Package ‘MungeSumstats’

May 13, 2024

Type Package

Title Standardise summary statistics from GWAS

Version 1.12.0

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, markdown, 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 RELEASE_3_19

git_last_commit f27f8e8

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-12

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
Contents

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
compute_sample_size ................................ 47
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_vcf_sample_ids .................................. 69
get_unique_name_log_file ............................. 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
Contents

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
**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**

```r
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**

**axel downloader**

**Description**

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

**Usage**

```r
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/](https://github.com/axel-download-accelerator/axel/)

Other downloaders: `downloader()`
check_access_token

**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**

```r
check_access_token()
```

**Value**

NULL or access_token depending on current authentication state

---

check_allele_flip

*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).*

**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**

```r
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,
 )
```
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 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. Note 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 tabix for fast querying.
**check_allele_merge**

- **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 or 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**: `snpsByYld`, filtered to SNPs of interest if loaded already. Or else NULL.
- **log_files**: log file list

---

**Description**

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

**Usage**

```r
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  Remove non-biallelic SNPs

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 tabix 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).
check_bp_range

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

Description

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

Usage

```r
check_bp_range(
  sumstats_dt,
  path,
  ref_genome,
  log_folder_ind,
  imputation_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.
- **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 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. **Note**
these columns will be in the formatted summary statistics returned. Default is FALSE.

**tabix_index**  
Index the formatted summary statistics with tabix 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**  
Standardize the CHR column

**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**

```r
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 tabix 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.
check_col_order

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

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

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  Drop Indels from summary statistics

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
)

check_drop_indels  Drop Indels from summary statistics
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 tabix 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

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

Description

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

Usage

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

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 tabix 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

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  

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

Description

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

Usage

```r
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 tabix 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

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

Description

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

Usage

```r
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 tabix 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**

```r
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 summary stats 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 summary stats 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 `tabix` 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  

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  

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

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_fraq  

Ensure all SNPs have frq score above threshold

Description

Ensure all SNPs have frq score above threshold

Usage

check_fraq(
  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 tabix 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_freq_maf

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

**Description**

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

**Usage**

```r
check_freq_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

**Ensure all SNPs have info score above threshold**

**Description**

Ensure all SNPs have info score above threshold

**Usage**

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

**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 tabix 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

---

**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**

```r
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. **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 inrelation to A1 now instead of A2.
check_miss_data

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 \( \text{Beta/SE or } P \left( Z = \text{sign(BETA)} \times \sqrt{\text{stats::qchisq(P,1,lower=FALSE)}} \right) \). 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.

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 imputed 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  Remove SNPs with missing data

Description

Remove SNPs with missing data

Usage

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

check_multi_gwas

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

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
)
check_multi_rs_snp

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

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
check_no_allele

Ensure that A1 & A2 are present, if not can find it with SNP and other allele

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

```r
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 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. **Note** 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 tabix 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**: snpsByld, 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

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

Description

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

Usage

```r
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 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. **Note** 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 tabix 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`: `snpIdsByld`, filtered to SNPs of interest if loaded already. Or else NULL
- `log_files`: log file list

Description

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

Usage

```r
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 `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.
- **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. Note 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 tabix 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 Ensure that SNP is present if not can find it with CHR and BP

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 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. Note 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 tabix 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 columns

check_numeric

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 format_sumstats.

cols

Names of columns that should be numeric. If any of these columns are not actually present in sumstats_dt, they will be skipped.
check_n_int

Ensure that the N column is all integers

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

Ensure all SNPs have N less than X std dev below mean

Description

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

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 datatable of the summary statistics file can also be passed directly to MungeSumstats using the path parameter.

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.

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 tabix 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: Ensure all SNPs are on the reference genome

Description

Ensure all SNPs are on the reference genome
check_on_ref_genome

Usage

check_on_ref_genome(
  sumstats_dt,  # 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.
  path,  # 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.
  ref_genome,  # Binary Should a check take place that all SNPs are on the reference genome by SNP ID. Default is TRUE.
  on_ref_genome,  # 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.
  indels = indels,  # 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. Note these columns will be in the formatted summary statistics returned. Default is FALSE.
  imputation_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_folder_ind,  # Index the formatted summary statistics with tabix for fast querying.
  check_save_out,  # Number of threads to use for parallel processes.
  tabix_index,  # list of log file locations
  nThread,  # version of dbSNP to be used for imputation (144 or 155).
  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 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. Note 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 tabix 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` : `snpsByld`, filtered to SNPs of interest if loaded already. Or else NULL
- `log_files` : log file list

Description

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

Usage

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

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.
- **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.
- **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 where switched based on `MungeSumstats` initial choice of A1, A2 from the input
check_range_p_val

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

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 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. Note these columns will be in the formatted summary statistics returned. Default is FALSE.
check_row.snp

**Value**

list containing sumstats_dt, the modified summary statistics data table object

**Source**

```r
```

**Description**

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

**Usage**

```r
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 tabix 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

Check if save path and log folder is appropriate

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 tempfile(fileext=".tsv.gz").
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.
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 tabix 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**

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

**Description**

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

**Usage**

```r
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:
  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 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. **Note** these columns will be in the formatted summary statistics returned. Default is FALSE.
- **tabix_index**: Index the formatted summary statistics with tabix for fast querying.
- **log_files**: list of log file locations
- **nThread**: Number of threads to use for parallel processes.
**check_small_p_val**

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

**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 <= 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 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. **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

**Source**

```r
```
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 tabix 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  

Ensure valid tabular format

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  

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

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

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

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  

Check for Z-score column

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

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

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). **Note** these columns will be in the formatted summary statistics returned. Default is FALSE.
- `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:=sign(BETA)*sqrt(stats::qchisq(P,1,lower=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.
- `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 or 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.
Details

\[ \text{np.sqrt(chi2.isf(P, 1))} \]

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

\[ \text{sumstats.dt[, Z:=sign(BETA) * sqrt(stats::qchisq(P, 1, lower=FALSE))]} \]

*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

\[ \text{list("sumstats.dt"=sumstats.dt)} \]

Description

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

Usage

\[ \text{column_dictionary(file_path)} \]

Arguments

\[ \text{file_path} \quad \text{Path to full summary stats file (or any really file you want to make a column dictionary for).} \]

Value

Named list of column positions.

Source

Borrowed function from echotabix.

\[ \text{eduAttainOkbayPath} <- \text{system.file("extdata", "eduAttainOkbay.txt", package = "MungeSumstats")} \]
\[ \text{tmp <- tempfile(fileext = ".tsv") file.copy(eduAttainOkbayPath, tmp) cdict <- MungeSumstats::column_dictionary(file_path = tmp)} \]
compute_nsize

*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.

**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**

```r
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). **Note** 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").
  - 0: N will not be computed.
  - >0: If any number >0 is provided, that value will be set as N for every row. **Note**: Computing N this way is incorrect and should be avoided if at all possible.
  - "sum": N will be computed as: cases (N_CAS) + controls (N_CON), so long as both columns are present.
  - "ldsc": N will be computed as effective sample size: Neff = (N_CAS+N_CON) / mean(N_CAS/(N_CAS+N_CON))(N_CAS+N_CON)==max(N_CAS+N_CON)).
  - "giant": N will be computed as effective sample size: Neff = 2 / (1/N_CAS + 1/N_CON).
  - "metal": N will be computed as effective sample size: Neff = 4 / (1/N_CAS + 1/N_CON).
compute_sample_size

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  Compute (effective) sample size

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":
  \[ \text{Neff} = \frac{(N_{\text{AS}}+N_{\text{ON}})*(N_{\text{AS}}/(N_{\text{AS}}+N_{\text{ON}}))}{\text{mean}\left(\frac{N_{\text{AS}}}{N_{\text{AS}}+N_{\text{ON}}}\right)\left(\frac{N_{\text{AS}}+N_{\text{ON}}}{N_{\text{AS}}+N_{\text{ON}}}\right)} \]
  bulik/ldsc GitHub Issue bulik/ldsc GitHub code
- "giant":
  \[ \text{Neff} = \frac{2}{1/N_{\text{AS}} + 1/N_{\text{ON}}} \]
  Winkler et al. 2014, Nature
compute_sample_size_n

- "metal":
  \[ Neff = \frac{4}{(1/N_{CAS} + 1/N_{CON})} \]
  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\>":
  \[ N = \<integer\> \]
  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

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"

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

**Description**

Compute Neff/N

**Usage**

```r
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":
    \[
    Neff = (N_{CAS} + N_{C0N}) \times \frac{(N_{CAS}/(N_{CAS} + N_{C0N}))}{\text{mean}((N_{CAS}/(N_{CAS} + N_{C0N}))]}\]

**Value**

No return

---

- "giant":
  \[
  \hat{N}_{eff} = \frac{2}{(1/N_{CAS} + 1/N_{C0N})}
  \]
  Winkler et al. 2014, Nature

- "metal":
  \[
  Neff = \frac{4}{(1/N_{CAS} + 1/N_{C0N})}
  \]
  Willer et al. 2010, Bioinformatics

- "sum":
  \[
  N = N_{CAS} + N_{C0N}
  \]
  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.
• "giant":
  \[ N_{eff} = \frac{2}{1/N_{CAS} + 1/N_{CON}} \]
  Winkler et al. 2014, Nature

• "metal":
  \[ N_{eff} = \frac{4}{1/N_{CAS} + 1/N_{CON}} \]
  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\>":
  \[ N = \<integer\> \]
  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  

Convert summary statistics to desired object type

Description

Convert summary statistics to desired object type

Usage

```r
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

DataFrame to data.table

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
downloader wrapper

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
)

download_vcf

Arguments

- **input_url**
- **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**
- **check_certificates**
- **timeout**
  - How many seconds before giving up on download. Passed to download.file.
  - 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.
download_vcf

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. WARNING: 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**  
*Drop duplicate columns*

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

**Usage**  
\[
\text{drop_duplicate_cols}(dt)
\]

**Arguments**  
- dt \hspace{1cm} data.table

**Value**  
Null output

---

**drop_duplicate_rows**  
*Drop duplicate rows*

**Description**  
Drop rows with duplicate values across all columns.

**Usage**  
\[
\text{drop_duplicate_rows}(dt, \text{verbose} = \text{TRUE})
\]

**Arguments**  
- dt \hspace{1cm} data.table  
- verbose \hspace{1cm} 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**

```r
find_sumstats(
  ids = NULL,
  traits = NULL,
  years = NULL,
  consortia = NULL,
  authors = NULL,
  populations = NULL,
  categories = NULL,
  subcategories = NULL,
  builds = NULL,
  pmids = NULL,
  min_sample_size = NULL,
  min_ncase = NULL,
  min_ncontrol = NULL,
  min_nsnp = NULL,
  include_NAs = FALSE,
  access_token = check_access_token()
)
```

**Arguments**

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

**Details**

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

**Value**

(Filtered) GWAS metadata table.

**Examples**

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

**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

Check that summary statistics from GWAS are in a homogeneous format

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 = FALSE,
  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,
               "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 datable 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 <= 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 \( Z = \text{sign}(BETA) \times \sqrt{\text{stats::qchisq}(P,1,\text{lower}=\text{FALSE})} \). 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 inputed 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. 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.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>infer_eff_direction</td>
<td>Binary Should a check take place to ensure the alleles match the effect direction? Default is TRUE.</td>
</tr>
<tr>
<td>strand_ambig_filter</td>
<td>Binary Should SNPs with strand-ambiguous alleles be removed. Default is FALSE.</td>
</tr>
<tr>
<td>allele_flip_check</td>
<td>Binary Should the allele columns be checked against reference genome to infer if flipping is necessary. Default is TRUE.</td>
</tr>
<tr>
<td>allele_flip_drop</td>
<td>Binary Should the SNPs for which neither their A1 or A2 base pair values match a reference genome be dropped. Default is TRUE.</td>
</tr>
<tr>
<td>allele_flip_z</td>
<td>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.</td>
</tr>
<tr>
<td>allele_flip_frq</td>
<td>Binary should the frequency (FRQ) column be flipped along with effect and z-score columns like Beta? Default TRUE.</td>
</tr>
<tr>
<td>bi_allelic_filter</td>
<td>Binary Should non-biallelic SNPs be removed. Default is TRUE.</td>
</tr>
<tr>
<td>flip_frq_as_biallelic</td>
<td>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.</td>
</tr>
<tr>
<td>snp_ids_are_rs_ids</td>
<td>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.</td>
</tr>
<tr>
<td>remove_multi_rs_snp</td>
<td>Binary Sometimes summary statistics can have multiple RSIDs on one row (i.e. related to one SNP), for example &quot;rs5772025_rs397784053&quot;. This can cause an error so by default, the first RS ID will be kept and the rest removed e.g.&quot;rs5772025&quot;. If you want to just remove these SNPs entirely, set it to TRUE. Default is FALSE.</td>
</tr>
<tr>
<td>frq_is_maf</td>
<td>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. &gt;0.5. By default this mapping won’t occur i.e. is TRUE.</td>
</tr>
<tr>
<td>indels</td>
<td>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.</td>
</tr>
<tr>
<td>drop_indels</td>
<td>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.</td>
</tr>
</tbody>
</table>
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, allele 2, 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 tempfile(fileext=".tsv.gz").

write_vcf Whether to write as VCF (TRUE) or tabular file (FALSE).

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_mungesumstatsmsgs 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 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. Note 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.

default

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 col-
umn header that is in your file is missing or the mapping we give is incorrect
you can supply your own mapping file. Must be a 2 column dataframe with col-
umn 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  Get access token for OAuth2 access to MR Base

Description
Get access token for OAuth2 access to MR Base

Usage
get_access_token()

Value
access token string

get_chain_file  Download chain file for liftover

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
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

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

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.
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 Downsampling the number of SNPs used when inferring genome build to save time.
standardise_headers Run standardise_sumstats_column_headers_crossplatform.
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 or the mapping file 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 proportion 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

**get_genome_builds**  
_Infer genome builds_

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

### Usage
```r
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**

Parse out json response from httr object

**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

name proposed name for the entry
log_files 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

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.

Value

sample_id
granges_to_dt  
*GenomicRanges to data.table*

**Description**
Convert a `GRanges` into a `data.table`.

**Usage**
```r
granges_to_dt(gr)
```

**Arguments**
- `gr`  
  A `GRanges` object.

**Value**
A `data.table` object.

**Source**
Code adapted from GenomicDistributions.

---

**gwasinfo**  
*Get list of studies with available GWAS summary statistics through API*

**Description**
Get list of studies with available GWAS summary statistics through API.

**Usage**
```r
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

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

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.gz',tempdir())

Description

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

Format

gunzipped tsv file

Details

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

Source

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

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

Arguments

ids List of Open GWAS study IDs (e.g. c("prot-a-664", "ieu-b-4760").
vcf_dir Where to download the original VCF from Open GWAS. WARNING: This is set
to tempdir() by default. This means the raw (pre-formatted) VCFs be deleted
upon ending the R session. Change this to keep the raw VCF file on disk (e.g.
vcf_dir="./raw_vcf").
vcf_download Download the original VCF from Open GWAS.
save_dir Directory to save formatted summary statistics in.
write_vcf Whether to write as VCF (TRUE) or tabular file (FALSE).
download_method "axel" (multi-threaded) or "download.file" (single-threaded).
quiet Run quietly.
force_new If a formatted file of the same names as save_path exists, formatting will be
skipped and this file will be imported instead (default). Set force_new=TRUE to
override this.
force_new_vcf Overwrite a previously downloaded VCF with the same path name.
nThread Number of threads to use for parallel processes.
parallel_across_ids If parallel_across_ids=TRUE and nThread>1, then each ID in ids will be
processed in parallel.
...

Arguments passed on to format_sumstats

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.
import_sumstats

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 <= 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:=sign(BETA)*sqrt(stats::qchisq(P,1,lower=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
import_sumstats

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
cautions. The different methods MungeSumstats will try and impute se (in
this order or priority) are:

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 inform-
ation 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 En-
sembl 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 refer-
ence 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 refer-
ence genome to infer if flipping is necessary. Default is TRUE.
import_sumstats

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").
import_sumstats

**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","ranges","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 inrelation 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 denotes 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. **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**

```r
# 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**

Tabix-index file: table

**Description**

Convert summary stats file to tabix format.

**Usage**

```r
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").
  - 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.
index_vcf

Value
Path to tabix-indexed tabular file

Source
Borrowed function from echotabix.

See Also
Other tabix: index_vcf()

Examples

```r
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 | Tabix-index file: VCF

Description
Convert summary stats file to tabix format

Usage

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Path to VCF.</td>
</tr>
<tr>
<td>verbose</td>
<td>Print messages.</td>
</tr>
</tbody>
</table>

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**  
Infer if effect relates to A1 or A2 if ambiguously named

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/freq naming.

3. If not 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
)
```
is_tabix

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sumstats_dt</td>
<td>data table obj of the summary statistics file for the GWAS.</td>
</tr>
<tr>
<td>dbSNP</td>
<td>version of dbSNP to be used for imputation (144 or 155).</td>
</tr>
<tr>
<td>sampled_snps</td>
<td>Downsample the number of SNPs used when inferring genome build to save time.</td>
</tr>
<tr>
<td>mapping_file</td>
<td>MungeSumstats has a pre-defined column-name mapping file which should cover</td>
</tr>
<tr>
<td></td>
<td>the most common column headers and their interpretations. However, if a col-</td>
</tr>
<tr>
<td></td>
<td>umn header that is in your file is missing of the mapping we give is incorrect</td>
</tr>
<tr>
<td></td>
<td>you can supply your own mapping file. Must be a 2 column dataframe with col-</td>
</tr>
<tr>
<td></td>
<td>umn names &quot;Uncorrected&quot; and &quot;Corrected&quot;. See data(sumstatsColHeaders) for</td>
</tr>
<tr>
<td></td>
<td>default mapping and necessary format.</td>
</tr>
<tr>
<td>nThread</td>
<td>Number of threads to use for parallel processes.</td>
</tr>
<tr>
<td>ref_genome</td>
<td>name of the reference genome used for the GWAS (&quot;GRCh37&quot; or &quot;GRCh38&quot;).</td>
</tr>
<tr>
<td></td>
<td>Argument is case-insensitive. Default is NULL which infers the reference Genome</td>
</tr>
<tr>
<td></td>
<td>from the data.</td>
</tr>
<tr>
<td>on_ref_genome</td>
<td>Binary Should a check take place that all SNPs are on the reference genome by</td>
</tr>
<tr>
<td></td>
<td>SNP ID. Default is TRUE.</td>
</tr>
<tr>
<td>infer_eff_direction</td>
<td>Binary Should a check take place to ensure the alleles match the effect direction?</td>
</tr>
<tr>
<td></td>
<td>Default is TRUE.</td>
</tr>
<tr>
<td>return_list</td>
<td>Return the sumstats_dt within a named list (default: TRUE).</td>
</tr>
</tbody>
</table>

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 | Convert current IDs to legacy IDs |

Description

Convert current IDs to legacy IDs

Usage

legacy_ids(x)

Arguments

x Vector of ids

Value

vector of back compatible ids

---

| liftover | Genome build liftover |

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 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. **Note** 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 GRanges instead of a data.table (default: FALSE).
- **style**: Style to return GRanges 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

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

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

---

**list_sumstats**  
*List munged summary statistics*

---

**Description**

Searches for and lists local GWAS summary statistics files munged by `format_sumstats` or `import_sumstats`.

**Usage**

```r
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**

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

Load the reference genome data for SNPs of interest

Description

Load the reference genome data for SNPs of interest

Usage

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>snps</td>
<td>Character vector SNPs by rs_id from sumstats file of interest.</td>
</tr>
<tr>
<td>ref_genome</td>
<td>Name of the reference genome used for the GWAS (GRCh37 or GRCh38)</td>
</tr>
<tr>
<td>dbSNP</td>
<td>version of dbSNP to be used (144 or 155)</td>
</tr>
<tr>
<td>msg</td>
<td>Optional name of the column missing from the dataset in question. Default is NULL</td>
</tr>
<tr>
<td>chr_filt</td>
<td>Internal for testing - filter reference genomes and sumstats to specific chromosomes for testing. Pass a list of chroms in format: c(&quot;1&quot;,&quot;2&quot;). Default is NULL i.e. no filtering.</td>
</tr>
</tbody>
</table>

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

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  Example logs file

Description

Example logs file produced by format_sumstats.

Usage

logs_example(read = FALSE)

Arguments

- **read**: Whether to read the logs file into memory.
### make_allele_upper

**Ensure A1 and A2 are upper case**

**Description**

Ensure A1 and A2 are upper case

**Usage**

```r
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

---

### messager

**Print messages**

**Description**

Print messages with option to silence.

**Usage**

```r
messager(..., v = TRUE)
```
**parse_dropped_chrom**

**Arguments**

... Message input.

v Whether to print messages.

**Value**

Null output.

---

**message_parallel**

*Send messages to console even from within parallel processes*

**Description**

Send messages to console even from within parallel processes

**Usage**

message_parallel(...)

**Value**

A message

---

**parse_dropped_chrom**

*Parse number of SNPs dropped due to being on chrom X, Y or MT*

**Description**

Support function for parse_logs.

**Usage**

parse_dropped_chrom(l)

**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(l)`

**Arguments**

- `l`  
  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(l)`

**Arguments**

- `l`  
  Lines of text from log file.

**Value**

Numeric
### parse_dropped_nonA1A2

**Parse number of SNPs dropped due to not matching the ref genome A1 or A2**

**Description**
Support function for `parse_logs`.

**Usage**

```r
parse_dropped_nonA1A2(l)
```

**Arguments**

- `l` Lines of text from log file.

**Value**

Numeric

### parse_dropped_nonBiallelic

**Parse number of SNPs dropped due to not being bi-allelic**

**Description**
Support function for `parse_logs`.

**Usage**

```r
parse_dropped_nonBiallelic(l)
```

**Arguments**

- `l` Lines of text from log file.

**Value**

Numeric
**parse_dropped_nonRef**

*Parse number of SNPs dropped due to being in the ref genome*

**Description**

Support function for `parse_logs`.

**Usage**

```r
parse_dropped_nonRef(l)
```

**Arguments**

- `l` 
  Lines of text from log file.

**Value**

Numeric

---

**parse_flipped**

*Parse number of SNPs flipped to align with the ref genome*

**Description**

Support function for `parse_logs`.

**Usage**

```r
parse_flipped(l)
```

**Arguments**

- `l` 
  Lines of text from log file.

**Value**

Numeric
parse_genome_build  

*Genome build inferred from the summary statistics*

**Description**

Support function for `parse_logs`.

**Usage**

```r
parse_genome_build(l)
```

**Arguments**

1  

Lines of text from log file.

**Value**

Character

---

parse_idStandard  

*Standardised IEU MRC OpenGWAS ID*

**Description**

Support function for `parse_logs`.

**Usage**

```r
parse_idStandard(l)
```

**Arguments**

1  

Lines of text from log file.

**Value**

Character
parse_logs

**Parse data from log files**

**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**

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

**Arguments**

- `save_dir` Top-level directory to recursively search for log files within.
- `pattern` Regex pattern to search for files with.
- `verbose` Print messages.

**Value**

Data frame of parsed log data.

**Examples**

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

parse_pval_large

**Parse number of SNPs with p-values >1**

**Description**

Support function for `parse_logs`.

**Usage**

```r
parse_pval_large(l)
```

**Arguments**

- `l` Lines of text from log file.
parse_pval_neg

**Description**
Support function for `parse_logs`.

**Usage**
`parse_pval_neg(l)`

**Arguments**
1. `l`: Lines of text from log file.

**Value**
Numeric

parse_pval_small

**Description**
Support function for `parse_logs`.

**Usage**
`parse_pval_small(l)`

**Arguments**
1. `l`: Lines of text from log file.

**Value**
Numeric
### `parse_report`

Parse "Summary statistics report" metrics

**Description**

Support function for `parse_logs`.

**Usage**

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

**Arguments**

- **l**: Lines of text from log file.

**Value**

Numeric

### `parse_snps_freq_05`

Parse number/percent of SNPs with FREQ values >0.5

**Description**

Support function for `parse_logs`.

**Usage**

`parse_snps_freq_05(l, percent = FALSE)`

**Arguments**

- **l**: 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(l)`

**Arguments**

- `l`

  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(l)`

**Arguments**

- `l`

  Lines of text from log file.

**Value**

Character
**preview_sumstats**

*Preview formatted sum stats saved to disk*

**Description**

Prints the first \( n \) lines of the sum stats.

**Usage**

```r
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**

*GWAS Amyotrophic lateral sclerosis ieu open GWAS project - Subset*

**Description**

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

**Format**

vcf document with 528 items relating to 99 SNPs

**Details**

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

**ALSvcf.vcf**

NA

**Source**

The summary statistics VCF (VCFv4.2) file was downloaded from https://gwas.mrcieu.ac.uk/datasets/ebi-a-GCST005647/ and formatted to a .rda with the following:

```r
#Get example VCF dataset, use
GWAS Amyotrophic lateral sclerosis ALS_GWAS_VCF <- readLines("ebi-a-GCST005647.vcf.gz")
#Subset to just the first 99 SNPs ALSvcf <- ALS_GWAS_VCF[1:528] writeLines(ALSvcf,"inst/extdata/ALSvcf.vcf")
```
Description

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.

deduAttainOkbay.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 are bi/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(tmp)
#DT read removes the .0's #remove those not on ref genome and with bi/tri allelic rmv <- c("rs192818565","rs79925071","rs1606974","rs1871109","rs73074378","rs7955289") eduAttainOkbay <- eduAttainOkbay[!MarkerName data.table::fwrite(eduAttainOkbay,file=tmp,sep=\"\")
eduAttainOkbay <- readLines(tmp) writeLines(eduAttainOkbay,"inst/extdata/eduAttainOkbay.txt")
```

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

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

**read_sumstats**

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

Description

Determine summary statistics file type and read them into memory

Usage

```r
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.
read_vcf

standardise_headers
Standardise headers first.
samples
Which samples to use:
- 1: Only the first sample will be used (DEFAULT).
- NULL: All samples will be used.
- c("<sample_id1>"", "<sample_id2>"", ...): Only user-selected samples will be used (case-insensitive).
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 or the mapping we give is incorrect, you can supply your own mapping file. Must be a 2 column dataframe with column names "Uncorrected" and "Corrected". See data(sumstatsColHeaders) for default mapping and necessary format.

Value
data.table of formatted summary statistics

Examples
path <- system.file("extdata", "eduAttainOkbay.txt","eduAttainOkbay <- read_sumstats(path = path)

read_vcf
Read in VCF file

Description
Read in a VCF file as a VCF or a data.table. Can optionally save the VCF/data.table as well.

Usage
read_vcf(
  path,
  as_datatable = TRUE,
  save_path = NULL,
  tabix_index = FALSE,
  samples = 1,
  which = NULL,
  use_params = TRUE,
  sampled_rows = 10000L,
download = TRUE,
vcf_dir = tempdir(),
download_method = "download.file",
force_new = FALSE,
mt_thresh = 100000L,
nThread = 1,
verbose = TRUE
)

Arguments

path
Path to local or remote VCF file.

as_datatable
Return the data as a data.table (default: TRUE) or a VCF (FALSE).

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.

samples
Which samples to use:
• 1: Only the first sample will be used (DEFAULT).
• NULL: All samples will be used.
• c("<sample_id1>"","<sample_id2>"...,) : Only user-selected samples will be used (case-insensitive).

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 this requires the VCF to be sorted, bgzip-compressed, tabix-indexed, which read_vcf will attempt to do.

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. WARNING: 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.

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.

verbose
Print messages.
**Value**

The VCF file in data.table format.

**Source**

```r
#### Benchmarking ####
library(VCFWrenchR)
library(VariantAnnotation)
path <- "https://gwas.mrcieu.ac.uk/files/ubm-a-2929/ubm-a-2929.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 #####

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

##### Remote file #####

```r
## Small GWAS (0.2Mb)
# path <- "https://gwas.mrcieu.ac.uk/files/ieu-a-298/ieu-a-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/ieu-a-1124/ieu-a-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**

```r
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**  
**Read VCF: INFO column**

**Description**

Parse INFO column in VCF file.

**Usage**

```r
read_vcf_info(sumstats_dt)
```

**Arguments**

- **sumstats_dt**
  
  Summary stats data.table.

**Value**

Null output.

---

**read_vcf_markername**  
**Read VCF: MarkerName column**

**Description**

Parse MarkerName/SNP column in VCF file.

**Usage**

```r
read_vcf_markername(sumstats_dt)
```

**Arguments**

- **sumstats_dt**
  
  Summary stats data.table.

**Value**

Null output.
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:
  • 1: Only the first sample will be used (DEFAULT).
  • NULL: All samples will be used.
  • c("<sample_id1>",<sample_id2>,...): Only user-selected samples will be used (case-insensitive).
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 this requires the VCF to be sorted, bgzip-compressed, tabix-indexed, which read_vcf will attempt to do.
as_datatable Return the data as a data.table (default: TRUE) or a VCF (FALSE).
**read_vcf_pval**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sampled_rows</td>
<td>First N rows to sample. Set NULL to use full sumstats_file when determining whether cols are empty.</td>
</tr>
<tr>
<td>download</td>
<td>Download the VCF (and its index file) to a temp folder before reading it into R. This is important to keep TRUE when nThread&gt;1 to avoid making too many queries to remote file.</td>
</tr>
<tr>
<td>vcf_dir</td>
<td>Where to download the original VCF from Open GWAS. WARNING: 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=&quot;/raw_vcf&quot;).</td>
</tr>
<tr>
<td>download_method</td>
<td>&quot;axel&quot; (multi-threaded) or &quot;download.file&quot; (single-threaded).</td>
</tr>
<tr>
<td>force_new</td>
<td>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.</td>
</tr>
<tr>
<td>tilewidth</td>
<td>The desired tile width. The effective tile width might be slightly different but is guaranteed to never be more than the desired width.</td>
</tr>
<tr>
<td>mt_thresh</td>
<td>When the number of rows (variants) in the VCF is &lt; 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.</td>
</tr>
<tr>
<td>nThread</td>
<td>Number of threads to use for parallel processes.</td>
</tr>
<tr>
<td>ntile</td>
<td>The number of tiles to generate.</td>
</tr>
<tr>
<td>verbose</td>
<td>Print messages.</td>
</tr>
</tbody>
</table>

**Value**

VCF file.

**Source**

```r
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**

Read VCF: p-value column

**Description**

Parse p-value column in VCF file.

**Usage**

```r
read_vcf_pval(sumstats_dt)
```
remove_empty_cols

Arguments

sumstats_dt Summary stats data.table.

Value

Null output.

register_cores

Register cores

Description

Register a multi-threaded instances using BiocParallel.

Usage

register_cores(workers = 1, progressbar = TRUE)

Arguments

workers integer(1) Number of workers. Defaults to the maximum of 1 or the number of cores determined by detectCores minus 2 unless environment variables R_PARALLELY_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

Remove empty columns

Description

Remote 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)
**report_summary**

**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.

---

**Description**

Prints report.

**Usage**

```r
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**  

**Toggle API address between development and release**

**Description**

From ieugwasr.

**Usage**

```r
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

```r
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:
  - 1: Only the first sample will be used (DEFAULT).
  - NULL: All samples will be used.
  - c("<sample_id1>"","<sample_id2>"...) : Only user-selected samples will be used (case-insensitive).
- **nThread**
  Number of threads to use for parallel processes.
- **verbose**
  Print messages.

Value

ScanVcfParam object.
**Description**

Sort summary statistics table by genomic coordinates.

**Usage**

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

**Arguments**

- `sumstats_dt`: data.table obj of the summary statistics file for the GWAS.
- `sort_method`: Method to sort coordinates by:
  - "data.table" (default) Uses `setorder`, which is much faster than "GenomicRanges" but less robust to variations in some sum stats files.
  - "GenomicRanges" Uses `sort.GenomicRanges`, which is more robust to variations in sum stats files but much slower than the "data.table" method.
- `sort_coords`: Whether to sort by coordinates.

**Value**

Sorted `sumstats_dt`

---

**sort_coords_datatable**

**Sort sum stats: data.table**

**Description**

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

**Usage**

```r
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

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

Standardise the column headers in the Summary Statistics files

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", "eduAttain0kby.txt", package = "MungeSumstats"))
sumstats_dt2 <- standardise_header(sumstats_dt=sumstats_dt)

sumstatsColHeaders  Summary Statistics Column Headers

Description

List of uncorrected column headers often found in GWAS Summary Statistics column headers. Note the effect allele will always be the A2 allele, this is the approach done for VCF(https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7805039). This is enforced with the column header corrections here and also the check allele flipping test.

Usage

data("sumstatsColHeaders")
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

tabular Include tabular formats.
tabular_compressed Include compressed tabular formats.
vcf Include Variant Call Format.
vcf_compressed Include compressed Variant Call Format.

Value

File formats
to_granges

**Description**

Convert a data.table to GRanges.

**Usage**

```r
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 end.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
to_vranges(sumstats_dt)

Arguments

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

Value

VRanges object

Usage

unlist_dt(dt, verbose = TRUE)

Arguments

dt  data.table
verbose  Print messages.

Value

dt with list columns turned into vectors.

validate_parameters  Ensure that the input parameters are logical

Description

Ensure that the input parameters are logical
validate_parameters

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_fraq_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 <= 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 (Beta/SE) or P (Z := sign(BETA) * sqrt(stats::qchisq(P,1,lower=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.

`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 imputed 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 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.
validate_parameters

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. 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 inrelation 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 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. Note 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 `tabix` 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 **VCF to DF**

**Description**

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

**Usage**

``` r
cvf2df(
    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 `VariantAnnotation ColapsedVCF/ExpandedVCF` object.

add_sample_names
Append sample names to column names (e.g. "EZ" -> "EZ_ubm-a-2929").

add_rowranges
Include rowRanges from VCF as well.

drop_empty_cols
Drop columns that are filled entirely with: NA, ".", or "".

unique_cols
Only keep uniquely named columns.

unique_rows
Only keep unique rows.

unlist_cols
If any columns are lists instead of vectors, unlist them. Required to be TRUE when unique_rows=TRUE.

sampled_rows
First N rows to sample. Set NULL to use full sumstats_file. when determining whether cols are empty.

verbose
Print messages.

Value
data.frame version of VCF

Source

Original code source

vcfR:

```r
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)
```
**Description**

Write sum stats file to disk

**Usage**

```r
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 `tempfile(fileext=".tsv.gz")`.
- `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 `[ , \t ; : ]` 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 `base::readLines` 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`. **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.
- `tabix_index`: Index the formatted summary statistics with `tabix` for fast querying.
- `nThread`: The number of threads to use. Experiment to see what works best for your data on your hardware.
write_sumstats

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", "eduAttainOkbay.txt", package = "MungeSumstats")
eduAttainOkbay <- read_sumstats(path = path)
write_sumstats(
    sumstats_dt = eduAttainOkbay,
    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
messager, 87
parse_dropped_chrom, 88
parse_dropped_duplicates, 89
parse_dropped_INFO, 89
parse_dropped_nonA1A2, 90
parse_dropped_nonBiallelic, 90
124

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_biallelic, 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_snps, 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_snps, 30
check_numeric, 31
check_on_ref_genome, 33
check_pos_se, 35
check_range_p_val, 36
check_row_snps, 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
INDEX

125

find_sumstats, 55
format_sumstats, 31, 57, 73, 84, 86, 93
formatted_example, 56

get_access_token, 64
get_chain_file, 64
get_eff_freq_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
messager, 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_markrname, 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