Package 'SumTool'

February 28, 2020

Title Summary Statistics Analysis Tool

Version 0.99.5

Date 2020-02-02

Maintainer Lilin Yin <ylilin@163.com>

Description
A tool for Summary data based analysis. It can do ld, ldscore computation, Zscore, Marginal effect imputation, summary data based SNP BLUP (SBLUP) solution, as well as LD regression for heritability and genetic correlation estimation. It featured with memory efficiency and parallel calculation. By the aid of package 'bigmemory', SumTool constructs memory-mapped files for genotype panel on disk instead of loading it all into Random Access Memory (RAM), which makes it possible to handle big data with limited computation resources. Additionally, all parallel processes are accelerated as fast as possible by OpenMP technology.

License GPL (>= 2)

URL https://github.com/YinLiLin/SumTool

BugReports https://github.com/YinLiLin/SumTool/issues

Encoding UTF-8

Imports utils, stats, methods, Rcpp

Depends R (>= 3.3.0), bigmemory

LinkingTo Rcpp, RcppArmadillo (>= 0.8.400.0.0), RcppProgress, BH, bigmemory

RoxygenNote 6.1.1

Roxygen list(markdown = TRUE)

SystemRequirements C++11

NeedsCompilation yes

Author Lilin Yin [aut, cre, cph],
Jian Zeng [aut, cph],
Jian Yang [aut, cph]

R topics documented:

`LDcal` *LD calculation*

Description

To calculate LD for all pairs of SNPs on given genome

Usage

```
LDcal(geno = NULL, index = NULL, threads = 1, lambda = 0,
  chisq = 0, correlation = TRUE, out = NULL, verbose = TRUE)
```

Arguments

`geno`	big.matrix (n * m), genotype
`index`	vector, only calculate LD for a subset with all SNPs (m * m1)
`threads`	number, the number of used threads for parallel process
`lambda`	number, add a value on the diagonal of LD matrix
`chisq`	number, generate a sparse LD matrix, if ld^2 * n < chisq, it will be set to 0
`correlation`	logical, if TRUE, the LD matrix will be constructed by correlation of all pairs
`out`	character, the path and prefix of output file, if not NULL, the LD matrix will stored in big.matrix, which will save much memory, but cost a bit more time
`verbose`	logical, whether to print the log information

Examples

```
gwas_bfile_path <- system.file("extdata", "gwas_geno", package = "SumTool")
data <- read_plink(bfile=gwas_bfile_path, threads=1)
geno <- data$geno
ld <- LDcal(geno=geno, threads=1)
```

`LDclump` *LD clumping*

Description

To remove high correlated SNPs on base of r2 and p-values

Usage

```
LDclump(geno = NULL, map = NULL, p = NULL, w = 1e+06,
  r2.cutoff = 0.25, p.cutoff = 1, verbose = TRUE, threads = 1)
```

Arguments

geno	bigmemory (n * m), genotype coded as 0, 1, 2
map	data.frame, the genomic information of SNPs. The columns should be in the order of c("SNP", "Chr", "Pos", "A1", "A2")
p	data.frame, at least 2 columns, the first column should be SNP names, the last column should be the pvalues
w	int, size of windows in bp. Default is 1e6
r2.cutoff	double, the threshold of r2, smaller cutoff results in less remaining SNPs, default 0.25
p.cutoff	double, the threshold of p-values, smaller threshold results in less remaining SNPs, default 1
verbose	logical, whether to print the log information
threads	int, the number of used threads for parallel process

Examples

```
ref_bfile_path <- system.file("extdata", "ref_geno", package = "SumTool")
p_path <- system.file("extdata", "P.txt", package = "SumTool")
# load data
data <- read_plink(bfile=ref_bfile_path, threads=1)
geno <- data$geno
map <- data$map
pdata <- read.table(p_path, header = TRUE)
snp <- LDclump(geno = geno, map = map, p = pdata, p.cutoff = 1, r2.cutoff = 0.25, w = 100
```

LDprune *LD pruning*

Description

To remove high correlated SNPs on base of r2 and MAF

Usage

```
LDprune(geno = NULL, map = NULL, w = 1e+06, b = 125000,
  r2.cutoff = 0.25, verbose = TRUE, threads = 1)
```

Arguments

geno	bigmemory (n * m), genotype coded as 0, 1, 2
map	data.frame, the genomic information of SNPs. The columns should be in the order of c("SNP", "Chr", "Pos", "A1", "A2")
w	int, size of windows in bp. Default is 1e6
b	int, set the buffer size in bp of each window, default 125000
r2.cutoff	double, the threshold of r2, smaller cutoff results in less remaining SNPs, default 0.25
verbose	logical, whether to print the log information
threads	int, the number of used threads for parallel process

Examples

```
ref_bfile_path <- system.file("extdata", "ref_geno", package = "SumTool")

# load data
data <- read_plink(bfile=ref_bfile_path, threads=1)
geno <- data$geno
map <- data$map
snp <- LDprune(geno = geno, map = map, w = 100000, b=50000, threads = 1)
```

`LDreg` *LD regression*

Description

To estimate heritability or genetic correlation

Usage

```
LDreg(sumstat = NULL, ldscore = NULL, wld = NULL, maxz2 = 80,
  maf = 0.05, nblock = 200, verbose = TRUE)
```

Arguments

`sumstat`	data.frame or list, results of single trait or multiple traits of GWAS summary statistics. For each summary statistic of a trait, the columns should be in the order of c("SNP", "Chr", "Pos", "A1", "A2", "BETA", "SE", "N"). If it's a data.frame, the heritability will be estimated, if it's a list containing multiple GWAS summary statistics, both heritability of traits and the genetic correlation of pair of traits will be estimated.
`ldscore`	data.frame, the ldscore for each SNP, the columns should be prepared in order of c("SNP", "Chr", "Pos", "A1", "A2", "MAF", "ldscore")
`wld`	data.frame, at least 2 colmuns, the first column is th SNP names, the last column is the weight for each SNP. Default equals to the 'ldscore' option
`maxz2`	double, max chi^2. Default is 80
`maf`	double, Minor allele frequency lower bound. Default is MAF > 0.05
`nblock`	int, number of blocks for jackknife estimation
`verbose`	logical, whether to print the log information

Examples

```
# single trait
sumstat1_path <- system.file("extdata", "sumstat1", package = "SumTool")
ldscore_path <- system.file("extdata", "ldscore", package = "SumTool")
sumstat1 <- read.table(sumstat1_path, header=TRUE)
ldscore <- read.table(ldscore_path, header=TRUE)
res1 <- LDreg(sumstat = sumstat1, ldscore = ldscore)

#multiple traits
sumstat2_path <- system.file("extdata", "sumstat2", package = "SumTool")
sumstat2 <- read.table(sumstat2_path, header=TRUE)
res2 <- LDreg(sumstat = list(sumstat1, sumstat2), ldscore = ldscore)
```

LDsore *LD score calculation*

Description

To estimate LD score for each SNPs

Usage

```
LDsore(geno = NULL, map = NULL, w = 1e+06, b = 125000, r2 = TRUE,
  adjust = TRUE, verbose = TRUE, threads = 1)
```

Arguments

geno	bigmemory (n * m), genotype coded as 0, 1, 2
map	data.frame, the genomic information of SNPs. The columns should be in the order of c("SNP", "Chr", "Pos", "A1", "A2")
w	int, size of windows in bp. Default is 1e6
b	number, set the buffer size in bp of each window, default 125000
r2	logical, calculate r2 or r
adjust	logical, whether to adjust the ldscore
verbose	logical, whether to print the log information
threads	number, the number of used threads for parallel process

Examples

```
ref_bfile_path <- system.file("extdata", "ref_geno", package = "SumTool")

# load data
data <- read_plink(bfile=ref_bfile_path, threads=1)
geno <- data$geno
map <- data$map
ldscore <- LDsore(geno = geno, map = map, w = 100000, b=12500, threads = 1)
```

read_plink *Data transformation*

Description

To read Plink binary format data into bigmemory (0, 1, 2) format

Usage

```
read_plink(bfile = "", maxLine = 10000, backingpath = NULL,
  descriptorfile = NULL, backingfile = NULL, verbose = TRUE,
  threads = 0)
```

Arguments

bfile	character, prefix of Plink binary format data.
maxLine	number, set the number of lines to handle at a time, bigger lines require more memory.
backingpath	the path to the directory containing the file backing cache.
descriptorfile	the name of the file to hold the backingfile description, for subsequent use with 'attach.big.matrix'; if 'NULL', the 'backingfile' is used as the root part of the descriptor file name. The descriptor file is placed in the same directory as the backing files.
backingfile	the root name for the file(s) for the cache.
verbose	logical, whether to print the log information
threads	number, the number of used threads for parallel process

Examples

```
ref_bfile_path <- system.file("extdata", "ref_geno", package = "SumTool")
data <- read_plink(bfile=ref_bfile_path, threads=1)
geno <- data$geno
map <- data$map
```

SBLUP *Summary data based SNP BLUP*

Description

To estimate joint SNP effect using GWAS summary statistics

Usage

```
SBLUP(sumstat = NULL, geno = NULL, map = NULL, lambda = NULL,
  w = 1e+06, threads = 1, verbose = TRUE)
```

Arguments

sumstat	data.frame, GWAS summary statistics of a trait. The columns should be in the order of c("SNP", "Chr", "Pos", "A1", "A2", "BETA", "SE, "N")
geno	bigmemory (n * m), genotype coded as 0, 1, 2
map	data.frame, the genomic information of SNPs. The columns should be in the order of c("SNP", "Chr", "Pos", "A1", "A2")
lambda	double, the ridge regression coefficient, lambda=m*(1/h2-1), m is the number of SNPs, h2 is the heritability of trait
w	int, size of windows in bp. Default is 1e6
threads	number, the number of used threads for parallel process
verbose	logical, whether to print the log information

Examples

```
sumstat_path <- system.file("extdata", "typed.marginal", package = "SumTool")
ref_bfile_path <- system.file("extdata", "ref_geno", package = "SumTool")

# load data
sumstat <- read.table(sumstat_path, header=TRUE)
data <- read_plink(bfile=ref_bfile_path, threads=1)
geno <- data$geno
map <- data$map
h2 <- 0.5
lambda = nrow(sumstat)*(1/h2-1)
eff <- SBLUP(sumstat = sumstat, geno = geno, map = map, lambda = lambda, threads = 1)
```

`SImputeB` *Summary data based imputation on marginal effect*

Description

To impute marginal effect using summary data by SImpute/SImpute-LD

Usage

```
SImputeB(ref.geno = NULL, ref.map = NULL, typed.geno = NULL,
  typed = NULL, w = 1e+06, b = 125000, lambda = NULL,
  maf = 1e-06, correlation = TRUE, verbose = TRUE, threads = 1)
```

Arguments

`ref.geno`	big.matrix (n1 * m1), reference genotype panel
`ref.map`	matrix (m1 * 5): SNPs, Chr, position, A1, A2
`typed.geno`	big.matrix (n2 * m2), individual level genotype for typed SNPs (this file is optional)
`typed`	matrix (m2 * 8): SNPs, Chr, position, A1, A2, BETA, SE, N
`w`	number, set the window size in bp, default 1000000
`b`	number, set the buffer size in bp of each window, default 250000
`lambda`	number, ridge regression value on LD matrix of typed SNPs: solve(Rtt + diag(lambda))
`maf`	number, SNPs whose minor allele frequency are lower than set value will not be imputed
`correlation`	logical, if TRUE, the LD matrix will be constructed by correlation of all pairs
`verbose`	logical, whether to print the log information
`threads`	number, the number of used threads for parallel process

Examples

```
#----------------SImpute----------------#

# get path of the attached files in package
ref_bfile_path <- system.file("extdata", "ref_geno", package = "SumTool")
typed_b_path <- system.file("extdata", "typed.marginal", package = "SumTool")

# reading data
data <- read_plink(bfile=ref_bfile_path, threads=1)
ref.geno <- data$geno
ref.map <- data$map
typed_b <- read.table(typed_b_path, header=TRUE)

# Impute marginal effect and se
xx <- SImputeB(ref.geno=ref.geno, ref.map=ref.map, typed=typed_b, threads=1)

#--------------SImpute-LD---------------#

gwas_bfile_path <- system.file("extdata", "gwas_geno", package = "SumTool")
gwas <- read_plink(bfile=gwas_bfile_path, threads=1)
typed.geno <- gwas$geno
typed.map <- gwas$map
#NOTE: the order of SNPs in 'typed.geno' should be consistent with the order in 'typed_b'
typed.geno <- deepcopy(typed.geno, cols = match(typed_b[, 1], typed.map[, 1]))

# Impute marginal effect and se
xx <- SImputeB(ref.geno=ref.geno, ref.map=ref.map, typed=typed_b, typed.geno=typed.geno,
```

`SImputeZ` *Summary data based imputation on Z-score*

Description

To impute Zscore using summary data by SImpute/SImpute-LD

Usage

```
SImputeZ(ref.geno = NULL, ref.map = NULL, typed.geno = NULL,
  typed = NULL, w = 1e+06, b = 125000, lambda = NULL,
  maf = 1e-06, correlation = TRUE, verbose = TRUE, threads = 1)
```

Arguments

`ref.geno`	big.matrix (n1 * m1), reference genotype panel
`ref.map`	matrix (m1 * 5): SNPs, Chr, position, A1, A2
`typed.geno`	big.matrix (n2 * m2), individual level genotype for typed SNPs (this file is optional)
`typed`	matrix (m2 * 6): SNPs, Chr, position, A1, A2, Z
`w`	number, set the window size in bp, default 1000000
`b`	number, set the buffer size in bp of each window, default 250000
`lambda`	number, ridge regression value on LD matrix of typed SNPs: solve(Rtt + diag(lambda))

maf	number, SNPs whose minor allele frequency are lower than set value will not be imputed
correlation	logical, if TRUE, the LD matrix will be constructed by correlation of all pairs
verbose	logical, whether to print the log information
threads	number, the number of used threads for parallel process

Examples

```
#----------------SImpute----------------#

# get path of the attached files in package
ref_bfile_path <- system.file("extdata", "ref_geno", package = "SumTool")
typed_z_path <- system.file("extdata", "typed.zscore", package = "SumTool")

# reading data
data <- read_plink(bfile=ref_bfile_path, threads=1)
ref.geno <- data$geno
ref.map <- data$map
typed_z <- read.table(typed_z_path, header=TRUE)

# Impute Zscore
xx <- SImputeZ(ref.geno=ref.geno, ref.map=ref.map, typed=typed_z, threads=1)

#--------------SImpute-LD---------------#

gwas_bfile_path <- system.file("extdata", "gwas_geno", package = "SumTool")
gwas <- read_plink(bfile=gwas_bfile_path, threads=1)
typed.geno <- gwas$geno
typed.map <- gwas$map
#NOTE: the order of SNPs in 'typed.geno' should be consistent with the order in 'typed_z'
typed.geno <- deepcopy(typed.geno, cols = match(typed_z[, 1], typed.map[, 1]))

# Impute Zscore
xx <- SImputeZ(ref.geno=ref.geno, ref.map=ref.map, typed=typed_z, typed.geno=typed.geno,
```