Package 'fGWAS'

October 8, 2018

Version 0.3.5

Date Oct 8, 2018

Title Functional Genome-wide Association Study

Author Zhong Wang, Center for Computational Biology at Beijing Forest University.

Maintainer Zhong Wang <wzhy2000@hotmail.com>

Description Analysis genetic effects to detect the significant SNPs associated with longitudinal traits based on fGWAS method.

Depends
R (>= 2.6.0), minpack.lm, snpStats, mvtnorm, parallel, methods, stats, graphics, grDevices, utils

License GPL-3

Suggests snow

URL http://www.github.com/wzhy2000/fgwas

LazyLoad no

R topics documented:

fg.adjust.inflation *Adjusting the p-values based on the LR distribution*

Description

Adjusting the p-values based on the LR distribution. Notice: it is applicable for the results of whole genome when the genome inflaion is not normal.

Usage

```
fg.adjust.inflation(object)
```

Arguments

object	A S3 object of scaning result returned by fg.snpscan.

Value

A new S3 object with adjusted p-values.

fg.get.pca *Calculating PCA matrix using PLINK software*

Description

Calculating PCA matrix for a genotype object using PLINK software.

Usage

```
fg.get.pca(object, plink.path)
```

Arguments

object	S3 object fgwas.gen.obj, indicating the genotype object generated by the fg.load.plink, fg.load.simple or fg.simulate.
plink.path	string, indicating PLINK command path, for the large PLINK data, the package loads partial PLINK data extracted by the 'plink' command rather than the whole data set.

Details

This function returns a matrix containing 20 PCAs which can be used as the covariates in fGWAS. Please specify the location of PLINK in the function calling.

Value

A matrix with 20 PCAs in columns.

Examples

```
obj1 <- fg.simulate( "Logistic", "AR1", 2000, 1000, 1:8,
    phe.missing=0.05, snp.missing=0.05,
    sig.pos=501, plink.format=FALSE, file.prefix = NULL );

m <- fg.get.pca( obj1$obj.gen, "~/lib/plink1.9/plink");
head(m)

#           PCA1         PCA2        PCA3       PCA4         PCA5         PCA6
#N_1  0.01379520 -0.017665100 -0.00619801 -0.0262379  0.008060390 -0.007591340
#N_2 -0.02162890 -0.000899263  0.01792060  0.0492322 -0.006867650 -0.007675600
#...
```

`fg.load.phenotype` *Loading phenotypic data*

Description

Loading longitudinal phenotypic curve and covariate files.

Usage

```
fg.load.phenotype(file.phe.long, file.phe.cov=NULL, file.phe.time=NULL,
    curve.type = NULL,
    covariance.type = NULL,
    file.plot.pdf = NULL,
    intercept = TRUE,
    options = list(verbose=F))
```

Arguments

`file.phe.long`	string, indicating the file name of longitudinal curve for each individual.
`file.phe.cov`	string, indicating the file name of covariates.
`file.phe.time`	string, indicating the file name of observed time points for each individual. If this file is not specified, the column indexes are used as measured times.
`curve.type`	string, indicating the curve type, available options are in the details. Default is 'auto' which means the package selects the curve type using curve fitting.
`covariance.type`	string, indicating the covariance structure type, available options are in the details. Default is 'auto' which means the package selects the covariance type using MLE.
`intercept`	boolean, indicating whether intercept is included in the model.
`file.plot.pdf`	string, indicating a PDF file name to illustrate the performance of curve fitting.
`options`	list, including `max.optim.failure`, `min.optim.success`, `R2.loop`, and `verbose`, default values are `max.optim.failure=100`, `min.optim.success=20`, `R2.loop=5`, `verbose=F`.

Details

The phenotype file(`file.phe.long`), observed time file(`file.phe.time`) and covariate file(`file.phe.cov`) must be a CSV file. The following sections illustrate the format of each data file.

1) The phenotype file. The first column is individual ID and the rest columns are sample data for each measurement. It looks like the following file. Please note missing data is coded as space or NA in all data files. For example:

```
ID, 1st, 2nd, 3rd, 4th, 5th, 6th, 7th
1, 2.9033, 4.118, 6.1495, 7.8161, 9.8379, 12.963, 14.918
2, 4.3306, 5.378, 7.0647, 9.3624, 11.439, NA,     15.701
3, 2.3997, 4.052, 5.5431, 7.6933, 9.8471, NA      12.849
4, 3.3044, 4.154, 5.8924, 7.7133, 9.2144, 10.945, NA
...
```

2) The measurement time file. The first column is individual ID and the rest columns are observed times as the following format. For example:

```
ID, 1st, 2nd, 3rd, 4th, 5th, 6th, 7th
1, 1, 2, 3, 4, 5, 6,  7,
2, 1, 2, 3, 4, 5, NA, 7,
3, 1, 2, 3, 4, 5, NA, 7,
4, 1, 2, 3, 4, 5, 6,  NA,
...
```

3) The covariate file. The first column is individual ID and the rest columns are covariate values as the following format. For example:

```
ID, X1, X2
1, 1.0, 0
2, 1.1, 1
3, 2.1, 1
4, 3.1, 0
...
```

The function returns a S3 object which details can be checked by the command `print` or `str`.

The following conents are exported by `print` command.

```
== Phenotype Object in fGWAS ==
  Longitudal value : /tmp/Rtmp0lhMx2/file58b1776d8395.csv
   -- Individual count : 1678
  Longitudal time : /tmp/Rtmp0lhMx2/file58b1131b9b1.csv
   -- Time count : 8
  Covariate file : /tmp/Rtmp0lhMx2/file58b1246a670b.csv
   -- Covariate count : 6
   -- Intercept : YES
  -- Estimate values : 36917.16 -1.553526 -2.702489 7.852712 -5.91549 7.210022 3.805883
  Curve type : auto
   -- Estimate type : Legendre2
   -- Estimate values : -36888.65 1.081318 -1.852532
```

```
Covariate type : auto
 -- Estimated type : TOEPH
-- Estimate values : 0.9082402 0.8627414 0.8142826 0.7713539 0.7200614 0.6669603 ...
```

9 curves have been implemented in current version, including:

1) "Logistic"

$$g(t) = \frac{a}{1 + b * e^{-r*t}}$$

2) "Bi-Logistic"

$$g(t) = \frac{a1}{1 + b1 * e^{-r1*t}} + \frac{a2}{1 + b2 * e^{-r2*t}}$$

3) "Pharmacology"

$$g(t) = \frac{E_{max} * t}{Ec_{50} + t} + E_0$$

4) "Exponential"

$$g(t) = a * e^{-r*t}$$

5) "Bi-Exponential"

$$g(t) = a_1 * e^{-r_1*t} + a_2 * e^{-r_2*t}$$

6) "Power"

$$g(t) = a * t^b$$

7) "Legendre2", Legendre Polynomial(2nd-order)

$$g(t) = u_0 + u_1 * t + u_2 * (3 * t^2 - 1)/2$$

8) "Legendre3", Legendre Polynomial(3rd-order)

$$g(t) = u_0 + u_1 * t + u_2 * (2 * t^2 - 1)/2 + u_3 * (5 * t^3 - 3t)/2$$

9) "Legendre4", Legendre Polynomial(4th-order)

$$g(t) = u_0 + u_1 * t + u_2 * (2 * t^2 + 1)/2 + u_3 * (5 * t^3 - 3t)/2 + ...$$

Value

This function returns a S3 object with the class label of `fgwas.phe.obj`:

`pheY`	Matrix, the longitidual curve data with the rowname indicating the individuals' id.
`pheX`	Matrix, the covariate with the rowname indicating the individuals' id.
`pheT`	Matrix, the observed time points with the rowname indicating the individuals' id.
`ids`	Vector, the common individuals' id in all data file
`obj.curve`	Curve object, S4 object inherited from `fg.curve.base`
`obj.covar`	Covariance object, S4 object inherited from `fg.covariance.base`
`est.curve`	List, including the curve type and estimated parameters.
`est.covar`	List, including the covariance type and estimated parameters.

summary.curve	List, including the result of curve fitting.
summary.covar	List, including the result of covariance fitting.
params	List, the parameters of function calling, including the file names of longitudinal data and covariate, curve type and covariance type
options	List, not used currently

You can plot the curve by the function plot or print summary information by the function print.

Examples

```
# use simulation to make a phenotype object and genotype object and generate the files.
obj.sim <- fg.simulate( "Logistic", "SAD1", 2000, 800, 1:6,
    phe.missing=0.05, snp.missing=0.05,
    sig.pos=301, plink.format=TRUE, file.prefix = "temp.fwgas.test1" );

# show the brief information for the phenotype object
obj.sim$obj.phe

# load the phenotype traits generated by the simulation
obj.phe <- fg.load.phenotype("temp.fwgas.test1.pheY.csv", NULL, "temp.fwgas.test1.pheT.csv",
    curve.type = "Logistic", "SAD1", file.plot.pdf = NULL,
    intercept = FALSE,  options = list(verbose=TRUE))

# show the brief information, this object is equal to 'obj.sim$obj.phe'
obj.phe;

# plot the phenotype traits into a PDF file without the curving fitting information.
plot( obj.phe, curve.fitting=FALSE, file.pdf = "temp.fwgas.test1.phe.pdf");

# remove all test files
unlink("temp.fwgas.test1.*");
```

---

fg.load.plink *Loading PLINK data set.*

Description

Loading genotype data from PLINK data set.

Usage

```
fg.load.plink(file.plink.bed, file.plink.bim, file.plink.fam,
    plink.command = NULL,
    chr = NULL,
    options = list(verbose=F))
```

Arguments

file.plink.bed	string, the name of PLINK bed file, containing the packed binary SNP genotype data
file.plink.bim	string, the name of PLINK bim file, containing the SNP descriptions
file.plink.fam	string, the name of PLINK fam file, containing subject(and, possibly, family) identifiers
plink.command	string, indicating PLINK command path, for the large PLINK data, the package loads partial PLINK data extracted by the 'plink' command rather than the whole data set.
chr	vector of string, indicating the chromosome number involved to do hypothesis test.
options	list, including force.split and verbose

Details

This fucntion try to avoid loading all genotype data in memory. The optional parameter force.split indicates to use the plink.command to split the genotype data accordingto chromose unit and then load the partial gentotype data gradually.

The following example show the contents exported by print command.

```
== Genotype Object in fGWAS ==
  Plink bed : /home/userx/proj/gwas2/bmi-c1c2-qc2.bed
  Plink bim : /home/userx/proj/gwas2/bmi-c1c2-qc2.bim
  Plink fam : /home/userx/proj/gwas2/bmi-c1c2-qc2.fam
  Data file :
  SNP count : 431670
  Total individuals : 1678
  Reference matrix object of class "fg.dm.plink"
  Data type:
  Description:
  SNP Count:  431670
  Individual Count:  1678
  Individual Used:  1678
  Plink Command:  plink
  chromosome:  all
```

Value

This function returns a S3 object with the class label of fgwas.gen.obj, including:

reader	a reference class fg.dm.plink, the reader object for plink data, the structure is demonstrated in the details.
n.snp	integer value, indicating the total SNP number.
n.ind.total	integer value, indicating the total individual number.
n.ind.used	integer value, indicating the used individual number except the missing data.
params	list, including the PLINK file names and 'plink' command assigned to this function calling.
options	list, two options: force.split and verbose, default values are TRUE.

You can print summary information by the function print.

Examples

```
# use simulation to generate PLINK data files.
objx <- fg.simulate( "Logistic", "SAD1", 2000, 800, 1:6,
    phe.missing=0.05, snp.missing=0.05,
    sig.pos=301, plink.format=TRUE,
    file.prefix = "temp.fgwas" );

# load genotype data from PLINK data file and return a genotype object
obj.gen <- fg.load.plink("temp.fgwas.geno.bed", "temp.fgwas.geno.bim", "temp.fgwas.geno.fam" );

# show the brief information of genotype object
obj.gen;
```

---

fg.load.simple *Loading simple SNP file*

Description

Loading the SNP data from a text file in simple format.

Usage

```
fg.load.simple( file.simple.snp, options=list(verbose=F))
```

Arguments

file.simple.snp	string, file name of SNP data table, the format is described in the details.
options	list, only including the item verbose.

Details

This table file contains SNP information and individual SNPs. The SNP information, including SNP name, chromosome, position, reference allele and alternate allele are located from the 1st column to 5rd column. The individual SNPs follow the SNP information at each row, encoding genotype type as 0,1,2 or NA (for missing data). For example:

```
SNP  CHR POS RefAllele AltAllele Sub1 Sub2 Sub3 Sub4 Sub5 ...
SNP1 1 1  A T  2  0  1  1 1 ...
SNP2 1 2  A G  2  1  1  0 0 ...
...
```

Value

A S3 object (fgwas.gen.obj) is returned by this function, including:

options	List,
reader	A reference class , indicating the reader object for simple data format.

params	List, including the file names and other optional parameters.
n.snp	Numeric, indicating the total SNP number.
n.ind.total	Numeric, indicating the total individual number.
n.ind.used	Nuermic, indicating the used individual number except the missing data.

Examples

```
# make a simple SNP data table
snp <- matrix( round( runif(20000, 0, 2) ), nrow=100)
colnames(snp) <- paste("sub", 1:200, sep="");
snp.name <- paste("snp", 1:100, sep="");
# make a data frame including snp information and snp data table
snp.mat <- data.frame(snp.name, chr=1, pos=1:100, Allele1="A", Allele2="B", snp);
# write the data frame into a tab-seperated table file
write.table(snp.mat, file="temp.fgwas.snp.mat.csv",
    row.names=FALSE, col.names=TRUE, quote=FALSE, sep="\t");
# load the snp data from a simple snp data file.
obj.gen <- fg.load.simple("temp.fgwas.snp.mat.csv");

# simulate the phenotype and genotype data and generate a simple snp data file
objx <- fg.simulate( "Logistic", "SAD1", 2000, 800, 1:6, phe.missing=0.05, snp.missing=0.05,
    sig.pos=301, plink.format=FALSE, file.prefix = "test1.fgwas" );
# load the snp data generated from the simulation
obj.gen <- fg.load.simple("test1.fgwas.geno.tab");
# call SNP scaning
r1 <- fg.snpscan(obj.gen, objx$obj.phe, snp.sub=c(290:310), options=list(ncores=1));

# simulate the phenotype and genotype data and generate PLINK data file
objx <- fg.simulate( "Logistic", "SAD1", 2000, 800, 1:6, phe.missing=0.05, snp.missing=0.05,
    sig.pos=301, plink.format=TRUE, file.prefix = "test1.fgwas" );

# load PLINK data file using the function in the snpStats package
obj.plink <- read.plink( "test1.fgwas.geno.bed", "test1.fgwas.geno.bim", "test1.fgwas.geno.fam");
# convert the PLINK data object into SNP data table
x0 <- as.data.frame( obj.plink$map[,c(2,1,4,5,6)])
x1 <-  matrix( as.numeric( (t(obj.plink$genotypes ))), nrow=800)
x1[which(x1==0)]<-NA
x1 <- x1 - 1;
x1 <-  matrix( x1, nrow=800);
colnames(x1) <- paste("N", 1:2000, sep="_");
snp.mat <- data.frame(x0, x1) ;
# write snp data table into a tab-seperated table file.
write.table(snp.mat, file="temp.fgwas.snp.mat.csv",
    row.names=FALSE, col.names=TRUE, quote=FALSE, sep="\t");

# load the snp data converted from the PLINK files
obj.gen <- fg.load.simple("temp.fgwas.snp.mat.csv");
# call SNP scaning in a short range(290:310)
r2 <- fg.snpscan(obj.gen, objx$obj.phe, snp.sub=c(290:310), options=list(ncores=1));
```

fg.qqplot *Drawing QQ plot for the fGWAS results.*

Description

Drawing QQ plot for the fGWAS results.

Usage

```
fg.qqplot(object, png.file, title = "", width = 480)
```

Arguments

object	A S3 object of scaning result returned by fg.snpscan.
png.file	string indicating PNG file name.
title	string indicating the title in the figure.
width	integer indicating the width in pixel unit.

Value

No return values. PNG file is exported if this function calling is successful.

See Also

fg.snpscan.

Examples

```
# simulate phenotype object and genotype object
r <- fg.simulate("Logistic", "AR1", 2000, 500, 1:7, sig.pos=250 );
# SNP scaning
obj.scan <- fg.snpscan(r$obj.gen, r$obj.phe, method="fgwas");

fg.qqplot( obj.scan, "fgwas-pkg-test.png", title="Simulation");
```

fg.select.sigsnp *Selecting significant SNPs*

Description

Selecting significant SNPs

Usage

```
fg.select.sigsnp( fgwas.scan.obj, sig.level=0.05, pv.adjust="bonferroni", options=list() )
```

Arguments

fgwas.scan.obj	A S3 object of scaning result returned by fg.snpscan.
sig.level	Numeric value indicating the p-value of criteria.
pv.adjust	string indicating whether p-value is adjusted value for multiple Comparisons, the optional values are holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none" defined in p.adjust.
options	Not used

Value

The function returns a sub-matrix including the significant SNPs.

Examples

```
# simulate the phenotype object and genotype object.
r<-fg.simulate("Logistic", "AR1", 2000, 500, 1:7 );
# call SNP scaning using 'fast' method
obj.scan <- fg.snpscan(r$obj.gen, r$obj.phe, covariance.type="AR1", method="fast");
# select the significant SNPs from the result object
tb<-fg.select.sigsnp(obj.scan, sig.level=1e-10, "fdr")

# show 5 lines of significant SNP table
head(tb);
# show SNP indexes in significant SNP table
show(tb$INDEX);
# show SNP names in significant SNP table
show(tb$NAME);
```

---

fg.simulate *Data Simulation*

Description

Data simulation is for demonstration or power test in the fGWAS package.

Usage

```
fg.simulate(curve.type, covariance.type, n.obs, n.snp, time.points,
    par0 = NULL,
    par1 = NULL,
    par2 = NULL,
    par.covar = NULL,
    par.X = NULL,
    sig.pos = NULL,
    phe.missing = 0.03,
    snp.missing = 0.03,
    plink.format = FALSE,
    file.prefix = NULL )
```

Arguments

curve.type	String, indicating the curve type, such as "Logistic", "Legendre2", available options are described in the details of fg.load.phenotype.
covariance.type	String, indicating the covariance structure type, such as "SAD1", "AR1", available options are described in the details of fg.load.phenotype.
n.obs	Numeric, indicating the individual number.
n.snp	Numeric, indicating the SNP number.
time.points	Numeric, indicating the measure time points for the longitudinal traits.
par0	Numeric vector, indicating the curve parameters for the genotype AA.
par1	Numeric vector, indicating the curve parameters for the genotype Aa.
par2	Numeric vector, indicating the curve parameters for the genotype aa.
par.covar	Numeric vector, indicating the parameters of the covariance matrix.
par.X	Numeric vector, indicating the covariate coefficients.
sig.pos	Numeric, indicating the significant SNP position.
phe.missing	Numeric, the missing rate of phenotype data.
snp.missing	Numeric, the missing rate of genotype data.
plink.format	Logical variable, indicating whether the PLINK data files are generated.
file.prefix	String, the prefix file name for the simulation data.

Details

The object structures of two S3 object, phenotype data object(fg.load.phenotype) and genotype object(fg.load.plink)

The mathematics model can be illustrated as the following formulas:

```
Y = X %*% par.X + curve_function ( par0, T ) + e if snp is AA \cr
Y = X %*% par.X + curve_function ( par1, T ) + e if snp is Aa \cr
Y = X %*% par.X + curve_function ( par2, T ) + e if snp is aa \cr
```

In above formulas, X and snp are geneated randomly, curve_function is selected by the curve.type, the residuals e follow the multivarite normal distribution N(0, covar), covariance matrix is built by the parameter covariance.type and par.covar.

Value

The function return a list containing two objects, including

obj.gen	genotype data object, S3 object with class name 'fgas.gen.obj', the format is identical with return of fg.load.simple.
obj.phe	phenotype data object, S3 object with class name 'fgas.phe.obj', the format is identical with return of fg.load.phenotype.

If plink.format is set to TRUE, the plink data set generated by the simulation process will be stored in the file specified by obj.gen$files.

Examples

```
# simulate the Logistic traits and genotype data.
obj1 <- fg.simulate( "Logistic", "AR1", 2000, 1000, 1:8,
    phe.missing=0.05, snp.missing=0.05,
    sig.pos=501, plink.format=FALSE, file.prefix = NULL );

# show the genotype object
print(obj1$obj.gen);
# show the phenotype object
print(obj1$obj.phe);
# plot the phenotype object
plot(obj1$obj.phe, curve.fitting=FALSE);

# simulate the Logistic traits and genotype data.and save the data into PLINK files
obj2 <- fg.simulate( "Logistic", "SAD1", 2000, 800, 1:6,
    phe.missing=0.05, snp.missing=0.05, sig.pos=301,
    plink.format=TRUE, file.prefix = "temp.fgwas" );


# show the genotype object
print(obj2$obj.gen);
# show the phenotype object
print(obj2$obj.phe);
# plot the phenotype object
plot(obj2$obj.phe, curve.fitting=FALSE);

# remove the phenotype files and PLINK files
unlink(c("temp.fgwas.geno.bed", "temp.fgwas.geno.bed", "temp.fgwas.geno.bed",
    "temp.fgwas.pheT.csv", "temp.fgwas.pheY.csv"));
```

`fg.snpscan` *Detecting significant SNPs by fGWAS model*

Description

Scaning SNPs with the fGWAS model and detect the significant SNPs.

Usage

```
fg.snpscan( fgwas.gen.obj, fgwas.phe.obj,
    method = "optim-fgwas",
    curve.type = NULL,
    covariance.type = NULL,
    snp.sub = NULL,
    options = list(verbose=F) )
```

Arguments

`fgwas.gen.obj`	S3 object `fgwas.gen.obj`, indicating the genotype object generated by the `fg.load.plink`, `fg.load.simple` or `fg.simulate`.

fgwas.phe.obj	S3 object fgwas.phe.obj, indicating the phenotype object generated by the fg.load.phenotype or fg.simulate.
method	string, indicating the statistical model used in the scaning process, available options are "gls", "mixed", "fast", "fgwas", "optim-fgwas". Default is 'optim-fgwas'.
curve.type	string, indicating the curve type used in the scaning process for the method "fast", "fgwas" and "optim-fast", instead of the original curve type in phenotype object. The full optional values are listed in fg.load.phenotype
covariance.type	string, indicating the type of covariance structure used in the scaning process for the method "fast", "fgwas" and "optim-fast", instead of the curve type in phenotype object. The full optional values are listed in fg.load.phenotype
snp.sub	vector of SNP index or SNP name, indicating the selected SNPs will be scanned, not the whole data set.
options	list, including verbose, ncores, max.optim.failure, min.optim.success, use.gradient, degree, default values are verbose=F, ncores=1, use.snowfall=TRUE, max.optim.failure=20 min.optim.success=2 use.gradient=FALSE degree=3.

Details

The parallel computing is available using the optional ncores item. **fGWAS** uses the **snowfall** for all methods. In Linux or Linux compatible OS, **parallel** can be used to the method "fast", "fgwas", and "optim-fgwas" to gain more faster parallel speed. If **parallel** is perferred, set use.snowfall=FALSE.

Due to the difficulty to estimate and provide initial parameters for the function optim, **fGWAS** sets the maximum optim failure time and minimum optim success time in the optional items.

For the method "fgwas" and "optim-fgwas", **fGWAS** performs the MLE using the numerical derivation by default use.gradient=FALSE, If use.gradient=TRUE, the gradient function is provided and used in the MLE.

For the method mixed, the default polynomial degree is 3, it can be customized with the optional item degree.

Value

A S3 object with the class name (fgwas.scan.obj) is returned by this function, including:

obj.gen	S3 object fgwas.gen.obj processed in this function
obj.phe	S3 object fgwas.phe.obj processed in this function
ret.gls	Matrix, results for each SNP scanned by the 'gls' method.
ret.mixed	Matrix, results for each SNP scanned by the 'mixed' method.
ret.fast	Matrix, results for each SNP scanned by the 'fast' method.
ret.fgwas	Matrix, results for each SNP scanned by the 'fgwas' or 'fgwas-optim' method.

If 'gls' is used to scan the SNPs, the result is stored in ret.gls with the following items:

INDEX	SNP index.

NAME	SNP name.
CHR	Chromosome.
POS	Base position.
NMISS	Number of missing.
MAF	Minor allel frequence.
LR	Likelihood ratio.
pv	p-value of Fisher.

If 'mixed' is used to scan the SNPs, the result is stored in `ret.mixed` with the following items:

INDEX	SNP index.
NAME	SNP name.
CHR	Chromosome.
POS	Base position.
Allel1	Allel 1.
Allel2	Allel 2.
MAF	Minor allel frequence.
NMISS	Number of missing.
pv	p-value of Fisher .
p.min	p-value of minimum values.
p.join	p-value of joint value.
p0	p-value of 0th-degree Legendre polynomial.
p1	p-value of 1st-degree Legendre polynomial.
p2	p-value of 2nd-degree Legendre polynomial.
p3	p-value of 3rd-degree Legendre polynomial.
p4	p-value of 4th-degree Legendre polynomial.

If 'fgwas', 'fgwas-optim' or 'fast' are used to scan the SNPs, the result is stored in `ret.fast` or `ret.fgwas` with the following items:

INDEX	SNP index.
NAME	SNP name.
CHR	Chromosome.
POS	Base position.
MAF	Minor allel frequence.
NMISS	Number of missing.
SNP0	Individual count of genotype 0.
SNP1	Individual count of genotype 1.
SNP2	Individual count of genotype 2.
GENO	Genotype count.
LR	Likelihood Ratio.

pv	p-value based on LR.
h0.X0	the 1st covariate coefficient under NULL hypothesis.
...	the other covariate coefficients under NULL hypothesis.
h0.a	the parameter 'a' in curve function(e.g. Logistic curve) under NULL hypothesis.
h0.b	the parameter 'b' in curve function(e.g. Logistic curve) under NULL hypothesis.
...	the other curve parameters in the curve function under NULL hypothesis.
h0X.rho	the parameter 'rho' in covariance structure (e.g. AR1) under NULL hypothesis.
...	the other parameters in covariance structure (e.g. AR1) under NULL hypothesis.
h1.X0	the 1st covariate coefficient under alternative hypothesis.
...	the other covariate coefficients under alternative hypothesis.
h1.G0.a	the parameter 'a' for genotype 0 in curve function (e.g. Logistic curve) under alternative hypothesis.
h1.G0.b	the parameter 'b' for genotype 0 in curve function (e.g. Logistic curve) under alternative hypothesis.
...	the other parameters for genotype 0 in curve function (e.g. logistic curve) under alternative hypothesis.
h1.G1.a	the parameter 'a' for genotype 1 in curve function (e.g. Logistic curve) under alternative hypothesis.
h1.G1.b	the parameter 'b' for genotype 1 in curve function (e.g. Logistic curve) under alternative hypothesis.
...	the other parameters for genotype 1 in curve function (e.g. logistic curve) under alternative hypothesis.
h1.G2.a	the parameter 'a' for genotype 2 in curve function (e.g. Logistic curve) under alternative hypothesis.
h1.G2.b	the parameter 'b' for genotype 2 in curve function (e.g. Logistic curve) under alternative hypothesis.
...	the other parameters for genotype 2 in curve function (e.g. logistic curve) under alternative hypothesis.
h1X.rho	the parameter 'rho' in covariance structure (e.g. AR1) under alternative hypothesis.
...	the other parameters in covariance structure (e.g. AR1) under alternative hypothesis.
h0.R2	R2 under NULL hypothesis.
h1.R2	R2 under alternative hypothesis.

See Also

fg.load.phenotype, fg.load.plink, fg.load.simple, fg.select.sigsnp

Examples

```
# simulate the phenotype object and genotype object
r<-fg.simulate("Logistic", "AR1", 2000, 500, 1:7, sig.pos=250 );
# SNP scaning in a short range (245:255) using 'fast' method
obj1.scan <- fg.snpscan(r$obj.gen, r$obj.phe, method="fast", snp.sub=c(245:255) );
# show the summary information of result object
obj1.scan;

# SNP scaning in a short range (245:255) using 'fgwas' method
obj2.scan <- fg.snpscan(r$obj.gen, r$obj.phe, method="fgwas", snp.sub=c(245:255) );
obj2.scan;

# check the full result table.
tb.full <- obj2.scan$ob.fgwas$results;

# plot Manhattan figure
plot(obj2.scan, file.pdf="temp.fwgas.obj2.scan.pdf");

# select significant SNPs
tb.sig <- fg.select.sigsnp(obj2.scan, sig.level=0.001,  pv.adjust = "bonferroni")
# plot the genetic curve effect for the significant SNPs
plot.fgwas.curve( obj2.scan, tb.sig$INDEX, file.pdf="temp.fwgas.obj2.curve.pdf");
```

`plot.fgwas.curve` *Plotting the genetic curve effects associated with SNP.*

Description

Plot the genetic curve effects based on the parameter estimation in the fGWAS model for each genotype.

Usage

```
plot.fgwas.curve( object, snp.sub, file.pdf=NULL, draw.rawdata=TRUE, draw.meanvector=TRUE, ... )
```

Arguments

`object`	Result object returned by the `fg.snpscan`.
`snp.sub`	Vector of SNP index or SNP name.
`file.pdf`	String indicating the PDF file name.
`draw.rawdata`	Logical value indicating whether the raw phenotype curves are drawn.
`draw.meanvector`	Logical value indicating whether the mean vectors for each genotype are drawn.
`...`	additional arguments affecting the plot, including `xlab`, `ylab`, `xlim`, `ylim`, `title`.

Details

This function will apply to the result obtained from 'fgwas', 'fgwas-optim' or 'fast' methond.

Value

No return values.

Examples

```
# simulate phenotype object and genotype object
r <- fg.simulate("Logistic", "AR1", 2000, 500, 1:7, sig.pos=250 );
# SNP scaning
obj.scan <- fg.snpscan(r$obj.gen, r$obj.phe, method="fast", snp.sub=c(245:255) );
# select significant SNPs by default.
tb.sig <- fg.select.sigsnp(obj.scan)
# plot the genteic curves for each significant SNP.
plot.fgwas.curve( obj.scan, tb.sig$INDEX, file.pdf="test.plot.pdf");
# remove the PDF file
unlink("test.plot.pdf")
```

---

plot.fgwas.phe.obj *Ploting the longitudinal curves*

Description

Ploting the longitudinal curves.

Usage

```
 ## S3 method for class 'fgwas.phe.obj'
plot(x, y, ..., curve.fitting = T, file.pdf = NULL)
```

Arguments

x	a phenotype object return by fg.load.phenotype or fg.simulate
y	required parameter in the generic plot function, not used.
...	additional arguments affecting the summary produced.
curve.fitting	boolean value indicating whether the fitted curve is plot.
file.pdf	string indicating the output pdf file name.

Details

The phenotype object is described in fg.load.phenotype.

Value

No return values, only figure is exported to PDF file.

Examples

```
# data simulation
r<-fg.simulate( "Logistic", "SAD1", 2000, 1000, c(2,4,6,8,10) );
# plot phenotype traits
plot(r$obj.phe, file.pdf="temp.fg.test.pdf");
# remove the PDF file
unlink("temp.fg.test.pdf");
```

`plot.fgwas.scan.obj` *Drawing Manhattan plot*

Description

Drawing p-values in Manhattan plot.

Usage

```
 ## S3 method for class 'fgwas.scan.obj'
plot(x, y, ..., file.pdf = NULL, sig.level=0.05)
```

Arguments

`x`	a result object return by `fg.snpscan`
`y`	required parameter in the generic plot function, but unused in this function.
`...`	additional arguments affecting the plot.
`file.pdf`	string indicating output PDF file name.
`sig.level`	numeric value indicating the location of threshold line on the Manhattan figure.

Details

N/A

Value

No return values, only figure is exported into PDF file.

Examples

```
# data simulation
r<-fg.simulate("Logistic", "AR1", 2000, 500, 1:7 );
# SNP scaning using 'fast' method
obj.scan <- fg.snpscan(r$obj.gen, r$obj.phe, method="fast");
# plot the Manhattan figure
plot(obj.scan, file.pdf="temp.fgwas.plot.pdf");
# remove the PDF file
unlink("temp.fgwas.plot.pdf");
```

`print.fgwas.gen.obj` *Printing brief information for the genotype object*

Description

Printing brief information for the genotype object.

Usage

```
 ## S3 method for class 'fgwas.gen.obj'
print( x, ..., useS4 = FALSE )
```

Arguments

`x`	a genotype object returned by `fg.load.plink` or `fg.load.simple`
`...`	additional arguments affecting the summary produced.
`useS4`	an argument used to match showDefault function. Fixed as FALSE.

Details

The genotype object is described in `fg.load.plink` or `fg.load.simple`.

The following example demonstrates the output of `print` command for a genotype object.

```
== Genotype Object in fGWAS ==
  Plink bed : /demo.bed
  Plink bim : /demo.bim
  Plink fam : /demo.fam
  Data file :
  SNP count : 431670
  Total individuals : 1678
  Reference matrix object of class "fg.dm.plink"
  Data type:
  Description:
  Chromosome:  all
  SNP Count:  431670
  Individual Count:  1678
  Individual Used:  1678
  Plink Command:  plink
```

Value

No return values, only output the brief information on the R console.

Examples

```
# data simulation
r<-fg.simulate("Logistic", "AR1", 2000, 500, 1:7 );
# call this function to print the genotype information
r$obj.gen;
# summarize the genotype object
summary(r$obj.gen);
```

`print.fgwas.phe.obj` *Printing brief information for the phenotype object*

Description

Printing brief information for the phenotype object.

Usage

```
 ## S3 method for class 'fgwas.phe.obj'
print( x, ..., useS4=FALSE )
```

Arguments

x	a phenotype data object returned by `fg.load.phenotype` or `obj.phe` in the `fg.simulate`
...	additional arguments affecting the summary produced.
useS4	an argument used to match showDefault function. Fixed as FALSE.

Details

The phenotype object is described in `fg.load.phenotype`.

The following example demonstrates the output of `print` command for a phenotype object.

```
== Phenotype Object in fGWAS ==
  Longitudal value : /tmp/Rtmp0lhMx2/file58b1776d8395.csv
   -- Individual count : 1678
  Longitudal time : /tmp/Rtmp0lhMx2/file58b1131b9b1.csv
   -- Time count : 8
  Covariate file : /tmp/Rtmp0lhMx2/file58b1246a670b.csv
   -- Covariate count : 6
   -- Intercept : YES
  -- Estimate values : 36917.16 -1.553526 -2.702489 7.852712 -5.91549 7.210022 3.805883
  Curve type : auto
   -- Estimate type : Legendre2
   -- Estimate values : -36888.65 1.081318 -1.852532
  Covariate type : auto
   -- Estimated type : TOEPH
  -- Estimate values : 0.9082402 0.8627414 0.8142826 0.7713539 0.7200614 0.6669603 ...
```

Value

No return values, only output the brief information on the R console.

Examples

```
# data simulation
r<-fg.simulate("Logistic", "AR1", 2000, 500, 1:7 );
# print brief phenotype information
r$obj.phe;
```

`print.fgwas.scan.obj` *Printing the significant SNPs or top SNPs.*

Description

Printing the significant SNPs or top SNPs.

Usage

```
 ## S3 method for class 'fgwas.scan.obj'
print( x,  ..., useS4 = FALSE )
```

Arguments

x	a result object return by `fg.snpscan`
...	additional arguments affecting the summary produced.
useS4	an argument used to match showDefault function. Fixed as FALSE.

Details

The result object is described in `fg.snpscan`.

The following example demonstrates the output of `print` command for a result object.

```
== Result from 'fGWAS' method ==
 SNP =  431670
            INDEX       NAME CHR       POS         MAF NMISS           pv
rs11081728 386746 rs11081728  18  29511663 0.169253731     3 5.413541e-15
rs11051162 301826 rs11051162  12  31074982 0.034090909     6 3.997973e-14
rs7876945  423711  rs7876945  23  13134844 0.292572464    22 5.341771e-14
rs12652390 146007 rs12652390   5 126322102 0.014379868     9 2.143879e-13
rs6029168  407076  rs6029168  20  39158207 0.324880668     2 2.182798e-13
rs7876326  424440  rs7876326  23  23626268 0.470481928    18 3.074301e-13
...
```

Value

No return values, only output the significant SNPs or top SNPs on the R console.

Examples

```
# data simulation
r <- fg.simulate("Logistic", "AR1", 2000, 500, 1:7 );
# SNP scaning
obj.scan <- fg.snpscan(r$obj.gen, r$obj.phe, method="fast");
# print the top or significant SNPs.
obj.scan;
```

---

summary.fgwas.gen.obj *Summarizing genotype object*

Description

Printing the summary information of the genotype object.

Usage

```
 ## S3 method for class 'fgwas.gen.obj'
summary( object, ... )
```

Arguments

object	a data object returned by fg.load.plink or fg.load.simple
...	additional arguments affecting the summary produced.

Details

The genotype object is described in fg.load.plink or fg.load.simple.

Value

A numeric vector including snp count, individual count and used individual count.

Examples

```
# data simulation
r<-fg.simulate("Logistic", "AR1", 2000, 500, 1:7 );
# print the brief information of gthe genotype object
r$obj.gen;
# output the sumamry information
summary(r$obj.gen);
```

summary.fgwas.phe.obj *Summarizing phenotype object.*

Description

Summarizing phenotype object.

Usage

```
 ## S3 method for class 'fgwas.phe.obj'
summary( object, ... )
```

Arguments

object	a phenotype object returned by fg.load.phenotype or obj.phe in the fg.simulate
...	additional arguments affecting the summary produced.

Details

None

Value

This function returns an extend table containing the covariate data , the longtudinal curve data and the measured times.

Examples

```
# data simulation
r<-fg.simulate("Logistic", "AR1", 2000, 500, 1:7 );

# print the brief information of the phenotype object
r$obj.phe;

# output the summary information
ext_phe <- summary(r$obj.phe);

# show the summary table
head(ext_phe);
```

summary.fgwas.scan.obj *Summarizing the result object*

Description

Summarizing the result object.

Usage

```
 ## S3 method for class 'fgwas.scan.obj'
summary( object, ... )
```

Arguments

object	a result object return by fg.snpscan
...	additional arguments affecting the summary produced.

Details

Below lists all columns for the analysis with curve 'Legendre2' and covariance 'AR1':

```
#obj.fgwas is result object of 'fgwas' method.
> colnames(summary(obj.fgwas))
 [1] "INDEX"      "NAME"       "CHR"        "POS"        "MAF"
 [6] "NMISS"      "SNP0"       "SNP1"       "SNP2"       "GENO"
[11] "LR"         "pv"         "L0"         "h0.X0"      "h0.X1"
[16] "h0.X2"      "h0.X3"      "h0.X4"      "h0.X5"      "h0.X6"
[21] "h0.u0"      "h0.u1"      "h0.u2"      "h0X.rho"    "h0X.sigma2"
[26] "L1"         "h1.X0"      "h1.X1"      "h1.X2"      "h1.X3"
[31] "h1.X4"      "h1.X5"      "h1.X6"      "h1.G0.u0"   "h1.G0.u1"
[36] "h1.G0.u2"   "h1.G1.u0"   "h1.G1.u1"   "h1.G1.u2"   "h1.G2.u0"
[41] "h1.G2.u1"   "h1.G2.u2"   "h1X.rho"    "h1X.sigma2" "h0.R2"
[46] "h1.R2"
```

Value

This function returns a data frame including the results of scanned SNPs and excluding covariate and curve object.

Examples

```
# data simulation
r<-fg.simulate("Logistic", "AR1", 2000, 500, 1:7, sig.pos=190 );

# SNP scaning
obj.scan <- fg.snpscan(r$obj.gen, r$obj.phe, covariance.type="AR1", method="fast");
```

```
# summarize the result object
tb <- summary(obj.scan);

# show the summary table
head(tb);
```

Index