

---

|                   |                            |
|-------------------|----------------------------|
| get_genome_builds | <i>Infer genome builds</i> |
|-------------------|----------------------------|

---

## 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,
  chr_filt = NULL
)
```

## 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.  |
| chr_filt         | Internal for testing - filter reference genomes and sumstats to specific chromosomes for testing. Pass a list of chroms in format: c("1","2"). Default is NULL i.e. no filtering            |

## 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)
}
```

---

|                   |   |
|-------------------|---|
| get_query_content | <i>Parse out json response from httr object</i> |
|-------------------|---|

---

**Description**

Parse out json response from httr object

**Usage**

```
get_query_content(response)
```

**Arguments**

|          |                  |
|----------|------------------|
| response | Output from httr |
|----------|------------------|

**Value**

Parsed json output from query, often in form of data frame. If status code is not successful then return the actual response.

---

`get_unique_name_log_file`

*Simple function to ensure the new entry name to a list doesn't have the same name as another entry*

---

**Description**

Simple function to ensure the new entry name to a list doesn't have the same name as another entry

**Usage**

```
get_unique_name_log_file(name, log_files)
```

**Arguments**

|                        |                             |
|------------------------|-----------------------------|
| <code>name</code>      | proposed name for the entry |
| <code>log_files</code> | list of log file locations  |

**Value**

a unique name (character)

---

`get_vcf_sample_ids`      *Get VCF sample ID(s)*

---

**Description**

Get VCF sample ID(s)

**Usage**

```
get_vcf_sample_ids(path)
```

**Arguments**

|                   |  |
|-------------------|--|
| <code>path</code> | 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. |
|-------------------|--|

**Value**

`sample_id`

---

|               |                                    |
|---------------|------------------------------------|
| granges_to_dt | <i>GenomicRanges to data.table</i> |
|---------------|------------------------------------|

---

**Description**

Convert a [GRanges](#) into a [data.table](#).

**Usage**

```
granges_to_dt(gr)
```

**Arguments**

`gr`                    A [GRanges](#) object.

**Value**

A [data.table](#) object.

**Source**

Code adapted from [GenomicDistributions](#).

---

|          |   |
|----------|---|
| gwasinfo | <i>Get list of studies with available GWAS summary statistics through API</i> |
|----------|---|

---

**Description**

Get list of studies with available GWAS summary statistics through API

**Usage**

```
gwasinfo(id = NULL, access_token = check_access_token())
```

**Arguments**

`id`                    List of MR-Base IDs to retrieve. If NULL (default) retrieves all available datasets

`access_token`        Google OAuth2 access token. Used to authenticate level of access to data

**Value**

Dataframe of details for all available studies

---

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/>  
`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/> 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/>  
`utils::download.file('ftp://hgdownload.cse.ucsc.edu/goldenPath/hg38/liftOver/hg38ToHg19.over.chain.`

---

|           |  |
|-----------|--|
| 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, downlaoded 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**

|                                  |  |
|----------------------------------|--|
| <code>ids</code>                 | List of Open GWAS study IDs (e.g. <code>c("prot-a-664", "ieu-b-4760")</code> ).  |
| <code>vcf_dir</code>             | 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> ). |
| <code>vcf_download</code>        | Download the original VCF from Open GWAS.  |
| <code>save_dir</code>            | Directory to save formatted summary statistics in.   |
| <code>write_vcf</code>           | Whether to write as VCF (TRUE) or tabular file (FALSE).  |
| <code>download_method</code>     | "axel" (multi-threaded) or "download.file" (single-threaded) .   |
| <code>quiet</code>               | Run quietly.   |
| <code>force_new</code>           | 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.   |
| <code>force_new_vcf</code>       | Overwrite a previously downloaded VCF with the same path name.   |
| <code>nThread</code>             | Number of threads to use for parallel processes.   |
| <code>parallel_across_ids</code> | If <code>parallel_across_ids=TRUE</code> and <code>nThread&gt;1</code> , then each ID in <code>ids</code> will be processed in parallel.   |
| <code>...</code>                 | Arguments passed on to <a href="#">format_sumstats</a>   |
|                                  | <code>path</code> Filepath for the summary statistics file to be formatted. A dataframe or datatable of the summary statistics file can also be passed directly to <code>MungeSumstats</code> using the <code>path</code> parameter.   |
|                                  | <code>ref_genome</code> 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).

`chain_source` source of the chain file to use in liftover, if converting genome build ("ucsc" or "ensembl"). Note that the UCSC chain files require a license for commercial use. The Ensembl chain is used by default ("ensembl").

`local_chain` Path to local chain file to use instead of downloading. Default of NULL i.e. no local file to use. NOTE if passing a local chain file make sure to specify the path to convert from and to the correct build like GRCh37 to GRCh38. We can not sense check this for local files. The chain file can be submitted as a gz file (as downloaded from source) or unzipped.

`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. Default is FALSE. This can be computed from Beta and SE with (Beta/SE) or P ( $Z := \text{sign}(\text{BETA}) * \sqrt{\text{stats::qchisq}(P, 1, \text{lower} = \text{FALSE})}$ ). **Note** that imputing the Z-score from P for every SNP will not be perfectly correct and may result in a loss of power. This should only be done as a last resort. Use 'BETA' to impute by BETA/SE and 'P' to impute by SNP p-value.

`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 (for Z & SE approach) 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(OR)
  2. Z x SE
- Default value is FALSE.

`es_is_beta` Binary, whether to map ES to BETA. We take BETA to be any BETA-like value (including Effect Size). If this is not the case for your

sumstats, change this to FALSE. Default is TRUE.

**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. 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.

**ignore\_multi\_trait** If you have multiple traits (p-values) in the study but you want to ignore these and instead use a standard named p-value, set to TRUE. By default is FALSE which will check for multi-traits.

**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.

**chr\_style** Chromosome naming style to use in the formatted summary statistics file ("NCBI", "UCSC", "dbSNP", or "Ensembl"). The NCBI and Ensembl styles both code chromosomes as 1-22, X, Y, MT; the UCSC style is chr1-chr22, chrX, chrY, chrM; and the dbSNP style is ch1-ch22, chX, chY, chMT. Default is Ensembl.

**rmv\_chrPrefix** Is now deprecated, do not use. Use chr\_style instead - chr\_style = 'Ensembl' will give the same result as rmv\_chrPrefix=TRUE used to give.

**rmv\_chr** Chromosomes to exclude from the formatted summary statistics file. Use NULL if no filtering is necessary. Default is c("X", "Y", "MT") which removes all non-autosomal SNPs.

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

**infer\_eff\_direction** Binary Should a check take place to ensure the alleles match the effect direction? 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.

`flip_frq_as_biallelic` Binary Should non-bi-allelic SNPs frequency values be flipped as 1-p despite there being other alternative alleles? Default is FALSE but if set to TRUE, this allows non-bi-allelic SNPs to be kept despite needing flipping.

`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.

`drop_indels` Binary, should any indels found in the sumstats be dropped? These can not be checked against a reference dataset and will have the same RS ID and position as SNPs which can affect downstream analysis. Default is False.

`drop_na_cols` A character vector of column names to be checked for missing values. Rows with missing values in any of these columns (if present in the dataset) will be dropped. If NULL, all columns will be checked for missing values. Default columns are SNP, chromosome, position, allele 1, allele2, effect columns (frequency, beta, Z-score, standard error, log odds, signed sumstats, odds ratio), p value and N columns.

`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`. **NOTE** - If LDSC format is used, the naming convention of A1 as the reference (genome build) allele and A2 as the effect allele will be reversed to match LDSC (A1 will now be the effect allele). See more info on this [here](#). Note that any effect columns (e.g. Z) will be in relation to A1 now instead of A2.

**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. Note the name of the log files (log messages and log outputs) are now the same as the name of the file specified in the save path parameter with the extension `'_log_msg.txt'` and `'_log_output.txt'` respectively.

**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)
```

---

|           |                              |
|-----------|------------------------------|
| index_vcf | <i>Tabix-index file: VCF</i> |
|-----------|------------------------------|

---

**Description**

Convert summary stats file to tabix format

**Usage**

```
index_vcf(path, verbose = TRUE)
```

**Arguments**

- path            Path to VCF.
- verbose        Print messages.

**Value**

Path to tabix-indexed tabular file

**Source**

Borrowed function from [echotabix](#).

**See Also**

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

## Examples

```
eduAttainOkbayPth <- system.file("extdata", "eduAttainOkbay.txt",
                                package = "MungeSumstats")
sumstats_dt <- data.table::fread(eduAttainOkbayPth, nThread = 1)
sumstats_dt <-
MungeSumstats:::standardise_sumstats_column_headers_crossplatform(
  sumstats_dt = sumstats_dt)$sumstats_dt
sumstats_dt <- MungeSumstats:::sort_coords(sumstats_dt = sumstats_dt)
path <- tempfile(fileext = ".tsv")
MungeSumstats::write_sumstats(sumstats_dt = sumstats_dt, save_path = path)

indexed_file <- MungeSumstats::index_tabular(path = path)
```

---

|                     |   |
|---------------------|---|
| infer_effect_column | <i>Infer if effect relates to a1 or A2 if ambiguously named</i> |
|---------------------|---|

---

## Description

Three checks are made to infer which allele the effect/frequency information relates to if they are ambiguous (named A1 and A2 or equivalent):

1. Check if ambiguous naming conventions are used (i.e. allele 1 and 2 or equivalent). If not exit, otherwise continue to next checks. This can be checked by using the mapping file and splitting A1/A2 mappings by those that contain 1 or 2 (ambiguous) or doesn't contain 1 or 2 e.g. effect, tested (unambiguous so fine for MSS to handle as is).
2. Look for effect column/frequency column where the A1/A2 explicitly mentioned, if found then we know the direction and should update A1/A2 naming so A2 is the effect column. We can look for such columns by getting every combination of A1/A2 naming and effect/frq naming.
3. If note found in 2, a final check should be against the reference genome, whichever of A1 and A2 has more of a match with the reference genome should be taken as **not** the effect allele. There is an assumption in this but is still better than guessing the ambiguous allele naming.

## Usage

```
infer_effect_column(
  sumstats_dt,
  dbSNP = 155,
  sampled_snps = 10000,
  mapping_file = sumstatsColHeaders,
  nThread = nThread,
  ref_genome = NULL,
  on_ref_genome = TRUE,
  infer_eff_direction = TRUE,
  return_list = TRUE
)
```



**Arguments**

|                     |  |
|---------------------|--|
| sumstats_dt         | data table obj of the summary statistics file for the GWAS.  |
| dbSNP               | version of dbSNP to be used for imputation (144 or 155).   |
| sampled_snps        | Downsample the number of SNPs used when inferring genome build to save time.   |
| 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. |
| nThread             | Number of threads to use for parallel processes.   |
| 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.  |
| on_ref_genome       | Binary Should a check take place that all SNPs are on the reference genome by SNP ID. Default is TRUE.   |
| infer_eff_direction | Binary Should a check take place to ensure the alleles match the effect direction? Default is TRUE.  |
| return_list         | Return the sumstats_dt within a named list (default: TRUE).  |

**Value**

list containing sumstats\_dt, the modified summary statistics data table object

**Examples**

```
sumstats <- MungeSumstats::formatted_example()
#for speed, don't run on_ref_genome part of check (on_ref_genome = FALSE)
sumstats_dt2<-infer_effect_column(sumstats_dt=sumstats,on_ref_genome = FALSE)
```

---

is\_tabix

*Is tabix*


---

**Description**

Is a file bgz-compressed and tabix-indexed.

**Usage**

```
is_tabix(path)
```

**Arguments**

path                      Path to file.

**Value**

logical: whether the file is tabix-indexed or not.  
logical

---

|            |  |
|------------|--|
| legacy_ids | <i>Convert current IDs to legacy IDs</i> |
|------------|--|

---

**Description**

Convert current IDs to legacy IDs

**Usage**

legacy\_ids(x)

**Arguments**

x                      Vector of ids

**Value**

vector of back compatible ids

---

|          |                              |
|----------|------------------------------|
| 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,  
  chain_source = "ensembl",  
  imputation_ind = TRUE,  
  chrom_col = "CHR",  
  start_col = "BP",  
  end_col = start_col,  
  as_granges = FALSE,  
  style = "NCBI",  
  local_chain = NULL,  
  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.  |
| chain_source       | chain file source used ("ucsc" as default, or "ensembl")   |
| 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"). 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").   |
| local_chain        | Path to local chain file to use instead of downloading. Default of NULL i.e. no local file to use. NOTE if passing a local chain file make sure to specify the path to convert from and to the correct build like GRCh37 to GRCh38. We can not sense check this for local files. The chain file can be submitted as a gz file (as downloaded from source) or unzipped.   |
| verbose            | Print messages.  |

**Value**

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

**Source**

[liftOver](#)

[UCSC chain files](#)

[Ensembl 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 | <i>Load the reference genome data for SNPs of interest</i> |
|----------------------|--|

---

## Description

Load the reference genome data for SNPs of interest

## Usage

```
load_ref_genome_data(  
  snps,  
  ref_genome,  
  dbSNP = c(144, 155),  
  msg = NULL,  
  chr_filt = 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   |
| chr_filt   | Internal for testing - filter reference genomes and sumstats to specific chromosomes for testing. Pass a list of chroms in format: c("1","2"). Default is NULL i.e. no filtering. |

## 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)
```

---

|              |                          |
|--------------|--------------------------|
| logs_example | <i>Example logs file</i> |
|--------------|--------------------------|

---

### Description

Example logs file produced by [format\\_sumstats](#).

### Usage

```
logs_example(read = FALSE)
```

### Arguments

|      |  |
|------|--|
| read | Whether to read the logs file into memory. |
|------|--|

**Value**

Path to logs file.

**Source**

```
eduAttainOkbayPth <- system.file("extdata", "eduAttainOkbay.txt", package = "MungeSumstats")
sumstats_dt <- data.table::fread(eduAttainOkbayPth) ##### Introduce values that need
to be fixed ##### sumstats_dt$Pval[10:15] <- 5 sumstats_dt$Pval[20:22] <- -5 sumstats_dt$Pval[23:25]
<- "5e-324" ss_path <- tempfile() data.table::fwrite(sumstats_dt, ss_path) log_folder
<- tempdir() reformatted <- MungeSumstats::format_sumstats( path = ss_path, ref_genome
= "GRCh37", log_folder = log_folder, log_mungesumstats_msgs = TRUE, log_folder_ind =
TRUE,) file.copy(reformatted$log_files$MungeSumstats_log_msg, "inst/extdata",overwrite
= TRUE)
```

---

|                   |                                 |
|-------------------|---------------------------------|
| make_allele_upper | Ensure A1 and A2 are upper case |
|-------------------|---------------------------------|

---

**Description**

Ensure A1 and A2 are upper case

**Usage**

```
make_allele_upper(sumstats_dt, log_files)
```

**Arguments**

log\_files            list of log file locations

**Value**

list containing sumstats\_dt, the modified summary statistics data table object and the log file list

---

|           |                |
|-----------|----------------|
| messenger | Print messages |
|-----------|----------------|

---

**Description**

Print messages with option to silence.

**Usage**

```
messenger(..., v = TRUE)
```

**Arguments**

- ... Message input.
- v Whether to print messages.

**Value**

Null output.

---

|                  |   |
|------------------|---|
| message_parallel | <i>Send messages to console even from within parallel processes</i> |
|------------------|---|

---

**Description**

Send messages to console even from within parallel processes

**Usage**

message\_parallel(...)

**Value**

A message

---

|                     |  |
|---------------------|--|
| parse_dropped_chrom | <i>Parse number of SNPs dropped due to being on chrom X, Y or MT</i> |
|---------------------|--|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

parse\_dropped\_chrom(1)

**Arguments**

- 1 Lines of text from log file.

**Value**

Numeric



---

parse\_dropped\_duplicates

*Parse number of SNPs dropped due to being duplicates*

---

### Description

Support function for [parse\\_logs](#).

### Usage

```
parse_dropped_duplicates(1)
```

### Arguments

1                      Lines of text from log file.

### Value

Numeric

---

parse\_dropped\_INFO

*Parse number of SNPs dropped due to being below the INFO threshold*

---

### Description

Support function for [parse\\_logs](#).

### Usage

```
parse_dropped_INFO(1)
```

### Arguments

1                      Lines of text from log file.

### Value

Numeric

---

|                       |   |
|-----------------------|---|
| parse_dropped_nonA1A2 | <i>Parse number of SNPs dropped due to not matching the ref genome A1 or A2</i> |
|-----------------------|---|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

parse\_dropped\_nonA1A2(1)

**Arguments**

1                      Lines of text from log file.

**Value**

Numeric

---

|                            |   |
|----------------------------|---|
| parse_dropped_nonBiallelic | <i>Parse number of SNPs dropped due to not being bi-allelic</i> |
|----------------------------|---|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

parse\_dropped\_nonBiallelic(1)

**Arguments**

1                      Lines of text from log file.

**Value**

Numeric

---

|                      |  |
|----------------------|--|
| parse_dropped_nonRef | <i>Parse number of SNPs dropped due to being in the ref genome</i> |
|----------------------|--|

---

### Description

Support function for [parse\\_logs](#).

### Usage

```
parse_dropped_nonRef(1)
```

### Arguments

|   |                              |
|---|------------------------------|
| 1 | Lines of text from log file. |
|---|------------------------------|

### Value

Numeric

---

|               |  |
|---------------|--|
| parse_flipped | <i>Parse number of SNPs flipped to align with the ref genome</i> |
|---------------|--|

---

### Description

Support function for [parse\\_logs](#).

### Usage

```
parse_flipped(1)
```

### Arguments

|   |                              |
|---|------------------------------|
| 1 | Lines of text from log file. |
|---|------------------------------|

### Value

Numeric

---

|                    |  |
|--------------------|--|
| parse_genome_build | <i>Genome build inferred from the summary statistics</i> |
|--------------------|--|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

parse\_genome\_build(1)

**Arguments**

1                      Lines of text from log file.

**Value**

Character

---

|                  |   |
|------------------|---|
| parse_idStandard | <i>Standardised IEU MRC OpenGWAS ID</i> |
|------------------|---|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

parse\_idStandard(1)

**Arguments**

1                      Lines of text from log file.

**Value**

Character

---

|            |                                  |
|------------|----------------------------------|
| 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**

|                       |   |
|-----------------------|---|
| <code>save_dir</code> | Top-level directory to recursively search for log files within. |
| <code>pattern</code>  | Regex pattern to search for files with.                         |
| <code>verbose</code>  | 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)
```

---

|                  |   |
|------------------|---|
| parse_pval_large | <i>Parse number of SNPs with p-values &gt;1</i> |
|------------------|---|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

```
parse_pval_large(1)
```

**Arguments**

|   |                              |
|---|------------------------------|
| 1 | Lines of text from log file. |
|---|------------------------------|

**Value**

Numeric

---

|                |   |
|----------------|---|
| parse_pval_neg | <i>Parse number of SNPs with p-values &lt;0</i> |
|----------------|---|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

```
parse_pval_neg(1)
```

**Arguments**

1                      Lines of text from log file.

**Value**

Numeric

---

|                  |  |
|------------------|--|
| parse_pval_small | <i>Parse number of SNPs with non-negative p-values &lt;=5e-324</i> |
|------------------|--|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

```
parse_pval_small(1)
```

**Arguments**

1                      Lines of text from log file.

**Value**

Numeric

---

|              |  |
|--------------|--|
| parse_report | <i>Parse "Summary statistics report" metrics</i> |
|--------------|--|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

```
parse_report(1, entry = 1, line = 1)
```

**Arguments**

1                      Lines of text from log file.

**Value**

Numeric

---

|                    |   |
|--------------------|---|
| parse_snps_freq_05 | <i>Parse number/percent of SNPs with <i>FREQ</i> values &gt;0.5</i> |
|--------------------|---|

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

```
parse_snps_freq_05(1, percent = FALSE)
```

**Arguments**

1                      Lines of text from log file.

**Value**

Numeric

---

`parse_snps_not_formatted`*Parse number of SNPs not correctly formatted*

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

```
parse_snps_not_formatted(1)
```

**Arguments**

1                      Lines of text from log file.

**Value**

Numeric

---

`parse_time`*Parse the total time taken the munge the file*

---

**Description**

Support function for [parse\\_logs](#).

**Usage**

```
parse_time(1)
```

**Arguments**

1                      Lines of text from log file.

**Value**

Character



---

|                  |  |
|------------------|--|
| preview_sumstats | <i>Preview formatted sum stats saved to disk</i> |
|------------------|--|

---

**Description**

Prints the first n lines of the sum stats.

**Usage**

```
preview_sumstats(save_path, nrows = 5L)
```

**Arguments**

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

**Value**

No return

---

|            |  |
|------------|--|
| 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 subsetting 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/ALScf.vcf.gz")

---

|                    |   |
|--------------------|---|
| 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 | <i>Determine summary statistics file type and read them into memory</i> |
|---------------|---|

---

**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;",...) : 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", "eduAttain0kbay.txt",
  package = "MungeSumstats"
)
eduAttain0kbay <- read_sumstats(path = path)
```

---

|          |                         |
|----------|-------------------------|
| read_vcf | <i>Read in VCF file</i> |
|----------|-------------------------|

---

**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_datatable    | 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           | Genomic ranges to be added if supplied. Default is NULL.  |
| 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.                                     |
| sampld_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&gt;1</code> 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>&lt; 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)
```

---

read\_vcf\_genome

*Read VCF genome*

---

**Description**

Get the genome build of a remote or local VCF file.

**Usage**

```
read_vcf_genome(
  header = NULL,
  validate = FALSE,
  default_genome = "HG19/GRCh37",
  verbose = TRUE
)
```

**Arguments**

- header           Header extracted by [scanVcfHeader](#).
- validate        Validate genome name using [mapGenomeBuilds](#).
- default\_genome   When no genome can be extracted, default to this genome build.
- verbose         Print messages.

**Value**

genome

---

|               |                              |
|---------------|------------------------------|
| read_vcf_info | <i>Read VCF: INFO column</i> |
|---------------|------------------------------|

---

**Description**

Parse INFO column in VCF file.

**Usage**

read\_vcf\_info(sumstats\_dt)

**Arguments**

- sumstats\_dt     Summary stats data.table.

**Value**

Null output.

---

|                     |                                    |
|---------------------|------------------------------------|
| read_vcf_markername | <i>Read VCF: MarkerName column</i> |
|---------------------|------------------------------------|

---

**Description**

Parse MarkerName/SNP column in VCF file.

**Usage**

read\_vcf\_markername(sumstats\_dt)

**Arguments**

- sumstats\_dt     Summary stats data.table.

**Value**

Null output.

---

|                   |                           |
|-------------------|---------------------------|
| read_vcf_parallel | <i>Read VCF: parallel</i> |
|-------------------|---------------------------|

---

**Description**

Read a VCF file across 1 or more threads in parallel. If tilewidth is not specified, the size of each chunk will be determined by total genome size divided by ntile. By default, ntile is equal to the number of threads, nThread. For further discussion on how this function was optimised, see [here](#) and [here](#).

**Usage**

```
read_vcf_parallel(  
  path,  
  samples = 1,  
  which = NULL,  
  use_params = TRUE,  
  as_datatable = TRUE,  
  sampled_rows = 10000L,  
  include_xy = FALSE,  
  download = TRUE,  
  vcf_dir = tempdir(),  
  download_method = "download.file",  
  force_new = FALSE,  
  tilewidth = NULL,  
  mt_thresh = 100000L,  
  nThread = 1,  
  ntile = nThread,  
  verbose = TRUE  
)
```

**Arguments**

|              |  |
|--------------|--|
| path         | Path to local or remote VCF file.  |
| 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;",...) : Only user-selected samples will be used (case-insensitive).</li></ul> |
| which        | Genomic ranges to be added if supplied. Default is NULL.   |
| 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.                    |
| as_datatable | Return the data as a <a href="#">data.table</a> (default: TRUE) or a <a href="#">VCF</a> (FALSE).  |



|                 |   |
|-----------------|---|
| sampled_rows    | First N rows to sample. Set NULL to use full sumstats_file. 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 nThread>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 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"). |
| download_method | "axel" (multi-threaded) or "download.file" (single-threaded) .  |
| 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.  |
| tilewidth       | The desired tile width. The effective tile width might be slightly different but is guaranteed to never be more than the desired width.   |
| mt_thresh       | When the number of rows (variants) in the VCF is < mt_thresh, 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.  |
| ntile           | The number of tiles to generate.  |
| verbose         | Print messages.   |

**Value**

VCF file.

**Source**

```
path <- "https://gwas.mrcieu.ac.uk/files/ieu-a-298/ieu-a-298.vcf.gz" ##### Single-threaded
#### vcf <- MungeSumstats:::read_vcf_parallel(path = path) ##### Parallel ##### vcf2 <-
MungeSumstats:::read_vcf_parallel(path = path, nThread=11)
```

---

|               |                                 |
|---------------|---------------------------------|
| read_vcf_pval | <i>Read VCF: p-value column</i> |
|---------------|---------------------------------|

---

**Description**

Parse p-value column in VCF file.

**Usage**

```
read_vcf_pval(sumstats_dt)
```

**Arguments**

sumstats\_dt      Summary stats data.table.

**Value**

Null output.

---

|                |                       |
|----------------|-----------------------|
| register_cores | <i>Register cores</i> |
|----------------|-----------------------|

---

**Description**

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

**Usage**

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

**Arguments**

workers      integer(1) Number of workers. Defaults to the maximum of 1 or the number of cores determined by detectCores minus 2 unless environment variables R\_PARALLELLY\_AVAILABLECORES\_FALLBACK or BIOCPARALLEL\_WORKER\_NUMBER are set otherwise. 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.

---

|                   |                             |
|-------------------|-----------------------------|
| remove_empty_cols | <i>Remove empty columns</i> |
|-------------------|-----------------------------|

---

**Description**

Remove columns that are empty or contain all the same values in a data.table.

**Usage**

```
remove_empty_cols(sumstats_dt, sampled_rows = NULL, verbose = TRUE)
```

**Arguments**

|              |  |
|--------------|--|
| sampled_rows | First N rows to sample. Set NULL to use full sumstats_file. when determining whether cols are empty. |
| verbose      | Print messages.  |

**Value**

Null output.

---

|                |   |
|----------------|---|
| report_summary | <i>Report info on current state of the summary statistics</i> |
|----------------|---|

---

**Description**

Prints report.

**Usage**

```
report_summary(sumstats_dt, orig_dims = NULL)
```

**Arguments**

|             |   |
|-------------|---|
| sumstats_dt | data table obj of the summary statistics file for the GWAS. |
|-------------|---|

**Value**

No return

---

|            |   |
|------------|---|
| select_api | <i>Toggle API address between development and release</i> |
|------------|---|

---

**Description**

From ieugwasr.

**Usage**

```
select_api(where = "public", verbose = TRUE)
```

**Arguments**

|       |  |
|-------|--|
| where | Which API to use. Choice between "local", "release", "test". Default = "local" |
|-------|--|

**Value**

No return

---

|                   |                   |
|-------------------|-------------------|
| select_vcf_fields | Select VCF fields |
|-------------------|-------------------|

---

### Description

Select non-empty columns from each VCF field type.

### Usage

```
select_vcf_fields(  
  path,  
  sampled_rows = 10000L,  
  which = NULL,  
  samples = NULL,  
  nThread = 1,  
  verbose = TRUE  
)
```

### Arguments

|              |  |
|--------------|--|
| path         | Path to local or remote VCF file.  |
| sampled_rows | First N rows to sample. Set NULL to use full sumstats_file. when determining whether cols are empty.   |
| which        | Genomic ranges to be added if supplied. Default is NULL.   |
| 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;", ...) : Only user-selected samples will be used (case-insensitive).</li></ul> |
| nThread      | Number of threads to use for parallel processes.   |
| verbose      | Print messages.  |

### Value

ScanVcfParam object.

---

|             |                       |
|-------------|-----------------------|
| sort_coords | <i>Sort sum stats</i> |
|-------------|-----------------------|

---

**Description**

Sort summary statistics table by genomic coordinates.

**Usage**

```
sort_coords(  
  sumstats_dt,  
  sort_coordinates = TRUE,  
  sort_method = c("data.table", "GenomicRanges")  
)
```

**Arguments**

- |             |   |
|-------------|---|
| sumstats_dt | <a href="#">data.table</a> obj of the summary statistics file for the GWAS.   |
| sort_method | Method to sort coordinates by: <ul style="list-style-type: none"><li>• "data.table" (default)Uses <a href="#">setorderv</a>, which is must faster than "GenomicRanges" but less robust to variations in some sum stats files.</li><li>• "GenomicRanges"Uses <a href="#">sort.GenomicRanges</a>, which is more robust to variations in sum stats files but much slower than the "data.table" method.</li></ul> |
| sort_coords | Whether to sort by coordinates.   |

**Value**

Sorted sumstats\_dt

---

|                       |                                   |
|-----------------------|-----------------------------------|
| sort_coords_datatable | <i>Sort sum stats: data.table</i> |
|-----------------------|-----------------------------------|

---

**Description**

Sort summary statistics table by genomic coordinates using a fast `data.table`-native strategy

**Usage**

```
sort_coords_datatable(  
  sumstats_dt,  
  chr_col = "CHR",  
  start_col = "BP",  
  end_col = start_col  
)
```

**Arguments**

- sumstats\_dt      [data.table](#) obj of the summary statistics file for the GWAS.
- chr\_col          Chromosome column name.
- start\_col        Genomic end position column name.

**Value**

Sorted sumstats\_dt

---

|                                      |
|--------------------------------------|
| sort_coord_genomicranges             |
| <i>Sort sum stats: GenomicRanges</i> |

---

**Description**

Sort summary statistics table by genomic coordinates using a slower (but in some cases more robust) GenomicRanges strategy

**Usage**

```
sort_coord_genomicranges(sumstats_dt)
```

**Arguments**

- sumstats\_dt      [data.table](#) obj of the summary statistics file for the GWAS.

**Value**

Sorted sumstats\_dt

---

|                    |   |
|--------------------|---|
| 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

- sumstats\_dt      data table obj of the summary statistics file for the GWAS.
- 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.
- uppercase\_unmapped      For columns that could not be identified in the mapping\_file, return them in the same format they were input as (without forcing them to uppercase).
- return\_list      Return the sumstats\_dt within a named list (default: TRUE).

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 | <i>Summary Statistics Column Headers</i> |
|--------------------|--|

---

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
"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$Corrected, sumstatsColHeaders$Uncorrected),]
rownames(sumstatsColHeaders)<-1:nrow(sumstatsColHeaders) sumstatsColHeaders$ordering
<- NULL #manually move FRWQUENCY to above MAR - github issue 95 frequency <- sumstatsColHeaders[sumstatsColHeaders$Corrected=="MAF",]
maf <- sumstatsColHeaders[sumstatsColHeaders$Uncorrected=="MAF",] if(as.integer(rownames(frequency))>
sumstatsColHeaders[as.integer(rownames(frequency)),] <- maf sumstatsColHeaders[as.integer(rownames(maf))>
<- frequency } 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
```

|                    |                             |
|--------------------|-----------------------------|
| supported_suffixes | List supported file formats |
|--------------------|-----------------------------|

### Description

## List supported file formats

## Usage

```
supported_suffixes(  
  tabular = TRUE,  
  tabular_compressed = TRUE,  
  vcf = TRUE,  
  vcf_compressed = TRUE  
)
```

## Arguments

|                                 |   |
|---------------------------------|---|
| <code>tabular</code>            | Include tabular formats.                |
| <code>tabular_compressed</code> | Include compressed tabular formats.     |
| <code>vcf</code>                | Include Variant Call Format.            |
| <code>vcf_compressed</code>     | Include compressed Variant Call Format. |

## Value

## File formats



---

|            |            |
|------------|------------|
| to_granges | To GRanges |
|------------|------------|

---

**Description**

Convert a [data.table](#) to [GRanges](#).

**Usage**

```
to_granges(  
  sumstats_dt,  
  seqnames.field = "CHR",  
  start.field = "BP",  
  end.field = "BP",  
  style = c("NCBI", "UCSC")  
)
```

**Arguments**

- sumstats\_dt      data table obj of the summary statistics file for the GWAS.
- seqnames.field    A character vector of recognized names for the column in df that contains the chromosome name (a.k.a. sequence name) associated with each genomic range. Only the first name in seqnames.field that is found in colnames(df) is used. If no one is found, then an error is raised.
- start.field        A character vector of recognized names for the column in df that contains the start positions of the genomic ranges. Only the first name in start.field that is found in colnames(df) is used. If no one is found, then an error is raised.
- end.field          A character vector of recognized names for the column in df that contains the end positions of the genomic ranges. Only the first name in start.field that is found in colnames(df) is used. If no one is found, then an error is raised.
- style              GRanges style to convert to, "NCBI" or "UCSC".

**Value**

GRanges object

---

|            |                    |
|------------|--------------------|
| to_vranges | Convert to VRanges |
|------------|--------------------|

---

**Description**

Convert to VRanges

**Usage**

```
to_vranges(sumstats_dt)
```

**Arguments**

sumstats\_dt      data table obj of the summary statistics file for the GWAS.

**Value**

VRanges object

---

|           |                            |
|-----------|----------------------------|
| unlist_dt | <i>Unlist a data.table</i> |
|-----------|----------------------------|

---

**Description**

Identify columns that are lists and turn them into vectors.

**Usage**

```
unlist_dt(dt, verbose = TRUE)
```

**Arguments**

dt                      data.table  
verbose                Print messages.

**Value**

dt with list columns turned into vectors.

---

|                     |   |
|---------------------|---|
| validate_parameters | <i>Ensure that the input parameters are logical</i> |
|---------------------|---|

---

**Description**

Ensure that the input parameters are logical

**Usage**

```
validate_parameters(  
  path,  
  ref_genome,  
  convert_ref_genome,  
  convert_small_p,  
  es_is_beta,  
  compute_z,  
  compute_n,  
  convert_n_int,  
  analysis_trait,  
  INFO_filter,  
  FRQ_filter,  
  pos_se,  
  effect_columns_nonzero,  
  N_std,  
  N_dropNA,  
  chr_style,  
  rmv_chr,  
  on_ref_genome,  
  infer_eff_direction,  
  strand_ambig_filter,  
  allele_flip_check,  
  allele_flip_drop,  
  allele_flip_z,  
  allele_flip_frq,  
  bi_allelic_filter,  
  flip_frq_as_biallelic,  
  snp_ids_are_rs_ids,  
  remove_multi_rs_snp,  
  frq_is_maf,  
  indels,  
  drop_indels,  
  check_dups,  
  dbSNP,  
  write_vcf,  
  return_format,  
  ldsc_format,  
  save_format,  
  imputation_ind,  
  log_folder_ind,  
  log_mungesumstats_msgs,  
  mapping_file,  
  tabix_index,  
  chain_source,  
  local_chain,  
  drop_na_cols,  
  rmv_chrPrefix
```

)

**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.   |
| es_is_beta         | Binary, whether to map ES to BETA. We take BETA to be any BETA-like value (including Effect Size). If this is not the case for your sumstats, change this to FALSE. Default is TRUE.   |
| compute_z          | Whether to compute Z-score column. Default is FALSE. This can be computed from Beta and SE with $(\text{Beta}/\text{SE})$ or $P (Z:=\text{sign}(\text{BETA})*\text{sqrt}(\text{stats::qchisq}(P,1,\text{lower}=\text{FALSE})))$ . <b>Note</b> that imputing the Z-score from P for every SNP will not be perfectly correct and may result in a loss of power. This should only be done as a last resort. Use 'BETA' to impute by BETA/SE and 'P' to impute by SNP p-value.   |
| 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 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.   |
| 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 sum-stats 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.  |
| chr_style              | Chromosome naming style to use in the formatted summary statistics file ("NCBI", "UCSC", "dbSNP", or "Ensembl"). The NCBI and Ensembl styles both code chromosomes as 1-22, X, Y, MT; the UCSC style is chr1-chr22, chrX, chrY, chrM; and the dbSNP style is ch1-ch22, chX, chY, chMT. Default is Ensembl. |
| rmv_chr                | Chromosomes to exclude from the formatted summary statistics file. Use NULL if no filtering is necessary. Default is c("X", "Y", "MT") which removes all non-autosomal SNPs.   |
| on_ref_genome          | Binary Should a check take place that all SNPs are on the reference genome by SNP ID. Default is TRUE.   |
| infer_eff_direction    | Binary Should a check take place to ensure the alleles match the effect direction? 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.  |
| flip_frq_as_biallelic  | Binary Should non-bi-allelic SNPs frequency values be flipped as 1-p despite there being other alternative alleles? Default is FALSE but if set to TRUE, this allows non-bi-allelic SNPs to be kept despite needing flipping.  |
| 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.  |
| drop_indels            | Binary, should any indels found in the sumstats be dropped? These can not be checked against a reference dataset and will have the same RS ID and position as SNPs which can affect downstream analysis. Default is False.   |
| check_dups             | whether to check for duplicates - if formatting QTL datasets this should be set to FALSE otherwise keep as TRUE. Default is TRUE.  |
| dbSNP                  | version of dbSNP to be used for imputation (144 or 155).   |
| write_vcf              | Whether to write as VCF (TRUE) or tabular file (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. <b>NOTE</b> - If LDSC format is used, the naming convention of A1 as the reference (genome build) allele and A2 as the effect allele will be reversed to match LDSC (A1 will now be the effect allele). See more info on this <a href="#">here</a> . Note that any effect columns (e.g. Z) will be in relation to A1 now instead of A2. |
| 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.   |
| 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   |
| 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.   |
| tabix_index   | Index the formatted summary statistics with <b>tabix</b> for fast querying.  |
| chain_source  | source of the chain file to use in liftover, if converting genome build ("ucsc" or "ensembl"). Note that the UCSC chain files require a license for commercial use. The Ensembl chain is used by default ("ensembl").  |
| local_chain   | Path to local chain file to use instead of downloading. Default of NULL i.e. no local file to use. NOTE if passing a local chain file make sure to specify the path to convert from and to the correct build like GRCh37 to GRCh38. We can not sense check this for local files. The chain file can be submitted as a gz file (as downloaded from source) or unzipped.   |
| drop_na_cols  | A character vector of column names to be checked for missing values. Rows with missing values in any of these columns (if present in the dataset) will be dropped. If NULL, all columns will be checked for missing values. Default columns are SNP, chromosome, position, allele 1, allele2, effect columns (frequency, beta, Z-score, standard error, log odds, signed sumstats, odds ratio), p value and N columns. |
| rmv_chrPrefix | Is now deprecated, do. not use. Use chr_style instead - chr_style = 'Ensembl' will give the same result as rmv_chrPrefix=TRUE used to give.  |

Value

No return

---

|        |                  |
|--------|------------------|
| vcf2df | <i>VCF to DF</i> |
|--------|------------------|

---

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</b> <a href="#">CollapsedVCF</a> / <a href="#">ExpandedVCF</a> 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.  |
| sampld_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 = NULL,
  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. <b>NOTE</b> - If LDSC format is used, the naming convention of A1 as the reference (genome build) allele and A2 as the effect allele will be reversed to match LDSC (A1 will now be the effect allele). See more info on this <a href="#">here</a> . Note that any effect columns (e.g. Z) will be in relation to A1 now instead of A2. |
| tabix_index | Index the formatted summary statistics with <a href="#">tabix</a> 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, [111](#)

## \* downloaders

axel, [6](#)

downloader, [51](#)

## \* internal

api\_query, [5](#)

axel, [6](#)

check\_access\_token, [7](#)

check\_allele\_flip, [7](#)

check\_allele\_merge, [9](#)

check\_bi\_allelic, [10](#)

check\_bp\_range, [11](#)

check\_chr, [12](#)

check\_col\_order, [13](#)

check\_drop\_indels, [13](#)

check\_dup\_bp, [14](#)

check\_dup\_col, [15](#)

check\_dup\_row, [16](#)

check\_dup\_snp, [17](#)

check\_effect\_columns\_nonzero, [18](#)

check\_empty\_cols, [19](#)

check\_four\_step\_col, [19](#)

check\_frq, [20](#)

check\_frq\_maf, [21](#)

check\_info\_score, [21](#)

check\_miss\_data, [23](#)

check\_multi\_gwas, [24](#)

check\_multi\_rs\_snp, [25](#)

check\_n\_int, [32](#)

check\_n\_num, [32](#)

check\_no\_allele, [26](#)

check\_no\_chr\_bp, [28](#)

check\_no\_rs\_snp, [29](#)

check\_no\_snp, [30](#)

check\_numeric, [31](#)

check\_on\_ref\_genome, [33](#)

check\_pos\_se, [35](#)

check\_range\_p\_val, [36](#)

check\_row\_snp, [37](#)

check\_save\_path, [38](#)

check\_signed\_col, [39](#)

check\_small\_p\_val, [40](#)

check\_strand\_ambiguous, [41](#)

check\_tabular, [42](#)

check\_two\_step\_col, [42](#)

check\_vcf, [43](#)

check\_vital\_col, [43](#)

check\_zscore, [44](#)

column\_dictionary, [45](#)

compute\_sample\_size, [47](#)

compute\_sample\_size\_n, [48](#)

compute\_sample\_size\_neff, [49](#)

convert\_sumstats, [50](#)

DF\_to\_dt, [51](#)

downloader, [51](#)

drop\_duplicate\_cols, [54](#)

drop\_duplicate\_rows, [54](#)

get\_access\_token, [64](#)

get\_chain\_file, [64](#)

get\_genome\_build, [65](#)

get\_query\_content, [68](#)

get\_unique\_name\_log\_file, [69](#)

get\_vcf\_sample\_ids, [69](#)

granges\_to\_dt, [70](#)

gwasinfo, [70](#)

index\_vcf, [79](#)

is\_tabix, [81](#)

legacy\_ids, [82](#)

logs\_example, [86](#)

make\_allele\_upper, [87](#)

message\_parallel, [88](#)

messenger, [87](#)

parse\_dropped\_chrom, [88](#)

parse\_dropped\_duplicates, [89](#)

parse\_dropped\_INFO, [89](#)

parse\_dropped\_nonA1A2, [90](#)

parse\_dropped\_nonBiallelic, [90](#)

- parse\_dropped\_nonRef, 91
- parse\_flipped, 91
- parse\_genome\_build, 92
- parse\_idStandard, 92
- parse\_pval\_large, 93
- parse\_pval\_neg, 94
- parse\_pval\_small, 94
- parse\_report, 95
- parse\_snps\_freq\_05, 95
- parse\_snps\_not\_formatted, 96
- parse\_time, 96
- preview\_sumstats, 97
- read\_vcf\_genome, 102
- read\_vcf\_info, 103
- read\_vcf\_markername, 103
- read\_vcf\_parallel, 104
- read\_vcf\_pval, 105
- remove\_empty\_cols, 106
- report\_summary, 107
- select\_api, 107
- select\_vcf\_fields, 108
- sort\_coord\_genomicranges, 110
- sort\_coords, 109
- sort\_coords\_datatable, 109
- supported\_suffixes, 112
- to\_granges, 113
- to\_vranges, 113
- unlist\_dt, 114
- validate\_parameters, 114
- \* **tabix**
  - index\_tabular, 78
  - index\_vcf, 79
- api\_query, 5
- axel, 6, 52
- check\_access\_token, 7
- check\_allele\_flip, 7
- check\_allele\_merge, 9
- check\_bi\_allelic, 10
- check\_bp\_range, 11
- check\_chr, 12
- check\_col\_order, 13
- check\_drop\_indels, 13
- check\_dup\_bp, 14
- check\_dup\_col, 15
- check\_dup\_row, 16
- check\_dup\_snp, 17
- check\_effect\_columns\_nonzero, 18
- check\_empty\_cols, 19
- check\_four\_step\_col, 19
- check\_freq, 20
- check\_freq\_maf, 21
- check\_info\_score, 21
- check\_ldsc\_format, 22
- check\_miss\_data, 23
- check\_multi\_gwas, 24
- check\_multi\_rs\_snp, 25
- check\_n\_int, 32
- check\_n\_num, 32
- check\_no\_allele, 26
- check\_no\_chr\_bp, 28
- check\_no\_rs\_snp, 29
- check\_no\_snp, 30
- check\_numeric, 31
- check\_on\_ref\_genome, 33
- check\_pos\_se, 35
- check\_range\_p\_val, 36
- check\_row\_snp, 37
- check\_save\_path, 38
- check\_signed\_col, 39
- check\_small\_p\_val, 40
- check\_strand\_ambiguous, 41
- check\_tabular, 42
- check\_two\_step\_col, 42
- check\_vcf, 43
- check\_vital\_col, 43
- check\_zscore, 44
- CollapsedVCF, 120
- column\_dictionary, 45
- compute\_nsize, 46
- compute\_sample\_size, 47
- compute\_sample\_size\_n, 48
- compute\_sample\_size\_neff, 49
- convert\_sumstats, 50
- data.table, 51, 70, 83, 93, 100, 101, 104, 109, 110, 113
- DataFrame, 51
- DF\_to\_dt, 51
- download.file, 51
- download\_vcf, 52
- downloader, 6, 51
- drop\_duplicate\_cols, 54
- drop\_duplicate\_rows, 54
- ExpandedVCF, 120

find\_sumstats, 55  
format\_sumstats, 31, 57, 73, 84, 86, 93  
formatted\_example, 56  
  
get\_access\_token, 64  
get\_chain\_file, 64  
get\_eff\_frq\_allele\_combns, 65  
get\_genome\_build, 65  
get\_genome\_builds, 67  
get\_query\_content, 68  
get\_unique\_name\_log\_file, 69  
get\_vcf\_sample\_ids, 69  
GRanges, 70, 83, 113  
granges\_to\_dt, 70  
gwasinfo, 70  
  
hg19ToHg38, 71  
hg38ToHg19, 71  
  
ieu-a-298, 72  
import\_sumstats, 72, 84, 93  
index\_tabular, 78, 79  
index\_vcf, 79, 79  
infer\_effect\_column, 80  
is\_tabix, 81  
  
legacy\_ids, 82  
liftover, 82  
list\_sumstats, 84  
load\_ref\_genome\_data, 85  
load\_snp\_loc\_data, 86  
logs\_example, 86  
  
make\_allele\_upper, 87  
mapGenomeBuilds, 103  
message\_parallel, 88  
messenger, 87  
  
parse\_dropped\_chrom, 88  
parse\_dropped\_duplicates, 89  
parse\_dropped\_INFO, 89  
parse\_dropped\_nonA1A2, 90  
parse\_dropped\_nonBiallelic, 90  
parse\_dropped\_nonRef, 91  
parse\_flipped, 91  
parse\_genome\_build, 92  
parse\_idStandard, 92  
parse\_logs, 88–93, 93, 94–96  
parse\_pval\_large, 93  
parse\_pval\_neg, 94  
  
parse\_pval\_small, 94  
parse\_report, 95  
parse\_snps\_freq\_05, 95  
parse\_snps\_not\_formatted, 96  
parse\_time, 96  
preview\_sumstats, 97  
  
raw\_ALSvcf, 97  
raw\_eduAttainOkbay, 98  
read\_header, 98  
read\_sumstats, 99  
read\_vcf, 100, 101, 104  
read\_vcf\_genome, 102  
read\_vcf\_info, 103  
read\_vcf\_markername, 103  
read\_vcf\_parallel, 104  
read\_vcf\_pval, 105  
register\_cores, 106  
remove\_empty\_cols, 106  
report\_summary, 107  
  
scanVcfHeader, 103  
select\_api, 107  
select\_vcf\_fields, 108  
setorderv, 109  
sort.GenomicRanges, 109  
sort\_coord\_genomicranges, 110  
sort\_coords, 109  
sort\_coords\_datatable, 109  
standardise\_header, 56, 110  
sumstatsColHeaders, 111  
supported\_suffixes, 112  
  
to\_granges, 113  
to\_vranges, 113  
  
unlist\_dt, 114  
  
validate\_parameters, 114  
VCF, 100, 101, 104  
vcf2df, 119  
  
write\_sumstats, 121