Package 'Gimpute'

June 27, 2018

Type Package

Title Gimpute: An efficient genetic data processing and imputation pipeline

Version 0.99.6

Date 2018-06-27

Author Junfang Chen, Dietmar Lippold, Emanuel Schwarz

Maintainer Junfang Chen <junfang.chen33@gmail.com>

Description Genotype imputation is essential for genome-wide association studies (GWAS) to retrieve information of untyped variants and facilitate comparability across studies. Based on widely used and freely available tools, we have developed Gimpute, an automated processing and imputation pipeline for genome-wide association data. Gimpute includes processing steps for genotype liftOver, quality control, population outlier detection, haplotype pre-phasing, imputation, post imputation, data management and the extension to other existing pipeline.

License GPL-3

Imports doParallel, lattice

Depends R (>= 3.5.0)

Suggests BiocStyle, knitr

VignetteBuilder knitr

biocViews Genetics, GenomicVariation, SNP, GenomeWideAssociation, QualityControl, VariantDetection, Alignment

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

R topics documented:

`checkAlign2ref` *Check the alignment with the imputation reference panel*

Description

Perform the alignment against a reference panel by considering the following parameters: variant name, genomic position and the allele profile. Output files are generated sequentially, so they are determined by the previous PLINK files.

Usage

```
checkAlign2ref(plink, inputPrefix, bimReferenceFile, out2, out2.snp, out3,
  out3.snp, out4, out4.snp, out4.snpRetained, nCore = 25)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK files.
`bimReferenceFile`	the reference file used for the alignment, which is a PLINK BIM alike format file.
`out2`	the prefix of the output PLINK binary files after removing SNPs whose genomic positions are not in the imputation reference, taking SNP names into account.
`out2.snp`	the output plain text file that stores the removed SNPs whose genomic positions are not in the imputation reference, taking SNP names into account.
`out3`	the prefix of the output PLINK binary files after removing SNPs whose genomic positions are not in the imputation reference, ingoring SNP names.
`out3.snp`	the output plain text file that stores the removed SNPs whose genomic positions are not in the imputation reference, ingoring SNP names.
`out4`	the prefix of the output PLINK binary files after removing SNPs whose alleles are not in the imputation reference, taking their genomic positions into account.
`out4.snp`	the output plain text file that stores the removed SNPs whose alleles are not in the imputation reference, taking their genomic positions into account.
`out4.snpRetained`	the output plain text file that stores the removed SNPs whose alleles are in the imputation reference, taking their genomic positions into account.
`nCore`	the number of cores used for computation. This can be tuned along with nThread.

Details

The output files are genrated in order. Genomic position includes chromosomal location and base-pair position of the individual variant. All monomorphic SNPs are retained for further processing.

Value

The set of aligned PLINK files from your own study compared with the imputation reference.

Author(s)

Junfang Chen

`chrWiseSplit` *Split genome-wide genotyping data into chromosome-wide PLINK binary files.*

Description

Split the whole genome genotyping data chromosome-wise; allow parallel computating for all chromosomes.

Usage

```
chrWiseSplit(plink, inputPrefix, chrXPAR1suffix, chrXPAR2suffix, nCore = 25)
```

Arguments

plink	an executable program in either the current working directory or somewhere in the command path.
inputPrefix	the prefix of the input PLINK binary files before splitting.
chrXPAR1suffix	if chromosome 25 is available and with PAR1, then generate the suffix with X_PAR1 for chrX_PAR1.
chrXPAR2suffix	if chromosome 25 is available and with PAR2, then generate the suffix with X_PAR2 for chrX_PAR2.
nCore	the number of cores used for parallel computation. The default value is 25.

Details

If chromosome 25 is also available, namely the pseudo-autosomal region of chromosome X, then further split chr25 (PAR or Chr_XY) into PAR1 and PAR2 according to the genomic coordination GRCh37 from https://en.wikipedia.org/wiki/Pseudoautosomal_region. The locations of the PARs within GRCh37 are: PAR1 X 60001 2699520; PAR2 X 154931044 155260560.

Value

The output PLINK binary files for each chromosome with the same prefix as the inputPrefix but appended with the chromosome codes, and possibly also the logical value for the pseudo-autosomal region (PAR) indicating if PAR exists in the input genotyping data or not.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "alignedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "alignedData.bim", package="Gimpute")
famFile <- system.file("extdata", "alignedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, " ."))
system(paste0("scp ", bimFile, " ."))
system(paste0("scp ", famFile, " ."))
inputPrefix <- "alignedData"
chrXPAR1suffix <- "X_PAR1"
chrXPAR2suffix <- "X_PAR2"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## chrWiseSplit(plink, inputPrefix, chrXPAR1suffix, chrXPAR2suffix, nCore)
```

chunk4eachChr *Chunk each chromosome into multiple segments*

Description

Chunk each chromosome genotyping data into multiple segments by a predefined window size.

Usage

```
chunk4eachChr(inputPrefix, outputPrefix, chrs, windowSize = 3e+06)
```

Arguments

inputPrefix	the prefix of the input PLINK .bim file for each chromosome, without the chromosome codes.
outputPrefix	the prefix of the output pure text files that keep all the chunks for each chromosome separately.
chrs	specifiy the chromosome codes for chunking.
windowSize	the window size of each chunk. The default value is 3000000.

Value

The output pure text files include all the chunks for each chromosome separately.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bimFile <- system.file("extdata", "gwas_data_chr23.bim", package="Gimpute")
system(paste0("scp ", bimFile, " ."))
inputPrefix <- "gwas_data_chr"
outputPrefix <- "chunks_chr"
chrs <- 23
print(chrs)
chunk4eachChr(inputPrefix, outputPrefix, chrs, windowSize=3000000)
```

extractByGenipe *Extract imputed markers using Genipe*

Description

Extract imputed markers located in a specific genomic region using Genipe. Note that, 1.) 'bed' PLINK binary format is specifically used for the output format. 2.) The markers of the whole chromosome are extracted together. For the filtering of maf and info will be done during post imputation.

Usage

```
extractByGenipe(inputImpute2, inputMAP, outputPrefix, format, prob)
```

Arguments

inputImpute2	the output from IMPUTE2.
inputMAP	the output PLINK MAP file from Genipe, which will be used for generating markers in a text file (only one column without column name).
outputPrefix	the prefix of the output files. [impute2_extractor]
format	the output format. Can specify either "impute2" for probabilities (same as impute2 format, i.e. 3 values per sample), "dosage" for dosage values (one value between 0 and 2 by sample), "calls" for hard calls, or "bed" for Plink binary format (with hard calls). [impute2]
prob	the probability threshold used when creating a file in the dosage or call format. [0.9]

Value

The extracted imputed files using genipe.

Author(s)

Junfang Chen

References

Lemieux Perreault, L. P., et al. (2016). genipe: an automated genome-wide imputation pipeline with automatic reporting and statistical tools. Bioinformatics, 32(23), 3661-3663.

---

genoQC *Quality control for genotype data*

Description

Perform quality control on the genotype data.

Usage

```
genoQC(plink, inputPrefix, snpMissCutOffpre = 0.05, sampleMissCutOff = 0.02,
  Fhet = 0.2, snpMissCutOffpost = 0.02, snpMissDifCutOff = 0.02,
  femaleChrXmissCutoff = 0.05, pval4autoCtl = 1e-06,
  pval4femaleXctl = 1e-06, outputPrefix, keepInterFile = TRUE)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`snpMissCutOffpre`	the cutoff of the missingness for removing SNPs before subject removal. The default is 0.05.
`sampleMissCutOff`	the cutoff of the missingness for removing subjects/instances. The default is 0.02.
`Fhet`	the cutoff of the autosomal heterozygosity deviation. The default is 0.2.
`snpMissCutOffpost`	the cutoff of the missingness for removing SNPs after subject removal. The default is 0.02.
`snpMissDifCutOff`	the cutoff of the difference in missingness between cases and controls. The default is 0.02.
`femaleChrXmissCutoff`	the cutoff of the missingness in female chromosome X SNPs. The default is 0.05.
`pval4autoCtl`	the p-value cutoff for controlling HWE test in either control or case subjects. Only autosomal SNPs are considered. The default is 0.000001
`pval4femaleXctl`	the p-value cutoff for controlling HWE test in female control subjects. Only chromosome X SNPs are considered. The default is 0.000001
`outputPrefix`	the prefix of the output PLINK binary files after QC.
`keepInterFile`	a logical value indicating if the intermediate processed files should be kept or not. The default is TRUE.

Details

The original PLINK files are implicitly processed by the following default steps: 1.) Set all heterozygous alleles of SNPs on male chrX as missing; 2.) SNP missingness < 0.05 (before sample removal); 3.) Subject missingness < 0.02; 4.) Remove subjects with |Fhet| >= 0.2; 5.) Reset paternal and maternal codes; 6.) SNP missingness < 0.02 (after sample removal); 7.) Remove SNPs with difference >= 0.02 of SNP missingness between cases and controls; 8.) Remove chrX SNPs with missingness >= 0.05 in females. (Optional, if no chrX data); 9.) Remove autosomal SNPs with HWE p < 10-6 in controls; 10.) Remove chrX SNPs with HWE p < 10-6 in female controls. (Optional, if no chrX data).

Value

The output PLINK binary files after QC.

Author(s)

Junfang Chen

References

Schizophrenia Working Group of the Psychiatric Genomics, C. (2014). Biological insights from 108 schizophrenia-associated genetic loci. Nature 511(7510): 421-427.

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "genoUpdatedData"
outputPrefix <- "2_12_removedSnpHweFemaleX"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## genoQC(plink, inputPrefix,
##        snpMissCutOffpre=0.05,
##        sampleMissCutOff=0.02,
##        Fhet=0.2,
##        snpMissCutOffpost=0.02,
##        snpMissDifCutOff=0.02,
##        femaleChrXmissCutoff=0.05,
##        pval4autoCtl=0.000001,
##        pval4femaleXctl=0.000001,
##        outputPrefix, keepInterFile=TRUE)
```

---

getGroupLabel *Get the outcome label of the genotype data*

Description

Get the group label from the PLINK FAM file.

Usage

```
getGroupLabel(inputFAMfile)
```

Arguments

`inputFAMfile` the PLINK FAM file.

Details

If the input FAM file also has missing outcomes, which are shown in the sixth column of FAM file as "0", then an error is given.

Value

The group label of the genotype data: "control" or "case" or "caseControl" indicating both groups exist.

Author(s)

Junfang Chen

Examples

```
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
getGroupLabel(inputFAMfile=famFile)
```

---

imputedByGenipe *Impute genotypes using Genipe*

Description

Perform imputation by Genipe for the autosomal and sex chromosome prephased known haplotypes with a reference panel. Note that pre-phasing using SHAPEIT is done without the refernce haplotypes.

Usage

```
imputedByGenipe(chrs, impRefDir, inputPrefix, shapeit, impute2, plink,
  fastaFile, segmentSize, thread4impute2, thread4shapeit)
```

Arguments

`chrs`	specifiy the chromosomes for imputation. There are four different options ("autosomes", "1, or 2, or 3...or 22", "23", "25"). 1.) 'autosomes': will impute chromosome 1 to 22 together; 2.) 'chrs' belongs to one of (1, 2,...22) then the imputation is done just for one autosomal chromosome. 3.) '23' will do the imputation for the non-pseudoautosomal region of chromosome 23. 4.) '25' imputes for the pseudoautosomal regions of chromosome 23.
`impRefDir`	the directory where the imputation reference files are located.
`inputPrefix`	the prefix of the input PLINK binary files.
`shapeit`	an executable SHAPEIT binary program in either the current working directory or somewhere in the command path.
`impute2`	an executable IMPUTE2 binary program in either the current working directory or somewhere in the command path.
`plink`	an executable program in either the current working directory or somewhere in the command path.
`fastaFile`	the human reference files for the initial strand check.
`segmentSize`	the length of a single segment for imputation.
`thread4impute2`	the number of threads for the imputation.
`thread4shapeit`	the number of threads for phasing.

Value

The imputed files using genipe.

Author(s)

Junfang Chen

References

Lemieux Perreault, L. P., et al. (2016). genipe: an automated genome-wide imputation pipeline with automatic reporting and statistical tools. Bioinformatics, 32(23), 3661-3663.

`mergeByGenipe` *Merge imputed files using Genipe*

Description

Concatenate IMPUTE2 output files and retrieve some statistics. This is automatically called by the main genipe pipeline to merge IMPUTE2 files generated for all the genomic segments. For details, see Genipe IMPUTE2 merger options.

Usage

```
mergeByGenipe(inputImpute2, chr, probability, completionRate, info,
  outputPrefix)
```

Arguments

`inputImpute2`	the output from IMPUTE2.
`chr`	specifiy the chromosome segment to be merged, on which the imputation was made.
`probability`	the probability threshold for no calls. [<0.9]
`completionRate`	the completion rate threshold for site exclusion. [<0.98]
`info`	the measure of the observed statistical information associated with the allele frequency estimate threshold for site exclusion. (<0.00)
`outputPrefix`	the prefix for the imputed output files.

Value

The merged imputed files using genipe.

Author(s)

Junfang Chen

References

Lemieux Perreault, L. P., et al. (2016). genipe: an automated genome-wide imputation pipeline with automatic reporting and statistical tools. Bioinformatics, 32(23), 3661-3663.

`phaseImpute2` *Phasing and imputation*

Description

Perform phasing, imputation and conversion from IMPUTE2 format into PLINK binary files.

Usage

```
phaseImpute2(inputPrefix, outputPrefix, prefix4final, plink, shapeit, impute2,
  gtool, windowSize = 3e+06, effectiveSize = 20000, nCore4phase = 1,
  nThread = 40, nCore4impute = 40, nCore4gtool = 40, infoScore = 0.6,
  outputInfoFile, impRefDIR, tmpImputeDir, keepTmpDir = TRUE)
```

Arguments

`inputPrefix`	the prefix of the input PLINK binary files for the imputation.
`outputPrefix`	the prefix of the output PLINK binary files after imputation and filtering out bad imputed variants.
`prefix4final`	the prefix of the output PLINK binary files after imputation.
`plink`	an executable program in either the current working directory or somewhere in the command path.
`shapeit`	an executable program in either the current working directory or somewhere in the command path.
`impute2`	an executable program in either the current working directory or somewhere in the command path.
`gtool`	an executable program in either the current working directory or somewhere in the command path.
`windowSize`	the window size of each chunk. The default value is 3000000.
`effectiveSize`	this parameter controls the effective population size. The default value is 20000.
`nCore4phase`	the number of cores used for phasing. This can be tuned along with nThread. The default value is 1
`nThread`	the number of threads used for computation. The default value is 40.
`nCore4impute`	the number of cores used for imputation. The default value is 40.
`nCore4gtool`	the number of cores used for computation. The default value is 40.
`infoScore`	the cutoff of filtering imputation quality score for each variant. The default value is 0.6.
`outputInfoFile`	the output file of impute2 info scores consisting of two columns: all imputed SNPs and their info scores.
`impRefDIR`	the directory where the imputation reference files are located.
`tmpImputeDir`	the name of the temporary directory used for storing phasing and imputation results.
`keepTmpDir`	a logical value indicating if the directory 'tmpImputeDir' should be kept or not. The default is TRUE.

Details

The whole imputation process mainly consists of the following steps: 1.) Phasing the input PLINK data using an existing imputation reference; 2.) Imputing the input PLINK data using phased results and an existing reference data; 3.) Converting IMPUTE2 format data into PLINK format. 4.) Combining all imputed data into whole-genome PLINK binary files. 5.) Filtering out imputed variants with bad imputation quality.

Value

1.) The filtered imputed PLINK binary files; 2.) The final PLINK binary files including bad imputed variants; 3.) A pure text file contains the info scores of all imputed SNPs with two columns: SNP names and the corresponding info scores.

Author(s)

Junfang Chen

References

1. Howie, B., et al. (2012). Fast and accurate genotype imputation in genome-wide association studies through pre-phasing. Nat Genet 44(8): 955-959.

2. Howie, B. N., et al. (2009). A flexible and accurate genotype imputation method for the next generation of genome-wide association studies. PLoS Genet 5(6): e1000529.

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "alignedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "alignedData.bim", package="Gimpute")
famFile <- system.file("extdata", "alignedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, " ."))
system(paste0("scp ", bimFile, " ."))
system(paste0("scp ", famFile, " ."))
inputPrefix <- "alignedData"
outputPrefix <- "gwasImputedFiltered"
prefix4final <- "gwasImputed"
outputInfoFile <- "impute2infoUpdated.txt"
tmpImputeDir <- "tmpImpute"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## phaseImpute2(inputPrefix, outputPrefix, prefix4final,
##              plink, shapeit, impute2, gtool,
##              windowSize=3000000, effectiveSize=20000,
##              nCore4phase=1, nThread=40,
##              nCore4impute=40, nCore4gtool=40,
##              infoScore=0.6, outputInfoFile,
##              impRefDIR, tmpImputeDir, keepTmpDir=TRUE)
```

`plotPCA4plink` *Population outlier detection*

Description

Principle component analysis (PCA) on the genotype data is performed to detect population outliers, and the first two PCs are plotted for the visualization.

Usage

```
plotPCA4plink(gcta, inputPrefix, nThread = 20, outputPC4subjFile,
  outputPCplotFile)
```

Arguments

`gcta`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`nThread`	the number of threads used for computation. The default is 20.
`outputPC4subjFile`	the pure text file that stores all the subject IDs and their corresponding eigenvalues of the first two principle components.
`outputPCplotFile`	the plot file for visualizing the first two principle components of all investigated subjects.

Details

Before population outlier detection, it's better to perform QC on the genotype data. Only autosomal genotypes are used for principle component analysis.

Value

The output pure text file and plot file for storing first two principle components of study subjects.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "QCdata.bed", package="Gimpute")
bimFile <- system.file("extdata", "QCdata.bim", package="Gimpute")
famFile <- system.file("extdata", "QCdata.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "QCdata"
outputPC4subjFile <- "2_13_eigenvalAfterQC.txt"
outputPCplotFile <- "2_13_eigenvalAfterQC.png" ## png format
## Not run: Requires an executable program GCTA, e.g.
## gcta <- "/home/tools/gcta64"
```

```
## plotPCA4plink(gcta, inputPrefix, nThread=20,
##               outputPC4subjFile, outputPrefix)
```

`prepareAnnoFile4affy` *prepare Affymetrix chip annotation file*

Description

Prepare Affymetrix chip annotation file into the format of interest.

Usage

```
prepareAnnoFile4affy(inputFile, outputFile, chipType)
```

Arguments

`inputFile`	an input pure text file that contains the chip annotation information.
`outputFile`	an output pure text file that stores the chip annotation information in a user-defined format.
`chipType`	a string name defines the type of the chip annotation file: 'SNPIDstudy', and 'rsIDstudy'.

Details

If the chip annotation file is not available for your study, it can be downloaded from http://www.well.ox.ac.uk/~wrayner/st
The chip annotation file is organized into two different types:

1. If the snp name of your study genotype data starts with "SNP_", then the chip type "SNPID-study" is used; Usually, Affymetrix chip data belongs to this category. The prepared output annotation file must at least consist of the following column names: SNPIDstudy, rs, chr, pos, strand.
2. If the snp name of your study genotype data starts with "rs", then the chip type "rsIDstudy" is used; The prepared output annotation file must at least consist of the following column names: SNPIDstudy, rs, chr, pos, strand. Illumina chip is often specified in this format.

The column "strand" must only have two kinds of values "-" and "+". Variants with all other values should be excluded.

Value

a pure text file that stores the prepared chip annotation information in a user-defined format.

Author(s)

Junfang Chen

`removedDoubleProbes` *Remove duplicated SNPs*

Description

Remove duplicated SNPs that have same rs-names or duplicated genomic position.

Usage

```
removedDoubleProbes(plink, inputPrefix, chipAnnoFile, chipType,
  outputSNPdupFile, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`chipAnnoFile`	an input chip annotation file. If the chip annotation file is not available for your study, it can be downloaded from http://www.well.ox.ac.uk/~wrayner/strand/.
`chipType`	a string name defines the type of the chip annotation file: 'SNPIDstudy', and 'rsIDstudy'. The detail is described in `prepareAnnoFile4affy`.
`outputSNPdupFile`	a pure text file that stores the duplicated SNP IDs, which are detected by the use of the chip annotation file.
`outputPrefix`	the prefix of the output PLINK binary files.

Details

Duplicated SNPs have two levels of meaning: 1.) SNPs have same rs-names but different versions of SNP ID in chip annotation file. e.g. SNP-A IDs for Affymetrix chip. 2.) SNPs with duplicated genomic position: the combination of base pair position and chromosomal location.

Value

The output PLINK binary files after removing duplicated SNP IDs.

Author(s)

Junfang Chen

See Also

`prepareAnnoFile4affy`

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
chipAnnoFile <- system.file("extdata", "chipAnno.txt", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData"
chipType <- "rsIDstudy"
outputSNPdupFile <- "snpDup.txt"
outputPrefix <- "removedDoubleProbes"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedDoubleProbes(plink, inputPrefix, chipAnnoFile,
##                     chipType, outputSNPdupFile, outputPrefix)
```

---

removedExclProbe *Remove improper SNPs*

Description

Remove SNPs that may be duplicated, or with unexpected SNP names.

Usage

```
removedExclProbe(plink, inputPrefix, excludedProbeIdsFile, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`excludedProbeIdsFile`	a pure text file that stores the SNP IDs, one per line, which need to be removed. If it is null, then no SNPs are removed.
`outputPrefix`	the prefix of the output PLINK binary files.

Details

excludedProbeIdsFile should be defined in a plain text file in advance. Improper SNPs such as AFFX, cnvi etc.. with unexpect format must be excluded.

Value

The output PLINK binary files after removing unwanted SNP IDs.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
excludedProbeIdsFile <- system.file("extdata", "excludedProbeIDs.txt",
                                    package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData" ## Specify the input PLINK file prefix
outputPrefix <- "1_06_removedExclProbe"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedExclProbe(plink, inputPrefix, excludedProbeIdsFile, outputPrefix)
```

---

removedInstFhet *Remove subjects with abnormal autosomal heterozygosity deviation*

Description

Remove subjects with great autosomal heterozygosity deviation.

Usage

```
removedInstFhet(plink, Fhet, inputPrefix, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`Fhet`	the cutoff of the autosomal heterozygosity deviation.
`inputPrefix`	the prefix of the input PLINK binary files.
`outputPrefix`	the prefix of the output PLINK binary files.

Details

If the cutoff of the autosomal heterozygosity deviation is set to be greater than 0.2, i.e. |Fhet| >= 0.2, then this analysis will automatically skip haploid markers (male X and Y chromosome markers).

Value

The output PLINK binary files after removing subjects with great autosomal heterozygosity deviation.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
Fhet <- 0.2
inputPrefix <- "genoUpdatedData"
outputPrefix <- "2_06_removedInstFhet"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedInstFhet(plink, Fhet, inputPrefix, outputPrefix)
```

---

removedInstMiss *Remove subjects with missing values*

Description

Remove Subjects or instances with missingness of greater than a certain threshold.

Usage

```
removedInstMiss(plink, sampleMissCutOff, inputPrefix, outputPrefix)
```

Arguments

plink	an executable program in either the current working directory or somewhere in the command path.
sampleMissCutOff	the cutoff of the missingness for removing subjects/instances.
inputPrefix	the prefix of the input PLINK binary files.
outputPrefix	the prefix of the output PLINK binary files.

Value

The output PLINK binary files after removing subjects with a pre-defined removal cutoff.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
sampleMissCutOff <- 0.02
inputPrefix <- "genoUpdatedData"
outputPrefix <- "2_05_removedInstMiss"
```

```
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedInstMiss(plink, sampleMissCutOff, inputPrefix, outputPrefix)
```

`removedMaleHetX` *Remove male subjects with haploid heterozygous SNPs*

Description

Determine the frequency of male subjects that have heterozygous SNPs on chromosome X and a reasonable cutoff to remove those affect males, if chromosome X data exists.

Usage

```
removedMaleHetX(plink, inputPrefix, hhSubjCutOff, outputPrefix,
  outputSubjHetFile, outputRetainSubjectFile, outputHetSNPfile)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`hhSubjCutOff`	the cutoff for removing male subjects with haploid heterozygous SNPs on the chromosome X.
`outputPrefix`	the prefix of the output PLINK binary files.
`outputSubjHetFile`	the output pure text file that stores male subjects that have heterozygous SNPs with their frequency (if any), i.e. the number of .hh SNPs in this male. Lines are sorted by descending number.
`outputRetainSubjectFile`	the output pure text file that stores male subjects that have heterozygous SNPs with their frequency after subject removal (if any). Lines are sorted by descending number.
`outputHetSNPfile`	the output pure text file that stores all heterozygous SNPs with their frequency (the number of males for this SNP) , if any. Lines are sorted by descending number.

Details

A haploid heterozygous is a male genotype that is heterozygous, which could be an error given the haploid nature of the male X chromosome. In principle, one has to remove all males that have heterozygous SNPs on the chromosome X. However, too many males might be removed in some data sets. Therefore a small percentage of such males in the data set is allowed.

Value

1.) The output PLINK binary files. 2.) A pure text file with two columns: heterozygous male subjects and their corresponding heterozygous SNPs. 3.) After subject removal, a pure text file consisting of two columns: heterozygous male subjects and their corresponding heterozygous SNPs. A pure text file with two columns: all heterozygous SNPs and their frequency.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "genoUpdatedData"
hhSubjCutOff <- 15 ##  can be tuned
outputPrefix <- "2_02_removedInstHetX"
outputSubjHetFile <- "2_02_instHetXfreqAll.txt"
outputRetainSubjectFile <- "2_02_instHetXfreqRetained.txt"
outputHetSNPfile <- "2_02_snpHHfreqAll.txt"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedMaleHetX(plink, inputPrefix, hhSubjCutOff,
##                 outputPrefix, outputSubjHetFile,
##                 outputRetainSubjectFile, outputHetSNPfile)
```

---

removedMonoSnp *Exclude monomorphic SNPs*

Description

Detect monomorphic SNPs from PLINK BIM file and exclude them if any.

Usage

```
removedMonoSnp(plink, inputPrefix, outputPrefix, outputSNPfile)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`outputPrefix`	the prefix of the output PLINK binary files after removing monomorphic SNPs.
`outputSNPfile`	the output pure text file that stores the removed monomorphic SNPs, one per line, if any.

Value

The output PLINK binary files after removing monomorphic SNPs and a pure text file with removed monomorphic SNPs.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "alignedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "alignedData.bim", package="Gimpute")
famFile <- system.file("extdata", "alignedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, " ."))
system(paste0("scp ", bimFile, " ."))
system(paste0("scp ", famFile, " ."))
inputPrefix <- "alignedData"
outputPrefix <- "removedMonoSnp"
outputSNPfile <- "monoSNP.txt"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedMonoSnp(plink, inputPrefix, outputPrefix, outputSNPfile)
```

`removedParentIdsMiss`

Reset paternal and maternal codes

Description

Reset paternal and maternal codes of non-founders if parents not present. Replace the paternal ID and maternal ID of subjects (childs) by the value zero if the paternal ID and the maternal ID do not belong to any subject (parent) with the same family ID as the child.

Usage

```
removedParentIdsMiss(plink, inputPrefix, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`outputPrefix`	the prefix of the output PLINK binary files.

Details

By default, if parental IDs are provided for a sample, they are not treated as a founder even if neither parent is in the dataset. With no modifiers, –make-founders clears both parental IDs whenever at least one parent is not in the dataset, and the affected samples are now considered founders.

Value

The output PLINK binary files.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "genoUpdatedData"
outputPrefix <- "2_07_removedParentIdsMiss"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedParentIdsMiss(plink, inputPrefix, outputPrefix)
```

---

removedSnpFemaleChrXhweControl

Hardy Weinberg Equilibrium test for chromosome X SNPs in female controls.

Description

Hardy Weinberg Equilibrium test for SNPs on the chromosome X in female controls.

Usage

```
removedSnpFemaleChrXhweControl(plink, inputPrefix, pval = 1e-06,
  outputPvalFile, outputSNPfile, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`pval`	the p-value cutoff for controlling HWE test in female control subjects. Only chromosome X SNPs are considered. The default value is 0.000001.
`outputPvalFile`	the output pure text file that stores chromosome X SNPs and their sorted HWE p-values.
`outputSNPfile`	the output pure text file that stores the removed SNPs, one per line.
`outputPrefix`	the prefix of the output PLINK binary files.

Value

The output PLINK binary files after HWE test on chromosomal X in female controls.

Author(s)

Junfang Chen

See Also

`removedSnpHWEauto`

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "genoUpdatedData"
outputPvalFile <- "2_12_snpHwePvalfemaleXct.txt"
outputSNPfile <- "2_12_snpRemovedHweFemaleXct.txt"
outputPrefix <- "2_12_removedSnpHweFemaleX"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedSnpFemaleChrXhweControl(plink, inputPrefix, pval=0.000001,
##                                outputPvalFile, outputSNPfile,
##                                outputPrefix)
```

---

removedSnpFemaleChrXmiss

remove chromosome X SNPs in females

Description

Remove SNPs on the chromosome X with a pre-defined cutoff for missingness in females.

Usage

```
removedSnpFemaleChrXmiss(plink, femaleChrXmissCutoff, inputPrefix, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`femaleChrXmissCutoff`	the cutoff of the missingness in female chromosome X SNPs.
`inputPrefix`	the prefix of the input PLINK binary files.
`outputPrefix`	the prefix of the output PLINK binary files.

Value

The output PLINK binary files.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
femaleChrXmissCutoff <- 0.05
inputPrefix <- "genoUpdatedData"
outputPrefix <- "2_10_removedSnpFemaleChrXmiss"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedSnpFemaleChrXmiss(plink, femaleChrXmissCutoff,
##                          inputPrefix, outputPrefix)
```

---

removedSnpHetX *Remove heterozygous SNPs in male chromosome X*

Description

Remove heterozygous SNPs in haploid male chromosome X only if chromosome X data exists.

Usage

```
removedSnpHetX(plink, inputPrefix, hhCutOff, outputPrefix, outputHetSNPfile,
  outputRetainSNPfile)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`hhCutOff`	the cutoff for removing male haploid heterozygous SNPs on the chromosome X.
`outputPrefix`	the prefix of the output PLINK binary files.
`outputHetSNPfile`	the output pure text file that stores all heterozygous SNPs with their frequency (the number of males for the corresponding SNP), if any. Lines are sorted by descending number.
`outputRetainSNPfile`	the output pure text file that stores retained heterozygous SNPs with their frequency (the number of males for the corresponding SNP), if any. Lines are sorted by descending number.

Details

A haploid heterozygous is a male genotype that is heterozygous, which could be an error given the haploid nature of the male X chromosome. In principle, one has to remove all heterozygous SNPs of chromosome X in males. However, too many SNPs might be removed in some data sets. Therefore a small percentage of such SNPs in the data set is allowed.

Value

1.) The output PLINK binary files. 2.) A pure text files (if any) with two columns: SNPs and their corresponding frequency. 3.) After SNP removal, a pure text files (if any) with two columns: SNPs and their corresponding frequency.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "genoUpdatedData"
hhCutOff <- 0.005 ##  can be tuned
outputPrefix <- "2_01_removedSnpHetX"
outputHetSNPfile <- "2_01_snpHHfreqAll.txt"
outputRetainSNPfile <- "2_01_snpHHfreqRetained.txt"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedSnpHetX(plink, inputPrefix, hhCutOff, outputPrefix,
##                outputHetSNPfile, outputRetainSNPfile)
```

---

removedSnpHWEauto *Hardy Weinberg Equilibrium test for autosomal SNPs*

Description

Remove autosomal SNPs deviating from Hardy Weinberg Equilibrium (HWE).

Usage

```
removedSnpHWEauto(groupLabel, plink, inputPrefix, pval = 1e-06,
  outputPvalFile, outputSNPfile, outputPrefix)
```

Arguments

`groupLabel`	a string value indicating the outcome label: "control", or, "case" or "caseControl" for both existing groups. For more details, see `getGroupLabel`.
`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`pval`	the p-value cutoff for controlling HWE test in either control or case subjects. Only autosomal SNPs are considered. The default value is 0.000001.
`outputPvalFile`	the output pure text file that stores autosomal SNPs and their sorted HWE p-values.
`outputSNPfile`	the output pure text file that stores the removed SNPs, one per line.
`outputPrefix`	the prefix of the output PLINK binary files.

Value

The output PLINK binary files after HWE test on the autosome.

Author(s)

Junfang Chen

See Also

removedSnpFemaleChrXhweControl, getGroupLabel.

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
groupLabel <- "control"
inputPrefix <- "genoUpdatedData" ## Specify the input PLINK file prefix
outputPvalFile <- "2_11_snpHwePvalAuto.txt"
outputSNPfile <- "2_11_snpRemovedHweAuto.txt"
outputPrefix <- "2_11_removedSnpHweAuto"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedSnpHWEauto(groupLabel, plink, inputPrefix, pval=0.000001,
##                   outputPvalFile, outputSNPfile, outputPrefix)
```

---

removedSnpMiss *Remove SNPs with missing values*

Description

Remove SNPs with missingness of greater than a certain threshold.

Usage

```
removedSnpMiss(plink, snpMissCutOff, inputPrefix, outputPrefix)
```

Arguments

plink	an executable program in either the current working directory or somewhere in the command path.
snpMissCutOff	the cutoff of the missingness for removing SNPs.
inputPrefix	the prefix of the input PLINK binary files.
outputPrefix	the prefix of the output PLINK binary files.

Value

The output PLINK binary files after removing SNPs with pre-defined missing values.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
snpMissCutOff <- 0.05
inputPrefix <- "genoUpdatedData"
outputPrefix <- "2_04_removedSnpMissPre"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedSnpMiss(plink, snpMissCutOff, inputPrefix, outputPrefix)
```

---

removedSnpMissDiff *Remove SNPs with difference in SNP missingness between cases and controls.*

Description

Remove SNPs with difference in SNP missingness between cases and controls. To test for differential call rates between cases and controls for each SNP

Usage

```
removedSnpMissDiff(plink, inputPrefix, snpMissDifCutOff, outputPrefix,
  groupLabel)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`snpMissDifCutOff`	the cutoff of the difference in missingness between cases and controls.
`outputPrefix`	the prefix of the output PLINK binary files.
`groupLabel`	a string value indicating the outcome label: "control", or, "case" or "caseControl" for both existing groups. For more details, see `getGroupLabel`.

Details

Only if both case-control groups exist in the input genotype data, differential SNPs are removed.

Value

The output PLINK binary files.

Author(s)

Junfang Chen

See Also

getGroupLabel.

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "genoUpdatedData"
snpMissDifCutOff <- 0.02
outputPrefix <- "2_09_removedSnpMissDiff"
groupLabel <- "control"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedSnpMissDiff(plink, inputPrefix, snpMissDifCutOff,
##                    outputPrefix, groupLabel)
```

---

removedSnpMissPostImp

Remove SNPs after post imputation

Description

Remove SNPs which have a non missing value for less than a predefined number of instances.

Usage

```
removedSnpMissPostImp(plink, inputPrefix, missCutoff, outputSNPfile,
  outputPrefix)
```

Arguments

plink	an executable program in either the current working directory or somewhere in the command path.
inputPrefix	the prefix of the input PLINK binary files.
missCutoff	the cutoff of the least number of instances for a SNP that is not missing. The default is 20.
outputSNPfile	the output file of SNPs with pre-defined missing values.
outputPrefix	the prefix of the PLINK binary files.

Value

The PLINK binary files after post imputation quality control and a pure text file contains SNPs with pre-defined missing values.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "alignedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "alignedData.bim", package="Gimpute")
famFile <- system.file("extdata", "alignedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, " ."))
system(paste0("scp ", bimFile, " ."))
system(paste0("scp ", famFile, " ."))
inputPrefix <- "alignedData"
outputPrefix <- "removedSnpMissPostImp"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedSnpMissPostImp(plink, inputPrefix, missCutoff=20, outputPrefix)
```

---

removedUnmapProbes *Remove SNPs not in the chip annotation file*

Description

Check if all SNPs are included in the chip annotation file. If some of the input SNPs are not included, then remove them.

Usage

```
removedUnmapProbes(plink, inputPrefix, chipAnnoFile, chipType, outputPrefix,
  outputSNPfile)
```

Arguments

plink	an executable program in either the current working directory or somewhere in the command path.
inputPrefix	the prefix of the input PLINK binary files.
chipAnnoFile	a pure text file that stores the chip annotation information.
chipType	a string name defines the type of the chip annotation file: 'SNPIDstudy', and 'rsIDstudy'. The detail is described in prepareAnnoFile4affy.
outputPrefix	the prefix of the output PLINK binary files.
outputSNPfile	a pure text file that stores the SNP IDs, one per line, which are not mapped to the chip annotation file.

Details

If the chip annotation file is not available for your study, you can download it from http://www.well.ox.ac.uk/~wrayner/str

Value

The output text file contains the removed SNP IDs, one per line. The PLINK binary files after removing unmapped SNP IDs.

Author(s)

Junfang Chen

See Also

prepareAnnoFile4affy

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
chipAnnoFile <- system.file("extdata", "chipAnno.txt", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData" ## Specify the input PLINK file prefix
outputSNPfile <- "1_07_removedUnmapProbes"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedUnmapProbes(plink, inputPrefix, chipAnnoFile, chipType,
##                    outputPrefix, outputSNPfile)
```

---

removedWrongAnceInst

Remove samples with incorrect ancestry

Description

Remove samples with the incorrect ancestry or keep samples at your own chioce.

Usage

```
removedWrongAnceInst(plink, inputPrefix, metaDataFile, ancestrySymbol,
  outputPrefix)
```

Arguments

plink	an executable program in either the current working directory or somewhere in the command path.
inputPrefix	the prefix of the input PLINK binary files.
metaDataFile	a pure text file that stores the meta information of the samples. This file must contain at least the following content (column names are in parentheses): family ID in the PLINK files (FID), individual ID in the PLINK files (IID), ID in the description files (descID), self identified ancestry (ance; e.g. AFR: African, AMR: Ad Mixed American, EAS: East Asian, EUR: European, SAS: South Asian), sex (sex; 1 = male, 2 = female), age (age), group (group; 0 = control/unaffected, 1 = case/affected). All unknown and missing values are represented by the value NA. Lines with a missing value for FID or IID are not contained.
ancestrySymbol	an indicator that shows the symbol of genetic ancestry. If it is null, then all samples are selected.
outputPrefix	the prefix of the output PLINK binary files.

Details

ancestrySymbol, such as 'EUR' stands for the European, 'EAS' for East Asian. See the metaDataFile for more details.

Value

The output PLINK binary files after checking the ancestry information.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
metaDataFile <- system.file("extdata", "1_01_metaData.txt",
                            package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData" ## Specify the input PLINK file prefix
ancestrySymbol <- "EAS"
outputPrefix <- "1_05_removedWrongAnceInst"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedWrongAnceInst(plink, inputPrefix, metaDataFile,
##                      ancestrySymbol, outputPrefix)
```

---

removedYMtSnp *Remove SNPs on the chromosome Y and mitochondrial DNA*

Description

Remove SNPs on the chromosome Y and mitochondrial DNA.

Usage

```
removedYMtSnp(plink, inputPrefix, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`outputPrefix`	the prefix of the output PLINK binary files.

Details

Note that if chromosome Y and mitochondrial DNA are available, they must be coded as 24 and 26, respectively.

Value

The output PLINK binary files after removing SNPs on the chromosome Y and mitochondrial DNA.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData" ## Specify the input PLINK file prefix
outputPrefix <- "1_11_removedYMtSnp"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removedYMtSnp(plink, inputPrefix, outputPrefix)
```

---

removeNoGroupId *Remove samples without group information*

Description

Remove samples without group/outcome/phenotype information, which is coded as -9 in the PLINK .FAM file.

Usage

```
removeNoGroupId(plink, inputPrefix, outputPrefix)
```

Arguments

plink	an executable program in either the current working directory or somewhere in the command path.
inputPrefix	the prefix of the input PLINK binary files.
outputPrefix	the prefix of the output PLINK binary files.

Value

The output PLINK binary files after removing samples without group information.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData" ## Specify the input PLINK file prefix
outputPrefix <- "1_04_removedNoGroupId"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removeNoGroupId(plink, inputPrefix, outputPrefix)
```

---

removeOutlierByPCs *Remove population outliers*

Description

Remove population outliers by using principle component analysis.

Usage

```
removeOutlierByPCs(plink, gcta, inputPrefix, nThread = 20, cutoff, cutoffSign,
  inputPC4subjFile, outputPC4outlierFile, outputPCplotFile, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`gcta`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`nThread`	the number of threads used for computation. The default is 20.
`cutoff`	the cutoff that distinguishes the eigenvalues of the outliers from ordinary population. If it is null, then there are no outliers or outliers are not required to be removed.
`cutoffSign`	the cutoff sign: 'greater' or 'smaller' that determines if the outliers should be greater or smaller than the cutoff value.
`inputPC4subjFile`	the pure text file that stores all the subject IDs and their corresponding eigenvalues of the first two principle components.
`outputPC4outlierFile`	the pure text file that stores the outlier IDs and their corresponding eigenvalues of the first two principle components.
`outputPCplotFile`	the plot file for visualizing the first two principle components of all subjects without population outliers.
`outputPrefix`	the prefix of the output PLINK binary files.

Details

This function is used for removing population outliers. If the outliers are necessary to be removed, then one uses the eigenvalues from the first principle component as a criterion to find out the outliers by assigning an appropriate cutoff.

Value

1.) The output PLINK binary files after outlier removal. 2.) The output pure text file (if any) for storing removed outlier IDs and their corresponding PCs. 3.) The plot file (if any) for visualizing the first two principle components after outlier removal.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "QCdata.bed", package="Gimpute")
bimFile <- system.file("extdata", "QCdata.bim", package="Gimpute")
famFile <- system.file("extdata", "QCdata.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "QCdata"
cutoff <- NULL ## no outlier to be removed
cutoffSign <- "greater" ## not used if cutoff == NULL
inputPC4subjFile <- "2_13_eigenvalAfterQC.txt"
outputPC4outlierFile <- "2_13_eigenval4outliers.txt"
outputPCplotFile <- "2_13_removedOutliers.png"
outputPrefix <- "2_13_removedOutliers"
## Not run: Requires an executable program PLINK and GCTA, e.g.
## plink <- "/home/tools/plink"
## gcta <- "/home/tools/gcta64"
## removeOutlierByPCs(plink, gcta, inputPrefix, nThread=20,
##                    cutoff, cutoffSign, inputPC4subjFile,
##                    outputPC4outlierFile, outputPCplotFile, outputPrefix)
```

---

removeSampID *Remove samples in PLINK files*

Description

Remove sample IDs that are useless such as duplicated or related IDs from PLINK binary files. These IDs are defined in a plain text file.

Usage

```
removeSampID(plink, removedSampIDFile, inputPrefix, outputPrefix)
```

Arguments

plink	an executable program in either the current working directory or somewhere in the command path.
removedSampIDFile	a pure text file that stores the useless sample IDs, each ID per line. If it is null, then duplicate the input PLINK files as the output files.
inputPrefix	the prefix of the input PLINK binary files.
outputPrefix	the prefix of the output PLINK binary files.

Value

The output PLINK binary files after removing certain sample IDs.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData" ## Specify the input PLINK file prefix
removedSampIDFile <- system.file("extdata", "excludedSampIDs.txt",
                                 package="Gimpute")
outputPrefix <- "1_02_removedExclInst"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removeSampID(plink, removedSampIDFile, inputPrefix, outputPrefix)
```

---

renamePlinkBFile *Rename PLINK binary files*

Description

Rename a set of PLINK binary files (.BED, .BIM and .FAM).

Usage

```
renamePlinkBFile(inputPrefix, outputPrefix, action)
```

Arguments

inputPrefix	the prefix of the input PLINK binary files.
outputPrefix	the prefix of the output PLINK binary files.
action	a string indicating if the action is "copy" or "move".

Details

The original input files can be retained using the action "copy" or removed by using "move".

Value

Renamed PLINK binary files.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData"
outputPrefix <- "dataCtl"
outputPrefix <- "1_02_removedExclInst"
## renamePlinkBFile(inputPrefix, outputPrefix, action="move")
```

---

setHeteroHaploMissing

Set haploid heterozygous SNPs as missing

Description

Set all heterozygous alleles of chromosome X SNPs in male as missing.

Usage

```
setHeteroHaploMissing(plink, inputPrefix, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`outputPrefix`	the prefix of the output PLINK binary files.

Value

The output PLINK binary files after setting haploid heterozygous SNPs as missing.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "genoUpdatedData.bed", package="Gimpute")
bimFile <- system.file("extdata", "genoUpdatedData.bim", package="Gimpute")
famFile <- system.file("extdata", "genoUpdatedData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "genoUpdatedData"
outputPrefix <- "2_03_setHeteroHaploMissing"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## setHeteroHaploMissing(plink, inputPrefix, outputPrefix)
```

---

splitXchr — *Split chromosome X into pseudoautosomal region and non-pseudoautosomal region.*

Description

Split chromosome X into pseudoautosomal region and non-pseudoautosomal region, if chromosome X data is available.

Usage

```
splitXchr(plink, inputPrefix, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`outputPrefix`	the prefix of the output PLINK binary files.

Details

Genomic coordinate system is on genome build hg19.

Value

The output PLINK binary files after splitting chromosome X into pseudoautosomal region and non-pseudoautosomal region.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData" ## Specify the input PLINK file prefix
outputPrefix <- "1_10_splitXchr"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## removeSampID(plink, inputPrefix, outputPrefix)
```

---

updatedSnpInfo *Update the SNP information*

Description

Update SNP information including SNP name, base-pair position, chromosomal location and the strand information.

Usage

```
updatedSnpInfo(plink, inputPrefix, chipAnnoFile, chipType, outputPrefix)
```

Arguments

plink	an executable program in either the current working directory or somewhere in the command path.
inputPrefix	the prefix of the input PLINK binary files.
chipAnnoFile	a pure text file that stores the chip annotation information. If the chip annotation file is not available for your study, it can be downloaded from http://www.well.ox.ac.uk/~wrayner/stra
chipType	a string name defines the type of the chip annotation file: 'SNPIDstudy', and 'rsIDstudy'. The detail is described in prepareAnnoFile4affy.
outputPrefix	the prefix of the output PLINK binary files.

Details

The SNP information in the chip annotation file is used as the reference.

Value

The output PLINK binary files after updating SNP information.

Author(s)

Junfang Chen

See Also

prepareAnnoFile4affy

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
chipAnnoFile <- system.file("extdata", "chipAnno.txt", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData"
chipType <- "rsIDstudy"
outputPrefix <- "updatedSnpInfo"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## updatedSnpInfo(plink, inputPrefix, chipAnnoFile,
##                chipType, outputPrefix, outputSNPfile)
```

---

updateGenoInfo *Update genotype information*

Description

Update genotype information of the original PLINK binary files involving subject metadata information remapping and SNP information rearrangement and conversion according to the annotation file.

Usage

```
updateGenoInfo(plink, inputPrefix, metaDataFile, removedSampIDFile,
  ancestrySymbol, excludedProbeIdsFile, chipAnnoFile, chipType, outputPrefix,
  keepInterFile = TRUE)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`metaDataFile`	a pure text file that stores the meta information of the samples. This file must contain at least the following content (column names are in parentheses): family ID in the PLINK files (FID), individual ID in the PLINK files (IID), ID in the description files (descID), self identified ancestry (ance; e.g. AFR: African, AMR: Ad Mixed American, EAS: East Asian, EUR: European, SAS: South Asian), sex (sex; 1 = male, 2 = female), age (age), group (group; 0 = control/unaffected, 1 = case/affected). All unknown and missing values are represented by the value NA. Lines with a missing value for FID or IID are not contained.
`removedSampIDFile`	a pure text file that stores the useless sample IDs, each ID per line. If it is null, then duplicate the input PLINK files from the last step as the output files.
`ancestrySymbol`	an indicator that shows the symbol of genetic ancestry. If it is null, then all samples are selected.

`excludedProbeIdsFile`
: a pure text file that stores the SNP IDs, one per line, which need to be removed. If it is null, no SNPs are removed.

`chipAnnoFile`
: a pure text file that stores the chip annotation information.

`chipType`
: a string name defines the type of the chip annotation file: 'SNPIDstudy', and 'rsIDstudy'. The detail is described in `prepareAnnoFile4affy`.

`outputPrefix`
: the prefix of the output PLINK binary files.

`keepInterFile`
: a logical value indicating if the intermediate processed files should be kept or not. The default is TRUE.

Details

The original PLINK files are implicitly processed by the following steps: 1.) remove duplicated subjects; 2.) update group ID and sex information; 3.) remove not labelled subjects; 4.) remove subjects with wrong ancestry; 5.) remove incorrectly annotated SNPs; 6.) remove SNPs that are not in the annotation file; 7.) remove duplicated SNPs; 8.) update SNP genomic position and strand information; 9.) split chromosome X into pseudoautosomal region (PAR) and non-PAR; 10.) remove SNPs on the chromosome Y and mitochondrial DNA. The metadata information file and the chip annotation file are used as the reference for the update. If the chip annotation file is not available for your study, it can be downloaded from http://www.well.ox.ac.uk/~wrayner/strand/.

Value

The output PLINK binary files after genotype information remapping.

Author(s)

Junfang Chen

References

Purcell, Shaun, et al. PLINK: a tool set for whole-genome association and population-based linkage analyses. The American Journal of Human Genetics 81.3 (2007): 559-575.

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData"
metaDataFile <- system.file("extdata", "1_01_metaData.txt",
                            package="Gimpute")
excludedProbeIdsFile <- system.file("extdata", "excludedProbeIDs.txt",
                                    package="Gimpute")
removedSampIDFile <- system.file("extdata", "excludedSampIDs.txt",
                                 package="Gimpute")
chipAnnoFile <- system.file("extdata", "chipAnno.txt", package="Gimpute")
ancestrySymbol <- "EUR"
outputPrefix <- "1_11_removedYMtSnp"
metaDataFile <- "1_01_metaData.txt"
chipType <- "rsIDstudy"
```

```
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## updateGenoInfo(plink, inputPrefix, metaDataFile, removedSampIDFile,
##                ancestrySymbol, excludedProbeIdsFile, chipAnnoFile,
##                chipType, outputPrefix, keepInterFile=TRUE)
```

`updateGroupIdAndSex` *Update group and geneder information*

Description

Replace group and gender information in the PLINK binary files by using the information from the metadata file.

Usage

```
updateGroupIdAndSex(plink, inputPrefix, metaDataFile, outputPrefix)
```

Arguments

`plink`	an executable program in either the current working directory or somewhere in the command path.
`inputPrefix`	the prefix of the input PLINK binary files.
`metaDataFile`	a pure text file that stores the meta information of the samples. This file must contain at least the following content (column names are in parentheses): family ID in the PLINK files (FID), individual ID in the PLINK files (IID), ID in the description files (descID), self identified ancestry (ance; e.g. AFR: African, AMR: Ad Mixed American, EAS: East Asian, EUR: European, SAS: South Asian), sex (sex; 1 = male, 2 = female), age (age), group (group; 0 = control/unaffected, 1 = case/affected). All unknown and missing values are represented by the value NA. Lines with a missing value for FID or IID are not contained.
`outputPrefix`	the prefix of the output PLINK binary files.

Details

Find the shared sample IDs between PLINK input files and metadata file. Use the information from the metadata file as the reference and update the group information/the outcome in the PLINK file. Group label should be 1 and 2. (1=unaff, 2=aff, 0=miss); missing phenotype will be indicated as -9.

Value

The output PLINK binary files after updating the gender and grouping information.

Author(s)

Junfang Chen

Examples

```
## In the current working directory
bedFile <- system.file("extdata", "controlData.bed", package="Gimpute")
bimFile <- system.file("extdata", "controlData.bim", package="Gimpute")
famFile <- system.file("extdata", "controlData.fam", package="Gimpute")
metaDataFile <- system.file("extdata", "1_01_metaData.txt",
                            package="Gimpute")
system(paste0("scp ", bedFile, bimFile, famFile, " ."))
inputPrefix <- "controlData" ## Specify the input PLINK file prefix
outputPrefix <- "1_03_replacedGroupAndSex"
## Not run: Requires an executable program PLINK, e.g.
## plink <- "/home/tools/plink"
## updateGroupIdAndSex(plink, inputPrefix, metaDataFile, outputPrefix)
```

Index