Package 'echoLD'

August 31, 2021

Type Package

Title echoverse module: LD downloading and processing

Version 0.99.0

Description echoverse module: LD downloading and processing.

URL https://github.com/RajLabMSSM/echoLD

BugReports https://github.com/RajLabMSSM/echoLD/issues

Encoding UTF-8

LazyData true

Depends R (>= 3.6.0)

SystemRequirements Python (>= 3.7.0)

biocViews

Imports echotabix,
downloadR,
magrittr,
dplyr,
utils,
R.utils,
stats,
methods,
Matrix,
parallel,
BiocManager,
data.table,
reticulate,
GenomeInfoDb,
GenomicRanges,
VariantAnnotation,
snpStats,
gaston,
rtracklayer,
LDlinkR,
adjclust

Suggests markdown,
rmarkdown,
remotes,

knitr,
BiocStyle,
covr,
testthat (>= 3.0.0)

Remotes github::RajLabMSSM/echotabix,
github::RajLabMSSM/downloadR

RoxygenNote 7.1.1

VignetteBuilder knitr

License GPL (>= 3) + file LICENSE

Config/testthat/edition 3

R topics documented:

BST1 **echolocatoR** *output example: BST1 locus*

Description

An example results file after running `finemap_loci` on the *BST1* locus.

Usage

```
data("BST1")
```

Format

data.table

SNP SNP RSID

CHR Chromosome

POS Genomic position (in basepairs)

... Optional: extra columns

Nalls2019

data.table

Details

Data originally comes from the Parkinson's disease GWAS by Nalls et al., (bioRxiv).

Source

```
root_dir <-"~/Desktop/Fine_Mapping/Data/GWAS/Nalls23andMe_2019/BST1/Multi-finemap"
BST1 <-data.table::fread(file.path(root_dir,"Multi-finemap_results.txt")) BST1 <-update_cols(dat
= BST1) BST1 <-find_consensus_SNPs(dat = BST1) usethis::use_data(BST1,overwrite = TRUE)
```

BST1_LD_matrix *LD with the lead SNP: BST1 locus*

Description

Precomputed LD within the *BST1* locus (defined in BST1. LD derived British, European-decent subpopulation in the UK Biobank. Only includes a subset of all the SNPs for storage purposes (including the lead GWAS/QTL SNP).

Usage

```
data("BST1_LD_matrix")
```

Format

data.table

SNP SNP RSID

CHR Chromosome

POS Genomic position (in basepairs)

... Optional: extra columns

UK Biobank Nalls 2019

matrix

Details

Data originally comes from UK Biobank. LD was pre-computed and stored by the Alkes Price lab (see here).

Source

```
data("BST1") finemap_DT <-BST1 # Only including a small subset of the full # LD matrix
for storage purposes. lead_snp <-subset(finemap_DT,leadSNP)$SNP snp_list <-finemap_DT[which(finemap
== lead_snp) -100:which(finemap_DT$SNP == lead_snp) + 100,]$SNP BST1_LD_matrix <-readRDS("../Fine_Ma
BST1_LD_matrix <-BST1_LD_matrix[snp_list,snp_list] usethis::use_data(BST1_LD_matrix,overwrite
= T)
```

get_LD_blocks *Identify the LD block in which the lead SNP resides*

Description

Identify the LD block in which the lead SNP resides

Usage

```
get_LD_blocks(
  dat,
  ss,
  stats = c("R.squared", "D.prime"),
  pct = 0.15,
  verbose = TRUE
)
```

Arguments

dat	SNP-level data table.
ss	**snpStats** object or LD matrix (containing r or r^2 values).
stats	a character vector specifying the linkage disequilibrium measures to be calculated (using the ld function) when x is a genotype matrix. Only "R.squared" and "D.prime" are allowed, see Details.
pct	minimum percentage of points for the plateau selection in capushe selection. See DDSE for further details
verbose	Print messages.

Value

A list with the input data and LD matrix (r^2),

Source

adjclust GitHub

get_UKB_MAF *Get MAF from UK Biobank.*

Description

If **MAF** column is missing, download MAF from UK Biobank and use that instead.

Usage

```
get_UKB_MAF(
  dat,
  output_path = file.path(tempdir(), "Data/Reference/UKB_MAF"),
  force_new_maf = FALSE,
  download_method = "axel",
  nThread = 1,
  verbose = TRUE,
  conda_env = "echoR"
)
```

Arguments

dat	SNP-level data.
output_path	Path to store UKB_MAF file in.
force_new_maf	Download UKB_MAF file again.
download_method	• "axel" : Multi-threaded • "wget" : Single-threaded • "download.file" : Single-threaded • "internal" : Single-threaded (passed to download.file) • "wininet" : Single-threaded (passed to download.file) • "libcurl" : Single-threaded (passed to download.file) • "curl" : Single-threaded (passed to download.file) or "download.file" (single-threaded) .
nThread	Number of threads to parallelize over.
verbose	Print messages.
conda_env	Conda environment to use.

Source

UKB

Examples

```
data("BST1")
dat <- data.frame(BST1)[, colnames(BST1) != "MAF"]
BST1 <- get_UKB_MAF(dat = dat)
```

---

liftover *Genome build liftover*

Description

Transfer genomic coordinates from one genome build to another.

Usage

```
liftover(
  dat,
  chrom_col = "CHR",
  start_col = "POS",
  end_col = start_col,
  build_conversion = c("hg38ToHg19", "hg19ToHg38"),
  as_granges = FALSE,
  verbose = TRUE
)
```

Arguments

dat	SNP-level data table.
chrom_col	Name of the chromosome column.
start_col	Name of the start position column.
end_col	Name of the end position column (can be same as start_col if all data is SNP-level).
build_conversion	From which to which genome build to lift over dat.
as_granges	Return lifted dat as GenomicRanges object.
verbose	Print messages.

Source

liftOver

UCSC chain files

Examples

```
data("BST1")
dat_lifted <- liftover(dat = BST1, build_conversion = "hg19ToHg38")
```

---

load_or_create *Procure an LD matrix for fine-mapping*

Description

Calculate and/or query linkage disequilibrium (LD) from reference panels (UK Biobank, 1000 Genomes), a user-supplied pre-computed LD matrix

Usage

```
load_or_create(
  locus_dir,
  dat,
  force_new_LD = FALSE,
  LD_reference = c("1KGphase1", "1KGphase3", "UKB"),
  ref_genome = "hg19",
```

```
  samples = NULL,
  superpopulation = NULL,
  local_storage = NULL,
  leadSNP_LD_block = FALSE,
  fillNA = 0,
  verbose = TRUE,
  remove_tmps = TRUE,
  as_sparse = TRUE,
  download_method = "axel",
  nThread = 1
)
```

Arguments

`locus_dir`	Storage directory to use.
`dat`	GWAS summary statistics subset to query the LD panel with.
`force_new_LD`	If LD file exists, create a new one.
`LD_reference`	LD reference to use: • "1KGphase1" : 1000 Genomes Project Phase 1 • "1KGphase3" : 1000 Genomes Project Phase 3 • "UKB" : Pre-computed LD from a British European-decent subset of UK Biobank.
`ref_genome`	Genome build of the LD panel (used only if providing custom LD panel).
`samples`	Sample names to subset the VCF by before computing LD.
`superpopulation`	Superpopulation to subset LD panel by (used only if `LD_reference` is "1KGphase1" or "1KGphase3".)
`local_storage`	Storage folder for previously downloaded LD files. If `LD_reference` is "1KGphase1" or "1KGphase3", `local_storage` is where VCF files are stored. If `LD_reference` is "UKB", `local_storage` is where LD compressed numpy array (npz) files are stored. Set to `NULL` to download VCFs/LD npz from remote storage system.
`leadSNP_LD_block`	Only return SNPs within the same LD block as the lead SNP (the SNP with the smallest p-value).
`fillNA`	Value to fill LD matrix NAs with.
`verbose`	Print messages.
`remove_tmps`	Remove all intermediate files like VCF, npz, and plink files.
`as_sparse`	Convert the LD matrix to a sparse matrix.
`download_method`	• `"axel"` : Multi-threaded • `"wget"` : Single-threaded • `"download.file"` : Single-threaded • `"internal"` : Single-threaded (passed to download.file) • `"wininet"` : Single-threaded (passed to download.file) • `"libcurl"` : Single-threaded (passed to download.file) • `"curl"` : Single-threaded (passed to download.file) or `"download.file"` (single-threaded) .
`nThread`	Number of threads to parallelize over.

Details

Options:

- Download pre-computed LD matrix from UK Biobank.
- Download raw VCF file from 1KG and compute LD on the fly.
- Compute LD on the fly from a user-supplied VCF file.
- Use a user-supplied pre-computed LD-matrix.

Value

A symmetric LD matrix of pairwise SNP correlations.

See Also

Other LD: `LD_1KG_download_vcf()`, `LD_1KG()`, `LD_custom()`, `LD_ukbiobank()`, `compute_LD()`, `filter_LD()`, `get_locus_vcf_folder()`, `ldlinkr_ldproxy_batch()`, `plot_LD()`, `popDat_1KGphase1`, `popDat_1KGphase3`, `rds_to_npz()`, `saveSparse()`, `save_LD_matrix()`, `snpstats_get_MAF()`, `translate_population()`

Examples

```
data("BST1")
data("locus_dir")
locus_dir <- file.path(tempdir(), locus_dir)
BST1 <- BST1[seq(1, 50), ]
## Not run:
LD_matrix <- load_or_create(
    locus_dir = locus_dir,
    dat = BST1,
    LD_reference = "1KGphase1"
)

## End(Not run)
```

`locus_dir` *Example results path for BST1 locus*

Description

Example results path for BST1 locus

Usage

```
data("locus_dir")
```

Format

path string

Source

`locus_dir <-"results/GWAS/Nalls23andMe_2019/BST1" usethis::use_data(locus_dir,overwrite = T)`

popDat_1KGphase1 *Population metadata: 1KGphase1*

Description

Individual-level metadata for 1000 Genomes Project (Phase 1).

Usage

```
data("popDat_1KGphase1")
```

Format

data.table

Source

```
popDat_URL <-"ftp://ftp-trace.ncbi.nih.gov/1000genomes/ftp/release/20110521/phase1_integrated_ca
popDat_1KGphase1 <-data.table::fread(text = trimws(gsub(",\t",",",readLines(popDat_URL))),sep
= "\t",fill = T,stringsAsFactors = F,col.names = c("sample","population","superpop","sex"),nThread
= 4) usethis::use_data(popDat_1KGphase1,overwrite = T)
```

See Also

Other LD: LD_1KG_download_vcf(), LD_1KG(), LD_custom(), LD_ukbiobank(), compute_LD(), filter_LD(), get_locus_vcf_folder(), ldlinkr_ldproxy_batch(), load_or_create(), plot_LD(), popDat_1KGphase3, rds_to_npz(), saveSparse(), save_LD_matrix(), snpstats_get_MAF(), translate_population()

popDat_1KGphase3 *Population metadata: 1KGphase3*

Description

Individual-level metadata for 1000 Genomes Project (Phase 3).

Usage

```
data("popDat_1KGphase3")
```

Format

data.table

Source

```
popDat_URL <-"ftp://ftp-trace.ncbi.nih.gov/1000genomes/ftp/release/20130502/integrated_call_samp
popDat_1KGphase3 <-data.table::fread(text = trimws(gsub(",\t",",",readLines(popDat_URL))),sep
= "\t",fill = T,stringsAsFactors = F,col.names = c("sample","population","superpop","sex"),nThread
= 4) usethis::use_data(popDat_1KGphase3,overwrite = T)
```

See Also

Other LD: LD_1KG_download_vcf(), LD_1KG(), LD_custom(), LD_ukbiobank(), compute_LD(), filter_LD(), get_locus_vcf_folder(), ldlinkr_ldproxy_batch(), load_or_create(), plot_LD(), popDat_1KGphase1, rds_to_npz(), saveSparse(), save_LD_matrix(), snpstats_get_MAF(), translate_population()

Index